UNIX® System Administration

Frank G. Fiamingo

fgf+@osu.edu

University Technology Services
The Ohio State University

September 23, 1998

This publication is available via the Internet as:

ftp://wks.uts.ohio-state.edu/sysadm_course/sysadm_book.ps

and

http://wks.uts.ohio-state.edu/sysadm_course/sysadm.html.

Also available via the Internet is ***Introduction to Unix***:

ftp://wks.uts.ohio-state.edu/unix_course/unix_book.ps

and

http://wks.uts.ohio-state.edu/unix_course/unix.html.

Acknowledgements: The author wishes to thank the following for helpful advice and discussions related to the material presented in this document: Harpal Chohan, Bob DeBula, Bob Manson, Steve Romig, and Bill Yang.

Table of Contents

PART I Introduction

CHAPTER 1 Overview

1.1 What is UNIX System Administration?

Systems administration is the installation and maintenance of the UNIX computer system. The system administrator will need to maintain the software and hardware for the system. This includes hardware configuration, software installation, reconfiguration of the kernel, networking, and anything else that's required to make the system work and keep it running in a satisfactory manner. To do this the system administrator can assume superuser, or root, privileges to perform many tasks not normally available to the average user of the system.

1.2 Daily Tasks of a System Administrator

1.2.1 Manage user logins

You add accounts by assigning login id's, groups, user id numbers, group id numbers, login directories, and set-up the users' login environments. You also need to balance the needs of various users, e.g. with quotas on disk space or limits on simultaneous processes.

1.2.2 Monitor system activity and security

You need to monitor disk status, system processes, user process activity, system security, and system log files to make sure that your resources are available and that only valid users have access to them.

1.2.3 Administer file systems, devices, and network services

You need to manage disk space usage, tape and CDROM devices and network services to make sure that these resources are available.

1.3 Startup and Shutdown

Startup is when you boot the system from the PROM. This can be from cdrom, disk, or over the network (ethernet). The shutdown programs, ***shutdown/reboot/halt***, allow you to close down the system in an orderly fashion.

1.4 Periodic Processes

Cron is the clock daemon. It executes periodic processes at pre-arranged times. You can use this to clean up old files, manage log files, backup the system to tape nightly, etc.

1.5 Managing File Systems

1.5.1 File System Backups

Backup and **restore** procedures are need to insure data integrity against disk crashes, users accidently deleting files, for the removal of seldom used programs to free up disk space, etc. You can usually automate this task.

1.5.2 Disk space quotas

Quotas restrict users to a finite disk space and can be set individually. This insures that individual users don't hog the available disk space.

1.6 Responsibilities to the users

You have the responsibility to provide access to disk space, CPU cycles, data integrity, operating system software updates, install necessary software, mail and network access, system security.

1.7 Hardware responsibilities

You are responsible for keeping the system running and maintaining it, adding new hardware, and making sure that everything is working properly.

1.8 Types of SunOS Systems

Standalone - system can function alone, independently of other systems.

Server - a standalone machine that can serve others, e.g. with disk space via NFS; can boot diskless workstations; can serve different architectures.

Dataless - has minimal disk space for systems programs and swap space only, shares file space via NFS mount of server disk space.

Diskless - has no disk; requires server for boot (via network), swap, and all program and file space.

AutoClient - similar to a diskless client except that it uses a local disk for caching. Requires a 100 MB local disk.

1.9 Resources for System Administrators

1.9.1 Network Resources

Usenet newsgroups/Mailing lists - via Internet through SONNET (the Ohio State University network).

WWW pages, you can start at the Workstation Groups home page: http://www-wks.acs.ohio-state.edu.

SunWorld Online (formerly ***Advanced Systems*** formerly ***SunWorld***) - now available via the World Wide Web at http://www.sun.com/sunworldonline/index.html.

1.9.2 Periodicals

Information Week - weekly publication for high-end business and technology users, Information Week, CMP Publications, Inc., 600 Community Drive, Manhasset, NY 11030.

SunExpert - monthly publication for Sun users, Computer Publishing Group, 1330 Beacon St. Brookline, MA 02146-3202.

UnixWorld - monthly publication, McGraw Hill, Inc., 1900 O'Farrell Street, San Mateo, CA 94403-1311.

/AIXtra - bimonthly publication for AIX users, IBM Corp., Mail Stop 40-B3-04, One East Kirkwood Blvd., Roanoke, TX 76299-0015.

RS/Magazine - monthly publication for RS/6000 users, Computer Publishing Group, 1330 Beacon St. Brookline, MA 02146-3202.

DECProfessional - monthly publication for DEC users, Cardinal Business Media, Inc., 101 Witmer Rd., Horsham, PA 17601.

SysAdmin - monthly publication, 1601 W. 23rd St., Suite 200, Lawrence, KS 66046-9950 (913-841-1631).

1.9.3 Books

1.9.3.1 Unix and the Internet

A Student's Guide to Unix, Harley Hahn (McGraw Hill, 1993, ISBN 0-07-025511-3).

UNIX in a Nutshell for BSD 4.3, A Desktop Quick Reference (O'Reilly & Associates, Inc. 1990, 0-937175-20-x).

UNIX in a Nutshell, A Desktop Quick Reference for System V & Solaris 2.0, Dan Gilly and the staff of O'Reilly & Associates, Inc. (O'Reilly & Associates, Inc. 1992, ISBN 1-56592-001-5)

The C Programming Language, 2nd Ed., Brian Kernighan and Dennis Ritchie (Prentice Hall, 1988, ISBN 0-13-110362-8).

Unix Shell Programming, Stephen Kochan and Patrick Wood (Hayden, 1990 ISBN 0-672-48448-X).

Programming Perl, Larry Wall and Randal L. Schwartz (O'Reilly & Associates, 1991, ISBN 0-937175-64-1).

The Whole Internet - User's Guide & Catalog, 2nd Ed., Ed Krol (O'Reilly, 1994, ISBN 1-56592-063-5).

Zen and the Art of the Internet, 3rd Ed., Brendan Kehoe (1994, ISBN 013-121492-6).

UNIX Power Tools, Jerry Peek, Tim O'Reilly, and Mike Loukides (O'Reilly & Associates, 1993, ISBN 0-679-79073-X). (Includes a CDROM of useful software for various OSs.)

1.9.3.2 System Administration

UNIX System Administration Handbook, 2nd Ed., Evi Nemeth, Garth Snyder, Scott Seabass and Trent Hein (Prentice-Hall, 1995, ISBN 0-13-151051-722). (Includes a CD-ROM)

Essential System Administration, ***2nd Ed.***, Aeleen Frisch (O'Reilly, 1995, ISBN 1-56592-127-5).

When You Can't Find Your UNIX System Administrator, Linda Mui (O'Reilly & Associates, Inc., 1995, ISBN 1-56592-104-6).

Solaris System Administrator's Guide, Janice Winsor (Ziff-Davis, 1993, ISBN 1-56276-080-7).

Solaris Advanced System Administrator's Guide, Janice Winsor (Ziff-Davis, 1993, ISBN 1-56276-131-5).

System Performance Tuning, Mike Loukides (O'Reilly & Associates, 1991, ISBN 0-937175-60-9).

Sun Performance and Tuning, Adrian Cockroft (Prentice Hall, 1995, ISBN 0-13-149642-5).

Unix System V Release 4 Administration, 2nd Ed., David Fiedler, Bruce Hunter, and Ben Smith (Hayden, 1991, ISBN 0-672-22810-6).

Managing NFS and NIS, Hal Stern (O'Reilly & Associates, 1991, ISBN 0-937175-75-7).

All About Administering NIS+, Rick Ramsey (SunSoft Press/Prentice Hall, 1992, ISBN 013-068800-2)

DNS and BIND, Paul Albitz and Cricket Liu (O'Reilly & Associates, 1993, ISBN 1-56592-010-4).

TCP/IP Network Administration, Craig Hunt (O'Reilly & Associates, 1992, ISBN 0-937175-82-X).

sendmail, Bryan Costales with Eric Allman and Neil Rickert (O'Reilly & Associates, 1994, ISBN 1-56592-056-2).

Panic! UNIX System Crash Dump Analysis, Chris Drake and Kimberley Brown (SunSoft Press, 1995, ISBN 0-13-149386-8). (Includes a CD-ROM).

1.9.3.3 Security

UNIX System Security, Patrick Wood and Stephen G. Kochan (Hayden Books, 1985, ISBN 0-8104-6267).

Practical UNIX & Internet Security, ***2nd Ed.***, Simon Garfinkel and Gene Spafford (O'Reilly & Associates, 1996, ISBN 1-56592-148-8).

Firewalls and Internet Security, W. R. Cheswick and S. M. Bellovin (Addison-Wesley, 1994).

Building Internet Firewalls, D. Brent Chapman and Elizabeth D. Zwicky (O'Reilly & Associates, Inc. 1995 ISBN 1-56592-124-0).

Improving the Security of Your UNIX System, David A. Curry (SRI International), available via anonymous ftp from www-wks.acs.ohio-state.edu:/pub/security/security-doc.tar.

1.10 UTS Software Support

University Technology Services UNIX Workstation Support - Software support for **SunOS/Solari**s (Sun), **Ultrix** and **Digital UNIX** (formerly OSF/1) (DEC), and **IRIX** (SGI).

1.10.1 Solaris

The Sun operating system, SunOS, along with the OpenWindows graphical user interface (GUI), make up the complete Sun UNIX environment. The latest release is Solaris 2.6, which includes SunOS 5.6, OpenWindows 3.6, and version 1.2 of the Common Desktop Environment (CDE). SunOS 5 is based on the System V Revision 4 version of UNIX. Solaris 2.4 runs on all SPARC hardware. Solaris 2.4 runs on all SPARC hardware except the Sun4 series (i.e. Sun 4/110, 4/280, etc.). Solaris 2.4 is still available for those who need it.

The latest release of the BSD version of UNIX for the SPARC architecture is Solaris 1.1.2, which includes SunOS 4.1.4 and OW 3_414. Solaris 1.1.2 runs on all SPARC hardware except the Sun4u series (UltraSPARCs).

Sun software is site licensed for all Ohio State University faculty, staff, and students, and can be borrowed from **UTS Customer Services**, 512 Baker Systems. All software is on CDROM.

1.10.2 IRIX

IRIX 5.3 is supported on all R3000 and R4000 hardware. IRIX 6.5 is supported on the R4000 and later hardware.

Software licenses must be purchased at the University Bookstore. The Bookstore receipt, along with the serial number(s) of the machine(s), must be presented before the software can be loaned to you.

SGI software is site licensed for all Ohio State University faculty, staff, and students, and can be borrowed from **UTS Customer Services**, 512 Baker Systems. All software is on CDROM.

CHAPTER 2

Disk Structure and Partitions

Modern disk drives include a CPU and memory to control the disk operation. The drive can accept many simultaneous requests, sort them, and process them concurrently. This minimizes the amount of head movement required to find all the requested data. It stores the data for all these commands in its own memory and can pre-fetch data that it expects you to ask for next, when it's not too busy with current requests.

2.1 Disk Structure and Partitions

2.1.1 Disk Structure

A hard disk is physically composed of a series of flat, magnetically coated platters stacked on a spindle. The spindle turns while the heads move between the platters, in tandem, radially reading/writing data onto the platters.

FIGURE 2.1 **Physical Disk Structure**

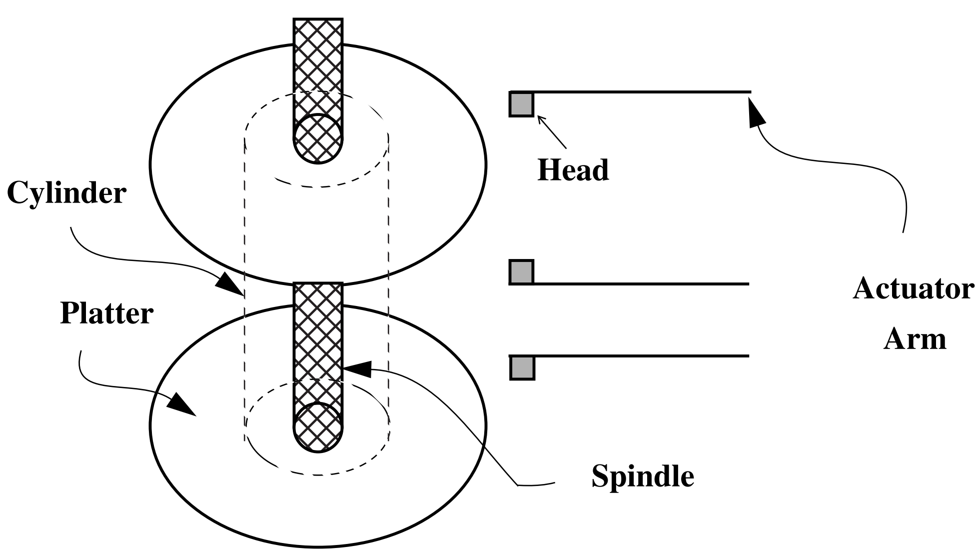
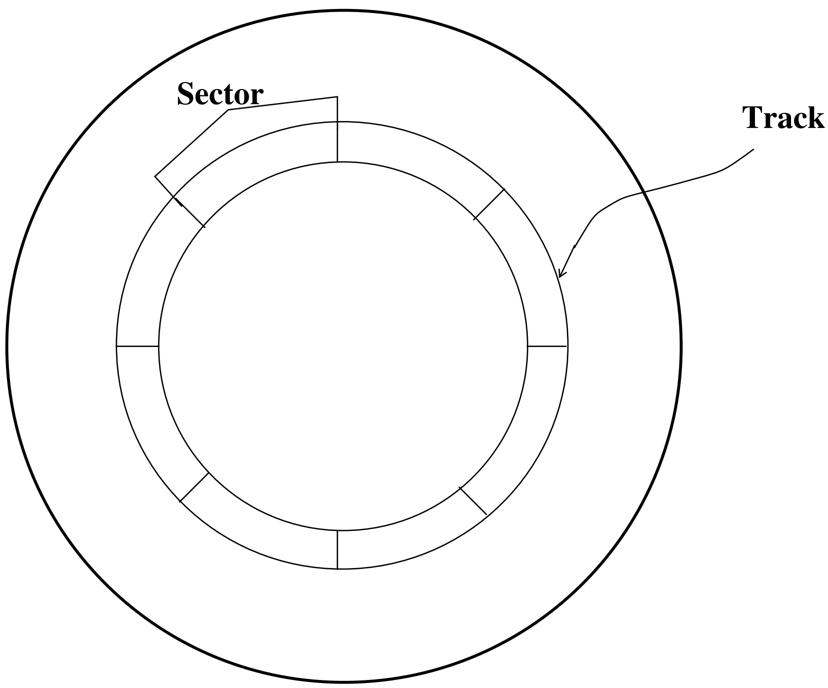


FIGURE 2.2 **Disk Platter**



2.1.2 Disk tracks, cylinders, and sectors

A disk is divided into **tracks**, **cylinders**, and **sectors**. A **track** is that portion of a disk which passes under a single stationary head during a disk rotation, a ring 1 bit wide. A **cylinder** is comprised of the set of tracks described by all the heads (on separate platters) at a single seek position. Each cylinder is equidistant from the center of the disk. A track is divided into segments of **sectors**, which is the basic unit of storage.

On Sun systems a **sector** is 512 bytes (1 disk block) of data, with header and trailer information. The latter make it possible for the controller to identify sectors, detect data errors, and perform error corrections when necessary. The actual layout of a disk sector will vary depending on the controller, but should look something like that shown in Fig. 2.3. There are two Preambles and a Postamble (whose sizes may vary due to rotational speed, etc., and are disk dependent). The Header field lets the controller know where the head is positioned, and the ECC field is for error correction.

FIGURE 2.3 **Sector**

Preamble 1	Header	Sync	Preamble 2	Sync	Data Field	ECC	Postamble
25 bytes	8 bytes	1 byte	25 bytes	1 byte	512 bytes	6 bytes	22 bytes

The number of sectors per track varies with the radius of the track on the platter. The outermost tracks is larger and can hold more sectors than the inner ones. These outer tracks also spin faster under the head than do the inner ones, because while the angular speed remains the same, the larger circumference results in more sectors spinning by in the same period for the outer tracks. Disk blocks are numbered starting at the outermost track, so put the data you expect to access most often on partition, or slice, 0.

2.1.3 Cylinder group

SunOS uses the Berkeley fast file system which uses **cylinder groups**. A group is formed form 32 or fewer cylinders on a disk (default 16). Each cylinder group has a redundant copy of the superblock, space for inodes, list of available blocks, and a list of data block usage within the cylinder group. Data blocks are spaced to minimize rotational delays and to keep blocks of the same file close together. By grouping cylinders in this way we reduce the amount of head movement, on average, required to access a file. The inode describing the file, and the data for the file, are likely to be in the same physical area of the disk. The position of the redundant superblock within each cylinder group is varied, so that they don't all reside on the same disk platter. This helps to insure that you can recover in the event of the loss of the primary superblock.

2.2 Disk Partitions

2.2.1 SunOS 4.1.X

The BSD and SunOS 4.1.X operating systems divide a disk into 8 **partitions**: a → h, some of which may be zero. Partition **c** covers the entire disk. On the root disk partition **a** is for the boot files and root directory, **b** is for swap space - virtual memory space for process and information that can't be contained in main memory, and **c** is the entire disk. Disk space is allocated in terms of **cylinders**, **tracks**, and **sectors/blocks**.

An example of the **partition table** on a SunOS 4.1.X disk might be:

```
# format sd0
format> partition
partition> print                                                    Corresponding
Current partition table (original sd0):                              File System
 partition a - starting cyl    0, # blocks    33120 (46/0/0)        / - root
 partition b - starting cyl   46, # blocks   125280 (174/0/0)       swap
 partition c - starting cyl    0, # blocks   828720 (1151/0/0)      entire disk
 partition d - starting cyl  220, # blocks    59760 (83/0/0)        /var
 partition e - starting cyl    0, # blocks        0 (0/0/0)
 partition f - starting cyl    0, # blocks        0 (0/0/0)
 partition g - starting cyl  303, # blocks   610560 (848/0/0)       /usr
 partition h - starting cyl    0, # blocks        0 (0/0/0)
```


2.2.2 SunOS 5.X

SunOS 5.X defines the partitions as numbers, rather than names. Also, instead of calling it a **partition** it's now called a **slice**. The root slice is then **0**, slice **1** is swap and slice **2** covers the entire disk.

On a SunOS 5.X the disk might be formatted something like this:

```
# format sd0
format> partition
partition> print
Volume:  nyssa
Current partition table (original):
Total disk cylinders available: 1866 + 2 (reserved cylinders)
```

Part	Tag	Flag	Cylinders	Size		Blocks	*Corresponding File System*
0	root	wm	0 - 73	20.23MB	(74/0/0)	41440	*/ - root*
1	swap	wu	74 - 293	60.16MB	(220/0/0)	123200	*swap*
2	backup	wm	0 - 1865	510.23MB	(1866/0/0)	1044960	*entire disk*
3	unassigned	wm	0	0	(0/0/0)	0	
4	var	wm	294 - 367	20.23MB	(74/0/0)	41440	*/var*
5	unassigned	wm	0	0	(0/0/0)	0	
6	usr	wm	368 - 1245	240.08MB	(878/0/0)	491680	*/usr*
7	unassigned	wm	1246 - 1865	169.53MB	(620/0/0)	347200	*/opt*

where **Flag** indicates writable/mountable (**wm**) and writable/unmountable (**wu**).

2.2.3 SGI IRIX 5.X

IRIX 5.X uses and enhanced version of the Unix file system called the Extent File System (EFS) and allows up to 11 partitions on your disk, some of which are used for diagnostic purposes only. The disk format command is ***fx***, which when using the **/label/show/all** menu shows:

```
# fx/label/show> all

----- current drive parameters-----
Error correction enabled          Enable data transfer on error
Don't report recovered errors     Do delay for error recovery
Don't transfer bad blocks         Error retry attempts          1
Do auto bad block reallocation (read)
Do auto bad block reallocation (write)
Drive readahead  enabled          Drive buffered writes disabled
Drive disable prefetch       0    Drive minimum prefetch        0
Drive maximum prefetch       0    Drive prefetch ceiling        0
Number of cache segments     6    CTQ disabled
Read buffer ratio        0/256    Write buffer ratio        0/256
```



```
----- current drive geometry-----
      Tracks/zone =  484      Sect/track =  108
    Alt sect/zone =   50      Interleave =    1        Cylinders = 3875
 Alt track/volume =    8   Cylinder skew =   15            Heads =    3
   Alt track/zone =    1      Track skew =   11   Data bytes/sec =  512
  Rotational rate = 4500
----- partitions-----
part  type       cyls            blocks          Megabytes   (base+size)
  0: efs        8 + 3053       2584 + 986119        1 + 482
  1: raw     3061 + 253      988703 + 81719       483 + 40
  8: volhdr     0 + 8             0 + 2584          0 + 1
 10: volume     0 + 3314          0 + 1070422       0 + 523
----- bootinfo-----
 root partition = 0     swap partition = 1    bootfile = /unix
----- directory entries-----
 0: sgilabel   block    2 size     512  2: ide        block  288 size  977920
 1: sash       block    3 size  140800
----- sgi-info-----
   serial = 0000                        name = SGI     IBMDSAS-3540    S47K
```

Here partition **0** contains the user files, including root, usr, etc. Larger systems may have /usr as a separate partition on partition **6**. Swap is partition **1**. The entire usable disk, excluding the volume header is partition **7**. The volume header is on partition **8**, including some diagnostic tools and standalone programs. The entire drive, including the volume header is partition **10**. The ***prtvtoc*** command will provide similar information without the destructive danger of ***fx***. The ***dvhtool*** command can be used to report or change the disk volume header. To **list** the header information a command similar to the following will work, specifying the raw device for the volume header:

```
# dvhtool -v list /dev/rdsk/dks0d1vh

Current contents:
        File name         Length      Block #
        sgilabel             512            2
        sash              140800            3
        ide               977920          288
```

In this listing **sash** is the standalone shell.

You can set aside additional maintenance partitions, if you have the disk space.

2.2.4 Ultrix 4.X

Ultrix 4.X uses the BSD 4.2 disk format with the disk divided into 8 partitions, a -> h. You can use ***chpt*** with the "**-q**" option (for query) to display the disk label, e.g.:

```
# chpt -q /dev/rrz2a
   /dev/rrz2a
   Current partition table:
   partition  bottom          top           size          overlap
       a          0        32767          32768          c,d,e,f,h
       b      32768       163839         131072          c
       c           0      1956863        1956864      a,b,d,e,f,g,h
       d          0            0              0          a,c,e,f,h
       e          0            0              0          a,c,d,f,h
       f          0            0              0          a,c,d,e,h
       g     163840      1956863        1793024          c
       h          0            0              0           a,c,d,e,f
```

2.2.5 Digital UNIX

OSF/1 uses the BSD disk format also. To display the disk partitions use the ***disklabel*** command with the "***-r***" option (for read only), e.g.:

```
# disklabel -r rz0
# /dev/rrz0a:
type: SCSI
disk: rz26
label:
flags:
bytes/sector: 512
sectors/track: 57
tracks/cylinder: 14
sectors/cylinder: 798
cylinders: 2570
sectors/unit: 2050860
rpm: 3600
interleave: 1
trackskew: 0
cylinderskew: 0
headswitch: 0            # milliseconds
track-to-track seek: 0   # milliseconds
drivedata: 0
8 partitions:
#      size   offset    fstype [fsize bsize cpg]
  a: 131072        0    4.2BSD   1024  8192  16    # (Cyl.    0 - 164*)
  b: 262144   131072    unused   1024  8192        # (Cyl.  164*- 492*)
  c:2050860        0    unused   1024  8192        # (Cyl.    0 - 2569)
  d: 552548   393216    unused   1024  8192        # (Cyl.  492*- 1185*)
  e: 552548   945764    unused   1024  8192        # (Cyl. 1185*- 1877*)
  f: 552548  1498312    unused   1024  8192        # (Cyl. 1877*- 2569*)
  g:1001000   393216    4.2BSD   1024  8192  16    # (Cyl.  492*- 1747*)
  h: 656644  1394216    4.2BSD   1024  8192  16    # (Cyl. 1747*- 2569*)
```


2.2.6 Label

For SunOS the **label** is contained on the first sector of the first partition. The next 15 sectors contain the **Boot Area**.

IRIX reserves the first block (also 512 bytes) but doesn't use it for anything.

2.2.7 Cylinder Groups

Following the **label** in the root partition, and in all the other partitions that are intended for the UNIX file system, we create a series of **Cylinder Groups**. Each Cylinder Group contains a **Superblock**, **Cylinder Group Summary Block**, **Inode Table**, and **Data Block Area**.

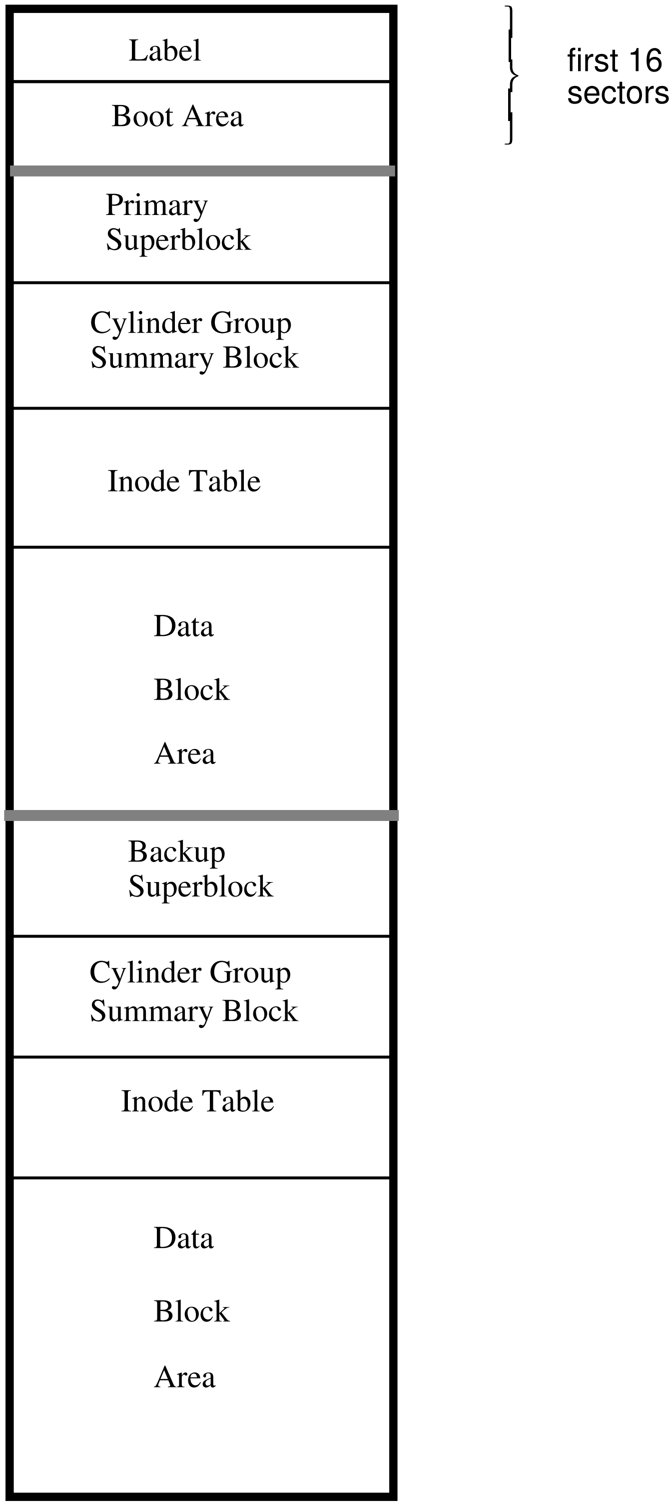
IRIX places the **Superblock** in the second block of the file system.

The **index node**, known as the **inode**, keeps track of the location of the files on the disk.

The first **Superblock** in a file system is the Primary one and the remainder are backup copies for that partition or slice. The **Cylinder Group Summary Block** keeps track of:

- the size of the file system
- the number of inodes and data blocks
- pointers to the last block, fragment, and inode used
- the number of available fragments
- the used inode map
- the free inode map.

FIGURE 2.4 **Logical Disk Layout**

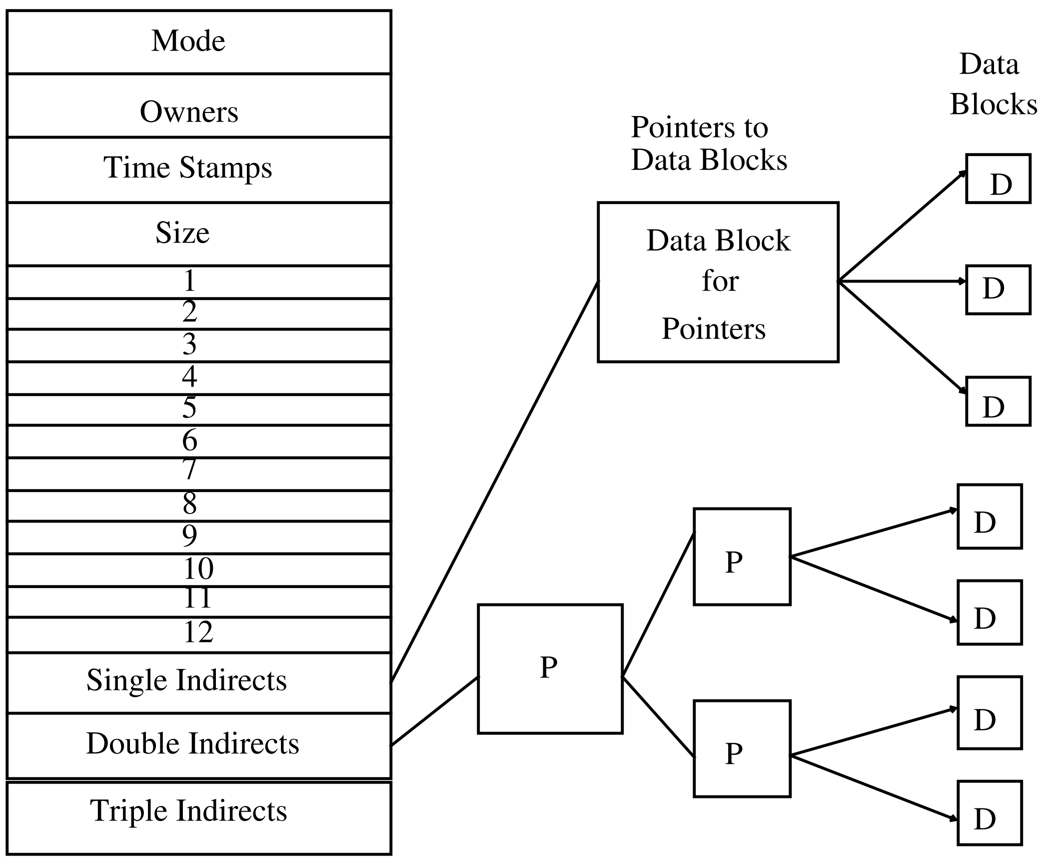


The default size for each **Data Block** is 8192 bytes, divided into 8 fragments of 1024 bytes each.

0	1	2	3	4	5	6	7

Inodes are assigned one per file. Each **Inode Block** consists of ownership, timestamps (creation, modification, access), size, number of hard links, and location of data block information for that file. The inode does not contain the name of the file. That is defined by the directory table information.

FIGURE 2.5 **Inode Block Contents**



Each pointer is 4 bytes long $\Rightarrow$ 8192 bytes/pointer block $\div$ 4 bytes/pointer $*$ 8192 bytes/data block

$\Rightarrow 1.7 * 10^{7}$ bytes for a single indirect.

$* (8192/4) \Rightarrow 3.4 * 10^{10}$ bytes for a double indirect.

$* (8192/4)^2 \Rightarrow 7.0 * 10^{13}$ bytes for a triple indirect.

The maximum size of a Unix file and a Unix file system are limited by the capabilities of the operating system. Those for SunOS and IRIX are listed in the table below.

TABLE 2.1 Maximum File and File System Sizes

Operating System	File Size	File System Size
SunOS 4.1.X	2 GB	2 GB
SunOS 5.X	2 GB	1 TB
SunOS 5.6	1 TB	1 TB
IRIX 5.X	2 GB	8 GB
IRIX 6.2	9,000,000 TB (64-bit Kernel) 1 TB (32-bit Kernel)	9,000,000 TB (64-bit Kernel) 1 TB (32-bit Kernel)

CHAPTER 3 Devices

For the operating system to recognize a hardware device you need to give the device a software name and have the driver that controls the device available to the kernel.

3.1 Logical Names

3.1.1 Disk and Tape Devices

For SunOS 4.1.X the disk and tape logical device names correspond to entries in the ***/dev*** (devices) directory which control access to the physical devices. Some of the devices are:

Disks: sd - SCSI controllers, can control 2 disks.
xy - Xylogics 450/451 SMD controller, can control 2 disks.
xd - Xylogics 7053 controller, can control 4 disks.

Tapes: st - SCSI controller: 1/4" QIC, 8mm, 4mm DAT
xt - 1/2" high density Xylogics 472 controller
mt - 1/2" low density Tapemaster controller, and others (sometimes linked to st)

CD-ROM: sr - SCSI controller

where SCSI stands for Small Computer System Interface.

3.1.2 Ethernet Devices

Ethernet: ie - Intel (82586 chip), Sun-3/75, Sun-3/100, Sun-3/200, Sun-4
le - Lance (AMD chip), Sun-3/xx, SPARCstations (Sun-4c, Sun-4m)
ec - **SGI** Irix
ln - **DEC** Ultrix and Digital UNIX

3.1.3 Device Controllers

The **controller** is the hardware that controls the communication between the system and the peripheral drive unit. It takes care of low level operations such as error checking, moving disk heads, data transfer, and location of data on the device.

3.1.4 Device Drivers

The **device driver** is the software that operates the **controller**. This software must be available to the kernel if you wish to use the device. The device drivers must match the device that you wish to use. Drivers perform functions to: probe, attach, open, close, read, write, reset, stop, timeout, select, strategy, dump, psize, ioctl, and process transmit and receive interrupts for a device. When a program attempts to access a device the kernel traps the request, looks up the appropriate information in it's tables, and transfers control to the device driver.

3.2 Disk Partitioning

3.2.1 SunOS 4.1.X

The disk is organized into logical partitions, each of which corresponds to a device entry, e.g. **/dev/sd0a**, where **sd** is the controller type, **0** represent disk0 attached to the controller, and **a** represents the partition. Partitions "**a**" through "**h**" are allowed, where "**c**" represents the entire disk. The ***format*** program writes a label to the disk on cylinder 0, track 0, sector 0, describing the partitions. Partitions allow you to subdivide your disks and separate data.

You can examine your disk partitioning scheme with the ***dkinfo*** command.

```
# dkinfo sd0
    sd0: SCSI CCS controller at addr f8800000, unit # 24
    1254 cylinders 9 heads 36 sectors/track
    a: 16848 sectors (52 cyls)
       starting cylinder 0
    b: 86184 sectors (266 cyls)
       starting cylinder 52
    c: 406296 sectors (1254 cyls)
       starting cylinder 0
    d: No such device or address
    e: No such device or address
    f: No such device or address
    g: 145476 sectors (449 cyls)
       starting cylinder 318
    h: 157788 sectors (487 cyls)
       starting cylinder 767
```

The devices listed correspond to the logical devices **/dev/sd0a** → **/dev/sd0h** and **/dev/rsd0a** → **/dev/rsd0h**, for the block and character (raw) devices, respectively.

3.2.2 SunOS 5.X

For SunOS 5.X the device naming convention has been changed considerably from that of SunOS 4.1.X. The new convention includes some of the devices' characteristics in the name. Sun's convention is slightly different from the SysV.4 naming convention, because SunOS 5.X limits disk partitions to eight per disk. If you install the binary compatibility package links are created with the old style names to the new device names, so you should be able to use either scheme.

Device names are split into three name spaces:

- physical
- logical
- SunOS 4.X compatible

In the physical name space devices have names consistent with the ones used by the Open Boot PROM. These are kept in the **/devices** directory. Devices that control other devices, such as the bus controller, have a subdirectory under this hierarchy.

The physical device name now contains the hardware information within the name. What was formerly known as /dev/sd0a might now be:

/devices/sbus@1,f8000000/esp@0,800000/sd@0,0:aor

/devices/iommu@0,10000000/sbus@0,10001000/espdma@5,8400000/esp@5,8800000/sd@0,0:a

Similarly, at the Open Boot PROM the first example device would be known as:

/sbus@1,f8000000/esp@0,800000/sd@0,0:a

This name uniquely identifies the physical location of the hardware device to the system. It has a series of node names, each separated by a slash (/) of the form

name@address:arguments

where

name	is a text string that usually has a mnemonic value, e.g. sbus, esp, sd
@	precedes the address parameter
address	text string, usually in the form hex_number, hex_number
:	precedes the arguments parameter
arguments	text string intended to pass additional information to the device driver

So in the examples above:

sbus@1,f8000000	represents the address on the main system bus occupied by the SBus,
esp@0,800000	represents the SBus slot number and offset within the slot (slot 0, offset 80000) for the SCSI controller, esp
sd@0	represents a SCSI disk, sd, attached to the SCSI bus with Target Number 0
0	is the SCSI Logical Unit Number of sd, and
a	is the disk Partition.

The logical device names are kept in **/dev** and are symbolic links to the physical device names in **/devices**. The logical names are what you will generally use. The logical disk names contain the controller number, target number if the disk is on a device bus, disk number, and slice (formerly partition) number. Every disk device has an entry in both the /dev/**dsk** and /dev/**rdsk** directories, for the block and raw disk devices, respectively.

So the logical device name for what was known under SunOS 4.1.X as /dev/sd0a would be:

/dev/dsk/c0t0d0s0

where

c0	Controller Number
t0	Target Number
d0	Disk Number
s0	Slice (Partition) Number

and this is a symbolic link to:

/devices/sbus@1,f8000000/esp@0,800000/sd@0,0:a

For disks that are directly attached you would drop the target number entry, e.g. something similar to:

/dev/dsk/c0d0s0.

The ***dkinfo*** command is not available for examining disks under SunOS 5.X. You can check disk information with the new command ***prtvtoc***, but this must be run with root permissions. Here's an example of the output you might see:

```
# prtvtoc /dev/rdsk/c0t3d0s2
    * /dev/rdsk/c0t3d0s2 partition map
    * Dimensions:
    *     512 bytes/sector
    *      80 sectors/track
    *       9 tracks/cylinder
    *     720 sectors/cylinder
    *    2500 cylinders
    *    1151 accessible cylinders
    * Flags:
    *   1: unmountable
    *  10: read-only
    *
    *                       First     Sector    Last
    * Partition  Tag  Flags    Sector     Count     Sector   Mount Directory
         0        2    00           0     37440      37439   /
         1        3    01       37440     66240     103679
         2        5    00           0    828720     828719
         5        6    00      103680    348480     452159   /opt
         6        4    00      452160    287280     739439   /usr
         7        8    00      739440     89280     828719   /home
```


where some of the **Tag** codes are:

Boot	1
Root	2
Swap	3
Usr	4

and the **Flags** are:

Mountable, read/write	00
Not Mountable	01
Mountable, read only	10

3.2.3 IRIX 5.X

IRIX has the physical devices in /dev, with disk entries in **/dev/dsk** (block devices) and **/dev/rdsk** (raw devices) for each of partitions 0 → 7, in the form **dksXdYsZ**. **X**, **Y**, and **Z** are numbers, with **X** starting at 0 for your default SCSI interface, **Y** starting at 1 for your first disk, and **Z** going from 0 through 7. Additionally there are raw device entries for the volume and volume header partitions, in the form **dksXdYvol** and **dksXdYvh**, respectively. for the root disk you can also reference the root device as **/dev/root** and **/dev/rroot**, for the block and character devices, respectively. There are similar entries for swap, **/dev/swap** and **/dev/rswap**; usr, **/dev/usr** and **/dev/rusr**; and an access to the header, **/dev/rvh**.

3.2.4 Ultrix and Digital UNIX

Ultrix and Digital UNIX (formerly OSF/1) follow the BSD style. The disk devices are know as **/dev/rz0a** → **/dev/rz0h** and **/dev/rrz0a** → **/dev/rrz0h** for the block and character devices, respectively, for physical devices **a** through **h**.

3.3 Disk Label and Bootblock

The disk ***label*** is put on the first sector of the first partition. This label contains the partitioning information for the disk. You can use the ***format*** program to format, check, partition, and label an unmounted disk. For Ultrix use ***rzdisk/radisk*** to format a SCSI/DSSI disk and ***chpt*** to change disk partitions. For IRIX 5.X use ***fx*** or ***dvhtool***.

The SunOS 4.1.X EEPROM expects to find bootblock code in the bootblock area of a disk, sectors 1 through 15 of the first partition. This program is put there by the ***installboot*** program and allows the PROM to locate the boot program on the disk. Under SunOS 5.X the boot program and the boot block uses the drivers resident on the PROM or on the Sbus card. So the bootblock area doesn't contain the actual location of the disk block where the boot program resides. The SunOS 5.X ***bootblk*** program can read the file system to locate the boot program.

3.4 Tapes

3.4.1 SunOS 4.1.X

The tape devices are generally referenced as the raw device, either **rst**, **rxt**, or **rmt** devices. For SCSI drives the tape device should have a target ID of either 4 or 5, **/dev/rst0** or **/dev/rst1**, respectively. If the drive can handle more than one density then adding 8 to the device number should access the higher density, e.g. **/dev/rst8** and **/dev/rst9**, respectively. You access the no-rewind device by prepending the device name with an "**n**", e.g. **/dev/nrst0** and **/dev/nrst1**, respectively.

3.4.2 SunOS 5.X

The tape naming convention has been changed for SunOS 5.X. The tape devices are found in the subdirectory **/dev/rmt**. The tape devices are numbered from 0 and may include in their name certain characteristics, such as tape density, whether it's a no-rewind device, and whether it should use BSD behavior. The latter specifies that when reading past an **EOF** mark it should return the first record of the next file and that when closing a no-rewind device it should skip a tape space forward.

The logical tape name would be something like:

    /dev/rmt/XYbn

where

X	specifies the Logical Tape Number
Y	specifies the Tape Density (l=low, m=medium, h=high, u=ultra, c=compressed)
b	specifies BSD Behavior
n	specifies the no-rewind device.

So if you want to use the 5 GByte capacity on a 2/5Gbyte 8mm tape you would use the device

    /dev/rmt/0h

which corresponds to the physical device:

    /devices/sbus@1,f8000000/esp@0,800000/st@4,0:h

For QIC drives l=>QIC-11, m=>QIC-24, and h=>QIC-150, though if your drive can only write one format that's what will be written regardless of the format selected.

3.4.3 IRIX 5.X

The default tape device is **/dev/nrtape**.

3.4.4 Ultrix and Digital UNIX

The default tape device is **/dev/rmtXD** and **/dev/nrmtXD**, where **X** is a number and **D** specifies the density, i.e. l, h, etc. The "**n**" in front of rmt specifies the no-rewind device.

CHAPTER 4 The UNIX File System

4.1 File Systems

Before you can use the disk partitions by the OS you need to construct a file system on them. Generally you create a separate file system on each partition, except those used for swap which are accessed as raw partitions, and then join them together to form a hierarchical, tree like structure.

4.1.1 File system implementation

The disk must first be formatted and partitioned before it can be used by the OS. You format the disk with the ***format*** command which uses the **/etc/format.dat** configuration file for parameter values. You construct a new file system with ***newfs/mkfs***. ***newfs*** is a friendly front-end to ***mkfs***. It reads the disk label, builds the file system, and installs the bootstrap program if its the root partition. It sets aside space for inodes (default is 1 inode per 2048 bytes of data space) and reserves free space for use only by root (default is 10%, which can be reset later with ***tunefs***). The new file system should be checked for internal consistency with ***fsck***, and can then be mounted by the OS.

4.1.2 Function and contents of superblock

The **superblock** contains information on the size of the file system, the number of inodes, the number of data blocks, the free and used inodes, and the block size for the file system. The superblock is kept in memory and in multiple locations on disk for each file system.

4.1.3 The inode area

The OS interprets requests to read/write/delete files by allocating **inodes** and data blocks. An area is set aside on each partition to store the inode table for that partition.

Inodes contain information on files and directories stored in the file system, their file permissions, link count, state and type of file, time stamps, size, and pointers to location of data blocks. The inodes do NOT contain the name of the file. An inode keeps track of its own state; whether its allocated or not.

4.1.4 Directories

Each **directory** contains the names of files within the directory and the inode numbers associated with these files. A directory is just an ordinary file in the data block area. It's a binary file, which contains tabular information similar to (e.g. for /usr):

2	.
2	..
3	lost+found
2688	export
5376	bin
10752	ucb
13440	etc
26880	include
4570	lib
94123	hosts
7	boot
102177	local
...	

where the current directory (.) and the parent directory (..) are the same, because /usr is on a separate disk partition. **lost+found** is created by ***newfs*** for use by ***fsck***.

4.1.5 Data area

The **data area** contains the users data, files, and directories. Symbolic links also reside in the data area, and point to files of directories on this or other file systems.

4.1.6 Making and mounting file systems - summary

1. ***format*** - format and partition the physical disk
   chpt - Ultrix command to partition the disk
2. ***newfs*** - construct the file system on each partition
3. ***fsck*** - check the new file systems for internal consistency
4. ***mount/umount*** - mount/unmount the file systems
5. ***/etc/fstab***, or ***/etc/vfstab*** (SunOS 5.X only) - edit this file to mount these file systems automatically at start of multi-user mode

4.2 File System Types

SunOS has 3 different types of file systems: **disk-based**, **distributed**, and **pseudo**. The **disk-based** file systems include hard disks, CDROMs, and diskettes. The **distributed** file systems manage network resources. The **pseudo** file systems are memory-based and do not use any disk space. They provide access to kernel information and facilities.

TABLE 4.1 File System Types

Type	Name	Description	SunOS 4.1.X	SunOS 5.X
Disk-based	ufs	UNIX File System, based on BSD Fat Fast File System (default)	yes (known as 4.2)	yes
	hsfs	High Sierra File System, used by CDROMs and supports Rock Ridge extensions. Very similar to ufs, except that it does not support writable media or hard links	yes	yes
	pcfs	PC File System, to allow read/write access to DOS formatted disks	yes	yes
	cachefs	Cache File System, allows use of local disk to store frequently accessed data from a remote file system or CDROM	no	yes
Distributed	nfs	Network File System, the default distributed file system type	yes	yes
	rfs	Remote File Share, AT&Ts RFS product	yes	no (only < 5.3)
	autofs	Automount File System, automounts NFS file systems, as needed, using NIS and NIS+ maps	no	yes
Pseudo	tmpfs	Temporary File System, file storage in memory and swap without the overhead of writing to a ufs file	yes	yes
	specfs	Special File System, allows access to the special character and block devices	yes	yes
	lofs	Loopback File System, creates a virtual file system which can overlay or duplicate existing files. The files are accessible from either path	yes	yes
	tfs	Translucent File System, allows mounting of a file system on top of existing files, with both visible	yes	no
	proc	Process Access File System, allows access to active processes and their images	no	yes
	fdfs	File Descriptor File System, allows access to file names using descriptors	no	yes
	namefs	Name File System, used by STREAMS for dynamic mounts of file descriptors on top of files	no	yes
	fifos	First In First Out File System, allows process access to named pipe files	no	yes
	swapfs	Swap File System, used by the kernel to manage swap space	no	yes

4.2.1 Temporary File System (tmpfs)

A temporary file system uses memory to simulate a traditional disk partition. Normal file system writes are scheduled to be written to disk along with access control information, but the files actually reside in memory only.

A good candidate for a tmpfs is a partition that will have many small files that will be accessed often, e.g. **/tmp**. This will considerably speed up their access time. **Tmpfs** files and directories are NOT saved when the system shuts down.

Tmpfs is recommended for systems that do a lot of compiling and loading of programs and have large amounts of memory (> 16 MB) and swap space.

Disadvantages are that it reduces the amount of swap space available for other process and that it is volatile.

To mount a temporary file system under SunOS 4.1.X as /tmp:

```
# mount -t tmp swap /tmp
```

where the ***-t*** option indicates the type is **tmp**.

To do this under SunOS 5.X you specify the ***-F*** option:

```
# mount -F tmpfs swap /tmp
```

Note that the file system type is specified as **tmp** in SunOS 4.1.X and **tmpfs** in SunOS 5.X.

In order to use **tmpfs** under SunOS 4.1.X the **TMPFS** option must be configured in the kernel, and an entry such as:

```
swap /tmp tmp rw 0 0
```

could be put in **/etc/fstab**.

Under SunOS 5.X the **/etc/vfstab** entry would look like:

#device	device	mount	FS	fsck	mount	mount
#to mount	to fsck	point	type	pass	at boot	options
swap	-	/tmp	tmpfs	-	yes	-

4.2.2 Translucent File System (TFS)

The **translucent** file system allows users to mount a writable file system on top of a read-only file system. The contents of the lower system remain visible when the file system is mounted in this way, so long as there is no file system of similar name in the top file system (SunOS 4.1.X only).

So **TFS** is a series of stacked file systems where searching for files is done from the top of the stack downward until the first file of that name is found.

Modification of files can be done on the top most file system only. If a user tries to remove a file from a directory not in the foremost file system TFS creates a whiteout in the topmost file system and leaves the lower one intact. Further attempts by the user to access that file are answered as if the file had been removed, when in fact it is still intact at the lower file system, and can be accessed by other users not using TFS.

TFS requires both the **LOFS** (loopback filesystem) and **TFS** (translucent filesystem) options be compiled into the kernel. It also requires the following line in the **/etc/inetd.conf** file:

```
tfsd/1-2	dgram	rpc/udp wait root /usr/etc/tfsd tfsd
```


To mount a TFS file system use the following command:

```
# mount -t tfs /src/fgf/test /usr/bin
```

Unmount with:

```
# umount /usr/bin
```

4.2.3 Swapfs

Swap and how it's managed by the OS has changed considerable in Solaris 2. If you have enough memory you can now run without swap space should you so desire. The OS now treats main memory as if it were a backing store. SunOS 4.1.X required that all memory have a physical backing store. So if you set aside less swap than physical memory, you couldn't use all the memory available. This also meant that the swap space was reserved even if the program and data could fit entirely in memory. This is no longer the case under the **virtual swap space** of SunOS 5.X. To implement this concept the pseudo file system, **swapfs** was created. Swapfs provides names for anonymous memory pages. Whenever a process executes a file system operation on a page named by swapfs, swapfs gains control of the page. Swapfs can change the name of the page and back it up with physical store, if necessary. Anonymous memory pages appear to the system as if they were backed up by real swap space, though this is not actually the case. As more memory is needed these pages can be moved to available physical swap space by swapfs.

Swapfs uses main memory as if it were swap space. So in effect swap space is expanded to include main memory as well as physical swap space. A certain fraction of main memory is always reserved for the kernel data structures and is not available to swapfs. When releasing swap space swapfs always releases main memory before physical backing swap space.

Under swapfs it's now also possible to remove swap devices and files while the system is running, so long as this swap area is not being used or if the files in this swap area can be moved to another swap area or memory.

All swap partitions, including the primary one, are now mounted through entries in **/etc/vfstab**, e.g.:

#device	device	mount	FS	fsck	mount	mount
#to mount	to fsck	point	type	pass	at boot	options
/dev/dsk/c0t3d0s1	-	-	swap	-	no	-

4.2.4 Cachefs

The cache file system, **cachefs**, lets you use a disk drive on a local system to cache frequently used data from a remote file system or CDROM. The cache is a temporary storage area for those files. The data is read from the original file system and stored in the cache on the local disk. The next time the file is accessed, it will come from the cache, after first insuring that the original file has not changed. This reduces network traffic and/or increases response time from a slow medium such as CDROM. The cache file system can store files from one or more remote file systems on a local disk. This should be useful in situations where you have enough disk space to set aside for **cachefs** and where your local machine is fast enough that you don't lose too much time caching the file the first time.

4.2.5 Autofs

The automounting file system, **autofs**, mounts file systems when access is requested and unmounts the file system after a few minutes of inactivity. There's a certain amount of overhead traffic required to maintain the NFS connection. **Autofs** allows you to break that connection when the file system is not being used and restart it again automatically when access is desired. This reduces network traffic. The automount daemon, ***automountd***, is run to mount file systems requested by autofs.

4.3 Compatibility

SunOS 5.X and SunOS 4.1.X formatted disks are compatible. There are a few tags that can be added to the 5.X disks during formatting or labeling that are ignored if the disk is used on a 4.1.X system. Likewise, if a 4.1.X disk is used on a 5.X system these missing tags will be assigned the default values.

The expanded disk label includes:

- **volume name** to identify the disk device, up to 8 characters
- **partition tags** to identify partition usage; valid tags are:

  unassigned
  boot
  root
  swap
  usr
  backup
  stand
  var
  home
- **partition flags** that specify read/write access and whether the partition is mountable; valid flags are:

w	read/write
r	read only
m	mountable
u	unmountable

The default partition tag and flag values for a disk are:

0	root	wm
1	swap	wu
2	backup	wm
3	unassigned	wm
4	unassigned	wm
5	unassigned	wm
6	usr	wm
7	unassigned	wm

The ***format*** utility can be used to set volume names and retag the partitions. The ***prtvtoc*** command can be used to examine the disk label. You can also examine the disk labels with the ***verify*** subroutine of the format command:

```
# format> verify
format> verify
Primary label contents:

ascii name      = <SUN0424 cyl 1151 alt 2 hd 9 sec 80>
pcyl            = 2500
ncyl            = 1151
acyl            =    2
nhead           =    9
nsect           =   80
Part     Tag          Flag     Cylinders        Size       Blocks
  0      root         wm        0 -   51       18.28MB    (52/0/0)
  1      swap         wu       52 -  143       32.34MB    (92/0/0)
  2     backup        wm        0 - 1150      404.65MB  (1151/0/0)
  3 unassigned        wm        0                  0      (0/0/0)
  4 unassigned        wm        0                  0      (0/0/0)
  5        -          wm      144 -  627      170.16MB   (484/0/0)
  6       usr         wm      628 - 1026      140.27MB   (399/0/0)
  7      home         wm     1027 - 1150       43.59MB   (124/0/0)
```

The ***fmthard*** command can be used to update the VTOC (Volume Table of Contents) of a hard disk. The disk needs to be first labeled by format.

4.4 Names & contents of important UNIX directories

TABLE 4.2 Unix Directories

Directory	Description	SunOS 4.1.X	SunOS 5.X	IRIX 5.X	Ultrix 4.X	Digital UNIX
/	root - kernel	yes	yes	yes	yes	yes
/sbin	files required to start the system and scripts to control the boot process	yes, but scripts are in /etc	yes	yes	no	yes
/etc	files required to boot the system and communicate, and scripts to control the boot process	yes	yes, but some scripts are in /sbin	yes	yes	yes
/etc/config	system configuration option files	no	no	yes	no	no
/etc/cron.d	cron access files and FIFO	no	yes	yes	no	no
/etc/default	default system configuration	no	yes	yes	no	no
/etc/dfs	distributed file sharing configuration	no	yes	no	no	no
/etc/fs	static file system specific mount commands	no	yes	no	no	no
/etc/fdmns	file domain names and devices, symbolic links to the file volumes	no	no	no	no	yes
/etc/inet	internet services configuration	no	yes	no	no	no
/etc/init.d	internet service scripts run by init	no	yes	yes	no	no
/etc/lib	shared libraries required for boot	no	yes	no	no	no
/etc/lp	line printer system configuration	no	yes	no	no	no
/etc/mail	mail configuration	no	yes	no	no	no
/etc/net	configuration for transport-independent network services	no	yes	yes	no	no
/etc/opt	optional software package configuration files	no	yes	yes	no	no
/etc/rc#.d	operations performed when entering run level # (S,0,1,2,3)	no	yes	yes	no	no
/etc/saf	service access facility configuration	no	yes	no	no	no
/etc/security	security audit configuration	no	yes	no	no	no
/etc/sec	"	no	no	no	yes	yes
/usr	directories of system files	yes	yes	yes	yes	yes
/usr/bin	system binary files	yes	yes	yes	yes	yes
/usr/etc	further system communication and administration programs	yes	no	yes	yes	no

TABLE 4.2 **Unix Directories**

Directory	Description	SunOS 4.1.X	SunOS 5.X	IRIX 5.X	Ultrix 4.X	Digital UNIX
/usr/sbin	"	no	yes	yes	no	yes
/usr/lib	libraries of object files, sendmail	yes	yes	yes	yes	yes
/usr/4lib	SunOS 4.1 libraries required for binary compatibility	no	yes	no	no	no
/usr/5bin	System V binaries	yes	no	no	no	no
/usr/5include	System V include files	yes	no	no	no	no
/usr/5lib	System V libraries	yes	no	no	no	no
/usr/aset	automated security enhancement tool files and programs	no	yes	no	no	no
/usr/ucb	BSD binaries	yes	yes	no	yes	yes
/usr/bsd	"	no	no	yes	no	no
/usr/ccs	compiler support systems	no	yes	no	no	yes
/usr/dt	CDE desktop hierarchy	no	yes	no	no	no
/usr/lib/fs	file system dependent modules	no	yes	no	no	no
/usr/lib/lp	line printer databases and programs	no	yes	no	no	no
/usr/lib/netsvc	network service utilities	no	yes	no	no	no
/usr/lib/nfs	NFS daemons and programs	no	yes	no	no	no
/usr/lib/nis	NIS+ programs and setup scripts	no	yes	no	no	no
/usr/lib/saf	SAF daemons and programs	no	yes	no	no	no
/var	directories for administrative programs and logs	yes	yes	yes	yes	yes
/var/adm	system log and account files	yes	yes	yes	yes	yes
/var/log	system log files	yes	yes	no	no	no
/var/spool/mail	mail spool directory	yes	no	no	yes	yes
/var/mail	mail spool directory	no	yes	yes	no	no
/var/yp	NIS tables and Makefile for updating NIS	yes	no	yes	yes	yes
/var/nis	NIS+ tables	no	yes	no	no	no
/var/spool	directories for cron, logs, etc.	yes	yes	yes	yes	yes
/var/sadm	databases maintained by package administration utilities	no	yes	no	no	no
/var/inst	databases maintained by inst utility	no	no	yes	no	no
/var/saf	service access facility log and account files	no	yes	no	no	no
/dev	devices directory	yes	yes	yes	yes	yes
/dev/dsk	block disk devices directory	no	yes	yes	no	no

TABLE 4.2 **Unix Directories**

Directory	Description	SunOS 4.1.X	SunOS 5.X	IRIX 5.X	Ultrix 4.X	Digital UNIX
/dev/rdsk	raw disk devices directory	no	yes	yes	no	no
/dev/pts	pseudo terminal (pty) devices directory	no	yes	yes	no	yes
/dev/rmt	raw tape devices directory	no	yes	yes	no	yes
/dev/term	terminal devices directory	no	yes	no	no	no
/dev/sad	entry points for STREAMS administrative drivers	no	yes	yes	no	yes
/devices	physical devices directory	no	yes	no	no	no
/home	user directories	yes	yes	no	no	yes
/usr/people				yes	no	
/usr/users					yes	
/tftpboot	client boot programs	yes	yes	no	no	no
/usr/local/boot				yes		
/tmp	temporary files	yes	yes	yes	yes	yes
/usr/local	locally installed files	optional	optional	optional	optional	optional
/opt	locally installed packages and files	no	yes	yes	no	yes
/kernel	contains the kernel and drivers for the kernel	no	yes	no	no	no
/platform	hardware specific files for kernel support	no	>=2.5	no	no	no
/stand	standalone environment programs, can be accessed from the PROM	no	no	yes	no	no
/proc	for process access file system, it provides access to all current processes	no	yes	yes	no	yes
/sys	object files to reconfigure the kernel	yes	no	no	yes	yes
/vol	vold mount points	no	yes	no	no	no

4.5 File structure of standalone and server machines

Standalone -	/	root
		swap
	/var	spool
	/usr	system programs
	/home	user space
	/opt	optional software packages (SunOS 5.X)

Server -	/	root
		swap
	/var	spool
	/usr	server system programs
	/home	user space
	/export/[root,swap,exec]	client root, swap, and /usr
	/opt	optional software packages (SunOS 5.X)
	/usr/local	optional software packages (SunOS 4.X).

4.6 Disk Partitioning

In the old days you normally partitioned the disks to allow just a little more space than actually needed for system files. For file systems that were likely to grow, like /home and /usr/local, you made as large as possible while balancing your needs and resources. The root partition was expected to have few files that would change on a daily basis (/etc/passwd being the notable exception), and this was a good thing. With few files changing there was less likelihood that the file system would be corrupted. Disks were not as reliable as they are now. A problem arises, though, if you need to add more space to a partition. Most OSs won't let you transparently add this space. You normally have to back up the disk, repartition the drives, and restore the files from the backup.

As operating systems grew in size, and this was especially noticeable in the transition to Solaris, more files, and more changing files, were placed in the root partition. For example, by default the Solaris install put /var in root, but /var now changes considerably every time you add a new software package or install an OS patch. So the old idea of a small, little-changing root partition doesn't hold, unless you separate /var on another partition.

There has been a lot of discussion of this topic in the system administration newsgroups recently. For standalone machines it's probably most efficient to just have two partitions: one for swap, and one for everything else. For servers it's better to isolate the different types of files on separate partitions. Below I'll summarize many of the arguments for and against the two positions.

TABLE 4.3 **Disk Partitions**

Argument	Separate Partitions	Combined Partitions
Proper Size	Difficult to maintain	No maintenance
Tape Backups	Easier with smaller partitions	Higher density tapes, stackers
Small Root Partition	Less chance of corruption	Can rebuild the OS quickly
Any Corrupted Partition	Can more easily restore that file system	Can boot diskless & rebuild the OS quickly
NFS File Service	Allow different mount options	One mount option
Quotas	Separate quotas by partition	One quota
Space	Can run out of space on one partition while lots of space on others	Still has space available to users
Runaway Programs	Fills space only on that partition	Fills all available space, this may shut down essential services, e.g. mail & logs

CHAPTER 5 File System Management

5.1 File System Management

5.1.1 Maintenance

Use ***fsck*** to examine the disk partitions at startup. The only time this should be disabled (e.g. fasthalt/fastboot, SunOS 4.1.X only) is if you are testing the boot procedure or a new kernel. Under SunOS 4.1.2 and above if you bring the system down cleanly the disk partitions will be marked **FSCLEAN**. The fsck will notice this and skip the check. This is okay. If the system has not come down cleanly then be sure to force ***fsck*** to check the disk the next time you boot.

5.1.2 File system updates

The operating system doesn't write immediately to disk when a file is modified. To save time it writes to a buffer cache which is much faster than writing to disk. When the buffer cache fills up the information, along with the appropriate inode numbers to identify the files, is written to disk. If the system is somehow interrupted before the buffer is written to disk, the file system on the disk can become corrupted.

5.1.3 Sync command

Sync causes the system to flush its buffers and write all waiting data to disk. Sync should be executed periodically by the OS, either in the kernel, or through a periodic program, e.g. cron. SunOS does this for you every 30 seconds using either the ***update*** (4.1.X) or ***fsflush*** (5.X) command.

5.1.4 Causes of file system corruption

Most common causes of file system corruption are due to improper shutdown or startup procedures, hardware failures, or NFS write errors. Shutdown should be done through one of the system shutdown commands; these sync the file system first. Never shut the system down by turning off the power. Taking a mounted file system off-line or physically write-protecting a mounted file system can also corrupt the disk. Improper startup includes not checking a file system for consistencies (fsck) before mounting it and not repairing any inconsistencies discovered by fsck. Hardware failures could be a bad block on disk, a bad disk controller, a power outage, or accidental unplugging of the system. Software errors in the kernel can also cause file system corruption.

5.2 Fsck

The ***fsck*** command checks and corrects file system inconsistencies. The file system should be unmounted or "quiet" when running fsck. Ideally, it should be unmounted, but this is not always possible for the root file system. fsck makes several passes through the file system, each time examining a different feature.

5.2.1 The lost+found directory

This directory is created when the file system is made by ***newfs***. ***fsck*** copies problem ("lost") files here. fsck can't create its own directories so newfs must do this first. When creating the directory newfs makes entries so that, should it need to, numerous files could be put there by fsck.

5.2.2 Superblock consistency

fsck checks for inconsistencies involving **file system size**, **free block count**, and **free inode count** in the superblock. ***fsck*** can not independently verify the file system size, but it can check that this size is larger than the sum of the superblock and inode blocks. All other fsck checks require that the file system size and layout information be correct.

5.2.3 Inode consistency

fsck checks for the allocation state, the format and type, link count, duplicate blocks (blocks already claimed by another inode), bad blocks, inode size, and block count for each of the inodes, starting with inode 2. Inode 0 is reserved to mark unused inodes, and inode 1 is reserved for a future service. If an inode has a non-zero link count, but fsck, when checking the directory entries, finds no reference for that inode, it places the file referenced by the inode in **lost+found**.

5.2.4 Data block consistency

fsck can't check ordinary data blocks, but it can check ***directory*** data blocks. These it checks for inode numbers pointing to unallocated inodes, out-of-bounds inode numbers, incorrect inode numbers for "**.**" and "**..**", and directories not connected to the file system. The latter it will link back into the file system by putting an entry for it in the **lost+found** directory. The directory entry for "**.**" should be the first entry in a directory block, and it should reference itself, i.e. have the inode number for the directory. The second entry in the directory should be "**..**", and reference the inode for the parent of this directory. For the root directory "**..**" would reference the inode pointing to itself. In addition to ordinary data blocks and directory data blocks there exist symbolic link data blocks. These contain the path name for the link.

5.2.5 Phases of fsck

fsck sets up tables for storing inodes and comparisons, verifies validity of ***fsck*** options, then runs through the 6 phases. After initializing it's tables ***fsck*** runs through:

Phase 1: Check Blocks and Sizes	- checks inodes for inconsistencies
Phase 2: Check Path-Names	- checks directory <-> inode consistencies
Phase 3: Check Connectivity	- checks that all directories are connected to the file system
Phase 4: Check Reference Counts	- compares link count information from Phases 2 & 3, correcting discrepancies
Phase 5: Check Cylinder Groups	- checks free blocks and the used inode maps for consistency
Phase 6: Salvage Cylinder Groups	- update the tables to reflect any changes made in earlier passes

5.2.6 Fsck corrective action

fsck will prompt for corrective action whenever an inconsistency is found. If the file system is modified you will need to reboot WITHOUT syncing the disk. You do NOT want to write the in-core copies of the system tables to the disk, as that will undo the corrective action taken by fsck.

5.3 Disk Check Commands

5.3.1 ncheck command

ncheck will generate names from **inode** numbers. ***ncheck*** can be used to find the pathnames of any files reported as problems by ***fsck***. Provide the list of inodes following the **-i** option, with a space separated (SunOS 4.X), or comma separated (no whitespace, SunOS 5.X) list, e.g. for SunOS 4.X:

```
# ncheck -i 8689 29478 12903 /dev/rsd0h
/dev/rsd0h:
8689                    /frank/sunos/disk_info
29478                   /frank/uts/www
12903                   /frank/cosug/membership
```

5.3.2 Disk geometry commands

As we saw earlier in this course ***dkinfo*** and ***prtvtoc*** are the commands to report the disk geometry and partitions for SunOS 4.1.X and 5.X, respectively. For Ultrix the command ***dkio*** has a similar function.

5.3.3 Disk space commands

df reports the free disk space or inodes on file systems, e.g. to report the disk space:

```
# df /dev/sd0h
Filesystem        kbytes   used   avail capacity  Mounted on
/dev/sd0h         303338  263320   9684    96%    /home
```

and to report the inodes, e.g.:

```
# df -i /dev/sd0h
Filesystem        iused  ifree  %iused  Mounted on
/dev/sd0h         11922  33134   26%   /home
```

du reports the number of disk blocks used by directory or file, e.g.:

```
# du src                        (-s ⇒ sum of disk blocks)
40    src/ntp/hp
418   src/ntp
66    src/traceroute/bin
66    src/traceroute/vj_traceroute
411   src/traceroute
830   src
```

In SunOS 5.X the commands ***df*** and ***du*** report in different formats. ***du*** uses 512-byte blocks by default, though the **-k** option will report in kilo-bytes. The **-k** option to ***df*** will report disk information in a format similar to that of SunOS 4.X, and the **-l** option specifies only local disks, e.g.:

```
# df -lk
Filesystem           kbytes     used     avail    capacity   Mounted on
/dev/dsk/c0t3d0s0     17295    12156      3419      78%      /
/dev/dsk/c0t3d0s6    134823   109088     12255      90%      /usr
/proc                     0        0         0       0%      /proc
fd                        0        0         0       0%      /dev/fd
swap                  22716        8     22708       0%      /tmp
/dev/dsk/c0t3d0s7     41807    15381     22246      41%      /home
/dev/dsk/c0t3d0s5    163311    75036     71945      51%      /opt
```

5.4 Swapping and Paging

SunOS uses virtual memory, so that disk area (swap space) is used as an extension of physical memory for temporary storage when the operating system tries to keep track of processes requiring more physical memory than what is available. When this happens the swap space is used for swapping and paging.

Paging is when individual memory segments, or pages, are moved to or from the swap area. When memory is low portions of a process (data areas, but not instructions which are available from local or remote file systems) are moved to free up memory space. Segments are chosen to be moved if they haven't been referenced recently. When the process next tries to reference this segment a page fault occurs and the process is suspended until the segment is returned to memory. A page fault is normally returned the first time a program is started, as it won't be in memory. It's then paged from the local or remote file system.

Swapping happens under a heavier work load. With swapping the kernel moves all segments belonging to a process to the swap area. The process is chosen if it's not expected to be run for a while. Before the process can run again it must be copied back into physical memory.

5.5 Adding swap space

You can add additional swap space as partitions or as files. Adding them as partitions minimizes overhead as you access the raw partition. To do this under SunOS 4.1.X you would add an entry to **/etc/fstab** similar to the following.

```
/dev/sd1b   swap   swap   rw   0 0
```

You can make a file suitable for use as swap with the ***mkfile*** command found in **/usr/etc** (SunOS 4.1.X) or **/usr/sbin** (SunOS 5.X), e.g.:

```
# mkfile 20m /export/swap/swapfile
```

Under SunOS 4.1.X you add this to the swap area with the ***swapon*** command, i.e.:

```
# /usr/etc/swapon /export/swap/swapfile
```

To automatically add this swap space when booting add the above entry to **/etc/rc.local**.

For SunOS 5.X you would use the ***swap*** command with the **-a** (add) option to add the swapfile, i.e.:

```
# /usr/sbin/swap -a /export/swap/swapfile
```

You can make an entry in **/etc/vfstab** to have this automatically added to the swap space after a reboot.

```
/usr/swapfile    -    -    swap    -    no    -
```

To display the available swap space under SunOS 5.X do the following:

```
# swap -l
swapfile            dev      swaplo   blocks    free
swapfs              -          0      123776    118600
/dev/dsk/c0t3d0s1   32,25      8       66232     50184
/usr/swapfile       -          8       30712     14360
```

To display the total swap space use ***swap -s*** in SunOS 5.X or ***pstat -s*** in SunOS 4.X, e.g.:

```
# swap -s
total: 18780k bytes allocated + 6444k reserved = 25224 used, 30084 available
```

SunOS 5.X allows you to delete swap space at any time. To do this use:

```
# swap -d /export/swap/swapfile
```

When the swap file is no longer in use it will be deleted from the available swap space and will no longer be accessible for swapping.

Swapping to a partition is a little more efficient than swapping to a file, though with the latest OS versions the difference is small. Swap files are convenient to set up, especially if you are only going to use them for a short time period. You can then delete them when the need has expired.

5.6 Setting up a Cache File System

The cache file system, ***cachefs***, is available on Suns starting with Solaris 2.3. It is intended to reduce access time to NFS or slow media (e.g. CDROM) file systems by storing the files on local disk when they're accessed the first time. Subsequent calls for that file will access the cache on the local disk. The original file system is the **back file system** and it's files are the **back files**. The file system used by cachefs is the **front file system** and it's files are the **front files**. You set up a cachefs using all or part of an existing file system, or a new partition. This front file system must be a UFS file system. It has to be writable, as a read-only file system would not allow caching. Also, quotas should not be set on this file system as they interfere with the control mechanisms of cachefs.

You create the cache with the ***cfsadmin*** command, specifying the local cache directory and the resource parameters to use for the cache. You then mount the file system you want cached using the **-F cachefs** option. By default cfsadmin uses the following resource parameters:

Cache Parameter	Default Value
maxblocks	90%
minblocks	0%
threshblocks	85%
maxfiles	90%
minfiles	0%
threshfiles	85%

where ***maxblocks*** sets the maximum number of blocks (in percentage) allowed for cachefs, and ***maxfiles*** sets the maximum number of inodes (in percentage) that can be used by cachefs in the front file system. These percentages refer to total available on the front file system, before reduction due to reserving free space for root-only write access. If the front file system is used for purposes in addition to cachefs you may not be able to achieve these maximum values, as there may be fewer resources available. The ***minblocks*** and ***minfiles*** parameters set minimum values for blocks and files, respectively, and when these minimum values have been reached on the front file system then ***threshblocks*** and ***threshfiles*** will be checked. Cachefs can only claim more than the minimum when the percentage of available resources remaining is greater than the threshold values. If the minimum, maximum, and threshold values are identical, cachefs is allowed to grow to the maximum, so long as the resources are available in the front file system.

So to set up a cachefs file system:

1. Use ***cfsadmin*** to create the cache directory and set the cache file system parameters. The cache directory should not exist prior to executing this command. Create the cache directory and set starting parameters with:

   ```
   # cfsadmin -c -o maxblocks=80,minblocks=30,threshblocks=60 /local/cache
   ```

2. Modify cache parameters with ***cfsadmin.*** You can only increase the cache size. To decrease it you need to remove and recreate the cache. To modify parameters**,** e.g.:

   ```
   # cfsadmin -u -o parameter1=value1,parameter2=value2 /local/cache
   ```

3. Mount the UFS back file system from the command line, e.g.:

```
# mount -F cachefs -o backfstype=nfs,cachedir=/local/cache server:/export/home /home
```

4. Mount a CDROM back file system, from the command line. If the file system is already mounted, as is will be if you're running the volume manager daemon, you need to specify the backpath options, e.g.:

```
mount -F cachefs -o backfstype=hsfs,cachedir=/local/cache,ro,backpath=/cdrom/cd_name \
    /cdrom/cd_name /mount/point
```

5. To mount a file system from /etc/vfstab use entries similar to:

```
#device              device          mount    FS        fsck    mount     mount
#to mount            to fsck         point    type      pass    at boot   options
server:/export/home  /local/cache    /home    cachefs   2       yes       rw,backfstype=nfs
```

6. Display cachefs information, including caching parameters and back file systems, after specifying the cache directory, e.g.:

```
# cfsadmin -l /local/cache
cfsadmin: list cache FS information
    maxblocks       90%
    minblocks       0%
    threshblocks    85%
    maxfiles        90%
    minfiles        0%
    threshfiles     85%
    maxfilesize     3MB
  server:_export_home
  _cdrom_cd_name
```

7. Delete a cached file system with cfsadmin, specifying the **cache_id** (or **all**) and the cache directory. First unmount the directory (**umount**), second delete the cachefs entry (**cfsadmin -d**), third update the resource counts for the cache (**fsck**), e.g.:

```
# umount /home
# cfsadmin -d server:_export_home /local/cache
# fsck -F cachefs /local/cache
```

The ***fsck*** command above will automatically correct consistency problems without user intervention. This is run automatically for you at boot time or when you mount the file system.

To delete all file systems in a cache directory, and the directory itself, use:

```
# cfsadmin -d all /local/cache
/home       auto_home       -fstype=cachefs, cache=/local/cache
```

The ***cachefsstat*** command will report the statistics for the Cache File System. It will display information about the hit rate, consistency checks, and number of modifications to files in the cache.

5.7 XFS (IRIX)

IRIX 6.2 includes the 64-bit journalled file system, **XFS**, as the default file system. It was included in IRIX 5.3 as an option, but EFS was the default file system for that release. It comes with a volume manager, ***xlv***, and supports disk striping, concatenation of disk partitions, and mirroring. It supports CacheFS, AutoFS, and NFS version 3. With 64-bit addresses it will support files and file system of up to 1 TB, under IRIX 5.3 and 9 million TB under IRIX 6.2 for systems supporting 64-bit kernels. XFS does not support disk quotas.

XFS is a **journalled file system**. It logs changes to the inodes, directories and bitmaps to the disk before the original entries are updated. Should the system crash before the updates are done they can be recreated using the log and updated as intended.

XFS uses a **space manager** to allocate disk space for the file system and control the inodes. It uses a **namespace manager** to control allocation of directory files. These managers use **B-tree indexing** to store file location information, significantly decreasing the access time needed to retrieve file information.

Inodes are created as needed and are not restricted to a particular area on a disk partition. XFS tries to position the inodes close to the files and directories they reference. Very small files, such as symbolic links and some directories, are stored as part of the inode, to increase performance and save space. Large directories use B-tree indexing within the directory file to speed up directory searches, additions and deletions.

5.8 File System Quotas

File System quotas are used to control disk space. This allows you to prevent users from monopolizing the disk space. For SunOS 4.1.X to run quotas you need first to provide the support in the kernel. This is done with the line:

```
options   QUOTA
```

in the configuration file. Quotas are only supported on locally mounted disks; quotas will work on NFS mounted file systems, but soft-limit warnings may not always be given. The file should be mounted with the ***quota*** option, e.g. in **/etc/fstab** there might be a line similar to:

```
/dev/sd0h        /home           4.2          rw,quota     1 2
```

A file named **quotas** must be set up at the root directory of each file system for which you wish to have quotas. This is a binary file that can be edited with ***edquota***. It should be owned by root with no access for other users, e.g.:

```
# touch /home/quotas
# chmod 600 /home/quotas
```

The script **/etc/rc** checks for quota consistency under SunOS 4.1.X with the command ***quotacheck***, which examines the disk usage on each file system against the disk quota file. This should be run on quiescent file systems (preferably unmounted). **/etc/rc** then turns the quotas on for each file system with ***quotaon***. This must be run on mounted file systems.

Quotas can be set for each user independently, and can be by either blocks or inodes. The former limits the amount of disk space available for the user, while the latter limits the number of files that the user can have. To set/change a user's limits use the ***edquota*** program. e.g. "***edquota username***". This creates an ASCII file of the current disk quotas for that user and then invokes an editor. After modifying the quotas you leave the editor and ***edquota*** takes this temporary file and merges it with the binary quota file.

The ASCII version of the quotas file might look something like:

```
fs /home   blocks (soft = 4000, hard = 5000) inodes (soft = 0, hard = 0)
```

where a "***0***" means there is no limit.

To check disk usage on a file system you can use the ***quot*** command, e.g.:

```
# quot /dev/rsd0h
    /dev/rsd0h:
97558        chohan
48915        root
41465        anup
31227        frank
14454        bobd
10301        jeffs
 9051        kalal
```

5.9 Miscellaneous useful commands

5.9.1 find command

find is used to search for files, matching a naming pattern, file type, permissions, date of last use, etc. It will search recursively through the directory tree. ***find*** can also execute commands based on the results, e.g.:

```
% find ~ -name src -print
/home/tardis/frank/src
```

Root's crontab file sometimes has an entry similar to the following:

```
30 3 * * * find / -name core -exec rm -f {} \; -o -fstype nfs -prune
```

This instructs the cron program to execute the find command at 3:30 am everyday. The find command searches every directory, except those on file systems of type "**nfs**", for files named "**core**" and removes them.

5.9.2 Removing files

To remove a file you will normally use the remove command, ***rm***, which removes (unlinks) files. Occasionally you may find that rm won't remove a file or directory. This most often happens when a hard link is made to a directory. If you can't find the link you can still remove the directory with the ***unlink*** command. You should then unmount the file system, ***fsck*** it, and remount it. ***fsck*** will update the link count changed by ***unlink***.

5.10 Log files

5.10.1 Daily System Logs

The system accounting programs keep log files of many system activities, including: logins, connect time, user processes, mail activity, error messages, etc. These system log files can grow quite large and need to be truncated occasionally. **/etc/syslog.conf** controls where the messages are sent, usually to files such as: **/var/log/syslog**, **/var/adm/messages**, or **/var/adm/SYSLOG** for system startup and system error messages, and **/var/spool/mqueue/syslog** for mail logs. (The Ultrix error report formatter, ***uerf***, puts data in **/usr/adm/syserr/syserr.hostname**). Daily and monthly process accounting information is kept in **/var/adm** and **/var/adm/acct/[nite,fiscal,sum]**.

Crash files, placed there by ***savecore***, are usually put in either **/var/crash/hostname** or **/var/adm/crash/hostname**.

5.10.2 Process Accounting

Process accounting information is contained in the file **/var/adm/pacct**. Support for system accounting must be built into the kernel for SunOS 4.1.X with "**options SYSACCT**" and "**pseudo-device sysacct**" lines in the configuration file. It's turned on with the ***accton*** command in **/etc/rc**.

A summary of accounting information is kept in the file **/var/adm/savacct**.

The program, ***/usr/ucb/lastcomm***, is used to show all commands run since accounting was started (**/var/adm/pacct** was created).

Statistics on each process, e.g. number of times called, CPU minutes, total elapsed time, etc., is given by the ***/usr/etc/sa*** command. It gets this information from **/var/adm/pacct** and puts it into **/var/adm/savacct**.

A record of all logins and logouts is kept in **/var/adm/wtmp.**

The record of current users is kept in **/etc/utmp**.

To list all user logins since **/var/adm/wtmp** was created use the ***/usr/ucb/last*** command, e.g.:

```
% last
amit     ttyp0  ivy               Tue May 15 16:54 - 16:55  (00:01)
chohan   ttyp7  galifrey.acs.oh   Tue May 15 13:04 - 17:05  (04:00)
amit     ttyp7  slippry1.acs.oh   Tue May 15 12:49 - 12:51  (00:01)
chohan   ttyp8  charm.acs.ohio-   Tue May 15 12:19 - 12:21  (00:01)
```

To show the connect time of all users since **/var/adm/wtmp** was created use ***/usr/etc/ac*** or ***/usr/lib/acct/acctcon***.

The file **/var/adm/lastlog** keeps the last login record for each user.

The general system message and error log file is **/var/adm/messages** and **/var/log/syslog**, as specified in **/etc/syslog.conf**.

The script, **/usr/lib/newsyslog**, is run periodically by ***cron*** to clean up **/var/adm/messages** and **/var/log/syslog**. You should modify this script, or write your own, to properly update all your log files.

CHAPTER 6 Startup and Shutdown

6.1 Booting

During the boot process the operating system is loaded into memory and executed. After doing diagnostic checks the system reads in the boot program from the disk, or other boot device. The boot program locates the kernel and loads it into memory. When the kernel is executed it does initial checks on system hardware resources and attempts to initialize the devices. The kernel then starts up a few processes, including ***init***, which executes the initialization scripts for the system.

6.1.1 EEPROM on CPU board

When you boot the EEPROM and the operating system it takes you through the following steps.

1. Self-test diagnostics, memory
2. Display identification- model, hostid, ethernet address
3. Probe bus for the boot device - SCSI, Network, etc.
4. PROM reads in the boot block
5. ***bootblk*** reads in the boot program
6. Boot program reads in the kernel
7. Kernel initializes the systems and starts the ***init*** process
8. ***init*** reads ***/etc/inittab*** (SunOS 5.X) and executes the ***RC*** scripts

When booting from disk the PROM finds the system boot block at sectors 1-15 of the boot partition. The bootblock program, ***bootblk***, reads in the boot program, ***/boot*** (SunOS 4.X). For SunOS 5.X ***bootblk***, the generic part of the boot program, reads in the file-specific part of the boot program: ***/ufsboot*** for diskfull boots, or ***/inetboot*** for diskless boots; these use the device driver on the PROM or on the SBus F-code PROM, so the boot block no longer contains the actual location of the disk block where the boot program resides. Under SunOS 5.X ***bootblk*** can read the ***ufs*** file system to locate the boot program, ***/ufsboot***.

The boot program then locates the ***kernel*** and passes control to it.

6.1.2 Operating System

6.1.2.1 Kernel

The ***kernel*** is loaded, ***/vmunix*** (SunOS 4.X) or ***/kernel/unix*** (SunOS 5.0-5.4) or ***/unix*** (SGI), and control passes to the operating system. Solaris 2.5 and above (SunOS 5.5+) has both a generic, platform-independent part (***/kernel/genunix***) and a core, or platform-specific part (***/platform/'uname -m'/kernel/unix***) of the kernel. These are combined to form the running kernel.

When the ***kernel*** starts it outputs information about its size and history, probes the bus to confirm the devices, those that it can't contact are ignored, it identifies the root, swap, and dump devices, starts up programs to manage physical memory and flush the kernel memory buffers, and then it invokes ***/sbin/init***.

6.1.2.2 sched

SunOS 5.X uses the real-time scheduler, ***sched***, which is started as process 0. This can be used to set priority for real-time processes so that they can be given "immediate" access to the kernel. The latency time on a SparcStation 2 is less than 1 millisecond on a lightly used machine and a maximum of 2 milliseconds with an arbitrary number of processes running.

6.1.2.3 swapper

SunOS 4.X uses the ***swapper*** daemon, process 0, to manage virtual memory. It moves processes from physical memory to swap space when more physical memory is needed. SunOS 5.X uses the swap file system, **swapfs**, to manage virtual memory.

6.1.2.4 update and fsflush

When a program makes a change to the file system it first writes to the in-core buffer in the kernel. The disk write normally occurs later, and is handled asynchronously. The user process continues without waiting for this to happen. The kernel initializes the program, ***update*** (SunOS 4.X, started by ***/etc/rc***) or ***fsflush*** (SunOS 5.X, process 3), that periodically (the default is every 30 seconds) flushes the in-core memory buffers to the disk by calling the ***sync*** command. This helps to minimize damage in the event of a crash.

6.1.2.5 pagedaemon and pageout

When a page of virtual memory is accessed the kernel page table is checked to determine if the page is currently in physical memory. If it is not, a page fault is registered and the daemon, ***pagedaemon*** (SunOS 4.X), or ***pageout*** (SunOS 5.X), both as process id 2, is used to bring the page into memory from the disk. If necessary, the page daemon moves a page of physical memory to the swap device to make room for the new page in physical memory, and updates the page table.

6.1.2.6 Init

/sbin/init starts and forks into the background to run forever (process 1; it must always be running). ***init*** then invokes the run control (**RC**) scripts (in **/etc** for SunOS 4.X, **/sbin** for SunOS 5.X) to perform system checks and start the daemon processes. ***init*** runs the scripts **/etc/rc.boot** and **/etc/rc.ip** (SunOS 4.1.X) or **/sbin/rcS** (SunOS 5.X) which runs ***fsck***. It then continues on with the boot sequence to run ***/etc/rc***, **/etc/rc.single**, and **/etc/rc.local** (SunOS 4.X) or **/sbin/rc2** and **/sbin/rc3** (SunOS 5.X). The System V version of ***init*** reads **/etc/inittab** to determine the actions to take at various run levels.

IRIX 5.X has its **RC** scripts in **/etc** and runs **bcheckrc**, **brc**, **rc2**, and **rc3**, as determined by **inittab**, as the appropriate run levels are reached. It uses the files in **/etc/config** to determine which daemons to start and the options to use for system services.

Digital UNIX has its **RC** scripts in **/sbin** and runs **bcheckrc**, **rc2**, and **rc3**, via **/etc/inittab**, as it moves through the various run-levels. The **RC** scripts source the resource definitions file, **/etc/rc.config** to determine the values for certain system parameters and whether or not to start particular services.

6.2 Run Levels (SunOS 5.X, IRIX 5.X)

Solaris 2 and IRIX 5.X have eight run levels, 0-6,s or S. The following table identifies the modes for these run levels.

TABLE 6.1 **System Run Levels**

Run Level	Function	Command
0	PROM monitor level (power-down)	init 0, shutdown -i0, halt
1,S,s	Single-user mode	init 1, shutdown -i1
2	Multi-user mode, NO resources shared	init 2, shutdown -i2
3	Multi-user mode, resources shared	init 3, shutdown -i3
4	Alternative multi-user mode (not currently used)	
5	Halt and software Poweroff the system	init 5, shutdown -i5
6	Halt and reboot to default state	init 6, shutdown -i6, reboot

You can determine the current run state with the command

```
# who -r
.    run-level 3  Feb 22 08:54   3   0  S
```

Additionally, ***init*** responds to the **q** or **Q** run levels, which cause ***init*** to reread **/etc/inittab**.

Digital UNIX has run levels **0,2,3,q,s**.

6.3 /etc/inittab (SunOS 5.X, IRIX 5.X, Digital UNIX)

Init reads **/etc/inittab** for the **initdefault** entry, which should be set to run level 3. ***Init*** then executes the scripts for entries with **sysinit** in the action field, and then for any entries with **3** in the action field. For the former it will execute **/sbin/autopush** and **/sbin/rcS**. For the latter it will execute **/sbin/rc2**, **/sbin/rc3**, ***/usr/lib/saf/sac***, and ***/usr/lib/saf/ttymon***. The **RC** scripts will execute the scripts in the directories **/etc/rc2.d** and **/etc/rc3.d**, respectively.

There should be at least one entry in **inittab** for each run level. The scripts in the **/etc/rc#.d** directories begin with either the letter **K** or **S**. When these scripts are executed by the **/sbin/rc#** script first the **K** (kill) files are run, then the **S** (start) files, to kill and start the various daemons needed for that run level. These scripts have names of the form:

```
[K,S][0-9][0-9]filename[0-99]
```

and are executed in ASCII sort order.

To start the daemons the **RC** scripts check for the existence of the **/etc/rc#.d** directory, then for any files beginning with "**S**" in that subdirectory, and then they execute those scripts with the "**start**" option.

The appropriate lines in the **RC** file, e.g. those in **/sbin/rc2**, to start the scripts beginning with "**S**", are:

```
if [ -d /etc/rc2.d ]
then
for f in /etc/rc2.d/S*
{
        if [ -s ${f} ]
        then
                case ${f} in
                        *.sh)       . ${f} ;;                   # source it
                        *)          /sbin/sh ${f} start ;;      # sub shell
                esac
        fi
}
fi
```

Then in **/etc/rc2.d** you would have scripts such as the **K20lp** script and the **S80lp** script to stop and start, respectively, the lineprinter scheduler. These scripts are actually identical and are run with either the **stop** or **start** options to cause the desired effect. Some of the scripts are symbolic links to files in the **/etc/init.d** directory. The **K** and **S** files for a service don’t have to be in the same **RC** directory. You might stop a service when entering run level 2, and start it when entering run level 3.

A typical script might look something like (substitute your daemon name for sample_daemon):

```
#!/bin/sh
# start up sample_daemon, installed by FGF, 04/12/96
#
case "$1" in
'start')
    if [ -x /opt/local/sbin/sample_daemon ]; then
        /opt/local/sbin/sample_daemon && echo "Starting sample_daemon ... "
    fi
    ;;
'stop')
    pid=`/usr/bin/ps -e | /usr/bin/grep sample_daemon |  /usr/bin/sed -e 's/^  *//' -e 's/ .*//'`
    if [ "${pid}" != "" ]; then
        echo "Stopping sample_daemon "
        /usr/bin/kill ${pid}
    fi
    ;;
```



```
*)
    echo "Usage: /etc/init.d/sample_daemon { start | stop }"
    ;;
esac
exit 0
```

To modify the run states you can write your own startup scripts, install scripts in **/etc/init.d** and make symbolic links to them in the **/etc/rc#.d** directory (with the proper K,S names), or add entries to **/etc/inittab**.

Your **/etc/inittab** file has entries of the form:

id:run-state:action:process

where:

- **id** — 1 or 2 characters to identify the entry
- **run-state** — the run level(s) for which this entry will be applicable. (If no run level is specified then all levels, 0 through 6, are assumed.)
- **action** — how the process will be treated by ***init.*** Valid keywords are:
  - **respawn** — start the process if it doesn't exist, restart it if it dies
  - **wait** — start the process and wait for it to terminate
  - **once** — start the process when entering the run level, but don't wait for it to complete and don't restart it if it dies
  - **boot** — process the entry only on boot up
  - **bootwait** — process the first time ***init*** moves from single- to multi-user state after a boot and wait for it to complete
  - **powerfail** — execute when ***init*** receives a power-fail signal, SIGPWR
  - **powerwait** — execute when a power-fail signal is received and wait for it to complete
  - **off** — send a SIGTERM to the process followed 5 seconds later by a SIGKILL to forcibly terminate it
  - **ondemand** — same as respawn
  - **initdefault** — process when ***init*** is initially invoked - sets the default run-level to enter
  - **sysinit** — process this entry before accessing the console and wait for it to complete
- **process** — the command or script to be executed; any legal "sh" syntax is valid

The ***init*** process first searches **/etc/inittab** for **initdefault** entries to determine the run-level. Next, **sysinit** entries are executed. Following this all process whose **run-state** matches the initdefault value are executed. Entries are processed starting from the top of the table and working down.

A typical **inittab** might look similar to:

```
ap::sysinit:/sbin/autopush -f /etc/iu.ap
fs::sysinit:/sbin/rcS                          >/dev/console 2>&1 </dev/console
is:3:initdefault:
p3:s1234:powerfail:/sbin/shutdown -y -i0 -g0 >/dev/console 2>&1
s0:0:wait:/sbin/rc0 off                        >/dev/console 2>&1 </dev/console
s1:1:wait:/sbin/shutdown -y -iS -g0  >/dev/console 2>&1 </dev/console
s2:23:wait:/sbin/rc2                           >/dev/console 2>&1 </dev/console
s3:3:wait:/sbin/rc3                            >/dev/console 2>&1 </dev/console
s5:5:wait:/sbin/rc5 ask                        >/dev/console 2>&1 </dev/console
s6:6:wait:/sbin/rc6 reboot                     >/dev/console 2>&1 </dev/console
of:0:wait:/sbin/uadmin 2 0                     >/dev/console 2>&1 </dev/console
fw:5:wait:/sbin/uadmin 2 2                     >/dev/console 2>&1 </dev/console
RB:6:wait:/sbin/sh -c ‘echo “\nThe system is being restarted.”’ >/dev/console 2>&1
rb:6:wait:/sbin/uadmin 2 1                     >/dev/console 2>&1 </dev/console
sc:234:respawn:/usr/lib/saf/sac -t 300
co:234:respawn:/usr/lib/saf/ttymon -g -h -p “‘uname -n‘ console login: “ -T sun -d /dev/console -l
    console -m ldterm,ttcompat
```

To cause init to reread **inittab** specify ***q*** or ***Q*** to the ***init*** (or ***telinit***) command, e.g.

init q -or- telinit q

This is similar to doing a “kill -HUP 1” under SunOS 4.X.

6.3.0.1 RC scripts

SunOS 4.1.X uses the **RC** scripts **rc.boot**, **rc.ip**, **rc**, **rc.single**, and **rc.local**, all in **/etc**. The major functions performed by these scripts are listed in the next table.

TABLE 6.2 SunOS 4.1.X RC Scripts

RC Script	Functions
rc.boot	bring up the network, matching hostnames with the interfaces rc.ip set the default route (required for a diskless client to mount /usr) mount /usr (read only) fsck local file systems
rc.ip	bring up the network
rc	rc.single mount local file systems check quotas rc.local add additional swap start lpd
rc.single	remount / and /usr read/write fix up mtab clean up /etc/ld.so.cache and /etc/utmp use tzsetup to set the timezone in the kernel load the keyboard translation table for the current keyboard

TABLE 6.2 **SunOS 4.1.X RC Scripts**

RC Script	Functions
rc.local	start the portmapper
	check for .UNCONFIGURED, if there reconfigure the system
	run tzsetup to set the timezone
	set the domainname if running NIS
	if an NIS server run ypserv
	if the NIS master run ypxfrd
	if an NIS client run ypbind
	run the RPC keyserver, keyserv
	set the netmask and broadcast for the network interfaces
	set the default route again
	diskless clients synchronize time-of-day with their server
	if there's no default route run the route daemon
	if specified mount /tmp on swap
	mount NFS files
	if a name server run the named daemon
	start the block I/O daemon, biod
	if an NFS server start the nfsd daemons
	clean up /etc/motd
	start the system log daemon, syslogd
	if specified check for a crash dump and save it
	initialize any specialized hardware
	start any local daemons, e.g. sendmail
	if specified, export NFS file systems
	if a diskless boot server run rpc.bootparamd
	start the file status monitor and locking daemons
	any other locally supplied calls

SunOS 5.X uses the **RC** scripts **rcS**, **rc0**, **rc1**, **rc2**, **rc3**, **rc5**, and **rc6** in **/sbin**. These start or stop services defined in the scripts contained in the **/etc/rc#.d** directories. The **S** scripts are run during startup, lower through higher run-level. The **K** scripts are run during shutdown, higher through lower run-level. These **RC** scripts provide the functions listed in the next table.

TABLE 6.3 SunOS 5.X RC Scripts

RC Script	Functions
/sbin/rcS	run the scripts in /etc/rcS.d
/etc/rcS.d/S30rootusr.sh	configure the network, match hostnames to interfaces set the default route mount /usr (read only)
/etc/rcS.d/S33keymap.sh	loads the keyboard mappings
/etc/rcS.d/S35cacheos.sh	configures the devices when running cachefs
/etc/rcS.d/S40standardmounts.sh	add physical swap space check and remount / and /usr read/write
/etc/rcS.d/S50/S50drvconfig	configure the /devices directory
/etc/rcS.d/S60devlinks	configure the /dev directory
/etc/rcS.d/S70buildmnttab.sh	mount file systems for single user mode
/sbin/rc0	run the scripts in /etc/rc0.d, kill all processes, sync the file systems, unmount all partitions, bring the system down
/etc/rc0.d/K10dtlogin	initiate the CDE tasks
/etc/rc0.d/K20lp	stop the line printer daemon
/etc/rc0.d/K42audit	stop the audit daemon
/etc/rc0.d/K47asppp	stop the PPP daemon
/etc/rc0.d/K50utmpd	stop the utmp daemon
/etc/rc0.d/K55syslog	shutdown the system log daemon
/etc/rc0.d/K57sendmail	stop the sendmail daemon
/etc/rc0.d/K66nfs.server	kill the nfs, mount, bootparam and rarp daemons
/etc/rc0.d/K68rpc	kill rpc daemons
/etc/rc0.d/K69autofs	stop the automount daemon
/etc/rc0.d/K70cron	shutdown cron
/etc/rc0.d/K75nfs.client	kill lockd, statd, and the automounter
/etc/rc0.d/K76ncsd	kill ncsd daemons
/etc/rc0.d/K85rpc	kill rpc daemons
/sbin/rc1	run the scripts in /etc/rc1.d, kill all processes, unmount all partitions, leave the system in single-user mode
/etc/rc1.d/K10dtlogin	initiate the CDE tasks
/etc/rc1.d/K42audit	stop the audit daemon
/etc/rc1.d/K47asppp	stop the PPP daemon
/etc/rc1.d/K50utmpd	stop the utmp daemon

TABLE 6.3 **SunOS 5.X RC Scripts**

RC Script	Functions
/etc/rc1.d/K55syslog	stop syslog
/etc/rc1.d/K57sendmail	stop sendmail
/etc/rc1.d/K65nfs.server	shutdown NFS services
/etc/rc1.d/K67rpc	shutdown RPC services
/etc/rc1.d/K68autofs	stop the automount daemon
/etc/rc1.d/K70cron	shutdown cron
/etc/rc1.d/K76ncsd	kill ncsd daemons
/etc/rc1.d/K80nfs.client	unmount all NFS file systems
/etc/rc1.d/S01MOUNTFSYS	mount all local file systems
/sbin/rc2	set the timezone run the scripts in /etc/rc2.d
/etc/rc2.d/K20lp	shutdown the line printer
/etc/rc2.d/K60nfs.server	shutdown NFS services
/etc/rc2.d/S01MOUNTFSYS	mount all local file systems
/etc/rc2.d/S05RMTMPFILES	clean up /tmp and /var/tmp
/etc/rc2.d/S20sysetup	print the system configuration if specified save the core image
/etc/rc2.d/S21perf	enable system performance accounting
/etc/rc2.d/S30sysid.net	if /.UNCONFIGURED exists reconfigure the system
/etc/rc2.d/S47asppp	start the PPP daemon
/etc/rc2.d/S69inet	configure the default route set the domainname
/etc/rc2.d/S70uucp	clean up uucp locks
/etc/rc2.d/S71rpc	start rpc start NIS(+) daemons
/etc/rc2.d/S71sysid.sys	if /.UNCONFIGURED exists reconfigure the system
/etc/rc2.d/S71yp	start up NIS (up) services
/etc/rc2.d/S72autoinstall	if /AUTOINSTALL exists re-install the OS
/etc/rc2.d/S72inetsvc	set the default interface for multicasting start inetd if a name server start named
/etc/rc2.d/S73nfs.client	start lockd and statd mount NFS file systems
/etc/rc2.d/S74autofs	start the automount daemon
/etc/rc2.d/S74syslog	start the system log daemon

TABLE 6.3 **SunOS 5.X RC Scripts**

RC Script	Functions
/etc/rc2.d/S75cron	start cron
/etc/rc2.d/S76nscd	start up the name service cache daemon
/etc/rc2.d/S80PRESERVE	save edit files in /usr/preserve
/etc/rc2.d/S80lp	start the line printer scheduler
/etc/rc2.d/S88utmpd	start up utmpd to clean up utmp entries
/etc/rc2.d/S88sendmail	start sendmail
/etc/rc2.d/S92rtvc-config	set SunVideo device permissions
/etc/rc2.d/S92volmgt	start the volume management daemon
/etc/rc2.d/S93cacheos.finist	final cachefs settings
/etc/rc2.d/S99audit	start the audit daemon
/etc/rc2.d/S99dtlogin	automatically start the CDE login window on the console
/sbin/rc3	run the scripts in /etc/rc3.d
/etc/rc3.d/S15nfs.server	start the processes required for remote file sharing
/etc/rc3.d/S20sample_daemon	start your daemon
/sbin/rc5	run /sbin/rc0, kill off all process, unmount all filesystems
/sbin/rc6	run /sbin/rc0, kill off all process, unmount all filesystems

6.3.0.2 Fsck runs

fsck checks file systems for internal consistency.

6.3.0.3 The daemon processes start

The RC scripts start the network daemons and mount remote file systems.

6.3.0.4 Init starts multi-user mode

At the conclusion of the RC scripts ***init*** starts multi-user mode and initiates ports and allows other users to login.

6.4 Sun PROM

6.4.1 Bootstrap Procedures

For Sun Microsystems hardware you can interact with the PROM monitor at any time by holding down the **STOP** key (the L1 key on older keyboards) and pressing the "**a**" key. If you're using a terminal keyboard you can use the "**break**" key.

The PROM monitor boot commands come in two forms: "old" style, with a "**>**" prompt; and "new" style with a "**ok**" prompt. The "new" style came in about the time the SPARC chip was first introduced, and the newer PROMs have both. On the latter you can get to the "new" style from the "old" by typing "**n <return>**".

The general form to specify the boot device is:

>b device(controller#,unit#,file#)pathname args

or

ok boot device(controller#,unit#,file#)pathname args

where the **controller#** is the host bus adapter # (always 0 if you only have one SCSI bus), the **unit#** is the tape drive or disk drive #, and **file#** is the partition on the drive or file on the tape. The **pathname** is the path to the kernel, and possible arguments include:

s	boot to single user mode only
a	ask for configuration information, i.e. root and swap devices and system file
r	reconfigure the system based on currently connected hardware devices (Solaris 2.X only)

At the "**>**" prompt the boot command is "**b**", while at the "**ok**" prompt it is "**boot**".

The default boot device can be configured in the EEPROM to allow:

>b -or- **ok** boot

Single user:

>b -s -or- **ok** boot -s

Boot from network:

>b le()

Boot a specific kernel from disk, e.g. for the kernel, vmunix:

>b sd(0,0,0)vmunix

6.4.2 Sun Boot PROM search sequence

The legal boot devices known by the EEPROM can be determined by

>b ?

This also gives the order the devices are polled; these devices may or may not be present.

On newer systems you can determine the SCSI devices on the system bus with:

ok probe-scsi

and on all SCSI busses with:

ok probe-scsi-all

Newer machines use the OpenBoot PROM with device names that specify the hardware controlling the device, as shown in Chapter 3, but also accept shorthand notation to specify the disk devices. In the latter "**disk**" represents the disk device on the main SCSI bus with SCSI target ID of 3, "**disk1**" has target ID 1, "**disk2**" has target ID 2, "**disk3**" has target ID 0, and "**cdrom**" has target ID 6.

PROM environmental variables can be read with the ***printenv*** command and set with ***setenv*** command when at the "**ok**" prompt.

To stop the machine and perform a manual boot type "**STOP-A**" then specify the device, e.g.:

>b st() -or- **ok** boot cdrom -or- **ok** boot disk

When the boot operations are completed the kernel is loaded and control passes to it.

6.5 SGI Indy PROM

When you turn on the power to your Indy workstation the System Startup Notifier is displayed and you're given the option to "**Stop for Maintenance**". Click on the icon or press <**Esc**> to display the System Maintenance Menu. Click on "**Enter Command Mode**" or type **5** at the menu. You're then presented with the "**>>** " prompt. From this menu you can specify a boot device, list the PROM environment variables with ***printenv*** or set them with ***setenv***, etc. To specify a boot device and disk partition or kernel, other than the defaults, use the form:

>> boot dksc(controller#,unit#,file#)pathname

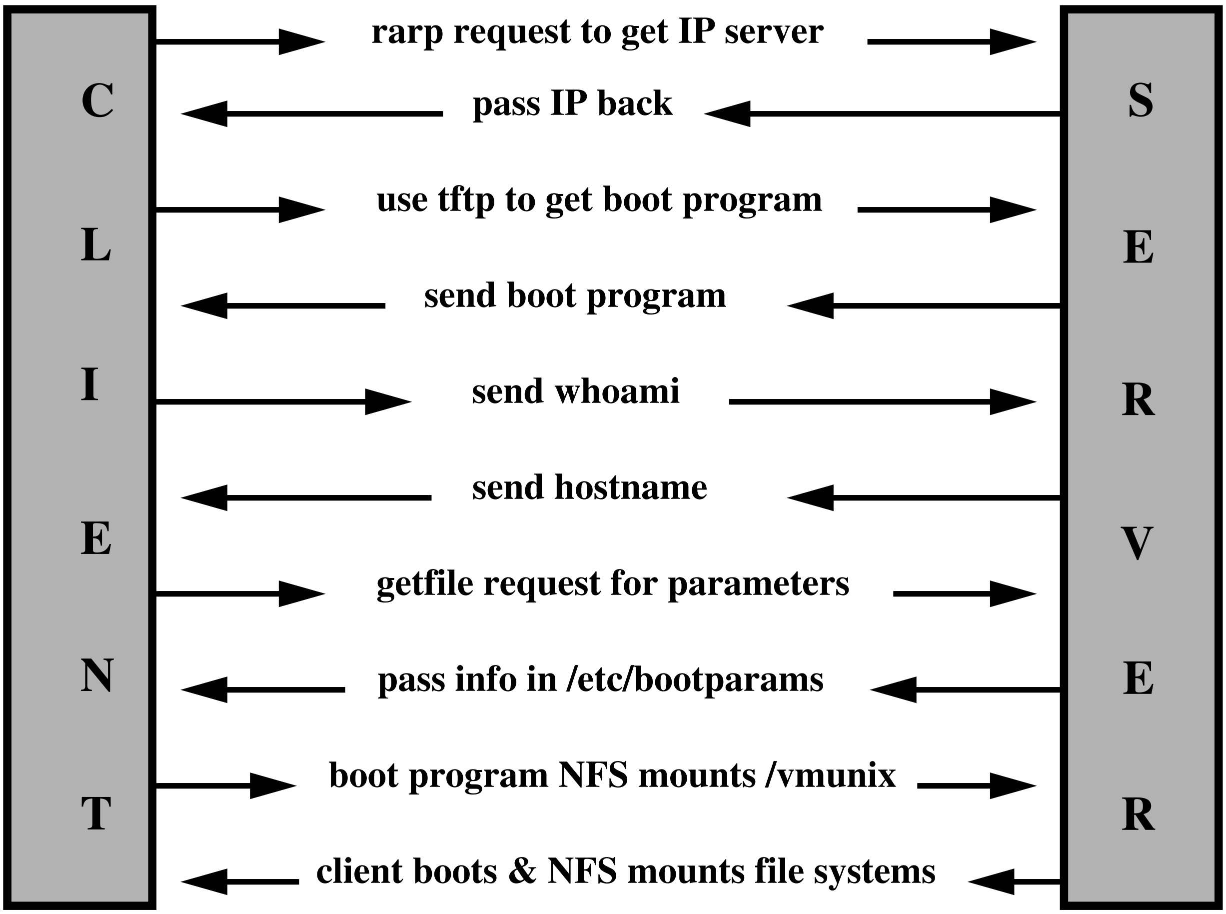
where the **controller#** is the host bus adapter # (always 0 if you only have one SCSI bus), the **unit#** is the disk drive #, and **file#** is the partition on the drive, and **pathname** is the path to the kernel.

6.6 Diskless Workstations

Diskless workstations need help in booting. To do this they first need to:

- Determine who they are
- Locate their boot server
- Locate their kernel
- Mount file systems from the server

FIGURE 6.1 **Client-Server Boot Exchange**



1. Boot PROM sends reverse address request packet (***rarp***) onto the network with its Ethernet address to find out who knows its Internet (IP) address (**/etc/ethers**: ethernet<=>hostname).
2. Server running reverse address resolution protocol daemon (***/usr/etc/rarpd***) answers with the IP address of the client (**/etc/hosts**: IP<=>hostname).
3. Client PROM uses trivial file transfer program (***tftp***) to load the boot program.
4. Server sends the ***boot*** program to the client.
5. Boot program issues ***whoami*** request to get the clients hostname.
6. Server looks up ***hostname*** from IP address and responds to the client (**/etc/hosts**).
7. Boot program issues a ***getfile*** request to determine boot parameters (**/etc/bootparams**: client root and swap locations).
8. Server running the boot parameter daemon (***/usr/etc/rpc.bootparamd***) responds with **/etc/bootparams** information.
9. Boot program NFS mounts the root file system, loads the kernel (***/vmunix***, for SunOS 4.X), and transfers control to the kernel.
10. The client proceeds to boot normally and NFS mounts other file systems.

6.7 Shutdown

Shutdown can be initiated at the PROM or by one of the shutdown programs of the operating system.

6.7.1 PROM

Stop-A -or- L1-A -or- **<Break>**

This takes you to the PROM level, but does NOT warn users about the shutdown. Once at this level you can execute one of the following PROM commands.

>c	- continue after system abort.
>g0 (Sun3)	- force system crash and sync the disks.
>sync (Sun4)	- force system crash and sync the disks.
ok sync (Sun4c/4m/4u)	- force system crash and sync the disks.

In addition to flushing the system memory buffers to disk, the PROM commands that sync the system also dump the in-core kernel and memory pages to the high end of swap on the disk. This can be useful if you want to debug the reason for a system crash. After rebooting you can use the ***savecore*** program to copy those areas of swap to the file system. Keep in mind, though, that these files can be quite large, especially if you have large amounts of physical memory. Generally, it takes a specialist in kernel architecture with access to the source code to accurately analyze these files.

6.7.2 Shutdown programs

Generally, you want to run a system program that will warn users and perform an orderly shutdown.

6.7.2.1 shutdown

This is an automated procedure to warn users and then brings the system down. When complete you can then power off the system or reboot. ***shutdown*** must be run as root. As an example, to shutdown the system in 10 minutes followed by a reboot, for SunOS 4.X, execute:

shutdown -r +10 - reboot in 10 minutes.

SunOS 5.X, which uses a different ***shutdown*** program. This program is a shell script to take you to the desired run state and is located in **/usr/sbin**. You can specify a grace period (***-g***) in seconds, a run level (***-i***), and auto-confirmation of answers (***-y***). So to halt the system in 2 minutes while warning the users of the impending shutdown execute:

shutdown -y -g120 -i0

6.7.2.2 halt/fasthalt and reboot/fastboot

halt and ***fasthalt*** synchronize the disk and then shutdown, but they do NOT warn users. ***fasthalt*** creates a file, **/fastboot** (SunOS 4.X only). If it exists **/etc/rc** skips the fsck step when booting. To halt without syncing the file system type "***halt -n***". ***reboot*** and ***fastboot*** are similar to ***halt***/***fasthalt***, but then they immediately reboot the system. ***fastboot*** also creates the file **/fastboot** (SunOS 4.X only).

6.7.2.3 Kill init

kill -TERM 1

This kills the ***init*** process. Since ***init*** must be running this will panic the system and force a reboot. It is **NOT** recommended.

6.7.2.4 Synchronize the disks

sync synchronizes the disks by updating the super block and forcing changed blocks to the disk. This should be called before the processor is halted abnormally. Any user can run ***sync*** at any time. You can also sync the system from the PROM, if necessary, as shown above.

6.8 Crashes

6.8.1 Panics and their causes

Some causes of system panics are:

- Memory errors
- Bugs in the Operating System
- Disk write errors - bad blocks on disk

The system logger, ***syslogd***, writes a log of system error messages in **/var/adm/messages** or **/var/log/syslog**. These can be helpful in tracking down the cause of a system problem or kernel panic.

For more information during recurrent crashes turn on ***savecore*** in **/etc/rc.local** (SunOS 4.X) or **/etc/rc2.d/S20sysetup** (SunOS 5.X). When enabled the system will save a core image of memory in the file /var/crash/‘uname -n‘**/vmcore.#** and the kernel’s namelist in the same directory in the file **vmunix.#** (SunOS 4.X) or **unix.#** (SunOS 5.X) during the reboot. These files can then be analyzed for the causes of the system panic. The appropriate lines in rc.local are:

```
# Default is to not do a savecore
# mkdir -p /var/crash/`hostname`
# echo -n 'checking for crash dump... '
# intr savecore /var/crash/`hostname`
```

6.8.2 Crash recovery

Reboot the system. Examine the console output for information that may help determine the cause of the problem. If necessary boot in single user mode:

```
>b -s
```

or with an alternate kernel, e.g.:

```
>b sd()vmunix.gen
```

For best recovery after a crash have good backups and perform a file system check, ***fsck***, when booting. If necessary reboot in single user mode and run ***fsck*** manually on an unmounted file system.

6.8.3 IRIX

At the "**>>** " PROM **Command Mode** prompt you can specify an alternate disk, disk partition, or kernel to boot from in the form:

```
>> boot dksc(0,1,7)unix.save
```

where **0** is the disk controller, **1** is the SCSI target number of the disk, **7** is the disk partition, and ***unix.save*** is the alternate kernel.

CHAPTER 7

Operating System Installation

7.1 Suninstall

Suninstall is a set of programs and files that are used to install SunOS, other Sun software, determine mount points, and resize disk partitions. The files are located in **/usr/etc/install** (SunOS 4.1.X) or **/usr/sbin** (SunOS 5.X).

7.1.1 Features of suninstall

The **/usr/etc/install/files** directory contains a record of the installation and any errors are kept in suninstall.log. Suninstall can be interrupted at any point and a record of current settings will be kept if run from SunOS. If run from MINIUNIX then the data is destroyed when you reboot. It supports various Sun architectures, and can build heterogeneous servers, standalone, and dataless workstations.

7.1.2 Installation of SunOS on a standalone workstation

1. Load the bootstrap program from cdrom/tape.
2. Load the standalone copy program from cdrom/tape.
3. Copy MINIUNIX from cdrom/tape to the disk swap partition (SunOS 4.X), or the root partition to memory (SunOS 5.X).
4. Start suninstall.
5. Load the requested cdrom/tapes(s).
6. Reboot the system.

7.2 SunOS 4.1.X

7.2.1 Boot CDROM contents:

The SunOS 4.1.4 (Solaris 1.1.2) CDROM, which includes the latest BSD version of SunOS, allows for both a full install and an upgrade from earlier versions of SunOS 4.1.X. The CD conforms to the hsfs file system. You can mount it and look at it under SunOS. If you do this you will see that the top level directory contains the subdirectories: export, patches, and sunupgrade. Under /export there are the subdirectories: exec and share. /export/share contains the manual pages. /export/exec holds the necessary executables to install the OS. The files are in uncompressed tar format. Under **/export/exec** we have (in 512 byte blocks):

/export/exec:

4	kvm/
512	proto_root_sunos_4_1_4*
4	sun4_sunos_4_1_4/

/export/exec/kvm:

4	sun4_sunos_4_1_4/	(kernel contained on boot file 1 on the CDROM)
4	sun4c_sunos_4_1_4/	(kernel contained on boot file 2 on the CDROM)
4	sun4m_sunos_4_1_4/	(kernel contained on boot file 3 on the CDROM)

/export/exec/kvm/sun4c_sunos_4_1_4: (similar sub-directories for Sun4 and Sun4m)

10400	kvm*
14000	miniroot_sun4c*
11216	sys*
13	xdrtoc*

/export/exec/sun4_sunos_4_1_4:

5856	debugging*	8128	demo*
6288	games*	3568	graphics*
1952	install*	2096	networking*
15264	openwindows_demo*	19040	openwindows_fonts*
46400	openwindows_programmers*	67024	openwindows_users*
1808	rfs*	640	security*
2752	shlib_custom*	1024	sunview_demo*
3680	sunview_programmers*	5328	sunview_users*
8016	system_v*	1424	text*
96	tli*	15376	user_diag*
57888	usr*	1216	uucp*
11920	versatec*		

/export/share:

4	sunos_4_1_4:	
	14992	manual

Since the files are tar formatted archives, so you can readily retrieve any files later on, as needed.

7.2.2 Before starting installation:

1. Read the "**SunOS 4.1.X Release & Install**" manual.
2. Read the "**READ THIS FIRST**" document provided.
3. Decide or obtain the information necessary to complete the install, including:

Hostname	- hostname.dept.ohio-state.edu- required if on SONNET
Ethernet address	- prints during the EEPROM self-test.
Internet address	- required if on the OSU network, SONNET; obtain from UTS.
Machine architecture	- Sun3/Sun3x/Sun4/Sun4c/Sun4m
Partitions and sizes	- disk partitions

Mount points - mount points need to exist before partitions can be mounted; exceptions are partition a (/) and partition b (swap).

Software desired - if disk space is available you can load everything, otherwise delete items you're unlikely to use.

7.2.3 Installation Example

Install the miniroot: Insert boot tape and at the PROM prompt type:

```
>b tape          -or-     ok boot tape
or
>b st()          for old Sun4c/Sun4/Sun3
```

or boot cdrom

```
>b cdrom         -or-     ok boot cdrom
or
>b sd(0,6,2)     for old Sun4c
>b sd(0,30,1)    for Sun4
```

After successfully installing the miniroot you'll be presented with the choice;

```
What would you like to do:
  1 - install SunOS mini-root
  2 - exit to single user shell
Enter a 1 or 2:
```

If you enter 1 you'll be given the opportunity to select, format, and relable the disk. Disks purchased from Sun are formatted at the factory so you shouldn't need to reformat them. You may wish to relable them to change partition sizes.

Once the mini-root is installed on your disk's swap space you'll be presented with:

```
Mini-root installation complete.
What would you like to do?
  1 - reboot using the just-installed miniroot
  2 - exit into single user shell
Enter a 1 or 2:
```

Exit to single user mode and start the installation:

```
# suninstall
…        (many screens and questions to configure your system; installation of the chosen software)
# reboot
>b sd(0,0,0)     -or-     >b sd()     -or-     >b
```

The default boot device can be set with the program ***eeprom***, e.g.:

```
# eeprom bootdev=sd\(0,0,0\).
```

7.3 SunOS 5.X

7.3.1 SunInstall

Our old friend SunInstall is still here. He's changed a little bit though. SunInstall no longer has MUNIX or miniroot. This is because it now works directly from CDROM. All temporary files are stored in memory and it does not require a dedicated swap partition. Since its using the file system from the CDROM its a bit slow; it takes about 5 minutes to boot SunOS 5.X. A significant advantage to this new method is that you can now modify the root and swap partitions of the hard disk with SunInstall, since the program is not using the hard disk.

7.3.2 Hardware Requirements

Solaris 2 runs only on SPARC computers. It will not run on Sun3s. Solaris 2.5-2.6 is supported on Sun4c, Sun4m, and Sun4u hardware, but not Sun4 hardware. Solaris 2.4 will run on all SPARC hardware except the UltraSPARCs (Sun4u). To run Solaris 2 the machine should have

- At least 16 MB of memory
- A minimum of 200 MB of hard disk space, with 400 MB required to install everything.

The machines can be

- Networked Standalone
- Dataless Client (root and swap on same disk)
- Diskless Client
- Homogeneous Server at SunOS 5.X (clients at same or lower release level)
- Heterogeneous Server at SunOS 5.X (clients at differing architecture at same or lower release level, including SunOS 4.1.1 or later)

A CDROM drive is required, either locally or on the subnet, to install the software.

7.3.3 Software Terminology

7.3.3.1 Packages and Clusters

The software is grouped into packages of files and directories related to an application. For example there will be a group for man pages and another for SunOS 4.1.X binary compatibility. These groups of software are labelled **packages.** A package is the standard way to deliver software under Solaris 2, both bundled and unbundled. There are 187 software packages in the Solaris 2.5 base distribution. There are standard commands for administering the packages that we'll look at later. Packages from Sun are generally identified by the **SUNWxxx** naming convention.

Packages may be grouped into **clusters** of related applications. For example, the Source Compatibility Support cluster includes:

Source Compatibility (root), SUNWscpr
Source Compatibility (usr), SUNWscpu
Source Compatibility Archive Library, SUNWsra
Source Compatibility Header Files, SUNWsrh

A cluster can be composed of one or more packages. The cluster names refer to logical names, such as **Source Compatibility Support** and **On-line Manual Pages**, not to the SUNWxxx convention.

7.3.3.2 Configuration Options

The packages and clusters are further grouped into four configuration options for SunInstall.

- **Core** - This contains the software needed to boot and run Solaris 2. It is sufficient for a standalone workstation and includes some networking software and the drivers to run OpenWindows. It does not include OpenWindows or the on-line manual pages.
- **End User** - This contains the software and end user programs required to run Solaris 2. It includes OpenWindows and the on-line manual pages.
- **Developer** - This contains software required to develop software, including: compiler tools, OpenWindows, and the man pages. It does not include any compilers or debuggers.
- **Entire** - This includes the entire Solaris 2 release. It does not include any compilers or debuggers, other than adb.

7.3.4 Partition Planning

All partitions can now be adjusted during SunInstall. You no longer need to run format before starting the installation. Unbundled applications usually have a default installation directory of **/opt** (for optional). Packages that might previously have tried to install into **/usr/lang** or **/usr/local** now suggest **/opt/SUNWxxx** as their default directory.

Its now possible to have smaller swap space than RAM, as idle processes are not automatically swapped out. This was discussed in the section on **swapfs** in an earlier chapter.

Sun’s suggestions for disk partitioning are as follows. I would suggest that for a server you increase **/** and **/opt**, and add a **/var** partition. For a standalone machine you might have only two partitions: one for the software and one for swap.

TABLE 7.1 **Server Disk Partitions, always**

File System	Minimum	Default	Maximum Req.	My Recommendation
/ (root)	12 Mbytes	16 Mbytes	17 Mbytes	20 Mbytes
swap	80 Mbyte	3 times the memory	varies	varies
/usr	30 Mbytes	160 Mbytes	181 Mbytes	220 Mbytes
/opt	5 Mbytes	5 Mbytes	varies	varies
/var	none	part of /	none	30 Mbytes; larger for servers

TABLE 7.2 **Server Disk Partitions, per diskless client**

Client SunOS Version	File System	Each Diskless Client	Each Release
5.X	/export/root	20 Mbytes	10 Mbytes
	/export/swap	24 Mbytes	
	/export/exec		15 Mbytes
4.X	/export/root	16 Mbytes	
	/export/exec	15 Mbytes/sun4 client arch	80 Mbytes

7.3.5 Installation

The installation process has three levels. ***Sysidtool*** prompts for system identification information. ***SunInstall*** does the installation. ***Admintool*** is a GUI tool used to add new users, set-up diskless clients, manage printers and databases.

Installation consists of:

1. Save the previous configuration and data files.
2. Boot from CDROM.
3. Provide ***sysidtool*** the system identification information.
4. Provide ***SunInstall*** the installation information.
5. Choose the Quick or Custom Install option.
6. For a server - identify diskless clients and the architectures to be supported.
7. Select the desired software.
8. Configure the disks to support the desired software and clients.
9. Start the installation.
10. Customize your system.

7.3.5.1 Pre-Installation Information

During installation you'll be asked to provide the following information.

- Hostname
- Host's IP address

Desired name service (NIS+, NIS, none). Should you choose NIS+ or NIS you will also need to provide:
domainname
NIS(+) server hostname
NIS(+) server IP address

Note: At this point you are choosing the NIS(+) domain name and server, not the IP domain name.

- Subnet information
- Geographic region
- Timezone and current date and time
- Configuration:
  Standalone
  Server
  Dataless Client
- Installation Type:
  Quick Install
  Custom Install
- Identify any diskless clients
- Choose the software configuration type:
  all
  core
  developer
  end-user
- Identify the disk(s) to use for the installation and whether or not they should be formatted.

7.3.6 Booting

You can boot from CDROM either locally or remotely. From the old Sun4 systems, i.e. 300 and 400 series at the **>** prompt type:

b sd(0,30,1).

From the older SPARCstations, i.e. SS1, 1+, IPC, and SLC at the ***ok*** prompt type:

boot sd(0,6,2).

From newer SPARCstations, i.e. SS2, SS10, ELC, IPX, 600MP, Sun4m and Sun4u hardware at the **ok** prompt type:

boot cdrom.

7.3.7 Setting Up an Install Server

An install server provides boot service for other machines for installing Solaris 2.X. If your CDROM drive is not directly attached to the machine you're upgrading you might want to set up an install server on the machine with the CDROM. You could also copy the CDROM to a hard disk on the install server. If you take the former route there's only a small temporary space required on the server, about 150 Kbytes per kernel architecture. Copying all of Solaris 2.X onto hard disk requires approximately 320 Mbytes of space. To copy the contents of the CDROM to disk do the following:

```
# mkdir /cdrom
# mount -F hsfs -o ro /dev/dsk/c0t6d0s0 /cdrom-on a 2.X server
-or-
# mount -t hsfs -r /dev/sr0 /cdrom-on a 4.1.X server
-then-
# cd /cdrom
# ./setup_install_server /export/install-or another install directory
```

From then on you can use ***/export/install***, or the specified install directory, in place of the CDROM.

After the client boots from the server installation proceeds as it would for a local installation.

7.3.7.1 Server at 4.1 or 4.1.1

Before you can boot the client using the Solaris 2.X CDROM you need to add the Rock Ridge extensions to the High Sierra File System to allow it to support longer file names, symbolic links, and deeper directories on the CDROM. These are located on the CDROM. To access them:

```
# mkdir /cdrom
# /etc/mount -t hsfs -r /dev/sr0 /cdrom
# cd /cdrom
# ./inst.rr
# cd /
# umount /cdrom
```

The new driver is enabled when you remount the CDROM, as below.

7.3.7.2 Server at 4.1.X (or after installing the rr extensions above)

First mount the CDROM:

```
# mkdir /cdrom
# /etc/mount -t hsfs -r /dev/sr0 /cdrom
```

Then change to the /cdrom directory and execute the command to install clients:

```
# cd /cdrom
# ./add_install_client machine_name architecture
```


-or-

./add_install_client -e ***ethernet_address*** -i ***ip_address machine_name architecture***

where ***machine_name*** is the name of the client and ***architecture*** is either **sun4c** or **sun4m** or **Sun4u**. The ***-e*** and ***-i*** options are necessary if the NIS server does not already have this information.

7.3.7.3 Server at 2.X

First mount the CDROM:

mkdir /cdrom

mount -F hsfs -o ro /dev/dsk/c0t6d0s0 /cdrom

You then use the ***Host Manager*** to introduce the client machine to the server. Entries that you'll add for the client with Host Manager include:

- Hostname
- IP address
- Ethernet address
- Type: standalone, diskless, or dataless
- Time zone region and time zone
- Remote or local (CDROM) installation for the client
- Media server for remote installation (machine with the CDROM)
- OS release to install from the remote server (Sun4c/Sun4m/Sun4u)

You then need to update the NIS(+) maps and you're ready to go.

7.3.7.4 Booting the Client

The server needs to be running the Reverse Address Resolution Protocol and the bootparam daemons, ***rarpd*** and ***rpc.bootparamd***, respectively. You should create the directory, **/tftpboot** if it doesn't already exist on the server. The boot program will be copied here when you add each client.

You cannot boot Solaris 2.X through a router. The install server must be on the same subnet as the client. You don't, however, need to have a complete install server on each subnet. You can have a pared down boot server on the subnet with the kernel architecture required for those subnet clients, and a full install server on another subnet.

After adding the machine as a client of the install server halt and boot the client from the network:

halt

ok boot net

After the client completes the boot process the ***suninstall*** program is invoked automatically.

7.3.8 JumpStart

JumpStart, or **auto-install**, is designed to make the installation easy and automatic. This allows you to pre-configure the information you need to do the install. It also allows you to readily install many machines with the same configuration, rather than having to worry about each of them individually. You make a stub on the disk containing the information about the packages you want to install. On the NIS(+) server you can configure the network information for the new system including the hostname, IP address, time zone and domain name.

If your system is already at Solaris 1.1 (SunOS 4.1.3) or higher you can run JumpStart to upgrade to Solaris 2.X. After backing up your system you would execute the script, **/etc/.install.d/re-preinstall-svr4** when in single user mode to perform the upgrade. The script uses ***/usr/etc/install/bpgetfile*** to try to locate a network version of the Solaris 2 CDROM. ***bpgetfile*** does an rpc broadcast to the local subnet to try to locate an install server with the install program. Failing that it checks locally for a mountable CDROM. Should it find either source it mounts the file system on **/.stub/mnt**. After checking to make sure that it has the necessary files for this architecture it starts the upgrade process and gives you a chance to back out. It reads any custom files that have been placed in **/.stub** to control the install.

7.3.8.1 Configuration Information

When a new machine boots it searches for an install customization directory. It first checks the local floppy drive and if none is found there it then queries the bootparam server for an entry with the keyword **install_config**. The bootparam server would have an entry similar to the following in the **/etc/bootparams** file or it's NIS(+) **bootparams** map:

client_name install_config=server:/export/install_info	-Solaris 1.X
-or-	
* install_config=server:/export/install_info	-Solaris 2.X, which takes wild card entries.

With ***add_install_client*** use the ***-c*** option to automatically setup the bootparam entry when you set up the client:

```
# ./add_install_client -s install_server:/cdrom -c config_server:/dir sunclient sun4
```

where ***/dir*** on ***config_server*** contains the ***auto-install*** configuration information.

There's a sample directory on the Solaris 2.X CDROM, **auto_install_sample**, that you can use as an example when setting up your configuration file. There is also a sample **rules** file there. The boot program provides the **auto-install** program with information about the machine being booted. That, in conjunction with the information provided to the server for this client, provides the **auto-install** program with sufficient information to determine hostname, host and network addresses, NIS(+) domainname, machine model and architecture type, kernel architecture, memory installed, disk size, and hostid for the client. The **rules** file then looks for sets of information defined by the Table below. **Auto-install** will then execute a script before the installation, **begin**, setup a profile based on the **class**, and execute a script when done, **finish**.

Casper Dik has made some good examples of these scripts available on the Internet at ftp://ftp.fwi.uva.nl/pub/solaris/auto-install/.   .

TABLE 7.3 **Auto-install Rules File**

Comparison	Match Type	Expected Data Type
hostname	exact	text string
hostaddress	exact	dotted decimal address
network	exact	dotted decimal address
domainname	exact	NIS(+) domain - text string
arch	exact	machine architecture - text string
karch	exact	kernel architecture - text string
model	exact	machine model, e.g.: 4_75
memsize	range	memory size in MB, e.g.: 16-64
disksize	disk_name range	specific disk size in MB, e.g.: c0t3d0 180-500
installed	disk_slice release	specific disk slice and OS release, e.g.: c0t3d0s0 solaris_2.5
totaldisk	range	total disk size in MB, e.g. 180-500
hostid	exact	8 digit hex number, e.g.: 80442a6b
any	ignored	always matches

Note, that the backslash, ****, at the end of a line of a **rules** line indicates that the line is continued on the next line. Lines beginning with **#** are comments.

So if your **rules** file has entries of the form (modified from an example by Casper Dik):

```
# key         value                begin              class              finish
# SPARCstation LX
karch         sun4m                                   &&\
model         'SUNW, SPARCstation-LX'                 &&\
memsize       16-64                                   &&\
disksize      rootdisk 180-550     scripts/start      packages/default   scripts/finish
# SPARCstations 1, 1+, 2, ELC, SLC, IPC, IPX
karch         sun4c                                   &&\
memsize       16-64                                   &&\
totaldisk     180-250              scripts/start      packages/no_var    scripts/finish
# Anything not matched - don't install
any           -                    scripts/no_match   -                  -
```

Where the lines are interpreted as:

- karch   sun4m   &&\
- model   'SUNW, SPARCstation-LX'   &&\

Specifies that a sun4m type architecture and the model SPARCstation LX are being selected.

- memsize 16-64 &&\
  disksize rootdisk 180-550 scripts/start packages/default scripts/finish

Specifies that if the memory size is 16-64 MB and the root disk size is 180-550 MB set the parameters defined in the **packages/default class** file. Before installation run the **begin** file, **scripts/start**, and upon completion the **finish** script, **scripts/finish**, will be run.

- karch sun4c &&\
  memsize 16-64 &&\
  totaldisk 180-250 scripts/start packages/no_var scripts/finish

Specifies that any machine with sun4c kernel architecture, and memory size of 16-64 MB, and total disk size of 180-250 MB should set the **class** parameters define in **packages/no_var** (e.g. so that /var is not a separate partition), and run the **begin** script, **scripts/start**, and the **finish** script, **scripts/finish**.

- any - scripts/no_match - -

This matches any machine, so if the machine was not flagged by one of the previous lines in the file install with the **scripts/no_match begin** script. If you left off this last line the machine would perform an interactive install if it didn't match a previous line.

An example of a **class** configuration file:

```
filesys any 24 /
filesys any 80 /usr
filesys any 43 /var nosuid
filesys any 0 /home
filesys any 0 /opt
filesys any free swap
cluster    SUNWCreq
package    SUNWvolr    delete
package    SUNWvolu    delete
cluster    SUNWClp
package    SUNWdoc
```

Where these lines are interpreted as:

- filesys any 24 /

The filesys entries specify the disk partitioning in MBs. The entry **partitioning** could be used with either **default** or **existing** to indicate that the disk should be partitioned based on the software selected or that the current partitions should be used, respectively. If not specified **default** is used.

- cluster SUNWCreq

This defines the cluster to be installed. If this is a configuration cluster the **add|delete** fields are ignored.

- package SUNWvolr delete

So the package SUNWvolr has be selected to be deleted from the previously chosen cluster.

Additional fields follow the format:

- product   product_name   product_version

to specify a product to install and the OS version (e.g. Solaris 2.5).

- filesys   any   (size | all)   filesystem

to define the size to partition various file systems, e.g.:
filesys   any   60   swap

- package   package_name   (add | delete)

to specify a package to install. The **add** and **delete** options specify whether the package should be added or deleted from the configuration cluster and clusters previously chosen, e.g.:
package   SUNWaudmo   add
package   SUNWman   delete

7.3.8.2 Automatic Installation

With existing machines specify the **install** option when booting to enable **auto-install**, i.e.:

ok boot net - install

Note: There must be a apace between the **-** and **install**.

Machines already installed with Solaris 2.X can be reconfigured by touching the **AUTOINSTALL** file in the root directory and then rebooting, e.g.:

touch /AUTOINSTALL

reboot

The machine will attempt to auto-install during the boot process.

7.3.9 Installation Files

The installation programs keep a record of the packages and files installed on the system in the **/var/sadm** directory.

/var/sadm/install_data - contains a **CLUSTER** file listing the installed clusters, and a log file of the installation, **install_log**.

/var/sadm/install - the **contents** file in this directory keeps a listing of the various packages installed on the system and their locations, permissions, ownership, and the package with which they are associated.

/var/sadm/softinfo - the INST_RELEASE and Solaris_2.X files in this directory contain the OS name and version number.

/var/sadm/pkg - the subdirectories here represent every software package installed on your system. Each subdirectory contains a file describing the package and another listing the files associated with the package.

/var/sadm/patch - there is a subdirectory here for each patch installed, with files to describe the patch and the package and files associated with it. There's also a script to backout the patch should that become necessary. If you're short of space in /var/sadm you can install the patch with the "***-d***" option to prevent saving the original files. In that case you won't be able to backout the patch.

7.4 Post Install Actions

After you’ve installed the operating system you should install the recommended patches, and then you can personalize the system to your needs. Below are some of the steps I take to finish and personalize the configuration on my systems.

TABLE 7.4 **Post Install Actions**

<table>
<tr><th>Procedure</th><th>Purpose</th><th>SunOS 4.X</th><th>SunOS 5.X</th></tr>
<tr><td>touch /TIMESTAMP</td><td>let's you know the start time</td><td>X</td><td>X</td></tr>
<tr><td>passwd</td><td>set a passwd on the root login</td><td>X</td><td>X</td></tr>
<tr><td>rm /etc/hosts.equiv</td><td>comes with "+"</td><td>X</td><td></td></tr>
<tr><td>vi /etc/netmasks
128.146.0.0 255.255.255.0</td><td>create an entry for the network, e.g. for 128.146</td><td>X</td><td>X</td></tr>
<tr><td>vi /etc/ttytab</td><td>change "secure" -> "unsecure"</td><td>X</td><td></td></tr>
<tr><td>ifconfig le0 netmask + broadcast xxx.yyy.zzz.255</td><td>reset the netmask and broadcast</td><td>X</td><td></td></tr>
<tr><td>route add default xxx.yyy.zzz.1 1</td><td>reset route</td><td>X</td><td>X</td></tr>
<tr><td>ftp wks.uts.ohio-state.edu</td><td>under /pub/sunpatches retrieve the patch files needed. Contact wks@wks.uts.ohio-state.edu for the current patch list for your version of the OS.</td><td>X</td><td>X</td></tr>
<tr><td>Install patches</td><td>follow READMEs for the individual patches</td><td>X</td><td>X</td></tr>
<tr><td>echo "xxx.yyy.zzz.1" > /etc/defaultrouter</td><td>create /etc/defaultrouter with the IP address of the default router</td><td>X</td><td>X</td></tr>
<tr><td>vi /etc/rc.local
mount /tmp
chmod 1777 /tmp
chmod g+s /tmp
ifconfig le0 broadcast `cat /etc/defaultrouter |sed -n "s/\.[0-9]$/\.255/p"`</td><td>edit /etc/rc.local
to mount /tmp as tmpfs (on swap) and to set the proper permissions on the directory
and to set the proper broadcast</td><td>X</td><td></td></tr>
<tr><td>vi /etc/fstab
swap /tmp tmp rw 0 0</td><td>add the line to mount /tmp on swap</td><td>X</td><td></td></tr>
<tr><td>Generate a new kernel and reboot with it</td><td>this is required for several of the OS patches</td><td>X</td><td></td></tr>
<tr><td>vi /.cshrc /.login /.profile</td><td>edit to taste and remove "." from path</td><td>X</td><td>X</td></tr>
<tr><td>vipw</td><td>protect all accounts, even sync
remove +: entry if not using NIS</td><td>X</td><td></td></tr>
<tr><td>Create necessary accounts</td><td>we’ll look at how to do this in a later Chapter</td><td>X</td><td>X</td></tr>
<tr><td>vi /etc/group</td><td>remove +: entry if not using NIS</td><td>X</td><td></td></tr>
</table>

TABLE 7.4 **Post Install Actions**

Procedure	Purpose	SunOS 4.X	SunOS 5.X
Add tcsh and/or bash to /usr/bin	much better than csh or sh for login (the sources can be obtained via anonymous ftp from tesla.ee.cornell.edu in /pub/tcsh and prep.ai.mit.edu in /pub/gnu for tcsh and bash, respectively.)	X	X
cat << EOF > /etc/shells /sbin/sh /bin/sh /bin/csh /bin/ksh /bin/bash /bin/tcsh EOF	add entries for all login shells, e.g.:	X	X
chown root /home	not caught by patch 100103	X	
rm -rf /var/spool/uucppublic	writable by everyone, so remove if not used	X	X
Install resolv+2.1.1 package /usr/lib/libresolv.a /usr/lib/libc.so.1.9.1 /usr/lib/libc.sa.1.9.1 Now execute ldconfig Then copy the new include files to /usr/include.	for DNS, or use NIS, it includes: resolver library shared library shared library to pick up the new libraries	X	
cat << EOF > /etc/resolv.conf domain acs.ohio-state.edu. nameserver 128.146.1.7 nameserver 128.146.48.7 search acs.ohio-state.edu magnus.acs.ohio-state.edu cis.ohio-state.edu eng.ohio-state.edu EOF	for DNS, with the IP domain, up to 3 nameservers, these are ns1.net and ns2.net and a search path	X	X
cat << EOF > /etc/host.conf order hosts,bind trim .magnus.acs.ohio-state.edu, .acs.ohio-state.edu nospoof on alert on EOF	used by resolv+ set the host database order to search trim the domains	X	

TABLE 7.4 Post Install Actions

Procedure	Purpose	SunOS 4.X	SunOS 5.X
vi /etc/nsswitch.conf hosts: files dns	set name service switch lookups set the host database order to search		X
vi /etc/syslog.conf define(LOGHOST,localhost) -or- vi /etc/hosts www.xxx.yyy.zzz hostname loghost	define LOGHOST (first line in file), or reference the files locally and remove the "ifdef('LOGHOST'..." entries, as desired add the alias loghost to your hostname entry, not to the localhost entry	X	X
chmod o-w /etc/*	remove general write permissions	X	X
Set up xntp, including change in /etc/services for udp service	Network Time Protocol	X	X
Install any other desired packages, e.g. perl, language compilers, etc.	make the system more usable	X	X
Backup the system	so you can reproduce the current state after a catastrophe.	X	X

7.5 Sun Patch List

Ohio State University members can usually find the necessary SunOS patches on the patch server, ftp://wks.uts.ohio-state.edu/pub/sunpatches/. If you don't find what you need there, contact wks@wks.uts.ohio-state.edu. Others should contact Sun Microsystems, or their software vendor, for patches.

7.5.1 SunOS 4.1.3_U1 (Solaris 1.1.1)

100103-12 SunOS 4.1.3;4.1.3_U1: set file permissions to more secure mode

101434-03 SunOS 4.1.3_U1: lpr Jumbo Patch

101436-08 SunOS 4.1.3_U1: patch for mail executable

101440-01 SunOS 4.1.3_U1: security problem: methods to exploit login/su

101508-14 SunOS 4.1.3_U1: Sun4m kernel patch

101558-07 SunOS 4.1.3_U1: international libc jumbo patch

101579-01 SunOS 4.1.3_U1: Security problem with expreserve for Solaris 1.1.1

101587-01 SunOS 4.1.3_U1: security patch for mfree and icmp redirect

101592-07 SunOS 4.1.3_U1: UFS File system Patch

101621-04 SunOS 4.1.3_U1: tty patch CTE zs driver gates reception on CD for hardware flow control

101625-02 SunOS 4.1.3_U1: ftp does not prompt for account information

101665-07 SunOS 4.1.3_U1: sendmail jumbo patch
101679-01 SunOS 4.1.3_U1: Breach of security using modload
101759-04 SunOS 4.1.3_U1: domestic (US only) libc jumbo patch
101784-04 SunOS 4.1.3_U1: rpc.lockd/rpc.statd jumbo patch
102060-01 SunOS 4.1.3_U1: Root access possible via passwd race condition
102177-04 SunOS 4.1.3_U1: NFS Jumbo Patch
100444-76 OpenWindows 3.0: OpenWindows V3.0 Server Patch 3000-124
100448-03 OpenWindows 3.0: loadmodule Patch
100452-72 OpenWindows 3.0: XView 3.0 Jumbo Patch
100478-01 OpenWindows 3.0: xlock crashes leaving system open
101435-02 [1]SunOS 4.1.3_U1: ypserv and ypxfrd fix

7.5.2 SunOS 4.1.4 (Solaris 1.1.2)

100103-12 [2]SunOS 4.1.3;4.1.3_U1: set file permissions to more secure mode
102264-02 SunOS 4.1.4: rpc.lockd patch for assertion failed panic
102394-02 SunOS 4.1.4: NFS Jumbo Patch
102414-01 SunOS 4.1.4: mail jumbo patch
102423-04 Sunos 4.1.4: Sendmail jumbo patch
102436-02 SunOS 4.1.4: Machine soft hangs and hangs on bootup (sun4m)
102544-04 SunOS 4.1.4: domestic (U.S. only) libc jumbo patch
102545-04 SunOS 4.1.4: international libc jumbo patch
100444-76 OpenWindows 3.0: OpenWindows V3.0 Server Patch 3000-124
100448-03 OpenWindows 3.0: loadmodule Patch
100452-72 OpenWindows 3.0: XView 3.0 Jumbo Patch
100478-01 OpenWindows 3.0: xlock crashes leaving system open
102516-04 [1]SunOS 4.1.4: UFS File system Patch

Some patches apply only to specific hardware, and 102544 and 102545 are mutually exclusive, as they apply to the domestic and international versions of the libraries, respectively.

7.5.3 SunOS 5.4 (Solaris 2.4)

101945-41 SunOS 5.4: kernel patch
101959-07 SunOS 5.4: lp jumbo patch
101973-16 SunOS 5.4: fixes for libnsl and ypbind
102042-05 SunOS 5.4: usr/bin/mail jumbo patch

1. This patch is on the Security list, but not the Recommended list, because it's assumed to be too application dependent and not relevant to all sites.
2. Not actually on the Recommended list for this release, but you will want to check the changes this script makes to be sure that you have similar file permission settings on your system

102044-01	SunOS 5.4: bug in mouse code makes "break root" attack possible
102066-09	SunOS 5.4: sendmail patch
102070-01	SunOS 5.4: Bugfix for rpcbind/portmapper
102165-03	SunOS 5.4: nss_dns.so.1 fixes
102216-07	SunOS 5.4: klmmod and rpcmod patch
102218-03	SunOS 5.4: libbsm fixes
102277-02	SunOS 5.4: nss_nisplus.so.1 fixes
102437-03	SunOS 5.4: /usr/ccs/bin/as has an internal error
102479-02	SunOS 5.4: DNS spoofing is possible per Cern ca-96.02
102656-01	SunOS 5.4: /dev/qec should protect against being opened directly
102664-01	SunOS 5.4: data fault in scanc() due to bad "cp" argument
102680-03	SunOS 5.4: fixes for ufsdump and wall
102693-03	SunOS 5.4: at and atrm fixes
102704-02	SunOS 5.4: jumbo patch for NIS commands
102711-01	SunOS 5.4: Creation of /tmp/ps_data is security problem
102741-01	SunOS 5.4: libm can hit SEGV in multi-threaded mode
102756-01	SunOS 5.4: expreserve still has security problem
102769-03	SunOS 5.4: statd fixes
102788-02	SunOS 5.4: Jumbo patch for sccs bug fixes.
102922-03	SunOS 5.4: inetd fixes
102960-01	SunOS 5.4: vipw has security problem
103070-01	SunOS 5.4: tip will read and print any uucp owned file
103270-01	SunOS 5.4: nissetup default permissions not secure enough
101878-13	OpenWindows 3.4: Xview Patch
102292-02	OpenWindows 3.4: filemgr (ff.core) fixes
103290-02	SPARCstorage Array 2.0: SSA Jumbo patch for Solaris 2.4 11/94, HW395
102049-02	SunOS 5.4: linker fixes
102303-05	[3]SunOS 5.4: POINT PATCH: linker fixes
102336-01	[3]SunOS 5.4: POINT PATCH: 1091205 - Password aging & NIS+ don't work

7.5.4 SunOS 5.5 (Solaris 2.5)

102971-01	SunOS 5.5: vipw fix
102980-07	SunOS 5.5: sendmail patch
103093-03	SunOS 5.5: kernel patch
103169-06	SunOS 5.5: ip driver and ifconfig fixes
103241-01	SunOS 5.5: Undefined symbol in libc.so.1.9

3. This patch is on the Security list, but not the Recommended list, because it's assumed to be too application dependent and not relevant to all sites.

103242-01	SunOS 5.5: linker patch
103266-01	SunOS 5.5: nissetup default permissions for password table not secure
103279-02	SunOS 5.5: nscd breaks password shadowing with NIS+
103447-03	SunOS 5.5: tcp patch
103468-01	SunOS 5.5: statd security problem
103667-01	SunOS 5.5: DNS spoofing is possible per Cern ca-96.02
103703-01	SunOS 5.5: nss_dns.so.1 source modification and rebuild for BIND 4.9.3
103708-01	SunOS 5.5: rpc.nisd_resolv rebuild for BIND 4.9.3
103746-01	SunOS 5.5: XFN source modifications for BIND 4.9.3
103815-01	SunOS 5.5: rdist suffers from buffer overflow
102832-01	OpenWindows 3.5: Xview Jumbo Patch
103300-02	OpenWindows 3.5: ff.core security patch
103017-04	SPARCstorage Array Solaris 2.5: Jumbo patch for SSA for Solaris 2.5

7.5.5 SunOS 5.5.1 (Solaris 2.5.1)

103582-01	SunOS 5.5.1: /kernel/drv/tcp patch
103594-03	SunOS 5.5.1: /usr/lib/sendmail fixes
103630-01	SunOS 5.5.1: ip and ifconfig patch
103663-01	SunOS 5.5.1: DNS spoofing is possible per Cern ca-96.02
103680-01	SunOS 5.5.1: nscd/nscd_nischeck rebuild for BIND 4.9.3
103683-01	SunOS 5.5.1: nss_dns.so.1 rebuild for BIND 4.9.3
103686-01	SunOS 5.5.1: rpc.nisd_resolv rebuild for BIND 4.9.3
103743-01	SunOS 5.5.1: XFN source modifications for BIND 4.9.3
103817-01	SunOS 5.5.1: rdist suffers from buffer overflow

7.6 IRIX 5.X

7.6.1 Installation

When you boot your SGI machine you'll have a few seconds to press the "**Stop for Maintenance**" button on the "**Starting up the System**" window. From there you'll be given the choices:

- Start System
- Install System Software
- Run Diagnostics
- Recover System
- Enter Command Monitor
- Select Keyboard Layout

Select "**Install System Software**" with the mouse. Then choose the source, e.g. "**Local CD-ROM**", for the software. The miniroot, including the installation tool, ***inst***, will be copied from CDROM to swap on the local disk. The system will then reboot from swap, putting you into the miniroot. Run ***inst***, from which you'll be given the "**Inst>** " prompt. At this point you have various options available to you with ***inst***. You can use ***list*** to list the software, e.g. "***list* * ***" will list all the packages available. Then ***install*** will add the product to the list to be installed, along with the defaults already marked, e.g. "***install print.man.bsdlpr***" to choose the man pages for the BSD style line printer package. After choosing your software type "***go***" to start the installation. When the installation is completed you can "***quit***" from the ***inst*** tool and reboot the system.

7.6.2 Post Install

Now you can personalize the system. Some things you might want to change include the following.

1. turn off the route daemon
   To do this edit **/etc/config/routed** and change "*on*" to "*off*".
2. set a default route
   Edit **/etc/init.d/network** and add a line similar to:
   /usr/etc/route add default xxx.yyy.zzz.1 1
   before the routed line.
3. remove the setuid/setgid bits from **/usr/lib/desktop/permissions** to close this security hole
4. get the latest BSD ***sendmail***, or install the sendmail patch, again for security concerns.

Also, read through the steps above for SunOS to see which might be applicable here.

Ohio State University members can usually find the necessary IRIX patches on the patch server, ftp://araminta.acs.ohio-state.edu/pub/sgi/patches/.

CHAPTER 8 Kernel Configuration

8.1 SunOS 4.1.X

The SunOS 4.1.X kernel that comes with the installation is configured to allow the use of all supported devices for the architecture. This makes it quite large and causes it to take up considerable memory. Since most systems will not have all the supported peripherals you can remove those that aren't needed, freeing memory space for use by programs. If you add additional devices, then you need to put the drivers back in and reconfigure and reinstall the kernel. It is not necessary to reconfigure the SunOS 5.X kernel, as this kernel loads only the drivers for the devices attached to the system.

8.1.1 Kernel configuration files

Templates for the kernel configuration can be found in the directory **/usr/share/sys/sun{3,3x,4,4c,4m}/conf**. Some of the templates are:

DL60	- diskless 4/60 (SS2)
DLS60	- diskless 4/60 with local swap
GENERIC	- default (all general supported devices)
GENERIC_SMALL	- default for generic_small (8 SCSI disks, 4 SCSI tapes, 2 CDROMs)
Makefile.src	- makefile for the compilation
NFS60	- to boot a disk-equipped machine from a server
README	- detailed directions for building the kernel
SDST60	- 4/60 with SCSI disks and tapes

Normally you will configure the kernel to match the hardware of a system e.g. disk(s)/diskless, tape(s), color monitor, etc.

Reconfiguring the kernel should save memory space and allow the kernel to execute faster.

8.1.2 Overview of Sysgen process

1. cd /usr/share/sys/sun{3,3x,4,4c,4m}/conf
2. cp GENERIC HOSTNAME - copy the configuration file
3. vi HOSTNAME - edit and revise as needed
4. config HOSTNAME - build the system configuration files

5. cd ../HOSTNAME - cd to the new directory
6. make - compile the new kernel
7. mv /vmunix /vmunix.gen - save the old kernel
8. cp vmunix / - install the new kernel
9. reboot - reboot using the new kernel

Sometimes the new kernel will not run properly. The patch may have been faulty; you may have left out defining one of the necessary parameters; the object files may have been corrupted, etc. If you can't boot from the new kernel for any reason, reboot using the old kernel and then repeat the steps above to regenerate a new kernel. Reboot with:

>b vmunix.gen

8.2 SunOS 5.X

8.2.1 Autoconfiguration

Under Solaris 2 the kernel is now modularized. Whenever the kernel needs a module it loads it and processes it. The kernel is now ***/kernel/unix*** for early versions of Solaris, SunOS 5.0-5.4). Solaris 2.5 and above (SunOS 5.5+) has both a generic, platform-independent part (***/kernel/genunix***) and a core, or platform-specific part (***/platform/'uname -m'/kernel/unix***) of the kernel. These are combined to form the running kernel.

You can customize the kernel with the **/etc/system** file. This configuration file contains commands to be read by the kernel during initialization. You can specify that modules be excluded, or loaded during initialization, rather than when first used, etc. You can set the root and swap devices to something other than the default value. You can even set the value of kernel parameters, e.g.:

set maxusers=16

Each type of module has it's own subdirectory in **/kernel**, e.g. the device drivers are under **/kernel/drv**. Each driver also has a configuration file associated with it to set the kernel parameter values. Solaris 2.5 and above again has a platform-independent set in **/kernel/drv** and a platform-dependent set in **/platform/'uname -m'/kernel/drv**.

A significant advantage to modularization is that the kernel now only loads the modules it needs, making more efficient use of memory. Also, you can add drivers without having to rebuild the kernel and reboot the system.

8.2.2 Accessing New Device Drivers

Should you add new device drivers they should be installed in ***/kernel***. You can add drivers with the ***add_drv*** command and remove them with the ***rm_drv*** command. Once the driver is installed and the new device connected reboot the system with:

ok boot -r

Alternatively, you can create the file **/reconfigure** before rebooting. The kernel will then be reconfigured during the boot process.

```
# touch /reconfigure
# reboot
```

One of these procedures is required for all drivers not installed initially. It causes the kernel to properly recognize the new drivers during the boot process.

8.2.3 Device Configuration

During the boot process devices are identified and new ones are automatically added to /devices and /dev. So you no longer have to execute ***MAKEDEV*** to configure the new devices. The equivalent is done for you with the new automatic reconfiguration process when you boot.

The Solaris 2.X system is responsible for assigning an unused major number when you add a device, so these should not be hard-coded into the drivers. Minor numbers are assigned by the driver.

Should you need to reconfigure the **/devices** directory you can do this with the ***drvconfig*** command. This should create the /devices directory tree from the attached hardware. It uses the ***dev_info*** tree of the kernel. The devices should be powered on when you run this command. Normally this is done for you whenever a new driver is installed with the ***add_drv*** utility and you reboot the system with the ***-r*** option. ***drvconfig*** uses the file ***/etc/minor_perm*** to determine the permissions to apply to the devices and the file ***/etc/name_to_major*** to assign major device numbers.

Use the utility ***prtconf*** to display the devices configured on your system.

```
# prtconf
System Configuration:  Sun Microsystems  sun4m
Memory size: 64 Megabytes
System Peripherals (Software Nodes):

SUNW,SPARCstation-5
    packages (driver not attached)
        disk-label (driver not attached)
        deblocker (driver not attached)
        obp-tftp (driver not attached)
    options, instance #0
    aliases (driver not attached)
    openprom (driver not attached)
    iommu, instance #0
        sbus, instance #0
            espdma, instance #0
                esp, instance #0
                    sd (driver not attached)
                    st (driver not attached)
```



```
                    sd, instance #0 (driver not attached)
                    sd, instance #1
                    sd, instance #2 (driver not attached)
                    sd, instance #3
                    sd, instance #4 (driver not attached)
                    sd, instance #5
                    sd, instance #6
            SUNW,bpp (driver not attached)
            ledma, instance #0
                le, instance #0
            SUNW,lpvi, instance #0
            SUNW,bpp (driver not attached)
            cgsix, instance #0
            power-management (driver not attached)
            SUNW,CS4231, instance #0
            afx-misc (driver not attached)
    obio, instance #0
        zs, instance #0
        zs, instance #1
        eeprom (driver not attached)
        slavioconfig (driver not attached)
        auxio (driver not attached)
        counter (driver not attached)
        interrupt (driver not attached)
        power (driver not attached)
        SUNW,fdtwo, instance #0
    memory (driver not attached)
    virtual-memory (driver not attached)
    FMI,MB86904 (driver not attached)
    pseudo, instance #0
```

8.2.4 Creation of the logical name space

The last stage of the automatic configuration process involves the generation of the logical name space to correspond with the new devices. Several utilities are used for this, depending on the type of device.

- ***disks*** adds /dev entries for hard disks
- ***tapes*** adds /dev entries for tape drives
- ***ports*** adds /dev and ***inittab*** entries for serial lines
- ***devlinks*** adds /dev entries for miscellaneous devices and pseudo-devices, according to the entries in **/etc/devlink.tab**

8.2.5 Tuning Kernel Parameters

Many kernel parameters scale relative to the value chosen for **maxusers**. You can change many others that affect the kernel and kernel modules by setting values for them in **/etc/system**. With **/etc/system** you can specify:

- kernel modules to be loaded automatically
- kernel modules not to be loaded automatically
- root and swap devices
- new values for kernel integer variables

To get a complete list of the tunable kernel parameters use the ***/usr/ccs/bin/nm*** command on the kernel, e.g.:

```
# /usr/ccs/bin/nm /kernel/unix                                       -for Solaris 2.4
# /usr/ccs/bin/nm /kernel/genunix /platform/‘uname -m‘/kernel/unix   -for Solaris 2.5
```

which yields over 5000 lines of kernel parameters, of the form:

```
Symbols from /kernel/unix:
[Index]     Value      Size   Type  Bind   Other   Shndx   Name
[1]        |      0|      0|FILE |LOCL |0    |ABS    |unix
```

Most of these you will never need to change. You should also be aware that kernel parameters and their meanings may change in latter releases of the OS, so you should not blindly copy **/etc/system** files to new machines.

You can get a list of the drivers and modules currently loaded and some selected kernel parameter values by using the ***/usr/sbin/sysdef*** command with the ***-i*** option as shown below.

```
# sysdef -i
[...]
* Loadable Objects
*
genunix
misc/consconfig
[...]
fs/nfs
                        hard link:  sys/nfs
fs/procfs
fs/specfs
fs/tmpfs
fs/ufs
[...]
sys/semsys
sys/shmsys
drv/arp
                        hard link:  strmod/arp
drv/arp
[...]
* Tunable Parameters
*
 1306624                maximum memory allowed in buffer cache (bufhwm)
    1002                maximum number of processes (v.v_proc)
      99                maximum global priority in sys class (MAXCLSYSPRI)
     997                maximum processes per user id (v.v_maxup)
      30                auto update time limit in seconds (NAUTOUP)
```

```
      25                          page stealing low water mark (GPGSLO)
       5                          fsflush run rate (FSFLUSHR)
      25                          minimum resident memory for avoiding deadlock (MINARMEM)
      25                          minimum swapable memory for avoiding deadlock (MINASMEM)
*
* Utsname Tunables
*
    5.5  release (REL)
  nyssa  node name (NODE)
  SunOS  system name (SYS)
 Generic  version (VER)
*
* Process Resource Limit Tunables (Current:Maximum)
*
Infinity:Infinity                 cpu time
Infinity:Infinity                 file size
7ffff000:7ffff000                 heap size
 800000:7ffff000                  stack size
Infinity:Infinity                 core file size
     40:     400                  file descriptors
Infinity:Infinity                 mapped memory
*
* Streams Tunables
*
     9                            maximum number of pushes allowed (NSTRPUSH)
 65536                            maximum stream message size (STRMSGSZ)
  1024                            max size of ctl part of message (STRCTLSZ)
*
* IPC Messages
*
   100                            entries in msg map (MSGMAP)
  2048                            max message size (MSGMAX)
  4096                            max bytes on queue (MSGMNB)
    50                            message queue identifiers (MSGMNI)
     8                            message segment size (MSGSSZ)
    40                            system message headers (MSGTQL)
  1024                            message segments (MSGSEG)
*
* IPC Semaphores
*
    10                            entries in semaphore map (SEMMAP)
    10                            semaphore identifiers (SEMMNI)
    60                            semaphores in system (SEMMNS)
    30                            undo structures in system (SEMMNU)
    25                            max semaphores per id (SEMMSL)
    10                            max operations per semop call (SEMOPM)
    10                            max undo entries per process (SEMUME)
 32767                            semaphore maximum value (SEMVMX)
 16384                            adjust on exit max value (SEMAEM)
*
* IPC Shared Memory
*
1048576                           max shared memory segment size (SHMMAX)
     1                            min shared memory segment size (SHMMIN)
   100                            shared memory identifiers (SHMMNI)
     6                            max attached shm segments per process (SHMSEG)
*
* Time Sharing Scheduler Tunables
*
60                                maximum time sharing user priority (TSMAXUPRI)
SYS                               system class name (SYS_NAME)
```


To get and set kernel driver configuration parameters you can use the command ***/usr/sbin/ndd***. At this time ***ndd*** only supports access to the TCP/IP modules. Use the "*-set*" option to set a value, without it you query the named device driver, e.g. to get a list of the IP driver parameters execute:

# ndd /dev/ip \?	- "?" indicates to list all parameters for the driver
?	(read only)
ip_ill_status	(read only)
ip_ipif_status	(read only)
ip_ire_status	(read only)
ip_rput_pullups	(read and write)
ip_forwarding	(read and write)
ip_respond_to_address_mask_broadcast	(read and write)
ip_respond_to_echo_broadcast	(read and write)
ip_respond_to_timestamp	(read and write)
ip_respond_to_timestamp_broadcast	(read and write)
ip_send_redirects	(read and write)
ip_forward_directed_broadcasts	(read and write)
ip_debug	(read and write)
ip_mrtdebug	(read and write)
ip_ire_cleanup_interval	(read and write)
ip_ire_flush_interval	(read and write)
ip_ire_redirect_interval	(read and write)
ip_def_ttl	(read and write)
ip_forward_src_routed	(read and write)
ip_wroff_extra	(read and write)
ip_ire_pathmtu_interval	(read and write)
ip_icmp_return_data_bytes	(read and write)
ip_send_source_quench	(read and write)
ip_path_mtu_discovery	(read and write)
ip_ignore_delete_time	(read and write)
ip_ignore_redirect	(read and write)
ip_output_queue	(read and write)
ip_broadcast_ttl	(read and write)
ip_icmp_err_interval	(read and write)
ip_reass_queue_bytes	(read and write)
ip_strict_dst_multihoming	(read and write)

To get the value of a specific driver:

```
# ndd /dev/ip ip_forwarding
2
```

To disable packet forwarding (i.e. on a firewall machine) set this value to "**0**", as is done in the startup script **/etc/init.d/inetinit**:

```
# ndd -set /dev/ip ip_forwarding 0
```


To set values for kernel parameters in **/etc/system** you would use the form:

set **module:variable**=*value*

some examples would be:

set maxusers=16

to raise **maxusers** above the default value of 8. Actually the default value for maxusers is chosen based on the amount of available memory, with a maximum of 2048, according to:

TABLE 8.1 **Solaris 2.X maxusers default values**

Memory Size	Maxusers value
< 32 MB	8
< 40 MB	32
< 64 MB	40
< 128 MB	64
≥ 128 MB	128

Maxusers affects the default settings for several other kernel table parameters according to the formula in the following table.

TABLE 8.2 **Kernel Parameter values affected by Maxusers**

Kernel Table	Kernel Variable	Variable Value
Callout	ncallout	16+max_nprocs
Inode	ufs_ninode	max_nprocs+16+maxusers+64
Name Cache Lookup	ncsize	max_nprocs+16+maxusers+64
Process	max_nprocs	10+16*maxusers
Disk Quota Structure	ndquot	(maxusers*NMOUNT)/4+max_nprocs
User Processes	maxuprc	max_nprocs-5

The parameters **npty** and **pt_cnt** are not automatically tuned with the size of memory or **maxusers**, and may need to be reset to allow more network connections on a large machine.

Another example where you might reset a kernel parameter is to have NFS always check that the request is coming from a port number < 1024 (i.e. a "trusted port"). Do this for Solaris 2.4 with:

set nfs:nfs_portmon=1

and for Solaris 2.5 with:

set nfssrv:nfs_portmon=1

where the module containing the parameter has changed from **nfs** to **nfssrv**.

Some kernel parameters that you might consider tuning are in the table below.

TABLE 8.3 **Some Tunable Kernel Parameters**

Parameter	Default Value	Practical Limit	Function
pt_cnt	48	3000	number of 5.X style pseudo-ttys.; sets the limit for the number of remote logins. Reboot with the "-r" option to create the /dev/pts entries.
npty	48	3000	number of 4.X style pseudo-ttys
ncsize	17*maxusers + 90	16000	Directory Name Lookup Cache (DNLC) size. Increase for NFS server with lots of clients. "vmstat -s" reports the cache hit rate.
ufs_ninode	17*maxusers + 90	34906	maximum number of inodes cached; should be at least as large as ncsize
maxuprc	16*maxusers + 5		set this if you want to limit the number of processes a user can have
bufhwm	0, which allows up to 2% of physical memory	20% of physical memory	maximum size of the buffer cache (Kbytes). Caches inode, indirect block, and cylinder group information. "sar -b" reports the buffer cache hit rate.

You need to be very careful about the changes you make in **/etc/system**. It's possible that by putting incorrect values in **/etc/system** you could leave the machine in a state in which it is unable to boot. Should this occur, boot with the "**-a**" option, and when the system asks you to provide the configuration file name input **/dev/null** instead of **/etc/system**. Then edit **/etc/system** to correct the problem and reboot again.

8.3 IRIX 5.X

The autoconfiguration script ***/etc/init.d/autoconfig*** is run at run-level 2, ***S23autoconfig***, during the boot process. If new boards or devices are found, or if changes have been made to the object files or system tuning files in **/var/sysgen/mtune/***, **/var/sysgen/master.d/***, or **/var/sysgen/system/*** the program will check the **/var/config/autoconfig.options** file to see if it should automatically generate a new kernel. The default "**-T**" option indicates this. Otherwise it will prompt to generate a new kernel. So you should rarely, if ever, need to generate a new kernel by hand.

autoconfig uses the ***lboot*** command to actually generate the new kernel and reads the ***/var/sysgen/stune*** file for the settings of any tunable parameters different from the defaults. This creates a new kernel and saves it as ***/unix.install***. When doing this by hand you should then copy the old kernel, ***/unix***, to a new name, e.g. ***/unix.save*** and reboot the system with "***reboot***".

The ***/usr/sbin/systune*** program can be used to examine or change kernel tunable parameters; in the latter case it will add entries to **/var/sysgen/stune**.

A few of the tunable parameters listed by ***systune*** are, e.g. for NFS parameters:

```
snfs (statically changeable)
svc_maxdupreqs = 136 (0x88)
nfs_portmon = 0 (0x0)
```

You can execute systune in interactive mode to examine and set parameters, e.g. to report and then raise the value for the number of system processes, **nproc**:

```
# systune -i
systune-> nproc
        nproc = 300 (0x12c)
systune-> nproc 500
        nproc = 300 (0x12c)
Do you really want to change nproc to 500 (0x1f4)? (y/n) y
In order for the change in parameter nproc to become effective,
reboot the system
systune-> quit
```

This creates the new kernel ***/unix.install***. The parameter change will take effect the next time you reboot the system. When this file exists ***/etc/init.d/autoconfig*** reconfigures the kernel as part of the boot process.

Should you need to recover from an unbootable kernel following an unsuccessful kernel regeneration, interrupt the boot process and go to "**System Maintenance Menu**". There select "**Command Monitor**". At the "**>>** " prompt boot from the old kernel, e.g.:

```
>> boot unix.save
```

8.4 Digital UNIX

Digital UNIX recommends that you be in single user mode when building the kernel. The steps to follow are:

1.	cp /vmunix /vmunix.save	- save the old kernel
2.	cp /genunix /vmunix	- install the generic kernel to be the running kernel
3.	/usr/sbin/shutdown -r +5	- shutdown the system
4.	Log on as root and take the system down to single user mode	
5.	/usr/sbin/shutdown +1	
6.	mount /usr	- remount the /usr file system
7.	/usr/sbin/doconfig	- you will be prompted for system configuration information. If you need to edit the resulting configuration file answer "yes" at the prompt. The new kernel will then be built and the path to it will be displayed.
8.	mv /sys/DECOSF/vmunix /vmunix	- move the kernel from the path displayed in the step above to the root directory
9.	/usr/sbin/shutdown -r now	- reboot the system

If the system fails to boot you can reboot to single user mode using the generic kernel (/genunix) and try again.

8.5 Ultrix

Ultrix is similar to SunOS 4.1.X when building a kernel. The steps to follow on a MIPS hardware platform are:

1.	cd /sys/conf	
2.	cp GENERIC HOSTNAME	- copy the configuration file
3.	vi HOSTNAME	- edit and revise as needed
4.	config HOSTNAME	- build the system configuration files
5.	cd /sys/MIPS/HOSTNAME	- change to the new configuration directory
6.	make	- compile the new kernel
7.	mv /vmunix /genvmunix	- save the old kernel
8.	cp vmunix /	- install the new kernel
9.	reboot	- reboot using the new kernel

CHAPTER 9 Adding Hardware

9.1 SunOS 4.1.X

9.1.1 Procedures for adding new hardware to a system

When adding new hardware to a SunOS 4.1.X system may need to reconfigure the kernel before the device will be supported. In general you should follow these steps.

1. Reconfigure the kernel to support the device.
2. Generate the special files that allow the kernel to communicate with the device. These can generally be created with **/dev/MAKEDEV** using the ***mknod*** command.
3. Halt the system and Connect the actual hardware; insert boards, disk, etc.
4. Reboot and Change any files that the system uses concerning the new hardware.

An excellent guide for all your modem and terminal concerns is available on the web, written by Celeste Stokely, **http://www.stokely.com/**.

9.1.2 MAKEDEV

MAKEDEV is a script located in **/dev** which puts its generated files in /dev. This is a device "make" file for devices such as tapes, disks, terminal multiplexors, printers, graphics/windows, etc. using the ***mknod*** command. If you're adding both 1/4" and 1/2" tape drives you need to ***MAKEDEV st0*** (1/4") first, then ***MAKEDEV xt0*** (1/2"), as the first links **xt** devices to the similar **st** devices.

If you're adding a new disk that has not yet been formatted, you will need to format it. This will require that there be an entry in **/etc/format.dat**, e.g. for the Sun supplied 424MB disk the entry would be:

```
disk_type = "SUN0424" \
        : ctlr = SCSI : fmt_time = 4 \
        : trks_zone = 9 : asect = 2 \
        : ncyl = 1151 : acyl = 2 : pcyl = 2500 : nhead = 9 : nsect = 80 \
        : rpm = 4400 : bpt = 26000
```

9.1.3 Files that control terminal logins

Besides having the proper device files in **/dev** you control terminal logins with the files **ttys**, **ttytab**, **gettytab**, and **termcap** in **/etc**.

9.1.3.1 ttytab

The entries in **/etc/ttytab** are used to turn ports on/off and denotes them as being **secure** or **unsecure** for root login. The entry also specifies the program to run, usually ***getty*** with the corresponding entry in **/etc/gettytab** to control the login, and expected terminal type, which must match an entry in **/etc/termcap**, e.g.

# name	getty	type	status	comments	
console	"/usr/etc/getty std.9600"	sun	on	local	secure
ttya	"/usr/etc/getty std.9600"	unknown	off	local	unsecure
ttyb	"/usr/etc/getty std.9600"	unknown	off	remote	unsecure

The file **ttys** is derived from **ttytab** by ***init***, and should not be edited.

```
12console
02ttya
02ttyb
```

For Ultrix **ttys** is more like **ttytab** in SunOS 4.1.X, and there is no ttytab.

Should you modify **ttytab** you will need to interrupt ***init*** to get it to reread this file. This can be done with the ***kill*** command:

```
# kill -HUP 1
```

where the **-HUP** (or **-1**) option (hangup) interrupts, but doesn't kill ***init***.

9.1.3.2 termcap

/etc/termcap is a data base of descriptions of terminal capabilities (BSD, SunOS 4.1.X). It is one large file with entries for all terminal types, e.g. the vt100 description would be similar to:

```
d0|vt100|vt100-am|vt100am|dec vt100:\
        :do=^J:co#80:li#24:cl=50\E[;H\E[2J:sf=5\ED:\
        :le=^H:bs:am:cm=5\E[%i%d;%dH:nd=2\E[C:up=2\E[A:\
        :ce=3\E[K:cd=50\E[J:so=2\E[7m:se=2\E[m:us=2\E[4m:ue=2\E[m:\
        :md=2\E[1m:mr=2\E[7m:mb=2\E[5m:me=2\E[m:is=\E[1;24r\E[24;1H:\
        :rf=/usr/share/lib/tabset/vt100:\
        :rs=\E>\E[?3l\E[?4l\E[?5l\E[?7h\E[?8h:ks=\E[?1h\E=:ke=\E[?1l\E>:\
        :ku=\EOA:kd=\EOB:kr=\EOC:kl=\EOD:kb=^H:\
        :ho=\E[H:k1=\EOP:k2=\EOQ:k3=\EOR:k4=\EOS:pt:sr=5\EM:vt#3:xn:\
        :sc=\E7:rc=\E8:cs=\E[%i%d;%dr:
```

9.1.3.3 terminfo

/usr/share/lib/terminfo/[1-9,A-Z,a-z]/* are a collection of files that describe terminal capabilities (SysV, SunOS 4.1.X, SunOS 5.X). These files are functionally equivalent to the entries in **/etc/termcap**.

9.1.3.4 gettytab

The **gettytab** file is a database used to describe terminal lines. The std.9600 entry referred to in **/etc/ttytab** above might look something like:

```
2|std.9600|9600-baud:\
    :sp#9600:
```


You could modify the **default** entry to write white on black and personalize your login message with an entry like:

```
default:\
    :ap:lm=\E[q\r\n%h login\72 :sp#9600:\
    im=\r\nUTS Workstation Lab\n:
```

9.1.3.5 getty

/etc/getty is the terminal login program. It is started by ***init***, reads **gettytab**, monitors the terminal line, and invokes the ***login*** program when a connection is made.

9.1.4 Adding a Modem

Under SunOS 4.1.X it is not necessary to modify the ***kernel*** when adding a standard modem, as it was under earlier versions of SunOS. However, if you add modem boards you may still need to modify the kernel. The new feature in SunOS4.1.x is the ***ttysoftcar*** command. You will need to make sure the device file exists for the modem and that the configuration files have been properly changed. The steps to take are:

1.	cd /dev	- go to the devices directory
2.	mknod device_name c major# minor#	- create the special device file
3.	vi /etc/ttytab	- edit and change, as needed
4.	kill -HUP 1	- send ***init*** a hangup signal
5.	ttysoftcar -a	- reset the ttys to their appropriate values, based on the /etc/ttytab "status" entry
6.	vi /etc/remote	- edit as needed

9.1.4.1 Make the device node

The ***mknod*** command builds the special file that you want the kernel to recognize as the modem, **cua0**. It should be a character type file with major and minor numbers of **12** and **128**, respectively. This corresponds to the same physical line as **ttya**, with major and minor numbers of **12** and **0**, respectively. The commands to create the device and set the proper permissions and ownerships are:

```
# mknod cua0 c 12 128
# chmod 600 cua0
# chown uucp cua0
```

9.1.4.2 Edit /etc/ttytab

```
ttya    "/usr/etc/getty std.19200"   sun       on    local    unsecure
cua0    "/usr/etc/getty std.19200"   unknown   off   remote   unsecure
```

9.1.4.3 Reinitialize init

Send ***init*** a hangup signal so that it will reread its initialization files.

```
# kill -HUP 1
```


9.1.4.4 Execute ttysoftcar

Kill off the ***getty*** process for this line, should one exist, then execute ***ttysoftcar*** to set the software carrier detect, i.e.:

```
# /usr/etc/ttysoftcar -a
```

Should you forget to kill the ***getty*** process ***ttysoftcar*** may hang. In the file **/etc/rc** uncomment the line:

```
/usr/etc/ttysoftcar -a > /dev/null 2>&1
```

9.1.4.5 Edit /etc/remote

Edit **/etc/remote** add the line:

```
cua0:dv=/dev/cua0:br#19200
```

You can then use ***/bin/tip*** or ***/bin/cu*** to access the line, e.g.:

```
# tip cua0
```

You exit tip and cu with "**~.**".

9.1.5 SCSI Device Designation and Description

SCSI, Small Computer System Interface, has become the standard for connecting peripheral devices: disks, tapes, cdroms, optical drives, etc. It was intended to provide device independence for a wide range or peripherals to the host computer. Devices conforming to the SCSI standard should be able to plug directly to the host's SCSI bus without requiring changes to the system hardware or software. There are several different versions of the standard (and as many implementations as there are manufacturers). The original standard, now called SCSI-1 or classical SCSI, will allow synchronous data transfers up to 5 MBytes/sec of 8 bit data. The current standard, SCSI-2, also known as Fast SCSI, and includes Wide SCSI. Fast SCSI allows asynchronous data transfers up to 10 MBytes/sec. SCSI-1 and SCSI-2 devices can co-exist on the same bus using a 50-conductor cable. Wide SCSI uses additional lines to transfer 16 or 32 data bits at a time, effectively doubling or quadrupling the maximum Bus transfer speed, up to 40 MBytes/sec, while using a 68-conductor cable. Narrow and wide SCSI devices can be mixed if the cable connections are made consistent.

SCSI-1 and SCSI-2 allow cable lengths totalling up to 6 meters, including internal cabling. Differential SCSI allows longer cable lengths, up to 25 meters. You can't directly mix differential and single-ended (classical) SCSI devices on the same SCSI bus without special translation devices.

SCSI devices are daisy-chained, and should be terminated on both ends of the chain. Classical SCSI uses passive terminators. Fast SCSI requires active terminators. These use voltage regulation to provide a more consistent termination resistance. Differential SCSI uses passive termination, but a different kind from classical SCSI.

The classical SCSI bus can accommodate 8 target devices, numbered 0 through 7. The last number, 7, is reserved for the connection between the bus and the system itself. Of the remaining seven the Sun kernel configuration file (SunOS 4.1.X) assumes that the first four will be used by disk devices, the next two by tape devices, and the last for a CDROM device, i.e.:

- **Disk** ⇒ targets 0,1,2,3
- **Tapes** ⇒ targets 4 & 5
- **CDROM** ⇒ target 6

You can change these defaults and regen the kernel.

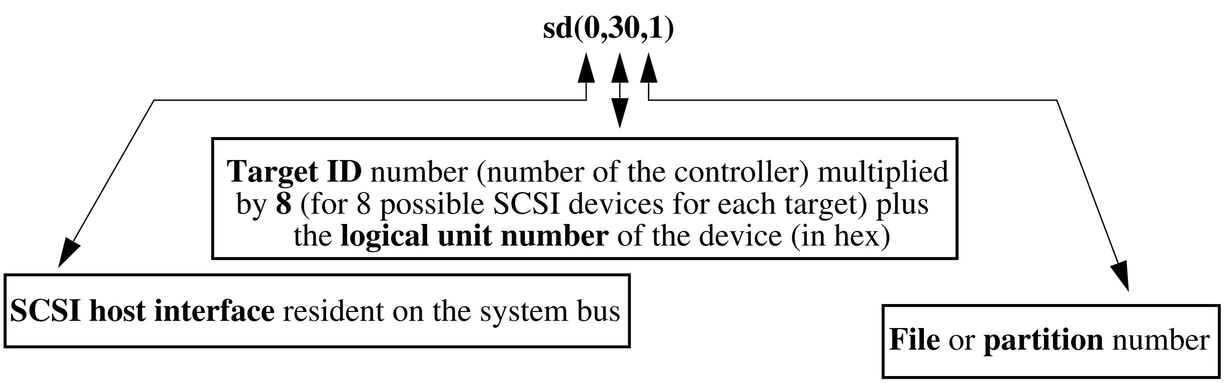
Wide SCSI allows 16 target devices, 0 through 15, with 15 the target ID of the host adaptor.

The larger the device number, the higher the priority of the device.

Only two SCSI devices can communicate at a time. So when a slow device is communicating with the host, i.e. a tape drive, the bus is effectively slowed down to the speed of the tape drive. When the tape drive is not in use the bus can resume a higher transfer rate of another device.

Earlier we mentioned that to boot a Sun4c machine from CDROM you would specify sd(0,6,2), or for a Sun4 machine you would specify sd(0,30,1) to the boot PROM. We will now look at where these numbers come from. As an example we look at sd(0,30,1).

FIGURE 9.1 **SCSI Device Designation**



SCSI devices may have their own controller to control a series of devices within the cabinet. For example you may have a cabinet with a controller and two disks that it controls. These external controllers can have 2 disks/target device; embedded controllers are limited to 1 disk/target device. Most SCSI devices today have embedded controllers, so you have one device for each SCSI ID on the Bus. Non-embedded devices are still available and may become more popular as the faster SCSI-2 protocol takes hold.

Sun's SCSI numbering scheme originated when non-embedded, external controller, devices were popular. One of their first products was called the "Shoebox" which had one controller to manage two disks, or a disk and tape drive. So when they mentioned disk drives as sd0 and sd1 they were talking about disks with the same SCSI host interface, but with different logical unit numbers, both with SCSI ID 0, but drive numbers 0 and 1. As embedded controllers became the norm, people generally referred to sd0 as the disk at SCSI ID 0 and sd1 as the disk at SCSI ID 1. But this is wrong, as they were both attached to the same SCSI controller. By using the drive number to specify both the SCSI ID and the disk number Sun was confusing people. Later Sun began using 3 octal numbers to represent the SCSI devices in the kernel configuration file, e.g.:

```
disk        sd3 at sc0 drive 011 flags 0
```

Here, in 011, the first number, 0, represents the SCSI interface number, the second number, 1, indicates the SCSI controller (target ID), and the last number, 1, indicates the drive number. A flag of 0 indicates a disk device, and 1 a tape device.

With embedded controllers the drive number is always 0, so people tend to refer to the device by it's SCSI target ID only, further confusing the issue.

The following tables map out the relationship between SCSI target ID and Unix disk number. In these tables **LUN** stands for **logical unit number**.

TABLE 9.1 **SCSI Target IDs, part 1**

External Controller	**Target ID**	**0**		**1**		**2**		**3**	
	LUN	0	1	0	1	0	1	0	1
	UNIX sd#	0	1	2	3	4	5	6	7
Embedded Controller	**Target ID**	**0**		**1**		**2**		**3**	
	LUN	0		0		0		0	
	UNIX sd#	0		2		4		6	

TABLE 9.2 **SCSI Target IDs, part 2**

Target ID	**0**		**1**		**2**		**3**		**4**		**5**		**6**	
LUN	0	1	0	1	0	1	0	1	0	NA	0	NA	0	NA
SCSI Drive #	0	1	8	9	16	17	24	25	32		40		48	
SCSI Hex#	0	1	8	9	10	11	18	19	20		28		30	
UNIX sd#	0	1	2	3	4	5	6	7	st0		st1		sr0	

To go back to our example then, we see from the second table that the CDROM device at SCSI target ID 6, **sr0**, has the number 30_{16}, so to boot from the second file on this device we would use: **sd(0,30,1)**.

9.1.6 Kernel Configuration File

The Sun3 and Sun4 GENERIC kernel configuration files have entries to describe the attached devices of the form:

```
# Support for the SCSI-2 host adapter with 2 disks and 1 1/4" tape
# on the first SCSI controller, 2 disks and 1 1/4" tape on the second
# SCSI controller, 2 embedded SCSI disks, and a CD-ROM drive.
controller          sc0 at vme24d16 ? csr 0x200000 priority 2 vector scintr 0x40
tape                st0 at sc0 drive 040 flags 1
```

tape	st1 at sc0 drive 050 flags 1
disk	sr0 at sc0 drive 060 flags 2
disk	sd0 at sc0 drive 000 flags 0
disk	sd1 at sc0 drive 001 flags 0
disk	sd2 at sc0 drive 010 flags 0
disk	sd3 at sc0 drive 011 flags 0
disk	sd4 at sc0 drive 020 flags 0
disk	sd6 at sc0 drive 030 flags 0

where the our example CDROM drive, sr0, has a target ID of 6 and an LUN of 0, or 060. This device would then be referenced by the SCSI Hex # of 30, i.e.:

target ID (6) * 8 + LUN (0) = $48_{10} = 30_{16}$

So you would boot from second file on this device as:

sd(0,30,1)

SPARCstation (Sun4c) devices are treated slightly differently. The internal (supplied) disk is target 3. So to avoid confusion (?) they remapped the target device numbers, as follows.

	Sun3/Sun3x/Sun4	Sun4c
Target 3	sd6⇒sd(0,18,0)	sd0⇒sd(0,0,0)
Target 1	sd2⇒sd(0,8,0)	sd1⇒sd(0,1,0)
Target 2	sd4⇒sd(0,10,0)	sd2⇒sd(0,2,0)
Target 0	sd0⇒sd(0,0,0)	sd3⇒sd(0,3,0)

The **NVRAM** on the CPU board is used to re-map the target SCSI ID numbers. The ordering of the targets is set with the parameter **sd-targets**, i.e.:

sd-targets 3 1 2 0 4 5 6 7

The Sun4c GENERIC kernel configuration file then has different entries to describe the attached peripherals, i.e.:

```
# The following section describes SCSI device unit assignments.
scsibus0 at esp                        # declare first scsi bus
disk sd0 at scsibus0 target 3 lun 0    # first hard SCSI disk
disk sd1 at scsibus0 target 1 lun 0    # second hard SCSI disk
disk sd2 at scsibus0 target 2 lun 0    # third hard SCSI disk
disk sd3 at scsibus0 target 0 lun 0    # fourth hard SCSI disk
tape st0 at scsibus0 target 4 lun 0    # first SCSI tape
tape st1 at scsibus0 target 5 lun 0    # second SCSI tape
disk sr0 at scsibus0 target 6 lun 0    # CD-ROM device
```

Sun doesn't support mixing embedded and non-embedded SCSI devices on the same host. Sometimes it works, and sometimes it doesn't. If it doesn't work, don't expect to get any help from Sun.

The SCSI bus must be terminated and this termination should be at the end of the bus line. The maximum length of all the SCSI cables, including internal cabling at the devices, is 6 meters.

9.2 SunOS 5.X

For most peripherals you will not need to add a device driver, as they are already included in the operating system. For those devices that do require a new driver, the driver and instructions should accompany the system. New device drivers should come in System V package format, so that you can install them with the ***pkgadd*** command or the Software Manager tool. We will discuss these commands in a latter chapter.

The next step is to get the system to recognize the device driver. If you touch the file **/reconfigure**, e.g.:

```
# touch /reconfigure
```

then when the system is rebooted it will reconfigure the kernel, as discussed in the chapter on Kernel Configuration.

Now you can shutdown the system, power it off, add the new hardware, and then boot new system. You will want to be sure to power on the peripheral devices before the CPU to insure that the system will recognize the peripherals.

If you did not create the **/reconfigure** file you can still reconfigure the kernel during the boot process by booting with the **-r** (reconfigure) option to the PROM boot command, i.e.:

ok boot -r

The kernel will then recognize the new drivers during the boot process and create the necessary device entries in **/devices** for the attached hardware using the ***drvconfig*** command.

You can examine the currently loaded device driver list with the **-i** option to the ***sysdef*** command.This will output several pages of information including the list of tunable parameters for the kernel shown in the last section.

9.3 IRIX 5.X

As mentioned in the chapter on **Kernel Configuration**, when new hardware is discovered during the boot process the kernel is automatically regenerated to recognize the new devices with the **/etc/init.d/autoconfig** script at run-level 2.

CHAPTER 10 Special Files

10.1 Special Files

UNIX **devices** are treated as **files**, in that I/O to devices is done in the same way as I/O to files. A file referring to a device is a special file in that it doesn't contain data (as a regular file) or information about other files (as a directory does). A special file informs the operating system about the location of the device it's associated with and the means by which it can communicate with the device. Special files are created with the ***mknod*** command and are stored in **/dev** (also **/devices** for SunOS 5.X). Once a special file exists I/O is performed with the device simply by reading or writing to the associated file.

10.1.1 Block and character devices

All I/O devices are classified as either **block** or **character** (**raw**) devices. The block special device causes the I/O to be buffered in large pieces. The character (raw) device causes I/O to occur one character (byte) at a time. Some devices, such as disks and tapes, can be both block and character devices, and must have entries for each mode. Terminals operate in character mode. The first entry in the permission field indicates either **b**→block, or **c**→character.

10.1.2 Major and minor devices

The **major** number identifies the kernel driver that communicates with the device. The **minor** number identifies the location of the device, i.e. it divides a physical device into logical devices.

The major number directs you to the proper controller and mode. Minor device numbers 0 → 7 refer to portions of drive 0; minor devices 8→15 refer to drive 1, etc.

For SunOS 5.X the system is responsible for assigning unused major numbers when you add a device, so these should not be hard-coded in the drivers. Minor numbers are assigned by the driver.

10.1.3 The mknod command

The ***mknod*** command is used to build special files, e.g. under SunOS 4.1.X:

```
# mknod cua0 c 12 128
```

generates a special file named **cua0**; it is a character-type device, **12** specifies the terminal controller, and **128** indicates that it will be a dial-out device. This uses the same physical line as **ttya**, the dial-in device with major and minor numbers **12** and **0**, respectively.

10.2 SunOS 4.X

Under SunOS 4.X the **/dev** directory contains the special files. The standard devices are installed with **/dev/MAKEDEV**, which uses the ***mknod*** command to create them. For a SCSI disk, represented by **sd** and **rsd** for the block and character devices, respectively, we have:

```
0 brw-r----- 1 root operator  7,  0 Apr 30  1995 /dev/sd0a
0 brw-r----- 1 root operator  7,  1 Apr 30  1995 /dev/sd0b
0 brw-r----- 1 root operator  7,  2 Apr 30  1995 /dev/sd0c
0 brw-r----- 1 root operator  7,  3 Apr 30  1995 /dev/sd0d
0 brw-r----- 1 root operator  7,  4 Apr 30  1995 /dev/sd0e
0 brw-r----- 1 root operator  7,  5 Apr 30  1995 /dev/sd0f
0 brw-r----- 1 root operator  7,  6 Apr 30  1995 /dev/sd0g
0 brw-r----- 1 root operator  7,  7 Apr 30  1995 /dev/sd0h
0 crw-r----- 1 root operator 17,  0 Apr 30  1995 /dev/rsd0a
0 crw-r----- 1 root operator 17,  1 Apr 30  1995 /dev/rsd0b
0 crw-r----- 1 root operator 17,  2 Apr 30  1995 /dev/rsd0c
0 crw-r----- 1 root operator 17,  3 Apr 30  1995 /dev/rsd0d
0 crw-r----- 1 root operator 17,  4 Apr 30  1995 /dev/rsd0e
0 crw-r----- 1 root operator 17,  5 Apr 30  1995 /dev/rsd0f
0 crw-r----- 1 root operator 17,  6 Apr 30  1995 /dev/rsd0g
0 crw-r----- 1 root operator 17,  7 Apr 30  1995 /dev/rsd0h
```

10.3 SunOS 5.X

For SunOS 5.X if you install the binary compatibility package the SunOS 4.X compatible names are created in **/dev**, but these are just links to the SysV type logical names in sub-directories, which are in turn links to the actual device names in **/devices**. The logical names are what you would generally use.

Under SunOS 5.X **/dev** is no longer a flat space, but has sub-directories for the types of devices:

dsk	block disk devices
rdsk	raw or character disk devices
rmt	tape devices
term	serial line devices
cua	dial-out modems
pts	pseudo terminals
fbs	frame buffers
sad	STREAMS administrative driver

The SunOS 4.X compatible names link you to the SysV type names, e.g.:

```
2 lrwxrwxrwx 1 root root 13 Mar  1 09:54 /dev/sd0a -> dsk/c0t3d0s0
2 lrwxrwxrwx 1 root root 12 Mar  1 09:54 /dev/rsd0a -> rdsk/c0t3d0s0
2 lrwxrwxrwx 1 root root 51 Dec 22 10:01 /dev/dsk/c0t3d0s0 ->
    ../../devices/sbus@1,f8000000/esp@0,800000/sd@3,0:a
2 lrwxrwxrwx 1 root root 55 Dec 22 10:01 /dev/rdsk/c0t3d0s0 ->
    ../../devices/sbus@1,f8000000/esp@0,800000/sd@3,0:a,raw
```

For the ***logical*** names the raw devices are in /dev/rdsk and the block devices are in /dev/dsk. The controller, target number, disk number, and slice number (partition) are described by the:

c# **t#** **d#** **s#**

entries, respectively, in the name, e.g.:

/dev/dsk/c0t3d0s0

which corresponds to the SunOS 4.X compatible entry for a SPARCstation of:

/dev/sd0a

and the physical name:

/devices/sbus@1,f8000000/esp@0,800000/sd@3,0:a

For the ***physical*** names:

sbus@1 indicates the slot number
esp@0 indicates the SCSI Host Adaptor
sd@3 indicates the SCSI Target Number
0 is the SCSI Logical Unit Number
a is the partition

10.3.1 Reconfiguring the /devices directory (SunOS 5.X)

Should you need to reconfigure the /devices directory you can do this with the ***drvconfig*** command. This should create the /devices directory tree from the attached hardware. It uses the **dev_info** tree of the kernel. The devices should be powered on when you run this command. Normally this is done for you whenever a new driver is installed with the ***add_drv*** utility and you reboot the system with the **-r** option. drvconfig uses the file **/etc/minor_perm** to determine the permissions to apply to the devices and the file **/etc/name_to_major** to assign major device numbers.

10.4 IRIX 5.X

In IRIX the physical devices are in **/dev** in a format that's sort of a merge of the BSD and SysV.4 styles. There are numerous files in /dev and a few sub-directories to separate the different types of devices, including:

abi	Application Binary Interface
dsk	block disk devices (ips, dks, xyl)
hl	files used by GTX hardware
pts	pseudo terminals
rdsk	raw or character disk devices
rmt	tape devices
sad	STREAMS administrative devices

SCSI disk devices have entries in **/dev/dsk** and **/dev/rdsk** of the form:

dks<controller-#>d<drive-#.>{s<partition-#>|vh|vol}

where **vh** represents the volume header (partition 8) and **vol** (partition 10) is the entire drive, e.g.:

```
dsk/
    0 brw-------  2 root  sys  128, 16 Mar 24 09:37 dks0d1s0
    0 brw-r-----  2 root  sys  128, 17 Mar 24 09:37 dks0d1s1
    0 brw-------  2 root  sys  128, 22 Mar 24 09:37 dks0d1s6
    0 brw-------  1 root  sys  128, 23 Mar 24 09:37 dks0d1s7
rdsk/
    0 crw-------  2 root  sys  128, 16 Apr  9 03:10 dks0d1s0
    0 crw-------  2 root  sys  128, 17 Mar 24 09:37 dks0d1s1
    0 crw-------  2 root  sys  128, 22 Mar 24 09:37 dks0d1s6
    0 crw-------  1 root  sys  128, 23 Mar 24 09:37 dks0d1s7
    0 crw-------  2 root  sys  128, 24 Mar 24 09:37 dks0d1vh
    0 crw-------  1 root  sys  128, 26 Mar 24 09:37 dks0d1vol
```

10.5 Ultrix and Digital UNIX

The physical devices are located in **/dev**, in a BSD style, and the **MAKEDEV** program is used to install the special files. For Digital UNIX there are also a few SysV.4 style sub-directories for **sad** and **streams**. SCSI disks are designated as **rz** and **rrz** for the block and character devices, respectively, with, for Digital UNIX:

```
0 brw-------  1 root  system  8,  0 May  7  1995 rz0a
0 brw-------  1 root  system  8,  1 May  7  1995 rz0b
0 brw-------  1 root  system  8,  2 May  7  1995 rz0c
0 brw-------  1 root  system  8,  3 May  7  1995 rz0d
0 brw-------  1 root  system  8,  4 May  7  1995 rz0e
0 brw-------  1 root  system  8,  5 May  7  1995 rz0f
0 brw-------  1 root  system  8,  6 May  7  1995 rz0g
0 brw-------  1 root  system  8,  7 May  7  1995 rz0h
0 crw-------  1 root  system  8,  0 May  7  1995 rrz0a
0 crw-------  1 root  system  8,  1 May  7  1995 rrz0b
0 crw-------  1 root  system  8,  2 May  7  1995 rrz0c
0 crw-------  1 root  system  8,  3 May  7  1995 rrz0d
0 crw-------  1 root  system  8,  4 May  7  1995 rrz0e
0 crw-------  1 root  system  8,  5 May  7  1995 rrz0f
0 crw-------  1 root  system  8,  6 May  7  1995 rrz0g
```

For Ultrix the major and minor numbers are different for block and character devices.

CHAPTER 11

System Directories

11.1 System Directories

The UNIX file system is laid out in a tree type structure, with the top level directory being the root. This contains the **kernel**, or the **/kernel** (SunOS 5.X) and **/platform** (SunOS 5.5+) directories, and the directories required by the operating system.

11.2 / - root

11.2.1 SunOS 4.1.X

SunOS 4.1.X standalone machine might have:

```
   1 lrwxrwxrwx  1 root     wheel          7 Jan 25  1995 bin ->
                                                          usr/bin
 120 -r--r--r--  1 root     bin       110352 Jan 25  1995 boot
  11 drwxr-sr-x  2 root     staff      11264 Apr 17 14:57 dev/
   3 drwxr-sr-x 11 root     staff       2560 May 23 13:59 etc/
   1 drwxr-sr-x  4 root     wheel        512 Jan 25  1995 export/
   1 drwxr-sr-x 10 root     ats          512 Oct  4  1995 home/
 256 -rwxr-xr-x  1 root     wheel     252913 Jan 25  1995 kadb*
   1 lrwxrwxrwx  1 root     wheel          7 Jan 25  1995 lib ->
                                                          usr/lib
   8 drwxr-xr-x  2 root     wheel       8192 Jan 25  1995 lost+found/
   1 drwxr-sr-x  2 root     staff        512 Oct 14  1994 mnt/
   1 drwxr-sr-x  2 root     wheel        512 Jan 25  1995 pcfs/
   1 drwxr-sr-x  2 root     staff        512 Jan 25  1995 sbin/
   1 lrwxrwxrwx  1 root     wheel         13 Jan 25  1995 sys ->
                                                          ./usr/kvm/sys
   4 drwxrwsrwt  4 root     wheel         80 Jul 12 04:15 tmp/
   1 drwxr-xr-x 26 root     wheel       1024 Feb  8  1995 usr/
   1 drwxr-sr-x  9 root     staff        512 Jan 25  1995 var/
1424 -rwxr-xr-x  1 root     daemon   1449841 Jan 25  1995 vmunix*
1712 -rwxr-xr-x  1 root     wheel    1740330 Jan 25  1995 vmunix.gen*
```


11.2.2 SunOS 5.X

A SunOS 5.X standalone machine would not have the kernels, but rather kernel directories, and would have a different boot file and a few additional directories, e.g.:

```
  2 drwxr-xr-x   2 root     root         512 Dec  7  1995 TT_DB/
  2 drwxr-xr-x   4 root     staff        512 Dec  7  1995 acs/
  2 lrwxrwxrwx   1 root     root           9 Dec  7  1995 bin ->
                                                  ./usr/bin/
  2 drwxr-xr-x   2 root     staff        512 Jun 27 15:35 cdrom/
  8 drwxr-xr-x  16 root     sys         4096 May 23 09:11 dev/
  2 drwxr-xr-x   5 root     sys          512 Dec  7  1995 devices/
  8 drwxr-xr-x  28 root     sys         4096 Jul 11 10:14 etc/
  2 drwxr-xr-x   4 root     sys          512 Dec  7  1995 export/
  2 drwxr-xr-x   6 root     sys          512 Jun 26 15:25 home/
  2 drwxr-xr-x   6 root     root        1024 Jun 25 08:46 jumpstart/
  2 drwxr-xr-x   9 root     sys          512 Dec  7  1995 kernel/
  2 lrwxrwxrwx   1 root     root           9 Dec  7  1995 lib ->
                                                  ./usr/lib/
 16 drwx------   2 root     root        8192 Dec  7  1995 lost+found/
  2 drwxr-xr-x   2 root     sys          512 Dec  7  1995 mnt/
  2 dr-xr-xr-x   2 root     root         512 Dec  7  1995 net/
  2 drwxrwxr-x  19 root     sys          512 Jun 25 08:43 opt/
  2 drwxr-xr-x   5 root     root         512 Nov  9  1995 pcnfs/
  2 drwxr-xr-x   3 root     sys          512 Dec  7  1995 platform/
 32 dr-xr-xr-x   2 root     root       16064 Jul 12 10:50 proc/
  2 drwxr-xr-x   2 root     sys          512 Dec  7  1995 sbin/
  8 drwxrwsrwt   6 sys      sys          755 Jul 12 10:40 tmp/
  2 drwxrwxr-x  31 root     sys         1024 Jun 27 13:19 usr/
  2 drwxr-xr-x  21 root     sys          512 Dec  7  1995 var/
  0 dr-xr-xr-x   6 root     root         512 May 23 09:12 vol/
```


11.2.3 IRIX 5.X

IRIX is similar to SunOS 5.X, but its kernel is a file in the root directory, ***/unix***, rather than under /kernel.

```
   1 drwxr-xr-x    2 root     sys        512 Mar  4  1994 CDROM/
   1 lrwxr-xr-x    1 root     sys          7 Apr 11  1994 bin -> usr/bin/
   1 lrwxr-xr-x    1 root     sys          4 Mar  4  1994 debug -> proc/
   6 drwxr-xr-x   15 root     sys       3072 Jul  9 09:56 dev/
   5 drwxr-xr-x   15 root     sys       2560 Jul  9 09:56 etc/
   1 drwxr-xr-x    2 root     sys        512 Mar 24  1995 lib/
  21 drwx------    2 root     sys      10752 Sep 23  1994 lost+found/
   1 drwxr-xr-x    6 root     sys        512 Mar 24  1995 opt/
  10 dr-xr-xr-x    2 root     sys       4848 Jul 12 11:05 proc/
   3 drwxr-xr-x    3 root     sys       1536 Mar 24  1995 sbin/
   1 drwxr-xr-x    2 root     sys        512 Mar 24  1995 stand/
   1 drwxrwxrwt    3 sys      sys        512 Jul 12 11:00 tmp/
6007 -rwxr-xr-x    1 root     sys    3075152 Jul  3 07:51 unix*
6007 -rwxr-xr-x    1 root     sys    3075156 Jul  3 07:55 unix.save*
   1 drwxr-xr-x   25 root     sys        512 Jul  9 09:56 usr/
   1 drwxr-xr-x   24 root     sys        512 Jun  8  1995 var/
```

11.2.4 Digital UNIX

```
   0 lrwxr-xr-x   1 root     system       7 May 23  1995 bin@ -> usr/bin/
   8 drwxr-xr-x   7 root     system    8192 Jul 12 07:41 dev/
   8 drwxr-xr-x  13 root     system    8192 Jul 12 10:41 etc/
7360 -rwxr-xr-x   1 root     system 7535240 Jul 25  1995 genvmunix*
   8 drwxr-xr-x  10 root     system    8192 Jul 11 15:34 home/
   0 lrwxr-xr-x   1 root     system       7 May 23  1995 lib@ -> usr/lib/
   8 drwxr-xr-x   2 root     system    8192 May 23  1995 mdec/
   8 drwxr-xr-x   2 root     system    8192 Jul 25  1995 mnt/
   8 drwxr-xr-x   2 root     system    8192 May 23  1995 opt/
  27 -rwxr-xr-x   1 root     system   27440 Jul 24  1995 osf_boot*
   8 dr-xr-xr-x   2 root     system    8224 Jul 12 11:08 proc/
   8 drwxr-xr-x  10 root     system    8192 May 24 13:17 sbin/
   8 drwxr-xr-x   2 root     system    8192 May 23  1995 subsys/
   0 lrwxr-xr-x   1 root     system       7 May 23  1995 sys@ -> usr/sys/
   8 drwxr-xr-x   5 root     system    8192 May 23  1995 tcb/
   8 drwxrwxrwt   4 root     system    8192 Jul 12 10:10 tmp/
   8 drwxr-xr-x  20 root     system    8192 May 22 14:45 usr/
   0 lrwxrwxrwx   1 root     system       7 May 23  1995 var@ -> usr/var/
8608 -rwxr-xr-x   1 root     system 8812896 Jun 20 11:15 vmunix*
8928 -rwx--x---   1 root     system 9134136 Jun 20 11:19 vmunix.save*
```

Where **/tcb** contains files and databases used with **enhanced security** for checking authorizations.

11.2.5 Ultrix

Ultrix is similar to SunOS 4.X with a few differences: the boot program is ***ultrixboot***; the kernel is still ***vmunix***, but the generic backup kernel is ***genvmunix***.

11.3 /etc - system and network configuration

/etc contains configuration files and networking programs that are used during the boot process and to control network access.

11.3.1 SunOS 4.1.X, configuration files

aliases	aliases.dir	aliases.pag	**bootparams**	**defaultdomain**
defaultrouter	ethers	**exports**	fbtab	format.dat
fstab	gettytab	**group**	**hostname.le0**	**hosts**
hosts.equiv	hosts.lpd	inetd.conf	magic	motd
mtab	netgroup	**netmasks**	networks	**passwd**
printcap	rc	rc.boot	rc.ip	**rc.local**
rc.single	remote	**resolv.conf**	rpc	sendmail.cf
services	shells	**syslog.conf**	ttys	**ttytab**

11.3.2 SunOS 5.X, configuration files and directories

aliases -> ./mail/aliases	asppp.cf*	auto_home
auto_master	bootparams	cron.d/
default/	**defaultdomain**	**defaultrouter**
dfs/	dt/	dumpdates
ethers	format.dat	fs/
group	**hostname.le0**	**hosts** -> ./inet/hosts
hosts.allow	hosts.deny	inet/
inetd.conf -> ./inet/inetd.conf	init.d/	**inittab**
issue	lib/	logindevperm
lp/	magic	mail/
mnttab	motd	net/
netconfig	netmasks -> ./inet/netmasks	networks -> ./inet/networks
nodename	nscd.conf	**nsswitch.conf**
nsswitch.files	nsswitch.nis	nsswitch.nisplus
ntp.conf	opt/	**passwd**
path_to_inst	profile	protocols -> ./inet/protocols
publickey*	rc0 -> ../sbin/rc0*	rc0.d/
rc1 -> ../sbin/rc1*	rc1.d/	rc2 -> ../sbin/rc2*
rc2.d/	rc3 -> ../sbin/rc3*	rc3.d/
rc5 -> ../sbin/rc5*	rc6 -> ../sbin/rc6*	rcS -> ../sbin/rcS*
rcS.d/	remote	**resolv.conf**
rmmount.conf	rmtab	rpc

saf/	security/	sendmail.cf -> mail/sendmail.cf
services -> ./inet/services	**shadow**	**shells**
skel/	ssh_config	ssh_host_key
ssh_host_key.pub	ssh_known_hosts	ssh_random_seed
sshd_config	**syslog.conf**	**system**
termcap -> ../usr/share/lib/termcap	ttydefs	utmp -> ../var/adm/utmp
utmpx -> ../var/adm/utmpx	**vfstab**	vold.conf
wtmp -> ../var/adm/wtmp	wtmpx -> ../var/adm/wtmpx	

11.3.3 IRIX 5.X, configuration files and directories

TIMEZONE	**aliases**	bootparams	bootptab
brutab	**config/**	cron.d/	default/
device.tab	ethers	**exports**	fscklogs/
fsd.tab	**fstab**	fstyp.d/	gettydefs
group	**hosts**	**inetd.conf**	**init.d/**
inittab	lastbackup	magic	mailcap
motd	mtab	netconfig	netgroup
netid	networks	**passwd**	printcap
protocols	rc0	rc0.d/	rc2
rc2.d/	rc3	rc3.d/	**resolv.conf**
rmtab	rpc	**sendmail.cf**	**services**
shadow	**syslog.conf**	ttytype	

11.3.4 Digital UNIX

TIMEZONE	acucap	**auth/**	disktab
exports	**fstab**	gettydefs	**group**
hosts	hosts.equiv	**inetd.conf**	**inittab**
lprsetup.dat	magic	motd	networks
ntp.conf	**passwd**	**printcap**	protocols
rc.config	remote	**resolv.conf**	**routes**
rpc	**sec/**	securettys	**services**
setup.conf	sia/	strsetup.conf	**svc.conf**
svcorder	sysconfigtab	**syslog.conf**	termcap@
zoneinfo/			

11.3.5 Ultrix

acucap	**aliases**	**auth**	**crontab**
disktab	dms	doconfig	dumpdates
elcsd.conf	**exports**	**fstab**	gettytab
group	**hosts**	**hosts.equiv**	**inetd.conf**
install_upgrade	krb.conf	motd	networks
ntp.conf	**passwd**	**printcap**	protocols
rc	**rc.local**	remote	**resolv.conf**
ris	rmtab	rpc	sec/
sendmail.cf	**services**	setld	setldlog
svc.conf	**syslog.conf**	termcap	**ttys**
utmp	zoneinfo/		

11.4 /usr - system programs, libraries, etc.

You don’t normally need to change these unless you want to change the functionality of a program, patch system programs, or plug security holes. Generally, you would install programs you write or port to the system in either **/usr/local** or **/opt/local**.

11.4.1 SunOS 4.1.X

5bin/	5include/	5lib/	adm -> ../var/adm/
bin/	boot -> ./kvm/boot/	demo/	diag/
dict/	etc/	export/	games/
hosts/	include/	kvm/	lang/
lddrv/	lib/	local/	man -> share/man/
mdec -> ./kvm/mdec/	net -> /var/net/	nserve -> ../etc/nserve/	openwin/
pub -> share/lib/pub/	sccs/	share/	spool -> ../var/spool/
src -> share/src/	stand -> ./kvm/stand/	sys -> kvm/sys/	tmp -> ../var/tmp/
ucb/	ucbinclude -> ./include/	ucblib -> lib/	xpg2bin/
xpg2include/	xpg2lib/		

11.4.2 SunOS 5.X

4lib/	5bin -> ./bin/	TT_DB/	adm -> ../var/adm/
aset/	bin/	ccs/	demo/
dict -> ./share/lib/dict/	dt/	include/	kernel/
lib/	local -> /opt/local/	mail -> ../var/mail/	man -> ./share/man/
net/	news -> ../var/news/	openwin/	platform/
preserve -> ../var/preserve/	proc/	pub -> ./share/lib/pub/	sadm/
sbin/	share/	snadm/	spool -> ../var/spool/
src -> ./share/src/	tmp -> ../var/tmp/	ucb/	ucbinclude/
ucblib/	vmsys/		

11.4.3 IRIX 5.X

Cadmin/	ToolTalk/	adm -> ../var/adm/	bin/
bsd/	catman/	cpu/	demos/
dist/	etc/	explorer/	gfx/
include/	lib/	local/	mail -> ../var/mail/
people/	preserve -> ../var/preserve/	relnotes/	sbin/
share/	spool -> ../var/spool/	src/	sysgen/
tmp -> ../var/tmp/			

11.4.4 Digital UNIX

adm -> ../var/adm/	bin/	ccs/	dict/
doc/	examples/	field/	include/
lbin/	lib/	local -> /home/local/	man -> share/man/
news -> ../var/news/	opt/	preserve -> ../var/preserve/	sbin/
share/	shlib/	skel/	spool -> ../var/spool/
sys/	tcb/	tmp -> ../var/tmp/	ucb -> ./bin/
var/			

11.4.5 Ultrix

adm@ -> var/adm	bin/	dict/	diskless/
etc/	examples/	field/	hosts/
include/	lib/	local/	man/
mdec/	preserve@ -> var/preserve	skel/	spool@ -> var/spool
src/	sys/	tmp@ -> var/tmp	ucb/
users/	var/		

CHAPTER 12 **User accounts**

12.1 User accounts

12.1.1 Registration

The user information is registered in the **passwd**, **group**, and, for SunOS 5.X, in the **shadow** files in **/etc**.

12.1.1.1 Password file - /etc/passwd

/etc/passwd contains 7 fields, each separated by "**:**", in the form:

login-id:password:user-id#:group-id#:User Info:home-dir:shell

where these fields represent:

- **login-id** 2→8 characters containing lower case alphabetic characters and numbers.
- **password** encrypted password, 13 characters, of which the first 2 are the salt. If this field is empty login does NOT prompt for a password. If this field contains 1→12 characters NO password will ever match.
- **user-id#** uid, numerical ID for the user, should be between 0 and 60000 (SunOS 4.1.X, Solaris 2.0-2.5). Solaris 2.5.1 uses a signed long for this value, MAXUID in /usr/include/sys/param.h, raising the limit to 2^{31}.
- **group-id#** gid, numerical ID for the group that the user belongs to, should be between 0 and 60000.
- **User Info** User's real name, etc.
- **home-dir** Path to the directory the user is logged in to.
- **shell** The user's initial shell program. The default shell if this is empty is /usr/bin/sh.

Valid entries within **passwd** would be:

```
sysdiag:*:0:1:System Diagnostic:/usr/diag/sysdiag:/usr/diag/sysdiag/sysdiag
frank:yPf3M5qMgglUc:101:10:Frank G Fiamingo:/home/tardis/frank:/usr/bin/csh
```

The **home directory**, field 6 of **/etc/passwd**, specifies the location of the user's home within the operating system. The user is placed here by the ***login*** program. For a normal login user this directory should be owned by the user.

The **shell**, field 7 of **/etc/passwd**, is the program run when the user logs in. Generally this is a shell that acts as a command interpreter, reading from a terminal and translating the commands into system actions, e.g. ***sh*** (Bourne shell), ***csh*** (C shell), or ***tcsh*** (extended C shell). Occasionally this is not a shell, but a stand-alone program, as in the sysdiag passwd entry given above. Here when you login as "sysdiag" you go directly into the systems diagnostics program.

For SunOS 4.1.X you would generally edit the **passwd** file using the ***vipw*** command. This saves a copy of **passwd** as **ptmp**, uses the ***vi*** editor by default (or the editor set by your VISUAL or EDITOR environment variable), and verifies the consistency of the root entry before writing the file back to passwd. The shell for the root account must be listed in **/etc/shells**, if the file exists. The **ptmp** file also serves as a lock against a simultaneous use of ***vipw***.

12.1.1.2 Group file - /etc/group

/etc/group contains 4 fields, each separated by a "**:**", in the form:

group-name:password:gid:comma-separated,list,of,names

where these fields represent:

- **group-name** Name of the group
- **password** If the password field is empty you are not prompted for a password when changing groups.
- **gid** Numerical ID for the group; should match the gid field for the passwd file.
- **list** comma-separated list of users who belong to this group.

Valid entries within ***group*** would be:

operator:*:5:frank,bobd

staff:*:10:

12.1.1.3 Shadow file - /etc/shadow (SunOS 5.X, IRIX 5.X)

SunOS 5.X uses additional security measures over the older OS. One of these is the **shadow** password scheme, which is used by default. The encrypted password is not kept in **/etc/passwd**, but rather in **/etc/shadow**. **/etc/passwd** has a placeholder, **x**, in this field. **passwd** is readable by everyone, whereas **shadow** is readable only by root. The shadow file also contains password aging controls.

/etc/shadow contains 9 fields, each separated by a "**:**", in the form:

login-id:password:lastchg:min:max:warn:inactive:expire:flag

where these fields represents:

- **login-id** login name
- **password** 13 character encrypted password
- **lastchg** number of days from Jan 1, 1970 to the last password change
- **min** minimum number of days required between password changes
- **max** maximum number of days the password is valid

- **warn** number of days before expiring the password that the user is warned
- **inactive** number of days of inactivity allowed for the user
- **expire** absolute date after which the login may no longer be used
- **flag** currently not used

The encrypted password field might also contain the entries:

NP	for no password is valid
LK	meaning the account is locked until the superuser sets a password

A typical **/etc/shadow** file might be:

```
root:st44wfkgx33qX::::::
daemon:NP:6445::::::
bin:NP:6445::::::
sys:NP:6445::::::
adm:NP:6445::::::
lp:NP:6445::::::
smtp:NP:6445::::::
uucp:NP:6445::::::
nuucp:NP:6445::::::
listen:*LK*:::::::
nobody:NP:6445::::::
noaccess:NP:6445::::::
```

The **shadow** password file is updated using the commands:

- ***passwd*** change the password and password attributes
- ***useradd*** add a new user
- ***usermod*** modify a user's login information
- ***userdel*** delete a user's login entry

If you presently have an **/etc/passwd** file under SunOS 4.X that you want to use with SunOS 5.X, you can use the ***pwconv*** command to convert the passwd file to the new style and create the **/etc/shadow** file.

The **/etc/shadow** file has specific fields to keep track of the last password change, the minimum and maximum time in days that the password is valid, the number of inactive days allowed between uses before the login ID is declared invalid, and an expiration date for the account. You can edit **/etc/shadow** and set these values, or use the ***useradd*** command to set limits on the account.

Sun recommends that you use the ***admintool*** or ***solstice*** utilities or the ***useradd*** command to add new users, rather than editing the **passwd** file. If you do edit the **passwd** file you'll want to use ***pwconv*** to update the ***passwd*** changes to the **shadow** file. The use of ***vipw*** is no longer recommended. It's included with the compatibility package, as ***/usr/ucb/vipw***, and you can still use it, but it does not update the shadow file, though it does remind you to do so.

12.2 Admittance - login procedure

Under SunOS 4.1.X ***init*** creates a process for each terminal port defined within **/etc/ttytab**. For each hardwired line it starts a ***getty*** process. For network ports init starts the ***inetd*** daemon process to monitor for telnet, ftp, etc. logins. When the user logs out init detects this event and restarts the getty process. Similarly, the ***getty*** process is used by IRIX, Digital UNIX, and Ultrix.

For SunOS 5.X ***init*** uses the **Service Access Facility** to control system access. We will look at this service in a latter chapter.

12.3 Password Aging, SunOS 4.1.X

With password aging you can set minimum and maximum lengths of time for which the password is valid. Only the superuser can change these values. Maximum time lengths force your users to change passwords regularly. Minimum lengths prevent them from quickly changing them back.

For SunOS 4.1.X password aging for a user is started with the ***passwd*** command, using either the ***-x*** (maximum) or ***-n*** (minimum) options and specifying a time limit in days and a user name. This will alter the encrypted password field by adding a comma and 2 digits to the end of it. The first digit is for the maximum time and the second for the minimum. For 14 days or less the digits are zero. For longer than 14 days add 1 for each 7 day period, after rounding up to the nearest whole week value. This means that you have a granularity of a week, with a minimum time of 2 weeks. To set a maximum time of 40 days, and a minimum time of 30 days, for the user frank, execute:

```
# passwd -x 40 frank
# passwd -n 30 frank
```

These numbers will be rounded to the next greatest whole week value, converted to weeks, and then have 2 subtracted. So the digit for maximum time will be 4, and that for the minimum time will be 3. You can set a maximum time without a minimum, but not the reverse. The next time the password is changed a 2 character time field will be appended to the encrypted password string, encoding the time into it. So the corresponding entry in **/etc/passwd** could be:

frank:yPf3M5qMgglUc**,437I**:101:10:Frank G Fiamingo:/home/tardis/frank:/usr/bin/csh

If there was no minimum then the 3 would be missing.

You can display the values the password aging fields with the ***-d*** option to ***passwd***, e.g.:

```
# passwd -d
frank           9/19/94         35          42
```

which displays the date the current password was chosen and the minimum and maximum ages allowed.

Unfortunately, password aging in SunOS 4.1.X works only with **/etc/passwd**, and not with NIS.

CHAPTER 13 Daily System Administration

13.1 User and Group Administration

For NIS (YP) networked machines this should be done on the NIS master. If you are using NIS+ then you will probably want to use ***admintool*** to make these changes, and this can be done from any networked machine as long as you are a member of the **sysadmin** group (gid=14).

13.1.1 SunOS 4.1.X

13.1.1.1 Adding Users

1. Edit the **/etc/passwd** file to add the user - use ***vipw***, as this program creates a lock file that prevents two people from trying to edit the password file at the same time. vipw also makes a copy of the original file in /etc/opasswd, and checks the consistency of the root password entry before saving the new version of the file.
2. Edit the **/etc/group** file to add the user to additional groups.
3. If you're using **NIS** update the databases on the server:
   # (cd /var/yp; make) -or- (cd /var/yp; make passwd)
4. Give the new user a **password** with the ***passwd*** command:
   # passwd username
   This will prompt you twice for the user's password, without echoing.
5. Change to what will be the new user's proposed parent directory:
   # cd /home/server
6. Create a directory for the new user:
   # mkdir username
7. Copy any startup files, e.g. ".login", ".cshrc" into this directory:
   # cp /usr/local/adm/users/.[a-zA-Z]* username
8. Set the proper user and group ownership of the directory and startup files:
   # chown -R username.groupnname username- SunOS 4.1.X
   # chown -R username:groupnname username- SunOS 5.X
9. Set the proper permissions on the directory and startup files:
   # chmod -R 700 username

13.1.1.2 Removing users

You can disable a user's login by editing **/etc/passwd** to change the encrypted password entry, or by removing the user's entire entry. If you're running with NIS you then need to remake the NIS databases before the change will take effect. To temporarily disable a user's login **replace the encrypted password field** with something between 1 and 12 characters. The normal entry has 13 characters; anything shorter (other than NULL) can't be matched by the login crypt program. To completely lock the user out also change their shell, e.g. to ***/bin/false***, so that it won't be valid. Also make sure that they're not running any background processes, ***cron*** processes, or ***at*** processes. Enhanced Security mode under Ultrix has a 24 character encrypted password field (2 salt plus 22 encrypted password characters) and allows passwords up to 16 characters.

13.1.1.3 Changing passwords

For root to change a user's password it's the same as creating one, as above. For a user to change their own password all they need to type is ***passwd***. The program will then prompt for their old password and twice for their new one. The new password is required to be at least 5 characters long if combined upper/lower case letters are used, and 6 characters long otherwise.

13.1.2 SunOS 5.X

The most convenient way to add or remove users and groups is to use the User Account Manager of ***admintool***. This OpenWindows GUI tool takes you through the necessary steps. We will look at admintool in a later chapter. These changes can also be made on the command line, as shown below.

13.1.2.1 Adding users

To add new users from the command line use ***useradd***. This updates the files **/etc/passwd** and **/etc/shadow**, and if necessary, **/etc/group**, and creates the **home** directory. You would execute this command in the form "***useradd*** [options] login-id", e.g.:

```
# useradd -u 1001 -g staff -d /export/home/frank -s /usr/bin/csh -d "Frank G Fiamingo" -m \
    -k /etc/skel frank
```

where the options used above refer to:

u	uid number
g	group name
d	home directory name
s	path to the shell
m	make the home directory
k	path to the skeleton dot files

The last step is to provide the user with a password, using the ***passwd*** command.

13.1.2.2 Adding groups

There is a command to add new groups, ***groupadd***. To add a group with gid 14 called **sysadmin**, you would execute:

```
# groupadd -g 14 sysadmin
```


13.1.2.3 Removing Users

You can use the ***passwd*** command to **lock** the password entry for a user to temporarily suspend their activities. This places ***LK*** in the password field of **/etc/shadow**. To remove a user completely use the command ***userdel***.

13.1.2.4 Modifying user and group entries

To change user and group entries use the commands ***usermod*** and ***groupmod***, respectively.

13.1.2.5 User Initialization Files

The **/etc/skel** directory contains default user initialization files used by ***useradd***. You can modify these as desired.

The **/etc/profile** is the system-wide Bourne and Korn shell profile script. It's executed before the user's **$HOME/.profile**. There is no similar system-wide script for the C shell under SunOS 4.1.X, though one can be set for the T-C shell, ***tcsh***. For SunOS 5.X if the user doesn't have their own .login, then ***csh*** will read **/etc/.login**.

13.2 Communicating with system users

13.2.1 The message of the day

The file **/etc/motd** this is printed by ***login*** on the terminal of each user that logs in. You can use this text file to let users know about significant changes on the system.

13.2.2 Broadcast messages

The programs wall and rwall allow you to write to all users terminals. This allows you to alert all users about immediate problems or impending shutdowns. You type in the message after invoking the program and end the message with Control-D (^D).

13.3 Running programs automatically, cron & at

cron executes periodic commands at specified times and dates. cron is a clock daemon that runs continuously on the system and schedules jobs to be run according to the ***crontab*** files. You should use the ***crontab*** command to update entries in the crontab database.

at executes a command once at a specified time.

Users are allowed to run the ***cron*** and ***at*** programs if their names are listed in the file **/var/spool/cron/[cron,at].allow** (SunOS 4.1.X) or **/etc/cron.d/[cron,at].allow** (SunOS 5.X). If this file doesn't exist then the file **/var/spool/cron/[cron,at].deny** (SunOS 4.1.X) or **/etc/cron.d/[cron,at].deny** (SunOS 5.X) is checked to see if permission should be denied. If neither file exists permission is refused for all but the superuser. If you wish to allow everyone permission create an empty [cron,at].deny file. Ultrix only allows the root user access to crontab.

The crontab files are kept in the directory **/var/spool/cron/crontabs** for both SunOS 4.1.X and 5.X. Each crontab file is named after the owner. Some typical entries in the **root** crontab file, /var/spool/cron/crontabs/**root**, might be:

```
5 0 * * * calendar -
15 0 * * * /usr/etc/sa -s >/dev/null
# save only last weeks worth of sendmail logs
5 4 * * 6 /usr/lib/newsyslog >/dev/null 2>&1
# backup file systems
10 0 * * 2-6 /usr/local/backup/cron-backup
```

There are 5 time fields and a command field to control and what program is executed by cron and when,

field	**values**
• minute	0 -> 59
• hour	0 -> 23
• date of month	1 -> 31
• month	1 -> 12
• day of week	0 -> 6 (0=Sunday)
• command	command or Bourne shell script

Time fields can contain single values, comma (**,**) separated values (match any listed values), hyphen (**-**) separated values (match any value in the range), or the wildcard (*****) (always match).

To edit a crontab file use the command "***crontab -e***". This will allow you to change the crontab file and will cause cron to re-read it when you're done. By default in Solaris 2.X ***crontab*** assumes the ***ed*** editor. What you set with your **EDITOR** environmental variable will override this.

To just list the contents of your crontab file use the command "***crontab -l***".

CHAPTER 14 Administration Tool & Solstice Adminsuite

14.1 Admintool

The Administration Tool, ***admintool***, uses a graphical user interface under OpenWindows, to allow you to administer a number of administrative databases on the network. It users a distributed administrative framework to allow you to perform system administrative functions over the network. You can add new systems, setup printers, and add new user accounts.

Members of the ***sysadmin*** group (gid 14) are allowed to modify the databases, both locally and remotely (pre SunOS 5.5), if they are also a member of the sysadmin group on the remote system. Members can create, delete, and modify the databases, while non-members have read-only permission on the databases. So, in general, if you are a member of the ***sysadmin*** group, you can run ***admintool*** under your own user id, and are not required to run it as root. NIS+ has its own method of security, so in addition to being a member of the sysadmin group one needs to have the appropriate permissions on the NIS+ tables to be changed. The sysadmin group, by default, does not exist on the system. You will need to create this group first if you want to use it.

With SunOS 5.5 the remote database functions have been relegated to Solstice Adminsuite, and Admintool only functions on local databases.

14.2 Solstice Adminsuite

With SunOS 5.5 Sun removed the network part of ***admintool*** and replaced it with Solstice Adminsuite, ***solstice***. This software comes on a separate CDROM and should be installed after the OS. It also requires a license, which you can readily get by returning the form supplied on the CDROM with your license information. You can also run the product in **DEMO** mode without the license. With ***solstice*** you can manage local and remote system databases, using NIS+, NIS, or the local files in /etc.

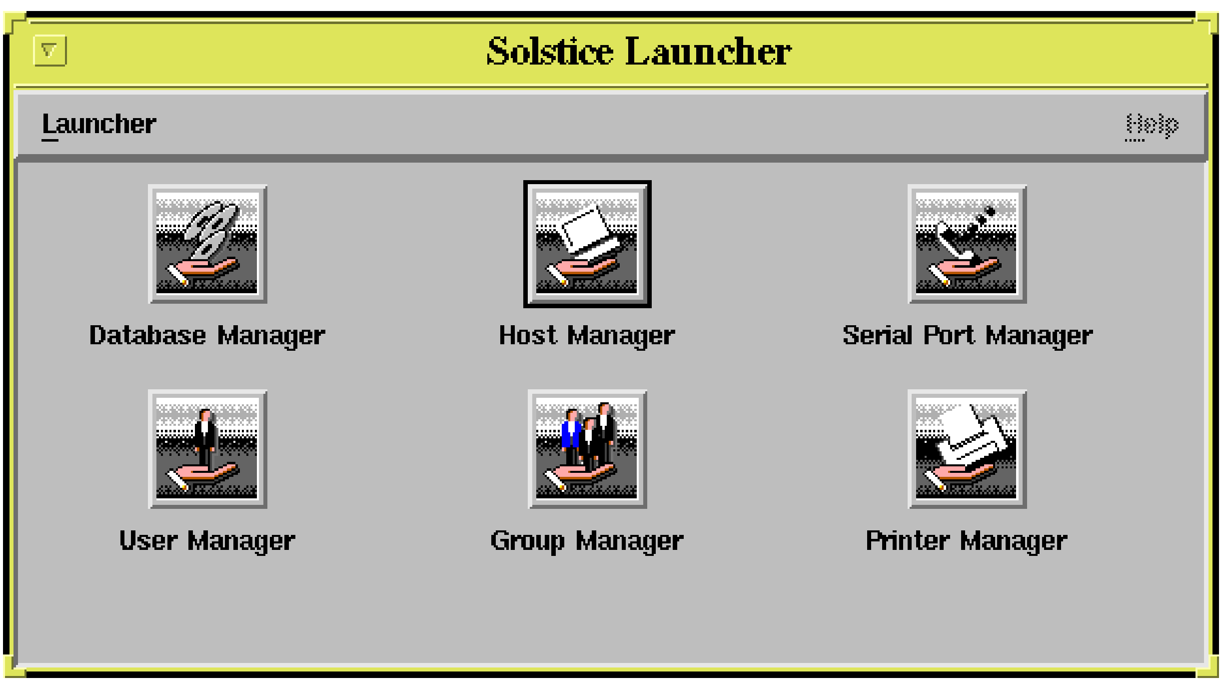
14.3 Services Managed

The SunOS 5.5+ version of ***admintool*** and ***solstice*** provide access to the following service functions:

TABLE 14.1 Services

Function	Administration Tool	Solstice AdminSuite
Database Manager	No	Yes
User Manager	Yes	Yes
Group Manager	Yes	Yes
Host Manager	Yes	Yes
Printer Manager	Yes	Yes
Serial Port Manager	Yes	Yes
Software Manager	Yes	No

When invoking ***solstice*** you should see a display similar to the following, from which you can select your choice of management tool.



14.3.1 Database Manager

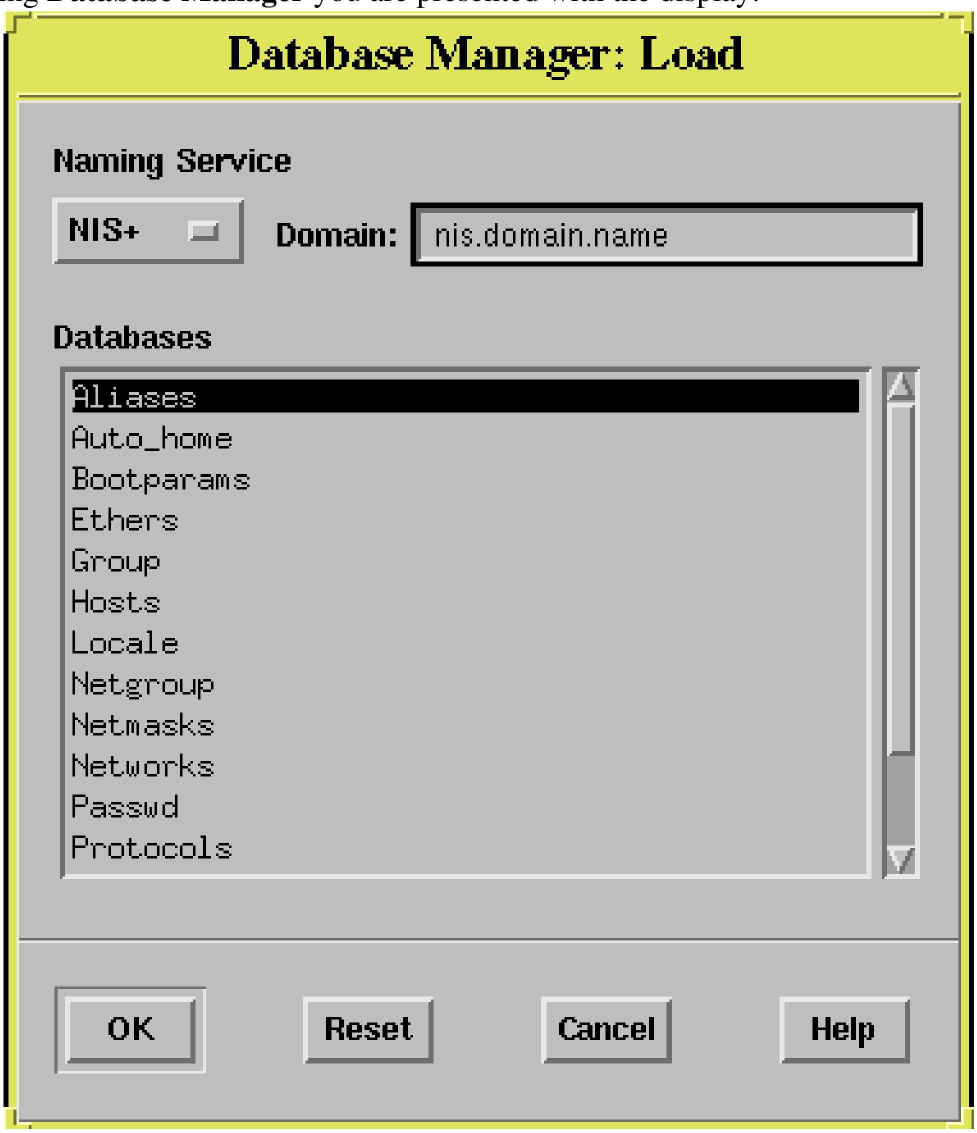
The **Database Manager** maintains the databases:

aliases	auto_home	bootparams	ethers
group	hosts	locale	netgroup
netmasks	networks	passwd	protocols
rpc	services	timezone	

using the naming services:

NIS+	Network Information Services Plus (replaces NIS)
NIS	Network Information Services (formerly known as YP)
None	text files in /etc

When selecting **Database Manager** you are presented with the display:



14.3.2 Host Manager

The **Host Manager** lets you add, delete, or modify the following information in the various host related databases:

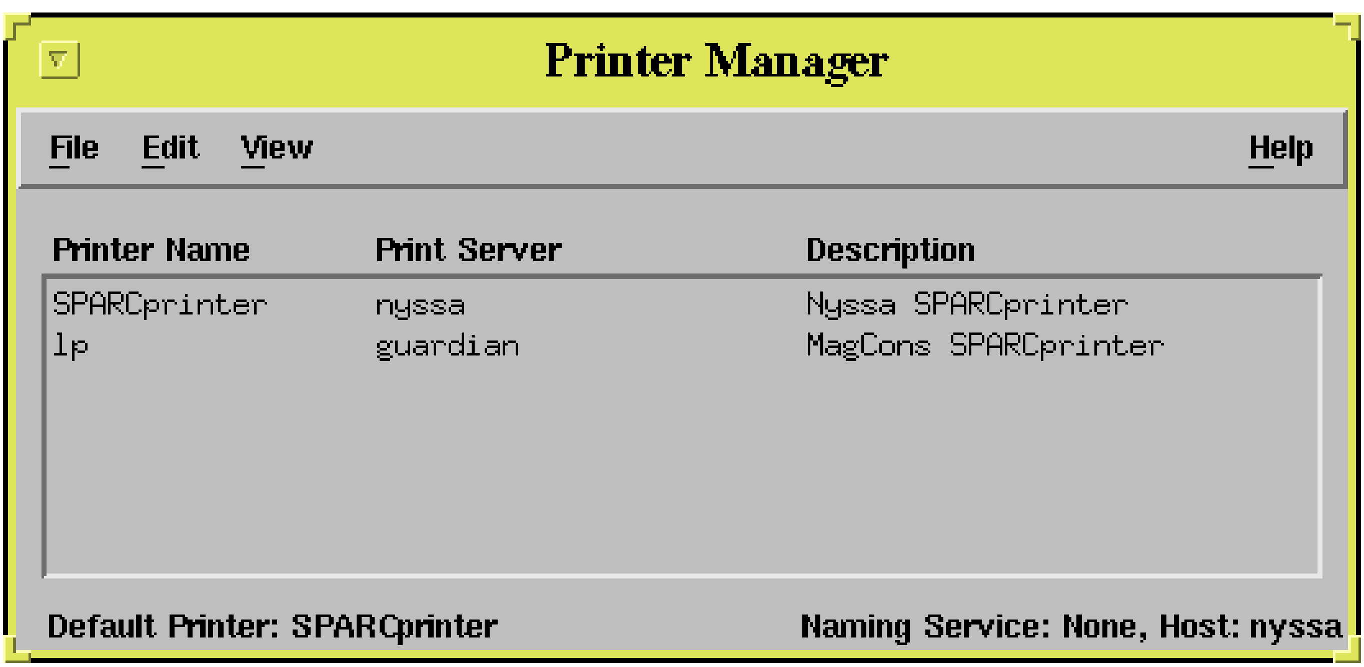
Host
Type
IP Address
Ethernet Address
Timezone
File Server

14.3.3 Print Manager

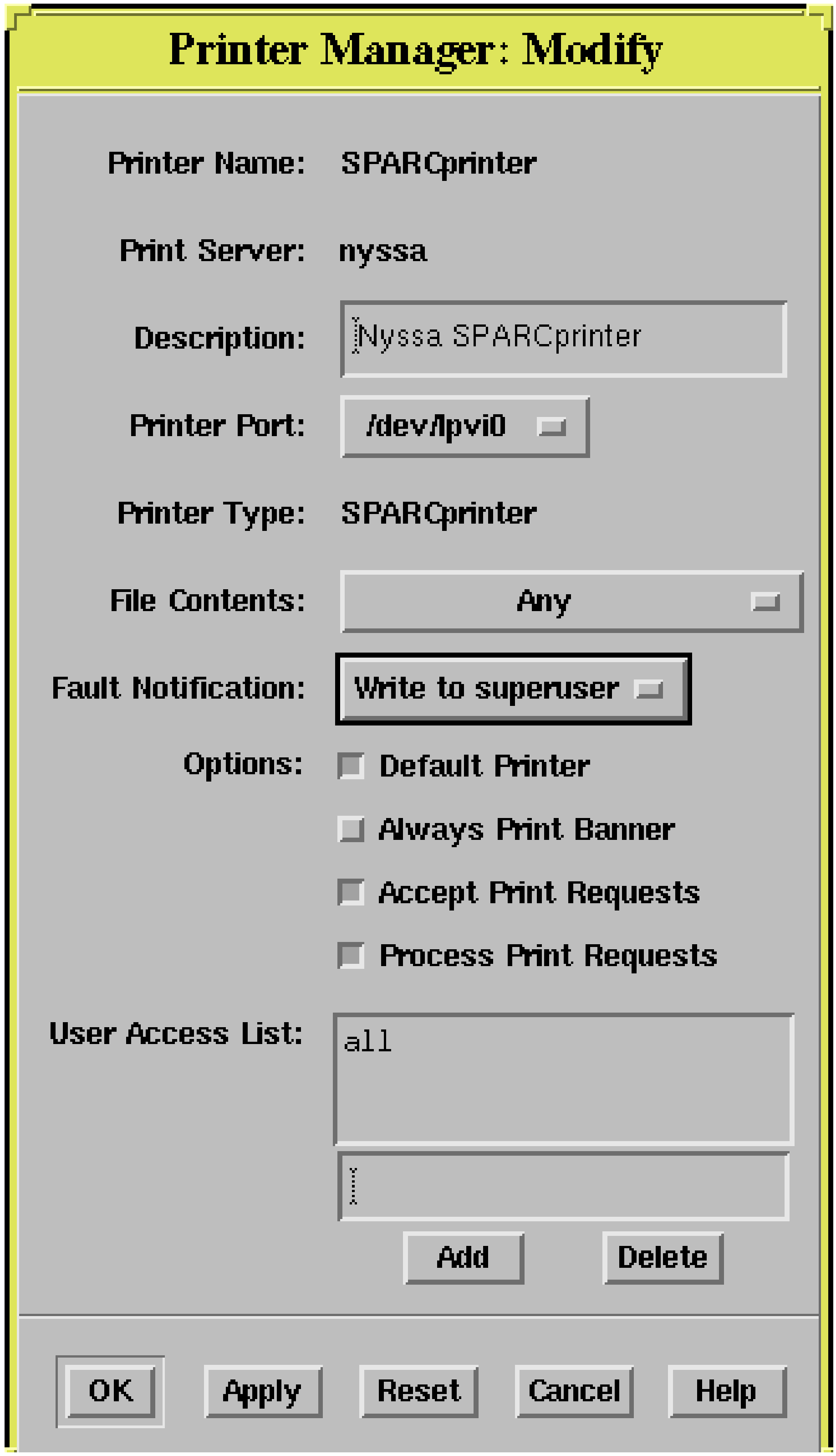
Use **Print Manager** to install and setup printers, using the functions

add access to a printer
install a new printer
modify the configuration for a printer
delete the information for a printer

When selecting **Print Manager** you are presented with the display:



You modify a printer by selecting the **Modify** option under **Edit** and get this display:



14.3.4 User Manager

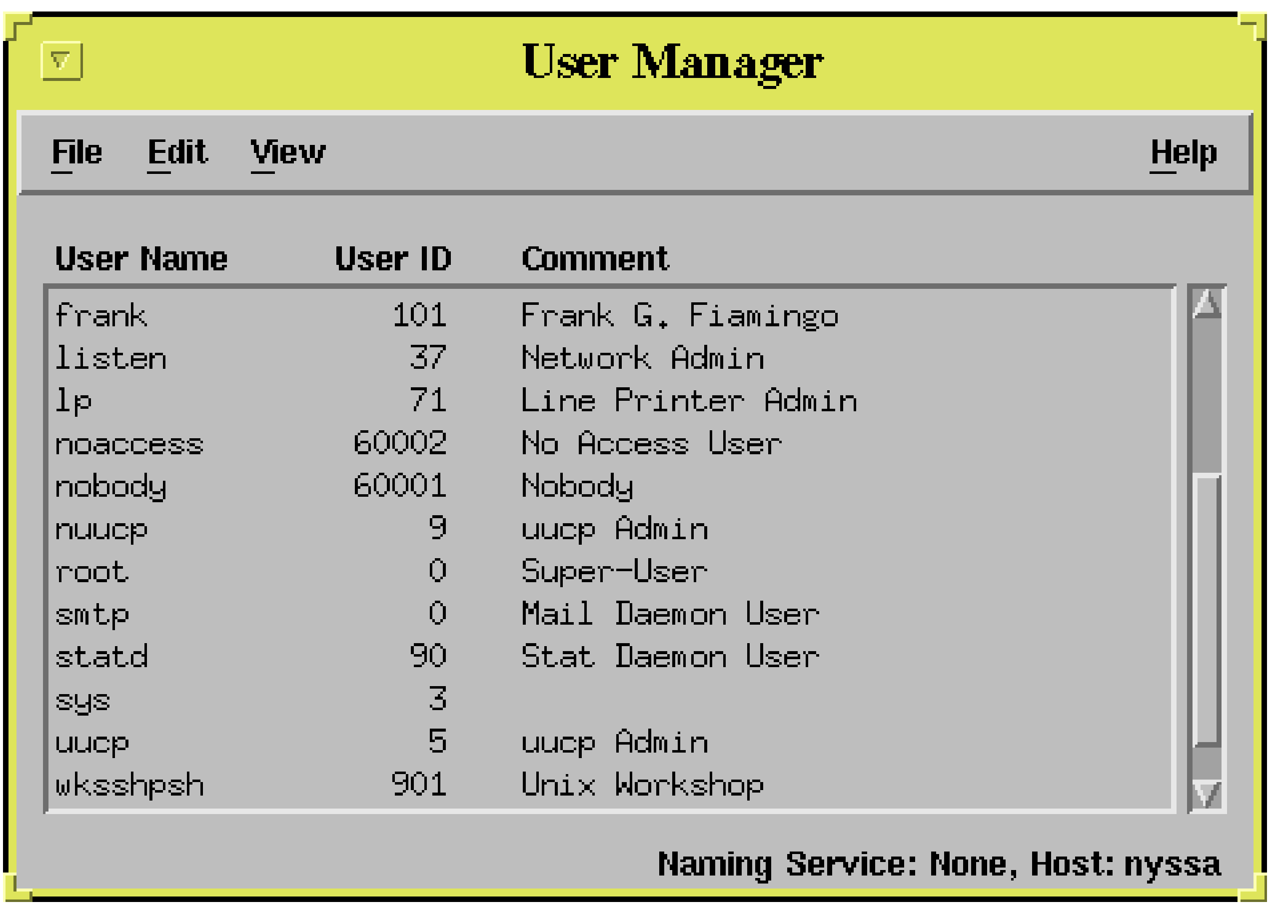
The **User Manager** is used to administer user accounts on a network, which can:

create new accounts
modify accounts
delete accounts

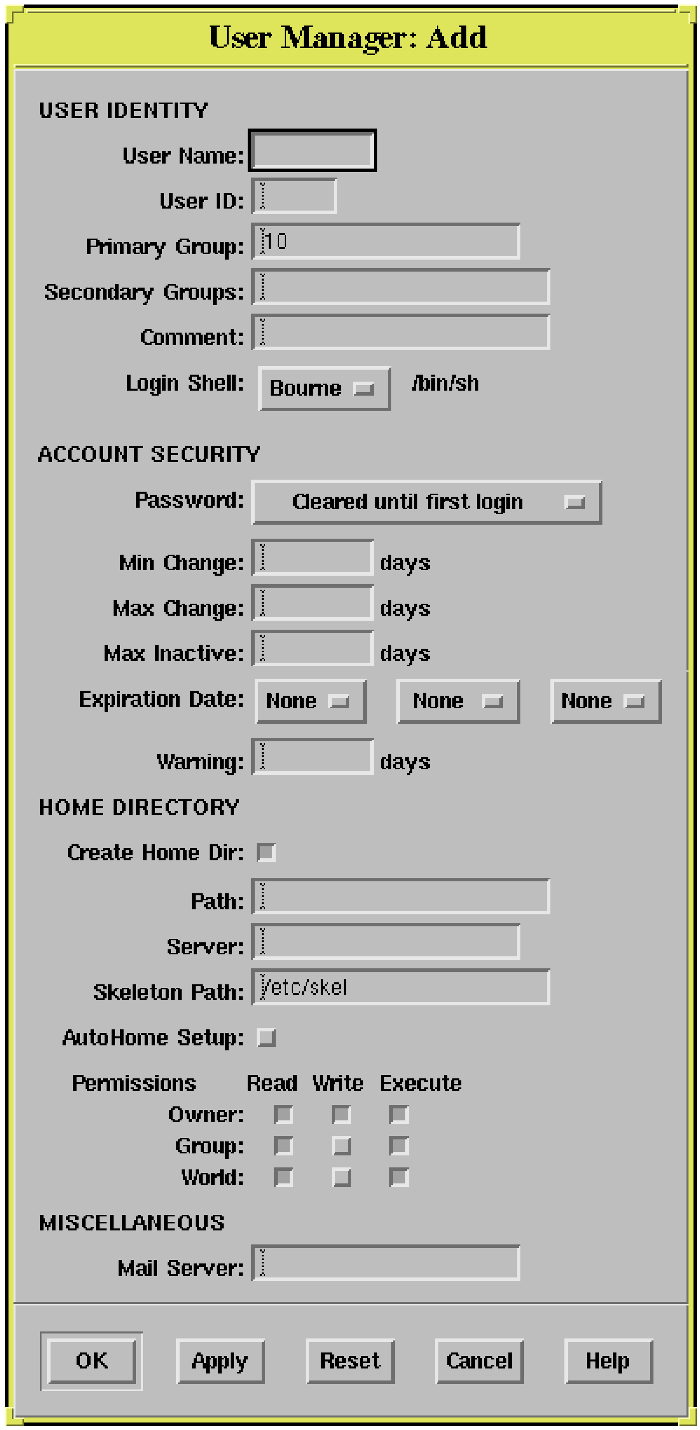
where user account information can be managed by any of the three naming services, **NIS+**, **NIS**, or **None**. The User Manager sets up the home directories with the appropriate file and account information and manages the databases:

aliases
auto_home
cred
group
passwd
shadow

After selecting **User Manager** you asked to choose the naming service, and are then presented with the display:



Choosing **Edit/Add** presents this window:

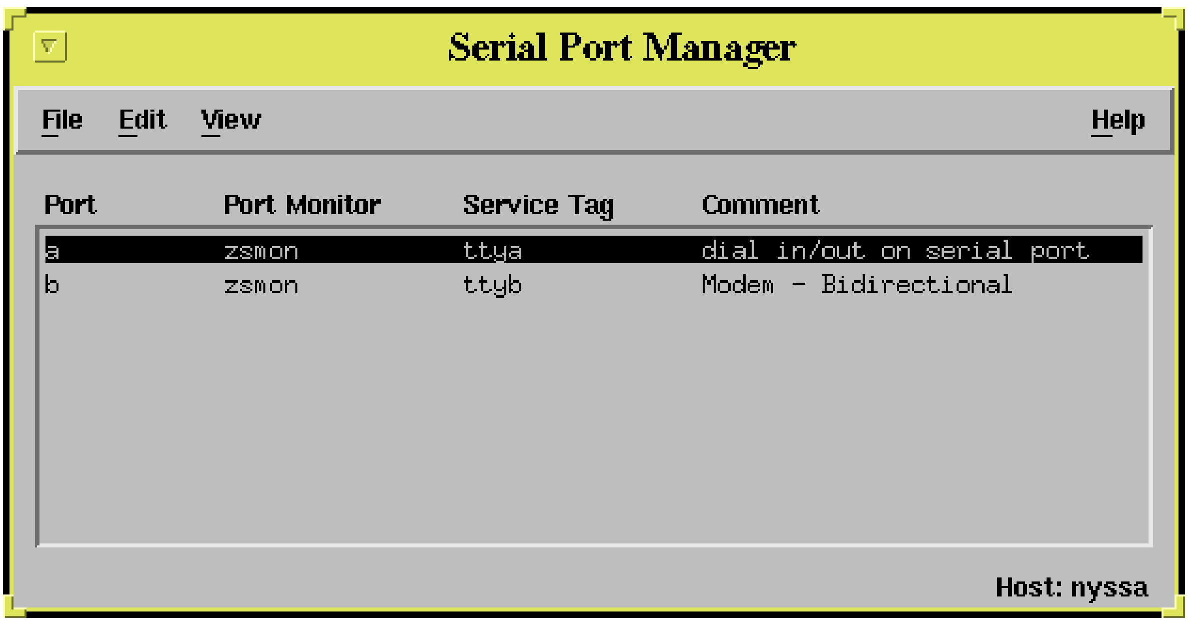


14.3.5 Serial Port Manager

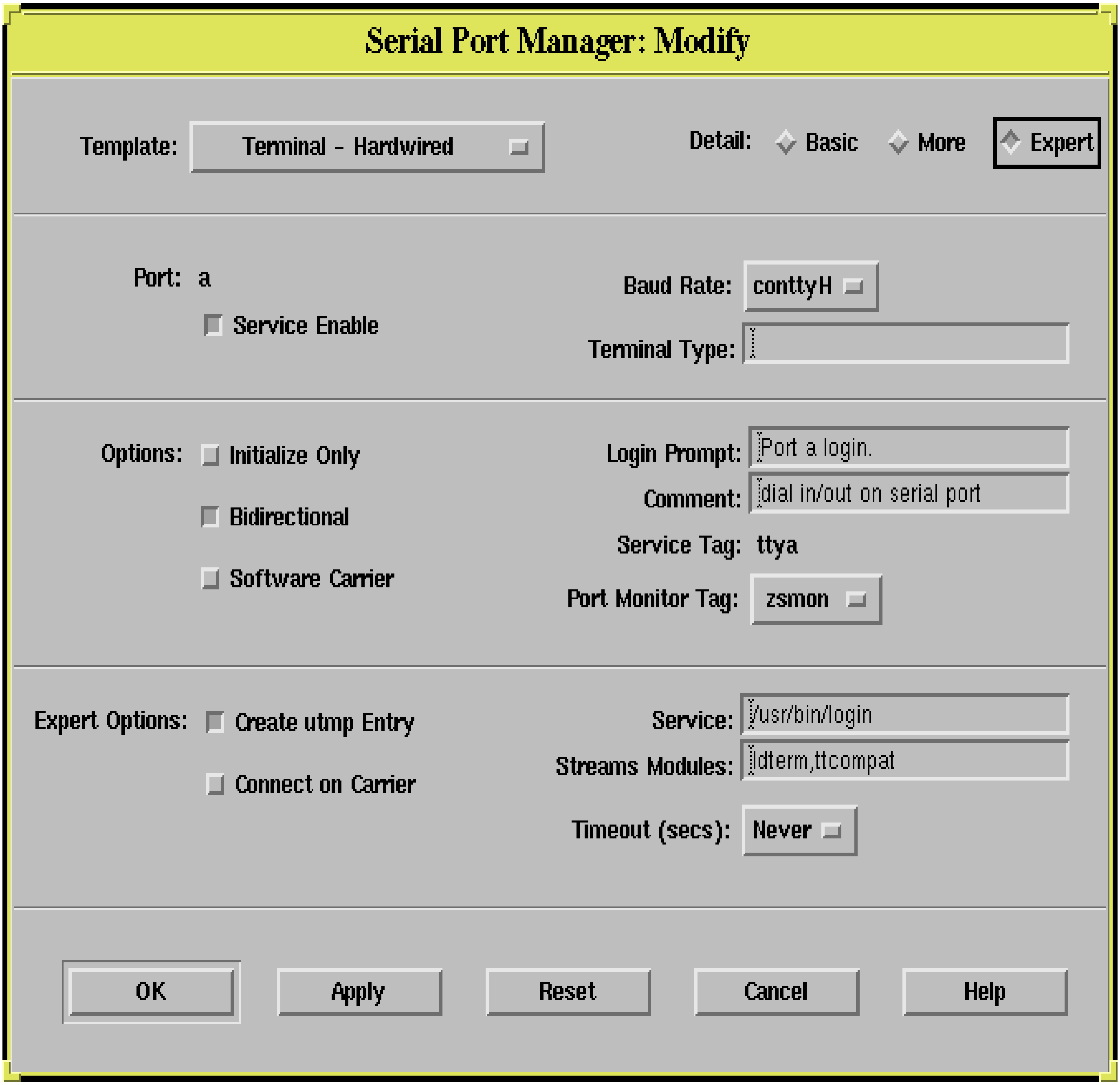
The **Serial Port Manager** lets you configure SAF for terminals and modems. It uses the ***pmadm*** command to configure the serial ports, providing templates for quick installation. You can setup, delete, or check the status of one or many ports. You can configure both local and remote system ports. You can **add**, **modify**, **disable**, or **delete** a service. The templates are provided for:

terminal	hardwired
modem	dial-in only
modem	dial-out only
modem	bidirectional
initialize only	no connection

When selecting **Serial Port Manager** you are presented with the display:



After choosing to **Edit** serial port a, and specifying **Expert** mode, we're given the following display:



14.4 The Distributed System Administration Daemon

The distributed system administration daemon, ***admind*** (SunOS 5.4 and below), or ***sadmind*** (SunOS 5.5 and above), accepts requests for the services preformed by ***admintool*** or ***solstice***, respectively, on the network. The ***admind*** or ***sadmind*** daemon is started automatically by ***inetd*** whenever a request is received, or it can be started on the command line. Before the request is acted upon the daemon must authenticate the client to the server. Once the client identity is verified the daemon uses this identity to allow authorization. The default security level for admind is **SYS**. You can use the more secure **DES** level by specifying the option, ***-S 2***, when invoking the daemon, after first making sure that all servers in the domain are properly set up to use DES security.

User and group identities are used for **authorization** as:

root ID	allows root privileges only on the local system. Root requests from a remote client are changed to user **nobody**. Root on the server is allowed to function as root.
user ID	ordinary users can **retrieve** information, but cannot modify it.
sysadmin group member	***admintool*** or ***solstice*** permission is granted to users who are members of this group on the system where the task is to be performed.

14.5 Program Locations

The programs, ***admintool*** and ***solstice***, are located in **/usr/bin**. The distributed administrative daemons, ***admind*** and ***sadmind***, are in **/usr/sbin**.

The executable programs for ***solstice*** are located in the **/opt/SUNWadm** directory, with some further programs and setup files under the **/usr/snadm** directory. With the ***-l*** and ***-c*** options to ***sadmind*** you specify that a log of requests be kept. By default this log is put in **/var/adm/admin.log**.

CHAPTER 15 Package Administration

15.1 Packages

SVR4 compliant software must be distributed in package format. SunOS 5.X and all unbundled Solaris 2 products are released in this format. Third party software should also be distributed as packages. There are a number of utilities to install, remove, and keep track of the various packages on your system. They keep extensive logs of what's actually installed on your system. There is also an OpenWindows GUI tool, ***swmtool***, that you can use to perform the functions of the following package commands.

Packages can be on tapes or CDROM. On CDROM the package has a directory hierarchy with subdirectories and files, along with possible scripts to control the installation.

15.1.1 pkginfo

The ***pkginfo*** command displays information about the software package specified. It could be a package to be installed, or resident on the system. You can check on individual packages or for the entire distribution media. e.g.

```
# pkginfo -d /cdrom/solaris_2_5_sparc/s0/Solaris_2.5  SUNWaccr
system          SUNWaccr              System Accounting,  (Root)
```

15.1.2 pkgadd

The ***pkgadd*** command is used to install packages on your system. e.g.:

```
# pkgadd -d /cdrom/solaris_2_5_sparc/s0/Solaris_2.5  SUNWaccr
    Processing package instance <SUNWaccr> from </cdrom>
    System Accounting
    ...
    Using </usr> as the package base directory
    ...
    Installation of <SUNWaccr> was successful
```

15.1.3 pkgrm

To remove packages currently installed on the system use the command ***pkgrm***. It's designed to only remove those files belonging exclusively to the package in question. You can execute this command interactively and it will prompt you for the actions to be taken. e.g.:

```
# pkgrm SUNWaccr
    The following package is currently installed:
            SUNWaccr                    System Accounting
    Do you want to remove this package [y,n,?,q] y
    ## Removing installed package instance <SUNWaccr>
    ## Verifying package dependencies
    ...
    ## Updating system information
    Removal of <SUNWaccr> was successful
```

15.1.4 pkgchk

To check on the attributes and integrity of packages use the ***pkgchk*** command. This command verifies the contents of the package against the system log files and reports any discrepancies along with an explanation of the problem. In this example option ***-a*** requests a check on the file attributes and option ***-p*** specifies the path.

```
# pkgchk -a -p /etc/passwd
    ERROR: /etc/passwd
            permissions <0644> expected <0777> actual
```

15.1.5 Package Log Files

The system log file for packages installed is kept in **/var/sadm/install/contents**. This file has a record for every file installed on the system with the ***pkgadd*** command. These records have the form:

```
filename filetype permissions owner group size(in bytes) checksum(of contents) time(of last modification) PackageList
```

where PackageList is the list of the packages associated with the file, e.g.:

```
/etc d none 0775 root sys SUNWcsr SUNWesu SUNWadmr SUNWbnur SUNWlpr SUNWnisr SUNWscpr
/etc/.login f none 0644 root sys 445 32698 720806491 SUNWcsr
/etc/TIMEZONE v none 0444 root sys 131 9791 720806487 SUNWcsr
/etc/aliases=./mail/aliases s none SUNWcsr
/etc/auto_home v none 0555 root bin 50 4502 720800466 SUNWcsr
/etc/auto_master v none 0555 root bin 83 7133 720800463 SUNWcsr
/etc/autopush=../sbin/autopush s none SUNWcsr
/etc/chroot=../usr/sbin/chroot s none SUNWscpr
/etc/clri=../usr/sbin/clri s none SUNWcsr
/etc/crash=../usr/kvm/crash s none SUNWcsr
/etc/cron=../usr/sbin/cron s none SUNWcsr
/etc/cron.d d none 0755 root sys SUNWcsr
/etc/cron.d/.proto f none 0744 root sys 82 5173 28800 SUNWcsr
/etc/cron.d/at.deny v none 0644 root sys 45 4171 28800 SUNWcsr
```



```
/etc/cron.d/cron.deny v none 0644 root sys 45 4171 28800 SUNWcsr
/etc/cron.d/logchecker f none 0555 bin bin 1178 27089 720797586 SUNWcsr
/etc/cron.d/queuedefs f none 0644 root sys 17 1164 28800 SUNWcsr
/etc/dcopy=../usr/sbin/dcopy s none SUNWcsr
/etc/default d none 0775 root sys SUNWcsr
/etc/default/cron f none 0555 bin bin 12 844 720797589 SUNWcsr
/etc/default/fs f none 0444 bin bin 10 768 720801815 SUNWcsr
/etc/default/login f none 0444 root sys 146 9532 720803469 SUNWcsr
/etc/default/passwd f none 0444 root sys 74 4934 720807242 SUNWcsr
/etc/default/su f none 0444 root sys 97 6692 720809999 SUNWcsr
```

The **contents** file is taken from the ***pkgmap*** file supplied with each package and logged in **/var/sadm/pkg** in a subdirectory known by the package name, in a file named ***pkgmap***. Also in this subdirectory there is a ***pkginfo*** file describing the package. The system does not supply any tools to list the files contained in a package, but you can examine the ***pkgmap*** files or use grep to search the **contents** file for this information.

15.2 Packages Distributed with Solaris 2.5

The 262 packages distributed with Solaris 2.5 server edition, as obtained from the ***pkginfo*** command, are listed in the table below for the 7 CDROMs in the set.

TABLE 15.1 **Solaris 2.5 Software**

CDROM	Type	Name	Description
adminsuite_2_1	application	SUNWsadmd	Solstice AdminSuite AnswerBook
adminsuite_2_1	application	SUNWlicja	FlexLM License System Japanese Localization
adminsuite_2_1	application	SUNWlicsw	FlexLM License System
adminsuite_2_1	application	SUNWlit	STE License Installation Tool
adminsuite_2_1	application	SUNWlitja	STE License Installation Tool Japanese Localization
adminsuite_2_1	system	SUNWpcr	SunSoft Print - Client, (root), Early Access
adminsuite_2_1	system	SUNWpcu	SunSoft Print - Client, (usr), Early Access
adminsuite_2_1	system	SUNWpsf	PostScript filters - Early Access, (Usr)
adminsuite_2_1	system	SUNWpsr	SunSoft Print - LP Server, (root), Early Access
adminsuite_2_1	system	SUNWpsu	SunSoft Print - LP Server, (usr), Early Access
adminsuite_2_1	system	SUNWsadma	Solstice AdminSuite system & network administration applications.
adminsuite_2_1	system	SUNWsadmc	Solstice AdminSuite system & network administration methods.
adminsuite_2_1	system	SUNWsadmo	Solstice AdminSuite object libraries.
adminsuite_2_1	system	SUNWscplp	SunSoft Print - Source Compatibility, (Usr)
adminsuite_2_1	system	SUNWspapp	Solstice AdminSuite print application
adminsuite_2_1	system	SUNWspman	On-Line Manual Pages
disksuite_4_0	application	SUNWabmd	DiskSuite 4.0 AnswerBook
disksuite_4_0	system	SUNWmd	Solstice DiskSuite
disksuite_4_0	system	SUNWmdg	Solstice DiskSuite Tool
networker_4_1_2	application	SUNWsbuc	Solstice Backup (Backup/Recover) Client Package

TABLE 15.1 **Solaris 2.5 Software**

CDROM	Type	Name	Description
networker_4_1_2	application	SUNWsbum	Solstice Backup (Backup/Recover) Man Pages
networker_4_1_2	application	SUNWsbus1	Solstice Backup (Backup/Recover) Server Package
networker_4_1_2	system	SUNWsbus2	Solstice Backup (Backup/Recover) Device Drivers
solaris_2_5_desktop_1_0	system	SUNWdtab	CDE DTBUILDER
solaris_2_5_desktop_1_0	system	SUNWdtdem	CDE DEMOS
solaris_2_5_desktop_1_0	system	SUNWdthed	CDE HELP DEVELOPER ENVIRONMENT
solaris_2_5_desktop_1_0	system	SUNWdtinc	CDE Includes
solaris_2_5_desktop_1_0	system	SUNWdtma	CDE man pages
solaris_2_5_desktop_1_0	system	SUNWdtmad	CDE developer man pages
solaris_2_5_desktop_1_0	system	SUNWmfdev	Motif Development Kit
solaris_2_5_desktop_1_0	system	SUNWmfdm	Motif Demos
solaris_2_5_desktop_1_0	system	SUNWmfman	CDE Motif Development Kit Manuals
solaris_2_5_desktop_1_0	system	SUNWtltkd	ToolTalk CDE developer support
solaris_2_5_desktop_1_0	system	SUNWtltkm	ToolTalk CDE manual pages
solaris_2_5_desktop_1_0	system	SUNWdtdst	CDE DESKTOP APPS
solaris_2_5_desktop_1_0	system	SUNWdthe	CDE HELP RUNTIME
solaris_2_5_desktop_1_0	system	SUNWdthev	CDE HELP VOLUMES
solaris_2_5_desktop_1_0	system	SUNWdtim	CDE DESKTOP APPS
solaris_2_5_desktop_1_0	system	SUNWdtrme	CDE README FILES
solaris_2_5_desktop_1_0	system	SUNWdtwm	CDE DESKTOP WINDOW MANAGER
solaris_2_5_desktop_1_0	application	SUNWdta	Solaris Common Desktop Env. AnswerBook 1.0.1
solaris_2_5_desktop_1_0	system	SUNWdtbas	CDE base
solaris_2_5_desktop_1_0	system	SUNWdtcor	CORE (CDE)
solaris_2_5_desktop_1_0	system	SUNWdtdmn	CDE daemons
solaris_2_5_desktop_1_0	system	SUNWdtdte	CDE DESKTOP LOGIN ENVIRONMENT
solaris_2_5_desktop_1_0	system	SUNWdtft	CDE fonts
solaris_2_5_desktop_1_0	system	SUNWdticn	CDE icons
solaris_2_5_desktop_1_0	system	SUNWmfrun	Motif RunTime Kit
solaris_2_5_desktop_1_0	system	SUNWtltk	ToolTalk CDE runtime
solaris_2_5_desktop_1_0	application	ISLIodbc	ODBC (Open DataBase Connectivity) Driver Manager
solaris_2_5_desktop_1_0	application	ISLIodbcD	Demo ODBC (Open DataBase Connectivity) Mutli-Dialect dBASE Driver
solaris_2_5_desktop_1_0	application	SUNWaws	Wabi 2.1 AnswerBook
solaris_2_5_desktop_1_0	application	SUNWwabi	Wabi Application
solaris_2_5_server_1_0	application	SUNWnskta	NSKit 1.2 AnswerBook
solaris_2_5_server_1_0	system	SUNWnsktr	NIS Server for Solaris (root)
solaris_2_5_server_1_0	system	SUNWnsktu	NIS Server for Solaris (usr)
solaris_2_5_server_1_0	application	SUNWaadm	Solaris 2.5 System Administrator AnswerBook
solaris_2_5_server_1_0	application	SUNWaman	Solaris 2.5 Reference Manual AnswerBook
solaris_2_5_server_1_0	system	SUNWipx	PC Protocol Services 1.1
solaris_2_5_sparc	system	AXILvplr.c	Axil platform links
solaris_2_5_sparc	system	AXILvplr.m	Axil platform links

TABLE 15.1 **Solaris 2.5 Software**

CDROM	Type	Name	Description
solaris_2_5_sparc	system	AXILvplu.c	Axil usr/platform links
solaris_2_5_sparc	system	AXILvplu.m	Axil usr/platform links
solaris_2_5_sparc	system	PFUcar.m	PFU/Fujitsu kernel/unix for Power Control Software
solaris_2_5_sparc	system	PFUdfb.m	S-4/Leia LCD Dumb Frame Buffer Driver
solaris_2_5_sparc	system	PFUvplr.m	PFU/Fujitsu platform links
solaris_2_5_sparc	system	PFUvplu.m	PFU/Fujitsu usr/platform links
solaris_2_5_sparc	application	SUNWabe	Solaris 2.5 User AnswerBook
solaris_2_5_sparc	system	SUNWaccr	System Accounting, (Root)
solaris_2_5_sparc	system	SUNWaccu	System Accounting, (Usr)
solaris_2_5_sparc	system	SUNWadmap	System administration applications
solaris_2_5_sparc	system	SUNWadmc	System administration core libraries
solaris_2_5_sparc	system	SUNWadmfw	System & Network Administration Framework
solaris_2_5_sparc	system	SUNWadmr	System & Network Administration Root
solaris_2_5_sparc	system	SUNWapppr	PPP/IP Asynchronous PPP daemon config files
solaris_2_5_sparc	system	SUNWapppu	PPP/IP Asynchronous PPP daemon and PPP login service
solaris_2_5_sparc	system	SUNWarc	Archive Libraries
solaris_2_5_sparc	system	SUNWast	Automated Security Enhancement Tools
solaris_2_5_sparc	system	SUNWaudio	Audio applications
solaris_2_5_sparc	system	SUNWaudmo	Audio demo programs
solaris_2_5_sparc	system	SUNWbcp	SunOS 4.x Binary Compatibility
solaris_2_5_sparc	system	SUNWbnur	Networking UUCP Utilities, (Root)
solaris_2_5_sparc	system	SUNWbnuu	Networking UUCP Utilities, (Usr)
solaris_2_5_sparc	system	SUNWbtool	CCS tools bundled with SunOS
solaris_2_5_sparc	system	SUNWcar.c	Core Architecture, (Root)
solaris_2_5_sparc	system	SUNWcar.d	Core Architecture, (Root)
solaris_2_5_sparc	system	SUNWcar.m	Core Architecture, (Root)
solaris_2_5_sparc	system	SUNWcar.ma	Core Architecture, (Root)
solaris_2_5_sparc	system	SUNWcar.u	Core Architecture, (Root)
solaris_2_5_sparc	system	SUNWcg6.c	GX (cg6) Device Driver
solaris_2_5_sparc	system	SUNWcg6.d	GX (cg6) Device Driver
solaris_2_5_sparc	system	SUNWcg6.m	GX (cg6) Device Driver
solaris_2_5_sparc	system	SUNWcg6.ma	GX (cg6) Device Driver
solaris_2_5_sparc	system	SUNWcg6.u	GX (cg6) Device Driver
solaris_2_5_sparc	system	SUNWcg6h	GX (cg6) Header Files
solaris_2_5_sparc	system	SUNWcsd	Core Solaris Devices
solaris_2_5_sparc	system	SUNWcsr	Core Solaris, (Root)
solaris_2_5_sparc	system	SUNWcsu	Core Solaris, (Usr)
solaris_2_5_sparc	system	SUNWdfb.c	Dumb Frame Buffer Device Drivers
solaris_2_5_sparc	system	SUNWdfb.d	Dumb Frame Buffer Device Drivers
solaris_2_5_sparc	system	SUNWdfb.m	Dumb Frame Buffer Device Drivers
solaris_2_5_sparc	system	SUNWdfb.ma	Dumb Frame Buffer Device Drivers

TABLE 15.1 **Solaris 2.5 Software**

CDROM	Type	Name	Description
solaris_2_5_sparc	system	SUNWdfb.u	Dumb Frame Buffer Device Drivers
solaris_2_5_sparc	system	SUNWdfbh	Dumb Frame Buffer Header Files
solaris_2_5_sparc	system	SUNWdial	Buttons/Dials (bd) Streams Module
solaris_2_5_sparc	application	SUNWdialh	Buttons/Dials (bd) Header Files
solaris_2_5_sparc	system	SUNWdoc	Documentation Tools
solaris_2_5_sparc	system	SUNWdtcor	CORE (CDE)
solaris_2_5_sparc	system	SUNWdxlib	Direct Xlib
solaris_2_5_sparc	system	SUNWesu	Extended System Utilities
solaris_2_5_sparc	system	SUNWfac	Framed Access Command Environment
solaris_2_5_sparc	system	SUNWffb.u	FFB System Software (Device Driver)
solaris_2_5_sparc	application	SUNWffbcf	FFB Configuration Software
solaris_2_5_sparc	application	SUNWffbmn	On-Line FFB Manual Pages
solaris_2_5_sparc	application	SUNWffbw	FFB Window System Support
solaris_2_5_sparc	application	SUNWffbxg	FFB XGL support
solaris_2_5_sparc	system	SUNWfns	Federated Naming System
solaris_2_5_sparc	system	SUNWfnspr	FNS Support For Printer Context
solaris_2_5_sparc	system	SUNWfnsx5	FNS Support For X.500 Directory Context
solaris_2_5_sparc	system	SUNWhea	SunOS Header Files
solaris_2_5_sparc	system	SUNWhinst	4.1* Heterogeneous Install Software
solaris_2_5_sparc	system	SUNWhmd	SunSwift SBus Adapter Drivers
solaris_2_5_sparc	system	SUNWhmdu	SunSwift SBus Adapter Headers
solaris_2_5_sparc	system	SUNWinst	Install Software
solaris_2_5_sparc	system	SUNWipc	Interprocess Communications
solaris_2_5_sparc	system	SUNWisolc	XSH4 conversion for ISO Latin character sets
solaris_2_5_sparc	application	SUNWkcspf	KCMS Optional Profiles
solaris_2_5_sparc	application	SUNWkcspg	KCMS Programmers Environment
solaris_2_5_sparc	application	SUNWkcsrt	KCMS Runtime Environment
solaris_2_5_sparc	system	SUNWkey	Keyboard configuration tables
solaris_2_5_sparc	system	SUNWkvm.c	Core Architecture, (Kvm)
solaris_2_5_sparc	system	SUNWkvm.d	Core Architecture, (Kvm)
solaris_2_5_sparc	system	SUNWkvm.m	Core Architecture, (Kvm)
solaris_2_5_sparc	system	SUNWkvm.ma	Core Architecture, (Kvm)
solaris_2_5_sparc	system	SUNWkvm.u	Core Architecture, (Kvm)
solaris_2_5_sparc	system	SUNWleo.d	ZX System Software (Device Driver)
solaris_2_5_sparc	system	SUNWleo.m	ZX System Software (Device Driver)
solaris_2_5_sparc	application	SUNWleoo	ZX XGL support
solaris_2_5_sparc	system	SUNWleor	ZX System Software (Root)
solaris_2_5_sparc	application	SUNWleow	ZX Window System Support
solaris_2_5_sparc	system	SUNWlibC	SPARCompilers Bundled libC
solaris_2_5_sparc	system	SUNWlibCf	SPARCompilers Bundled libC (cfront version)
solaris_2_5_sparc	system	SUNWlibm	SPARCompilers Bundled libm

TABLE 15.1 **Solaris 2.5 Software**

CDROM	Type	Name	Description
solaris_2_5_sparc	system	SUNWlibms	SPARCompilers Bundled shared libm
solaris_2_5_sparc	system	SUNWloc	System Localization
solaris_2_5_sparc	system	SUNWlpmsg	LP Alerts
solaris_2_5_sparc	system	SUNWlpr	LP Print Service, (Root)
solaris_2_5_sparc	system	SUNWlps	LP Print Service - Server, (Usr)
solaris_2_5_sparc	system	SUNWlpu	LP Print Service - Client, (Usr)
solaris_2_5_sparc	system	SUNWman	On-Line Manual Pages
solaris_2_5_sparc	system	SUNWmfrun	Motif RunTime Kit
solaris_2_5_sparc	system	SUNWnisr	Network Information System, (Root)
solaris_2_5_sparc	system	SUNWnisu	Network Information System, (Usr)
solaris_2_5_sparc	system	SUNWoladd	OPEN LOOK Alternate Desktop Demos
solaris_2_5_sparc	system	SUNWolaud	OPEN LOOK Audio applications
solaris_2_5_sparc	system	SUNWolbk	OpenWindows online handbooks
solaris_2_5_sparc	system	SUNWoldcv	OPEN LOOK document and help viewer applications
solaris_2_5_sparc	system	SUNWoldem	OPEN LOOK demo programs
solaris_2_5_sparc	system	SUNWoldim	OPEN LOOK demo images
solaris_2_5_sparc	system	SUNWoldst	OPEN LOOK deskset tools
solaris_2_5_sparc	system	SUNWoldte	OPEN LOOK Desktop Environment
solaris_2_5_sparc	system	SUNWolimt	OPEN LOOK imagetool
solaris_2_5_sparc	system	SUNWolinc	OPEN LOOK include files
solaris_2_5_sparc	system	SUNWolman	OPEN LOOK toolkit/desktop users man pages
solaris_2_5_sparc	system	SUNWolrte	OPEN LOOK toolkits runtime environment
solaris_2_5_sparc	system	SUNWolslb	OPEN LOOK toolkit/desktop static/lint libraries
solaris_2_5_sparc	system	SUNWolsrc	OPEN LOOK sample source
solaris_2_5_sparc	system	SUNWowbcp	OpenWindows binary compatibility
solaris_2_5_sparc	system	SUNWowrqd	OpenWindows required core package
solaris_2_5_sparc	system	SUNWpcmci	PCMCIA Card Services, (Root)
solaris_2_5_sparc	system	SUNWpcmcu	PCMCIA Card Services, (Usr)
solaris_2_5_sparc	system	SUNWpcmem	PCMCIA memory card driver
solaris_2_5_sparc	system	SUNWpcser	PCMCIA serial card driver
solaris_2_5_sparc	application	SUNWpexcl	PEX Runtime Client Library
solaris_2_5_sparc	application	SUNWpexh	PEX Client Developer Files
solaris_2_5_sparc	application	SUNWpexsv	PEX Runtime Server Extension
solaris_2_5_sparc	system	SUNWploc	Partial Locales
solaris_2_5_sparc	system	SUNWploc1	Supplementary Partial Locales
solaris_2_5_sparc	system	SUNWplow	OpenWindows enabling for Partial Locales
solaris_2_5_sparc	system	SUNWplow1	OpenWindows enabling for Suppl. Partial Locales
solaris_2_5_sparc	system	SUNWpppk	PPP/IP and IPdialup Device Drivers
solaris_2_5_sparc	system	SUNWrdm	On-Line Open Issues ReadMe
solaris_2_5_sparc	system	SUNWrtvc	SunVideo Device Driver
solaris_2_5_sparc	application	SUNWrtvcu	SunVideo Runtime Support Software

TABLE 15.1 **Solaris 2.5 Software**

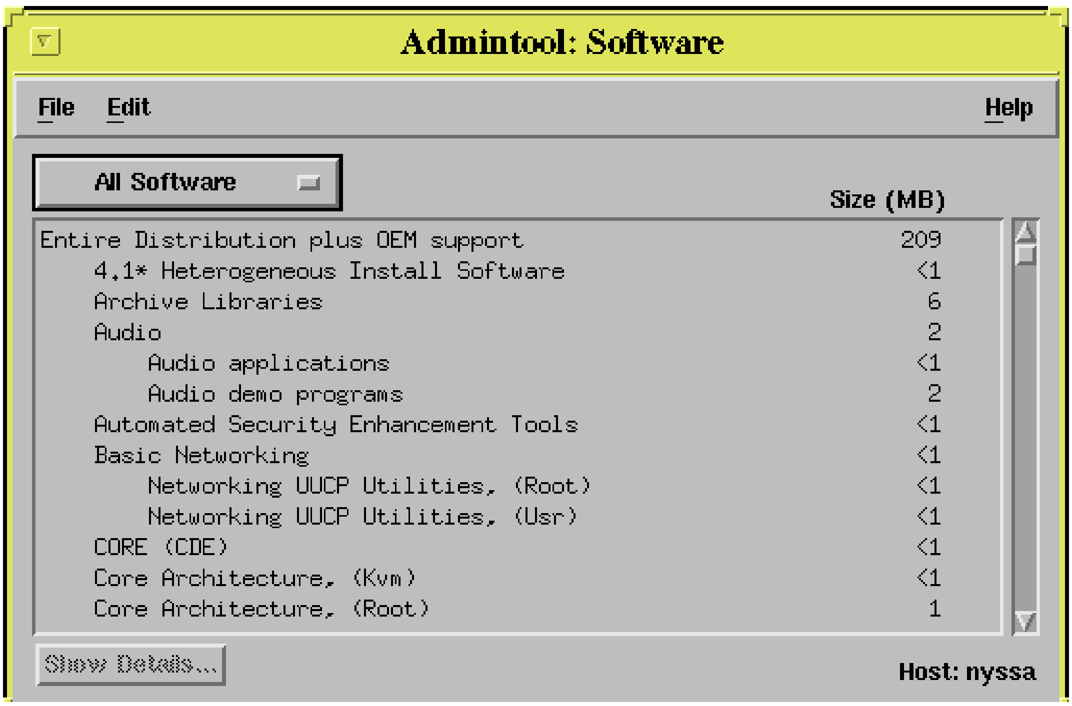
CDROM	Type	Name	Description
solaris_2_5_sparc	system	SUNWsadml	Solstice Admintool Launcher.
solaris_2_5_sparc	system	SUNWscbcp	SPARCompilers Binary Compatibility Libraries
solaris_2_5_sparc	system	SUNWscpr	Source Compatibility, (Root)
solaris_2_5_sparc	system	SUNWscpu	Source Compatibility, (Usr)
solaris_2_5_sparc	system	SUNWsprot	SPARCompilers Bundled tools
solaris_2_5_sparc	system	SUNWsra	Source Compatibility Archive Libraries
solaris_2_5_sparc	system	SUNWsrh	Source Compatibility Header Files
solaris_2_5_sparc	system	SUNWssadv	SPARCstorage Array Drivers
solaris_2_5_sparc	system	SUNWssaop	SPARCstorage Array Utility
solaris_2_5_sparc	system	SUNWsutl	Static Utilities
solaris_2_5_sparc	application	SUNWsx	SX Shareable Library
solaris_2_5_sparc	application	SUNWsxow	SX Window System Support
solaris_2_5_sparc	system	SUNWsxr.m	SX Video Subsystem Drivers
solaris_2_5_sparc	application	SUNWsxxgl	SX XGL Support
solaris_2_5_sparc	application	SUNWsxxil	SX XIL Support
solaris_2_5_sparc	system	SUNWtcx.m	TCX System Software (Device Driver)
solaris_2_5_sparc	application	SUNWtcxow	TCX Window System Support
solaris_2_5_sparc	application	SUNWtcxu	TCX XGL Support
solaris_2_5_sparc	system	SUNWter	Terminal Information
solaris_2_5_sparc	system	SUNWtltk	ToolTalk runtime
solaris_2_5_sparc	system	SUNWtltkd	ToolTalk developer support
solaris_2_5_sparc	system	SUNWtltkm	ToolTalk manual pages
solaris_2_5_sparc	system	SUNWtnfc	TNF Core Components
solaris_2_5_sparc	system	SUNWtnfd	TNF Developer Components
solaris_2_5_sparc	system	SUNWtoo	Programming Tools
solaris_2_5_sparc	application	SUNWvlxil	VIS/XIL Support
solaris_2_5_sparc	system	SUNWvolg	Volume Management Graphical User Interface
solaris_2_5_sparc	system	SUNWvolr	Volume Management, (Root)
solaris_2_5_sparc	system	SUNWvolu	Volume Management, (Usr)
solaris_2_5_sparc	system	SUNWxcu4	XCU4 Utilities
solaris_2_5_sparc	system	SUNWxcu4t	XCU4 make and sccs utilities
solaris_2_5_sparc	application	SUNWxgldg	XGL Generic Loadable Libraries
solaris_2_5_sparc	application	SUNWxgler	XGL English Localization
solaris_2_5_sparc	application	SUNWxglft	XGL Stroke Fonts
solaris_2_5_sparc	application	SUNWxglh	XGL Include Files
solaris_2_5_sparc	application	SUNWxglrt	XGL Runtime Environment
solaris_2_5_sparc	system	SUNWxi18n	X Windows I18N Common Package
solaris_2_5_sparc	application	SUNWxildg	XIL Loadable Pipeline Libraries
solaris_2_5_sparc	application	SUNWxiler	XIL English Localization
solaris_2_5_sparc	application	SUNWxilh	XIL Header Files
solaris_2_5_sparc	application	SUNWxilow	XIL Deskset Loadable Pipeline Libraries

TABLE 15.1 **Solaris 2.5 Software**

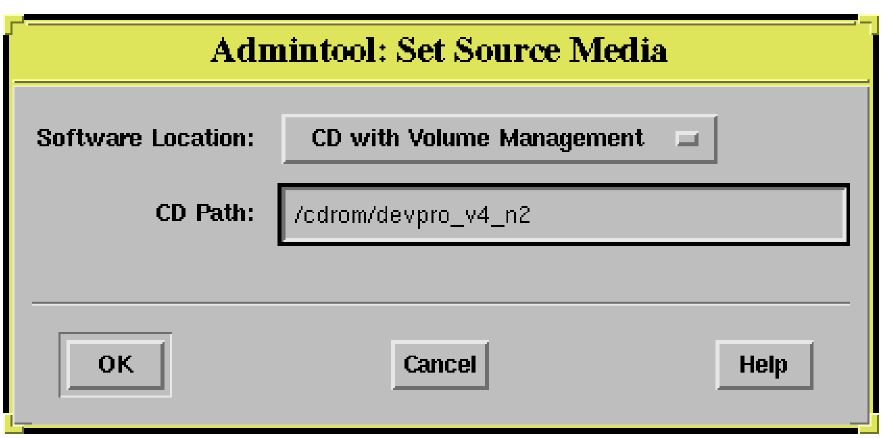
CDROM	Type	Name	Description
solaris_2_5_sparc	application	SUNWxilrt	XIL Runtime Environment
solaris_2_5_sparc	system	SUNWxwacx	AccessX client program
solaris_2_5_sparc	system	SUNWxwcft	X Windows common (not required) fonts
solaris_2_5_sparc	system	SUNWxwdem	X Windows demo programs
solaris_2_5_sparc	system	SUNWxwdim	X Windows demo images
solaris_2_5_sparc	system	SUNWxwdv	XWindows Window Drivers
solaris_2_5_sparc	system	SUNWxwdxm	DPS motif library
solaris_2_5_sparc	system	SUNWxwfnt	X Windows platform required fonts
solaris_2_5_sparc	system	SUNWxwfs	Font server
solaris_2_5_sparc	system	SUNWxwinc	X Windows include files
solaris_2_5_sparc	system	SUNWxwman	X Windows online user man pages
solaris_2_5_sparc	system	SUNWxwmod	OpenWindows kernel modules
solaris_2_5_sparc	system	SUNWxwoft	X Windows optional fonts
solaris_2_5_sparc	system	SUNWxwopt	nonessential MIT core clients and server extensions
solaris_2_5_sparc	system	SUNWxwplt	X Windows platform software
solaris_2_5_sparc	system	SUNWxwpmn	X Windows online programmers man pages
solaris_2_5_sparc	system	SUNWxwpsr	Sun4u-platform specific X server aux filter modules
solaris_2_5_sparc	system	SUNWxwslb	X Windows static/lint libraries
solaris_2_5_sparc	system	SUNWxwsrc	X Windows sample source
solaris_2_5_sparc	system	TSBWvplr.m	Toshiba platform links
solaris_2_5_sparc	system	TSBWvplu.m	Toshiba usr/platform links
upd_sol_2_5_smcc	application	SUNWabhdw	Solaris 2.5 on Sun Hardware AnswerBook
upd_sol_2_5_smcc	system	SUNWbtryh	Battery Module Header File
upd_sol_2_5_smcc	system	SUNWbttry.m	Battery Streams Module
upd_sol_2_5_smcc	system	SUNWcpr.m	Suspend, Resume package
upd_sol_2_5_smcc	system	SUNWcpr.u	Suspend, Resume package
upd_sol_2_5_smcc	system	SUNWird.m	Infra Red Device Driver based on MIC
upd_sol_2_5_smcc	system	SUNWirdh	Infra Red Device Driver Header File
upd_sol_2_5_smcc	system	SUNWpmc.m	Power Management Chip Driver
upd_sol_2_5_smcc	system	SUNWpmch	Power Management Chip Driver Header Files
upd_sol_2_5_smcc	system	SUNWpmman	Power Management Man Pages
upd_sol_2_5_smcc	system	SUNWpmow	Power Management OW Utilities
upd_sol_2_5_smcc	system	SUNWpmr	Power Management config file and rc script
upd_sol_2_5_smcc	system	SUNWpmu	Power Management binaries
upd_sol_2_5_smcc	system	SUNWvts	Online Validation Test Suite
upd_sol_2_5_smcc	system	SUNWvtsmn	Online Validation Test Suite Man Pages
upd_sol_2_5_smcc	system	SUNWvygmn	SPARCstation Voyager Man Pages

15.3 Swmtool

swmtool is an X-windows GUI to the package commands. In later versions it's part of Admintool. With it you can install, upgrade, or remove the software packages on a local or remote system. Starting with Solaris 2.5 it's now part of the Admintool set of programs. It checks **/var/sadm/install/contents** for the packages installed. To use it bring up the tool, and select "**Add**" under the "**Edit**" menu to install new software.

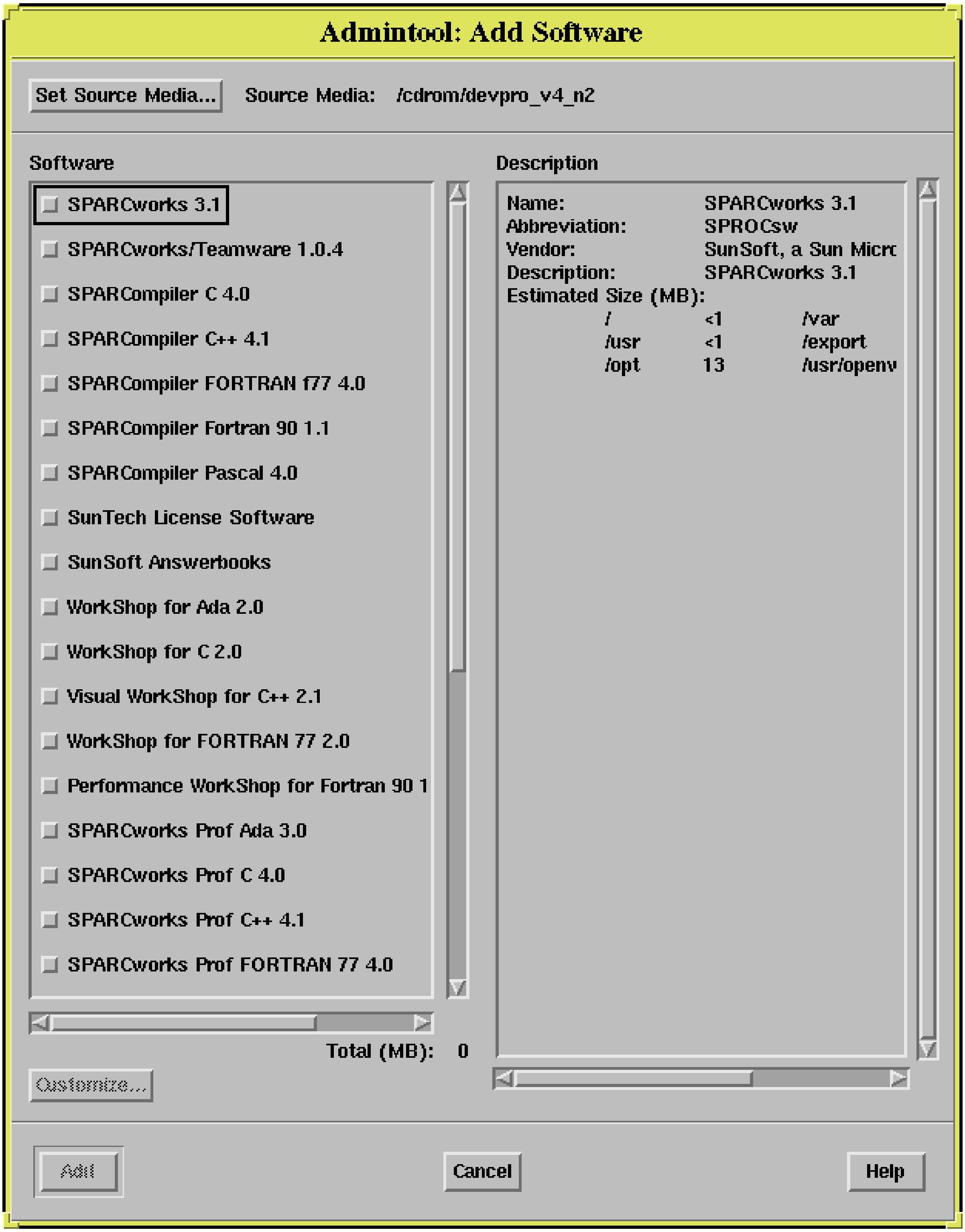


In the pop-up menu specify the source to use, click on the desired action and let it go to work. e.g., for the SunSoft Workshop Developers Products CDROM specify the CDROM mount point as below.

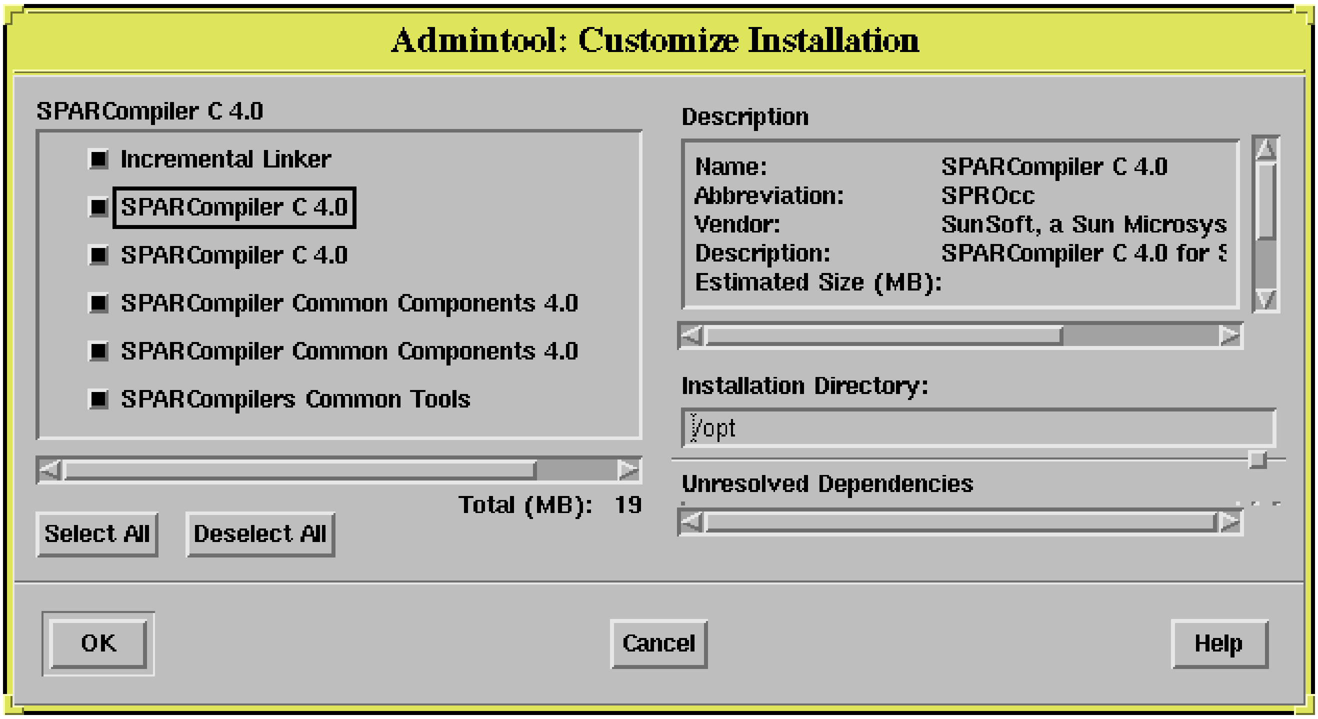


Click on "**OK**" and the tool will automatically read the package information from the CDROM and provide you with the choice of install options, as below.

Click on the desired package and start the install.



Clicking on **SPARCompiler C 4.0** and then choosing "**Customize...**" brings the next pop-up menu so that you can customize the installation process and start the install.



15.4 SunOS 4.X

SunOS 4.X uses ***/usr/etc/install/add_services*** to install system software. It doesn't keep any records of where the software is installed on the machine.

15.5 IRIX 5.X

IRIX uses the software installation tool, ***inst***. It can be invoked either from the command line, or in standalone mode from the miniroot. In the latter case we saw some examples in the Chapter on OS Installation. At the command line you can invoke inst either with command line options, or in interactive mode, e.g.:

```
inst

    Default distribution to install from: .

    For help on inst commands, type "help overview".

    Inst Main Menu

     1. from [source]               Specify location of software to be installed
     2. list [keywords] [names]     Display information about software subsystems
     3. go                          Perform software installation and removal now
     4. install [keywords] [names]  Select subsystems to be installed
     5. remove [keywords] [names]   Select subsystems to be removed
     6. keep [keywords] [names]     Do not install or remove these subsystems
     7. step [keywords] [names]     Interactive mode for install/remove/keep
     8. conflicts [choice ...]      List or resolve installation conflicts
     9. help [topic]                Get help in general or on a specific word
    10. view ...                    Go to the View Commands Menu
    11. admin ...                   Go to the Administrative Commands Menu
    12. quit                        Terminate software installation

    Inst>
```

inst keeps records of where software is installed in **/var/inst**.

15.6 Digital UNIX and Ultrix

Both Digital UNIX and Ultrix use ***setld*** to install system software. Mount the CDROM, change to the desired directory, e.g. **/mnt/RISC/BASE**, and run the ***setld*** command, i.e.:

```
# setld -l
```

You will then be prompted for the packages to load.

Software can be loaded from disk, CDROM, tape, or over the network from an install server.

A log is kept of the transactions in **/var/adm/smlogs/setld.log** (Digital UNIX) or **/etc/setldlog** (Ultrix).

You can use the ***fverify*** command to verify that the specified files have the correct files size, checksum, user id, group id, mode and file type as the installed file. ***/usr/lbin/fverify*** (Digital UNIX) or ***/etc/stl/fverify*** (Ultrix) will check the databases in **/usr/.smdb./*.inv** or **/usr/etc/subsets/***, respectively, for the inventory files.

CHAPTER 16 Backup Procedures

16.1 Backup Procedures

One of your most important functions as a System Administrator is to maintain the integrity of the data on your system. Since hardware does break and people make mistakes it is imperative that you make frequent backups of the file systems. That way in the event of a disk crash or accidental deletion of files you can recover a recent version of the data or program. Generally you back up data from disk to tape (1/2" 9-track, 200 MB; 1/4" cartridge, 150 MB; 4mm DAT, 1-12 GB; 8mm, 2-10 GB; or DLT 20 GB) or removable optical disks, for long term storage. If you have the disk space you can consider making backup copies of critical data files on other disk partitions. If your system doesn't have a backup medium, emphasize to the powers that be that *someday* your disk will crash and you will *not* be able to recover their data.

16.2 Backup strategies

Take a full dump of all the file systems soon after installation and personalizing the system. After this periodically take full backups of all file systems, e.g. weekly or monthly, and take incremental backups of all file systems weekly or daily, if needed. If your system is heavily used for file storage, or if the data stored there are hard to reproduce backup your file systems daily. Set up and stick to a regular schedule.

Backups should be done on quiescent file systems. This can be either single user mode, or with no one on the system, e.g. late at night.

16.2.1 Full backups

A full backup is a complete copy of all your file systems. Should your file system be blown away you can recreate it exactly as it was at the time of the full backup. These should be done monthly or weekly on each file system.

16.2.2 Incremental backups

Incremental backups copy only files that were added or changed since the last lower level dump are backed up. Since most of your files, e.g. system files, are static they will not be included in the dump.

This can save considerable space and time. A complete restoration of a damaged file system will them require the last full dump followed by the incremental dump(s). Incremental dumps should be done weekly or daily, depending on file system activity and importance.

16.3 Backup and Restore Commands

16.3.1 Dump

The program ***dump*** (SunOS 4.1.X, IRIX 5.X, Ultrix, Digital UNIX) or ***ufsdump*** (SunOS 5.X) can be used to backup a complete file system. There are 10 levels of dumps, 0-9. 0 is a full dump, while levels 1-9 are incremental dumps. The lower the number the more complete the dump. A level 1 dump will include everything changed since the last level 0 dump. A level 9 dump will only include those files changed since the last lower numbered dump. The manuals sometimes recommend some weird dump sequence involving every possible level through different days of the week, with a monthly period, to minimize tape usage. However, this makes it nearly impossible to figure out what you need to do to restore a particular file. Pick a simple schedule that's easy to follow and stick to it.

To use the ***/usr/etc/dump*** or ***/usr/sbin/ufsdump*** program, e.g. to an 8mm tape drive, we'll use a command line similar to the following to dump the root device, /dev/rsd0a.

```
/usr/etc/dump 0ufsdb /dev/nrst8 6000 54000 126 /dev/rsd0a
```

where ***0ufsdb*** call for:

0	- full dump; dump level (0->9).
u	- update the record for dumps, /etc/dumpdates.
f	- dump file; e.g. /dev/nrst8, where nrst indicates "no rewind".
s	- size of the tape volume you're dumping to, e.g. 6000 ft.
d	- tape density; e.g. 54000 bpi for 8mm tape.
b	- tape block size; e.g. 126

When you specify the size of the tape volume be conservative. Deliberately reduce it a few percent from the actual length, as the SunOS 4.1.X ***dump*** program doesn't know how to determine end-of-tape and will try to write to the full size specified, if needed. Also, any additional tapes needed for this backup will be assumed to have the same size as that specified for the first tape. The SunOS 5.X dump program, ***ufsdump***, can detect end-of-tape, and so the size parameter is not needed here.

The following is a sample ***dump*** output when backing up the /usr partition to a remote tape drive.

```
DUMP: Date of this level 0 dump: Sat Oct  1 04:56:03 1994
DUMP: Date of last level 0 dump: the epoch
DUMP: Dumping /dev/rsd0g (/usr) to /dev/nrst8 on host tardis
DUMP: mapping (Pass I) [regular files]
DUMP: mapping (Pass II) [directories]
DUMP: estimated 254102 blocks (124.07MB) on 0.07 tape(s).
DUMP: dumping (Pass III) [directories]
DUMP: dumping (Pass IV) [regular files]
DUMP: 41.11% done, finished in 0:07
```

```
DUMP: 81.87% done, finished in 0:02
DUMP: DUMP: 254102 blocks (124.07MB) on 1 tape
DUMP: DUMP IS DONE
DUMP: level 0 dump on Sat Oct  1 04:56:03 1994
DUMP: Tape rewinding
```

dump and ***ufsdump*** keep a record in **/etc/dumpdates** of files they have backed up in the form:

```
#filesystem          level   date
/dev/rsd0a           0       Sat Oct  1 04:54:52 1994
/dev/rsd0g           0       Sat Oct  1 04:56:03 1994
                -or, for SunOS 5.X-
/dev/rdsk/c0t3d0s0   0       Sat Oct  1 04:54:52 1994
/dev/rdks/c0t3d0s5   0       Sat Oct  1 04:56:03 1994
```

16.3.2 Sample backup scripts

You can write a script to do the periodic backups, requiring operator intervention, e.g.:

```
#     Script to do a complete backup of the system
#     A dataless system to a tape drive on a server.
echo "***************************************************************"
echo "This program will allow you to backup GALLIFREY onto magtape"
echo "        Follow the directions given below."
echo "***************************************************************"
echo "Mount tape for partition a and g"
echo "         then type RETURN "
read start
echo " ...working - Starting GALLIFREY backup   "
/usr/etc/dump 0ufsdb server:/dev/nrst8 6000 54000 126 /dev/sd0a && echo "Done with partition a ..."
/usr/etc/dump 0ufsdb server:/dev/rst8 6000 54000 126 /dev/sd0g && echo "Done with partition g ..."
/usr/bin/mt -f /dev/rst8 rewoffl
```

Backup scripts similar to this can be run by you as root, or by an operator. You may wish to set up a ***login*** in **/etc/passwd** similar to:

```
backup:ogHt5C0Z.bcD2:20:5:Remote Backup to Server:/etc/adm:/etc/adm/backup
```

where the backup script is the shell, ***/etc/adm/backup***. You or the operator would login as ***backup*** and the program would run at login. When the program terminates you would be logged out.

You can also set up your backup script to be run by cron. If the tape is large enough to hold the entire backup the following script could be set to run periodically. You will just need to make sure that the proper tapes are in the drive at the appropriate times.


```
#!/bin/sh
#       Cron script to do a complete backup of the system
#/dev/sd0a             15663     6335     7762    45%    /
#/dev/sd0g            138511   101061    23599    81%    /usr
#/dev/sd2h            268319     4208   237280     2%    /home
#/dev/sd2f            458671    36844   375960     9%    /usr/local
#/dev/sd2a             47711      297    42643     1%    /var
HOST=`hostname`
admin=frank
Mt=/bin/mt
Dump=/usr/etc/dump
device=/dev/nrst0
size=6000
dens=54000
blksz=126
# Failure - exit
failure () {
        /usr/ucb/mail -s "Backup Failure - $HOST" $admin << EOF
$HOST
Cron backup script failed.  Apparently there was no tape in the device.
EOF
        exit 1
        }
# Dump Failure - exit
dumpfail () {
        /usr/ucb/mail -s "Backup Failure - $HOST" $admin << EOF
$HOST
Cron backup script failed.  Could not write to the tape.
EOF
        exit 1
        }
# Success
success () {
        /usr/ucb/mail -s "Backup completed successfully - $HOST" $admin << EOF
$HOST
Cron backup script was apparently successful.  The /etc/dumpdates file is:
`/bin/cat /etc/dumpdates`
EOF
        }
# Confirm that the tape is in the device
$Mt -f $device rewind || failure
$Dump 0ufsdb $device $size $dens $blksz /dev/sd0a || dumpfail
$Dump 0ufsdb $device $size $dens $blksz /dev/sd0g || dumpfail
$Dump 0ufsdb $device $size $dens $blksz /dev/sd2h || dumpfail
$Dump 0ufsdb $device $size $dens $blksz /dev/sd2f || dumpfail
($Dump 0ufsdb $device $size $dens $blksz /dev/sd2a || dumpfail) && success
$Mt -f $device rewoffl
```

16.3.3 Restore

You can restore entire file systems or you can interactively restore individual files with the restore program, ***restore*** (SunOS 4.1.X) or ***ufsrestore*** (SunOS 5.X). These programs restore files relative to your current directory. On a full restore they place a file, **restoresymtable**, in the current directory, that's used to pass information to a further instance of restore, for restoring incremental dumps. This file can be safely removed only after all of the incremental dumps have been restored.

To do a complete restore of a damaged file system, e.g. /dev/sd0h, you might try:

```
# newfs /dev/rsd0h          - to clear and re-create the file system.
# mount /dev/sd0h /mnt      - to mount the file system temporarily.
# cd /mnt                   - move to the new file system.
# restore -r                - restore a level 0 dump of the file system.
Later, incremental dumps can then be restored.
# umount /mnt               - unmount the file system.
# fsck /dev/rsd0h           - check the file system for consistency.
# mount /dev/sd0h /home- mount the file system
```

Restore can also be run interactively and you can specify the device, e.g.:

```
# restore -if /dev/rst9
```

restore then first recreates the file system in memory so that you can use some UNIX type commands, i.e. ***ls***, ***cd***, and ***pwd***, to move around the file system. You can then "***add***" entries to a table of files to "***extract***" from the tape.

A special case is restoration of the **root** file system. For this you need to boot from tape or CDROM. After restoring the file system you also need to re-install the boot block program, ***bootblk***. This is done with ***installboot***, as in the following for a SCSI disk on SunOS 4.1.X:

```
# /usr/mdec/installboot /boot bootsd /dev/rsd0a
```

and for SunOS 5.X:

```
# /usr/sbin/installboot /usr/platform/‘uname -i‘/lib/fs/ufs/bootblk /dev/rdsk/c0t3d0s0
```

As you can see the syntax is dependent on both the hardware platform and software version, so read the man page before using ***installboot***.

16.3.4 Remote dumps and restores

Dumps and restores can be done locally or remotely, across a network. The major difference is that when you specify location of the backup media for the remote device you need to include the system name, e.g. ***tardis:/dev/nrst8***. You can also specify a different user on the remote machine, e.g.: ***frank@tardis:/dev/nrst8***. The remote machine's **/etc/hosts.equiv** or user's **.rhosts** file would have to allow access.

16.3.5 Tape Archive program, tar

The tape archive program, tar, can be used to copy files to and from tape or across a network. If you're working with individual files or directories you'll probably want to use ***tar*** for this service. Most UNIX systems have tar, so it's convenient for moving files between different systems. UNIX source archives, e.g. those on **archive.cis.ohio-state.edu** and many other places, are often stored as compressed tar files. Compression generally saves 1/2 - 2/3 of the original file space. A compressed tar file usually has a name similar to **filename.tar.Z**, where the "**Z**" stands for Lempel-Ziv compression. The GNU compression program, gzip, uses a different compression scheme, signified by ending the filename with "**z**" or "**gz**". A compressed file is a binary file.

To generate a tar file:

```
# tar -cvf filename.tar list-of-files
```

This puts the files into tar format and stores them in ***filename.tar***.

You could just as easily put them onto tape, e.g.:

```
# tar -cvf /dev/rmt8 list-of-files
```

Some of the options for tar are:

- **c** - create a new tarfile.
- **v** - verbose, print out the file names as they are archived.
- **f** - use the next argument as the output file.
- **t** - list the files

You can extract files from tape or a tarfile with:

```
# tar -xvf /dev/rmt8
```

So if you have a compressed tarfile you would first use ***uncompress*** to uncompress the tarfile, then ***tar*** with the "***-x***" option on the tarfile to extract the programs and directories, similar to:

```
# uncompress filename.tar.Z- which produces "filename.tar" as output.
# tar -xvf filename.tar    - which extracts the files, e.g.
    filename/Makefile
    README
    main.c
    header.h
     ...
```

16.3.6 cpio

cpio copies files in and out of a cpio archive. The Solaris 2.X packages on the install CDROM are compressed cpio archives. To examine one of these packages:

```
# zcat file | cpio -idumB
```

where the cpio options indicate:

- **i** copy in
- **d** create directories as needed
- **u** copy unconditionally, even replacing newer files of the same name
- **m** retain modification times
- **B** block I/O 5120 bytes/record
- **t** list the table of contents of the input file

You can create your own cpio archives with ***cpio***. ***cpio*** reads and writes to stdin and stdout, respectively.

PART II Network Services

Service Access Facility

The Network

Network Administration

Distributed File System Administration

Network Information Services

Adding Clients

Usenet

CHAPTER 17 Service Access Facility

17.1 Overview of Service Access

The **Service Access Facility** (**SAF**) is used by SunOS 5.X to control access to terminals, modems and network services, such as remote print requests. It's designed to be flexible and to treat local and network requests in a similar fashion. The init and login programs have been re-written for Solaris 2 and part of their previous functions now belong to SAF. SAF is not a program, but rather a package of daemons and administrative commands.

The **Service Access Controller** (***sac***) is the master SAF process. It's spawned by ***init*** at run level 2 and controls the port monitors. It can add or remove and start or stop the port monitors. The port monitors control either a serial or network port. They connect incoming requests to services, which are arbitrary processes, such as login. The port monitor administrative commands can add or delete and start or stop services for the ports.

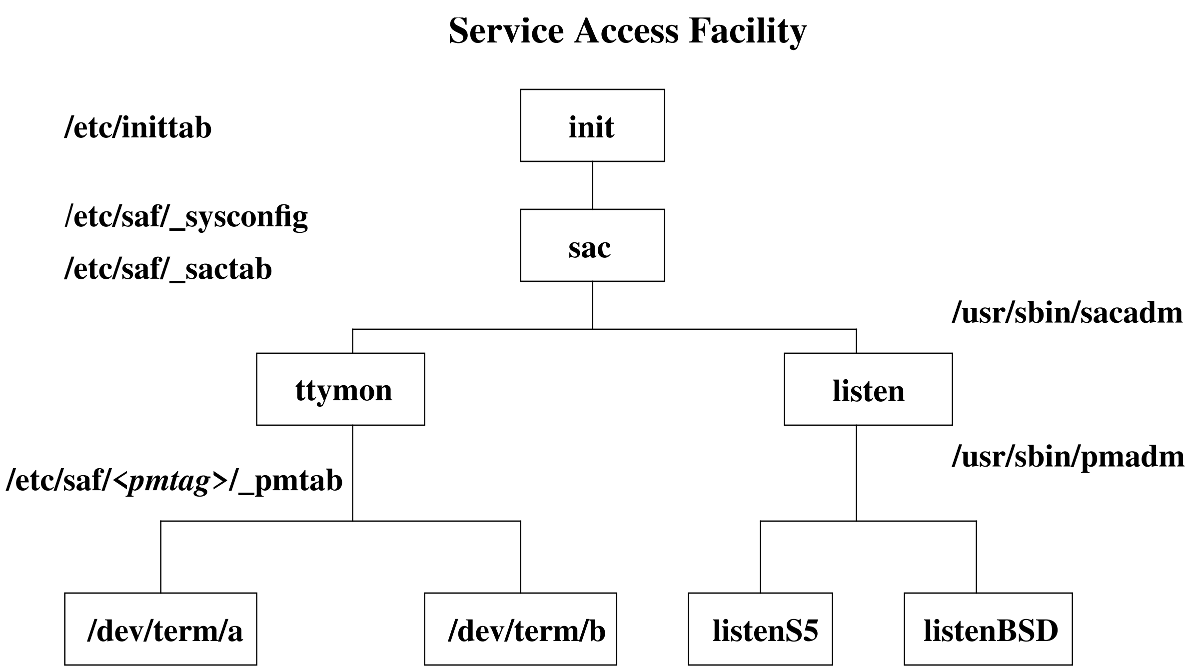
The ***getty*** process is no longer used. It was considered too inflexible in that the only service it provided was ***login***. Also, it didn't scale well to large numbers of ports, as you had to run a ***getty*** process for every port it monitored.

The SAC administers the port monitors with the ***sacadm*** command. Each port monitor can control one or many ports. The ports are administered through the ***pmadm*** command. The ***pmadm*** command controls the services provided by ***ttymon*** and ***listen***. One ***ttymon*** daemon serves multiple serial ports and one ***listen*** daemon provides multiple services to the network ports.

Serial ports can be configured using the **Serial Port Manager** facility of the ***admintool*** GUI described in a later chapter, or by using the commands specified below.

17.2 Service Access Facility Overview

FIGURE 17.1 **Service Access Facility Overview**



17.3 Service Access Controller

The Service Access Controller follows these steps:

1. The sac program is started by ***init*** when entering run level 2 through an entry in **/etc/inittab**,

    sc:234:respawn:/usr/lib/saf/sac -t 300

2. When ***sac*** is invoked it reads the configuration script **/etc/saf/_sysconfig** to customize it's environment.
3. After interpreting the **_sysconfig** file ***sac*** reads it's administrative file, **/etc/saf/_sactab**, which specifies the port monitors to start.
4. For each port monitor ***sac*** starts it forks a child process.
5. Each child process then interprets its specific port monitor configuration script, **/etc/saf/<*pmtag*>/_config**. Lastly the child process execs the port monitor specified by the **_sactab** entry.

An entry is made in **/var/saf/_log** whenever ***sac*** starts or stops a port monitor.

17.4 Port Monitors

There are two types of port monitors, TTY monitors that listen for incoming connections on serial devices, and network listeners, that listen for incoming requests on the network ports.

17.4.1 ttymon

The ***ttymon*** monitor manages the TTY ports. It monitors, sets terminal modes, baud rates, and starts the specified service, e.g. login, for the port. The ttymon process sets the line disciplines according to the values specified in **/etc/ttydefs** (which replaces ***gettytab***). ***ttymon*** replaces the ***getty*** process. A single ***ttymon*** command can monitor multiple ports. This command is configured using the ***sacadm*** program and it's services are specified using the ***pmadm*** and ***ttyadm*** commands.

17.4.2 listen

The network port manager is ***listen***. Requests received through the network, such as remote print requests, are processed by listen, which invokes the appropriate servers to provide the service. The listen daemon is configured by ***sacadm*** while specific service information is provided by the ***pmadm*** and ***nlsadmin*** commands.

17.4.3 sacadm

The ***sacadm*** command administers both ***ttymon*** and ***listen***. It can be used to add and remove, start and stop, and enable and disable port monitors.

17.4.4 pmadm

The ***pmadm*** command associates a service with an instance of a port monitor. It embeds a ***ttyadm*** or ***nlsadmin*** command when invoked to provide specific information for a port monitor.

17.4.5 ttyadm

The ***ttyadm*** command provides information specific to ***ttymon*** to the ***pmadm*** command. Information such as whether the port is bi-directional, which STREAMS modules to push, the baud rate, and the service to provide.

17.4.6 nlsadmin

The ***nlsadmin*** command provides information specific to ***listen*** to the ***pmadm*** command. Information such as the full path name of the server process, or the FIFO, or the named STREAM that the server uses to listen for services.

17.5 Setting Up a Terminal

To configure an ASCII terminal for login service perform the following steps.

1. Connect the terminal to the system. Here we'll assume that it is serial port A.
2. Issue the ***sacadm*** command to add a ttymon port monitor named **zsmon**:
   # sacadm -a -p zsmon -t ttymon -c /usr/lib/saf/ttymon -v ‘ttyadm -V‘
   where
   -a is the flag to **add** the port
   -p specifies the **pmtag** zsmon as the port monitor tag
   -t specifies the **type** of port monitor as ttymon
   -c defines the **command** string to start the port monitor
   -v specifies the **version** number of the port monitor (provided by ttyadm -V)
   You can use the sacadm program to report the status for a port monitor:
   # sacadm -l

PMTAG	PMTYPE	FLGS	RCNT	STATUS	COMMAND
zsmon	ttymon	-	0	ENABLED	/usr/lib/saf/ttymon

3. Should you need to you can remove any existing service for ttya do the following:
   # pmadm -r -p zsmon -s ttya
   where
   -r indicates to **remove** a service from the port monitors administrative file
   -p specifies the **pmtag** zsmon associated with the port monitor
   -s specifies the **service** tag
4. Use ***pmadm*** to associate the port monitor with the new service:
   # pmadm -a -p zsmon -s a -i root -fu -v ‘ttyadm -V‘ -m "‘ttyadm -l 9600 -d /dev/term/a -i ’terminal disabled’ -s /usr/bin/login -T tvi925 -S y‘"
   where for ***pmadm*** we have the options
   -a indicates to **add** a service to the port monitor administrative file
   -i is the **identity** assigned to the service tag when it's started (must be in **/etc/passwd**)
   -f specifies one or both of the two **flags**:
      x do not enable the service through the port monitor
      u create a utmp entry for the service
   -v specifies the **version** number (using ttyadm for ttymon, nlsadmin for listen)
   -m specifies the part of the **port monitor** administrative file entry for this service,
   and for ***ttyadm*** we have the options
   -l specifies the **label** in the **/etc/ttydefs** file to use as the starting point for the baud rate
   -d specifies the full path name of the **device** file for the TTY port
   -i specifies the **inactive** (disabled) message to sent to the port when it is disabled
   -s specifies the **service** to be invoked when a connection request is received by the port
   -T sets the default **terminal** type
   -S sets the **software** carrier value
      y turns software carrier on
      n turns software carrier off

5. This puts an entry in **/etc/saf/zsmon/_pmtab** and enables the terminal. You can read this file with:

```
# pmadm -l
    PMTAG     PMTYPE     SVCTAG   FLGS   ID      <PMSPECIFIC>
    zsmon     ttymon     ttya     u      root    /dev/term/a I - /usr/bin/login - 9600
    ldterm,ttcompat ttya login:  - tvi925 y  #
```

The actual entry in the **_pmtab** file is:

```
# VERSION=1
ttya:u:root:reserved:reserved:reserved:/dev/term/a:I::/usr/bin/login::9600:ldterm,ttcompat:ttya login\: ::tvi925:y:#
```

Rather than typing in the ***sacadm*** and ***pmadm*** commands on the command line as done above, with all the possibilities for error, you can edit their control files and add your entries. To have these take effect you execute the sacadm command with the ***-x*** option, telling it to reread it's database files for the ***-p pmtag*** specified, e.g.:

1. Create **/etc/saf/_sactab**:

```
# cat <<EOF_SACTAB > /etc/saf/_sactab
# VERSION=1
zsmon:ttymon::0:/usr/lib/saf/ttymon:     # TTY Port Monitor
EOF_SACTAB
```

2. Make a directory for the **zsmon** port monitor

```
# mkdir /etc/saf/zsmon
```

Change to the new directory and create the **_pmtab** file

```
# cd /etc/saf/zsmon
# cat <<EOF_PMTAB >  _pmtab
# VERSION=1
ttya:u:root:reserved:reserved:reserved:/dev/term/a:I::/usr/bin/login::9600:ldterm,ttcompat:ttya login\: ::tvi925:y:#
EOF_PMTAB
```

3. Create the **log** directory and file

```
# mkdir /var/saf/zsmon
# touch /var/saf/zsmon/log
```

4. Re-initialize ***SAF***

```
# sacadm -x -p zsmon
```

17.5.1 TTY Monitor Control Commands

You control the TTY port monitor with ***sacadm***, e.g.:

# sacadm -e -p zsmon	- enable a port monitor to allow it to service new requests
# sacadm -d -p zsmon	- disable a port monitor, this prevents it from starting new services for incoming connections, but does not affect present services
# sacadm -s -p zsmon	- restart the port monitor
# sacadm -k -p zsmon	- kill the TTY port monitor

17.5.2 Removing a TTY Monitor

Removing a port monitor deletes the information in the configuration file associated with the port monitor. Port monitor information in the configuration files can not be updated or changed with ***sacadm***. To reconfigure a port monitor delete it and add a new one,

```
# sacadm -r -p zsmon
```

17.5.3 Adding Services to a TTY Monitor

The ***pmadm*** command is used to add services to a port monitor. The ***service tag*** and ***port monitor tag*** uniquely identify the service. So when specifying the pmadm command to add a service you need to specify both the service (***-s***) and port monitor instance (***-p***) through which the service is made available.

17.5.4 Disable/Enable a TTY Service

To disable a TTY service use ***pmadm*** with the "***-d***" option; to re-enable it use the "***-e***" option to ***pmadm***:

```
# pmadm -d -p zsmon -s a
# pmadm -e -p zsmon -s a
```

17.6 Network Port Monitors

The ***listen*** port monitor is the network listener daemon. Multiple services can be provided by each instance of ***listen***. The administrative files for ***listen*** are configured using the ***pmadm*** and ***nlsadmin*** commands.

The network listener plays a role similar to that of ***inetd*** and provides additional services, such as network printing service between System V and BSD machines.

The listen monitor can be

- **enabled** monitoring requests for service and invoking the responsible servers
- **disabled** not monitoring new requests, but previous servers remain functional
- **killed** then all servers previously invoked by this instance of listen are disabled.

17.6.1 Adding a Listener

The ***sacadm*** command is used to add the listener:

```
# sacadm -a -p tcp -t listen -c /usr/lib/saf/listen -v ‘nlsadmin -V‘
```

where:	***-a***	**adds** the port monitor
	-p	specifies the **pmtag** associated with the port monitor
	-t	specifies the port monitor **type**
	-c	specifies the **command** to execute when starting the port monitor
	-v	specifies the **version** number (from ***nlsadmin***)

17.6.2 Listing a Listener

You can use the ***sacadm*** command to list the status of the listener,

```
# sacadm -l -p tcp
    PMTAG    PMTYPE    FLGS    RCNT    STATUS     COMMAND
    tcp      listen    -       0       ENABLED    /usr/lib/saf/listen tcp #
```

17.6.3 Listener Control Commands

To control a listener use the ***sacadm*** command for the ***tcp*** port monitor, i.e.:

```
# sacadm -e -p tcp     - enable a listener
# sacadm -s -p tcp     - start a listener
# sacadm -d -p tcp     - disable a listener
# sacadm -k -p tcp     - kill a listener
# sacadm -r -p tcp     - remove a listener
```

17.6.4 Adding Services to the Listener

The ***nlsadmin*** command is used to present listen-specific configuration information to the ***pmadm*** command. It associates an instance of a listen process with the specific service called for by that listener.

To add a service use ***pmadm*** and ***nlsadmin***, e.g. to add a listen process for the SunOS5.X print request:

```
# pmadm -a -p tcp -s lp -i root -v ‘nlsadmin -V‘ -m "‘nlsadmin -o /var/spool/lp/fifos/listenS5‘"
```

where for ***nlsadmin*** the option

-o specifies the full pathname to the FIFO or named STREAM used by the server process to receive the connection

17.6.5 List a Listen Status

You can use the ***-l*** option to the ***pmadm*** command to check the listener status,

```
# pmadm -l -p tcp
    PMTAG    PMTYPE    SVCTAG    FLGS    ID      <PMSPECIFIC>
    tcp      listen    lp        -       root    - - p - /var/spool/lp/fifos/listenS5 #
```

17.6.6 Enable a Listen Service

Enable a listen service with the ***-e*** option to ***pmadm*** and specify appropriate service ***tag***:

```
# pmadm -e -p tcp -s lp
```


17.7 Terminal Control

SunOS 5.X uses the System V **terminfo** database for terminal control by default, not the termcap file of BSD. The **termcap** file and its associated utilities are provided with the compatibility package, but are not intended for general use.

The serial port naming convention has been changed for SunOS 5.X. It now allows for more ports. The device names of SunOS 4.X are still there as symbolic links to the new names, e.g. **/dev/ttya** is a symbolic link to **/dev/term/a**, which is a symbolic link to the physical device, **/devices/zs@1,f1000000:a**.

When logging in ***stty*** and ***tput*** are used to configure the terminal I/O values to match the characteristics expected for the terminal.

17.7.1 The terminfo database

The terminfo database contains the descriptions of the terminal capabilities. It's used by programs such as ***vi*** and the **curses** package to control the screen.

The database is kept in the **/usr/share/lib/terminfo** directory with subdirectories specified by the first character of the names of the **terminfo** files. Each **terminfo** description is a separate, compiled file, within the subdirectory matching the first character of its name [1-9,a-z,A-Z].

The **terminfo** files are functionally equivalent to the individual entries in the **termcap** file, however, the **terminfo** files are in a compiled format, not ASCII, as is the **termcap** file. The entries in a **terminfo** file are described in the terminfo(4) man page.

You can display the contents of a **terminfo** file in a format similar to a **termcap** entry using the ***infocmp*** command. With no options and no TERMINFO environment variable set ***infocmp*** assumes the desired terminfo file is that of the TERM variable, e.g.:

```
# infocmp
    #       Reconstructed via infocmp from file: /usr/share/lib/terminfo/x/xterm
    xterm|vs100|xterm terminal emulator,
            am, km, mir, msgr, xenl,
            cols#80, it#8, lines#65,
            acsc=``aaffggjjkkllmmnnooppqqrrssttuuvvwwxxyyzz{{||}}~~,
            bel=^G, blink=@, bold=\E[1m, clear=\E[H\E[2J, cr=\r,
            csr=\E[%i%p1%d;%p2%dr, cub=\E[%p1%dD, cub1=\b,
            cud=\E[%p1%dB, cud1=\n, cuf=\E[%p1%dC, cuf1=\E[C,
    ...
            sgr0=\E[m, smacs=^N, smkx=\E[?1h\E=, smso=\E[7m,
            tbc=\E[3g,
```


17.7.2 Terminal Definitions

The terminal definitions are contained in the **/etc/ttydefs** file. ***ttymon*** uses this file to configure the terminal and set the baud rates for the ports. The **ttydefs** file contains fields for the following:

- ***ttylabel*** string to match the TTY port's **ttylabel** field
- ***initial-flags*** the initial **termio** settings
- ***final-flags*** the final **termio** settings for the connection
- ***autobaud*** **A**=enabled, **blank**=disabled
- ***nextlabel*** **next** ttydefs entry to try

A few of the entries in the **ttydefs** file are:

```
38400:38400 hupcl:38400 sane hupcl::19200
19200:19200 hupcl:19200 sane hupcl::9600
9600:9600 hupcl:9600 sane hupcl::4800
4800:4800 hupcl:4800 sane hupcl::2400
2400:2400 hupcl:2400 sane hupcl::1200
1200:1200 hupcl:1200 sane hupcl::300
300:300 hupcl:300 sane hupcl::38400
auto:hupcl:sane hupcl:A:9600
console:9600 hupcl opost onlcr:9600 sane::console
pty:9600 hupcl opost onlcr:9600 sane::pty
```

17.7.3 Changing Terminal Definitions

The ***sttydefs*** command is used to maintain the terminal definitions file. This command can add and remove line settings and use hunt sequences in the **ttydefs** file. The change options can only be used by root, but anyone can use the ***-l*** option to display the entry for a particular label, e.g.:

```
% sttydefs -l 9600
    ------------------------------------------------
    9600:9600 hupcl:9600 sane hupcl::4800
    ------------------------------------------------
    ttylabel:          9600
    initial flags:     9600 hupcl
    final flags:       9600 sane hupcl
    autobaud:          no
    nextlabel:         4800
```

The options available to ***sttydefs*** are:

-l	display matching ttylabel
-a	**add** a record using the ttylabel
-b	enable autobauding
-n	specify the **nextlabel** value
-i	specify the **initial-flags** value
-f	specify the **final-flags** value
-r	**remove** the record for ttylabel

17.8 Summary

In summary we have the following tables for the SAF commands and control files.

TABLE 17.1 **Command Summary**

Command Name	Description
sacadm	***sac*** administrative command to add, remove, disable, enable, and monitor port monitors
pmadm	service administration command to associate a ***port*** monitor instance with the service to be provided
ttyadm	provides ***ttymon*** specific information, such as port device name, to the ***pmadm*** command
nlsadmin	provides ***listen*** specific information, such as the server that will answer requests, to the pmadm command

TABLE 17.2 **File Summary**

File Name	Description
/etc/saf/_sysconfig	system configuration script
/etc/saf/_sactab	***sac*** administrative file to configure the port monitors controlled by ***sac***
/etc/saf/<pmtag>	directory for the ***pmtag*** port monitor
/etc/saf/<pmtag>/_pmtab	administrative file to control the port monitor configuration for the services provided by ***pmtag***
/var/saf/_log	log file for ***sac***
/var/saf/<pmtag>	directory for the log files created for ***pmtag***

CHAPTER 18 The Network

18.1 The Network

Network types and access - Local Area Network (LAN) and Wide Area Network (WAN). LAN → high speed connection between machines within a site. WAN → geographically remote machines. Ethernet uses Carrier Sense Multiple-Access Collision Detection (CSMA/CD) to share a single transmission line. This means the system listens on the line prior to attempting to send a packet, any system can send at any time, and should a collision occur the system will sense this and retransmit after a random delay.

The International Standard Organization's Open System Interconnection (ISO/OSI) model is used by Sun and many other vendors as the network protocol. From the top down the layers are:

TABLE 18.1 **Network Protocol Layers**

Practical	ISO/OSI	Function
Application	*Application*	provides network services, e.g. mail, ftp, telnet, NFS, YP, DNS, WWW
	Presentation	XDR (eXternal Data Representation); transformation services such as text compression, conversion between character codes (EBCDIC → ASCII), etc. so that it can be recognized by other machines.
	Session	RPC (Remote Procedure Call); enables programs to establish connections with each other via names rather than socket addresses; recovers from failed connections.
Transport	*Transport*	TCP (Transmission Control Protocol), UDP (User Datagram Protocol); TCP provides reliable communication between pairs of processes on the network, it establishes connections through "sockets" which are determined from the IP address and the port number; UDP provides a low overhead transmission service, but with less error checking.
IP	*Network*	IP (Internet Protocol); connects subnets to the Internet; handles fragmentation/recombination, routing and buffering; initiates and terminates connections between machines.
Physical	*Data Link*	defines data frames; controls data encapsulation; detects and possibly corrects errors; determines how the line is to be shared by the multiple machines.
	Physical	provides an electrical connection, e.g. through coaxial cable, between machines; defines procedures for starting and ending sessions; transfers packets.

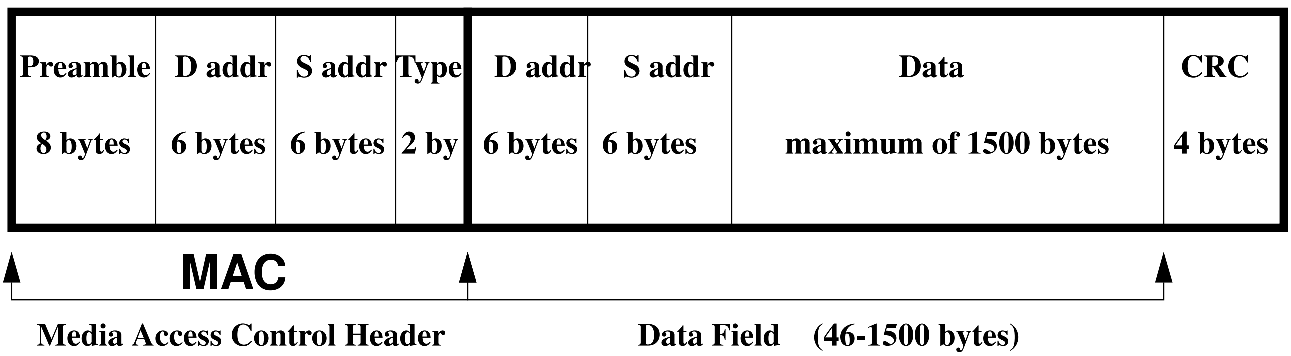
18.2 Hardware used in a network

- ***Controller*** - e.g. Intel (ie0 - Sun3, Sun4), Lance (le0 - Sun4m, Sun4c, Sun3/50, Sun3/60).
- ***Transceiver Cable*** - connects the controller to the transceiver box.
- ***Transceiver Box*** - electrically isolates the system from the rest of the network.
- ***Coaxial Cable*** - the ethernet backbone.
- ***Switch*** - examines the data packet to determine the destination, then sends the packet only over the segment hosting the recipient machine. If the packet is addressed to a machine on the same segment, the packet never leaves that segment. This minimizes traffic on the network segments that don't need to see the packet.
- ***Bridge*** - operates at the Data Link layer. Designed for transparent connection of networks. Bridges and Switches allow you to break the network into smaller segments that increase the overall throughput of the total network.
- ***Router*** - joins 2 networks at the network layer; forwards packets of a particular protocol from one subnet to another; translates messages between different protocols, e.g. DECnet and TCP/IP.
- ***Gateway*** - joins different types of networks; translates one protocol into another, e.g. between OSU's SONNET backbone and the local subnets.
- ***Terminal Server*** - attach systems on a local area network to serial devices, e.g. terminals and printers; may support LAT and Telnet protocols.

18.3 Ethernet Frame

Ethernet traffic is transported in units of a **frame**, where each frame has a definite beginning and end. The form of the frame is in the figure below.

FIGURE 18.1 **Ethernet Frame**



In this model we define:

- **Preamble** Field used for synchronization, 64-bits
- **Destination Address** Ethernet address of the destination host, 48-bits
- **Source Address** Ethernet address of the source host, 48-bits
- **Type** Type of data encapsulated, e.g. IP, ARP, RARP, etc, 16-bits.
- **Data Field** Data area, 46-1500 bytes, which has
  Destination Address Internet address of destination host
  Source Address Internet address of source host
- **CRC** Cyclical Redundancy Check, used for error detection

The data to be sent is encapsulated by each layer, from the Application down to the Physical, and each adds it's own header information. When data is received each layer strips off it's header and then passes the packet up to the next layer. The Transport Layer makes sure that the source and destination, hosts and ports, can be identified, and includes a sequence number so that a file can be broken into multiple packets and recombined on the receiving end. The Internet Layer determines how the frames will be delivered, including fragmenting them to send along a path with a smaller maximum transmission unit (MTU) or recombining them for a larger MTU path. It determines the routing used to get to the destination. The Network Layer provides the encapsulation of the datagram into the frame to be transmitted over the network. It includes the ethernet addresses of the source machine and of the next hop towards the destination. These addresses are rewritten with each hop.

18.4 Trouble shooting the Ethernet

Some common error messages related to the network that you might come across are:

- **le0 no carrier - transceiver cable problem?**
  Check the transceiver cable to make sure that you are properly connected to the network. Sun, especially on their older boxes, is notorious for having bad connections here; try a different transceiver box. It could also be another hardware problem on the network, such as a damaged cable, or a faulty bridge or router.
- **le0 ethernet jammed**
  Make sure that the ethernet cable is terminated at both ends.
- **unknown host**
  The remote hostname can't be resolved into an IP address. Try using the IP address. If this works you need to check your name resolution.
- **network unreachable**
  Your machine doesn't have a route to the remote host. Use "***netstat -rn***" to check the routing tables and set a default route if necessary.
- **no answer** or **Connection timed out** or **cannot connect**
  Your machine has a route to the remote host, but is not receiving any response from it. The network may be down, or the remote host may not have a route back to your machine, or one or both machines may be misconfigured. Check your network configuration with ***ifconfig*** and ***netstat***.

18.4.1 etherfind

With SunOS 4.1.X you can use ***etherfind*** to examine network traffic. For ***etherfind*** to work your network interface must be in **promiscuous** mode, i.e. have the appropriate streams NIT support enabled in the kernel. This support is required for a diskless boot server, but is something you may want to disable on other machines.

To examine all traffic originating or terminating at the workstation "nyssa":

```
# etherfind -p -i le0 -src nyssa -o -dst nyssa
                              icmp type
    lnth    proto   source           destination   src port   dst port
     60     tcp     leela.acs.ohio   nyssa         login      1021
    138     udp     tardis           nyssa         2049       1023
    138     udp     tardis           nyssa         2049       1023
    118     udp     tardis           nyssa         652        684
     74     tcp     leela.acs.ohio   nyssa         login      1021
     60     tcp     leela.acs.ohio   nyssa         login      1021
```

To examine traffic between machines "nyssa" and "leela" the command would be:

```
# etherfind -p -i le0 -between nyssa leela
```

18.4.2 snoop

SunOS 5.X has the ***snoop*** command to allow you to inspect packets on the network. This command has numerous options for determining which packets to examine. To examine all packets to or from host "nyssa" you would execute:

```
# /usr/sbin/snoop host nyssa
Using device le0 (promiscuous mode)
    nyssa.acs.ohio-state.edu -> ace.acs.ohio-state.edu RSTAT C Get Statistics
    ace.acs.ohio-state.edu -> nyssa.acs.ohio-state.edu RSTAT R Get Statistics
    tardis.acs.ohio-state.edu -> nyssa.acs.ohio-state.edu XWIN C port=1085
    nyssa.acs.ohio-state.edu -> tardis.acs.ohio-state.edu XWIN R port=1085
    nyssa.acs.ohio-state.edu -> gallifrey.acs.ohio-state.edu RSTAT C Get Statistics
    gallifrey.acs.ohio-state.edu -> nyssa.acs.ohio-state.edu RSTAT R Get Statistics
    tardis.acs.ohio-state.edu -> nyssa.acs.ohio-state.edu XWIN C port=1085
    nyssa.acs.ohio-state.edu -> peri.acs.ohio-state.edu RSTAT C Get Statistics
```

which displays the originating and destination addresses, the protocol used, and the port used.

18.4.3 IRIX 5.X

IRIX's NetVisualizer product contains ***netsnoop*** and other programs useful for analyzing network problems. ***netsnoop*** will allow you to check for bad ethernet packets, etc.

18.4.4 Digital UNIX

Digital UNIX has the PD program, ***tcpdump***, available for network packet analysis. This will provide information similar to Sun's ***snoop*** program.

18.5 Monitoring the network

18.5.1 arp

The address resolution protocol program, ***arp***, is useful for determining other machines broadcasting on your subnet. The ***-a*** option will display the current ARP entries from the kernel, e.g.:

```
% arp -a
Net to Media Table
Device   IP Address                     Mask              Flags   Phys Addr
------   --------------------           ---------------   -----   ---------------
le0      gallifrey.acs.ohio-state.edu   255.255.255.255           08:00:20:0c:63:66
le0      tardis.acs.ohio-state.edu      255.255.255.255           08:00:20:06:85:c9
le0      ace.acs.ohio-state.edu         255.255.255.255           08:00:20:0c:3f:ec
le0      nyssa.acs.ohio-state.edu       255.255.255.255   SP      08:00:20:0c:a2:93
le0      224.0.0.0                      240.0.0.0         SM      01:00:5e:00:00:00
```

18.5.2 ping

ping sends an echo packet to the network host and reports on whether or not it replies, e.g.:

```
% ping nisca
nisca.acs.ohio-state.edu is alive
```

18.5.3 traceroute

traceroute is a PD program for tracing the route taken by a packet enroute to a host. To trace a packet to SunSite (Sun's anonymous ftp archive at UNC) execute:

```
% traceroute sunsite.unc.edu
  traceroute to sunsite.unc.edu (152.2.22.81), 30 hops max, 40 byte packets
   1  son-se7-eth2.acs.ohio-state.edu (128.146.6.1)  10 ms  0 ms  10 ms
   2  son-se4-eth3.acs.ohio-state.edu (164.107.100.1)  10 ms  10 ms  0 ms
   3  kc1.acs.ohio-state.edu (128.146.3.1)  20 ms  10 ms  10 ms
   4  cicnet.ohio-dmz.net (192.68.143.1)  20 ms  30 ms  10 ms
   5  um-osu.cic.net (131.103.11.46)  20 ms  30 ms  30 ms
   6  fd-0.enss131.t3.ans.net (192.203.195.1)  20 ms  30 ms  20 ms
   7  t3-2.Cleveland-cnss41.t3.ans.net (140.222.41.3)  30 ms  20 ms  40 ms
   8  t3-3.Cleveland-cnss40.t3.ans.net (140.222.40.4)  30 ms  20 ms  20 ms
   9  t3-1.New-York-cnss32.t3.ans.net (140.222.32.2)  50 ms  40 ms  50 ms
  10  t3-1.Washington-DC-cnss56.t3.ans.net (140.222.56.2)  40 ms  50 ms  60 ms
  11  t3-2.Greensboro-cnss72.t3.ans.net (140.222.72.3)  60 ms  60 ms  60 ms
  12  t3-0.Greensboro-cnss73.t3.ans.net (140.222.73.1)  50 ms  60 ms  60 ms
  13  t3-0.Greensboro-cnss75.t3.ans.net (140.222.75.1)  50 ms  60 ms  50 ms
  14  t1-0.enss150.t3.ans.net (140.222.150.1)  120 ms  140 ms  190 ms
  15  rtp3-gw.concert.net (192.101.21.253)  170 ms  180 ms  200 ms
  16  uncch-gw.concert.net (128.109.3.2)  160 ms  130 ms  110 ms
  17  SunSite.unc.edu (152.2.22.81)  130 ms  100 ms  110 ms
```

18.5.4 netstat

netstat shows the status of a network and displays network tables, e.g. to display the statistics concerning packets transferred, errors, etc.:

```
% netstat -i
Name   Mtu    Net/Dest        Address     Ipkts      Ierrs  Opkts      Oerrs  Collis  Queue
lo0    8232   loopback        localhost   4866839    0      4866839    0      0       0
le0    1500   128.146.116.0   nyssa       28898831   598    17568833   1123   332910  0
```

To display the routing tables use the ***-r*** option (***-n*** to prevent host name lookups), e.g.:

```
% netstat -rn
Routing               Table:
  Destination         Gateway               Flags   Ref    Use        Interface
--------------------  --------------------  -----   -----  ------     ---------
127.0.0.1             127.0.0.1             UH      0      4579185    lo0
128.146.116.0         128.146.116.4         U       3      7624       le0
224.0.0.0             128.146.116.4         U       3      0          le0
default               128.146.116.1         UG      0      41960
```

18.5.5 traffic

traffic graphically displays the ethernet traffic, but must be run from a SunView window. This program is available only under SunOS 4.1.X.

18.5.6 xtr

X-windows version of traffic is ***xtr***. For both traffic and xtr you need to run ***/usr/etc/rpc.etherd*** to collect the ethernet statistics to be displayed by these programs.

18.6 Difference between Ethernet and Internet Addresses

18.6.1 Ethernet address

Ethernet addresses are assigned by the manufacturer and are arbitrary. This number is burned into the machine's ID PROM on the CPU board of all Sun workstations. It is composed of 6 1-byte fields for a total of 48 bits. This number is unique and is associated with a particular ethernet device. The values of the ethernet addresses for a subnet are kept in **/etc/ethers**, e.g.:

```
08:00:20:06:50:C2    ivy
08:00:20:06:AD:E4    nyssa
08:00:20:06:A3:4E    gallifrey
00:00:A7:00:1D:4E    bongo
08:00:20:07:9D:64    leela
```

A server requires this information in order to boot a diskless client.

18.6.2 Internet (IP) address

The Internet address is a 32-bit number (4 8-bit fields), that represent the individual machine and subnet of the network. Each 8-bit field is designated by a decimal number between 0 and 255, in the form: a.b.c.d. These addresses are divided into two parts: a network address and a host address. If the first bit of the address is 0, then this is a Class A address, allowing up to 2^{24} - 2 hosts on a network. Class B networks have the first 2 bits as 10, and allow up to 2^{16} - 2 hosts on a network. Class C address have the first 3 bits as 110, and allow up to 2^{8} - 2 hosts on a network. There are also Class D addresses. These are reserved for multicasting, and have their first four bits as 1110. The following Table illustrates the characteristics of the **IP Classes**.

TABLE 18.2 **IP Classes**

Class	First 4 bits	# Network Bits	# Host Bits	Network Number
A	0xxx	7	24	1 to 127
B	10xx	14	16	128 to 191
C	110x	21	8	192 to 223
D	1110	28	Multicast	224 to 239

These correspond to the following IP addresses characteristics.

TABLE 18.3 **IP Addresses**

Class	Maximum # of Networks	Maximum # of Hosts	Address Range	Network Address	Host Address
A	128	16,777,214	1.*.*.* to 127.*.*.*	a	b.c.d
B	16,384	65,534	128.*.*.* to 191.*.*.*	a.b	c.d
C	2,097,152	254	192.*.*.* to 223.*.*.*	a.b.c	d

The **Network** and **Broadcast** addresses are reserved and not used for actual hosts. A Network addresses has a host value of **0**, as in 128.146.116.0. A Broadcast address has all **ones** in the host address, e.g. 128.146.116.255. Earlier versions of SunOS (4.X) use the old style, all **zeroes**, to indicated the broadcast address, e.g. 128.146.116.0. All Sun systems accepts broadcasts from both the 0 and 255 addresses. If you are running SunOS 4.X you should reset the broadcast in **/etc/rc.local** to use the 255 address.

An address starting with **127** in the first field designates a **loopback** address, 127.0.0.1. This refers to the internal interface used by the machine to send a packet to itself. This is usually designated as interface **lo0**.

On a national basis IP addresses are assigned by the Network Information Center (NIC). Locally, these are assigned by the OSU/UTS NIC. A valid IP address and name would look like:

128.146.116.4 nyssa.acs.ohio-state.edu

where	128	→	.edu	NIC
	.146	→	.ohio-state	
and	.116.4	→	nyssa.acs	UTS
where	.116	→	a subnet of .acs.ohio-state.edu	
and	.4	→	nyssa	

IP and hostname pairs are kept in **/etc/hosts**, which might have the contents:

```
127.0.0.1        localhost
128.146.116.4    nyssa nyssa.acs.ohio-state.edu loghost
```

There are 3 class B networks assigned to the Ohio State University:

128.146

140.254

164.107

Which should leave OSU with ample room for expansion for the near future.

CHAPTER 19 Network Administration

19.1 Network Initialization

On startup the **RC** scripts are run to configure the system and the network interface. Through these scripts the system mounts it's local file systems and those it will use over the network. SunOS 5.X uses most of the scripts in **/etc/init.d** when the system enters run level 2. Scripts such as **rootusr**, **inetinit**, **inetsvc**, **sendmail**, **rpc**, **nfs.client**, and **nfs.server** start and stop the network services.

19.1.1 Configuration Files

The configuration files are in the **/etc** directory. Some of these are shown in the following table.

TABLE 19.1 Configuration Files

SunOS 4.1.X	SunOS 5.X	Description
aliases	aliases -> ./mail/aliases	sendmail aliases file
defaultrouter	defaultrouter	IP address of the default router
defaultdomain	defaultdomain	NIS(+) domain name
hostname.xxx	hostname.xxx	host name for the ***xxx*** interface
hosts	hosts -> ./inet/hosts	hosts file
hosts.equiv	hosts.equiv	file of equivalent hosts
inetd.conf	inetd.conf -> ./inet/inetd.conf	configuration file for /usr/sbin/inetd
NA	netconfig	network configuration database
netmasks	netmasks -> ./inet/netmasks	netmask value
NA	nodename	host name for the system
NA	nsswitch.conf	configuration file for the name service switch
remote	remote	remote host description file for tip
resolv.conf	resolv.conf	configuration file for domain name service
sendmail.cf	./mail/sendmail.cf	sendmail configuration file
services	services -> ./inet/services	Internet services file

19.1.2 /etc/bootparams

Diskless clients depend on the server to inform them of their root and swap partitions. The server keeps this information in the **/etc/bootparams** file, e.g.:

```
ivy     root=tardis:/export/root/ivy \
        swap=tardis:/export/swap/ivy
```

19.1.3 File System Mount Options

When mounting a file system you can specify a number of options to indicate the type of file system and to control access to the file system. The following are valid mount options.

4.2	block special device (BSD 4.2 file system type) (SunOS 4.X only)
ufs	block special device (SunOS 5.X)
nfs	NFS file system type
tmp	TMPFS file system type (SunOS 4.X only)
tmpfs	tmpfs file system type (SunOS 5.X only)
swap	swapfs file system type (SunOS 5.X only)
rw|ro	read/write (default), or read-only
bg|fg	if the first attempt fails retry the mount in the background, or foreground (default)
suid|nosuid	allow (default), or disallow, setuid execution
quota|noquota	enable, or disable, quota checking on this file system (applies locally only)
soft	the nfs mount is interruptible
hard	the client will continue trying until the server responds (default)
intr|nointr	allow, or disallow (default) the process to be interrupted on hard mounts
retry=n	retry the mount operation **n** times (defaults to 10000)
rsize=n	set the read buffer to **n** bytes (defaults to SunOS 4.X: 8192; SunOS 5.X: 32768)
wsize=n	set the write buffer to **n** bytes (defaults to SunOS 4.X: 8192; SunOS 5.X: 32768)
timeo=n	set the NFS timeout value to **n** tenths of a second (defaults to 7)
noac	no attribute and name lookup caching.
retrans=n	set the NFS retransmission tries to **n** (defaults to 3)
actimeo=n	set the minimum and maximum cache times for files and directories to **n** seconds (no default)
acregmin=n	retain cached attributes at least n seconds after file is modified (defaults to 3)
acregmax=n	retain cached attributes no more than n seconds after file is modified (defaults to 60).
acdirmin=n	retain cached attributes at least n seconds after a directory is modified (defaults to 30)
acdirmax=n	retain cached attributes no more than n seconds after a directory is modified (defaults to 60).
secure	set DES authentication for NFS transactions
port=n	set the server IP port number to **n** (defaults to NFS_PORT).

These mount options are valid both on the command line for the ***mount*** command and in the mount table: **/etc/fstab** (most Unices) or **/etc/vfstab** (SunOS 5.X).

19.1.4 File System Mounting, SunOS 4.1.X

SunOS 4.1.X specifies the file systems to be mounted in the file **/etc/fstab**. For example a file **server** might have in **/etc/fstab**:

/dev/sd0a	/	4.2 rw,nosuid	1 1
/dev/sd0f	/var	4.2 rw	1 3
/dev/sd0g	/usr	4.2 rw	1 2
/dev/sd0h	/home	4.2 rw	1 4
/dev/sd1b	/export/swap	4.2 rw,nosuid	1 5
/dev/sd1a	/export/root	4.2 rw,nosuid	1 6
swap	/tmp	tmp rw	0 0

A diskless **client** might have in its **/etc/fstab**:

tardis:/export/root/blueagle	/	nfs rw	0 0
tardis:/export/exec/sun4.sunos.4.1.4	/usr	nfs ro	0 0
tardis:/export/share/sunos.4.1.4	/usr/share	nfs ro,soft,bg	0 0
tardis:/usr/local	/usr/local	nfs rw,hard,bg	0 0
tardis:/home/tardis	/home/tardis	nfs rw	0 0
tardis:/var/spool/mail	/var/spool/mail	nfs rw,noac	0 0
tardis:/export/exec/kvm/sun4c.sunos.4.1.4	/usr/kvm	nfs ro	0 0

where the indicated keywords and a few other valid ones have the following meanings:

The last two numbers are the dump interval, in days, and the order in which ***fsck*** checks the disk.

19.1.5 File System Mounting, SunOS 5.X

19.1.5.1 The mount table, /etc/vfstab

SunOS 5.X specifies it's mount table in **/etc/vfstab**, not **/etc/fstab**. The format has been changed a bit also. The fields in this table are:

- **device-to-mount** the block special device for a local file system, or the **server:/dir** designation for a remote one
- **device-to-fsck** the raw special device to be used by fsck
- **mount-point** the mount point for the file system
- **FS-type** file system type, e.g. ufs, nfs, rfs, swapfs, tmpfs, proc
- **fsck-pass** specifies whether the file systems are checked sequentially or in parallel
- **mount-at-boot** specify if the file system should be automatically mounted at boot
- **mount-options** the list of comma-separated options used by mount (no spaces)

A **vfstab** file might look something like the following. Each field must contain an entry, so where no option is called for a hyphen (**-**) is used.

#device	device	mount	FS	fsck	mount	mount
#to mount	to fsck	point	type	pass	at boot	options
/proc	-	/proc	proc	-	no	-
fd	-	/dev/fd	fd	-	no	-
swap	-	/tmp	tmpfs	-	yes	-
/dev/dsk/c0t3d0s0	/dev/rdsk/c0t3d0s0	/	ufs	1	no	-
/dev/dsk/c0t3d0s6	/dev/rdsk/c0t3d0s6	/usr	ufs	2	no	-
/dev/dsk/c0t3d0s5	/dev/rdsk/c0t3d0s5	/opt	ufs	3	yes	-
/dev/dsk/c0t3d0s1	-	-	swap	-	no	-
/acs/nyssa/0/swapfile	-	-	swap	-	no	-
/dev/dsk/c0t6d0s0	-	/cdrom	ufs	-	no	ro
tardis:/home/tardis	-	/home/tardis	nfs	-	yes	hard,intr,bg

The **fsck pass** value specifies whether or not the file system is checked. If this field contains a value of 1 or greater the file system is checked. Non ufs type file systems with a zero fsck pass value are checked. For **ufs** file systems if this value is zero (**0**) or hyphen (**-**) the file system is not checked. For values greater than 1 the files systems are checked in parallel if the preen option (***-o p***) is used with ***fsck*** (this is the default for ufs file systems in **/sbin/rcS**).

The list of mounted file systems is kept in the **/etc/mnttab** file.

19.1.5.2 Default File System Types

When using the ***mount*** command on the command line the default file system type for ***local*** operations is specified in the file **/etc/default/fs**, with the **LOCAL** parameter, and is set to **ufs**, i.e.:

```
LOCAL=ufs
```

For **remote** file systems the default is specified in the file **/etc/dfs/fstypes**, and is set to **nfs**.

When using the ***mount*** command these defaults are assumed unless otherwise specified, e.g. by using the ***-F*** option:

```
# mount -F file-type file-system mount-point
```

The actual mount command used and the available options are determined by the file-type specification. The man pages for mount_ufs, mount_nfs, mount_hsfs, mount_rfs, and mount_tmpfs describe the options available. The actual commands are located in **/usr/lib/fs** under subdirectories named for the file-types.

19.1.6 File System Mounting, IRIX 5.X

IRIX uses **/etc/fstab** to specify its file systems, e.g.:

```
/dev/root / efs rw,raw=/dev/rroot 0 0
mail_server:/var/spool/mail /var/mail nfs hard,bg,intr,noac 0 0
home_server:/home/frank /usr/people/frank nfs hard,bg,intr 0 0
file_server:/usr/local /usr/local nfs ro,hard,bg,intr 0 0
```

Here, for a local device the *raw* partition is specified as one of the mount options.

19.1.7 File System Mounting, Digital UNIX

Digital UNIX uses **/etc/fstab** with a format very similar to SunOS 4.X, except for swap space. This is referenced with an **sw** mount option and for multiple swap areas you can specify the priority, e.g.:

```
/dev/rz0a     /       ufs rw 1 1
/proc         /proc   procfs rw 0 0
/dev/rz0g     /usr    ufs rw 1 2
/dev/rz0b     swap1   ufs sw,pri=0 0 2
/dev/rz1b     swap2   ufs sw,pri=1 0 2
/dev/rz0h     /home   ufs rw 1 3
file_server:/usr/local    /usr/local nfs rw,hard,bg,intr 0 0
```

19.1.8 File System Mounting, Ultrix

Ultrix uses **/etc/fstab** with a format similar to SunOS 4.X, except that fields are separated by a colon (**:**) instead of whitespace, e.g.:

```
/dev/rz2a:/:rw:1:1:ufs::
/dev/rz2g:/usr:rw:1:2:ufs::
/dev/rz6e:/usr/local:rw:1:2:ufs::1:2:ufs::
```

19.2 Host Names and addresses

19.2.1 Static Tables

The table of host names and IP addresses is kept in **/etc/hosts**, in the form:

```
# IP-address        hostname alias
127.0.0.1           localhost loghost
128.146.116.4       nyssa nyssa.acs.ohio-state.edu loghost
128.146.116.1       tardis tardis.acs.ohio-state.edu
```

Generally you will keep the local host name here, its server, or for a server, its diskless clients, and maybe a few other frequently used machines.

Diskless machines require that the server know their ethernet address, kept in the **/etc/ethers**, e.g:

```
# ethernet-address    hostname
00:00:A7:00:11:3D     bongo
```

19.2.2 Dynamic name resolution

Network names and addresses change and new hosts are constantly being added to the network, so it's impossible to keep the static host table up-to-date. To serve this need we have Domain Name Servers (**DNS**) on the network. These are authoritative, or query the authoritative servers to determine IP address, when given host names. The DNS server will be running ***named***, the Internet domain name server daemon (***/usr/[etc,sbin]/in.named***).

For SunOS 4.1.X you can get this automatically through the Network Information Services (**NIS**) when you set the option "**B=-b**" in **/var/yp/Makefile** and re-initialize the NIS maps. Then NIS will automatically query the name server specified in **/etc/resolv.conf** for hosts not found in the NIS maps.

For SunOS 5.X you turn on this service by specifying "**dns**" for the **host** entry in the network switch configuration file, **/etc/nsswitch.conf**. You can also have the system query NIS and or the local **/etc/hosts** file by specifying those, in the desired order, on this entry, e.g.:

```
hosts:          dns nis files
```

Queries to DNS will then be resolved using the information supplied in **/etc/resolv.conf**.

The **/etc/resolv.conf** file contains the IP domain name of the system and a list of name servers to use. For SunOS 5.X you can also specify a search path to use, e.g.:

```
domain acs.ohio-state.edu.
nameserver 128.146.1.7          <---- ns1.net.ohio-state.edu, authoritative for OSU
nameserver 128.146.48.7         <---- ns2.net.ohio-state.edu
search acs.ohio-state.edu magnus.acs.ohio-state.edu eng.ohio-state.edu ohio-state.edu
```

The **domain** will automatically be appended to any host name not having a dot (**.**) in the name. The first **nameserver** listed will be considered primary and queried first. Additional ones will be queried, in order (up to a maximum of 3), if the primary one does not respond to the request.

Many resolvers will accept the **search** field, whereby names to be resolved have these strings appended and then checked for resolution, in the order specified, until one is resolved.

19.2.3 IRIX 5.X

IRIX has a similar **/etc/resolv.conf** file, but it is also used to specify the host resolution order with a line similar to:

```
hostresorder local nis bind
```

in addition to those above.

19.2.4 Ultrix and Digital UNIX

Ultrix and Digital UNIX specify the order to search in **/etc/svc.conf**, with a line similar to:

```
hosts=local,bind
```


19.3 Services

19.3.1 /etc/services

The services available on your system through the network are described in the file **/etc/services**. This database matches services available with their port numbers and protocol, e.g. a few of the many network service entries are:

ftp	21/tcp	# File Transfer Protocol
telnet	23/tcp	# Telnet
smtp	25/tcp	# Simple Mail Transfer Protocol
tftp	69/udp	# Trivial File Transfer Protocol
www	80/tcp	# World Wide Web
ntp	123/tcp	# Network Time Protocol
ntp	123/udp	# Network Time Protocol

19.3.2 /etc/inetd

The internet services daemon, ***inetd***, is started in the **RC** scripts. Inetd responds to requests for services on your machine. It monitors the services specified in **/etc/inetd.conf** and uses the corresponding ports and protocol specified in **/etc/services**. For each service specified in the services database there is a corresponding entry in the inetd.conf file. So for the above example with the **telnet** service there will be a corresponding entry in **inetd.conf** to start the telnet service when a request is received on the network port **23**. This entry will be:

```
telnet stream tcp nowait root /usr/etc/in.telnetd in.telnetd
```

Inetd starts up the required daemon to respond to the request for the specified port. After the connection is made (e.g. at port 23 for telnetd) the transaction is moved to some higher port number. Port numbers 0->1023 are considered "trusted ports" and can only be monitored by root. Each connection is identified by a set of 2-32-bit numbers and 2-16-bit numbers:

Host number of connection's origination

Port number of connection's origination

Host number of connection's target

Port number of connection's target

19.3.3 Remote Procedure Calls

The NFS and NIS protocols, among others, use Remote Procedure Calls (**RPC**) to request and respond to queries for information over the network. The services and the RPC program number they use are listed in the **/etc/rpc** database, in the form:

# rpc-program-server	rpc-program-number	aliases
portmapper	100000	portmap sunrpc <--- SunOS 4.1.X
rpcbind	100000	portmap sunrpc rpcbind<--- SunOS 5.X
nfs	100003	nfsprog

19.4 Network Programs

19.4.1 ifconfig - Configure the Network Interface

Configure the network interfaces with the **ifconfig** command. For each interface you can report or assign the IP, ethernet and broadcast addresses, enable or disable the interface, set the netmask, and protocols for the interface.

Syntax

ifconfig [interface] [address] [options]

Common Options

-a	apply the action to **all** interfaces (SunOS 4.X and 5.X only)
-au	apply the action to all "**up**" interfaces (SunOS 4.X and 5.X only)
-ad	apply the action to all "**down**" interfaces (SunOS 4.X and 5.X only)
up	bring the interface **up**. This happens automatically when you set the first address on the interface.
down	bring the interface **down**. The system will no longer send messages through this interface.
trailers|-trailers	set the flag to use, or disable, "**trailer**" link level encapsulation. "trailers" is no longer used, and it set, is ignored.
arp|-arp	enable, or disable, the use of Address Resolution Protocol (arp) to map between network level and link level address (defaults to arp)
plumb|unplumb	setup and open, or destroy and close, the streams necessary to for TCP/IP to use the interface. After using unplumb the device will not be reported by "ifconfig -a". (SunOS 5.X only).
broadcast *address*	set the address for broadcasting to the local subnet. The default broadcast address is the machine address with the host part of the address set to all 1's, except for SunOS 4.X which defaults to all 0's in the host part of the address.
netmask *mask*	set the mask for how much of the address to use for the network part of the address and how much to use for the subnet (host part) of the address.
ether *address*	set the ethernet address

Examples

ifconfig is usually executed at several points in **RC** scripts, first to bring up each interface, and then again later to reset the netmask and broadcast for each. To report on the network interface do the following, where **le0** is the primary interface name on most Sun workstations:

```
# ifconfig le0
le0: flags=63<UP,BROADCAST,NOTRAILERS,RUNNING>
inet 128.146.116.4  netmask  ffffff00  broadcast  128.146.116.255
ether 8:0:20:fa:1b:2c
```

where the netmask value of ffffff00 is equivalent to 255.255.255.0 and the ethernet address is reported only if you are the superuser.

The file **/etc/netmasks** contains information for non-default netmasks. For SunOS 5.X entries should have the network address use zeroes to fill out the octets, while in SunOS 4.1.X it should not, e.g:

# Network	netmask	
128.146.0.0	255.255.255.0	<-- used by SunOS 5.X
128.146	255.255.255.0	<-- used by SunOS 4.1.X

ifconfig uses this file for its default settings, i.e. when doing the following:

```
# ifconfig le0 netmask +
```

19.4.2 Logical Interfaces (SunOS 5.X)

SunOS 5.X allows the use of multiple logical interfaces for each physical network interface. So a single physical connection can have more than one IP address. The physical interface must first be "plumbed", to make it visible to ***ifconfig*** e.g.:

```
# ifconfig le0 plumb
```

Then the logical interfaces can be configured using the device_name:logical_unit_number format, while retaining the logical unit number 0 for the default physical interface. Valid logical unit numbers are 1 through 255. So to set the first logical interface do:

```
# ifconfig le0:1 IP_address up
```

Each logical interface can have its own network address, netmask, etc. Then create the file **/etc/hostname.le0:1** containing the desired hostname for that logical interface. It should then automatically be configured after each reboot.

19.4.3 route - Network Routing

Normally you would just use the default route to get to one of the network routers (or the server for your subnet) and not have to worry about managing the network routing tables on your system. You can have the system set the default route on startup by placing the IP address of the default router in the file **/etc/defaultrouter**. If you do need to manage the network routing tables then you can run the network routing daemon, ***in.routed***. This will be started for you through the **RC** scripts if no default route exists (i.e. **/etc/defaultrouter** is empty or non-existent).

Syntax

route [options] [add|delete] [host|net] destination [gateway [metric]]

Common Options

-f	flush the routing tables
-n	don’t map the IP addresses to host names
add|delete	add, or delete, a route to the destination
host|net	interpret the **destination** as a host or network, respectively
destination	network destination address
gateway	the network gateway address through which packets are sent
metric	number of hops to **destination**, required with the **add** option. A metric of 0 indicates an interface on the local machine; specify this if all destinations are local. A metric of 1 indicates it’s on the local subnet.

Examples

To add the server as the default router for a workstation, first kill the route daemon on the workstation, if it's running, then **flush** the existing route with:

```
# route -f
```

Lastly, add the **default** route for the interface:

```
# route  add  default  128.146.116.1  1
```

where **default** is the designation used to indicate the **destination address** for all non-local packets, **128.146.116.1** is the address of the **router** for the sub-net, and it is **1 hop** away.

19.4.4 netstat - Show Network Status

Report the status of the network, with its interfaces and sockets, with netstat.

Syntax

netstat [options] [system] [core]

Common Options

-a	show status of all sockets, including server processes
-f address_family	report only statistics related to the address_family, one of:
inet	AF_INET address family
unix	AF_UNIX family
-i	show status of auto-configured interfaces only
-m	show management statistics (or STREAMS statistics, SunOS 5.X only)
-n	don't map the IP addresses to host names
-r	show the routing tables, or statistics (with -s)
-s	show the per-protocol statistics, or routing statistics (with -r)
-g	show multicast groups (SunOS 5.X only)
-M	show multicast groups (IRIX only)
-p	show the address resolution protocol (ARP) statistics (SunOS 5.X only)
-v	verbose (SunOS 5.X only)
system	defaults to the kernel, e.g. /vmunix
core	specify the kernel core file when examining ***savecore*** output, e.g. for SunOS 4.1X: **system=vmunix.0**, **core=vmcore.0**.

Examples

You can check the routes that the machine is actually using with ***netstat***, e.g.:

```
# netstat -rn
Routing tables
Destination      Gateway         Flags  Refcnt  Use         Interface
127.0.0.1        127.0.0.1       UH     1       10007       lo0
default          128.146.116.1   UG     27      55222481    le0
128.146.116.0    128.146.116.4   U      29      138605429   le0
```

The -i option will show the status of the network interfaces, e.g.:

\# netstat -i

Name	Mtu	Net/Dest	Address	Ipkts	Ierrs	Opkts	Oerrs	Collis	Queue
le0	1500	128.146.116.0	server	33168177	864	34382907	19	49045	0
le1	1500	128.146.6.0	server-gw	25310460	1193	20675896	0	62690	0
lo0	1536	loopback	localhost	458882	0	458882	0	0	0

where

mtu	maximum transmission unit
Ipkts	Input packets
Ierrs	Input errors
Opkts	Output Packets
Oerrs	Output errors
Collis	Collisions
Queue	number in the Queue

Often a shortage of buffers can lead to input errors. If the input error rate, Ierrs/Ipkts ≥ 0.00025 (0.025%), you may want to experiment with increasing the receive buffers. Here Ierrs/Ipkts is 0.000026 for le0 and 0.000047 for le1.

Network saturation can be investigated by looking at the collision rate, **Collis**, for a few days. If any of the following are true than one should be concerned.

(Collis + Ierrs + Oerrs)/(Ipkts + Opkts) > 0.02 (2%)   (0.0007, .07%)

Collis/Opkts > 0.02 (2%)   (0.0014, .14%)

Oerrs > 0%

This may indicate that the network is saturated. ***Traffic*** or ***xtr*** can give you an indication of the ethernet usage. If it consistently reports 35% utilization then the ethernet segment is saturated and you should look at ways to distribute the load.

The ***netstat*** command has a few new options for SunOS 5.X to report on the new TCP/IP features, including IP multicasting. **IP multicasting** allows the sender to transmit one packet to be received by one or more, but not necessarily all, hosts on a network. This is useful in multi-party communications when sending the same data to multiple destinations. With logical addressing you can send to a service, rather than a host. The default multicast address is 244.0.0.0. The address 244.0.0.1 is permanently assigned to the group of all hosts and gateways participating in IP multicasting. Host groups are identified by this address and interested hosts listen on this address.

19.5 SunOS 5.X

19.5.1 Host Name

The SunOS 5.X system host name is stored in several files. These files are **/etc/nodename**, **/etc/hostname.xxx, /etc/inet/hosts**, and in the **hosts** files in the directories, **/etc/net/[ticlts,ticots,ticotsord]**. These three refer to the loopback transport providers of the same names. Should you ever change the host name you'll need to change it in all these files. Another way to change the hostname is to touch **/etc/.UNCONFIGURED** and then reboot the machine. On the way back up you will be prompted for the system identification information, including hostname, IP address, NIS/NIS+ type and server, etc.

19.5.2 The rpcbind Server

The server, ***rpcbind***, replaces ***portmap*** as the service providing addresses of server programs to client programs. The ***rpcbind*** server is started by the script **/etc/init.d/rpc**.

19.5.3 Selections

19.5.3.1 Network Services

The network services to use are specified in the file **/etc/nsswitch.conf**. Here you can specify the various name and service databases and the order in which to search the databases.

The **databases** associated with this switch are:

aliases	for sendmail
automount	for the automounter
bootparams	for bootparamd
ethers	for mapping ethernet address to hostnames
group	for getting group names
hosts	for checking host names
netmasks	for ifconfig
networks	for Internet network addresses
passwd	for checking login entries
protocols	for protocol names
publickey	for the rpc public key
rpc	for the rpc service addresses
services	for the network services list

These databases can use the following **sources**:

files	e.g. /etc/hosts, /etc/passwd, etc.
nis	NIS
nisplus	NIS+
dns	Domain Name Service
compat	allows use of **+/-** entries in passwd and group files (for NIS)

A typical **nsswitch.conf** file is:

```
passwd:       compat files nis
group:        files nis
# consult /etc "files" only if nis is down.
hosts:        dns nis [NOTFOUND=return] files
networks:     nis [NOTFOUND=return] files
protocols:    nis [NOTFOUND=return] files
rpc:          nis [NOTFOUND=return] files
```



```
ethers:         nis [NOTFOUND=return] files
netmasks:       nis [NOTFOUND=return] files
bootparams:     nis [NOTFOUND=return] files
publickey:      nis [NOTFOUND=return] files
netgroup:       nis
automount       files nis
aliases:        files nis
# for efficient getservbyname() avoid nis
services:       files nis
sendmailvars:   files
```

19.6 Ultrix and Digital UNIX

The **/etc/svc.conf** file specifies the databases and services, e.g.:

```
# Note: White space allowed only after commas or newlines.
# File Format
# database=service,service
#
# The database can be:
#       aliases
#       group
#       hosts
#       netgroup
#       networks
#       passwd
#       protocols
#       rpc
#       services
# The service can be:
#       local
#       yp
#       bind (hosts ONLY)
aliases=local
group=local
hosts=local,bind
netgroup=local
networks=local
passwd=local
protocols=local
rpc=local
services=local
SECLEVEL=BSD   # for backward compatibility ONLY
```

19.7 Miscellaneous Configuration Files

19.7.1 /etc/syslog.conf

The system log daemon, ***syslogd***, records it's information in log files on the system as determined by its configuration file, **/etc/syslog.conf**. These log files, usually **/var/adm/messages** and/or **/var/log/syslog** or **/var/adm/SYSLOG** (IRIX) or **/var/adm/syserr/syserr.*hostname*** (Ultrix) will, among others, contain messages related to problems with the network. The Solaris 2.X ***syslogd*** requires that the **m4** macro process program be installed (in **/usr/ccs/bin**) to interpret the **syslog.conf** file. ***syslogd*** can be run with the **-d** option to debug problems with **syslog.conf** entries.

19.7.2 /etc/hosts.equiv

We sometimes want to allow other users, on systems we trust, to login over the network without supplying a password. In other words, we presume that authentication on their system is equivalent to our own. We can accomplish this by putting their system hostname in the file **/etc/hosts.equiv**. When an rlogin or rsh request comes in from one of these hosts there is no prompt for a password. This applies for all users except the root user. SunOS 4.X is delivered with "**+**" in this file, meaning ***ALL*** hosts are trusted. If you are not going to use this file remove it. On a per user basis the file **~/.rhosts** is equivalent to the above file; this also includes the root user.

19.7.3 Terminals

The terminal configuration files are covered in the chapters on **Adding Hardware**, and the **Service Access Facility**.

19.7.4 Mail

We will look at the mail related configuration files, **sendmail.cf** and **aliases** in the **Mail** chapter.

CHAPTER 20 Distributed File System Administration

20.1 Distributed File Systems

There are two distributed file systems used by SunOS, the Network File System (**NFS**) and the Remote File Sharing (**RFS**) system. These both operate over the network and allow you to share files. The latter also allows you to share devices. These are both available in SunOS 4.1.X and 5.2, though RFS did not survive beyond 5.2. The commands for using these programs are different in the two releases. A machine can simultaneously run both NFS and RFS.

20.1.1 The Network File System

The Network File System was developed by Sun Microsystems and is licensed to many vendors. It allows the sharing of file systems and directories, and provides a common login environment regardless of the network machine on which you login. It's a service that is designed to be machine independent and transparent to the user.

NFS uses remote procedure calls (RPC) through the external data representation protocol (XDR) to communicate between machines. The user doesn't have to know any of the details. When things are working properly local and remote file systems will appear as one big local file system to the user.

The major functions of NFS are mount/export directories from/to other computers, on/off your local network, so that they can be accessed as if they were local. An NFS client can mount files systems from more than one NFS server. These mounts are done through the ethernet. The NFS server does not maintain state information about its clients open files; this must be done by the client. The server program is small and efficient, while the client program has to do most of the work.

NFS supports diskless workstation booting and automounting, and allows you to mount NFS directories on top of other NFS directories.

20.1.2 The Remote File Sharing System

The Remote File Sharing system, was developed by AT&T to allow UNIX workstations to share files over a network. It allows workstations to act as clients of servers.

RFS provides access to files and directories without the user having to know where the resource is located. A nameserver is used to register resource names, so the client machine doesn't need to know where the resources are. Resources are moved simply by changing entries in the nameserver registry.

RFS allows users to mount special directories so that they can share devices (e.g. tape drives) residing on other machines.

RFS is a stateful protocol; the server maintains state information of local resources. The server knows what each client is doing to it's files at all times. The server can detect client crashes, so cache consistency is guaranteed.

RFS can NOT be used to boot diskless clients.

RFS does NOT support symbolic linking or automounting.

RFS does NOT support mounting of a directory on top of an existing RFS directory.

20.2 NFS Protocol

The NFS protocol uses RPCs to communicate between client and server. The client issues an RPC request for information from the server which replies with the result. If requests go to a machine with different byte ordering XDR can translate between them. There are 16 different RPCs used by NFS version 2 to request and regulate file access.

The RPCs run on top of the UDP protocol. UDP is faster than TCP, but doesn't provide any error checking. NFS relies on the built-in retry logic of the RPCs to make sure that requests and replies arrive at their destinations. The client can specify block sizes, number of retry attempts, and time to wait values when it mounts the servers files, with defaults of 8k blocks (read: **rsize**, write: **wsize**), 5 retries (**retrans**), and a 1 second timeout (**timeo**). If the client doesn't receive an acknowledgment within the timeout period it sends the request again. To prevent overloading the server it then doubles the time-to-wait period. The client continues the cycle until the server responds or the retry limit is reached. If the latter occurs you get the familiar "*nfs server not responding*" error message. Since the NFS protocol is stateless the client receiving this error has no information to decide if the problem is with the network or with the server. Processes trying to access server files, e.g. df, will happily wait until the server responds, as it is blocked until it receives a reply. If the server crashed the client program will pick up where it left off after the server comes back on line. You can use "**soft**" mounts to give you the ability to break out from stalled RPC send/receive requests. If you really want to ensure that write requests are completed, though, you should use "**hard**" mounts, and also specify "**intr**" if you want to be able to abort the command.

Before a client issues an RPC request to the server it checks to see if the desired data is already cached from an earlier request. If the data is newer than the cache attribute timeout value (**actimeo**, with a default of 30 seconds) than the data is used, otherwise it sends a request to the server to compare the modification time of it's cached file with that of the server's file. If the server's file is newer a request to resend the data is issued.

NFS version 3, used by IRIX 5.3+ and SunOS 5.5+ has some significant enhancements over earlier versions. NFS can now run on top of the TCP protocol. Additionally it now supports safe asynchronous writes, finer access control, and larger file transfer sizes, with less overhead. Since NFS is stateless you want to make sure that the server has really performed the write request to a stable storage area before acknowledging it to the client. Version 3 allows unsafe, asynchronous, writes to be committed to stable storage reliably.

The maximum transfer size has been increased from 8 kB to 4 GB, where the machines negotiate the transfer size, up to 64 KB, the maximum allowed for both UDP and TCP. The protocol, either TCP or UDP, is also negotiated between the machines, defaulting to TCP if both ends support it. The new protocol now allows 64-bit file offsets, up from the former 32-bit limit, supporting arbitrarily large file sizes. The new version is more efficient, e.g. it returns the file attributes after each call, eliminating the need to issue a separate request for this information.

Solaris 2.5 and IRIX 5.3+ NFS implementations support both version 3 and version 2 of the protocols, so that they can reliably communicate with clients and servers supporting either, with full backwards compatibility. Both NFS versions use port 2049 and should have such and entry for both udp and tcp in /etc/services. The 22 RPC requests used be NFS version 3 are listed below.

TABLE 20.1 NFS RPC calls

NFS version 3	NFS version 2	Description
void	null	Does nothing, except make sure the connection is up
GETATTR	getattr	get file, or directory, attributes, e.g. file type, access times & permissions
SETATTR	setattr	set file, or directory, attributes
LOOKUP	lookup	lookup file name in a directory
ACCESS		check access permissions for a user
READLINK	readlink	read the data from a symbolic link
READ	read	read from a file
WRITE	write	write to a file
CREATE	create	create a file or symbolic link
MKDIR	mkdir	create a directory
SYMLINK	symlink	create a symbolic link
MKNOD		create a special device node
REMOVE	remove	remove a file (delete the directory entry)
RMDIR	rmdir	remove a directory (delete the subdirectory entry from a directory)
RENAME	rename	rename a file or directory
LINK	link	create a link to an object
READDIR	readdir	read from a directory
READDIRPLUS		extended read from a directory
FSSTAT	statfs	get dynamic file system state information
FSINFO		get static file system state information
PATHCONF		retrieve POSIX information for the filesystem
COMMIT		commit the cached data on the server to stable storage (force a flush of data previously written to the server)

20.3 SunOS 4.1.X

20.3.1 NFS

20.3.1.1 Server

For an NFS server the important command is ***/usr/etc/exportfs.*** This command must be run to enable clients to mount the file systems. You can specify file systems to be exported on the command line, or you can use the ***-a*** (all) option to the default export file, **/etc/exports**.

/etc/exports contains the list of system directories to export, who has access to them, and the nature of the access (e.g. rw or ro, root, etc.). A file server might have the following exports file to service diskless and dataless clients of multiple hardware architectures.

```
/home -access=nyssa:blueagle:leela
/usr -root=blueagle,access=blueagle
/usr/local -root=nyssa,access=nyssa:blueagle
/usr/sun3/local -access=leela
/var/spool/mail -access=nyssa:blueagle:leela
/export/share -access=blueagle:leela:nyssa
/export/exec/sun3x.sunos.4.1.1 -access=leela
/export/exec/kvm/sun4c.sunos.4.1.4 -access=blueagle
/export/exec/kvm/sun3x -access=leela
/export/root/leela -root=leela,access=leela
/export/root/swap/leela -root=leela,access=leela
/export/root/blueagle -root=blueagle,access=blueagle
/export/swap/blueagle -root=blueable,access=blueagle
```

If you don't specify restrictions it defaults to allow read/write access to all, e.g. if **exports** contains:

```
/
```

your **root** directory is accessible to everyone on the net.

After you edit **/etc/exports** to make the directories mountable by other systems you need to run:

```
# /usr/etc/exportfs -a
```

The server needs to run the NFS mount daemon, ***rpc.mountd***, and several NFS service daemons, ***nfsd*** (typically 8 for a low use server). These two daemons handle NFS mount requests and client requests, respectively. NFS also requires that the block IO daemons, ***biod***, be running (normally 4 are started) to buffer read-ahead and write-behind requests. These are used for all client requests, both local and through NFS. The block IO daemons are always started by **rc.local**; the other NFS daemons started for the server during boot by **rc.local** only if **/etc/exports** exists. The relevant lines in **rc.local** are:

```
if [ -f /usr/etc/biod ]; then
        biod 4;         (echo -n ' biod')       >/dev/console
fi
if [ -f /etc/exports ]; then
        > /etc/xtab
        exportfs -a                             >/dev/console
        nfsd 8 &        (echo -n ' nfsd')       >/dev/console
        rpc.mountd -n                           >/dev/console
fi
```


If you create an **exports** file later you will need to start these by hand before your system can become a server.

The file **/etc/xtab** contains the list of files actually exported via ***exportfs***. If you execute exportfs without any options it will display this list.

You can use the ***showmount*** command to see who is mounting your file systems, though it's not a very accurate representation of the current state.

20.3.1.2 Client

The client normally mounts all file systems listed in **/etc/fstab** during the boot process. Those of type **nfs** are mounted when the "***mount -at nfs***" command is issued in **/etc/rc.single**, e.g. an nfs entry in fstab might be:

```
tardis:/home /home nfs rw 0 0
```

You can also mount file systems from the command line, e.g.:

```
# mount -t nfs tardis:/home /home
```

This will issue an NFS request to the server, tardis, to mount the file system, /home, on the local directory, /home.

The client needs to be running the block IO daemon, ***biod***, to buffer NFS requests (normally 4 are started).

Another way to mount file systems is to use the automounter. The automount daemon, ***automount***, will automatically mount the desired file system whenever a file/directory in that file system is accessed. It intercepts any requests for access to the file system and then uses the information in a NIS map or local file to decide how and where to mount the file system. If no access is made after a few minutes the file system is unmounted again.

20.3.2 RFS

20.3.2.1 Security

Security for RFS is provided through:

- machine passwords
- read-only access to resources
- can restrict machines allowed to access resources
- can force users off at any time
- file access based on user and group id mapping

20.3.2.2 Components of RFS

The RFS Domain consists of:

- primary name server
- secondary name servers
- RFS resource servers
- RFS clients

The ***RFS*** **name server** maintains information about the RFS domain, available resources, and passwords for use in the domain.

The ***RFS*** **file server** can advertise resources from itself or those that it has **NFS** mounted.

The ***RFS*** **client** mounts resources advertised from an **RFS file server**.

20.3.2.3 Installation and Initialization

The ***RFS*** software must be loaded from the Installation tape or CDROM, either at installation or later with the ***add_services*** utility.

The following ***kernel*** options are required in the configuration file in order to use RFS:

```
options          RFS
options          VFSSTATS
pseudo device    tim64
pseudo device    tirw64
pseudo device    tcptli32
pseudo device    sp
pseudo device    clone
pseudo device    snit
pseudo device    pf
pseudo device    nbuf
```

You need to choose a unique **domain name**, having a maximum of 14 characters (SysV restriction). Then create the file, **/usr/nserve/rfmaster**, on the primary name server. This file should contain the names and IP addresses of the primary and secondary RFS name servers in the format:

```
RFS_domain             server_type      RFS_domain.hostname
RFS_domain.hostname    A                hex_IP_address
```

where ***server_type*** should be replaced by either ***P*** (primary) or ***S*** (secondary), and separate each field of a line by a space or tab.

To convert an IP address to hex use the following command for hosts listed in /etc/hosts:

```
% hostrfs tardis
\x00021450809274010000000000000000
```

The database file, **/usr/nserve/rfmaster**, should be readable by everyone, but writable only by root (mode 644).

Initialize RFS on the name **server** with:

```
# dorfs init RFS_domain tcp [port num]
```

where **init** initializes the services for the domain and **tcp** specifies the network protocol type.

The RFS daemons are started with:

```
# dorfs start
```

This will start the daemons: ***listener***, ***rfudaemon***, ***rfs:server***, ***rfs:recovery***, ***rfs:rfdaemon***

RFS startup can be done at boot time by uncommenting the associated lines in **/etc/rc**.

Initialize RFS on the **client** by first copying the **/usr/nserve/rfmaster** file from the server, again setting the permissions to 644. After the primary name server is started initialize RFS with:

```
# dorfs init RFS_domain tcp [port num]
```


and start the RFS daemons with:

dorfs start

20.3.2.4 Advertising and Monitoring the Resources

The ***server*** now needs to advertise resources it wishes to export with the ***adv*** command.

An example whereby you want to allow the tape devices of the server to be available to the clients, you could create the directory **/dev/rdev** on the server. The files in /dev/rdev would be links to the tape devices, e.g.:

```
# ln /dev/rst8 /dev/rdev/rst8
# ln /dev/rmt8 /dev/rdev/rmt8
```

Then you advertise the resources in /dev/rdev to the client:

```
# adv -r -d "tardis devices" tardisdevs /dev/rdev nyssa
```

Here I have advertised the **tardis devices** directory, **/dev/rdev**, read-only, which will be known only to the client nyssa, with the name **tardisdevs**. If you want to advertise resources automatically create a file **/etc/rstab** on the RFS server. It should be a shell script of ***adv*** commands and should be executable by all, mode 755.

To **display** the advertised properties on the server use the ***adv*** command without options, e.g.:

```
# adv
    tardisdevs     /dev/rdev  "tardis devices"  read-only nyssa
```

To ***unadvertise*** a resource on the server use the ***unadv*** command, e.g.:

```
# unadv resource
```

To find out what resources are available on the client use the ***nsquery*** command, e.g.:

```
# nsquery
    RESOURCE       ACCESS          SERVER          DESCRIPTION
    tardisdevs     read-only       rfs_acs.tardis  tardis devices
```

Mount the resource on the client with the ***mount*** command, specifying the RFS device, e.g.:

```
# mount -r -d tardisdev /mnt
```

where "***-r***" specifies read-only, and "***-d***" is followed by the name of the RFS resource.

To unmount the directory use the ***umount*** command, e.g.:

```
# umount -d tardisdev
```

To mount an RFS resource automatically at boot time add an entry to **/etc/fstab** similar to:

```
tardisdev          /mnt          rfs ro 0 0
```

To unadvertise and forcibly unmount a resource from all clients type at the **server** use ***fumount***, e.g.:

```
# fumount [-w seconds] resource
```

The superuser can monitor client use of server resources with the ***rmntstat*** command:

```
# rmntstat
    RESOURCE       PATH          HOSTNAMES
    tardisdevs     /dev/rdev     rfs_acs.nyssa
```

The ***fuser*** command can be used on the client to see who is using the resource, e.g.:

```
# fuser tardisdevs
```


20.4 SunOS 5.X

SunOS 5.X uses a common set of commands and files to administer both the Network File System (**NFS**) and the Remote File Sharing (**RFS**) system. This combination is known as the Distributed File System (**DFS**). These common set of commands for DFS replace the individual commands for NFS and RFS of SunOS 4.X. Through DFS you can share files, directories, and devices over the network.

The operative word with **DFS** is **sharing**. With DFS we **share** file systems, rather than **export** them. /etc/exports and exportfs are gone. In place of these files we have the control files in the **/etc/dfs** directory and the ***share*** and ***shareall*** commands. The files in **/etc/dfs** are:

- **dfstab** containing commands for sharing resources across the network
- **fstypes** which registers the DFS packages on the system, i.e. nfs and rfs
- **sharetab** containing a table of local resources being shared

20.4.1 /etc/dfs/dfstab

This server file contains a series of share commands for sharing the resources. Each line consists of a ***share*** command specifying the resource to be shared, the file system type, a description of the resource and options specifying client access to the resource. The commands in this file are automatically executed when entering run level 3. An example of a **dfstab** would be:

```
#       place share(1M) commands here for automatic execution
#       on entering init state 3.
#
#       share [-F fstype] [ -o options] [-d "<text>"] <pathname> [resource]
#       .e.g,
#       share  -F nfs  -o rw=engineering  -d "home dirs"  /export/home2
share -F nfs -o ro=ace:tardis:gallifrey -d "nyssa cdrom" /cdrom
```

20.4.2 /etc/dfs/fstypes

For each distributed file system type installed on the system there is a line in **fstypes** that begins with the file system type name followed by a description of the package, e.g. for nfs:

```
nfs nfs utilities: version 11.4.2
```

20.4.3 /etc/dfs/sharetab

This contains a table of shared local resources and is created by the ***share*** command. There is a line for each resource shared containing the pathname of the resource, the resource being shared, the file system type, specific options specifying how the resource is being shared, and a description of the resource, e.g.:

```
/cdrom    -    nfs    ro=ace:tardis:gallifrey    nyssa cdrom
```


20.4.4 Daemons

The block I/O daemons, ***biod***, have been replaced by kernel threads. NFS write requests are queued and assigned to a kernel thread on a per-mount basis. The kernel thread performs the asynchronous write request. The kernel threads are created on demand, with up to 8 per mount point allowed. When there are no outstanding write requests no threads exists for that mount point.

8 nfs daemons, ***nfsd***, and the mount daemon, ***mountd***, are started when you initiate run level 3 by the **/etc/rc3.d/S15nfs.server** script. These daemons are physically located in **/usr/lib/nfs**. If the **/etc/dfs/dfstab** file is empty when entering run level 3 the ***mountd*** and ***nfsd*** daemons are not started. To start these daemons later you can place an entry in the **dfstab** file and execute *"**init 1**"*, and then return to run level 3, or you can execute the **nfs.server** script mentioned above.

NFS clients require the ***statd*** and ***lockd*** daemons, also located in /usr/lib/nfs.

20.4.5 The Automount File System, autofs

Autofs mounts file systems when they are accessed and unmounts them after a specified period of inactivity. It uses the automount daemon, ***/usr/lib/autofs/automountd***, to control the mounting of file systems. This daemon is started at run-level 2 by the **/etc/init.d/autofs** script which also mounts the file systems with the ***/usr/sbin/automount*** command.

The automount system doesn't use /etc/vfstab to specify file systems. It uses the maps specified in the **/etc/auto_master** file and in the NIS(+) system. (This file is known as **auto.master** for NIS.) The **auto_master** file has entries of the form:

```
#mount-point     map-name        [ mount-options ]
+auto_master
/net             -hosts          -nosuid
/home            auto_home
```

In this file **+auto_master** refers to an NIS(+) master map. If one exists insert those entries as if they were part of this file. The remaining entries specify the directory to automount the file and the automount map associated with it. The **-hosts** map entry specifies all the NFS exported file systems in the NIS(+) **hosts** database. These will be mounted on **/net**, e.g. the file systems for a host, **tardis**, will be mounted on /net/tardis. For the last entry in auto_master there is a corresponding **/etc/auto_home** file with contents:

```
+auto_home
```

indicating that the **auto_home** NIS(+) map should be used. (This file is known as **auto.home** for NIS.) When the location of this file is not specified by the complete path name it is follows the convention determined by the ***automount*** entry in /etc/nsswitch.conf, e.g.:

```
automount:       files nis
```

which specifies that the files in ***/etc*** should be checked first, followed by the ***NIS*** maps. The **auto_home** table maps login names with directories, and is managed through the NIS(+) system.

The automount maps can be direct or indirect. The direct maps specifies a mount point on the client for a specific directory on the server. An indirect map refers to a table of automount points. The indirect map is the more common way of using the automounter.

In a direct map you can specify more than one server from which to access read-only file systems, e.g.:

```
/usr/man -ro        server1:/usr/man server2:/usr/man server3:/usr/man
```

The system will mount the nearest available server, with those on the same subnet being given preference.

You can use certain variables in these maps by prefacing a dollar sign to the variable name. The variable names recognized by the automounter are:

Variable Name	**Variable Meaning**	**Example**
ARCH	hardware architecture	sun4c
CPU	processor type	sparc
HOST	hostname	nyssa
OSNAME	operating system name	SunOS
OSREL	operating system release number	5.5
OSVERS	operating system version	FCS1.0

When you make an addition or deletion to a direct map you need to run the automount command to have the change take effect. Modifications to existing entries don't require you to do this.

To modify the master NIS+ maps use the ***nistbladm*** command then run the automount command to have the changes take effect. We'll look at the nistbladm command in the chapter on NIS+ later in the course.

You can access non-NFS file systems through the automounter, including removable media and cachefs file systems. For these you need to specify the file system type and the device file or cache to use. To mount a cachefs file system put an entry similar to the following in master map:

```
/home      auto_home      -fstype=cachefs, cache=/local/cache
```

You can not automount a file system on top of another automounted file system.

By default when you boot your system it will try to automount the home directories known to the NIS(+) server. If you don't want to run the automounter move the file, **/etc/rc2.d/S74autofs**, to a name not beginning with "**S**", e.g. old.S74autofs.

20.4.6 Utilities

20.4.6.1 Mounting and Unmounting Resources

To mount and unmount resources we have the ***mount***, ***mountall***, ***umount***, and ***umountall*** commands. With the ***mount*** command you can specify the file system type (***-F***) and options to be used (***-o***) for the mount. With the ***mountall*** command you can designate a file system type (***-F***) and either local (***-l***) or remote (***-r***) file systems. Use ***umount*** to unmount a file system. There are similar options for the ***umountall*** command with the addition of a kill (***-k***) option to send a kill signal to all processes with open files on the indicated systems.

Your local systems are mounted when going to run level 2 by **/etc/rc2.d/S01MOUNTFSYS**. NFS clients resources are mounted at run level 2 also by the **S73nfs.client** script.

20.4.6.2 Sharing and Unsharing Resources

To share resources use the ***share*** and ***shareall*** commands and unshare them with the ***unshare*** and ***unshareall*** commands. You can specify file system types (***-F***) a description of the resource (***-d***) and various options to control client access (***-o***, with ro/rw, or rw=client[:client2]). With the ***unshare(all)*** commands you can only specify a file system type, so you can unshare all nfs file types with the command:

```
# unshareall -F nfs
```

The ***shareall*** command shares all resources specified in the **/etc/dfs/dfstab** file, or a named file.

When invoked with no arguments the ***share*** command displays the resources currently shared, e.g.:

```
# share
    -     /cdrom    ro=ace:tardis:gallifrey     "nyssa cdrom"
```

20.4.6.3 Displaying Available Resources

To display mounted resources information use the ***dfmounts*** command. This command shows the local resources that are shared along with the clients that have the resource mounted.

```
# dfmounts
    RESOURCE        SERVER      PATH        CLIENTS
    -               nyssa       /cdrom      gallifrey
```

To display available resources from remote or local systems use the ***dfshares*** command, e.g.:

```
# dfshares
    RESOURCE        SERVER      ACCESS      TRANSPORT
      nyssa:/cdrom  nyssa       -           -
```

20.5 DFS Command Summary

The following table summarizes the commands used to administer Distributed File Systems in SunOS.

TABLE 20.2 DFS Command Summary

SunOS 4.X	SunOS 5.X	Description
mount -a	mountall	Mount all file systems
umount -a	umountall	Unmount all file systems
exportfs	share	Share file systems
exportfs -u	unshare	Unshare file systems
exportfs -a	shareall	Share all file systems
showmount -d	dfmounts	Show mounted file systems
showmount -e	dfshares	Show shared file systems

20.6 IRIX 5.X, Ultrix and Digital UNIX

IRIX 5.X, Ultrix, and Digital UNIX all use **/etc/exports** to specify the files available for sharing over the network. IRIX, similar to SunOS 4.X, requires you to run ***/usr/etc/exportfs*** to actually export those files. Ultrix and Digital UNIX do not use the ***exportfs*** command.

20.7 NFS statistics

20.7.1 netstat

netstat can be used to show the per-protocol statistics with the ***-s*** options, e.g. on SunOS 4.1.X:

```
# netstat -s
    udp:
            0 incomplete headers
            0 bad data length fields
            0 bad checksums
            0 socket overflows
    tcp:
            21392 packets sent
                    13925 data packets (1565473 bytes)
                    23 data packets (901 bytes) retransmitted
            …
```

If **udp** reports socket overflows then increase the number of ***nfsd***s, as user processes aren't draining the sockets quickly enough. Typically a SunOS 4.X server starts, by default, 8 NFS daemons. On some systems it may be more appropriate to have 12 → 20 ***nfsd***s.

20.7.2 nfsstat

The ***nfsstat*** command can be used to display statistics related to **NFS** activity. This command is useful when trying to debug **NFS** and **RPC** problems. ***nfsstat*** also has options to show both client and server information.

20.7.2.1 Server

On the server use ***nfsstat -ns*** (***-n*** ⇒ NFS information; ***-s*** ⇒ server) to examine the statistics, e.g.:

```
% nfsstat -ns
    Server nfs:
    calls       badcalls
    69350       0
    null        getattr     setattr     root        lookup      readlink    read
    0  0%       54682 78%   266  0%     0  0%       7138 10%    748  1%     3352  4%
    wrcache     write       create      remove      rename      link        symlink
    0  0%       1465  2%    421  0%     247  0%     84  0%      5  0%       0  0%
    mkdir       rmdir       readdir     fsstat
    3  0%       0  0%       902  1%     37  0%
```


Of these RPC calls, root and wrcache are not currently used by NFS.

If **readlink** is high (>10%) replace symbolic links with mount points wherever possible on the client to improve NFS performance.

If **getattr** is > 50% check for non-default attribute caching.

20.7.2.2 Client

To display client statistics, on the client execute ***nfsstat -rc*** (***-r*** ⇒ RPC information; ***-c*** ⇒ client), e.g.:

```
% nfsstat -rc
    Client rpc:
    calls     badcalls   retrans   badxid   timeout   wait   newcred   timers
    307703    54         31        24       82        0      0         2037
```

where

calls	total number of RPC calls received
badcalls	timeouts resulting from RPC error
retrans	retransmission count
badxid	duplicate responses from server
timeout	# of RPC calls timed out
wait	calls that had to wait on a busy CLIENT handle
newcred	refreshes of authentication information

If **retrans > 5%** of total calls, then requests are not reaching the server.

If **badxid ~ timeout**, then most requests are reaching the server, and the server is the bottleneck.

If **badcalls ~ timeout**, then soft-mounted filesystems are failing.

You can check the NFS mounted file system states for the **client** with ***nfsstat -m*** (***-m*** ⇒ NFS stats for each mounted file system), e.g.:

```
% nfsstat -m
/usr/local from server:/usr/local
 Flags:   vers=2,proto=udp,auth=unix,hard,intr,dynamic,rsize=8192,wsize=8192,retrans=5
 Lookups: srtt=7 (17ms), dev=4 (20ms), cur=2 (40ms)
 Reads:   srtt=7 (17ms), dev=4 (20ms), cur=2 (40ms)
 Writes:  srtt=31 (77ms), dev=3 (15ms), cur=5 (100ms)
 All:     srtt=7 (17ms), dev=4 (20ms), cur=2 (40ms)

/opt/ftp from susan:/opt/ftp
 Flags:   vers=3,proto=tcp,auth=unix,hard,intr,link,symlink,acl,rsize=32768,wsize=32768,retrans=5
 All:     srtt=0 (0ms), dev=0 (0ms), cur=0 (0ms)
```

where

srtt	smoothed round-trip time
dev	estimated deviation
cur	current backed-off timeout value

If **srtt > 50 ms**, then the mount point is slow, either at the server or because of network problems.

If **Lookups: cur > 80 ms**, or **Reads: cur > 150 ms**, or **Writes: cur > 250 ms**, it's taking tool long to process the requests on the server side (either server or network).

If you frequently see the "***NFS server not responding***" error message it maybe time to increase the **timeo** setting on the mount in **/etc/fstab** or **/etc/vfstab** (SunOS 5.X).

To correct for slow servers, (i.e. **badxid ~ timeout**) increase the RPC timeout (**timeo** option of the ***mount*** command). To correct for **badcalls ~ timeout**, increase **retrans** and possibly **timeo** option values. It is recommended that soft mounts not be used for writable filesystems or for executable files. Soft is recommended for only non-executable file systems mounted read-only. For other filesystems '**hard,intr,bg**' is recommended. If the network is the bottleneck (i.e. **badxid ~ 0**) it may be necessary to decrease the NFS buffer sizes: **rsize** and **wsize**, on the **client** from 8kB to 2kB. Network bottlenecks can also have other causes, e.g. the interconnection device (gateway, router, bridge) may be limiting.

CHAPTER 21

Network Information Services (NIS and NIS+)

21.1 What is it and what does it do for you?

The Network Information Service (**NIS**) allows networked machines to have a common interface regardless of the workstation that you log into. This service was formerly known as the Yellow Pages, or YP. With NIS you have the same passwd and group files (same uid and gid) and can be placed into the same home directory on each of your machines.

These services are considerably expanded under SunOS 5.X as Network Information Services Plus (**NIS+**). The Solaris 2 CDROM provides an NIS+ version that will run under SunOS 4.1.X in case you want to mix and match servers.

21.2 NIS

21.2.1 Initialization

Install the NIS software during installation with ***suninstall***, or later with /usr/etc/install/***add_services***.

Initialize the NIS domain by running ***/usr/etc/ypserv***, on the server and on its clients running ***/usr/etc/ypbind***. This is done in **/etc/rc.local**. The NIS servers can also be NIS clients. You can have slave servers for redundancy.

You need to specify a **domainname**, e.g. department, etc. in **/etc/rc.local**. This is completely separate from the IP domain name. Normally the NIS domainname is put in the file **/etc/defaultdomain** for use during startup. If this file does not exist or has the contents "noname", it is assumed that you are not using NIS. The domainname can be set or displayed with the ***domainname*** command.

You originally set up the NIS databases on the server with the command ***/usr/etc/yp/ypinit -m/s*** (master/slave). In the simple case the server is the master for all maps in the database. All databases are built from scratch with ***ypinit***. To update changed databases, e.g. after installing a new user:

```
# cd /var/yp;  make
```

This will push the new databases to all the machines in the NIS domain.

If you have more than one NIS server you may wish to bind a particular machine with a specific server. This can be done with the ***ypset*** command in conjunction with using the ***-ypset*** option to ***ypbind***.

To display your current NIS server use the ***ypwhich*** command.

To display contents of the NIS tables you can use the ***ypcat*** and ***ypmatch*** commands. ***ypcat*** lists the specified table. ***ypmatch*** matches a keyword with the specified table, e.g.:

```
% ypmatch frank passwd
    frank:jkl/fdasjklKY:101:10:Frank G Fiamingo:/home/tardis/frank:/usr/bin/tcsh
```

21.2.2 Databases controlled by NIS

The information in the NIS maps is in a database format using the **ndbm** library. Each map has 2 files: .pag, and .dir. These are contained in a subdirectory of **/var/yp** named after your NIS "domain". The databases are:

Name	**Service**
aliases	mail aliases and addresses
bootparams	boot and NFS mount information for diskless clients
ethers	hostname and ethernet addresses
group	group names and gid's
hosts	hostname and internet addresses
netgroup	netgroup membership list
netid	map of local userID/groupID/group access-list and hosts for DES
netmasks	network number and netmask
networks	network number and internet name
passwd	username and password information
protocols	internet protocol names and numbers
publickey	public and secret keys for secure NFS
rpc	RPC program name and number
services	internet service name, port number, and protocol

To tell the SunOS 4.1.X system to use the NIS database for passwd and group files put entries such as:

```
+::0:0:::
```

as the last entry in the **/etc/passwd** file of the NIS clients, i.e. all NIS password entries are valid on this host. Other examples of limitations and exclusions are, for **/etc/passwd**:

+frank:	- frank is a valid user, use his entry from the NIS database.
+frank::::/home/new/frank:	- frank is a valid user, all entries are as in the NIS database, except his login directory.
+@group:*:0:0:::/bin/true	- the group "group" can't login, but users in this group can refer to their home directories.
-@group::0:0:::	- exclude this group from entry.

and for **/etc/group**:

+:	- all entries in the NIS group database are valid here.
+group:	- the NIS group "group" is valid.
+project:::frank,bob	- only the member frank and bob of group "project" are valid.

SunOS 5.X clients will use the NIS database if **nis** and **compat** (for NIS +/- entry compatibility) are specified for the **passwd** entry in **/etc/nsswitch.conf**, e.g.:

```
passwd:          compat files nis
```

To use the default NIS passwd table there is no need to add additional entries to **/etc/passwd** on the SunOS 5.X client.

21.3 NIS+

SunOS 5.X provides an enhanced version of NIS, **NIS+**, that is upwardly compatible with NIS. The new service provides for a hierarchical name space, similar to that used by the Internet. This allows for a distributed authority mechanism. User's can be given access to an entire database, or just particular entries within a database. Administrators can be restricted to changing files only within their domain.

NIS+ propagates only changes in the maps, not the entire map. This allows for much faster updates. Entries are changeable anywhere on the NIS+ network. You don't have to be on the server to change the maps.

The authorization model for NIS+ is similar to that for the UNIX file system. Each item in the namespace has an access rights list associated with it. These rights grant access to owner of the item, group owner of the item, and all others.

21.3.1 Domains

The NIS+ domain is composed of a directory object and all of its children. The NIS+ namespace is made up of all the domains below the root directory. Each name is composed of a series of characters separated by a (**.**). These character sequences are known as **labels**. The label furthest to the right is closest to the root of the namespace. The (**.**) name is reserved to indicate the global root namespace; the root directory name always ends with a (**.**). NIS+ names are not case sensitive.

The **root server** is the server for the root (.) domain. There is only one root server for a domain.

A **master server** serves a domain. A master server is a client of the server directly above it in the hierarchy.

A **replica server** is a copy of the master server, formerly known as a ***slave*** server. This provides redundancy for the service.

21.3.2 Objects

There are three types of **objects**:

- **directory objects** which form the framework of the namespace
- **table objects** which store the information
- **group objects** which are used for security

The **directory objects** are at the top of the namespace. Directory objects contain the names, addresses, and authentication information for systems within the domain. Objects within the database are stored as children of the directory object. The directory object at the top of the hierarchy is known as the **root** directory. You can add directory objects beneath the root directory and beneath other directory objects.

The **table objects** identify **table** databases. The table object contains the scheme by which columns within the table can be identified and searched. Each table contains information about users, machines, or resources on the network. The normal set of 16 tables store information for:

hosts	bootparams	password	cred
group	netgroups	mail aliases	timezone
networks	netmasks	ethers	services
protocols	rpc	auto.home	auto.master

The **group objects** contain a list of members of the group. An NIS+ group is a collection of users and workstations identified by a single name. They are assigned access rights as a group. Essentially, this is used to set security.

All **objects** have a common set of properties. These are:

principal owner
group owner
access rights
unique id
time to live values

Also, each object type specifies information describing the type.

Link objects point to the name of another object.

21.3.3 Names

In general you can name directories any name you like. Two names are reserved, however: **org_dir** and **groups_dir**. They are reserved only for the objects that store the NIS+ table and group objects, respectively. An NIS+ **domain** consists of a directory object, the groups_dir and org_dir subdirectories, and a set of NIS+ tables.

Names that identify objects in the namespace are known as **regular names**.

Index names identify rows within a table. These are compound names containing a **search criterion** and a **regular name**. The regular name specifies the table to search, while the search criterion specifies the column values to search for within the table.

21.3.4 Authorization and Authentication

NIS+ authorization allows four classes of **principals**:

- owner — of the object
- group — set of specified users
- world — set of authenticated users
- nobody — all clients

and four **access rights**:

- read — read contents of objects
- modify — change objects
- create — add objects to tables and directories
- destroy — remove objects from tables and directories

Authentication is based on secure RPC. Solaris 2 supports three levels:

- none — no authentication
- LOCAL — AUTH_SYS RPC authentication
- DES — AUTH_DES Secure RPC

DES authentication is the most secure, but if you are running with Secure RPC you will not be able to mount files from servers not running Secure RPC (i.e. SunOS 4.X servers).

Authentication is performed for every NIS+ request. If credentials can not be confirmed the client is treated as **nobody**.

21.3.5 Configuration

The familiar ***yp**** commands have been replaced with commands beginning with ***nis***. The NIS+ administrative commands are located in **/usr/bin**, **/usr/sbin** and **/usr/lib/nis**.

Starting with SunOS 5.3 Sun has added some scripts to assist you in setting up an NIS+ system. These scripts can be found in **/usr/lib/nis**. They automate setting up servers, clients, and populating NIS+ tables. The scripts are:

- ***nisserver*** — set up NIS+ servers, root master, non-root master, and replica servers
- ***nisclient*** — initialize NIS+ credentials for hosts and users
- ***nispopulate*** — populate NIS+ tables from files or NIS maps

21.3.5.1 Initialize a Server

The nisinit command is used to setup a client, master server, or replica server for NIS+. To initialize the root server use the ***-r*** option:

```
# nisinit -r
```

This should only be run once for the name space. It uses the **domainname** specified in **/etc/defaultdomain** and places it's root object in the directory **/var/nis**.

21.3.5.2 Tables

The ***nissetup*** shell script is found in **/usr/lib/nis**. It creates org_dir and groups_dir directories and the standard tables, though empty, in an NIS+ directory. The domain should have first been created with the ***/usr/bin/nismkdir*** command. Subdirectories are removed with the ***nisrmdir*** command. Copies of the information are automatically passed to replica servers.

21.3.5.3 Credentials

The ***/usr/bin/nisaddcred*** command is used to create credentials for an NIS+ principal. These credentials are stored in the cred.org_dir public key table. You can add **local** or **des** credentials for the principal, e.g.:

```
# nisaddcred -p <uid> -P login.domain local
```

21.3.5.4 Permissions

Change permission attributes of an object with the ***/usr/bin/nischmod*** command. You must have modify access to the object before you can change the attributes.

The ***/usr/bin/nisls*** command can be used to list the **objects** and **permissions** of an NIS+ directory.

21.3.5.5 Table Entries

The ***/usr/lib/nis/nisaddent*** utility is used to add table entries. It can use NIS maps, /etc files, NIS+ tables, or command line arguments as it's source. With ***nisaddent*** you can dump entries from a table into a file. To enter the /etc/hosts table into the NIS+ database you could do the following.

```
# cat /etc/hosts | /usr/lib/nis/nisaddent -av hosts
    adding stdin to table hosts.org_dir.your.domain.
    adding/updating localhost
    adding/updating nyssa
    ...
```

You can administer NIS+ tables with ***/usr/bin/nistbladm***. This command will allow you to create and delete tables, add entries to and modify entries within tables, and remove entries from tables.

You can display NIS+ tables and objects with the ***/usr/bin/niscat*** command, e.g.:

```
# niscat -h netmasks.org_dir
    # number mask comment
    128.146          255.255.255.0
```

The commands ***nismatch*** and ***nisgrep*** in /usr/bin can be used to match keywords and grep for regular expressions, respectively, in NIS+ tables.

21.3.5.6 Defaults

Default values for principal name, domain name, host name, group name, access rights, time to live, and search path can be obtained with the ***nisdefaults*** command in /usr/bin.

21.3.6 NIS+ Setup

These next few sub-sections indicate how to set up **root**, **master**, and **replica** servers, and **client** machines.

21.3.6.1 Root Master Server

1. Choose a domainname
   # domainname acs.ohio-state.edu.
   # domainname > /etc/defaultdomain
2. Choose the NIS+ version for nsswitch.conf
   # cp /etc/nsswitch.nisplus /etc/nsswitch.conf
3. Initialize the server
   # nisinit -r
   where
   -r root server
4. Start the daemon
   # rpc.nisd -rS 0
   where
   -r indicates a root server
   -S 0 sets the security level to 0, i.e. non-secure, does not enforce access controls
5. Setup the NIS+ directory structure
   # /usr/lib/nis/nissetup acs.ohio-state.edu.
6. Add data to the tables
   cat <file> | nisaddent -a <tablename>
   where
   -a specifies to add entries without deleting existing entries
7. Verify the entries, e.g.
   # niscat hosts.org_dir

21.3.6.2 SubDomain Server (Non-Root Master)

The **root** and **master** servers can be the same machine.

1. Become a client of the parent domain
   # domainname wks.acs.ohio-state.edu.
   # domainname > /etc/defaultdomain
   # cp /etc/nsswitch.nisplus /etc/nsswitch.conf
   # nisinit -c -H <domain server hostname>
   where
   -c initializes an NIS+ client
   -H hostname specifies hostname is the trusted server

2. Start the daemon in non-secure mode
   # rpc.nisd -S 0
3. Make the directories for the databases
   # nismkdir -m <subdomain server name> wks.acs.ohio-state.edu.
   where
   -m hostname   create the directory with hostname as the master server
4. Restart the NIS+ daemon
   # ps -ef | grep rpc
   # kill <pid>
   # rpc.nisd -S 0
5. Setup the NIS+ tables
   # /usr/lib/nis/nissetup wks.acs.ohio-state.edu.
6. Add data to the tables
   # cat <file> | nisaddent -a <tablename>

21.3.6.3 Replica Server

A replica server binds to a domain.

1. Become a client of the parent domain
   # domainname wks.acs.ohio-state.edu.
   # domainname > /etc/defaultdomain
   # cp /etc/nsswitch.nisplus /etc/nsswitch.conf
   # nisinit -c -H <domain server hostname>
2. Start the daemon
   # rpc.nisd -S 0
3. Make the directories for the databases
   # nismkdir -s <replica server hostname> acs.ohio-state.edu.
   where
   -s hostname   specify hostname to be a replica server for the existing directory, <domain name>
4. Replicate the domain
   # /usr/lib/nis/nisping acs.ohio-state.edu.

21.3.6.4 Client

A client binds to a sub-domain.

1. Setup the sub-domain
   # domainname wks.acs.ohio-state.edu.
   # domainname > /etc/defaultdomain
2. Choose the NIS+ version for nsswitch.conf
   # cp /etc/nsswitch.nisplus /etc/nsswitch.conf
3. Initialize the client
   # nisinit -c -H <domain server hostname>

21.3.7 Credential Setup

To gain authorization to change NIS+ databases you need to create your security credentials for the NIS+ principals. These credentials are stored in the **cred.org_dir** table in the default NIS+ domain.

21.3.7.1 Root Master

Setting Up Credentials for the Root Master Server

1. Login as root on the root master server and create the credential for the root master at the highest security level
   # nisaddcred des
2. Create the group nisadmin and the master host to the group
   # nisgrpadm -c nisadmin.acs.ohio-state.edu.
   # nisgrpadm -a nisadmin.acs.ohio-state.edu. master_host_name.acs.ohio-state.edu.
3. Update the NIS+ keys
   # nisupdkeys acs.ohio-state.edu.
   # nisupdkeys org_dir.acs.ohio-state.edu.
   # nisupdkeys groups_dir.acs.ohio-state.edu.
4. Kill and restart the rpc.nisd with the new security level enforced
   # ps -ef | grep rpc.nisd
   # kill rpc.nisd_pid_number
   # rpc.nisd -r
5. Set the permissions and group ownerships for the directories
   # nischmod g=rmcd acs.ohio-state.edu. org_dir.acs.ohio-state.edu. groups_dir.acs.ohio-state.edu.
   # nischgrp nisadmin.acs.ohio-state.edu. acs.ohio-state.edu.
6. Set the environmental variable NIS_GROUP. To do this permanently add this variable to /.profile and /.login, e.g.
   # setenv NIS_GROUP nisadmin.acs.ohio-state.edu.

21.3.7.2 Clients

Setting Up Credentials for Client Hosts

1. Login as root on the root master server and define the client host as a principal. You'll be prompted for the root password of the client host. You can also add the client host to the group nisadmin.acs.ohio-state.edu.
   # nisaddcred -p unix.host_name@acs.ohio-state.edu -P host_name.acs.ohio-state.edu. des
2. To allow the root user on the client host to update the maps, add that host to the NIS+ group, nisadmin.acs.ohio-state.edu.
   # nisgrpadm -a nisadmin.acs.ohio-state.edu. host_name.acs.ohio-state.edu.
3. Login as root on the client host and enter the password for root of that host.
   # keylogin -r
   Password:

4. If the root user on the client host is to update the maps, then on the client host set the environmental variable NIS_GROUP. To do this permanently add this variable to /.profile and /.login, e.g.
   # setenv NIS_GROUP nisadmin.acs.ohio-state.edu.

21.3.7.3 Users

Setting Up Credentials for Users

1. Login as root on the root master server and create the user account. This can be done with admintool. Add a password for the user account using the nispasswd command and add the credentials using nisaddcred.
   # admintool
   # nispasswd login_name
   Password:
   # nisaddcred -p uid# local
   # nisaddcred -p unix.uid#@acs.ohio-state.edu -P login_name.acs.ohio-state.edu. des
   Password:
2. To allow the user to change the NIS+ maps, the user must be added to the NIS+ group, nisadmin.acs.ohio-state.edu.
   # nisgrpadm -a nisadmin.acs.ohio-state.edu. login_name.acs.ohio-state.edu.
3. If the user is to update the maps using admintool you must create the group ***sysadmin*** with gid=14 and then add this user as a member of the sysadmin group.
4. Set the user's environment variable NIS_GROUP. To do this permanently add this variable to ~/.profile and ~/.login, e.g.
   # setenv NIS_GROUP nisadmin.acs.ohio-state.edu.

CHAPTER 22 Adding Clients

22.1 Clients

There are four types of SunOS clients: diskless, dataless, standalone, and AutoClient. These must all be served by an NFS file server over the network. The **diskless** client has no disk. It must get root, swap, home, and system files from the server. The **dataless** client has a small local disk. It may have root and swap local, with system and home files on the server. The **standalone** client has all the files necessary to run the OS but may desire additional file systems from the server, e.g. the user home directories. The **AutoClient** is similar to the diskless client, except that it has a small disk for local caching.

An AutoClient should have an entire disk, of at least 100 MB, devoted to it. It uses the Cache File System (cachefs) to locally store /, /usr, and other cachefs mounted file systems. It can also use NFS to mount other file systems. It allows for fast access with centralized administration because there's no permanent data on the client. Everything is quickly reproduced by rebooting over the network from the server. You can install a client with the **Solstice Host Manager** GUI tool of SunOS 5.5+.

Under SunOS 5.3-5.4 you add clients with the **Admintool Host Manager** GUI tool. We look at these ***admintool*** and ***solstice*** utilities in another chapter.

22.2 Server configuration and software

A server can be of the same (homogeneous) or different (heterogeneous) architecture from the client(s). If different it needs to have the executables for the client architecture installed in addition to its own, e.g if you have a sun4 server with sun3 and sun3x clients you need to have:

/export/root	client root partitions	/
/export/swap	client swap files	swap
/export/exec/sun3	Sun3 client system files	/usr
/export/exec/kvm/sun{3,3x}	client kernel files	/usr/kvm
/export(usr)/share	files shared by all systems, e.g. man	/usr/share
/export/sun3/local	locally compiled programs for Sun3/3x	/usr/local

Under SunOS 4.1.X if your server was not installed as a heterogeneous server you can add different architecture executables to the server by running the program ***/usr/etc/install/add_services*** along with the appropriate install tapes/CDROMs for these architectures. ***Add_services*** is a menu-based program that sets up a system as a server for an additional architecture or to add additional software from the release tapes/CDROMs.

22.3 Installing the client of a server, SunOS 4.1.X

On the server add the client's ethernet address to **/etc/ethers**, and the IP address to **/etc/hosts**. Then cd to **/usr/etc/install** and run ***add_client***. e.g.

```
# (cd /usr/etc/install; ./add_client)
usage:   add_client [[options] clients]
        where 'clients' is the name of the any number of clients to add and
         [options] are one or more of the following for each client:
             -a arch       architecture type (e.g. sun3, sun4, etc.)
             -e path       path to client's executables
             -f path       path to client's share location
             -h path       path to client's home directory
             -i            interactive mode - invoke full-screen mode
             -k path       path to client's kernel executables
             -m path       path to client's mail
             -p            print information of existing client
             -r path       path to client's root
             -s path       path to client's swap
             -t termtype   terminal to be used as console on client
             -v            verbose mode - reports progress while running
             -y type       client's NIS type (client, or none)
             -z size       size of swap file (e.g. 16M, 16000K, 32768b  etc.)
             -n            prints parameter settings and exits w/o adding client
```

To run the program interactively use the ***-i*** option to ***add_client***, e.g.:

```
# (cd /usr/etc/install; ./add_client -i)
```

This will invoke a full-screen display for entering the client information similar to the display used during Suninstall.

This program will add all the necessary information to the ***/etc/bootparams*** file, make the boot file in **/tftpboot** (using the client's hex IP address in the file name, which is a symbolic link to the boot program for the appropriate architecture of the client), create the client's root file system in **/export/root/*client_name***, create the client's swap file as **/export/swap/*client_name***, add entries in **/etc/exports** to export the necessary file systems to the client, and add entries in the client's **/etc/fstab** file (**/export/root/*client_name*/etc/fstab** on the server) to enable the client to mount the server's file systems at boot. You'll need to make sure that the appropriate entry is in **/etc/ethers** for the clients ethernet address, hostname pair.

You will then need to run ***exportfs*** to correctly export the file systems to the new client. You should then be able to boot the new client. First, though, check the server's **/etc/bootparams** and **/etc/exports** file to make sure that the entries are appropriate.

22.4 JumpStart

To add a client that you want to boot via jumpstart follow the steps below. First copy the install CDROM to disk. In this example that CDROM is copied to **/jumpstart/solaris_2_5**.

1. Go to the jumpstart directory and add the client:

   cd /jumpstart/solaris_2_5
2. This updates /etc/bootparams, /etc/ethers, /etc/hosts, & /tftpboot.

   ./add_install_client -i ip_address -e ethernet_address -s nyssa:/jumpstart/solaris_2_5/export/exec/kvm/sparc.Solaris_2.5 hostname sun4c
3. If you're running with **TCPwrapper** enable the client to access ***in.tftpd*** by editing **/etc/hosts.allow**.
4. If **/tftpboot** did not exist at boot time run "***/etc/init.d/nfs.server start***" to enable the server to start ***/usr/sbin/rpc.bootparamd***.
5. Boot the client from the network:

   ok boot net - install

22.5 AutoClient

Solstice AutoClient software comes with Solaris 5.5.1 on the same CDROM as the AdminSuite software. Both these applications should be installed. You can use either the **Solstice Host Manager** tool or the /opt/SUNWadm/2.2/bin/***admhostadd*** command to add the client. First setup the OS Server; again, this can be done with the Solstice Host Manager tool. If you're using **NIS** be sure to setup a **timezone** map with entries of the form:

hostname timezone   or   domainname timezone

e.g.:

nis_domain US/Eastern

The ***admhostadd*** program lets you specify the same type information you would fill in the Host Manager. Use it in the form:

```
admhostadd -i IP_addr [ -e ethernet_addr ]
        [ -x type=host_type ] [ -x tz=timezone ] [ -x term=type ]
        [ -x fileserv=file_server ] [ -x root=directory ]
        [ -x swap=directory ] [ -x swapsize=size  [ -x disconn=Y|N ]
        [ -x install=Y|N ] [ -x installpath=server:/path ]
        [ -x bootpath=server:/path ] [ -x profile=server:/path ]
        [ -x os=version ] [ -x diskconf=configuration ]
        [ -x ns=NIS|NIS+|NONE ] [ -x domain=domain|rhost=host ]
        host
```


As with the diskless client you need to make sure that your /etc/ethers and /etc/bootparams files (or the NIS(+) equivalent maps) are properly setup and that the OS server properly shares the files needed by the client.

The AutoClient can then be booted from the network similar to a diskless client, e.g.:

```
ok boot net
```

PART III Selected Topics

Useful Utilities

Print Service

Mail

World Wide Web

Usenet

System Security

Secure Shell

CHAPTER 23 Usenet

23.1 Usenet

Usenet is a world-wide network of computers, most of them UNIX, that run netnews software. It's a public forum for the exchange of news articles, similar to bulletin boards. The articles are exchanged between sites by mutual agreement between the System Administrators of the sites. This requires either an ethernet or UUCP connection between the machines. Users can post, read, and reply to articles in any of over 13000 different topics, or newsgroups on the Internet. Locally you can find newsgroups that range from osu.general (site specific to OSU) to comp.sys.sun.admin (the Sun administrators list) to alt.tv.simpsons.

The major **Usenet** headings are:

comp	- computer science related groups
sci	- sciences other than computer science,
news	- netnews software and general interest for netnews users,
rec	- discussions of recreational activities,
soc	- discussions of social topics,
talk	- extended discussion of special topics (e.g. talk.politics),
misc	- groups other than those listed above.
also:	
can	- Canadian
clari	- Clarinet (UPI) news feed
bit	- BITNET lists

Groups within these headings are created by consensus of the Usenet users. In addition there are other groups such as "**alt**" for alternate, that are created at the whim of individual users. These groups may not be carried by all Usenet news sources. Additionally there are groups with more limited distribution, such as **cle**, **cmh**, and **oh**, and local groups, such as **osu**, **cis**, and **uts**.

23.2 Reading news, rn/rrn/xrn/trn/nn

Reading news requires a program that can select news groups, articles within those groups, and page through them. It should also be able to keep track of articles you've read and topics you want to see, or don't want to see.

Many popular read news programs are derived from ***rn***:

- ***rn*** - simple unthreaded newsreading,
- ***trn*** - adds threading and other useful features to ***rn***; can use **nntp** to read news on a remote server.
- ***xrn*** - X-windows newsreader which uses **nntp** to query a news server.

There are also nntp newsreaders for Macs and PCs. The OSU HomeNet/ResNet/OfficeNet software from UTS now includes these. With this software you can connect to the news server from your desktop computer.

23.3 Network news transfer protocol, nntp

nntp is the Network News Transfer Protocol. This protocol controls the transfer of news articles form the news server to your machine. NNTP **server** machines contain a full installation of USENET news. They allow remote sites to connect and read, transfer and/or post news, as controlled by the **nntp_access** file in **/usr/lib/news**, e.g.:

```
host/net        read/xfer/no        post/no        newsgroups  (!not.allowed)
```

nntp can operate either as a stand-alone server, or as a server under ***inetd***. News articles are placed in the news spool as numbered files in directories specified by the group name, e.g. ***comp.sys.sun.admin*** would become **/var/spool/news/comp/sys/sun/admin**.

There are three major **news servers** on Campus:

magnus.acs.ohio-state.edu
zaphod.mps.ohio-state.edu
news.cis.ohio-state.edu

23.4 Disk space requirements

A full news feed requires massive amounts of disk space, currently a Gbyte or more of disk space per day and growing rapidly. Also, since the articles are generally smaller than the default inodes/disk space of 2 Kbytes, you may run out of inodes before you run out of disk space. You should anticipate this when you set up the news partition and run ***newfs*** with the desired inode density.

23.5 Relevant UNIX newsgroups

23.5.1 UNIX - news

clari.nb.unix	UPI stories related to UNIX

23.5.2 SunOS

comp.sys.sun.admin	Sun administrators list
comp.sys.sun.apps	
comp.sys.sun.announce	
comp.sys.sun.hardware	
comp.sys.sun.misc	
comp.sys.sun.wanted	
comp.unix.solaris	Solaris 2 (SunOS 5.X) related concerns
osu.sys.sun	Local Sun related concerns

23.5.3 HP-UX

comp.sys.hp.hpux	HP-UX administrators list
comp.sys.hp.hardware	
comp.sys.hp.apps	
comp.sys.hp.misc	
osu.sys.hp	Local HP-UX related concerns

23.5.4 Ultrix

comp.unix.ultrix	Ultrix administrators list
osu.sys.dec.ultrix	Local Ultrix related issues

23.5.5 SGI

comp.sys.sgi.announce	
comp.sys.sgi.admin	SGI administrators list
comp.sys.sgi.apps	
comp.sys.sgi.bugs	
comp.sys.sgi.graphics	
comp.sys.sgi.hardware	
comp.sys.sgi.misc	
osu.sys.sgi	Local SGI related issues

23.5.6 Linux

osu.sys.linux	Local Linux users list
comp.os.linux.admin	Linux administrators list
comp.os.linux.announce	
comp.os.linux.development	
comp.os.linux.misc	
comp.os.linux.help	

23.5.7 NeXT

comp.sys.next.sysadmin	NeXT users list
comp.sys.next.announce	
comp.sys.next.misc	
comp.sys.next.programmer	
comp.sys.next.hardware	
comp.sys.next.software	
comp.soft-sys.nextstep	
cmh.sys.next	Local NeXT users list

23.5.8 AIX

comp.unix.aix	IBMs AIX users list

23.5.9 Digital Unix and OSF/1

comp.unix.osf.osf1	OSF/1 related concerns

23.5.10UNIX - technical

comp.unix.admin	UNIX Administration
osu.network	Networking issues at OSU/OSC
osu.unix	Local UNIX concerns
comp.unix.shell	UNIX shell (sh, csh, tcsh, bash, etc.)

23.5.11 Security

alt.security	Discussions of Security Issues
comp.security.announce	Security Announcements
comp.security.misc	
comp.security.unix	

23.5.12 Sources

alt.sources
comp.sources.misc
comp.sources.reviewed
comp.sources.sun
comp.sources.unix

23.5.13 Perl

comp.lang.perl.moderated
comp.lang.perl.announce
comp.lang.perl.modules
comp.lang.perl.tk
comp.lang.perl.misc

CHAPTER 24 Useful Utilities

24.1 Format online manual pages, catman

catman creates the display files used by the manual command, ***man***. These files are put in directories under /usr/man to match the /usr/man/man* entries, i.e. **/usr/man/cat[1-8,l,n]** (SunOS 4.1.X) or **/usr/man/cat[1[,b,c,f,m,s],2,3[,b,c,e,g,i,k,m,n,r,s,t,x],4,4b,5,6,7,9[,e,f,s],l,n]** (SunOS 5.X) and a database of the one-line synopses are put in **/usr/man/whatis** (SunOS 4.1.X) or **/usr/man/windex** (SunOS 5.X) for use by the ***whatis*** and "***man -k keyword***" commands. Running ***catman*** doubles the space required to contain the man pages, but allows the ***man*** command to execute considerably faster.

The man pages generally follow the conventions given in the following table.

TABLE 24.1 **Man Page Placements**

SunOS 4.X	SunOS 5.X	Description
man1	man1	user commands - from the shell prompt
man2	man2	system calls - C functions interfacing between user programs and the kernel
man3	man3	user level library functions - C library functions for user programs
man4	man7 & man9	device drivers and network interfaces - describes access to special files in /dev
man5	man4	file formats - describes formats used by system programs
man6	man6	games and demo descriptions
man7	man5	miscellaneous - including standards and text processing
man8	man1m	system administration - commands for system maintenance and operation
manl	manl	locally installed man pages
mann	mann	new man pages

You can install other man pages under any hierarchy, e.g. **/usr/local/man** or **/usr/lang/man**, and make them accessible to the ***man*** command by setting the **MANPATH** environment variable to include them, i.e. for the C-shell:

```
% setenv MANPATH /usr/local/man:/usr/man:/usr/lang/man
```

and for the Bourne shell:

```
MANPATH=/usr/local/man:/usr/man:/usr/lang/man ; export MANPATH
```

24.2 System process status, ps

ps displays information about processes currently running. The results of the ***ps*** command are very system dependent, so read the **man pages** for the specifics on your machine.

Without options ***ps*** tells you what current programs you own, e.g.:

```
% ps
  PID    TT   STAT   TIME   COMMAND
 12263   p6   S      0:02   -tcsh (tcsh)
 12608   p6   R      0:00   ps
```

where **PID** is the process ID number, **TT** is the terminal port, **STAT** is the status of the program (i.e. runnable, R; stopped, T; waiting, P or D; sleeping, S; idle, I; terminated, Z), **TIME** is the CPU time consumed, and **COMMAND** is the program.

The options to ps and it's display is a little different between SunOS 4.1.X and 5.X. To look at all process running on the system, by all users, use the options "*auxww*" under SunOS 4.1.X and the options "*-ef*" under SunOS 5.X, e.g. for SunOS 4.1.X:

```
% ps -auxww
  USER   PID  %CPU  %MEM  SZ   RSS  TT  STAT  START   TIME   COMMAND
  frank  514  38.5  3.2   144  376  p2  R     13:43   0:00   ps -auxww
  root   113  0.8   0.1   24   16   ?   S     May 14  16:10  update
  root     2  0.0   0.0   0    0    ?   D     May 14  0:00   pagedaemon
  root     1  0.0   0.0   72   0    ?   IW    May 14  0:00   /sbin/init -
  root   44   0.0   0.0   56   0    ?   IW    May 14  0:00   portmap
  frank  141  0.0   0.0   96   0    co  IW    May 14  0:00   -tcsh (tcsh)
  root   102  0.0   0.0   72   0    co  IW    May 14  0:00   rpc.statd
  root    50  0.0   0.0   56   0    co  IW    May 14  0:00   keyserv
  bin     47  0.0   0.0   40   0    ?   IW    May 14  0:00   ypbind
  root    74  0.0   0.0   24   0    ?   I     May 14  0:01     (biod)
  root    76  0.0   0.0   24   0    ?   I     May 14  0:01     (biod)
  root    90  0.0   0.0   72   0    ?   IW    May 14  0:00   syslogd
  root   117  0.0   0.0   80   0    ?   IW    May 14  0:00   cron
  root    77  0.0   0.0   24   0    ?   I     May 14  0:01     (biod)
  root   101  0.0   0.0   80   0    co  IW    May 14  0:00   rpc.lockd
  frank  174  0.0   0.0   0    0    ?   Z     May 14  0:00   <defunct>
  root     0  0.0   0.0   0    0    ?   D     May 14  0:01   swapper
```

This includes additional information, where **USER** is the owner of the process, **MEM** is the percentage of real memory the process is using, **SZ** is the size of the data and stack segments (in Kbytes), **RSS** is the real memory used (in Kbytes), and **START** is the time (or day) when the program was started.

24.3 Swap space and kernel inode usage, pstat

pstat lists the contents of certain system tables kept by the kernel. Its available only with SunOS 4.1.X, e.g.:

```
% pstat -T
    301/1888        files
    694/946         inodes
     83/522         processes
     16/ 32         files
    14232/88296     swap
```

where it shows the file table, both the used and cached inodes, process table, stream table, and used and available swap space (in Kbytes). With the "***-s***" option you can get a little more information about swap space, e.g.:

```
% pstat -s
    10968k allocated + 2648k reserved = 13616k used, 74680k available.
```

The "***swap -s***" command of SunOS 5.X will provide similar information.

There are other options to ***pstat*** to provide further system information.

24.4 top

Top is a PD program available on the Internet. The ***top*** program displays a screenful of the top cpu processes that is updated every few seconds.

```
last pid:  2638;  load averages:  0.15,  0.20,  0.22                  16:37:53
66 processes:  63 sleeping, 2 zombie, 1 on cpu
Cpu states:  0.0% idle, 78.5% user,  2.5% kernel,  7.6% wait, 11.4% swap
Memory: 26332K real, 1652K free, 61096K swap, 38568K free swap

  PID USERNAME PRI NICE   SIZE    RES STATE   TIME   WCPU    CPU COMMAND
 2638 frank     34    0  1832K  1016K sleep   0:00  4.07%  4.07% xwd
  243 frank     24    0 16136K  4968K sleep1440:28  0.15%  1.75% Xsun
 2637 frank     34    0  1044K   932K cpu     0:00  0.10%  1.16% top
  286 frank     24    0  1100K   584K sleep   0:06  0.03%  0.39% tcsh
  280 frank     34    0  3276K   616K sleep  19:01  0.02%  0.19% perfmeter
  282 frank     34    0  3276K   660K sleep   4:30  0.02%  0.19% perfmeter
  264 frank     34    0  2216K   856K sleep   0:20  0.02%  0.19% olvwm
  269 frank     34    0  2948K   596K sleep   0:17  0.02%  0.19% xterm
  271 frank     34    0  3576K  1972K sleep   0:02  0.02%  0.19% xterm
 2077 frank     34    0 14528K  8176K sleep  11:21  0.00%  0.00% maker4X.exe
    1 root      34    0   692K     0K sleep   8:05  0.00%  0.00% init
 1269 frank     34    0  5632K  1336K sleep   1:28  0.00%  0.00% mailtool
  278 frank     34    0  3272K   612K sleep   0:36  0.00%  0.00% perfmeter
   91 root      34    0  1720K   192K sleep   0:21  0.00%  0.00% rpcbind
 2459 frank     34    0  9780K   432K sleep   0:19  0.00%  0.00% navigator
```

24.5 vmstat

To report on virtual memory statistics; process, virtual memory, disk, trap, and CPU activity use the ***vmstat*** command, e.g.:

```
% vmstat
 procs     memory            page            disk          faults      cpu
 r b w   swap  free  re  mf pi po fr de sr f0 s1 s3 s5   in   sy   cs us sy id
 0 0 0    984  1840   0   1  1  0  1  0  0  0  0  0  0   29  509  151 25  7 68
```

The "***-S 5***" options will report on swapping, rather than paging activity every 5 seconds, e.g.

```
% vmstat -S 5
 procs     memory            page            disk          faults      cpu
 r b w   swap  free  si  so pi po fr de sr f0 s1 s3 s5   in   sy   cs us sy id
 0 0 0    984  1840   0   0  1  0  1  0  0  0  0  0  0   29  509  151 25  7 68
 0 0 0  61172   628   0   0  0  0  0  0  0  0  0  0  0   24  111   39  0  0 100
 0 0 0  61172   628   0   0  0  0  0  0  0  0  0  0  0   72  395   73  2  2 96
 0 0 0  61172   628   0   0  0  0  0  0  0  0  0  2  0   35  212   45  6  2 92
 0 0 0  61172   628   0   0  0  0  0  0  0  0  0  0  0   18  129   44  0  0 100
 0 0 0  61172   628   0   0  0  0  0  0  0  0  0  0  0   55  324   65  3  2 95
 0 0 0  61172   628   0   0  0  0  0  0  0  0  0  0  0   58  305   61  3  2 96
 0 0 0  61172   628   0   0  0  0  0  0  0  0  0  0  0   34  144   45  0  1 99
```

where

procs

r	in run queue
b	blocked for resources (i/o, paging, etc.)
w	runnable

memory (usage of virtual and real memory)

swap	swap space currently available (Kbytes)
free	size of free list (Kb)

page/swap activity

si	swap-ins
so	swap-outs
pi	page-ins (Kb/s)
po	page-outs (Kb/s)
fr	Kb freed/sec
de	anticipated short term memory shortfall (Kb)
sr	pages scanned/sec

disk

Number of disk operations/sec for each of up to 4 disks.

faults (Trap/Interrupt average rate)

in	(non-clock) device interrupts/sec
sy	system calls/sec
cs	CPU context switches/sec

cpu

us	user time
sy	system time
id	idle

This can provide useful information for evaluating NFS file server performance. You could run this for about an hour during peak periods to collect meaningful statistics. CPU idle time should be at least 10% inorder for the system to efficiently schedule daemons and to process protocols.

The "***-s***" option will display the contents of the sum structure, related to paging events, e.g.

```
% vmstat -s
        0 swap ins
        0 swap outs
        0 pages swapped in
        0 pages swapped out
  6458837 total address trans. faults taken
   752003 page ins
   135318 page outs
  1419068 pages paged in
   515004 pages paged out
    76738 total reclaims
    71392 reclaims from free list
        0 micro (hat) faults
  6458837 minor (as) faults
   734466 major faults
  1338667 copy-on-write faults
  2067746 zero fill page faults
  2443859 pages examined by the clock daemon
      156 revolutions of the clock hand
  1374036 pages freed by the clock daemon
    43658 forks
     1497 vforks
    54907 execs
811460734 cpu context switches
691373136 device interrupts
 46740506 traps
2727532551 system calls
 34362625 total name lookups (cache hits 93%)
    10675 toolong
133937046 user   cpu
 37507920 system cpu
362695349 idle   cpu
  1386715 wait   cpu
```

If the "**total name lookups**" cache hit rate is a low percentage (< 70%) on a SunOS 4.1.X NFS server you should consider increasing the value of **MAXUSERS** to 128 and increase **nbuf** (default is 27, increase to 64 for 1-4 disks, 112 for > 4 disks). SunOS 5.X automatically sizes MAXUSERS to fit available memory.

Increasing **MAXUSERS** also increases the values of **nproc** (number of processes allowed), **ninode** (inode cache), **ncsize** (directory cache table), **nfile** (# open files allowed), and **ncallout** (callout queue).

In general, for a SunOS 4.1.X server, if you have 32 MB RAM on your server set MAXUSERS to:

64	≤ 4 disks, or ≤ 10 simultaneous users
128	> 4 disks, or > 10 simultaneous users

Buffer cache is another parameter that can have a large affect on performance. You should reserve about 10% of kernel memory for disk I/O cache to reduce disk I/O. This means increasing ***nbuf*** to 64 for systems with 4 disks (and> 60% busy) or to 112 for systems with > 4 disks.

Additional hardware you can add to increase performance would be a Prestoserve or NC400 board to enhance NFS performance. (With NFS version 3, these hardware cards may not produce as large an improvement as they did with version 2.) If the ethernet traffic is limiting you could add additional ethernet controllers. On compute servers it helps to increase the memory. You can also balance the load across disks and ethernets available to the server.

24.6 iostat

iostat reports on I/O statistics, terminal and disk I/O and CPU utilization. With the following option it reports this information for every disk on the system every 5 seconds on a SunOS 5.X machine:

% iostat 5

```
      tty         fd0           sd1           sd3           sd5          cpu
 tin tout Kps tps serv  Kps tps serv  Kps tps serv  Kps tps serv  us sy wt id
   0    4   0   0    0    1   0   37    2   0   33    0   0  100  25  7  0 68
   0    0   0   0    0    0   0    0    0   0    0    0   0    0   9  7  0 84
   0    0   0   0    0    0   0    0   16   2   80    0   0    0   3  4  0 93
   0    0   0   0    0   12   3    8   88  22   12    0   0    0   2  6 25 68
   2    1   0   0    0    9   2    8   23   6   13    0   0    0   0  1  9 90
   1   33   0   0    0   23   5    9   19   5   13    0   0    0   2  3  9 86
   2   63   0   0    0    5   1   10    0   0    0    0   0    0   1  2  0 96
   1 4389   0   0    0  218  57   10   11   3   15    0   0    0  45 25  8 22
   0 6728   0   0    0  150  58   12    0   0    0    0   0    0  63 29  8  0
   0 7053   0   0    0  126  66   13    1   0   12    0   0    0  59 36  5  0
   0 4902   0   0    0  330  76  265   40   6   25    0   0    0  44 26  9 21
   0 7571   0   0    0  182  57   11   32   6   17    0   0    0  60 31  9  0
   0 7900   0   0    0  144  39    9    4   1   10    0   0    0  56 36  8  0
   0 7885   0   0    0  139  33   11    0   0    0    0   0    0  59 35  6  0
   0 8100   0   0    0  142  29   13   18   3   16    0   0    0  59 37  4  0
   0 2143   0   0    0   44  11   12    4   1   10    0   0    0  18  9  2 71
   3   15   0   0    0  283  35   40    8   1   16    0   0    0   2  4 45 49
   0  171   0   0    0  675  84   73   29   5   18    0   0    0   3  8 85  4
   0    0   0   0    0  106  13  375    3   0   17    0   0    0   1  2  0 97
   2  344   0   0    0    0   0    0    0   0    0    0   0    0   3  4  0 93
```

where the headings refer to:

tin	characters read from terminals
tout	characters written to terminals
kps	Kilo bytes transferred/second
tps	transfers/second
serv	average service time (milliseconds) (estimated)
us	user mode (% of CPU time)
sy	system mode (% of CPU time)
wt	waiting for I/O (% of CPU time)
id	idle (% of CPU time)

The **"*-D*"** option reports the disk activity, reads/sec, writes/sec, and % disk utilization, for each disk, e.g.:

% iostat -D

```
          fd0            sd1            sd3            sd5
 rps  wps  util  rps  wps  util  rps  wps  util  rps  wps  util
   0    0   0.0    0    0   0.1    0    0   0.3    0    0   0.0
```

A disk is heavily loaded if the utilization rate, **util**, is ≥ .75, or I/O operations (**rps + wps**)   40 IOPS. To enhance system performance you might try to spread the load more evenly among the disks. Run **tmpfs**, which reduces disk and network traffic (for diskless clients) and reduces the number of writes to the filesystem where **/tmp** is located.

The "-x" option reports extended disk activity, e.g.:

```
                                     extended disk statistics
disk      r/s  w/s   Kr/s   Kw/s wait actv  svc_t  %w  %b
fd0       0.0  0.0    0.0    0.0  0.0  0.0    0.0   0   0
sd1       0.1  0.0    0.4    0.3  0.0  0.0   36.9   0   0
sd3       0.1  0.1    0.8    0.8  0.0  0.0   33.2   0   0
sd5       0.0  0.0    0.0    0.0  0.0  0.0   99.5   0   0
```

where the headings refer to:

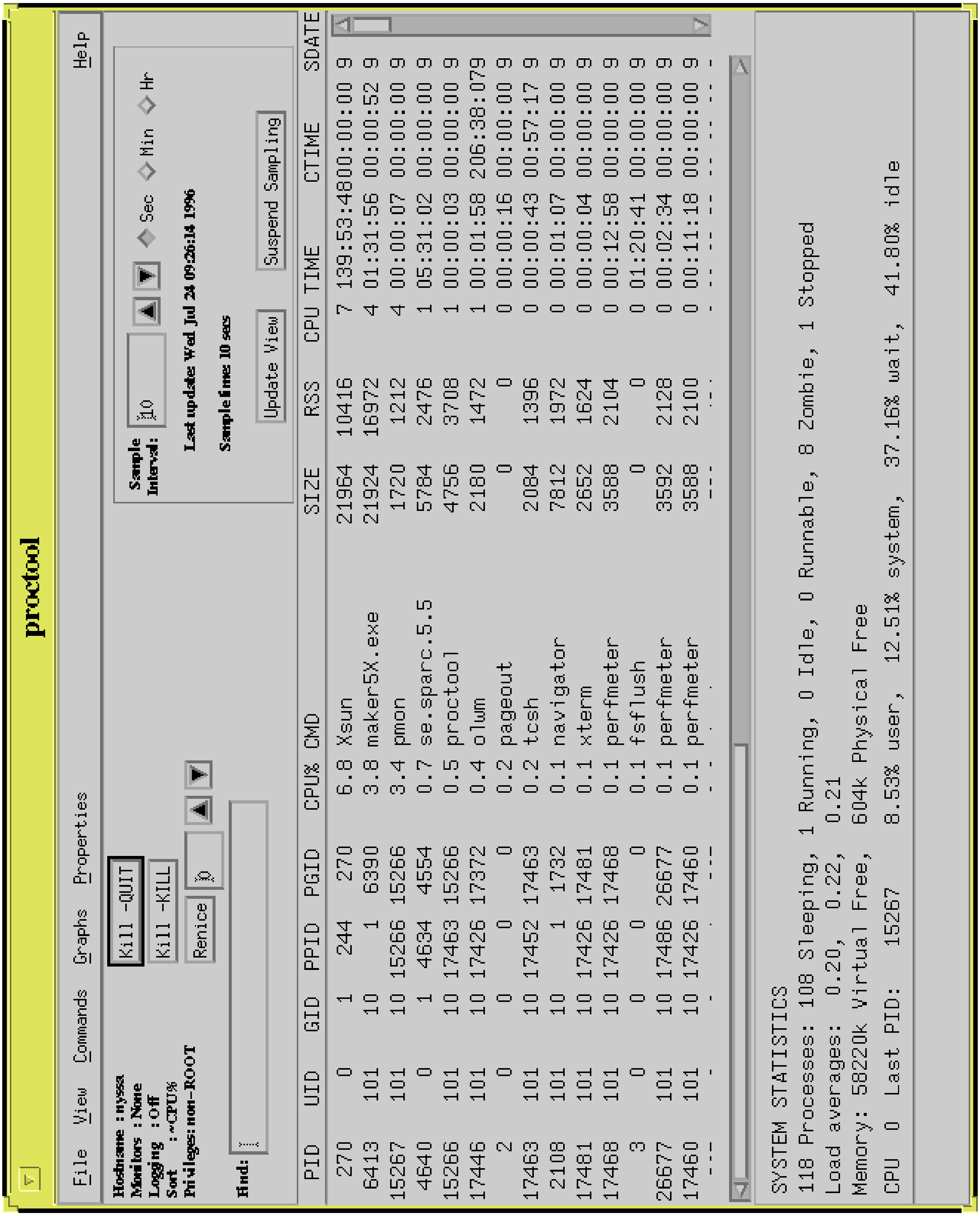
disk	name of the disk
r/s	reads/second
w/s	writes/second
Kr/s	Kilobytes read/second
Kw/s	Kilobytes written/second
wait	average number of transactions waiting for service (length of queue)
actv	average number of transactions currently being served
svc_t	average service time (milliseconds)
%w	% of time there are transactions waiting for service
%b	% of time the disk is busy

24.7 ProCtool

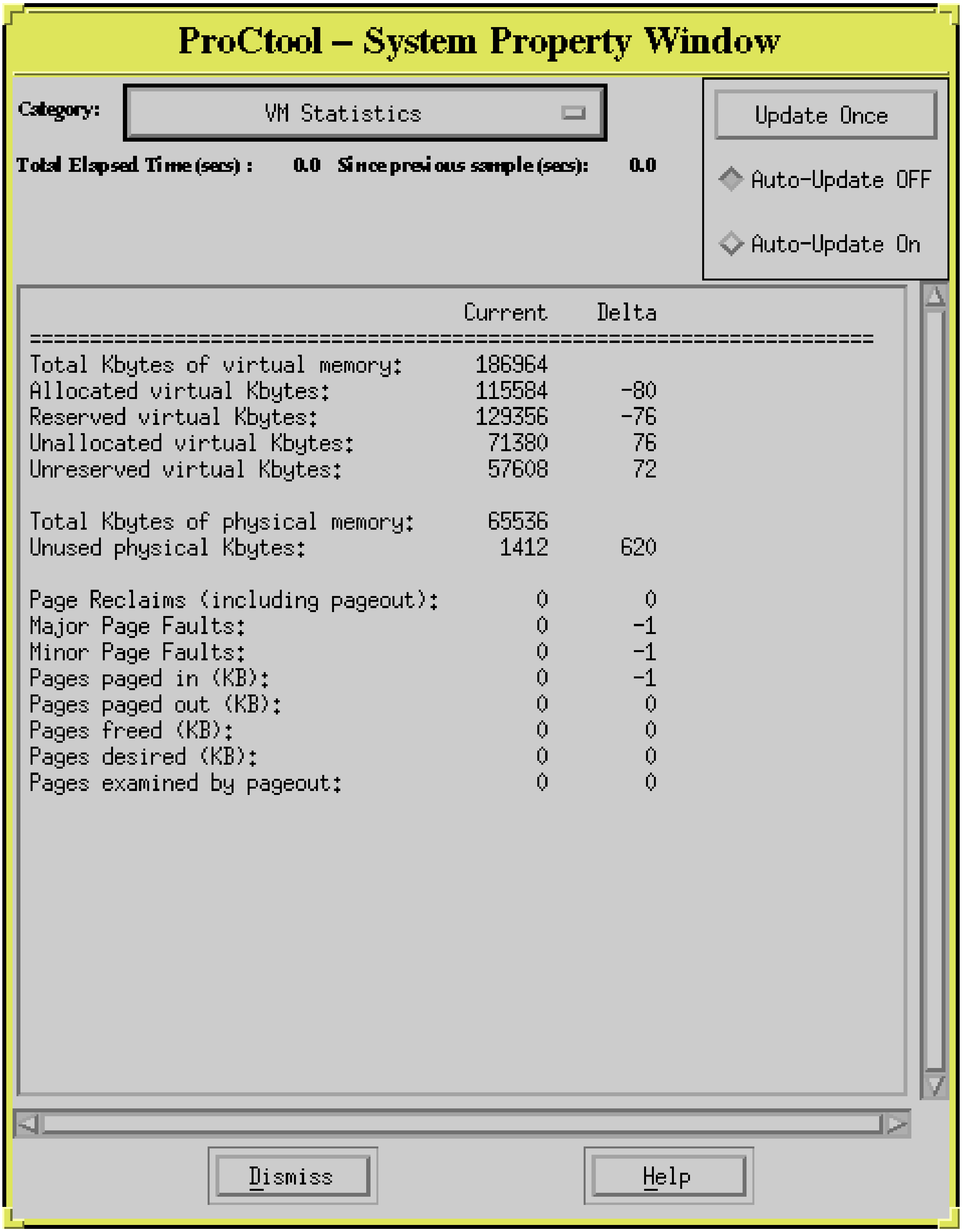
ProCtool is a process monitoring and management tool developed by a couple of Sun engineers, but not officially supported by Sun (Walter Nielsen, walter.nielsen@west.sun.com, and Morgan Herrington, morgan.herrington@west.sun.com). There are versions for Solaris 2.2 and above. It provides the functionality of ps, top, iostat, and much more in a graphical presentation under Open Look. It will continually update the display by sampling the kernel tables. Among it's many features are: report on all active processes (sorted by choice of characteristic); turn off or on CPUs on an MP box; kill or renice selected processes; send signals to a set of processes; report on VM and I/O usage, and paging rate and memory map, etc.; and graph system characteristics. ***ProCtool*** can be obtained via anonymous ftp from sunsite.unc.edu in /pub/sun-info/mde, or locally from www-wks.acs.ohio-state.edu in /pub/proctool.

The desired characteristics for display are setable with the **View** pop-up window under **Viewpoint** and include the following options:

- ADDR - Address of process
- CLS - Scheduling Class
- CMD - Command
- CMDLINE - Command and Arguments
- CPU - CPU 'tick' count
- CPU# - Processor Number
- CPU% - Percentage of CPU Utilization
- CS/S - Context Switches/Sec
- CTIME - Children user+sys CPU time
- FLAGS - Process flags
- GID - Group ID
- HEAP - Heap Size (KBytes)
- IO/S - Characters read+written/Sec
- LWP - LWP count
- MPF/S - Minor Page Faults/Sec
- MSGS/S - Messages sent+received/Sec
- NICE - Nice value
- PF/S - Major Page Faults/Sec
- PGID - Process Group ID
- PID - Process ID
- PPID - Parent Process ID
- PRI - Priority
- RSS - Resident Set Size (KBytes)
- SDATE - Start Date
- SID - Session ID
- SIZE - SIZE (KBytes)
- ST - Process state
- STACK - Stack Size (KBytes)
- STIME - Start Time
- TIME - user+sys CPU time
- TTY - Controlling Terminal
- UID - User ID
- USER - Username
- WCHAN - Address process is waiting on


proctool
File View Commands Graphs Properties Help
Hostname : nyssa
Monitors : None
Logging : Off
Sort : ~CPU%
Privileges: non-ROOT
Kill -QUIT
Kill -KILL
Renice
Find:
Sample Interval: 10
Sec Min Hr
Last update: Wed Jul 24 09:26:14 1996
Sample time: 10 secs
Update View
Suspend Sampling
PID UID GID PPID PGID CPU% CMD SIZE RSS CPU TIME CTIME SDATE
270 0 1 244 270 6.8 Xsun 21964 10416 7 139:53:4800:00:00 9
6413 101 10 1 6390 3.8 maker5X.exe 21924 16972 4 01:31:56 00:00:52 9
15267 101 10 15266 15266 3.4 pmon 1720 1212 4 00:00:07 00:00:00 9
4640 0 1 4634 4554 0.7 se.sparc.5.5 5784 2476 1 05:31:02 00:00:00 9
15266 101 10 17463 15266 0.5 proctool 4756 3708 1 00:00:03 00:00:00 9
17446 101 10 17426 17372 0.4 olwm 2180 1472 1 00:01:58 206:38:079
2 0 0 0 0 0.2 pageout 0 0 0 00:00:16 00:00:00 9
17463 101 10 17452 17463 0.2 tcsh 2084 1396 0 00:00:43 00:57:17 9
2108 101 10 1 1732 0.1 navigator 7812 1972 0 00:01:07 00:00:00 9
17481 101 10 17426 17481 0.1 xterm 2652 1624 0 00:00:04 00:00:00 9
17468 101 10 17426 17468 0.1 perfmeter 3588 2104 0 00:12:58 00:00:00 9
3 0 0 0 0 0.1 fsflush 0 0 0 01:20:41 00:00:00 9
26677 101 10 17486 26677 0.1 perfmeter 3592 2128 0 00:02:34 00:00:00 9
17460 101 10 17426 17460 0.1 perfmeter 3588 2100 0 00:11:18 00:00:00 9
SYSTEM STATISTICS
118 Processes: 108 Sleeping, 1 Running, 0 Idle, 0 Runnable, 8 Zombie, 1 Stopped
Load averages: 0.20, 0.22, 0.21
Memory: 58220k Virtual Free, 604k Physical Free
CPU 0 Last PID: 15267 8.53% user, 12.51% system, 37.16% wait, 41.80% idle

The Properties/System Properties pop-up window is:



24.8 System usage, uptime, users, who and w

uptime shows you how long the system has been running, the number of users presently logged in, and the average load over the last 1, 5, and 15 minutes, e.g.:

```
% uptime
     2:07pm  up 12 days,  3:37,  7 users,  load average: 0.11, 0.02, 0.00
```

rup is similar to uptime, but it gives the status of remote machines, e.g.:

```
% rup
tardis          up 12 days,  3:41,     load average: 0.00, 0.00, 0.00
nyssa           up  2 days,  5:27,     load average: 0.61, 0.15, 0.05
blueagle        up           3:18,     load average: 0.08, 0.02, 0.01
```

users/rusers list the users on the local/remote machines of your network.

```
% users
    frank jeffs jeffs rsf steele steele steele
% rusers
    blueagle    kitw kitw
    nyssa       frank frank
    tardis      steele steele steele jeffs frank rsf jeffs
```

who/w/whodo list who is logged in on the system and the programs they're running, e.g.:

```
% who
    frank      ttyp0   May 4 15:40     (nyssa:0.0)
    robert     ttyp1   Apr 16 14:54    (davros)
    michael    ttyp3   May 4 07:49     (turlough)
% w
    10:49am up 22 days, 2:04, 5 users, load average: 0.00, 0.00, 0.00
    User       tty     login@      idle    JCPU   PCPU    what
    frank      ttyp0   Mon 3pm             16     3       w
    robert     ttyp1   16Apr92     16:06   13:04  6       -tcsh
    william    ttyp2   8:08am              25     1       -tcsh
    michael    ttyp3   Mon 7am     27:00   8      7       twm -display 128.146.116.25:0.0
% w -d                      (SunOS 4.1.X only)
    10:49am up 22 days, 2:04, 5 users, load average: 0.00, 0.00, 0.00
    User       tty        login@     idle    JCPU   PCPU    what
                       12989 -csh & 0 1
                       13881 w -d
    frank      ttyp0      Mon 3pm            16     3       w -d
                       5661 -tcsh & 0 1
    robert     ttyp1      16Apr92    16:06   13:04  6       -tcsh
                       13534 -tcsh & 0 1
                       13879 elm & 1 0
    michael    ttyp3      Mon 7am    27:00   8      7       twm -display 128.146.116.25:0.0
```

```
% whodo frank            (SunOS 5.X only)
Wed Aug  7 10:34:13 EDT 1996
nyssa
console       frank    13:46
      ?           17374    0:00 Xsession
      ?           17455    0:00 xclock
      ?           17446    3:05 olwm
      ?           17424    0:06 ttsession
      ?           17453    0:18 xterm
    pts/6         17478    0:07 tcsh
      ?           17454    1:55 xterm
    pts/5         17475    6:53 tcsh
      ?           17447    0:00 olwmslave
pts/8         frank    13:08
    pts/8          3532    0:25 ssh
pts/7         frank    13:47
    pts/7         17486    0:03 tcsh
      ?           17061    0:05 xterm
    pts/10        17062    0:03 tcsh
    pts/7          3566    9:08 mailtool
      ?            8426    0:00 xterm
    pts/17         8427    0:00 tcsh
pts/9         frank     8:24
    pts/9         14501    0:25 ssh
pts/3         frank     9:46
    pts/3         29243    0:00 rlogin
```

24.9 File Compression, compress & gzip

You can save a lot of valuable disk space by compressing seldom used files. Most versions of Unix come with the ***compress/uncompress*** utilities. Depending on the file contents this may save 50-70% of the space.

There's a GNU compression utility, ***gzip/gunzip*** that does an even better job of compressing files. The source is available as ftp://www-wks.acs.ohio-state:/pub/solaris2/src/gzip-1.2.4.tar.Z.

24.10 Shells, tcsh & bash

The extended C-shell, ***tcsh***, and the Bourne-again shell, ***bash***, provide command name completion, extended history features, in-line editing with both ***emacs*** and ***vi*** modes, command recall via up/down arrow keys, etc. The sources are available in ftp://tesla.ee.cornell.edu and ftp://prep.ai.mit.edu/pub/gnu , for ***tcsh*** and ***bash***, respectively, or both can be obtained locally in ftp://www-wks.acs.ohio-state:/pub/solaris2/src

CHAPTER 25 Print Service

Print Service

25.1 SunOS 4.1.X

25.1.1 Line Printer Spooler System

25.1.1.1 Printer configuration file, /etc/printcap

The file **/etc/printcap** contains the database of printer capabilities and location of the spool files. Each entry of the file describes a printer with fields separated by "**:**". The first entry is the name the printer is known by and any aliases separated by "**|**". Subsequent entries indicate the location and capabilities of the printer.

A print **client** machine might have a printcap entry similar to:

```
lp|ps|postscript|PostScript:\
        :lp=:rm=tardis:rp=lp:sd=/var/spool/lp:lf=/dev/console:
lw|LaserWriter:\
        :lw=:rm=peri:rp=lw:sd=/var/spool/lw:mx#0:lf=/var/spool/lp-log:
```

while the print **server** might have an entry:

```
# PostScript printer driven by TranScript sftw (PostScript & TranScript, tm  Adobe Sys. Inc.)
lp|ps|postscript|PostScript:\
        :lp=/dev/lp:sd=/var/spool/lp:lf=/var/spool/lp-log:af=/var/spool/lp.acct:\
        :br#9600:rw:fc#0000374:fs#0000003:xc#0:xs#0040040:mx#0:sf:sb:\
        :if=/usr/local/lib/ps/psif:\
        :of=/usr/local/lib/ps/psof:gf=/usr/local/lib/ps/psgf:\
        :nf=/usr/local/lib/ps/psnf:tf=/usr/local/lib/ps/pstf:\
        :rf=/usr/local/lib/ps/psrf:vf=/usr/local/lib/ps/psvf:\
        :cf=/usr/local/lib/ps/pscf:df=/usr/local/lib/ps/psdf:
# LaserWriter Printer
lw|LaserWriter:\
        :lp=/dev/ttya:br#9600:ms=-parity,onlcr,ixon,decctlq:\
        :sd=/var/spool/lw:lf=/var/spool/lp-log:mx#0:
```

The various filters specified in the **printcap** file include:

if	plain-text jobs input filter
of	output (banner) filter
rf	FORTRAN style text files filter
tr	troff data filter
nf	ditroff data (device independent troff) filter
df	TeX data (DVI format) filter
cf	cifplot data filter
gf	graph data filter
vf	raster image filter

The **server** controls who may print on it's printer by the entries in **/etc/hosts.equiv** and **/etc/hosts.lpd**. The former regulates the remote shell commands also, the latter regulates only printing access.

25.1.1.2 Spool Directory

In the above example the spool directory is **/usr/spool/lp**. Files concerning the line printer setup are in this directory. It contains the lock file, the log file, and the status of the printer. The file to be printed is first copied here and deleted after the printing is complete.

25.1.1.3 Accounting File

An accounting file that records the number of pages printed for each job, and who requested the print is kept, as specified by **/etc/printcap**. In our example that file is **/var/spool/lp.acct**.

25.1.2 User Commands

The LP spooler uses the commands: lpr, lprm, and lpq to submit jobs, remove jobs, and query the job queue, respectively.

25.1.2.1 lpr

lpr submits the print job to the spool area for printing. Use the "***-Pprinter*** " option to specify a particular printer defined in **/etc/printcap** other than the default, ***lp***, entry. ***lpr*** will default to the **PRINTER** environment variable, if it's set.

25.1.2.2 lprm

lprm removes jobs from the spool queue. This must be invoked by the user who submitted the job, or by the superuser.

25.1.2.3 lpq

lpq displays the list of files in the spool queue. Again you can use the "***-Pprinter*** " option to specify a particular printer defined in **/etc/printcap** other than the default, ***lp***, entry.

25.1.3 Line Printer Daemon, lpd

The line printer daemon, ***lpd***, is started up in **/etc/rc** and creates a lock file, **/var/spool/lpd.lock**, to prevent two copies of the daemon from running simultaneously.

The relevant lines in **/etc/rc** will be similar to:

```
if [ -f /usr/lib/lpd ]; then
    rm -f /dev/printer /var/spool/lpd.lock
    /usr/lib/lpd;     (echo -n ' printer')     >/dev/console
fi
```

lpd reads the **/etc/printcap** file to learn about existing printers and accepts print requests from users.

25.1.4 The printer control program, lpc

lpc controls the printers described in **/etc/printcap**. It's an interactive command that can be used to start/stop a printer, enable/disable spooling for a printer, rearrange jobs in the spool queue, display the status of each printer and their spool queues and printer daemon.

25.2 SunOS 5.X

SunOS 5.X uses the System V print service, as does HP-UX, which is considerably different from the SunOS 4.X/BSD version. It offers more power and flexibility, has additional commands, and is a little more complicated to set up. This setup can be considerably simplified by using the **Print Manager Facility** of ***admintool***.

The compatibility package for SunOS 5.X provides the SunOS 4.X print commands, but they actually just forward the request to the new print service. The **/etc/printcap** file and the ***lpd*** daemon are no longer used. The printer capabilities are now defined within the ***terminfo*** database, and locations are defined within the print service configuration files. The new print service includes a large set of administrative and user commands and a new set of daemons.

The new print service can interoperate with both System V and BSD printers, it has PostScript filters bundled in, it supports alternate character sets and more flexible job scheduling. Additionally, it can group similar printers into a class and can restrict printer access for individual users.

When a print request is received, the file is not spooled by default to the queue unless the "***-c***" options is given. The service detects the format of the job by the filtering software, and if necessary, the file's contents are converted to match the printer. The service keeps track of every job submitted and allows the user and system administrator to move, stop, or remove the job. When problems occur the service provides the system administrator with the error message. The print service coordinates both local and remote printers.

The print service can distinguish between a printer and a destination for the request. Previously these terms were synonymous. This allows you to group similar printers into a **class**, so that the request can be forwarded to any available printer within the class.

The terms used by the new print service are:

- **Printer** name assigned to the device, maximum of 14 characters
- **Class** name assigned to a group of similar printers
- **Destination** target for the print request, either a class or an individual printer

25.2.1 Print Scheduler

The print scheduler is started when entering run level 2 by the **/etc/rc2.d/S80lp** script. All this script does is **start** the print scheduler, ***/usr/lib/lpsched***, or **stop** it with the ***/usr/lib/lpshut*** command. The scheduler manages the print requests and must be running for the print service to operate. It identifies the filter for any necessary conversion and queues the file for the printer. It runs the interface program to initialize a local printer and downloads the request when the printer is ready.

Each print client and server must have at least one ***lpNet*** daemon running. The ***lpNet*** daemon is started by ***lpsched*** to handle network print requests. The ***lpNet*** daemon requires a port monitor be configured by the Service Access Facility, so that a registered listen service is available to handle incoming network requests for each local printer.

25.2.2 Print Filtering

Every print request is examined for the content type, some of which are:

PS
simple
tex
troff
raster

The print job needs to match the content type of the printer, so it is necessary that every printer be associated with at least one content type. The system can use the content type to match a job to a particular printer.

The service uses the ***/usr/sbin/lpfilter*** command to call the filter that will convert the contents of a file to that accepted by the target printer. This same ***lpfilter*** command is used to register new filters with the print service.

25.2.3 Printer Initialization

The descriptive file for the printers use the **terminfo** database in **/usr/share/lib/terminfo** with the file in the subdirectory beginning with the first character of the printer name. So for a PostScript printer named PS the description file is /usr/share/lib/terminfo/P/PS. Using ***infocmp*** we can examine the contents of this file:

```
# infocmp PS
#       Reconstructed via infocmp from file: /usr/share/lib/terminfo/P/PS
```

PS|PSR|PS-b|PS-r|PS-br|Fake PostScript entry,
cols#80, lines#66,
cpi=null, csnm=^D, lpi=null, scs=^D, slines=^D, u9=^D,

The interface programs to initialize local printers are found in **/usr/lib/lp/model**. A standard initialization script is supplied called **standard**, which takes its initialization information from the **terminfo** database. This program initializes the printer port, uses ***stty*** to configure the line settings, sends the appropriate control sequences to the printer, and sets printer parameters such as whether or not to print a banner page.

25.2.4 Printer Configuration

The printer control commands are located in **/usr/lib**. So you should have this directory in your path. The print administrative commands available to you are:

- ***lpadmin*** configure the print service
- ***lpfilter*** administer the filters for the print service
- ***lpforms*** manage the paper forms for the print service
- ***lpmove*** move print requests to another print destination
- ***lpsched*** start the print service
- ***lpshut*** stop the print service
- ***lpsystem*** register remote printers with the print service
- ***lpusers*** set print queue priorities for jobs submitted by a user

25.2.4.1 Installing a Local Printer

To configure a local printer perform the following steps.

1. Change the ownership and set the permissions on the serial port:
   # chown lp /dev/term/a
   # chmod 600 /dev/term/a
2. Add the printer and associate it with a port
   # lpadmin -p printer_name -v /dev/term/a
   where
   -p specifies the printer name, and
   -v specifies the device used by the printer
   This registers the printer name with the print service.
3. Associate a content type with the printer
   # lpadmin -p printer_name -I simple
   where
   -I specifies the content type.
   If you don't specify the content type **simple** is assumed, meaning that printer can only deal with ASCII contents.

4. Associate a printer type with the printer, if necessary. This is used by the interface program to initialize the printer before downloading a request
   # lpadmin -p printer_name -T proprinter
   where
   -T specifies the printer type
5. Allow the printer to accept requests and enable the queue
   # accept printer_name
   # enable printer_name

25.2.4.2 Installing a Local PostScript Printer

To configure a local PostScript printer there are two changes to the above procedure. One is that the content and type are specified as PS, and the other is the installation of PostScript filters.

1. Change the ownership and set the permissions on the serial port:
   # chown lp /dev/term/a
   # chmod 600 /dev/term/a
2. Add the printer and associate it with a port
   # lpadmin -p printer_name -v /dev/term/a
3. Associate a content and printer types with the printer
   # lpadmin -p printer_name -I PS -T PS
4. Register the PostScript filters with ***lpfilter***
   # cd /etc/lp/fd
   # lpfilter -f download -F download.fd
   # lpfilter -f dpost -F dpost.fd
   # lpfilter -f postdaist -F postdaisy.fd
   # lpfilter -f postdmd -F postdmd.fd
   # lpfilter -f postio -F postio.fd
   # lpfilter -f postior -F postior.fd
   # lpfilter -f postmd -F postmd.fd
   # lpfilter -f postplot -F postplot.fd
   # lpfilter -f postprint -F postprint.fd
   # lpfilter -f postreverse -F postreverse.fd
   # lpfilter -f posttek -F posttek.fd
5. Allow the printer to accept requests and enable the queue
   # accept printer_name
   # enable printer_name

25.2.4.3 Removing a Local Printer

To remove a local printer from the print service do the following.

1. Suspend the queue from accepting new requests
   # reject -r “printer printer_name is down” printer_name

where

-r indicates a reason for removing the printer (displayed by ***lpstat***)

2. Stop printing by disabling the printer
   # disable -W -r "printer printer_name is down" printer_name
   where some options to ***disable*** are:
   -W specifies to wait until the current request if finished printing
   -c specifies to cancel the current printing request
   -r indicates a reason for removing the printer
3. Remove the printer from the print service
   # lpadmin -x printer_name

25.2.4.4 Installing a Remote Printer

To install a remote printer you need to register the remote host with the print service on both the clients and server and configure the network listener on the print server. The file **hosts.lpd** is no longer used. Use the ***lpsystem*** command to register the print clients with the print service. ***lpsystem*** inserts a one line entry in **/etc/lp/Systems** describing the service.

SunOS 4.X/BSD Clients to Solaris 2 Server

1. Register the service
   # lpsystem -t bsd print_server_name
   where
   -t specifies the remote system type, either ***s5*** or ***bsd***
2. Create an instance of the ***listen*** port monitor to monitor the network for print requests
   # sacadm -a -p tcp -t listen -c "/usr/lib/saf/listen tcp" -v 'nlsadmin -V'
3. Obtain the print server's universal ***address*** in hex
   # lpsystem -A
   00020203809274040000000000000000
   where the first four digits, ***0002***, represent the internet protocol family, the next four, either ***0203*** or ***0ACE***, represent the BSD printer port (515 -> 203_{16}) or System V listen port (2766-> $0ACE_{16}$), respectively, the next 8 digits, ***80927404***, represent the hex IP address of the server (replaced by zeroes in later versions of SunOS 5.X), and the final 16 zeroes are padding.
4. Configure the ***listenBSD*** service to monitor incoming requests from BSD clients
   # pmadm -a -p tcp -s lpd -i root -v 'nlsadmin -V' -m 'nlsadmin -o /
   var/spool/lp/fifos/listenBSD -A "\00020203809274040000000000000000"'

Solaris 2/System V Clients to Solaris 2 Server

1. Register the service
   # lpsystem -t s5 print_server_name
2. Create an instance of the ***listen*** port monitor to monitor the network for print requests, if necessary. If you're already monitoring the network through the previous example you don't want to create a new listener.
   # sacadm -a -p tcp -t listen -c "/usr/lib/saf/listen tcp" -v 'nlsadmin -V'

3. Add the service to identify the STREAM used by the ***lp*** print service to receive connection requests
pmadm -a -p tcp -s lp -i root -v ‘nlsadmin -V‘ -m ‘nlsadmin -o /var/spool/lp/fifos/listenS5‘
4. Add the service ***0***, which is the ***nlps*** server, to use the System V listen address for print requests, ***0ACE***
pmadm -a -p tcp -s 0 -i root -v ‘nlsadmin -V‘ -m ‘nlsadmin -c /usr/lib/saf/nlps_server -A "\00020ACE809274040000000000000000"‘
5. Add the BSD ***lpd*** service to the print server. (This step appears to be a bug, which may change in future versions. There’s no obvious reason why the service should be required to listen at both ports inorder to work, but appears to be necessary in practice.)
pmadm -a -p tcp -s lpd -i root -v ‘nlsadmin -V‘ -m ‘nlsadmin -o /var/spool/lp/fifos/listenBSD -A "\000202038092740400000000000000000"‘

25.2.4.5 Installing a Print Client

Solaris 2.x Client to 4.X Server

For the SunOS 5.X print client to print to a remote BSD printer you don’t have to specify the content or printer type, nor have to register any filters. These features are presumed to be taken care of by the remote service.

1. Define the 4.X printer server as type ***bsd***
lpsystem -t bsd print_server_name
2. Define the printer name, using the same name as on the print server
lpadmin -p printer_name -s print_server_name
3. Define the printer type as unknown
lpadmin -p printer_name -T unknown -I any
4. Start the printer
accept printer_name
enable printer_name
5. Define a default printer (optional)
lpadmin -d printer_name

Solaris 2.X Client to Solaris 2.X Server

1. Define the Solaris 2 printer server as type ***s5***
lpsystem -t s5 print_server_name
2. Define the printer name, using the same name as on the print server
lpadmin -p printer_name -s print_server_name
3. Define the printer type as unknown
lpadmin -p printer_name -T unknown -I any
For a PostScript printer you should specify the parameters here, as in steps 3 and 4 of “**Installing a Local PostScript Printer**”, above.

4. Start the printer
   # accept printer_name
   # enable printer_name
5. Define a default printer (optional)
   # lpadmin -d printer_name

25.2.5 Print Commands

25.2.5.1 Print

The ***lp*** command is used to issue print requests. It's very similar to ***lpr***, but with a few differences in the options allowed. Some of the more frequently used options to lp are:

-c	make a **copy** of the file to the spool directory before printing
-d	select the **destination** printer or class of printers for the request
-m	send **mail** upon completion of the print job
-n number	specify the **number** of copies to be printed
-o option	printer dependent **options**, such as **nobanner**, **nofilebreak**, etc.
-t title	print **title** on the banner page
-w	**write** a message to the user's terminal after the file is printed

25.2.5.2 Status

To check the status of the printer and jobs submitted use the ***lpstat*** command. Some of the options available to this command are:

-a	report whether print destinations are **accepting** requests
-c	report names of all **classes** and their members
-d	report the system default **destination**
-o	report the status of **output** requests
-r	report the status of the LP scheduler
-R	report the position of the job in the queue
-s	print a status **summary**, including scheduler status, default destination, classes and printer known to the service, etc.
-t	report all status information (-s option plus the acceptance and idle/busy status of all the printers)

To report the status of the printers:

```
# lpstat -t
    scheduler is running
    system default destination: lp
    system for lp: tardis
    lp accepting requests since Tue Dec 22 11:10:02 EST 1992
    printer lp is idle. enabled since Tue Dec 22 11:10:02 EST 1992. available.
```

So to check if the scheduler is running:

```
# lpstat -r
    scheduler is running
```

With no options lpstat prints the status of all the user's print requests.

25.2.5.3 Cancel a Print Request

To cancel a print request use the ***cancel*** command. Only the user who submitted the request and the superuser can cancel a request. To cancel specify the printer and the job number:

```
# cancel lp-20
    request "lp-20" cancelled
```

To cancel all print requests:

```
# cancel -u frank lp
```

25.2.5.4 Move a Print Request

To move a request to another print queue use the ***lpmove*** command, specifying the printer and job number of the original request and new destination printer:

```
# lpmove lp-20 sparc
```

To move all jobs from one printer to another specify the old and new printer destinations:

```
# lpmove lp sparc
```

25.2.5.5 Controlling Access

The system administrator can set access restrictions on printers with the ***lpadmin*** command. Use the ***-u*** option to **allow** or **deny** access to individual users.

```
# lpadmin -p lp -u allow:frank,bobd,jeffs
# lpadmin -p lp -u deny:any,body,we,want,to
```

Set the default printer with the ***lpadmin*** command:

lpadmin -d *default_printer_name*

25.2.5.6 User Commands

The user commands are located in **/usr/bin**. The following table compares them with the SunOS 4.X commands.

TABLE 25.1 **User Commands**

SunOS 4.1.X	SunOS 5.X	Description
lpr	lp	submit a request to the printer
lpq	lpstat	report on the status of the print request and service
lprm	cancel	cancel a print request

25.2.5.7 Administrative Commands

Administrative commands are located in **/usr/lib**. Actually the files here are symbolic links to the actual files residing in **/usr/sbin** and **/usr/lib/lp**. The commands ***accept*** and ***reject*** are in **/usr/sbin**, and the commands ***enable*** and ***disable*** are in **/usr/bin**.

TABLE 25.2 **Administrative Commands**

SunOS 4.1.X	SunOS 5.X	Description
NA	accept	enable a destination (printer or class)
NA	reject	disable a destination (printer or class) from further requests
lpc enable	enable	enable the queue for the named printer
lpc disable	disable	disable the queue of the named printer for further requests
lpc	lpadmin	configure the print service
/etc/hosts.[equiv,lpd]	lpsystem	register remote hosts with the print service
NA	lpmove	move requests to new destinations
lpd	lpsched	start the print service
NA	lpshut	stop the print service
NA	lpusers	change user priority settings
NA	lpfilter	register filters for the print service

25.2.6 Configuration Files

The configuration files for the print service are kept in **/etc/lp** and the spooling directory is **/var/spool/lp**. These files are described in the following table.

TABLE 25.3 **Configuration Files and Directories**

File	Type	Description
/usr/lib/lp	directory	contains LP daemons, filters and interface programs
/etc/lp/Systems	file	list of remote hosts registered with the print service
/etc/lp/default	file	name of default destination
/etc/lp/fd	directory	contains filter description files
/etc/lp/filter.table	file	filter table
/etc/lp/logs	symlink	to /var/lp/logs, for the usage log
/etc/lp/printers	directory	contains a sub-directory for each printer
/etc/lp/printers/<name>/configuration	file	configuration for the printer <name>
/var/lp/logs	directory	log files for the print service
/var/spool/lp/SCHEDLOCK	file	lock file for lpsched
/var/spool/lp/system/pstatus	file	current status of print service
/var/spool/lp/tmp	directory	spool directory

25.3 IRIX 5.X

IRIX has both the BSD and SysV lineprinter packages along with the lineprinter driver package ***Impresario***.

25.4 Ultrix and Digital UNIX

Ultrix and Digital UNIX use the BSD ***lpr/lpd*** system.

CHAPTER 26 Mail

26.1 Send and receive electronic mail via SMTP, sendmail

When a mail program such as ***mail*** tries to send a message it issues a request to ***sendmail***, which processes the mail with the specified options. ***sendmail*** creates a list of recipients from the information and expands any aliases, including mailing lists. At this step syntax is checked and local addresses are verified. Duplicate recipients are removed, e.g. the same person being a member of two groups. If no addresses are valid the message is returned with an error message. sendmail then tries to deliver the message. If it can't deliver the message immediately it stores the header and body of the message in temporary files in a queue (**/var/spool/mqueue**) and tries to send it again later.

SMTP stands for Simple Mail Transfer Protocol, and is the protocol used for Internet mail. It requires an entry in **/etc/services**, i.e.:

```
smtp      25/tcp      mail
```

You can telnet to the mail port to see how you server is responding, i.e.:

```
# telnet localhost 25
```

It should respond with the fully qualified domain name (fqdn), otherwise, your machine may have trouble communicating with other mail servers.

26.2 Network mail configuration file

The **configuration** file used by sendmail is **/etc/sendmail.cf** (SunOS 4.1.X) or **/etc/mail/sendmail.cf** (SunOS 5.X). This file is read by sendmail when you start it up. It includes macros that define how header information is to be processed. Values in the header might be ignored, changed, or additional lines might be added to the header to assist in the delivery of mail. One of the important steps is the address re-writing rules. These rules search for patterns and replace them with specified strings. It also specifies the location of files and directories to be used by sendmail. These are generally, **/var/spool/mail** (SunOS 4.1.X) or **/usr/mail** (SunOS 5.X) for mail delivered to users on your machine, **/usr/spool/mqueue** for undelivered mail storage, and **/var/spool/mqueue/syslog** or **/var/log/syslog** for the mail log file.

Within **sendmail.cf** one macro might specify your Internet **subdomain**, e.g.:

```
## Change the D and E macros to accommodate your subdomains.
## Note that this configuration will do its very best to generate all addresses as coming from $D.
DDacs.ohio-state.edu
DEohio-state.edu
```

The "**D**" in the first column "**defines**" the following **D** and **E** as acs.ohio-state.edu and ohio-state.edu, respectively. These can be recalled later in the file as **$D** and **$E**.

Another macro might specify the style of a **header** line, e.g.:

```
## Pick one of the next 2 lines, based on your syntactic preference for
## "joe@host.domain (Joe Random)" or "Joe Random <joe@host.domain>".
#Dq$g$?x ($x)$.
Dq$?x$x $.<$g>
```

And still another might specify where to forward **BITNET** mail:

```
# Strange nonstandard nets - attempt to hand to forwarder.
R$+<@$+.bitnet>        $@$>0$1%$2.bitnet<@ohstbh.acs.ohio-state.edu>
```

The syntax of the re-writing rules is explained in the ***Sun AnswerBook Mail Administration Guide*** and in the book ***sendmail*** by Costales, et.al.

26.3 The mail alias file

You can use the mail alias file, **/etc/aliases** (SunOS 4.1.X) or **/etc/mail/aliases** (SunOS 5.X) to redirect mail or to send mail to a group of people. The ***newaliases*** command must be called to rebuild the aliases database before this information can be used by ***sendmail***. Some examples of entries in the **aliases** file are:

```
# Alias for mailer daemon; returned messages from our MAILER-DAEMON
# should be routed to our local Postmaster.
MAILER-DAEMON: postmaster
postmaster: frank
#
# A mailing list for a group
staff_group:     frank@nyssa,jim,bobd@leela,smith-b@magnus,
                 baker,bill@ohstmvsa,jones@epsilon.eng.ohio-state.edu
owner-staff_group: frank
#
# COSUG ACS Mailing List (aliases included in file)
cosug            ::include:/acs/tardis/0/frank/cosug/aliases
cosug-request    : frank
owner-cosug      : cosug-request
```

#

Pipe mail through a program

test: "| /usr/local/bin/testpgm"

The ***vacation*** program uses the latter form by placing in the user's **.forward** file contents similar to:

\frank, "|/usr/bin/vacation frank"

System mail **logs** are generally kept in **/var/spool/mqueue/syslog**, or **/var/log/syslog**, or **/var/adm/messages**, as determined by an entry in **/etc/syslog.conf**, e.g.:

mail.debug ifdef('LOGHOST', /var/spool/mqueue/syslog, @loghost)

Information is kept on every message sent, who sent it, to whom, the size, the status of the message, and the time, e.g.:

Jul 24 12:05:24 peri sendmail[1090]: MAA01090: from=<frank@peri.acs.ohio-state.edu>, size=555, class=0, pri=60555, nrcpts=2, msgid=<199607241605.MAA03311@nyssa.acs.ohio-state.edu>, proto=SMTP, relay=nyssa [128.146.226.22]

Jul 24 12:05:28 peri sendmail[1092]: MAA01090: to=<mgsmiley@magnus.acs.ohio-state.edu>, ctladdr=<frank@peri.acs.ohio-state.edu> (2523/11), delay=00:00:04, xdelay=00:00:04, mailer=tcp, relay=postbox.acs.ohio-state.edu. [128.146.214.20], stat=Deferred: Connection reset by peer during client greeting with postbox.acs.ohio-state.edu.

Jul 24 12:05:29 peri sendmail[1092]: MAA01090: to=<frank@magnus.acs.ohio-state.edu>, ctladdr=<frank@peri.acs.ohio-state.edu> (2523/11), delay=00:00:05, xdelay=00:00:01, mailer=tcp, relay=mail0.uts.ohio-state.edu. [128.146.214.29], stat=Sent (MAA26210 Message accepted for delivery)

Jul 24 12:10:50 peri sendmail[1107]: MAA01090: to=<mgsmiley@magnus.acs.ohio-state.edu>, ctladdr=<frank@peri.acs.ohio-state.edu> (2523/11), delay=00:05:26, xdelay=00:00:00, mailer=tcp, relay=postbox.acs.ohio-state.edu. [128.146.214.20], stat=Sent (MAA23783 Message accepted for delivery)

26.4 Installation of sendmail

1. Edit **sendmail.cf** to put in the desired values for your system.
   You can create the "fast" or "frozen" version of sendmail.cf, with the "***-bz***" option but you probably don't want to. If you do, it's installed in **sendmail.fc**. This will execute faster, though on today's CPUs this isn't that much of an advantage anymore. sendmail.cf will be ignored when sendmail.fc exists.
2. Kill and restart the ***sendmail*** daemon. Use "***ps***" to determine the process ID of sendmail.
   # kill #PID#
   # /usr/lib/sendmail -bd -q1h
   This starts the ***daemon*** (***-bd***) and requests that it process messages in the **queue** every 1 hour (***-q1h***).

To check the list of messages in the queue use:

% /usr/lib/sendmail -bp or mailq.

You can test the address rewriting rules of the sendmail.cf file by running sendmail in test mode, e.g., to see how the mail server, will treat the address, frank@magnus, going through ruleset 0:

```
% /usr/lib/sendmail -bt
ADDRESS TEST MODE (ruleset 3 NOT automatically invoked)
Enter <ruleset> <address>
> 0 frank@magnus
rewrite: ruleset  0   input: frank @ magnus
rewrite: ruleset 98   input: frank @ magnus
rewrite: ruleset 98 returns: frank @ magnus
rewrite: ruleset 97   input: frank @ magnus
rewrite: ruleset  3   input: frank @ magnus
rewrite: ruleset 96   input: frank < @ magnus >
rewrite: ruleset 96 returns: frank < @ magnus . acs . ohio-state . edu . >
rewrite: ruleset  3 returns: frank < @ magnus . acs . ohio-state . edu . >
rewrite: ruleset  0   input: frank < @ magnus . acs . ohio-state . edu . >
rewrite: ruleset 98   input: frank < @ magnus . acs . ohio-state . edu . >
rewrite: ruleset 98 returns: frank < @ magnus . acs . ohio-state . edu . >
rewrite: ruleset 95   input: < > frank < @ magnus . acs . ohio-state . edu . >
rewrite: ruleset 95 returns: frank < @ magnus . acs . ohio-state . edu . >
rewrite: ruleset  0 returns: $# smtp $@ magnus . acs . ohio-state . edu . $: frank < @ magnus . acs .
      ohio-state . edu . >
rewrite: ruleset 97 returns: $# smtp $@ magnus . acs . ohio-state . edu . $: frank < @ magnus . acs .
      ohio-state . edu . >
rewrite: ruleset  0 returns: $# smtp $@ magnus . acs . ohio-state . edu . $: frank < @ magnus . acs .
      ohio-state . edu . >
```

26.5 Security

Sendmail is a very complicated package and over the years a number of intentional and unintentional security holes have been found in it. If you are running with the vendor supplied sendmail, you most likely have an insecure version. This could result in someone cracking your system and gaining root access. Your vendor probably has a patched version available. Make sure you're running with the patched version, or the latest BSD version. The latest BSD version is at least 8.9.1. You can get this via anonymous ftp from: ftp://ftp.sendmail.org/pub/sendmail/.

26.6 Mail programs, mail, Mail, Columbia mm, elm, etc.

These and other programs are used to read and send mail messages. Generally when invoked they bring the user's mail from the system mail spool, **/var/spool/mail/*username*** (SunOS 4.1.X) or **/usr/mail/*username*** (SunOS 5.X) and put it into a mailbox in the individuals home directory, usually named **mbox**. They can also read mail from mbox and other named files. Startup files are **.mailrc** for ***mail*** and ***mailtool***, **.mmrc** for Columbia ***mm***, and **.elm/elmrc** for ***elm***. These customize the mail environment for the user. Should a user wish to have mail forwarded from this machine to another, the user can create the file, **.forward**, in their home directory containing the intended address, e.g.:

```
frank@magnus.acs.ohio-state.edu
```

This file should be readable by all and not have the execute bits set (e.g. mode 644).

CHAPTER 27 World Wide Web

27.1 WWW

The World Wide Web (WWW) uses the HyperText Markup Language (HTML), a subset of the Standard Generalized Markup Language (SGML), as one of its formats. This allows you publish your own multimedia documents on the network. The Internet protocol used is HyperText Transfer Protocol (http), which allow the client and server to negotiate the transfer representation of the document. We'll use this as an example of how to set up a complicated server on your workstation.

27.2 URLs

Uniform Resource Locators, URLs, reference where a resource can be found in the form:

service_scheme://machine_name[:port_number]/directory/sub-directory_list/file[?keyword]

where the items in brackets are optional. Some of the more common **service_schemes** are:

- **ftp** - file transfer protocol, a ";type=<type_code>" may be used to indicate the file type (e.g. an "**I**" for image, or an "**A**" for text)
- **http** - hypertext transfer protocol
- **gopher** - gopher protocol
- **news** - Usenet news via NNTP
- **telnet**, **rlogin**, or **tn3270**

The **service_scheme** could, optionally, include a user name and password, if required for the service.

The **machine_name** may be followed by a decimal **port_number**, separated from the machine_name by a colon, if something other than the default port number, 80, is used by the service.

The remainder of the address is the path, with subdirectories separated by "**/**" and ending with the desired file name or program. The latter may optionally be followed by a "**?**" and a **keyword** that can be used as an argument to the program. The specs for URLs can be found on:

http://www.w3c.org/Addressing/URL/Overview.html

27.3 WWW Server

You have many choices of free servers, but one stands out; the others that were once most popular have been discontinued because Apache is so good.

- **Apache** - Hypperreal's extensions to the original NCSA server.
  http://www.apache.org/
  This usually comes in source form, though binaries may be available.

27.4 WWW Browsers

There are four main browsers that you might be interested in:

- **Netscape** - from Netscape Communications, Corp., the second generation browser
  http://home.netscape.com/download/ - for a pre-compiled binary.
- **Mozilla** - the public development version of Netscape Navigator.
  http://www.mozilla.org/ - for source or binary.
- **HotJava** - from Sun, written in Java and requires Java to run
  Comes with Solaris 2.6+, or get it from http://java.sun.com/products/hotjava/.
- **Lynx** - a powerful text only browser
  http://lynx.browser.org/.

27.5 Setting up your Server

For our example we'll use the Apache httpd server. After retrieving the source from the reference above check out the README and src/INSTALL files for installation and configuration information.

To compile your own daemon uncompress and un-tar the source tree and edit the default control files in the conf directory: srm.conf, access.conf, httpd.conf. Then edit the src/Configuration file to select a compiler to use, options, and the modules to include. Some things you might change:

CC= cc	or	CC=gcc
CFLAGS= -DMAXIMUM_DNS DXBITHACK	or	CFLAGS= -O2
AUX_CFLAGS= -DSOLARIS2	or	AUX_CFLAGS= -DSUNOS4
AUX_LIBS= -lsocket -lnsl	or	AUX_LIBS=
Module agent_log_module		
Module referer_log_module		
Module config_log_module mod_log_config.o		

You can set the paths for the various services, control files, and log files in **httpd.h**. Some examples of entries you might change are:

```
#define HTTPD_ROOT "/usr/local/etc/httpd¨
```



```
#define DEFAULT_ADMIN "[no address given]"    -->             "webmaster"
#define DEFAULT_PORT 80
#define DEFAULT_XFERLOG "logs/access_log"
#define DEFAULT_INDEX "index.html"
#define ACCESS_CONFIG_FILE "conf/access.conf"
```

27.5.1 Compilation of the programs

To compile the http daemon, **httpd**, first go to the src subdirectory (**cd src**) and type "**./Configure**" and respond to any questions, then type “**make**”. It will default to using the **Makefile** in that directory. Then build any support files needed in the support directory (**cd ../support**) after editing the **Makefile** to select the desired compiler and programs. Then install the necessary programs in their desired directories. The full list of steps can be found at: ftp://wks.uts.ohio-state.edu/pub/solaris2/src/UTSinfo_apache-httpd-1.0.3, along with the source and compiled binaries in the file apache_1.0.3.tar.gz in the same directory.

WWW will support applications other than just display. There are a few sample auxiliary programs you can compile in the **cgi-src** directory and install in the **cgi-bin** directory, which already contains a few sample shell scripts. If you’re not going to support these services than you can ignore this step.

27.5.2 Configuration

For complete documentation on how to set up your server use **netscape** or **mosaic** to web to http://www.apache.org/. There you can find step-by-step instructions on how to configure the server.

The configuration file can be found in the **conf** directory. There is an example file you can use, **httpd.conf-dist**, to create your server configuration file, **httpd.conf**.

Some of the entries you’ll want to check out are:

ServerType **standalone** or **inetd**
Port 80
User http
Group http
ServerAdmin **you@your.address**
ServerRoot **/usr/local/httpd**

The latter determines the directory hierarchy for your service. It could have sub-directories such as: **cgi-bin**, **conf**, **htdocs**, **icons**, and **logs**.

This is a service you don’t want to run as root, so you should create a special user and group just for it. So in **/etc/passwd** you might have an entry similar to:

```
http:nologin:999:999:World Wide Web Server:/usr/local/http:/bin/false
```


and a **/etc/group** entry similar to:

http:*:999:frank

You can run your server either as a standalone server, in which case you would start it up in an RC script, or as a service controlled by ***inetd***. In the latter case you would need an entry in **/etc/services** similar to:

```
http        80/tcp        # WWW server
```

and another in **/etc/inetd.conf** similar to:

```
http stream tcp nowait http /usr/local/etc/httpd -d /usr/local/httpd -f /usr/local/httpd/conf/httpd.conf
```

where

-d	specifies the ServerRoot and where the daemon will look for it's configuration file (not necessary if you use the default ServerRoot path in the configuration file.)
-f	specifies the configuration file

To set it up as a standalone server you might put an entry similar to the following in an **RC** script, e.g. **/etc/rc.local** for SunOS 4.1.X:

```
if [ -f /usr/local/etc/httpd -a -d /usr/local/httpd -a -f /usr/local/httpd/conf/httpd.conf ]; then
   /usr/local/etc/httpd -d /usr/local/httpd -f /usr/local/httpd/conf/httpd.conf; echo -n ' httpd'
fi
```

For SunOS 5.X set up a script to start and stop the service as you go through run level 2.

Running ***httpd*** as a standalone daemon is more efficient, but running as a service of ***inetd*** provides greater access control. If you're using **TCPwrapper** you can specify which machines or subnets have access to your **http** service when each connection is controlled by ***inetd***.

27.6 Home Page

To complete your service you'll want to set up a Home Page on your server.

You'll need to know a little bit about HTML and the following primer will help you get started:

http://www.ncsa.uiuc.edu/General/Internet/WWW/HTMLPrimer.html

A good style guide can be found at:

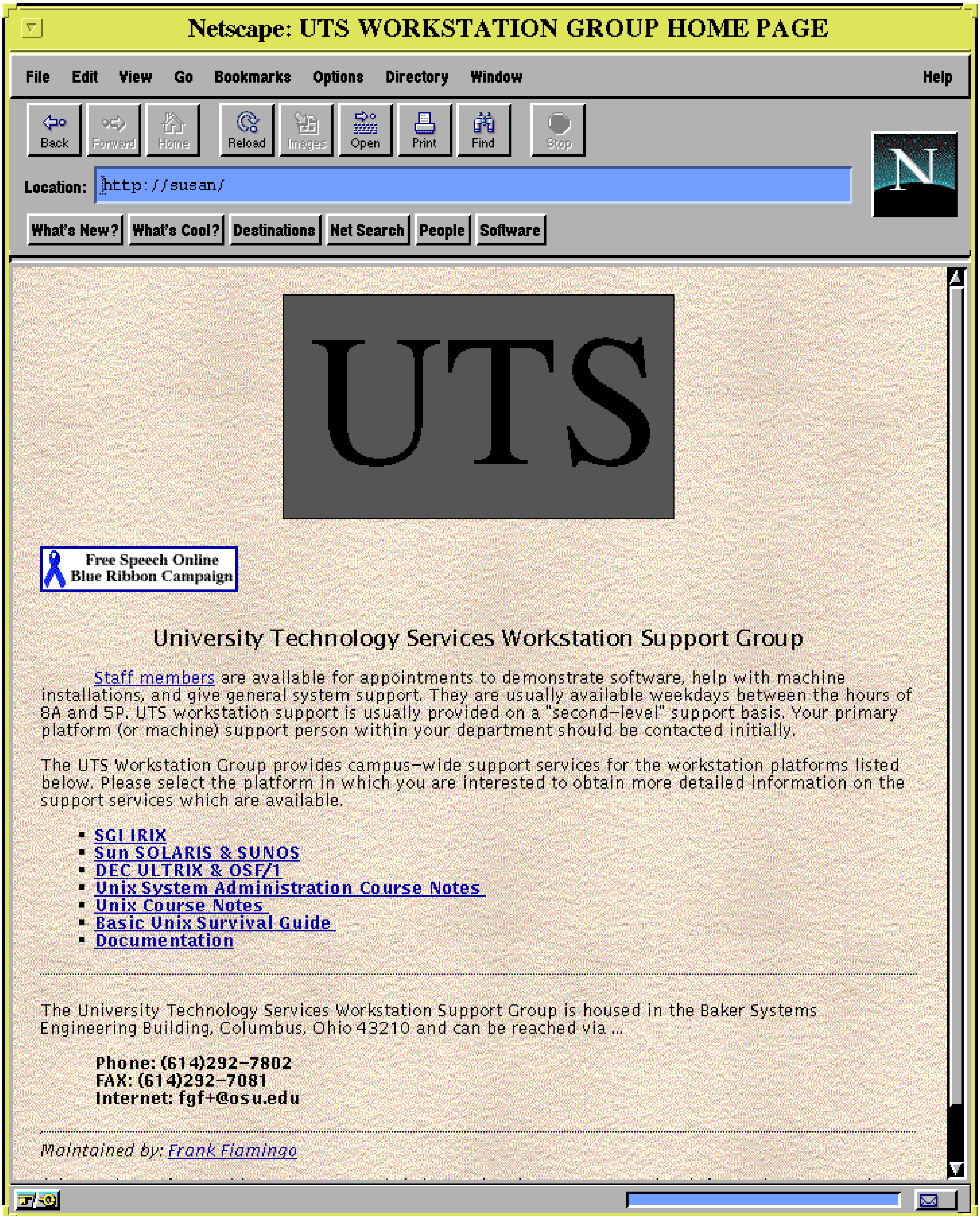
http://www.sun.com/styleguide/

A simple home page could be something like this one, which you could once find on http://wks.uts.ohio-state.edu, with the following HTML:

```
<HTML>
<HEAD>
<body background = "/icons/paper.jpg" link="#0000ac">
<TITLE>UTS WORKSTATION GROUP HOME PAGE</TITLE>
<!--Owner_Info="Frank Fiamingo, University Technology Services">
<LINK REV=MADE HREF="mailto:fgf+@osu.edu">
</HEAD>
<BODY>
<A NAME="loc0"></A>
<H2>
<CENTER><IMG SRC="/icons/uts_wks_logo.gif" ALT=""></CENTER>
<P>
<P>
<A HREF="http://www.eff.org/blueribbon.html"><IMG SRC="/icons/rib_bar_wh.gif"></A>
<P>
<P>
<A NAME="loc1"></A>
<CENTER>University Technology Services Workstation Support Group</CENTER></H2>
<P> <A NAME ="loc2"></A><DD><A HREF="/uts_wks_people.html">Staff members</A> are
    available for appointments to demonstrate software, help
with machine installations, and give general system support. They are usually
available weekdays between the hours of 8A and 5P. UTS workstation support is
usually provided on a "second-level" support basis. Your primary
platform (or machine) support person within your department should be
contacted initially.

<P>The UTS Workstation Group provides campus-wide support services for
the workstation platforms listed below. Please select the platform in
which you are interested to obtain more detailed information on the support
services which are available.

<UL>
<LI>     <B><A HREF="http://araminta.acs.ohio-state.edu/ats_wks_sgi_home.html"> SGI
    IRIX</A></B></LI>
<LI>     <B><A HREF="/sun/home.html"> Sun SOLARIS & SUNOS</A></B></LI>
<LI>     <B><A HREF="http://axpjob.acs.ohio-state.edu">DEC ULTRIX & OSF/1</A></B></LI>
<LI>     <B><A HREF="/sysadm_course/sysadm.html">
   Unix System Administration Course Notes </A></B></LI>
<LI>     <B><A HREF="/unix_course/unix.html">
   Unix Course Notes </A></B></LI>
<LI>     <B><A HREF="/basic_unix_guide/unix_guide.html">
   Basic Unix Survival Guide </A></B></LI>
<LI>     <B><A HREF="inventory.html">Documentation</A></B></LI>

</UL>
<HR>
<BR>
<P>The University Technology Services Workstation Support Group is housed
```

```
in the Baker Systems Engineering Building, Columbus, Ohio 43210
and can be reached via ...
<P>
<B><DL COMPACT>
<DD>Phone: (614)292-7802
<DD>FAX: (614)292-7081
<DD>Internet: fgf+@osu.edu
</DL></B>
<A NAME="loc6"></A>
<HR>
<ADDRESS>Maintained by: <A HREF="/uts_wks_frank.html">Frank Fiamingo</A></ADDRESS>
<P>
<ADDRESS>(The services referenced here are constantly being updated.
For more complete information contact the author.) </ADDRESS>
</BODY>
</HTML>
```

When you find a page you like on the Web you can use your browser to display or save the HTML for the page, allowing you to learn from the examples you view. When viewed by **Netscape** this page looks like:


Netscape: UTS WORKSTATION GROUP HOME PAGE
File Edit View Go Bookmarks Options Directory Window Help
Back Forward Home Reload Images Open Print Find Stop
Location: http://susan/
What's New? What's Cool? Destinations Net Search People Software
UTS
Free Speech Online
Blue Ribbon Campaign
University Technology Services Workstation Support Group
Staff members are available for appointments to demonstrate software, help with machine installations, and give general system support. They are usually available weekdays between the hours of 8A and 5P. UTS workstation support is usually provided on a "second-level" support basis. Your primary platform (or machine) support person within your department should be contacted initially.
The UTS Workstation Group provides campus-wide support services for the workstation platforms listed below. Please select the platform in which you are interested to obtain more detailed information on the support services which are available.
SGI IRIX
Sun SOLARIS & SUNOS
DEC ULTRIX & OSF/1
Unix System Administration Course Notes
Unix Course Notes
Basic Unix Survival Guide
Documentation
The University Technology Services Workstation Support Group is housed in the Baker Systems Engineering Building, Columbus, Ohio 43210 and can be reached via ...
Phone: (614)292-7802
FAX: (614)292-7081
Internet: fgf+@osu.edu
Maintained by: Frank Flamingo

CHAPTER 28 System Security

28.1 Security Concerns

No system can be made completely secure and usable at the same time. So you have to balance your security concerns against your computational needs.

You may decide that security is not a big concern at your site, but you can't ignore it completely. The information you keep on your system probably has some value to you. At the very least you usually don't want it altered or destroyed. If for no other reason, you need some security just to protect your good name. You wouldn't want some malicious hacker to break into your account and send thousands of hateful messages to every newsgroup in existence. Another reason to secure your system is to prevent it's use as a staging ground for attacking other systems on the network. You could, conceivably, be liable for damages.

Security is a shared responsibility. Every user on the system is capable of compromising security. They need to chose good passwords, change them periodically, and not share them. Teach them to report to you any suspicious activity, e.g. does the ***lastlogin*** reported match the last time they logged in? are there any files in their directory that they didn't put there?, etc.

Outside hackers are not your biggest security problem. Your highest risks to your data are from bugs and errors in the OS and from disasters. So you need to make sure that you keep good backups. Can you restore your system completely from backups? If your tapedrive fails, can you read your tapes on another drive?

You should analyze your system so that you know what you're protecting, why you're protecting it, what value it has, who has responsibility for it. Then you can plan your security needs accordingly.

Create a simple, generic policy for your system that your users can readily understand and follow. It should protect the data you're safeguarding, as well as, the privacy of the users. Some things it might include are: who has access to the system, who's allowed to install software on the system, who owns the data, disaster recovery, and appropriate use of the system.

28.2 What needs to be Secured?

You need to secure wherever your data is stored, transmitted, or accessed. This would include:

- **Disks** on the machine
- **Tape** backups
- **Network** connections
- **Serial** connections - modems, terminals, etc.

You should be concerned not just with loss or theft, or alteration of data, but also with loss of services.

If your machine has extremely sensitive data it shouldn't be on an outside network.

It may be that you'll need to isolate your site with a firewall. Should you need to do this check out the firewall books listed in Chapter 1.

28.3 Security Programs

There are a number of PD programs you can get to help make your system more secure. Some packages you might consider installing are:

- **COPS** - checks system service and file access privileges
- **TCP Wrapper** or **Xinetd** - checks network service connections for access privileges
- **Tripwire** - maintains a checklist and signature for files in it's database to detect changes in these files
- **Tiger** - checks system and file permissions, including anonymous ftp (more up-to-date than COPS)
- **Securelib** - secures UDP and RPC connections
- **lsof** - list open files on your machine
- **Swatch** or **Watcher** - for active audit trail watching
- **Crack** - check password against dictionaries and simple algorithms
- **PEM** or **PGP** - for mail and file security and content verification
- **SATAN** - Security Analysis Network Tool for Auditing Networks, checks for commonly known network security holes
- **SSH** - Secure SHell, replaces rlogin, rsh, and rcp with secure, encrypted, connections

For any program of this type you need to make sure that you protect the programs and databases from tampering. It doesn't help if, e.g. with **Tripwire**, you compare an altered file against an altered database. The best way to prevent tampering is to store the master copies on a physically write-protected disk or off-line.

You might have logs sent to another machine, so that they can't be altered on this machine.

Many of these programs are archived on the **COAST** (Computer Operations, Audit, and Security Technology) archive at Purdue University, **ftp://coast.cs.purdue.edu/pub**, under the direction of Prof. Gene Spafford. Some can be found local to OSU on **ftp://ftp.net.ohio-state.edu/pub/security**.

28.4 Security Response Teams

Locally, at tOSU, subscribe to **netwog-security-request@cicada.net.ohio-state.edu**. This mailing list will alert you to any new security concerns expressed by the following organizations, and others.

CERT - Computer Emergency Response Team at Carnegie Mellon University, cert@cert.org.

CIAC - Computer Incident Advisory Capability, for DOE contractors, ciac-notes-request@llnl.gov.

FIRST - Forum of Incident Response and Security Teams, first-request@first.org, to get on their mailing list or check out http://www.first.org.

28.5 The password and group files

The **/etc/passwd**, **/etc/group**, and **/etc/shadow** files should be writable only by **root**. Any entry in /etc/passwd that has a uid of "**0**" (zero) is a **ROOT** entry, regardless of the name by which it is called. SunOS 4.1.X doesn't require you to set a root password when you install the OS. Make sure that you do set one. SunOS 5.X requires that you set a root password as the final step in SunInstall. Make sure that you set a good one.

Passwords should be chosen that are difficult to guess. A study done in 1978 showed that 16% of all passwords are 3 characters or less, and that 86% of chosen passwords could be described as insecure. A more recent study showed that simply trying 3 guesses on each account: the login name, login name in reverse, and the two concatenated, would obtain access to 8 - 30% of the accounts on a typical system.

Use a password that contains mixed case alphabetic characters and numbers. It should be 6 - 8 characters long to make the number of possible combinations extremely large. For 62 possible characters in each position (26 lower case + 26 upper case + 10 digits) there are 62^n possible combinations. This is 238328 for a 3 character password and $2.18*10^{14}$ for an 8 character password. In contrast, if you only use lower case letters there are 26^3, or 17576 combinations for a 3 character password and $2.09*10^{11}$ in an 8 character one.

Your password, though difficult to guess, should be easy to remember. If you have to write it down it's not secure. A study by Daniel V. Klein reported in his paper, ***Foiling the Cracker: A Survey of, and Improvements, to Password Security*****,** (available from ftp://www-wls.acs.ohio-state.edu:/pub/security/Dan_Klein_password_security.ps.Z) emphasizes the poor choice of passwords found on many systems. The following table is from this paper regarding the passwords cracked from a sample set of 13,797 accounts solicited from the Internet.

TABLE 28.1 **Passwords Cracked**

Type of Password	Size of Dictionary	Duplicates Eliminated	Search Size	# of Matches	Pct. of Total	Cost/Benefit Ratio[a]
User/account name	130[b]	-	130	368	2.7%	2.830
Character sequences	866	0	866	22	0.2%	0.025
Numbers	450	23	427	9	0.1%	0.021
Chinese	398	6	392	56	0.4%[c]	0.143
Place names	665	37	628	82	0.6%	0.131
Common names	2268	29	2239	548	4.0%	0.245
Female names	4955	675	4280	161	1.2%	0.038
Male names	3901	1035	2866	140	1.0%	0.049
Uncommon names	5559	604	955	130	0.9%	0.026
Myths & legends	1357	111	1246	66	0.5%	0.053
Shakespearean	650	177	473	11	0.1%	0.023
Sports terms	247	9	238	32	0.2%	0.134
Science fiction	772	81	691	59	0.4%	0.085
Movies and actors	118	19	99	12	0.1%	0.121
Cartoons	133	41	92	9	0.1%	0.098
Famous people	509	219	290	55	0.4%	0.190
Phrases and patterns	998	65	933	253	1.8%	0.271
Surnames	160	127	33	9	0.1%	0.273
Biology	59	1	58	1	0.0%	0.017
/usr/dict/words	24474	4791	19683	1027	7.4%	0.052
Machine names	12983	3965	9018	132	1.0%	0.015
Mnemonics	14	0	14	2	0.0%	0.143
King James bible	13062	5537	7525	3	0.6%	0.011
Miscellaneous words	8146	4934	3212	54	0.4%	0.017
Yiddish words	69	13	56	0	0.0%	0.000
Asteroids	3459	1052	2407	19	0.1%	0.007
Total	86280	23553	62727	3340	24.2%	0.053

a. In all cases, the cost/benefit ratio is the number of matches divided by the search size. The more words that needed to be tested for a match, the lower the cost/benefit ratio.
b. The dictionary used for user/account name checks naturally changed for each user. Up to 130 different permutations were tried for each.
c. While monosyllabic Chinese passwords were tried for all users (with 12 matches), polysyllabic Chinese passwords were tried only for users with Chinese names. The percentage of matches for this subset of users is 8% - a greater hit ratio than any other method. Because the dictionary size is over $16x10^6$, though, the cost/benefit ratio is infinitesimal.

28.6 File and Directory Permissions

Use the ***chmod***, ***chgrp***, and ***chown*** commands to set the correct file and directory permissions.

Shell scripts should NOT be run **setuid** or **setgid**. Use ***find*** to search your directories for setuid/setgid files, e.g.:

```
find / -type f -a \( -perm -4000 -o -perm -2000 \) -print
```

where ***find*** looks for any regular file (***-type f***) that also (***-a*** = and) has either permission bits set for **setuid** (4000) or (***-o***) **setgid** (2000), and prints the names of those found. When doing a long listing (***ls -al***) file permissions will look like:

Octal	**Owner/Group/Other**
755	rwxr-xr-x
4755	rw**s**r-xr-x
2755	rwxr-**s**r-x
644	rw-r--r--
4644	rw**S**r--r--
2644	rw-r-**S**r--

In this listing the **s** and **S** indicate setuid/setgid permissions.

28.7 EEPROM Security

On ***Sun*** workstations and servers you can interact with the boot EEPROM (NVRAM) at any time by holding down the **STOP** (L1) key and pressing the "**a**" key. If you're using a dumb terminal as the console the "**break**" key has the same effect. You can remove this feature from the kernel, but otherwise, it's there for anyone to use or abuse. This chip stores the configuration information for the machine, including the hostid and the ethernet address.

Mark Henderson's **change-sun-hostid** package provides a lot of useful information about Sun NVRAMs, including how to change the hostid and how to recover should the NVRAM battery fail. It can be found at: http://www.squirrel.com/squirrel.

Using **STOP-A**, or **break**, anyone can interrupt your machine and reboot from CDROM or ***their*** disk, and have complete access to your files. To help prevent this you should **password protect** your EEPROM. You are allowed 3 levels of EEPROM security, **none-secure**, **command-secure**, and **fully-secure**. The first one is the default, i.e. no security. Anyone can issue any command at the EEPROM prompt. With **command-secure** a password would have to be used to boot from anything other than the default device. The most secure is **fully-secure**, where the password has to be supplied to boot in all cases. The EEPROM password is different from the OS password. Should you forget your EEPROM password you won't be able to change it unless you have access to the running system; from there you can use the ***eeprom*** command to reset any EEPROM parameters. So whatever you choose for this password, make sure it's easy to remember or you might just lock yourself out of your machine. In which case, you might have to buy a new EEPROM (which in some cases involves swapping the CPU).

28.8 Secure the console port

28.8.1 SunOS 4.1.X

Root can only login to ports labeled ***secure*** in **/etc/ttytab**. Unless your console is in a locked room all ports should be labeled ***unsecure***. This will require you to first login as yourself and then ***su*** to root. It also requires that the root password be entered when booting in single user mode from the disk.

28.8.2 SunOS 5.X

SunOS 5.X requires the root password whenever you enter single user mode, both when booting, and when using init to move to single user run levels.

SunOS 5.X has the **/etc/default** directory which contains files that set the default policies for the system. They specify whether to allow remote root logins, what the minimum password length should be, whether to create an su log file, etc.

28.8.2.1 /etc/default/login

This file specifies login policy. A typical file might contain:

```
HZ=100                  #
TIMEZONE=EST5EDT        # set the timezone variable for the shell
#ULIMIT=0               # set the file size limit for the shell, 0 -> no limit
CONSOLE=/dev/console    # root can only login on this device
PASSREQ=YES             # Null passwords are not allowed
ALTSHELL=YES            # set the shell environment variable
SYSLOG=YES              # log all root logins and multiple failed attempts
UMASK=022               # set the initial umask
```

To allow remote root logins comment out the **CONSOLE** entry. To prevent root logins everywhere, even the console, set the **CONSOLE** entry to "***=/dev/null***".

28.8.2.2 /etc/default/passwd

This file specifies the minimum password length and password aging restrictions.

```
MAXWEEKS=               # Length of time the password is valid
MINWEEKS=               # Minimum time between password changes
PASSLENGTH=6            # Minimum password length
```

28.8.2.3 /etc/default/su

This file specifies the notification procedure for when ***su*** is executed.

```
SULOG=/var/adm/sulog    # Log all su attempts to this file
#CONSOLE=/dev/console   # Log successful su attempts to the console
```


28.8.3 IRIX

/etc/default/login defines the console and whether or not root login is permitted, as with SunOS 5.X.

28.8.4 Ultrix

If the terminal is labelled "***secure***" in **/etc/ttys** root can login on that device.

28.8.5 Digital UNIX

/etc/securettys is used to specify which terminals will allow root logins. When Enhanced Security mode is enabled the file, **/etc/auth/system/ttys**, contains the terminal access database and keeps records of the last access to the terminals.

28.9 Security Loopholes

28.9.1 /etc/hosts.equiv

In SunOS 4.1.X this file is distributed with the contents "**+**", i.e. every host on the network is trusted. Any wildcard characters should be removed from this file. Use specific host names. If you're not going to have any trusted hosts just delete the file. If you are going to use it be careful. Entries such as:

```
machine_name  user_name
```

mean that user, ***user_name***, from ***machine_name*** can login as ***any*** user on your host. Also, contrary to the manual "**-**" acts as "**+**".

28.9.2 .rhosts

This file is similar to **/etc/hosts.equiv**, but for a specific user. Each user may create their own **.rhosts** file and allow the indicated account from another machine access to their login without a password. A **.rhosts** file in the root directory allows root access, which may occasionally be necessary for network backups.

28.9.3 /etc/exports

If no access is specified in **/etc/exports** for a file system, then every host has access to that file system. Avoid entries such as:

```
/home
```

28.9.4 NFS mounts

When mounting file systems via NFS, if you can't trust the system you're mounting from, always make sure you mount the file systems with the ***nosuid***, or don't mount it. This prevents anyone from running suid programs from those file systems.

```
# mount -o nosuid,bg,intr untrusted:/home /u_home
```

28.9.5 FTP

FTP is often used for anonymous login and sharing of files (e.g. archives). This should be done in a secure manner (see the Manual). Put an "*" in the password field of user **ftp**, do a change root to **~ftp**, and use a non-valid shell, e.g. ***/bin/false*** for the user **ftp**. You can limit password ftp access to your system with the **/etc/ftpusers** and **/etc/shells** files. If the user's name is in the **ftpusers** file access is denied. If the user's shell is not in the **shells** file access is denied.

28.9.6 Trivial FTP, TFTP

This is used to allow diskless workstations, X-terminals, and network routers to boot from servers without authentication. Again this should be done by using a change root to **/tftpboot.** The entry below in **/etc/inetd.conf** will do this.

```
tftp  dgram  udp  wait  root   /usr/etc/in.tftpd   in.tftpd -s /tftpboot
```

28.9.7 Mail

Remove the ***decode*** aliases from **/etc/aliases** (SunOS 4.1.X) and **/etc/mail/aliases** (SunOS 5.X). Should there be any other aliases that pipe programs through commands make sure that there is no way to obtain a shell or send commands to a shell from the alias. Make sure your sendmail doesn't support the debug command. Check this by telneting to your SMTP port and typing "***debug***".

28.9.8 PATH

Your executable path, and that of root should not contain "**.**", i.e. the present directory. It should only contain directories that are known to be secure. e.g. a PATH such as

```
PATH=.:/bin:/usr/bin:/usr/ucb
```

will first check in the present directory for the specified file. Should a user put an executable file in **/tmp** with a common name, e.g. "***ls***", typing "***ls***" when in **/tmp** will execute their command, ***/tmp/ls***.

Some people advocate putting "**.**" at the end of your PATH. That's not sufficient, especially if you're prone to typing mistakes, e.g. typing ***mroe*** instead of ***more*** will not be found in one of the system files, but a thoughtful cracker could have one lying in wait for you.

28.9.9 /etc/inetd.conf

This file controls access to many of the services on your system. Some of these services you may not want to provide access to. Remove or comment out entries to such services and then send ***inetd*** a hangup signal (***kill -HUP*** on the process) so that it will reread this file.

You could also install **TCPwrapper** so that you control which machines or networks can access individual services.

28.9.10 tmpfs, /tmp

When **tmpfs** is used **/tmp** is re-created after each reboot. Make sure that the sticky bit is set i.e.; the mode should be **1777**. The **sticky bit** must be set so that users can't change files they don't own.

28.9.11 /etc/utmp

Login accounting records are written in **/etc/utmp**. This file should NOT be writable by everyone, as it is commonly distributed, e.g. SunOS 4.1.X, this is often the case. Remove general write permission from this file by setting it to mode 644.

28.10 Additional Security Features in SunOS 5.X

SunOS 5.X includes a number of security features not present in SunOS 4.X. Some are set by default, others can be set using the Automated Security Enhancement Tool (**ASET**). Among the new features are:

- shadow password file, **/etc/shadow**
- **/etc/default** directory containing files that set system access security controls
- restricted shell, ***/usr/lib/rsh***
- **ASET**
- optional **Kerberos** support
- **Solstice AdminSuite** (***solstice***) security levels
- **password** is required when entering single-user mode

28.10.1Restricted Shell

Restricted shells allow you to control the user's environment. The restricted shell, ***rsh***, allows the user to do everything allowed by ***sh***, except:

- change directory
- set the value of $PATH
- specify the path of command names containing /
- redirect output (> and >>)

The ***restricted*** shell is ***/usr/lib/rsh***. This should not be confused with the ***remote*** shell, which is ***/usr/bin/rsh***.

Don't rely too heavily on the restricted shell. It's not that restricted. While you can't specify a command name that begins with "/", you can specify arguments that do. So if ***cat*** is in your path you could type:

```
% cat /etc/passwd
```

and have a look at the password file. Also, some programs, such as editors and ***telnet***, allow you to escape out to a shell and editors can edit/view any file with read access allowed on the system.

28.10.2 Automated Security Enhancement Tool

ASET allows you to monitor and restrict access to system files. It can be configured for three security levels: low, medium, and high.

At ***low*** level ASET doesn't modify any system files, but reports on potential security weaknesses.

At ***medium*** level some system files may be modified to restrict access. This should not affect system services. It will report on security weaknesses and changes performed.

At ***high*** level further restrictions are made to provide a secure system. System parameters are changed to provide minimal access. Most system applications should still work normally, but security is considered more important than applications at this level.

At the highest level the checks performed by ASET are:

- verify appropriate permissions for system files
- verify contents of system files
- check consistency and integrity of entries in passwd and group
- check contents of system configuration files
- check environment files: .profile, .cshrc, .login
- verify appropriate eeprom settings to restrict console login access
- disables IP packet forwarding so that the system can be used as a firewall or gateway machine

It checks files such as:

/etc/hosts.equiv	for "**+**" entries
/etc/inetd.conf	for ***tftp***, ***ps***, ***netstat***, and ***rexd*** entries
/etc/aliases	for the ***decode*** alias
/etc/default/login	for root access via the ***CONSOLE=*** entry
/etc/vfstab	for world-readable/writable file systems
/etc/dfs/dfstab	for files shared without restrictions
/etc/ftpusers	at high security places ***root*** in this file to disallow access for root
/var/adm/utmp	changes world-writable access at high security level
/var/adm/utmpx	"
/.rhosts	removes this for medium and high security levels

ASET uses the directory **/usr/aset** for its scripts and reports. Some of the scripts used to control ASET actions are tune.low, tune.medium, and tune.high in the **/usr/aset/masters** directory, which specify file ownership and permissions.

ASET requires the package **SUNWast** be installed on the system.

28.11 SRI Security Report

SRI International released (April 1990) a report on system security: ***Improving the Security of your UNIX System*, by David A. Curry**. This is available as **ftp://www-wks.acs.ohio-state.edu/pub/security/security-doc.tar.Z**. The final security checklist of this document, Appendix A is reproduced here.

SECURITY CHECKLIST

This checklist summarizes the information presented in the paper (***Improving the Security of your UNIX System***, by David A. Curry), and can be used to verify that you have implemented everything described.

Account Security

- [] Password policy developed and distributed to all users
- [] All passwords checked against obvious choices
- [] Expiration dates on all accounts
- [] No ‘‘idle’’ guest accounts
- [] All accounts have passwords or ‘‘*’’ in the password field
- [] No group accounts
- [] ‘‘+’’ lines in passwd and group checked if running Yellow Pages

Network Security

- [] hosts.equiv contains only local hosts, and no ‘‘+’’
- [] No .rhosts files in users’ home directories
- [] Only local hosts in ‘‘root’’ .rhosts file, if any
- [] Only ‘‘console’’ labeled as ‘‘secure’’ in ttytab (servers only)
- [] No terminals labeled as ‘‘secure’’ in ttytab (clients only)
- [] No NFS file systems exported to the world
- [] ftpd version later than December, 1988
- [] No ‘‘decode’’ alias in the aliases file
- [] No ‘‘wizard’’ password in sendmail.cf
- [] No ‘‘debug’’ command in sendmail
- [] fingerd version later than November 5, 1988
- [] Modems and terminal servers handle hangups correctly

File System Security

- [] No setuid or setgid shell scripts
- [] Check all ‘‘nonstandard’’ setuid and setgid programs for security
- [] Setuid bit removed from /usr/etc/restore
- [] Sticky bits set on world-writable directories
- [] Proper umask value on ‘‘root’’ account
- [] Proper modes on devices in /dev

Backups

- [] Level 0 dumps at least monthly
- [] Incremental dumps at least bi-weekly

28.12 CERT Security Advisories

Below is a truncated version of one of the more recent **CERT** advisories. All CERT advisories are available at **ftp://cert.org/pub/cert_advisories**.

28.12.1rdist Vulnerability

===

CERT(sm) Advisory CA-96.14

July 24, 1996

Topic: **Vulnerability in rdist**

This advisory supersedes CA-91:20.rdist.vulnerability and CA-94:04.SunOS.rdist.vulnerability.

- ---

The CERT Coordination Center has received reports that a new vulnerability in rdist has been found and an exploitation script is widely available. Current reports indicate that the script works on x86-based versions of the UNIX Operating System; however, we believe that it would not be difficult to write variants that work on other instruction sets and configurations.

The CERT/CC Staff recommends following the steps in Section III.A. to determine if your system is vulnerable and to disable vulnerable programs, then following your vendor's instructions (Section III.B and Appendix A). Until you can install a vendor patch, you may want to install a freely available version of rdist, noted in Section III.C.

As we receive additional information relating to this advisory, we will place it in

ftp://info.cert.org/pub/cert_advisories/CA-96.14.README

We encourage you to check our README files regularly for updates on advisories that relate to your site.

- ---

I. Description

The rdist program is a UNIX Operating System utility used to distribute files from one host to another. On most systems, rdist is installed as set-user-id root, a necessity due to its design. Unfortunately, this setting makes it a favorite target for vulnerability investigation.

A new vulnerability in rdist has been discovered and reported. The vulnerability lies in the lookup() subroutine where the value of a command line argument is used to overflow the subroutine call stack. If that argument is specially crafted with native machine code lookup() returns control to the code added to the call stack instead of the subroutine that called lookup(). If, for example, this added code uses a member of the exec system call family and names /bin/sh as the program to be executed, that shell is then run with set-user-id root privileges. No matter what code is added, the code runs with set-user-id root privileges.

An exploitation program, which is circulating on the Internet, take advantage of this vulnerability. While it purports to work only on x86-based versions of the UNIX Operating System, variants tuned to other instruction sets and configurations are straightforward to write.

II. Impact

On unpatched systems, anyone with access to a local account can gain root access.

III. Solution

We urge you to follow the steps in Section A to determine if your system is potentially vulnerable and, if it is, to turn off rdist while you decide how to proceed.

If you need the functionality that rdist provides, install a vendor patch (Sec. B). Until you can do so, you may want to install a freely available version of rdist that does not need to be installed as set-user-id root and is, therefore, not susceptible to the exploitation described in this advisory (Sec. C).

...

CHAPTER 29 Secure Shell, SSH

29.1 Secure SHell

Normal IP traffic has the following weaknesses that can be exploited to compromise security:

weak authentication	based on IP addresses that can be spoofed or reusable passwords that can be sniffed
no privacy	packets can be sniffed
no integrity protection	connections can be hijacked

Secure SHell (**SSH**) was designed to address these problems by providing a stronger authentication mechanism to identify both hosts and users and to enable secure connections between machines for executing commands and remote shells between them. It can be used to directly replace the functions of ***rsh***, ***rcp*** and ***rlogin***. It can also be used, in many cases, instead of **telnet** and **ftp** and to forward other connections, such as those between **X**, **pop** or **nntp** servers and clients.

The current method of communicating between machines allows anyone to sniff the packets on the network. Passwords and all data are sent along in plain text and can be readily captured and analyzed. Secure shell foils sniffing attempts by encrypting the packets (using ciphers) and by only allowing connections with known machines (using RSA public key technology to authenticate). In general, it never trusts the network. Should an attacker gain root access to your machine through another means, however, **SSH** can then be compromised also. The encryption method, and indeed whether or not encryption is even turned on, is a setable parameter. Make sure you choose the values that will properly protect your system.

SSH can be used to replace the ***rsh/rcp/rlogin*** programs, or to work with them. If you always want to have a secure connection, then replace them. If you want to allow connections to remote machines that don't have **SSH**, then let it work with them. If the remote machine doesn't support ***ssh*** it will then fall back to using the r-programs, after first informing the user that the communication will not encrypted.

More information about ***ssh*** can be obtained from the **SSH** home page **http://www.cs.hut.fi/ssh/** and from the documentation files that come with the source code. Ssh was developed by Tatu Ylonen at the Helsinki University of Technology, ylo@cs.hut.fi. There is an **SSH** mailing list. You can get information about how to subscribe to the list by sending mail to ssh-request@clinet.fi.

I'm writing this report as an aid to the novice administrator to install, configure, and make use of this unique security tool. A script for easy installation is included at the end of this Chapter.

29.1.1 Description of SSH

Secure SHell is designed to provide strong authentication and secure communications over what are normally insecure channels. It allows remote logons, remote execution of commands, and remote copies, acting as a direct replacement for ***rlogin***, ***rsh***, ***rcp***, and ***rdist***. It provides the following features:

• Strong authentication	SSH can use **.rhosts** together with RSA based host authentication, and pure RSA authentication.
• Improved privacy	Encryption of all communications are automatic and transparent. Key exchange is done with RSA. The session is encrypted with a cipher (IDEA, DES, or triple-DES). Encryption is started before authentication so that no passwords are ever sent in the clear.
• Secure X11 sessions	**DISPLAY** is automatically set on the server machine, forwarding any X11 connections over the secure channel.
• Port forwarding	Bi-directional redirection of arbitrary TCP/IP ports can be done through the encrypted channel.
• Automatic	Replace the insecure programs with secure ones and everything happens automatically for the users. Old **.rhosts** files will still be valid, but with strong authentication, if the system administrator installs host key files.
• Never trusts the network	With RSA authentication nothing but the private key is trusted.
• Prevents spoofing	The client and the server each use RSA to authenticate the other. The client authenticates the server at the start of each connection, and the server authenticates the client before it allows **.rhosts** or **/etc/hosts.equiv** access. This prevents DNS, routing, or IP-spoofing and man-in-the-middle attacks.
• Host authentication key	typically 1024 bits. These can be generated and distributed centrally and automatically or manually by each user for their own use. Both the central and per-user host keys are used.
• User authentication keys	typically 1024 bits. Each user can create any number of RSA user authentication keys for their own use. The public keys are stored in a private file. The user provides the private key to authenticate.
• Server key regeneration	The server regenerates its RSA key (normally 768 bits) automatically every hour (configurable) and never saves it in a file. Session keys are exchanged after encryption using both the server key and the server host key. This prevents capturing a session and deciphering it at a later time.
• Authentication agents	can hold the user's RSA authentication keys. These would typically be running on the user's laptop or local machine and there is no need to store the RSA authentication keys anywhere else. SSH automatically forwards the connection to the authentication agent, never revealing the keys. The protocols are only used to verify that the agent has a user's key.
• Customizable	The client has customizable configuration files, both system-wide and per-user. Different options can be specified for different hosts.
• *rsh* fallback	If the server machine is not running sshd a warning is displayed and then *ssh* automatically falls back to using conventional *rsh*.
• Compression	*gzip* compression of all data, including forwarded X11 and TCP/IP port data, is optional.

29.1.2 What SSH Does Not Do

SSH does not protect you from anyone having root access on your local machine or on the server machine. Root on either of these machines could monitor your session or replace programs with trojan horses. So basic security on the client and server machines still needs to be maintained.

29.2 SSH Programs

The **SSH** package includes the server program, r-program replacements, a program to generate and register the keys, and a ***perl*** script to probe and report the public keys of hosts on a network or DNS subdomain.

TABLE 29.1 **Ssh Programs**

Program	Description
sshd	Server program - listens for connections from client machines, authenticates the connection and starts the service
ssh	Client program - used to send remote commands (rsh replacement) of remotely login (rlogin replacement) to another machine
slogin	Symbolic link to ssh replacing rlogin
scp	Copy files to another machine (rcp replacement)
ssh-keygen	Create authentication keys for hosts and users
ssh-agent	Authentication agent - holds RSA authentication keys
ssh-add	Register new keys with the agent
make-ssh-known-hosts	Script to probe hosts on a network for their public keys. Used to populate /etc/ssh_known_hosts.

29.2.1 Usage

The user connects to other machines with commands similar to:

```
% ssh remote_host command
% ssh remote_host
% xterm -e ssh remote_host &
```

29.2.2 Debugging

To get debugging information you can run the server process with "**-d**" or the user process with "**-v**":

```
# sshd -d
% ssh -v host
```


29.3 Control Files

The following table lists the files used by **SSH** to hold information necessary to verify the host or user and to configure the connection. Not all of these files are necessarily used. This depends on the restrictions you specify for the server options.

TABLE 29.2 Files used by SSH

Directory	File	Usage
/etc	ssh_host_key	machine private key, accessible only by root
	ssh_host_key.pub	machine public key. This file has one line of the form: 1024 37 94512...(lots of numbers)...34891 root@this_machine
	ssh_random_seed	seed for the random number generator, accessible only by root
	ssh_known_hosts	system-wide known public host keys of machines. Public keys are put here, one per line, with a format similar to **~/.ssh/authorized_keys**: system name, number of bits in modulus, public exponent, modulus, and optional comment field, all separated by spaces. The system name can include aliases and IP addresses separated by commas, e.g. (all on one line) nyssa,nyssa.acs.ohio-state.edu,128.146.116.4 1024 41 50812...(lots of numbers)...72391 root@nyssa These can be obtained from the **/etc/ssh_host_key.pub** of each other host. If you have enabled StrictHostKeyChecking in **/etc/ssh_config** then you must manually add the desired host's public key to this file so that sshd will allow an RSA authenticated connection. Otherwise, if the host's entry is not in this file ssh will add it to the users local file, **~/.ssh/known_hosts**. Generate the entries as root on the host, with ***ssh-keygen***.
	ssh_config	system-wide ssh configuration file. Provides defaults for parameters not specified in the users' **~/.ssh/config**. See the table below for a list of keywords and default arguments.
	sshd_config	***sshd*** (ssh server daemon) configuration file. Lines beginning with # and empty lines are comments. Configuration lines have the form: "**keywords arguments**", where the keywords are case sensitive. See the table below for a list of keywords and default arguments.
	sshd.pid	process id number of the latest ***sshd***.
	nologin	limits logins to root user only, if it exists. The contents of this file will be displayed to any user trying to login in.
	environment	environment variables to set at login. Lines should be of the form "**name=value**".
	hosts.equiv	lists hosts and users allowed to use rlogin/rsh if **RhostsAuthentication** or **RhostRSAAuthentication** is set.
	shosts.equiv	same as **/etc/hosts.equiv**, but only for ssh.
	sshrc	commands to execute when the user logs in before starting the user's shell.

TABLE 29.2 **Files used by SSH**

Directory	File	Usage
$HOME (~)	.rhosts	provides .rhosts authentication if enabled by the ssh configuration files.
	.shosts	same as **~/.rhosts**, but only for ssh.
	.Xauthority	used by ssh to store the authorization cookie for the X11 server. Ssh verifies that X11 forwarded connections carry this cookie. When the connection is opened the real cookie replaces this one. All X11 displays automatically go through the encrypted channel via a proxy X server created by ssh. Ssh will set the DISPLAY environment variable pointing to the server machine with a display number greater than zero.
$HOME/.ssh (~/.ssh)	known_hosts.	used in conjunction with /etc/ssh_known_hosts. This is ignored if **StrictHostKeyChecking** is enabled
	authorized_keys	list of public keys of users that are allowed access to this account without a password. Generate the entries as the user on the host with ***ssh-keygen*** and provide a passphrase. Additional security options can be specified here. The user's local public key, kept in **~/.ssh/identity.pub**, should be in this authorized_keys file on the remote machine. This file replaces the function of **~/.rhosts** when using RSA authentication. It allows the user to login without providing a password. This file has one key per line, each in the form: 1024 37 44765081...(lots of numbers)...86828 frank@other_machine
	identity	local private key of the user.
	identity.pub	local public key of the user. This should be copied to **~/.ssh/authorized_keys** on the remote machine. This file has one line of the form: 1025 35 5574508...lots of numbers)...74727 frank@this_machine
	random_seed	contains the seed for the random number generator. It should be read/write only for the user and should not be changed by the user.
	config	configuration file for the user. The format is the same as for the system-wide ssh configuration file, **/etc/ssh_config**.
	environment	environment variables to set at login for this user. Similar to **/etc/environment** and read after that file.
	rc	same as **/etc/sshrc**, but for the individual user.

29.3.1 Configuration Options

SSH allows you to specify command line options and will read configuration options from a user file (**~/.ssh/config**) and a system-wide configuration file (**/etc/ssh_config** and **/etc/sshd_config**), with preference in the order: option, user, system. Valid keywords and their arguments for the options to the ***ssh*** and ***sshd*** configuration parameters are in the following table.

TABLE 29.3 Keywords and Arguments

Keyword	Arguments	Default	Server or Client	Comment
AllowHosts	host_names host_ipaddresses	all hosts	Server	Hosts allowed to login. Space separated list of hostname or IP addresses. Wildcards: "*" and "?" are accepted for pattern matches
BatchMode	yes/no	no	Client	Should passphrase/password querying be disabled
Cipher	idea/des/3des/arc-four/tss/none	idea	Client	Specifies the cipher to use for encryption of the session
Compression	yes/no	no	Client	Compress the session data
CompressionLevel	1-9	6	Client	Compress using the gzip algorithm: 1->fast (poor); 9->slow (best)
ConnectionAttempts	integer	?	Client	Number of tries per second to attempt before falling back to rsh or exiting.
DenyHosts	hostname host_ipaddress	none	Server	Deny login from these hosts. Space separated list of hostname or IP addresses.
EscapeChar	~/^<char>/none	~	Client	The escape character to use.
FallBackToRsh	yes/no	yes	Client	Should the connection fall back to rsh if connection is refused by the remote host (i.e. no ***sshd*** is running)
FascistLogging	yes/no	no	Server	Should verbose logging be enabled.
ForwardAgent	yes/no	yes	Client	Should the connection to the authentication agent be forwarded to the remote machine.
ForwardX11	yes/no	yes	Client	Should X11 connections be forwarded over the secure channel and have **DISPLAY** set.
GlobalKnownHostsFile	file	/etc/ssh_known_hosts	Client	File to use instead of the default.
Host	host_names host_ipaddresses	none	Client	Restrict the configuration options following, up to the next Host declaration, to the desired host(s). Wildcards: "*" and "?" are accepted for pattern matches.
HostKey	host_key_file	/etc/ssh_host_key	Server	File to use instead of the default.
HostName	hostname	command line option	Client	Nicknames or abbreviations for hosts
IdentityFile	file	~/.ssh/identity	Client	File(s) containing users authentication identity
IgnoreRhosts	yes/no	no	Server	Should **~/.rhosts** and **~/.shosts** be used. **/etc/hosts.equi**v and **/etc/shosts.equiv** are still used.
KeepAlive	yes/no	yes	Both	Should the system send keepalive messages to the remote connection. Both client and server should agree on this.

TABLE 29.3 Keywords and Arguments

Keyword	Arguments	Default	Server or Client	Comment
KeyRegenerationInterval	time	3600	Server	Automatic key regeneration interval, in seconds
LocalForward	local_port remote_host:port	none	Client	The local tcp/ip port is forwarded to the remote host:port on the remote machine via the secure channel
LoginGraceTime	time	600	Server	Successful login must be accomplished within this period, in seconds.
PasswordAuthentication	yes/no	yes	Both	Should password authentication be allowed.
PermitEmptyPasswords	yes/no	yes	Server	Should empty passwords by permitted.
PermitRootLogin	yes/nopwd/no	yes	Server	Should root logins be permitted. "nopwd" disallows password authenticated root logins.
PidFile	pid_file	/etc/sshd.pid	Server	File to use instead of the default.
Port	port#	22	Both	Port to connect to on the remote host or to listen to on this machine
PrintMotd	yes/no	yes	Server	Should /etc/motd be printed at login.
ProxyCommand	command_string	none	Client	Command to connect to the remote server
QuietMode	yes/no	no	Server	Should the system run in quiet mode, i.e. log only fatal errors.
RandomSeed	random_seed_file	/etc/ssh_random_seed	Server	File to use instead of the default.
RemoteForward	remote_port local_host:port	none	Client	The remote tcp/ip port is forwarded to local host:port via the secure channel
RhostsAuthentication	yes/no	no	Both	Should rhosts based authentication be tried
RhostsRSAAuthentication	yes/no	yes	Both	Should rhosts based authentication with RSA host authentication be tried
RSAAuthentication	yes/no	yes	Both	Should RSA authentication be tried. The identity file must exist or an authentication agent must be running
ServerKeyBits	#bits	768	Server	Specify the number of bits to use in the server key, minimum 512.
StrictHostKeyChecking	yes/no	no	Client	If yes, hosts will not be automatically added to ~/.ssh/known_hosts and connections will be rejected to a host whose host key has changed
StrictModes	yes/no	yes	Server	Should strict checking of permissions be done on authentication files.
SyslogFacility	syslog_code	DAEMON	Server	Specify the logging code to use.
User	remote_user	your_login_id	Client	Become a different user on the remote end of the ssh connection
UserKnownHostsFile	file	~/.ssh/known_hosts	Client	File to use for the users' known hosts
UseRsh	yes/no	yes	Client	Should rlogin/rsh be used for this host
X11Forwarding	yes/no	yes	Server	Should X11 forwarding be permitted.

29.4 Setting up the Service

29.4.1 Files necessary to trust a user across the network

To trust a user from host A on host B **/etc/ssh_known_hosts** on both A and B should have the public keys of the other machine and the user should have their public key from host A in their **~/.ssh/authorized_keys** file on host B. Since the RSA authentication uses the private key, contained in **~/.ssh/identity**, if hosts A and B share the same NFS mounted home directory for the user putting the public key for the user, from **~/.ssh/identity.pub**, in **~/.ssh/authorized_keys** will mean that the user is trusted in both directions, i.e. from A⇒B and from B⇒A.

Root is treated as any other user, with its files in the directory **/.ssh**. For root and other system logins you may want to use an empty passphrase when creating the key. This is especially true if you want to run ***cron*** jobs between machines as this user, because there will not be anyone there to provide the passphrase when the job runs. The passphrase does provide an additional level of security. Should someone break into your system the private key could be stolen, but without the passphrase they would not be able to exploit it on the remote system.

These files will have entries similar to the following, where the keys have been truncated for brevity and each entry should be on a single line.

File	Contents
/etc/ssh_known_hosts	hostname,list,of,aliases,IP_addr key_size exponent host_key root@hostname
~/.ssh/known_hosts	hostname,list,of,aliases,IP_addr key_size exponent host_key root@hostname

e.g., the two files above might contain:

```
nyssa,nyssa.acs.ohio-state.edu,128.146.116.4 1024 37
    120868350090604089005971557002264781523818788127296256909647515960497982627466
    root@nyssa
susan,susan.acs.ohio-state.edu,www-wks.acs.ohio-state.edu,128.146.116.32 1024 35
    206297116078594680112446644696531356796278353005287817794587469777554961618889
    root@susan
```

~/.ssh/authorized_keys key_size exponent host_key user@hostname

e.g.:

```
1024 37
    287615623236504102828255516467970261345966571750574014601611091414106110923656
    frank@nyssa
1024 35
    261345966557401405287817795875946801144664466539060089057970263596571750574014
    frank@susan
```

~/.ssh/identity.pub key_size exponent host_key user@hostname

e.g.:

```
1024 37
    287615623236504102828255516467970261345966571750574014601611091414106110923656
    frank@nyssa
```

In these files aliases are separated by commas (**,**) and fields are separated by spaces.

29.4.2 Configuration Files

The server (***sshd***) configuration file is **/etc/sshd_config**. To allow hosts from a couple of subnets, use RSA authentication, but not Rhosts authentication, try a configuration file similar to the following:

```
# This is ssh server system-wide configuration file.
Port 22
AllowHosts 128.146.226.* 128.146.116.*
ListenAddress 0.0.0.0
HostKey /etc/ssh_host_key
ServerKeyBits 768
LoginGraceTime 600
KeyRegenerationInterval 3600
PermitRootLogin yes
QuietMode no
FascistLogging no
PrintMotd no
SyslogFacility LOCAL6
RhostsAuthentication no
RhostsRSAAuthentication no
RSAAuthentication yes
PasswordAuthentication no
```

The client (***ssh***) configuration files are: **/etc/ssh_config** for the system, and **~/.ssh/config** for a user. This configuration file disallows Rhosts authentication, but sets RSA and Password authentication, and enables StrictHostKeyChecking.

```
# This is ssh client system-wide configuration file.  This file provides defaults for users, and the values
# can be changed in per-user configuration files or on the command line.
# Configuration data is parsed as follows:
#  1. command line options
#  2. user-specific file
#  3. system-wide file
# Any configuration value is only changed the first time it is set.  Thus, host-specific definitions should
# be at the beginning of the configuration file, and defaults at the end.

# Site-wide defaults for various options
RhostsAuthentication no
RhostsRSAAuthentication no
RSAAuthentication yes
PasswordAuthentication yes
StrictHostKeyChecking yes
```

29.4.3 Generating the keys, ssh-keygen

To generate the keys use ***ssh-keygen***. It will populate the files: **~/.ssh/identity** and **~/.ssh/identity.pub** for the user. If the user is root do this without a passphrase, and then you can copy these to: **/etc/ssh_host_key** and **/etc/ssh_host_key.pub**. Then to allow hosts and users to connect you copy the public keys from the remote hosts and users to the comparable files on this host, and for the desired user.

Host: **/etc/ssh_host_key.pub** ⇒ **/etc/ssh_known_hosts** This is required if **StrictHostKeyChecking** is turned on in **sshd_config**. If this is not turned on than the user's **~/.ssh/known_hosts** file will be updated when they connect to other hosts.

User: **~/.ssh/identity.pub** ⇒ **~/.ssh/authorized_keys**

Each user must use ***ssh-keygen*** to generate their own unique set of keys. For additional security they should provide a passphrase.

29.4.4 Authentication and Encryption

The default authentication mechanism is **RSA**, based on public key cryptography. This scheme has separate keys for encryption and decryption. With sufficiently large keys it is not possible to guess the decryption key given the encryption key. This allows one to publicly provide the encryption key so that other users or machines can encrypt their message with it. Then only the holder of the private decryption key should be able to decrypt the message. This private key can be optionally protected with a passphrase for additional security.

Several encryption algorithms are available. The default, and most secure, is **idea**.

29.5 Login Process

Sshd controls the login process through the following steps:

1. Print the last login time (if the login is via tty and a command was not specified) and **/etc/motd** (if not prevented by the configuration file or by **~/.hushlogin**).
2. Record the login time (if the login is via tty).
3. If **/etc/nologin** exists, print the file and quit (except for root login).
4. Convert to run with privileges of the user.
5. Configure the environment.
6. If **/etc/environment** exists, read it and add it to the environment.
7. If **~/.ssh/environment** exists, read it and add it to the environment.
8. Change directory to the user's $HOME.
9. If **~/.ssh/rc** exists, run it with the user’s shell; if not, if **/etc/sshrc** exists, run it; otherwise run ***xauth***. When X11 spoofing is enabled the **rc** files are fed an X11 authentication protocol ($proto), cookie ($cookie) and $DISPLAY and the script is expected to call ***xauth*** to store the cookie.
10. Run the user's shell or command.

29.6 Installation

The source can be obtained from a number of places, including ftp://ftp.net.ohio-state.edu/pub/security/ssh/, with the latest version being 1.2.26.
To compile the source and install the software do the following:

1. zcat ssh-1.2.26.tar.gz | tar -xvBf - ; cd ssh-1.2.26 # Open up the files
2. Configure the setup, (see the files README, OVERVIEW, INSTALL and the man pages). The default is to put the client files will be installed in the **bin** directory under the prefix (default is **/usr/local**) and the server in **sbin**, e.g.:
   ./configure --prefix=/opt/local --with-rsh=/bin/rsh
3. make
4. make install
5. Set the daemon up to run at boot. The following script should do this for you. It can be found as: ftp://wks.uts.ohio-state.edu/pub/solaris2/src/setup_ssh.sh. This script will:
   a enable the daemon to be started at boot time
   b generate the host key for the machine
   c sets up default configuration files for clients and server
   d log server connections using LOCAL6 through syslogd to **/var/log/sshd_log**
   e start the server
   Later you can edit the control files as desired. Should you change entries in **/etc/sshd_config** you will need to send a HUP signal to ***sshd*** so that it will reread this file.

```
#!/bin/sh
# Frank Fiamingo      March 15, 1996
# Script to setup sshd
# name:   ssh
# vers:    1.2.13
# source: ftp://ftp.net.ohio-state.edu/pub/security/ssh
date=`date +%m/%d/%y`
top=/usr
OS=`uname -s`
OSlevel=`uname -rlcut -c1`
if [ "$OSlevel" = "5" ]; then
        if [ "$OS" = "SunOS" ]; then
                top=/opt
        fi
fi
if [ "$OSlevel" = "5" ]; then          # Solaris 2.X or IRIX 5.X
        if [ ! -f /etc/init.d/sshd ];then
          cat << EOF_init.d > /etc/init.d/sshd
#!/bin/sh
#
# start up sshd, installed by $USER, $date
#
case "\$1" in
'start')
```

```
        if [ -x $top/local/sbin/sshd ]; then
            $top/local/sbin/sshd && \\
                        echo "Starting sshd daemon, takes about 1 minute... "
        fi
            ;;
'stop')
            [ ! -f /etc/sshd.pid ] && exit 0
            syspid=\`cat /etc/sshd.pid\`
            if [ "\$syspid" -gt 0 ]; then
                        echo "Stopping the sshd daemon."
                        kill -15 \$syspid 2>&1 | /bin/grep -v "no such process"
            fi
            ;;
*)
            echo "Usage: /etc/init.d/sshd { start | stop }"
            ;;
esac
exit 0
EOF_init.d
             chmod 755 /etc/init.d/sshd
             (cd /etc/rc2.d ; ln -s ../init.d/sshd S99sshd )
            fi
fi          # end if for OSlevel=5

if [ "$OSlevel" = "4" ]; then           # Solaris 1.X
            if [ -f /etc/rc.local ]; then
             grep $top/local/sbin/sshd /etc/rc.local >/dev/null 2>&1 ||
              cat << EOF_rc.local >> /etc/rc.local
#
# sshd daemon, installed by $USER, $date
if [ -x $top/local/sbin/sshd ]; then
            $top/local/sbin/sshd && echo ' Starting sshd '
fi
EOF_rc.local
            else
             echo "/etc/rc.local not found ..."
            fi
fi          # end if for OSlevel=4

if [ ! -f /etc/ssh_host_key ];then
            echo ""
            echo "We're now going to generate the host key for this machine."
            echo "We'll use a null passphrase."
            echo "This will take a little while ..."
            rm -f /.ssh/identity /.ssh/identity.pub
            (echo /.ssh/identity | ssh-keygen -N "" ) && echo "Done."
            cp /.ssh/identity /etc/ssh_host_key && chmod 600 /etc/ssh_host_key
            cp /.ssh/identity.pub /etc/ssh_host_key.pub
fi
# Configure the client service with the file /etc/ssh_config
if [ ! -f /etc/ssh_config ];then
            cat << EOF_ssh > /etc/ssh_config
```

```
# This is the ssh client system-wide configuration file.
# It provides the defaults, whose values can be changed in
# the user's own configuration file or on the command line.
RhostsAuthentication no
RhostsRSAAuthentication no
RSAAuthentication yes
PasswordAuthentication yes
StrictHostKeyChecking yes
EOF_ssh
fi
# Configure the daemon with the file /etc/sshd_config
if [ ! -f /etc/sshd_config ];then
        cat << EOF_sshd > /etc/sshd_config
# This is the ssh server system-wide configuration file.
Port 22
AllowHosts 128.146.226.* 128.146.116.*
ListenAddress 0.0.0.0
HostKey /etc/ssh_host_key
ServerKeyBits 768
LoginGraceTime 600
KeyRegenerationInterval 3600
PermitRootLogin yes
QuietMode no
FascistLogging no
PrintMotd no
SyslogFacility LOCAL6
RhostsAuthentication no
RhostsRSAAuthentication no
RSAAuthentication yes
PasswordAuthentication no
EOF_sshd
fi
# If the daemon configuration file was set up at install, make sure
# that we log to local6
grep "SyslogFacility LOCAL6" /etc/sshd_config >/dev/null 2>&1 ||
        if sed -e 's/DAEMON/LOCAL6/' /etc/sshd_config > tmp_sshd_config
        then
          mv tmp_sshd_config /etc/sshd_config
        else
          echo "SyslogFacility LOCAL6" >> /etc/sshd_config
        fi
# If the daemon's currently running, stop it.
if [ -f /etc/sshd.pid ];then
        kill -15 `cat /etc/sshd.pid`
fi
# Make sure that syslog logs sshd reports to a separate file
# In the following we use tabs, not spaces, as separators.
grep local6 /etc/syslog.conf >/dev/null 2>&1 ||
  (echo "local6.debug           /var/log/sshd_log" >> /etc/syslog.conf;\
  touch /var/log/sshd_log; \
  kill -HUP `cat /etc/syslog.pid` )
# Start the daemon
```



```
$top/local/sbin/sshd
echo ""
echo "This host should now be running the sshd daemon."
echo "You will still need to edit /etc/ssh_known_hosts to put the "
echo "desired public host keys for the machines you want to trust."
```

PART IV Summary

SunOS/Solaris Command Summary

UTS UNIX Workstation Support

CHAPTER 30

Summary of SunOS/Solaris Differences

30.1 SunOS 4.1.X and 5.X Administrative Command Differences

The summary of SunOS administrative command differences is given in the following table.

TABLE 30.1 Administrative Commands

SunOS 4.X	SunOS 5.X	Comments
add_services	pkgadd swmtool	Add software packages.
arch	uname -m	Determine the system architecture.
at	at	Security is more restricted under SunOS 5.X.
automount	automount	The new master file names are auto_master and auto_home. The default home directory is /export/home/<username>.
bar	NA	Use tar or cpio -H bar to replace bar.
biff -y	chmod o+x /dev/tty	Set the tty permissions, as biff is not available.
biff -n	chmod o-x /dev/tty	Set the tty permissions, as biff is not available.
biod	NA	Block I/O daemon.
cc	/opt/SUNWspro/bin/cc	Separate product.
chown	chown	How it treats symbolic links is changed. The command now follows the link and changes permissions on the file. To change the ownership of the link use chown -h.
dcheck	NA	File system directory consistency check.
dd	dd	Now uses 2-byte, rather than 4-byte words.
devinfo	devinfo sysdef -d	Information reported has been changed.
df	df -k	Output format and options are changed.
dkinfo	prtvtoc	Reports similar information; privileged command in SunOS5.4-.
dorfs	rfstart/rfstop	RFS commands.

TABLE 30.1 **Administrative Commands**

SunOS 4.X	SunOS 5.X	Comments
du	du -k	Now reports in 512 byte, rather than 1024 byte blocks.
dump	ufsdump	Some new options. Now recognizes end-of-media.
etherfind	snoop	Similar functions.
exportfs	share	For both NFS and RFS.
extract_files	NA	Extract files from installation media.
extract_patch	NA	Extract patches from installation media.
extract_unbundled	pkgadd swmtool	Add software packages.
fastboot	init 6	Run level 6.
fasthalt	init 0	Run level 0.
file	file	No longer has the -L option.
find	find	No longer has the -n cpio option.
fsck	fsck	Changed.
hostid	sysdef -h	Sysdef is used to report the current system definitions, including peripherals attached and drivers loaded.
hostname	uname -n	Uname prints current system definitions.
init	init	Many changes, including run levels, etc.
intr	NA	Allow the following command to be interruptible.
iostat	iostat	Some options are changed.
ldconfig	NA	Configure the cache for the run-time link editor, ld.so.
lpc	lpsched	LP scheduler.
lpd	lpadmin	LP configuration command.
lpq	lpstat	Status of LP jobs.
lpr	lp	Some different options.
lprm	cancel	Cancel an LP job.
lptest	NA	Generate a test pattern for the line printer.
ls	ls	Some options are changed.
mach	uname -p	Report the machine type.
make	make	Now located in /usr/ccs/bin (package SUNWsprot).
makekey	NA	Generate an encryption key.
mkfs	mkfs	Changed to support additional file system types.
mknod	mknod	No longer have to be root to create character and block special files.
modstat	modinfo	Displays information about the kernel modules loaded.
mount	mount	Changed to include additional file system types.
ncheck	ncheck	Changed to include additional file system types.

TABLE 30.1 **Administrative Commands**

SunOS 4.X	SunOS 5.X	Comments
portmap	rpcbind	Maps universal addresses to RPC program number.
printenv	env	Print the user's environment variables.
ps	ps	Options are changed, e.g. use ps -ef instead of ps aux.
pstat	sar	Reports on system activity.
pstat -s	swap -s	Reports on swap space available.
rdump	ufsdump	Remote drives can be specified.
restore	ufsrestore	File system restore program.
rpc.etherd	NA	Server for ethernet statistics.
rpc.lockd	lockd	File locking daemon.
rpc.mountd	mountd	Mount daemon.
rpc.rquotad	rquotad	Server for remote quotas.
rpc.statd	statd	Network status monitor.
rpc.yppasswdd	rpc.yppasswdd	NIS password daemon; install NIS compatibility package, SUNWnsktu.
rrestore	ufsrestore	Remote drives can be specified.
rusage	NA	Resource usage for the specified command.
shutdown	shutdown	Significant changes.
stty	stty	Some options have been changed.
suninstall	suninstall	Significant changes.
swapon	swap -a	Add swap space.
ttysoftcar	NA	Modem carrier control.
tzsetup	NA	Timezone setup. Set with the /etc/default/init file.
umount	umount	Changed to include additional file system types.
unload	pkgrm	Remove a software package.
update	fsflush	Flush the memory buffers.
vipw	/usr/ucb/vipw	/etc/passwd editing; also allows editing of /etc/shadow.
vmstat	vmstat	Some options are changed.
who	who	Additional options available.
whoami	id	Print the username.
yppasswd	passwd yppasswd nispasswd	The yppasswd command is still available for changing password information on an NIS server. Use nispasswd to access NIS+ servers.
ypserv	/usr/lib/netsvc/yp/ypserv rpc.nisd	NIS daemon (install package SUNWnsktu). NIS+ uses this daemon to service requests for information.

30.2 SunOS 4.1.X and 5.X Administrative File Differences

The following table lists some of the important files that have been changed.

TABLE 30.2 Administrative Files

SunOS 4.X	SunOS 5.X	Comments
/boot	/ufsboot	Boot program.
/etc/auto.master	/etc/auto_master	Automounter configuration file.
/etc/auto.home	/etc/auto_home	Automounter configuration file.
/etc/exports	/etc/dfs/dfstab	Files shared by NFS and RFS.
/etc/fstab	/etc/vfstab	Table of files to mount.
/etc/gettytab	/etc/ttydefs	Terminal definitions.
/etc/passwd	/etc/passwd /etc/shadow	Shadow password file is now used.
/etc/printcap	/usr/share/lib/terminfo /etc/lp /etc/printers.conf	Database of printer and terminal characteristics. Directory of printer information. Database of configured printers (5.6+).
/etc/rc	/sbin/rc# /etc/rc#.d/	The /etc/rc#.d subdirectory scripts are now used, which each rc# script and rc#.d directory controlling the run-level #.
/etc/rc.boot	"	"
/etc/rc.local	"	"
/etc/rc.single	"	"
/etc/termcap	/usr/share/lib/terminfo	Database of printer and terminal characteristics.
/etc/ttytab	/etc/inittab	Table of services to be started by init.
NA	/etc/saf	Directory of SAF services.
NA	/etc/default/login	Defaults for login. Root login limited to console.
/usr/share/man	/usr/share/man	Man page organization has been changed. System administration man pages are in 1M. You can set an environment variable to specify the order of search for directories and sections.
/var/spool/mail	/var/mail	Mail spool directory.
/vmunix	/kernel/unix /platform/\uname -m‘/kernel/unix	The hardware independent UNIX kernel. The hardware dependent part of the kernel.

CHAPTER 31 UTS UNIX Workstation Support

31.1 UTS WORKSTATION SUPPORT TEAM

The Ohio State University / University Technology Services (UTS) Workstation Support Team consists of:

Alan Albertus - Manager, Consultation (alan+@osu.edu)

Rob Funk - ***Sun***/SunOS & Solaris, general Unix (Baker Systems 452, 2-7802, funk+@osu.edu)

Bob Debula - ***SGI***/IRIX, ***DEC***/Ultrix, Digital UNIX (Baker Systems 454, 2-4843, bobd+@osu.edu)

We can also draw on the expertise of other UTS staff, including:

Harpal Chohan - X-Windows, Usenet news, Packet audio/video (chohan+@osu.edu)

Mohammed Rahman - SAS and statistical applications (rahman.5@osu.edu)

Jerry Martin - Network Information Center, network planning (jerry+@osu.edu)

Clifford Collins, Steve Romig, Mowgli Assor - Security concerns (security@net.ohio-state.edu)

HP (HP-UX) - Support is provided by the College of Engineering. Contact Jim Gaynor (site@ee.eng.ohio-state.edu).

IBM RS6000 (AIX) - Tom Merrick in Engineering may be able to provide help (merrick.2@osu.edu).

31.2 Software

University Technology Services has site-licensed software from *Sun Microsystems* for *Sun* SPARC hardware. This software can be borrowed from the **Information Center** (Baker 512, 2-2626). This software includes:

SunOS
C/C++
Fortran
Pascal
SunNet Manager
PC-NFS

For ***DEC*** workstations we have **Ultrix** 4.5, **Digital UNIX** (formerly **OSF/1**) (for Alpha) and the ***DEC*** applications FORTRAN, DECNet, SQL, and DECFUSE. Also, on CD we have the Ultrix Consolidated Software Distribution (2 CDs), and DECwindows for OSF/MOTIF. You need to buy into the CSLG program to have access to this software; contact Chuck Sechler, 2-4843, for details.

For ***SGI*** workstations we have **IRIX** 5.3, 6.2, 6.3, 6.4, and 6.5, and the Varsity Pack software. OS maintenance and the Varsity Pack software need to be purchased through the Bookstore. For details contact **Bob DeBula**.

This and other workstation software we have are available through the **Information Center** Additional software includes: OSF/Motif 1.2.2 source, DECwrite for Sun workstations, Maple for most workstations of interest, NCAR graphics, NQS, SAS for Sun and HP workstations, and WordPerfect 5.1 for Sun workstations. Of these, only SAS and WordPerfect have a cost involved.

SunOS software ***patches*** can be obtained from Sun Support by workstation support staff. When obtained these are put up for anonymous ftp at **ftp://wks.uts.ohio-state.edu/pub/sunpatches/**.

IRIX software patches can be obtained from SGI directly, at **http://support.sgi.com/**. Other SGI related information can be found via anonymous ftp and www on **araminta.acs.ohio-state.edu**.

If you need further information, contact one of us above.

