Information Services & Technology

A Brief Introduction to UNIX Computer Systems at Boston University

Richard Sharp
Sean Pfeiffer
Jason Heirtzler

Boston University Information Services & Technology
111 Cummington Street
Boston, Massachusetts 02215

Version 0.62 27 August 2009

Revision history:

3 September 1993: 0.30, revised and expanded, R. Sharp and S. Fisher
10 September 1993: 0.31, updated screen diagrams, S. Fisher
29 September 1993: 0.32, fixed typo on page 9, updated login diagram, R. Sharp and S. Fisher
13 January 1994: 0.33, minor text changes, R. Sharp
13 September 1994: 0.34, minor text changes, R. Sharp
21 September 1994: 0.35, update print queue information, add section on mail quotas, R. Sharp
14 February 1995: 0.36, update addresses at end, R. Sharp
25 August 1995: 0.37, general and Kerberos updates, R. Sharp
16 August 1996: 0.38, minor text changes, R. Sharp
8 January 1997: 0.39, add references to ACS1-6 on pages 2 and 3, R. Sharp
29 July 1997: 0.40, minor text updates, R. Sharp and S. Pfeiffer
7 January 1998: 0.41, convert to FrameMaker 5.5, R. Sharp
26 June 1998: 0.42, minor text updates, R. Sharp and S. Pfeiffer
6 August 1998: 0.43, minor text updates, R. Sharp and S. Pfeiffer
13 August 1998: 0.44, minor text updates, R. Sharp and S. Pfeiffer
27 August 1998: 0.45, change dialup phone number to 617-378-1111, R. Sharp
16 March 1999: 0.46, update mail quota information, R. Sharp and S. Pfeiffer
16 August 1999: 0.47, update help system information, mail clients, etc., R. Sharp and S. Pfeiffer
17 December 1999: 0.48, update dialup information, R. Sharp and S. Pfeiffer
18 August 2000: 0.49, add URLs for: Ethics policy, Kerberos password changes, e-mail forwarding and vacation automated response information, R. Sharp and S. Pfeiffer
18 August 2001: 0.50, updated to reflect replacement of X terminals with PCs and X-Win32, increased disk and mail quotas, R. Sharp and S. Pfeiffer
4 April 2002: 0.51, updated dictionary information, R. Sharp and S. Pfeiffer
23 May 2002: 0.52, updated disk quota examples, R. Sharp and S. Pfeiffer
29 August 2003: 0.53, minor updates, MyACS, SSH, printing, etc., R. Sharp and S. Pfeiffer
9 January 2004: 0.54, added sections on SilkyMail and SpamAssassin, R. Sharp
30 July 2004: 0.55, numerous updates, S. Pfeiffer and R. Sharp
31 August 2004: 0.56, quota updates, R. Sharp
22 December 2004: 0.57, quota updates, added reference to Horde, removed SPSS, R. Sharp
25 August 2005: 0.58, minor text updates, S Pfeiffer and R. Sharp
1 September 2006: 0.59, minor text updates, S Pfeiffer and R. Sharp
15 December 2006: 0.60, reformat for Word, minor text updates, R. Sharp
11 August 2008: 0.61, updated in light of new mail system, S. Pfeiffer and R. Sharp
27 August 2009: 0.62, numerous updates, S. Pfeiffer and R. Sharp

TABLE OF CONTENTS

Introduction

This document was written as the handout for introductory UNIX tutorials, but can be used by any reader to get started using UNIX. Street addresses, telephone numbers, and e-mail addresses for resources described below are listed at the end of this document under the heading How to Get More Information.

Open Access to UNIX Computing at Boston University

Information Services & Technology (IS&T) provides students, faculty, and staff at Boston University with open access to a variety of computing resources. These include:

- UNIX systems;
- public computing facilities equipped with Microsoft Windows-based workstations, Windows software, X-Win32 software for connecting to UNIX systems via the X Windows System, and laser printers; and
- a high-speed, fiber-optic campus network interconnecting these resources and linking them to the Internet.

The primary shared academic system is a cluster of large, multiuser IBM RS/6000 computers which use the UNIX operating system. This cluster, called ACS (for Academic Computing System), provides a variety of statistical and scientific software packages used in course work and research; programming languages; access to communication and information services via the international research and education network known as the Internet; and much more. The X Window System provides a graphical user interface to UNIX.

You can connect to ACS locally from personal computers and terminals connected to the Campus Network, including Windows-based workstations in the BU Common @ Mugar, located on the first floor of Mugar Memorial Library, or remotely from any location via the Internet. To open an account on ACS, visit the IT Help Center at 533 Commonwealth Avenue or the IT Help Center @ Mugar at 771 Commonwealth Avenue. You will choose a BU login name and Kerberos password, then present your current Boston University ID to complete your application. Your account should be available within twenty-four hours. Note that many departments maintain their own special purpose UNIX systems for use by their students and faculty.

What is UNIX?

UNIX is an operating system which runs on computers manufactured by Hewlett-Packard, IBM, Silicon Graphics, Sun, and many others. Linux, which runs on PCs, is a variant of UNIX. BU Linux is a free ($5 for CD media), locally customized version of Linux, configured with an emphasis on security; see the web page linux.bu.edu for more information. An operating system forms the basis for the interface between a person and a computer; it provides many basic services such as input, output, management of files on disks, and security. UNIX is to the above computer systems what DOS or Windows

is to an IBM PC, or the Macintosh OS to a Macintosh computer. Developed at Bell Laboratories in 1969, the popularity of UNIX has spread rapidly since it was first licensed to Universities around 1974 and soon thereafter for commercial use.

There are several varieties of UNIX, the most common being those based on BSD, the Berkeley Standard Distribution, developed at the University of California at Berkeley; and System V, developed by AT&T. The ACS cluster ties together several IBM RS/6000 systems running AIX, IBM's version of UNIX. Like many modern versions of UNIX, AIX combines characteristics and commands from both BSD and System V. Solaris (the version of UNIX used on Sun computer systems in many departments) also combines characteristics from BSD and System V, but in a slightly different way. Some of the examples shown in this document may be a bit different from what you see in the version of UNIX you use, but these minor differences should cause few problems.

Why Use UNIX?

Unlike the single–user operating systems used on personal computers, UNIX is a timesharing system, allowing many people to use the computer at the same time. UNIX provides many useful tools including text editors and formatters, electronic mail, and programming languages. In addition, software vendors have created thousands of application packages to address specific needs. These include scientific packages and libraries such as *Mathematica* and *IMSL*; statistical packages such as *SAS*, *Minitab*, *BMDP*, and *TSP*; sophisticated desktop publishing products such as *FrameMaker* and *Interleaf*; graphics packages such as *IslandDraw* and *IslandPaint*; spreadsheets such as *Wingz*; and many others. The disks of a large timesharing UNIX system offer convenient, centralized storage. Files are backed up on a regular basis and are always available from anywhere in the world. At the owner's discretion, files can easily be shared with others.

Connecting to a UNIX System from the Campus Network

There are two major ways to connect to UNIX systems at Boston University. The simplest method is through a single-window, non-graphical secure shell interface called SSH, available in the PuTTY program for PCs and built in as SSH for Macintosh OS X. A far more powerful and flexible method is to connect through a multi-windowed, graphical interface. Examples include X-Win32 for PCs and X11 for Macintosh, which offer a graphical interface called the X Window System. PuTTY for PCs and X11 for Macintosh are free, and X-Win32 is available for free download under BU's site license. See www.bu.edu/tech/help/unix for details on these and other programs. The next two sections describe connections using these two methods: non-graphical and graphical.

Connecting Using a Non-graphical Interface: SSH

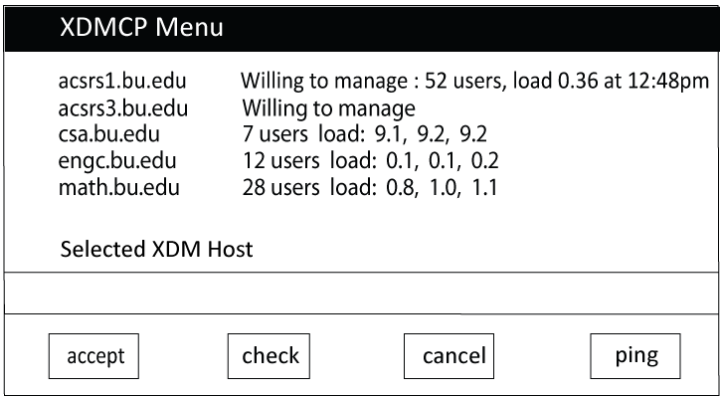
From a UNIX system, you can simply type `ssh` ***`hostname`***, where ***`hostname`*** is the name of the UNIX system to which you wish to connect. If you are using an SSH program on a personal computer, e.g., PuTTY on a Microsoft Windows-based PC or SSH

on Macintosh OS X, open that program and specify the name of the UNIX system to which you wish to connect. Details on configuring PuTTY are available at www.bu.edu/tech/help/unix. If you want to connect to the ACS cluster, you can choose any of the ACS systems, e.g., ACS1, ACS2, ACS3, ACS4, ACS5, or ACS6. Often, ACS1 is the busiest, so you may want to choose one of the other ACS systems.

Non-graphical interfaces work well even over slow connections, such as dialup modems. Dialup service to the Campus Network and the Internet is available at speeds comparable to those offered by commercial Internet service providers. For PPP service from anywhere except on-campus offices (i.e., from everywhere except numbers in the 353 and 358 exchanges), dial 617-378-1111. For PPP and non-PPP service from on-campus offices, dial 617-353-9600. For non-PPP service from all locations except on-campus offices, dial 617-378-5000. For more details on modems and connecting to machines through the Campus Network, see the website www.bu.edu/computing/dialup. Graphical interfaces work best through high speed connections. Such access is available throughout campus. For information on ResNet connections, see www.bu.edu/tech/help/resnet.

Connecting Using a Graphical Interface: The X Window System

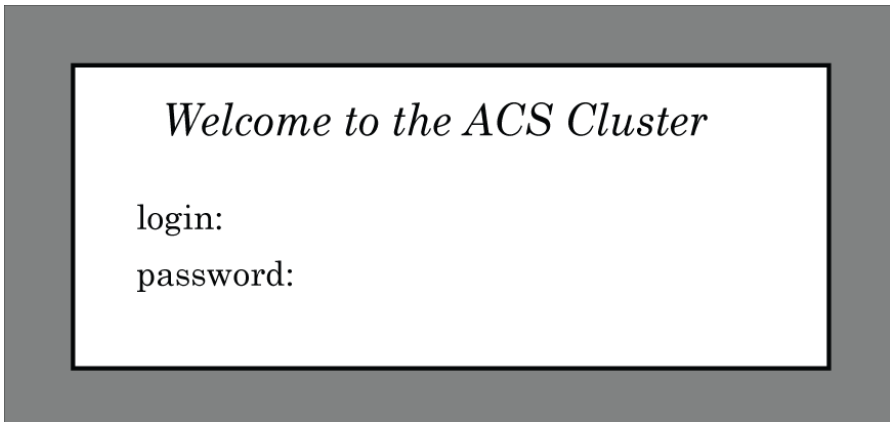
When you use the X Window System in XDMCP mode, e.g., through the X-Win32 program on a PC or X11 on a Macintosh, a list of host (computer) names will be displayed on the screen, something like the following:



Choose the host you want to use from the list and click on its name in the menu to connect to that computer. If you want to connect to the ACS cluster, you can choose any of the ACS systems, e.g., ACS1, ACS2, ACS3, ACS4, ACS5, or ACS6. Often, ACS1 is the busiest, so you may want to choose one of the other ACS systems. The hosts menu will disappear and a login window will appear which says "Welcome to the ACS Cluster".

If you accidently click connect for the wrong host and receive a login window for a computer that you can't use, type **`control-c`** (hold down the **`control`** key while typing **`c`**) to get back to the main hosts menu.

The login window should look similar to the following picture:



Logging in to a UNIX System

General Notes

Once you are connected to a UNIX system and are at the UNIX "login:" prompt, log in by entering the login name (account name) and password assigned to you. If you make a mistake while typing your login name or your password, type **`control-u`** (hold down the **`control`** key while typing **`u`**) to delete your entry and start again. See your system administrator if you have not yet received a login name and password.

```
login: account-name
password: password
```

For security reasons, your password will not be echoed on the screen when you enter it.

Most systems will display all the information described below but, depending on the version of UNIX, the order may be different. Differences specific to logins in the X Window System are described in the next section. If you make a mistake in entering either your login name or your password, you will be asked to start again by reentering your login name. If you have logged on through a non-graphic terminal, you should see a message informing you of the time and location of your last login. You should check this to make sure that you remember logging in at that time. If not, someone may have broken into your account. If you suspect unauthorized use of your account, use the *passwd* program (see the section *Changing Your Password*) immediately to change your password and report your suspicions to your system administrator. The next line on your screen contains information about the current release of the operating system. Following this line there may be a welcome message and several lines of information. These lines are the contents of the file */etc/motd* (message of the day), placed there by the system administrators to convey important news about the system.

In order for your terminal and the operating system to communicate properly, UNIX must know what type of terminal you are using. Typically, your terminal type will be automatically determined during the login process. If you see a line something like one

of the following, your terminal type has been set.

```
TERM = vt100
```
or

```
TERM = xterm
```

If the system cannot determine your terminal type, you will probably see a query similar to the following (note the parentheses).

```
TERM = (vt100)
```

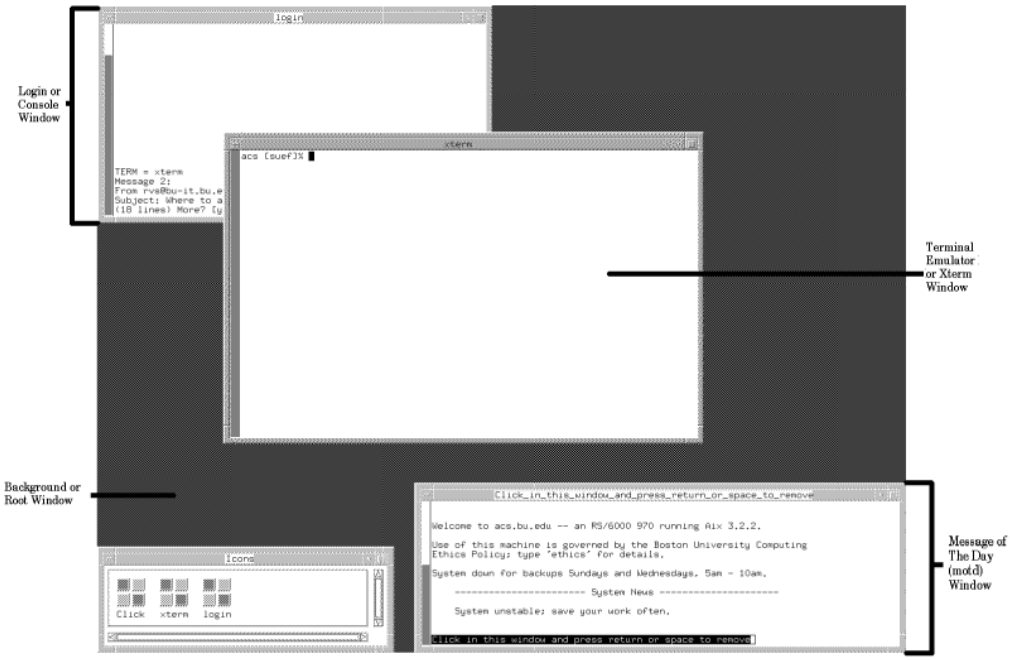
If you are using a vt100 terminal or a personal computer running a terminal emulation program, you should press **return** or **enter** to confirm this terminal type. If you are using a terminal different from the type displayed within the parentheses, you should instead type the name of your terminal, e.g., **xterm** type followed by **return**. A list of terminal types supported by each system appears in the file */etc/termcap*. When in doubt, **vt100** is usually the safest choice. The system will now accept input from and direct output to your terminal based upon the characteristics of the terminal type you have specified. Specifying the correct terminal type is very important; if you specify the wrong terminal type your screen will probably not display lines correctly, especially in full-screen applications. If you make a mistake, the simplest way to make sure that the terminal type is set correctly is to log out and then log in again to specify the correct terminal type.

One command which may be executed automatically during the login process is *msgs*. The *msgs* command checks to see if there are any new system messages. A single copy of these system messages is available for everyone to read; don't confuse these system messages with your personal mail messages. If there are new system messages, *msgs* will display the subject of the new message and ask you if you would like to read it. If you answer **y** or just press **return**, the message will be displayed on your screen. If you answer **n**, the message will not be displayed and you will not be asked about that message again. If you answer **q**, no messages will be displayed, but the next time you execute the *msgs* command you will start again at the point you quit reading messages by typing **q**.

When you have finished your session, you can log out by typing **logout**.

Additional Notes for Logging in Using the X Window System

When you have successfully typed your login name and password on an X terminal the X Window System will be started automatically. The computer will pause for a moment and a small hourglass will appear on the screen. The hourglass is similar in function to the clock on the Macintosh and means that the system is processing your request: in this case, an automated request to start the Motif window manager. After the hourglass turns into an arrow, the X Window System will appear on the screen and should look similar to the following picture:



Description of Start-up Applications in the X Window System

When you log in to a session using the X Window System, several programs are started for you automatically. Refer to the picture above to see some of the following:

A *login window*: The login window is in the upper left-hand corner of your screen. The title bar will say "login". It is partially obstructed from view. The login window is an *xterm* application. It has a UNIX prompt and you can type commands in this window. When it first appears it will display the *msgs* program described above. These messages should be read regularly as they contain important information from the system administrators. To log out of your session completely, type **`logout`** or **`exit`** or **`control-d`** (hold down the **`control`** key while typing **`d`**) in this window. When you are completely logged out, the window displaying the host list will reappear.

You should exercise caution when using the login window. If you accidentally type **`logout`** or **`exit`** or **`control-d`** in this window the system will assume you want to log out completely and will terminate any other programs you are running, even if

your work has not been saved. Because of this, the login window should usually not be used except to read the messages displayed there.

A *terminal emulator*: The window in the center of the screen is called an *xterm* window. It can be used as a normal terminal for typing commands. If the machine prompts you for a terminal type like `TERM=(vt100)` while you are using an *xterm* window, type **`xterm`**. Any UNIX command used on a vt100-type terminal should work normally in this window.

The *message of the day (motd)*: On ACS, the *motd* window displays announcements from the system managers. If the window is partially hidden by another window, click once on the window with the left mouse button to move it to the top so that you can read its contents. After reading the information there, type the space bar to make the window exit. You will see this window each time you log in to ACS. This window is unique to ACS; other UNIX computers at Boston University do NOT display this window when using X.

A *window manager*: Everything else on your screen is related to the window manager, called mwm (Motif Window Manager), which is another program currently running in your X session. A window manager controls the "look and feel" of the X session by providing the capability to create, move, resize, rearrange, iconify, and destroy windows. For more information on how to use the X Window System see the handout *Using the X Window System at Boston University*.

Special Characters and Control Sequences

UNIX interprets a number of special characters and control sequences. Among these are **`^s`** (**`control`**–**`s`**), **`^q`**, **`^u`**, **`^d`**, and the **`delete`** key. Typing **`^s`** (holding down the **`control`** key while typing **`s`**) will stop the output from being displayed on your screen. It has the same effect as pressing the **`hold screen`** key on a vt220 terminal, and is useful to prevent long listings from flying off the screen before you have a chance to read them. You may accidentally type **`^s`** and then think that your terminal is broken, since no further output is displayed on the screen and nothing you type is printed on the screen. Typing **`^q`** will resume the output to the screen and allow input to be printed on the screen again. This is an easy condition to check before concluding that something is wrong with the computer or your terminal.

Another useful character is produced by the **`delete`** key. At the UNIX command prompt and within most programs, pressing the **`delete`** key will erase the character immediately to the left of the cursor position. At the command prompt, typing **`^u`** (**`control`**–**`u`**) will erase all the characters from the cursor position back to the beginning of the command line. This is convenient when you have typed in a lengthy command, found a mistake near the beginning or changed your mind and want to start over. A **`^d`** indicates the end of input and, if typed at the login command prompt, indicates that you wish to terminate the session and log out.

Changing Your Password

Your password provides the primary security for your files. You should choose a password which is difficult to guess, is not a proper name or in any dictionary (on-line dictionaries are often used in password breaking programs), and includes at least one digit or punctuation mark and a couple of capital letters (UNIX is case sensitive).

You should not give your password to anyone! The ACS system uses a special password called a Kerberos password. Your Kerberos password is the key to easy access to University records and a growing number of services, including your ACS account and Ph. If you are a student, you can use your Kerberos password with the Student Link to display your class registration, grades, transcripts, and financial records. You can also view and update personal information, such as your address and telephone number. Anyone who knows your Kerberos password gains access not only to your ACS account but also to your personal, academic, and financial records. The Student Link is available on the Web at www.bu.edu/studentlink.

You are responsible and liable for all use of your accounts. Giving your password to anyone or using anyone else's password is prohibited by Boston University's Conditions of Use and Policy on Computing Ethics. Any violation may be cause for disciplinary action. See www.bu.edu/computing/ethics to review the responsibilities you assume when you create a password or use Boston University computing facilities.

To change your Kerberos password at any time, visit the website www.bu.edu/computing/accounts/kerberos. You will be asked several questions to make sure that you are the account owner, then prompted to enter a new password. You will be asked to retype the new password to ensure that there were no spelling errors. None of these passwords will be echoed on your screen.

Using Man, Apropos, Web-based Help, and Msgs

As you read about UNIX commands in this document, you can learn more by reading the UNIX manual pages. UNIX manual pages are available on-line. Whenever you have a question about a particular UNIX command, you can use the *man* utility to display the manual page for that command on your screen. If the manual page is longer than one screen full, it will be displayed one screen at a time. To view the next screen, press the **space** bar; to view the previous screen, type **b**; to advance one line at a time, press the **return** key. To quit before coming to the end, type **q**. As an example, try typing **man mkdir** at the UNIX command prompt to view the manual page for *mkdir*.

```
acs4% man mkdir
```

NAME

mkdir – make a directory

SYNOPSIS

mkdir dirname ...

DESCRIPTION

Mkdir creates directories. Standard entries, '.', for the directory itself, and '..' for its parent, are made automatically.

The current **umask**(2) setting determines the mode in which directories are created. Modes may be modified after creation by using **chmod**(1V).

Mkdir requires write permission in the parent directory.

SEE ALSO

chmod(1V), **rmdir**(1), **rm**(1)

The first line gives the name of the command and a brief statement of its function. The SYNOPSIS section shows the command format, usually the command name followed by a number of arguments and possibly some optional qualifiers. In this case, the synopsis shows that in order to create a directory you would type **`mkdir`** followed by one argument: the name of the directory you wish to create. Other commands might list optional qualifiers (often preceded by a "–") between the command and its arguments to allow more control over the results. The DESCRIPTION gives more detailed information about the command. Other command names used in the description are usually highlighted and their manual section given in parentheses. The SEE ALSO section lists other related commands or topics which you might want to read about, again using *man*. Try typing **`man ls`** to learn more about the *ls* command. For more information on *man*, type **`man man`**.

If you don't know the name of a particular command, but you want to find the names of commands which relate to a given topic, you can use *apropos* to get a list of relevant manual page entries. You could then use *man* to get details on any of the commands that seemed interesting. Here is a partial listing of the entries returned for "directory".

```
acs4% apropos directory
cd (1)- change working directory chdir
ls (1V)- list contents of directory
mkdir (1)- make a directory
pwd (1)- print working directory name
rmdir (2)- remove a directory file
```

For extensive information on local topics, see the Web-based help system at www.bu.edu/tech/help/acs.

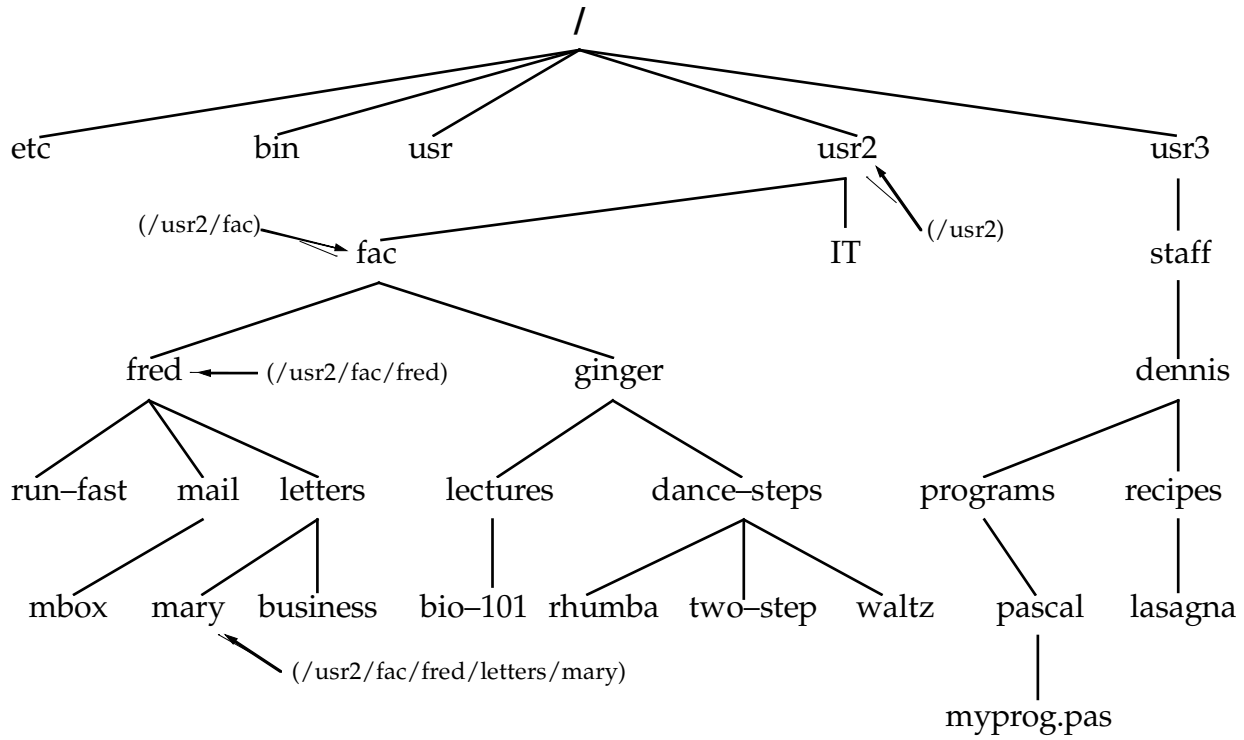
More timely information, such as a scheduled shutdown or a change to the local environment, is placed in messages which can be read using the *msgs* command. This command is usually executed automatically when you log in, and you are given the opportunity to read new messages at that time. You can always go back and read older messages by specifying the number of the message, or use a negative number to go back that number of messages from the final message. For example, typing **`msgs 4`** would begin by offering to display message number four and then offer to display each following message in sequence. Typing **`msgs -2`** would offer to display the last two messages. Type **`man msgs`** for more details.

File Structure: ASCII Files and Others

ASCII (American Standard Code for Information Interchange) files are plain text files containing only simple printable characters with no special formatting. ASCII files can be created with text editors such as *Emacs, vi, or Pico,* and can be displayed with simple UNIX utilities such as *cat* and *more* and can be sent to others using electronic mail. Files produced by word processors, spreadsheets, and database or other programs are non–ASCII files; they contain special non–printing codes and cannot be edited or displayed with simple UNIX utilities. Editing these special files with a text editor will probably destroy them beyond all repair. Object files produced by compilers and executable images produced by the linker are other examples of special files which should not be approached with the standard UNIX display, printing, and editing utilities.

Directory Structure

UNIX uses a tree directory structure, much like MS–DOS. The top level of this upside-down tree is denoted by "/" and is called the root. Note that UNIX always uses "/" (slash) to denote directory paths where MS–DOS uses "\" (backslash). Levels immediately below are indicated by names like */usr* or */bin*. Each level below adds another slash and another name. When you log in, you are automatically placed in your home directory, named something like */usr2/fac/account–name* or */u6/ugrad/account–name.*



This diagram represents a simplified view of a typical directory tree. It shows someone named "fred", with */usr2/fac/fred* as his home directory, under which he has subdirectories named *notes*, *mail* and *letters*. In his letters subdirectory he has two files, one a letter

to Mary and the other a business letter. The *mail* directory contains his *mbox* file, which in turn contains all the mail messages he has saved. The *notes* subdirectory is empty at this time. Fred's home directory can be viewed as the root of his own subtree within the file system.

When Fred logs in to the system, he is automatically placed in his home directory, so his home directory is said to be initially his "current working directory". Any files he references or creates will be in this current working directory, unless he specifies another directory. For example, if Fred logs in and creates a file named *stuff*, it will appear in his home directory, since that is his current working directory when he logs in. As long as he does not change his current working directory, he can simply refer to the file as *stuff*, and the system will automatically prepend (add at the front) the current working directory, */usr2/fac/fred*, to the file name *stuff*, to reference */usr2/fac/fred/stuff*. If Fred wants to access the file *mary* in his *letters* subdirectory, he will have to change his current working directory to be the *letters* subdirectory (*/usr2/fac/fred/letters*), or prepend a full or partial directory path to the file name. A full path name always starts with a "/" and describes the entire path to the file from the root of the entire file system tree. A partial path name references the file from the current working directory. If Fred has his home directory as his current working directory, he could refer to the file *mary* either with the full path name, */usr2/fac/fred/letters/mary*, or the partial path name relative to his current working directory, *letters/mary*. In the case of the partial path name, which does not begin with a "/", the system prepends the current working directory, */usr2/fac/fred* to the partial path name/file name, *letters/mary*, and references */usr2/fac/fred/letters/mary*.

You will learn how to change current working directories in the section *Looking at Directories and Files*. You can read in the section *Using Man, Apropos, Web-based Help, and Msgs* about the *mkdir* command, used to create subdirectories under your home directory. Subdirectories are very useful for organizing your files.

Looking at Directories and Files

To print the name of your current working directory, type **`pwd`**. To change your current working directory to be another directory, use the *cd* command. For example, if Fred's current working directory is his home directory, he could change to his *letters* directory by using *cd* to specify either the relative path name or the absolute path name of the letters directory:

```
acs3% cd letters
```

or

```
acs3% cd /usr2/fac/fred/letters
```

To change from any directory back to your home directory, type **`cd`** alone, specifying no directory.

Before proceeding, it is necessary to say a few words about accounts. Each account has associated with it two numbers: the user id (uid), which is unique, and the group id

(gid), which is shared with other accounts in that group. It is not necessary to know these numbers (in fact, when displayed they are usually converted from a number to a name), but you should be aware of any other members in your group. A group may be composed of several people working in one office, a professor and her graduate students, or any other group of people who wish to share files without making those files available to others outside the group. On the ACS cluster, the default groups are often very large, e.g., all undergraduates: *ugrad*. You can control access to your files by members of your own group and by others.

To display a list of files in the current directory, type **`ls`**. The *ls* command has many options, but the most useful ones for beginners are the "–l", "–g", "–d" and "–F" options. The "–l" option produces a long listing, the "–g" option displays the group ownership of the files, the "–d" option specifies that *ls* should provide information on the directory itself rather than the files in that directory, and the "–F" appends symbols to the end of certain filenames to show that they are special in some way. Note that "–F" must be capitalized. These options can be used separately or combined following a single "–".

Note that in AIX, the version of UNIX used on ACS, the "–l" option also causes the group ownership to be displayed. In AIX, if the "–g" option is specified, the long listing will be displayed except that the owner column will be omitted. The examples shown below are from the AIX version of *ls*. If you are not using AIX, simply add the "–g" flag to see the same result.

```
acs3% pwd
/usr2/fac/fred

acs3% ls
letters        mail          run-fast

acs3% ls -lF
total 33
drwxr-xr-x  1 fred   dancers    512 Sep 25  15:14  letters/
drwx------  2 fred   dancers    512 Oct 19  16:27  mail/
-rwxr-x---  1 fred   dancers  31024 Sep 02  13:02  run-fast*
-rwxr-----  1 fred   dancers  31024 Sep 02  13:02  run-fast.c
```

The long listing above tells us about the files in the current directory, */usr2/fac/fred*. The first ten characters are codes which describe each file and the permissions or modes which have been set for it. If the first code character is a "–", it indicates that the entry is a regular file. If the first code is a "d", the entry is a directory. The next nine characters are arranged in three sets of three. The first set shows permissions for the owner of the file, the second shows permissions for the group, and the third shows permissions for the rest of the world. An "r" means that the file can be read, a "w" that the file can be written or modified, an "x" means that a file can be executed (if it is a program) or searched (if it is a directory), and a "–" in any of these positions means that the corresponding permission is denied.

Using this code, the first file, *letters*, with a first code character of "d", is a directory. The "–F" on the *ls* command line has also shown the file to be a directory in a more graphic manner: a "/" has been appended to the file name. Note that the "/" is not actually a

part of the file name, but a graphic indication that the file is a directory. It may be read, written, and searched by the owner, read and searched by people in the "dancers" group, and read and searched by everyone else who has an account on the machine. The file is 512 bytes (characters) in size and was created at 15:14 on 25 September.

The next file, *mail*, has a "d" as the first character in the long listing, indicating that it is also a directory. The owner can read, write, and search for files in the *notes* directory. Access is completely barred both to members of the "dancers" group and others.

The "–F" option has placed a "*" after the filename *run–fast*, indicating that it is an executable program. Again, the "*" is not part of the filename; typing **`rm run-fast*`** would delete any and all files in this directory which begin with the characters "run–fast"! The file *run–fast* may be read, written, and executed by the owner, and read and executed by members of the "dancers" group. All access to others has again been denied.

Chmod and Chgrp

You can change the protection modes on a file which you own by using the *chmod* utility. When using *chmod*, you can specify the mode in either absolute or symbolic form. The symbolic form is easier to understand, and will be used in this example. The owner of the file is referred to as "u" (for user), the group as "g" and others as "o". Permissions are symbolized by "r" for read, "w" for write, and "x" for execute. Permissions can be granted by specifying "+" and removed by specifying "–". Suppose Fred wishes to change the protection mode on the file *run–fast* so that everyone who is not in the "dancers" group may read and execute the file. He could use the following command to grant these two additional permissions.

```
acs4% chmod o+rx run-fast
```

Now suppose Fred decides that run–fast has potential for a new movie and wants to protect the file from everyone but himself. He could use

```
acs4% chmod g-rx,o-rx run-fast
```

You can change the group ownership of a file using the *chgrp* command. You must belong to the specified group and be the owner of the file. Suppose that Fred has sold the movie rights for *run–fast* to a film studio and wants to share the file with the film's producers. In addition to being in the group "dancers", Fred shares with the producers another group, "moguls". Fred uses *chgrp* to change the group ownership of the file to "moguls", then grants the group read and execute permission to the file.

```
acs4% chgrp moguls run-fast
acs4% chmod g+rx run-fast
acs4% ls -l run-fast
-rwxr-x---  1 fred    moguls   31024 Sep  2  13:02  run-fast
```

For more detailed information on *ls*, *chmod* and *chgrp*, use the *man* utility, described in the section *Using Man, Apropos, Web-based Help, and Msgs*.

Creating and Removing Directories: mkdir, rmdir, rm –r

Directories are created with the UNIX command *mkdir*. The following example will create a directory named *newdir* under the current directory.

```
acs4% mkdir newdir
```

If a directory contains no files, it can be removed with the UNIX command *rmdir*.

```
acs4% rmdir olddir
```

To remove a directory and all the sub–directories and all the files below that directory, use the "–r" (recursive) flag on the UNIX command *rm*.

```
acs4% rm –r oldjunk
```

For more information, see the UNIX manual pages for *mkdir*, *rmdir*, and *rm*.

Viewing and Printing ASCII Files

You can view ASCII files by typing **`cat filename`**, which will simply display the contents of the file on your screen. If the file is longer than one screen-full, it is often more convenient to use **`more filename`**, which will display the file one screen at a time. To view the next screen, press the **`space`** bar; to view the previous screen, type **`b`**; to advance one line at a time, press the **`return`** key. To quit, type **`q`**. To print a plain ASCII file on a postscript laser printer, use **`lpr`** ***`filename`*** or **`enscript`** ***`filename`***; to direct the output to a specific printer, use either **`lpr –P`*`printername filename`* or **`enscript –P`*`printername filename`**. More information is available on each of these commands, especially *enscript*, from *man* (see the section *Using Man, Apropos, Web- based Help, and Msgs*). You can read more about printing through BU's MyPrint facility at www.bu.edu/tech/help/myprint.

Manipulating Files: cp, mv, rm

All files, whether ASCII or special, can be copied, renamed, moved to a different location, and removed. To copy (make an exact duplicate of) a file, use the *cp* command.

```
acs2% cp file1 file2
```

where ***`file1`*** is the name of an existing file and ***`file2`*** is the name for the new copy of that file. The original file will remain unchanged and a copy will be placed in ***`file2`***. You can also copy a file to a different directory:

```
acs2% cp file directory
```

where ***`file`*** is the name of an existing file and ***`directory`*** is the name of a directory where the new file will be copied. The new file in the specified directory will have the same name as the original file.

Files can be renamed or moved to a new location in a similar manner, using the UNIX command *mv*.

```
acs2% mv filename1 filename2
```

moves, or renames, ***filename1*** to ***filename2***.

```
acs2% mv filename1 directory2
```

moves the file named ***filename1*** to a new directory, specified by ***directory2***. Files can be removed or deleted from the system entirely using the UNIX command *rm*.

```
acs2% rm filename
```

More details can be found in the UNIX manual pages for *cp*, *mv*, and *rm*, available through the UNIX *man* utility (see the section *Using Man, Apropos, Web-based Help, and Msgs*).

The C–shell: Basic Features

Once you are logged in, you should see the shell prompt (the UNIX command prompt). The shell, which serves as the command interpreter, is usually the C–shell (*csh*) or an enhanced version of the C–shell called the TC–shell (*tcsh* – see below). The prompt will probably be a "%" sign, possibly preceded by the local hostname, possibly also preceded by the name of your current working directory enclosed in square brackets. This shell is your interface to the operating system: your means of communicating with UNIX. It is this shell which will actually interpret your commands, locating the commands you specify and passing these utilities the parameters and qualifiers you specify by typing on the command line. As stated in the manual page for *csh*, "It provides a number of convenient features for interactive use... including filename completion, command aliasing, history substitution, job control, and a number of built-in commands.... The C–shell provides variable, command and filename substitution."

The C–shell is a powerful, flexible, and complex command interpreter which can be used both interactively and in shell script programming. Despite its complexity, the beginner may use it easily, hardly aware of its existence. This section gives only a few details on the C–shell. See the section titled *The C–shell: Some Advanced Features*, and other references such as *The UNIX C Shell Field Guide* by Gail and Paul Anderson (Prentice–Hall 1986), or the on-line UNIX manual pages on *csh* for more information on how to use the various features of the C–shell.

A Variation on the C-shell: the TC-shell

The TC–shell is the default shell on ACS. All comments made in this document regarding the C–shell also apply to the TC–shell. Although there are many advantages to the TC–shell, the most obvious are the capabilities for using the arrow keys to recall previous commands, and for using common *Emacs* commands to recall and edit shell commands (one more reason to learn the *Emacs* text editor). The TC-shell will read initialization commands from the .cshrc file, just like the C-shell. While most people find this convenient and do all their shell customizing in the .cshrc file, you should be aware that if your home directory contains a *.tcshrc* file, the TC–shell will read this file instead of the *.cshrc* file. To avoid confusion: unless you have a specific need to switch

back and forth between the C–shell and the TC–shell, do all of your customizing in your *.cshrc* file and don't create a *.tcshrc* file. For more information on the TC–shell, type **`man tcsh`**.

History Substitution in the C–shell

The C–shell keeps a record of the commands you have executed and you can recall these commands for re–execution or modification and subsequent execution. To display a list of recently executed commands, type **`history`** at the C–shell prompt. To repeat execution of the last command, type **`!!`** (pronounced bang–bang in UNIXese). To repeat the execution of command 47 in the *history* list, type **`!47`**. Suppose you enter the following command, mistakenly typing **`myfail`** for **`myfile`**:

```
acs2% ls myfail
```

Instead of retyping the entire command, you can simply modify the erroneous part of the command with

```
acs2% ^ail^ile
ls myfile
```

Note that in this case, **`^`** is the actual "up-caret" character, not a designation for the "control" key. The C–shell will reprint the corrected command, substituting "ile" for "ail", and then execute the command. There are many other tricks for *history* substitution; see the on-line UNIX manual pages for *csh*.

Filename Substitution in the C–shell

The C–shell also provides several metacharacters for filename substitution. Of these, the most common are the asterisk (*) and the "?" sign. The asterisk (*) will match any (zero or more) characters, while the "?" sign will match any single character. As an example,

```
acs5% ls man*
```

will list any filenames beginning with "man", while the following will list any files beginning with "ma", followed by any single character, followed by "n".

```
acs5% ls ma?n
```

Be very careful when using wildcards when deleting files with the *rm* command (see the section *Manipulating Files* for information on the *rm* command). The following will delete all files in the current directory beginning with "m"!

```
acs5% rm m*
```

The Tilde Metacharacter

Another useful C–shell metacharacter is the tilde (~), or "twiddle", as it is pronounced in UNIXese. The tilde is shorthand for, "the home directory of". So, ~fred refers to "the home directory of fred", and a lone ~ refers to your own home directory.

Passing Special Characters through the C–shell

Placing a backslash (\) immediately before a character tells the C–shell not to interpret that character as special, but to pass it on to the program exactly as it was entered. Multiple special characters can be passed through the C–shell by enclosing them in single or double quotes. See the section on *Dictionaries* for examples.

Connections: An Overview of the Campus Network

Boston University's Campus Network employs the latest technology to route communications among computing systems throughout the Charles River Campus. Thousands of ports supporting communications rates at more than 10 million bits per second are interconnected via optical fiber and high-speed routers. Additional links connect the Charles River Campus to the Boston University Medical Campus, Harvard, and MIT. The Campus Network is also connected to the Internet and all major regional, national, and international research and educational networks, providing students and faculty with electronic access to people and resources throughout the world.

Using SSH to Log in to Other Machines

If you have an account on more than one computer, you can use *ssh* to establish a login session from the first computer to a second. As an example, to establish a login session on acs from a login session on crsa, you can use the following:

```
crsa% ssh acs.bu.edu
```

SSH (secure shell) encrypts all traffic over the connection, including your password. You can download free implementations of SSH for Windows-based PCs and Macintosh computers, see www.bu.edu/tech/help/unix. See the manual page on *ssh* for more details.

Communication: E-mail Programs

Access to e-mail is one of the most popular reasons for using a computer; everyone from novice to expert comes to depend upon it. You can exchange mail with people on your system, other systems on campus, and millions of systems around the world. There are several mail programs to choose from on ACS. The following descriptions of supported mail programs are provided to help you choose one that suits your needs.

- Web-based access to ACS e-mail: You can access your ACS e-mail from anywhere in the world, anytime you have access to a Web browser. Our Web-based mail interface, Horde, requires no configuration for basic use, and is available at www.bu.edu/webmail.
- Personal computer e-mail clients: UNIX systems such as ACS support the IMAP protocol, allowing you to choose from a number of compatible personal computer e-mail clients. These clients provide a graphical user interface for

reading and sending e-mail and offer various options for downloading mail from your UNIX account to your personal computer and/ or storing it on the UNIX server. For more information on how to choose the right e-mail client for your needs, see www.bu.edu/tech/help/email or talk to someone at the IT Help Center.

- Pine: Pine is a menu-driven electronic mail program for UNIX and IBM PCs and compatibles. The UNIX version is available on ACS. Features of Pine include a menu-driven interface; on-line, context-sensitive help; and an address book. To use Pine, type **`pine`** at the system prompt. To view on-line documentation, type **`man pine`** at the system prompt or view the help screens within Pine. A free handout is available, *Using Pine*. NOTE: when you use Pine to send mail to multiple recipients, you must separate the addresses with commas rather than just spaces; otherwise, only the first recipient will receive the mail.

Pine has the ability to call your choice of editor to compose messages. In addition to Emacs and vi, we have installed another editor that you may want to try: Pico (type **`pico`** to run, **`man pico`** for information). Pico looks much like a very simple version of Emacs, with the commands appearing at the bottom of the screen. By default, Pine uses a version of Pico as its editor.

Mail Quotas on ACS

To help ensure that ACS disk space does not become full, an automated mail quota system has been implemented. The system will send you warning messages should your mail begin to approach the quota limits. To check your current use of mail quota, go to the website www.bu.edu/computing/myacs.

Filtering Spam

Boston University uses SpamAssassin to tag suspected spam messages as they pass through our central mail gateways. You can use these tags to filter out unwanted mail. For more information, see www.bu.edu/tech/help/email/spam.

Using Ph, the Online Directory, and LDAP to Find Account Names

The Ph program can be helpful when you want to send mail to someone at Boston University, but you don't know their account name. You can also use the Ph program to get information about people at Boston University, even if they don't have a computer account. At the ACS system prompt, type **`ph`** ***`firstname lastname`*** to view the information. Type **`man ph`** for many more details about what you can do with Ph.

There is also a Web interface to the online Directory at www.bu.edu/directory and an LDAP server at ldap.bu.edu. If you use the LDAP server, you must configure your search base to be "o=Boston University, c=US" (without the quotes).

Forwarding E-mail; Setting an Automated Vacation Response

You can forward your ACS e-mail by specifying a forwarding address at the website www.bu.edu/computing/email/forwarding. You can also set up an automated e-mail response while you are away on vacation, at www.bu.edu/computing/email/vacation.

Transferring Files

You can use s*ftp* (secure file transfer program) to transfer files between computers. See www.bu.edu/tech/help/ftp for information on free sftp programs for Windows-based PCs and Macintosh computers. Below is an example of a file transfer using s*ftp* initiated on bio, a UNIX system, to copy a file from ACS, another UNIX system. At the "sftp>" prompt type **`help`** for a list of commands or **`help`** ***`commandname`*** for details on a specific command, or type **`man sftp`** on a UNIX machine for more details.

```
csa% sftp acs
Connecting to acs...
fred@acs's password: password
sftp> ls
200 PORT command successful.
150 Opening data connection for /bin/ls.
Mail
News
datafile
sftp> get datafile
Fetching datafile
sftp> bye
csa%
```

Another way to transmit an ASCII file is to mail the file:

```
acs5% mail username@host < UNIX-file-name
```

See the section titled *The C–shell: Some Advanced Features* for an explanation of the "<" (input redirection) sign.

Disk Quotas

Many UNIX systems on campus, including ACS, limit the amount of disk space each person can use. The easiest way to view the current status of your disk, mail, and print quotas is to go to www.bu.edu/computing/myacs.

If you are logged on to ACS, you can see your disk quota by typing **`quota`**:

```
acs2% quota
User       Disk Quota         Curr.Used          Balance    F.S.
fred           50000K            22042K           27058K    /u3
```

The output of this command shows that Fred has used 22.042 Megabytes of disk space. If Fred adds files and uses more space, eventually totalling 50 Megabytes (the "Disk Quota" column, above), he will receive a system message saying that he has exceeded his quota. For more information on the *quota* command, type **`man quota`**.

You can also determine the amount of disk space you are currently using with the disk

usage (*du*) command. The "s" switch specifies to report only the sum for all your files rather than information for each file; the "k" switch specifies to express that number in 1024-byte blocks (Kbytes); and the "~" at the end specifies to report information on all files in your home directory and all subdirectories below it.

```
acs2% du -sk ~
1442    /u3/ugrad/fred
```

Fred's files are using 1442 Kilobytes of disk space. Divide the number of Kbytes by 1000 to determine Megabytes. 1442 Kbytes is equal to 1.4 Mbytes, which is safely under the current 25 Mbyte quota for undergraduate students.

You can run the program *disk_clean* (type **`disk_clean`**) to help you find files which are likely candidates for deletion. The *disk_clean* program does not delete files, it simply looks through your directories for files which are usually considered temporary or old. Examples include core files (large pictures of memory which you will never need unless you plan to debug programs), editor backup files (e.g., *Emacs* saves the old version of any file edited under the name *filename~*), old versions of files used by news readers and mail, and intermediate files created by programs such as T_EX and $LaTeX$. If you need help, please talk to someone at one of the Consulting Services Help Desks.

You can conserve disk space by using either the *compress* or *gzip* program. Type **`man compress`** or **`man gzip`** for details. If, after using *compress* or *gzip*, you find it impossible to store your work within your disk quota, you can request an increase at www.bu.edu/computing/myacs.

Printing

High speed printing is available in the BU Common @ Mugar at 771 Commonwealth Avenue. For complete information, please see www.bu.edu/tech/help/myprint.

Initial Setup: Dot Files

When your account is set up, you are provided with a few files which set up your environment for you. These files, e.g., .login, .cshrc, and .emacs, are called "dot" files, since they begin with a dot. When you log in to the system, the command interpreter, usually the C–shell (*csh*) or an enhanced version of the C–shell called the TC–shell (*tcsh*), first executes any commands contained in your *.cshr*c file and then any commands in your *.login* file. You need not be concerned with the contents of these files initially, but at some point you may discover that you want to change something in one of these files. These files may be listed by typing **`ls -a`**, (see *Looking at Directories and Files*) and modified using an editor such as Emacs. Be sure you understand exactly what you are doing before you edit these files, since mistakes can cause you subtle problems! An editing mistake in one of these files could prevent you from logging in again for a second attempt.

The C–shell: Some Advanced Features

This section briefly describes two slightly more advanced features of the C–shell (and TC–shell): job control and input/output redirection. See the publications mentioned in the earlier section the C–shell, or the on-line UNIX manual pages on *csh* and *tcsh* for more information.

Job Control in the C–shell

In the beginning, you will probably be satisfied to execute one job at a time, completing each task before moving on to the next. UNIX is capable of running many jobs simultaneously on your behalf (at least, it will appear to run these jobs simultaneously) and you can control these jobs with the C–shell. Since you can only communicate with one job at a time, your other jobs must either be stopped or running in the background (similar to running in batch mode on some other operating systems). The C–shell offers the following special characters to control jobs. Remember that the "^" character here refers to the **control** key, so **^z** means **control-z**.

- **^z** Suspend the current job (does not kill the job, it can be resumed)
- **^y** Same, but don't suspend the job until the program tries to read the terminal
- **^c** Send an interrupt signal (usually kills the job, but some programs trap this)

Suppose you begin a long C compilation and then decide you want to check your files:

```
acs5% cc myfile.c –o myfile                  invoke the C compiler
^z                                           suspend the C compiler
Stopped                                      C–shell acknowledges job stopped
acs5% ls                                     do a directory listing
file1   myfile.c   oldfile                   names of files in your directory
acs5% jobs                                   check to see what jobs you have
[1] +Stopped      cc myfile.c –o myfile      just one job, which is suspended
acs5% %1                                     resume the cc job (job 1)
```

Note that the C–shell built-in command *jobs* shows you all the jobs you have running from this shell and whether they are stopped or running in the background. You can run jobs in the background by appending an ampersand, "&", to the command line:

```
acs5% cc myfile.c –o myfile &                run C compiler in the background
acs5% jobs
[1] Running       cc myfile.c –o myfile      shows compiler running
acs5%
[1] Done          cc myfile.c –o myfile      C–shell notifies job completed
acs5% cc other.c –o other                    start another job in foreground
^z                                           suspend it
Stopped                                      C–shell acknowledges job stopped
acs5% jobs                                   list all jobs for this shell
[1] +Stopped      cc other.c –o other
acs5% bg %1                                  continue job 1 in the background
acs5% kill %1                                kill job 1
[1] Killed        cc other.c –o other
```

As the example illustrates, you can resume a job by typing **%*n*** where ***n*** is the job number displayed within square brackets by the command *jobs*. You can bring the current job to the foreground by typing **fg**, or foreground or background any job with

fg %*n* and **bg %*n***, respectively. You can kill a job, either stopped or running, by typing **kill %*n***. Note that the number shown by *jobs* is not the real process id, it is just a convenient handle for numbering jobs created under a particular shell.

You can often regain control of a "hung" terminal by logging in to the same account on another terminal and using the UNIX command *ps* to find the real process id of the runaway job and then killing it from the second terminal.

```
acs2% tty
/dev/ttyp2
acs2% ps
   PID   TT       STAT     TIME     COMMAND
23929    p2       S        0:05     -csh
28466    p2       R        0:00     ps
17832    04       R        9:37     lisp     job under same account, on tty04
acs2% kill 17832                             kill it by PID number
acs2% ps
23929    p2       S        0:05     -csh
28479    p2       R        0:00     ps       PID 17832 is gone, terminal free
```

If you had an editing session in progress on the other terminal, you could try using the "–HUP" flag, as in **kill -HUP 17832**. This will simulate "hanging up the phone" and most editors will try to preserve any changes you made to the file.

Should you try this and the job does not die, as a last resort use the "–KILL" flag:

```
acs2% kill -KILL 17832
```

Using the C-shell to Redirect Input and Output

Input and output can be redirected using special C–shell metacharacters. Most UNIX commands read from the standard input and write to the standard output. In most cases, standard input is the keyboard and standard output is the terminal screen. It is often useful to redirect the output to a file, where it can be preserved and studied. As a simple example, the *date* command will print the current date and time on the screen. The following command uses the ">" metacharacter to redirect the output, printing the current date and time into a new file named *stuff*.

```
acs2% date > stuff
```

If the C–shell variable "noclobber" is set, any previously existing file named *stuff* will not be destroyed. Instead, you will get a message, "stuff: File exists." Otherwise, the old file *stuff* will be destroyed, even if it contains your entire Ph.D. thesis, and replaced by the new file, containing only the date and time. You can, and probably should, set the "noclobber" variable by including the following line in your *.cshrc* file (this is already done for you on the ACS cluster).

```
set noclobber
```

The next command uses two consecutive ">" metacharacters to append the current date and time to the end of an existing file, rather than creating a new file.

```
acs2% date >> oldstuff
```

The following will "pipe" the output of the *date* command to the *lpr* (lineprinter) com-

mand, which will in turn print the current date and time on the printer.

```
acs2% date | lpr
```

Although used less often, input can also be redirected, using the "<" metacharacter. For an example, see the earlier section of this document titled *Transferring Files*.

Application Software

Some of the products described below, such as the text editors and the *troff* document formatter, are available on all UNIX systems on campus. Others are third-party software application programs which do not come with the operating system, but must be purchased from other vendors. Each department owning a computer determines what additional applications are needed on that computer and purchases them. The sections below describe a few applications which may be of general interest, but not all these applications are available on all computers. Talk to your local system administrator or look in the help files on your computer to determine which applications are available to you. Documentation for most of these applications, including tutorials written by the vendors, can be borrowed from the IT Help Center @ Mugar.

Dictionaries

The complete Oxford English Dictionary, Second Edition, is available to the Boston University community on the Web at www.bu.edu/oed2/. Merriam-Webster's Collegiate Dictionary (Tenth Edition) – part of Britannica Online – is available at www.search.eb.com.

Word Processing

At least two word processing applications are in use on UNIX systems around campus: *IslandWrite* and *WordPerfect*. *IslandWrite* works in close harmony with two other packages for producing graphics: *IslandDraw* and *IslandPaint*. These packages are reminiscent of some which run on the Macintosh. The "Island trio" is available on ACS. The UNIX version of *WordPerfect*, available on some departmental machines, should look familiar to anyone who has used *WordPerfect* on a personal computer.

Text Editors

Editors are used to enter and manipulate text in plain text (ASCII) files. In an ASCII file, all the characters are visible and printable, whereas files produced by word processors usually include many non–printing control sequences that contain formatting information for the document. An ASCII file can be created by one editor, then edited later with another editor, but it may be difficult or even impossible to transfer files between different word processing programs.

At least two text editors are standard on all local UNIX systems: *vi* and *GNU Emacs*. Of these, the more powerful and recommended is *Emacs*. *Emacs* can do many kinds of text

manipulation and, after you've gained some experience, you can even write your own special *Emacs* programs (called macros) to customize the editor. Start *Emacs* by typing **`emacs filename`**. *Emacs* has an on–line tutorial, which can be invoked from the system prompt by typing **`teach-emacs`**, or from within *Emacs* by typing **`control-h t`** (hold down the **`control`** key down while typing **`h`**, then release the **`control`** key and type **`t`**). This interactive tutorial will give you the fundamentals in a few minutes. Another useful command for those new to Emacs is **`control-x p`**, which will toggle on and off the display of a basic command list at the bottom of the Emacs window. Type **`man emacs`** for a brief synopsis; the standard reference is *The GNU Emacs Manual* (Free Software Foundation). There is also an excellent book which some people find more useful for learning and using *Emacs*, *Learning GNU Emacs (3rd Edition)* by Debra Cameron, Bill Rosenblatt, and Eric Raymond (O'Reilly & Associates, Inc. 2004, 534 pages).

ACS has one other simple editor, *Pico*. To run *Pico*, from the ACS command prompt, type **`pico`**; for details, type **`man pico`**. Pico displays a list of commonly-used commands at the bottom of the screen.

Document Preparation Utilities

Several packages are available for more sophisticated document preparation, often called electronic publishing. *Troff* is the old standard on UNIX systems and can be used on any terminal, including vt100s and vt220s. It will process files created with a text editor, such as Emacs, which contain embedded formatting commands for a typesetting device, such as a laser printer. For more information see the man pages for *troff*, *tbl* (for typesetting tables) and *eqn* (for typesetting equations).

T_EX is widely available and, like *troff*, formats interspersed text and commands in a plain ASCII file to produce output for a laser printer or other typesetting device. T_EX was written by Donald Knuth particularly for producing sophisticated mathematical textbooks. The standard reference book is *The T_EXbook* by Donald Knuth (Addison-Wesley 1990, 483 pages). $LaTeX$ is a formatting system with a lot of predefined T_EX macros which, depending on your point of view, either make T_EX easier or get in your way. The standard reference book is *$LaTeX$: A Document Preparation System (2nd Edition)* by Leslie Lamport (Addison-Wesley 1994, 288 pages). The *xdvi* program can be used to preview DVI files produced by T_EX on an X terminal. See the manual pages for $LaTeX$, T_EX, *dvips*, and *xdvi*.

Newer packages include *FrameMaker* and *Interleaf*. Unlike *troff* and T_EX, both *FrameMaker* and *Interleaf* display the document on a bit-mapped screen, so both require an X terminal or workstation. To start *FrameMaker*, type **`maker`**; to start *Interleaf*, type **`ileaf`**. *FrameMaker* is more commonly used on campus, but *Interleaf* now offers an X Window System version of their full product to university academic departments for free. See your local help files for more information, if these products are installed on your system. All of the applications mentioned in this section are available on ACS.

Mathematica

Mathematica is a popular and sophisticated package from Wolfram Research for performing symbolic and numerical calculations, with graphical capabilities for presenting results. To run *Mathematica*, type `math`. Type `mathematica` to run the *Mathematica* notebook. See the help file on *Mathematica* and type `man math` for more information.

How to Get More Information

IS&T maintains a set of UNIX documentation and other manuals for public use at the IT Help Center @ Mugar. This documentation set includes a number of beginner's guides and contains much useful information to help you answer most of your questions. Most departments owning UNIX systems have a complete set of UNIX documentation available for your use. Departments also have documentation sets with tutorials and reference guides for other software purchased by the department.

Tutorials

During the first two months of each semester, IS&T offers about 100 free tutorial sessions in computing to Boston University students, faculty, and staff. These short (usually 2 to 3 hours), noncredit tutorials include introductory and intermediate classes on the UNIX operating system, graphics, personal computing, and many more topics. View the calendar and full text of the tutorial descriptions on the Web at www.bu.edu/computing/tutorials.

Books

There are many books which introduce UNIX in more detail than this document. *Learning the UNIX Operating System (5th Edition)* by Jerry Peek, Grace Todino-Gonguet, and John Strang (O'Reilly & Associates, Inc. 2001, 174 pages with index) is an introductory book for people new to UNIX. *UNIX in a Nutshell (4th Edition)* (O'Reilly & Associates, Inc. 2005, 904 pages) is a quick reference guide for commands and options, shell syntax (*sh* and *csh*), the *vi* and *ex* editors, *sed*, *awk*, *troff*, *tbl*, *eqn*, *adb*, *sdb*, *make*, *sccs*, and others. *The UNIX Programming Environment* by Brian W. Kernighan and Rob Pike (1984, Prentice–Hall, Inc., Englewood Cliffs, New Jersey 07632), is another commonly used book and is usually available at the BU Bookstore.

Help via E-mail

You can get help and answers to questions on most machines by sending e–mail to "help" on that machine. You can also talk to staff at the IT Help Center, at 533 Commonwealth Avenue, or the IT Help Center @ Mugar, at 771 Commonwealth Avenue. Addresses and telephone numbers appear at the end of this document.

Other Documents

The following documents are available free of charge from the IT Help Center.

- *ACS*
- *Using the Campus Network*
- *Using the X Window System at Boston University*
- *Using Pine*
- *Getting Started with Emacs*

Boston University's World Wide Web Connection

Boston University's website at www.bu.edu provides access to a wide variety of information. BU websites with information on computing include:

www.bu.edu/tech/help/
www.bu.edu/computing/dialup/
www.bu.edu/directory/

Addresses

Information Services & Technology (IS&T)
Main Office
111 Cummington Street, First Floor
617-353-2780
e-mail: it@bu.edu (Use this address only for general questions about computing at Boston University. For questions regarding a particular computer, send mail to *help* on the computer you are using. In particular, questions about the ACS cluster should be mailed to help@acs.bu.edu)

IT Help Center
533 Commonwealth Avenue
617-353-HELP (4357)
e-mail: ithelp@bu.edu

IT Help Center @ Mugar
771 Commonwealth Avenue
617-353-HELP (4357)
e-mail: ithelp@bu.edu

Scientific Computing and Visualization, Computer Graphics Laboratory
111 Cummington Street, Room 203
617-353-2780
e-mail: scv@bu.edu