Unix Reference

Table of Contents

Appendix

Help and Information

To talk with ITC or ACHS staff about computer questions or for information about courses call:

ITC Help Desk	924-3731	For general questions and information start here.
ACHS	982-4025	For ACHS (Health Sciences) specific questions.

After normal business hours or for more detailed requests send email to “consult@virginia.edu”.

To request a Unix account on the ACHS Unix cluster send email to “accounts@virginia.edu”.

Dialin Phone Numbers:

28.8k Modems	963-4624	(60-minute limit)
	296-8963	(30- minute limit)
14.4k Modems	243-7673	(60-minute limit)
	982-5084	(30-minute limit)

1. Introduction

These notes are intended to give a general overview of computers and the associated network and peripherals that are connected to them, and as an introduction to the Unix operating system. Computer systems consist of hardware (terminals, computers, printers, network, etc.) and software (operating systems, network software, computer programs, etc.). Both parts will be addressed, at least briefly, mainly to explain the terms one may hear.

Unix is an operating system (the program the computer uses to do its work, like DOS or Windows on a PC) that supports **multiple users** (more than one person working on the machine at the same) and **multiple tasks** (each user can run more than one program simultaneously). It was created in the early 1970's at Bell Labs as an **open** system (users could look at each other's files and at the operating system files) for programmers. It's main attractions was that it was based on **modules** (small, single purpose programs) that when used together were very powerful, and which users could write themselves to extend the utility of the system. Since it was open, users could look at operating system code and change it. And they did. So today there are many flavors of Unix (AIX, BSD, IRIX, OSF1, Solaris, etc.), all similar in functionality, but slightly different in syntax and operation. This also allowed it to be **portable**, or able to be modified to run on different manufacturer's computers. It has become one of the most popular operating systems for higher-powered personal workstations, particularly those used in a networked environment. It's power and popularity has caused the decline of more expensive mainframes. And declining prices are making it competitive with high end PC's.

Unix is case sensitive, lower case letters are different than upper case (different than DOS). Almost all commands user lower case letters, but you can use either for creating file names.

2. Unix File Structure and Paths

File Structure

Unix uses a **hierarchical directory and file structure**, sometimes referred to as a tree structure with directories (similar to folders) containing files and other sub-directories. The top-level directory is called the root and is indicated by a **forward slash "/" (opposite of DOS "\").** Most Unix machines are set up in a similar fashion with special directories for the operating system files (/etc), executable commands (/bin), libraries (/lib), and user files, also called accounts (/home/loginid). (See Appendix for diagram "Unix File System")

Unix requires that each user have a loginid or username and password that must be entered at the login prompt to be allowed to use the computer (the same as Novell). When a user logs in he is said to be located in his home directory, i.e. - the operating system attaches his login process with the directory that contains his files, such as /home/loginid. The user can move around the directories and look at files using Unix commands.

Paths

A file's **absolute path** is the list of directories and sub-directories starting from root that indicates the location of the file in the directory structure. For example, the path of a file in a users home directory might look like: /home/loginid/file, where each directory and file is separated by a forward slash. A file's **relative path** is the list of directories and sub-directories starting from your current location in the directory structure that indicates the location of the file. For example, if a user just logged in and is in his home directory, the relative path of a file in his sub-directory "mail" is mail/file. Notice there is no leading forward slash. Relative paths never start with a slash, only absolute paths starting at root.

There are two **special directory representations** that are useful in certain situations. A single period "." denotes the current directory, while two periods ".." indicates the parent directory or directory above the current one. The notation "../.." indicates the directory two levels above the current directory, etc.

3. Interfaces

Command Line

Unix started out as a command line operating system, i.e. - you had to type in commands after a prompt from the operating system (usually a $). A program called a shell displayed the prompt and read in the command after the user pressed the "enter" or carriage return key. The shell program then interpreted the command and passed it to the operating system also known as the Unix kernel. Several different shell programs were developed (sh, ksh, csh, tcsh) to meet the needs of different users. As software became more sophisticated, and users desired a friendlier and more graphically oriented interface, menus were created for ASCI terminals and window systems where developed for graphics monitors. (See Appendix for diagram "Unix Operating System")

Umenu

At UVa people who got Unix accounts a few years ago on the IBM RS/6000's did not have a choice and had to use the command line. Now new accounts are created with the Umenu (Unix menu) interface as the default. New users can configure their accounts so that the menu is not used and they get a command line prompt. Users with old accounts can update their accounts to use umenu by running the program "install.umenu". The Umenu interface lets people start using Unix machines without having to know any Unix commands, and do things like read electronic mail and news, look at documentation, and travel the Information Highway. There is a menu item "Go to Unix" that users can select to get out of Umenu and get a prompt where they can execute Unix commands.

X-Window

Unix machines with graphics monitors can run the sophisticated X-window interface program. It is especially useful when run on a machine that is attached to the network as it will allow you to run and display output from programs on other remote machines. There are special programs, called X-window clients, that are more graphically oriented than their ASCI counterparts, that also enable users to read email and news, view graphics, etc. Users can open windows that emulate a terminal session (xterm) where they can execute Unix commands. The X-window interface can also be run on PC's and Mac's that have ethernet connections, graphics monitors and the proper software installed. ITC supports eXceed for PC's and MacX for Mac's.

4. Unix Commands

Basic Commands

Unix command names are usually abbreviated forms of the real command name. They are mostly formed from by removing the vowels of the word or by using the first letters of multi-word commands. Only enough letters are retained to make the command unique. For example the command to copy a file is named “cp”. That for changing directories is “cd”. They may seem hard to remember at first, but once they are learned they are much faster to type than the full command names would be. Here is a list of the most common commands with their function:

alias	prints list of aliases and commands
cat *file*	(concatenate) display *file*
cat *file2* >> *file1*	appends *file2* to end of *file1*
cd *dir*	change directory to *dir*
chmod go+r *file*	changes permission of *file*
chsh	changes users default shell
compress / uncompress *file*	compresses or uncompresses *file*
cp *file1 file2*	copy *file1* to *file2*
diff *file1 file2*	displays differences between files
du -s	calculates disk space of current directory
echo $VARIABLE	shows value of VARIABLE
file *file*	displays type of file (text, binary, etc)
find . -name *filename* -print	finds occurrences of *filename*
grep *string file*	prints lines in *file* containing *string*
head *file*	display first six lines of *file*
kill -9 *PID*	stops program with process ID *PID*
ln -s *name1 name2*	creates symbolic link *name2* to file *name1*
ls	list filenames in a directory
man *command*	show the on-line manual entry for *command*
manpage *command*	print the manpage on default printer
mkdir *dir*	make a new directory *dir*
more *file*	read or display *file* on the screen
mv *file1 file2*	move or rename *file1* to *file2*
nohup *command* &	starts *command* in background and continues even if you logout
passwd	change password
ps -u *loginid*	shows processes started by user *loginid*
ps -ef	shows all processes running on machine
rm *file*	remove or delete *file*
rmdir *dir*	remove or delete directory *dir*
set	shows values of all defined variables
sort *file*	displays sorted lines in *file* on screen
tail *file*	display the last 10 lines of *file* tail *file*
whence *command*	shows what directory *command* is located

Special Characters

In the above examples a user would substitute the actual names for *file* and *dir*. There is a way to choose many files by using a **wildcard character**, an asterisk (*), in place of all or part of a file or directory name. So for example, is you wanted to list all the files in a directory that began with the letter "a", you could use the command "ls a*".

Most commands can take one or more optional parameters or flags. The syntax is similar for most commands, a space following the command name, then a minus sign followed by the oparameter. For instance, to see a different format of listing file information you can use the command "ls -al". This will show information about the size of the files, who created them, who can read and modify them, as well as their names.

To find out what parameters a command can take, you can look up the manual page (the syntax and somewhat terse explanation of a command) with the "man" command. To print a hardcopy of the manual page use the "manpage" command.

The "cd" command is used to move around the file system and changes the current position to the specified directory. Typing **"cd" by itself returns you to your home directory**. Typing "cd .." moves you up one level to the parent directory. You can add additional instances of ".." to move up several directories, for example "cd ../../.." will move you up three directories.

Stopping Programs

Since Unix workstations are multi-user systems, they are usually never turned off, since someone may be connected to it over the network. So if something goes wrong with a program, for instance it no longer responds to the keyboard (referred to as a hung program), or if a program is taking too long to finish, you can't just turn off the machine like you sometimes do with a PC. Instead you can hold the control key down and press the letter "c" key (denoted by ^C or CTL-C in some documents). This should interrupt or kill the program and give you a shell prompt.

Sometimes CTL-C doesn't work, in that case you should try logging in on another terminal to try to stop the program. On the other terminal issue the command:

ps -fu *loginid* (RS/60000, SGI)
ps -ux | grep *loginid* (Sun)

The output of this command will be several lines of information, one for each process you are running. The last column will contain the program name, while the second column will show what is called the process ID or PID. Use this number with the "kill" command to stop the process. If the other terminal is hung, you need to kill the login session which is the shell process, or line ending in "-ksh". Make sure you don't kill your present session, look for the process with the older STIME value. Stop the process with the "kill" command substituting the actual process ID number for PID below:

kill -9 *PID*

This procedure can also be used to stop background and batch jobs.

Exiting

When you are finished with your work you should end your computing session by typing “exit”. This will log you out of the computer. If you are remotely connected to a Unix system through a PC and you do not logout but just turn off the PC, you may leave programs running or corrupt files such as your mailbox.

Printing Commands

There are several commands used to print files, depending on how you want the output to look. The most basic command, “lpr”, just sends a file to the printer without modify it in any way. A more sophisticated command found on some Unix machines called “enscript” will perform some basic formatting and font changes, but the output can only be printed by a PostScript laser printer such as an Apple LaserWriter or HP LaserJet 4m.

When a file is submitted for printing it is placed in a print queue while it waits for the printer. There is a command to check the status of print queues, “lpq”, and one to delete a file from a queue “lprm”.

Typical syntax for these commands are:

lpr -P*printer file*	send *file* to *printer* as is
lpq -P*printer*	check status of *printer* queue
lprm -P*printer job#*	remove *job#* from *printer* queue
enscript -2rhG -P*printer file*	format *file* and send to *printer*
manpage -P*printer command*	send manpage of *command* to *printer*

The parameters for the “enscript” command direct the output to be in two columns rotated sideways in landscape mode, with no header page, and a gaudy header line with the name of the file and date printed. The “-P*printer*” option specifies which printer the file should be directed to. If no printer is specified the users default printer is used if one has been set. Users defaults will be discussed later in the “Startup Files” section. A list of all the printers accessible from ITC machines is kept in the file “/etc/printcap”. Users can look at this file to find a printer that is located near them by using the Unix command “more /etc/printcap”. The name of the printer is located near the left margin and the printer type and location is found on the first line of each printer’s specifications.

Network Commands

There are several basic network commands available to users. They mainly provide ways to copy files between computers over the network and allow users to login to remote computers.

ftp *hostname* (file transfer protocol) connects to remote machine *hostname* and allows users to transfer files. Commands within ftp are:

ascii sets ascii mode for transfer of text file (usually default)
binary sets binary mode for transfer of binary (non-text) files get *file* transfers *file* from remote to local machine
put *file* transfers *file* from local to remote machine

telnet *hostname* connects to *hostname* and allows user to login and execute commands on remote machine

rlogin *hostname* — same as telnet but more robust, used only between two Unix machines

For PC users who are not running Lan Work Place (LWP) or who are dialing in through a modem, ITC has created some programs to make uploading and downloading files from their PC's to Unix accounts easier. The recommended way to use them is from within Umenu. At the Unix prompt type "umenu" then select "Commands Menu", then either "Upload a file" or "Download a file". Users can also execute the commands "upload" and "download" from the Unix prompt, but they will not see the verbose instructions provided by Umenu.

PC users who have Lan Workplace (LWP) installed can use the "Rapid Filer" utility to move files between their PC and Unix accounts. It is menu driven and easier to use than the "upload" and "download" programs. Mac users have a similar utility called "fetch".

X-Window Commands

If you are working at an X-terminal or a workstation running X-Windows, you can have the output from a program on a remote machine displayed on your terminal. For this to work your terminal must allow the remote machine to open a window on your display, and the remote machine must be told where to open the window. Most terminals are already configured to allow other machines to open a window on their display. To check if the one you are using is set this way execute the command "xhost". If the response is "access control disabled, clients can connect from any host" you don't have to do anything else. If the response is anything else, then you must execute the following command in an xterm window on your terminal (not one logged into the remote machine):

xhost +

To direct the program output from the remote machine to your X-terminal or workstation you must rlogin to the remote machine and then execute the following command on the remote machine:

DISPLAY=*machine.subnet*.Virginia.EDU:0.0

Substitute the name of your xterminal or workstation for "machine.subnet". Then you can execute the program on the remote machine and its output should appear in a window on your terminal.

5. Startup Files, Aliases and Variables

Startup Files

When a user logs in, the operating system sets certain defaults. Many of the defaults can be configured by the user by changing parameters contained in special environment files located in his home directory called startup files. These files all start with a period, and also called hidden files since they usually don't show up in a directory listing. A user will not want to change most defaults, but some like the default printer, editor, mail program, and location of special programs, may be customized. Most of these changes can be made through the "System Customization" menu item in Umenu which gives instructions and prompts for the new setting after showing the choices. They also

can be modified by editing the files manually, though caution is recommended when modifying any of these files. New users will also want to register for electronic mail so the university mail handling computer knows where to deliver their email. This can also be done in the "System Customization" menu item in Umenu or by executing the command "mailreg".

Users with older accounts may not have the most recent startup files or may have startup files they copied from someone else. To replace the old files with the newest ones a user can execute the command "install.startup" and the old startup files will be moved into a sub-directory and the new files put in place. The user can then customize the account as described above. The user's files will remain unchanged. If for some reason the user wants to go back to using the old startup files the "restore.startup" command will move them back into place. In both cases the user must logout and log back in to start using the changed files.

There are two startup files that contain most of the information a user may want to modify. The file .profile in older accounts or .variables.ksh in new accounts, contain the variables for the default printer, editor, mail program and path which is the list of directories searched for commands and programs. The variables are usually capitalized and are assigned values by the syntax "VARIABLE=value". To see a display of all the defined variables and their values, you can use the command "set" with no parameters. (See Appendix for "Sample .variables.ksh File")

One of the features of Unix is the use of aliases for long commands. The file .kshrc contains pre-defined aliases, but you can add your own if desired. The format is "alias word="command". So instead of typing a long command with many options, you can create a one-word alias that will execute the long command when you invoke it. To see a display of all the defined aliases and their values, you can use the command "alias" with no parameters. (See Appendix for "Sample .kshrc File")

The .kshrc file also contains a line to set your command-line editor which is used to re-execute previous commands without having to retype them. You should set it to match your default editor specified in the .variables.ksh file. Once set you are able to use editor commands to call back and edit previous commands. To call back the last command issued for the two possible editors types use:

vi	Esc K
emacs, jove	Ctrl P

6. I/O Redirection and Pipes

Normally a Unix command takes its input (called standard in) from the keyboard and sends it's output (called standard out) to the screen. A nice feature of Unix is the ability to redirect a command's standard in and out so it takes it's input from a file or another command and/or sends it's output to another command or file. This ability lets you connect several commands in a row, called a pipe. There are special characters used to perform these functions:

>	write output from a command to a file
>>	append of a command to a file
<	read input for a command from a file
|	send output from one command to another

Here are some examples of the use of these functions:

ls > *filelist*	put output of ls command into *filelist*
cat *file2* >> *file1*	append *file2* to *file1*
mail *loginid* < *filemsg*	read input for mail to *loginid* from *filemsg*
ps -ef | grep *string*	send output of ps to input of grep command

7. File Permissions

Every file and directory has an owner and belongs to a group, and has a set of permissions that controls access by the owner, group and all other non-group member accounts. The various owner, group, and permission settings for a file can be viewed by executing what is called a long listing of the file, ie. using the "-l" option for the list command. See the page labeled "File Permissions" in the Appendix for detailed information.

8. Shell Commands and Scripts

The program that displays the prompt and interprets your Unix commands is called a shell program. Shells such as ksh, have their own set of commands that can be used on the command line or in small programs called scripts to perform more complex operations. The man page for the shell gives the full definition of shell command syntax. Some of the most useful ksh shell commands follow.

for loop

```
for i in `/bin/ls -1`; do
        d=`echo $i | cut -f1 -d. `
        mv $i $d.tif
done
```

if test

```
if [ -f junk ]; then
        echo "File exists"
else
        echo "No file"
fi
```

9. Unix Programs

The openness of Unix encourages programmers to create several programs that perform the same task but in slightly different ways. Following is a description of some of the more important general purpose programs available at UVa.

Editors — For creating and modifying files:

emacs, jove, upenove	modeless, full featured, fairly complicated, ITC default
vi	mode, full featured, fairly complicated, Unix standard
pico	modeless, basic, easy to use, may be new default

Mailers — For reading and sending electronic mail (email) messages:

mail (mush)	ITC default for non-Umenu accounts
elm, upenelm	default for Umenu
pine	may be new default

News — For reading Usenet News (electronic bulletin boards):

rn, trn	older standard
upent	Umenu default

Information Access — For accessing various databases of information available worldwide on the Internet:

gwis	ITC's gopher service, ASCI based, general info
lynx	ASCI based World Wide Web (WWW) viewer
mosaic, netscape	X-window, Windows, and Mac WWW viewer

Unix Help — For finding Unix documentation and answers to frequently asked questions (FEQ's):

man	standard viewer for online Unix manual pages
hints	viewer for ITC documents and FEQ's. Can also be accessed from within Umenu and gwis.

10. Unix Documentation and Books

Here's a list of some of the Unix documentation and books available. This is by no means exhaustive, only some of the more popular ones are mentioned. The documents are free and all can be obtained at ITC in Wilson Hall and some at ACHS on the third floor of Hospital West multi-story building. Unix books can be obtained in the computer section of most bookstores.

ITC Docs

U-001 Unix Resource List (List of Unix commands and where to access info.)
U-001A Uva's Unix Menu System (Umenu)
U-002 Introduction to the Unix Operating System
U-003 JOVE Text Editor on the RS/6000 or Sun Computer
U-004 Vi Text Editor
U-010 Unix Mail Overview
U-010A A Primer for Mail on RS/6000 Computers (Mush)
U-011 USENET News
U-014 Introduction to the X Window System
U-014A More About X Windows
U-030 Unix Mass Storage System
TIB-165 An Introduction to PC Telnet (For IBM PC-compatibles)
M-022 Using Telnet and FTP on the Macintosh
TIB-167 TCP/IP - A Primer of User Commands (info on telnet and ftp on Unix)
TIB-175A An Introduction to Mosaic and the World Wide Web

Books

G. Todino & J. Strang, *Learning the Unix Operating System*, O'Reilly & Assoc. M. Levine Young & J. Lavine, *Unix for Dummies*, IDG Books

APPENDIX

A. Overview of Computer Hardware

Here is a brief comparison of the common devices one may use or hear about.

Terminal	Display and input device (a TV with a keyboard, no computer). Must be connected to a host computer through a serial port or modem. Also used with UBLAN (Ungermann Bass Local Area Network, discussed later). Usually displays only text (ASCII). Many manufacturers. (VT100, ADM5, etc.)
X-Terminal	Display and input device. Runs X-Window server software. Must be connected to hosts via network. Displays both text and graphics (bit mapped). Many manufacturers. (NCD, IBM, HP, etc.)
Mac Micro	Personal computer, runs Mac operating system, can be standalone, connected to UBLAN, or networked. Can display both text and graphics with proper hardware. Apple is the only manufacturer.
PC Micro	Personal computer, usually running DOS or Windows, can be standalone, connected to UBLAN, or networked. Can display both text and graphics with proper hardware. Many manufacturers, IBM was first, but many clones are sold such as Gateway, etc.
Workstation	Can be used as a personal computer on someone's desk or as a multi-user mainframe in a machine room. At one time these machines were known as mini-computers. Usually running Unix, but some run other operating systems such as VMS. Can be standalone, but usually networked. Can display both text and graphics with proper hardware. Many manufacturers such as IBM, SUN, SGI, DEC, HP, etc.
Mainframe	Large, multi-user computers, usually kept in a machine room somewhere with users connecting their terminal via serial lines or network. Runs proprietary operating system from manufacturer. Usually displays text but can display graphics with proper hardware. Few manufacturers, IBM being one of the largest. These machines are usually quite expensive.

B. Network Technologies

Networks are the cables, electronic boxes and software that connect computers and terminals with each other. They can be small local area networks called LAN's (a few computers in one room) or can be very large, complicated wide area networks (WAN) created by connecting many LAN's together. At UVa many buildings have LAN's which are connected by routers to large cables called "the backbone" which go through the entire university. The UVa network is in turn connected via dedicated phone lines to a larger network called the Internet, which is comprised of the networks of many schools and businesses throughout the USA and the world. The various terms and parts of a network are described below. (See Appendix for diagram "Uva Network")

These are the main hardware parts of UVa's network in about the order of development.

UBLAN	Ungermann Bass Local Area Network - Early system to connect remote terminals to central mainframes. Uses serial line technology within buildings and cable TV technology between buildings (backbone). Terminals and PC's with serial ports connected via NIU (Network Interface Units) to backbone. NIU in central machine room connected to serial ports of mainframe. Relatively slow (9600 bps, bits per second), mainly text oriented, single session (one process or program at a time).
Ethernet	Faster (10 Mbps, mega or million bits per second), multi-session, computer to computer, local area network. Computers and X-terminals connected via transceivers or ethernet cards and thick, thin, or twisted pair (phone) cables, which in turn are connected to the backbone. Supports text and graphics. Computers can have multiple sessions with several computers simultaneously.
Token Ring	Another type of network with capabilities similar to ethernet, but not as widely used.
Fiber-Optic	Cabling that uses light pulses traveling through fine glass fibers. Faster than other wiring systems and unaffected by electrical disturbances. Mostly used as main trunk (backbone) to connect buildings. Also called FDDI (Fiber Distributed Data Interface).
Router	Device that connects building LAN (subnet) to backbone. Acts as filter passing only data that needs to go places outside of subnet and keeping local data in and visa versa.

C. Network Software Protocols

These are the main network software programs that are used by computers.

AppleTalk	Used to connect Apple products and devices and most PostScript printers, and PC's and Unix machines that are running AppleTalk conversion software.
Netware (IPX)	Used to connect Novell file and print servers with users' PC's.
TCP/IP	Used to connect Unix machines, X-terminals and PC's running TCP/IP software. Most common utilities are telnet, ftp, lpr, NFS, and NIS.
SNA	Used to connect IBM mainframes and terminals.

D. Network Names and Addresses

IP Addresses

Every computer must have a unique IP (Internet Protocol) address assigned to it. This identifies the machine and is used by other machines to communicate with it. The IP address consists of four groups of numbers separated by periods. At UVa all addresses start with 128.143, this was assigned by the Internet authorities, and only machines at UVa can begin with these numbers. The third group of numbers identify the subnet or local area network where the machine resides. The subnet usually is associated with a building, but if there are many machines in one building it may be broken into more than one subnet. The last group of numbers is the computer number. There can be only one computer on the subnet with a particular number. An example of an IP address here at UVa is "128.143.2.20".

Domains

Since it is awkward for humans to remember long numbers, a system was developed to translate machine names to IP addresses. Computers called nameservers contain the databases that have the names and corresponding IP addresses, They are contacted by other computers whenever someone types in a computer name and they return the IP address. The names used for computers have four parts separated by periods. All computer names at UVa end with Virginia.EDU which is called the domain. The second part is the sub-domain and usually corresponds to the subnet. Subnets can have several sub-domains since they are logical groups and do not have to correspond to physical arrangements. The first part of the domain name is the name of the computer. An example of a full domain name here at UVa is "avery.med.Virginia.EDU".

E. Printers

Printing has become fairly complicated lately. Printers can be physically attached to a computer or to the network. Print jobs can be sent via the network to printers on the other side of the university, either directly, or after passing through several computers. If a print job hangs (doesn't do anything), it sometimes can take real detective work to find the problem. Printers on the network can accept jobs from many places, and it may not be easy to determine if your job is the one causing it to hang, or if there's a job

from somewhere else that's causing the problem. Here is a short description of both types.

Local — Directly connected to Mac, PC, Unix workstation, or mainframe. Can print directly to it with "Print Screen" key on PC's running DOS and from inside some programs, but only from that PC. If attached to a Unix workstation or a Mac with the proper software installed, other machines may be able to print to it.

Network — Printer is connected to network directly, not to a host computer. Most printers in this category are PostScript printers and use the AppleTalk protocol. Mac's can print to them directly, PC's running Novell Netware and Unix machines can print to them if the servers are running the AppleTalk conversion software. Some newer printers can also communicate directly with all three Mac, Novell and Unix.

F. Unix Network File Sharing - NFS

With the advent of networking and large file servers, a method was developed to allow the sharing of files between computers in a way transparent to users. This method called NFS (Network File System), provides a way to make directories on the server appear as if it were a directory on the local system. The system therefore has two parts, one part on the server and one part on the local client. (See Appendix for diagram "NFS - Network File System")

Server — The server is the machine with the disk containing the files to share. It exports the directories to all or a specified list of machines to read or read and write files. That is, it gives the machines specified permission to share the files, either to just look at them or to modify them.

Client — The client is the local machine which wants to access the files on the server. It mounts the directories from the server on a local directory, usually at bootup. That is, it requests a network connection from the server and permission to share files, then connects the remote directory on the server with a local directory.

When a user on a client machine wants to access a file which is in an NFS directory, the client machine sends a request to the server, which in turn sends a copy of the file back to the client. The user reads it or modifies the copy and it is sent back. The files remain on the server and only appear to be on the client.

Standard NFS makes a permanent network connection between the client and server. The automount or amd version is a dynamic version of NFS and only makes a network connection when an access request is made. This makes the automount version somewhat slower, but it reduces system overhead.