

SCL Mini Manual 2
Unix for the SCL

September 28, 1999

Contents

1	The Basic UNIX Commands	3
1.1	Beginning work	4
1.1.1	Logging In	4
1.1.2	Changing Your Password	4
1.1.3	Logging Out	5
1.2	Manipulating Files	5
1.2.1	Change working directory	5
1.2.2	Present Working Directory name	5
1.2.3	Create a subdirectory	5
1.2.4	Remove directories	6
1.2.5	Remove files or directories	6
1.2.6	List contents of directory	6
1.2.7	Catenate, print, and merge	8
1.2.8	Copies file to new file (fname remains)	8
1.2.9	Display file on the screen	8
1.2.10	Move or rename files or directories	8
1.2.11	Print a hard copy of the file	8
1.2.12	Send/read an electronic mail	9
1.2.13	Transfer files to/from remote site	10
1.2.14	Login on remote terminal	10
1.2.15	Dial-In general information	11
2	Text edit - vi	13
2.1	Starting and Quitting the Editor	13
2.1.1	A skeleton vi session	13
2.1.2	Entering command-mode and ex-mode commands	13
2.1.3	Entering text	14
2.1.4	Moving the cursor	14
2.1.5	Joining two lines into one	14
2.1.6	Moving by larger chunks	15
2.1.7	Deleting text	15
2.1.8	Undo	15

2.1.9	Searching and replacing	15
2.1.10	Range specifiers	15
2.1.11	Copying and moving text	16
2.1.12	Writing text	16

Chapter 1

The Basic UNIX Commands

Before you start your first SunOS work session, make sure that your system hardware is set up and that you have an account on your system.(Your system administrator should have arranged this for you.)

The best way to learn about the UNIX system is to use it. In the following pages, we will present a series of tutorials designed to teach you to use the UNIX operating system. We will introduce several commands and explain basic UNIX system concepts. You are advised to read this tutorial and try the examples while sitting at the terminal.

Throughout the tutorials we have consistently used the symbols and procedures outlined below.

- The ”%” character indicates the UNIX system shell prompt. The shell program prints this character on an otherwise empty line to signal its readiness to accept your next command.
- A command line includes all the characters you type following the shell prompt (%) until you press carriage return to terminate this line.
- A command line consists of one or more distinct elements. Each element is a sequence of non-blank characters, separated from other elements by one or more spaces. The first element is always the name of the command (or a program). All additional elements are called as arguments or parameters. For example, in the command line `cp oldfile newfile`, `cp` is the command name `oldfile` is the first and `newfile` is the second argument.
- A special class of arguments is called options. They usually appear immediately after the command name and consist of a leading dash (-) which is usually followed by one or more letters and numbers. Each option (letter) specifies a modification to the normal command operation.

1.1 Beginning work

1.1.1 Logging In

`login` - log in to the machine

Your workstation screen will look something like this:

```
student console login:
```

This is called the *system login prompt*. Usually, the word before `login:` is the name of your machine. In SCL, one machine name is `student`, the other one is `fisher`.

Since the SunOS operating system will allow more than one person to use the system at the same time, it requires you to identify yourself. When you get your account on the system, remember your *username* and *password*. Your *username* is your last name. Your password restricts use of your account to those people who know his password. You'll learn how to change your password in later.

Example:

```
student console login: johnson
Password: *****
Last login: Thu Sep 14 09:19:26 on console
Sun Microsystems Inc. SunOS 5.4 Generic July 1994
```

1.1.2 Changing Your Password

`passwd` - change password.

Description:

When altering a password, the program prompts for the current password and then for the new one. The caller must supply both. The new password must be typed twice to forestall mistakes.

New passwords must be at least four characters long if they use a sufficiently rich alphabet and at least six characters long if monospace. These rules are relaxed if you are insistent enough. Only the owner of the name or the super-user may change a password; the owner must prove he knows the old password.

Example:

```
passwd<CR>
Old password: [Enter your old password and Press <CR>]
New password: [Enter your new password and Press <CR>]
Retype new password: [Enter your new password again and press <CR>]
```

Notes:

- None of the passwords will be displayed on the screen.
- In case you forget your password, you will have to contact the the system administrator (Dr.McKean) to delete your old password so that you can log in and reset it.

1.1.3 Logging Out

When you've finished your work session, you should secure your screen to protect the integrity of your files. Logging out, which ends a work session, is one way to do this.

`logout` - logout the system

1.2 Manipulating Files

1.2.1 Change working directory

`cd` [*directory*]

Description

Directory becomes the new working directory. The user must have execute (search) permission in directory.

Example

```
% cd temp <CR>
```

- will change the working directory to the `./temp` from `.`
- will display a failure message in case `temp` doesn't exist in the current working directory or is not a directory.

```
% cd <CR>
```

- will change the current directory to the user's login directory.

1.2.2 Present Working Directory name

`pwd` - Present Working Directory name

Description

`pwd` prints the pathname of the working (current) directory.

1.2.3 Create a subdirectory

`mkdir` *dirname* - Create a subdirectory

Description

`mkdir` creates specified directories in mode `777`. Standard entries, `.`, for the directory itself, and `..` for its parent, are made automatically. `mkdir` requires write permission in the parent directory.

Example

```
/home/student/zhao : mkdir temp
```

- will create a subdirectory with a name "temp" in the current directory.

```
/home/student/zhao : mkdir temp/in-temp
```

- will create a subdirectory in the subdirectory temp.

1.2.4 Remove directories

`rmdir dirname` - Remove directories

Example

```
/home/student/zhao : rmdir temp
```

- subdirectory temp will only be removed if it has no entry.

1.2.5 Remove files or directories

`rm filename` - remove file

`rm -r dirname` - remove directory

1.2.6 List contents of directory

`ls [-a] [-d] [-l]` - List contents of directory

Description

`ls` lists the contents of the directory when filename is a directory, and lists the name of the file when filename is a file. Output is sorted alphabetically by default. When no argument is given, the current directory is listed. When several arguments are given, the arguments are sorted appropriately, and file arguments are processed before directories and their contents.

Options

- a List all entries; in the absence of this option, entries whose names begin with a period (.) are not listed (except for the superuser, for whom `ls` normally prints even files that begin with.).
- d If argument is a directory, list only its name; often used with `-l` to get the status of a directory.
- l List in long format, giving mode, number of links, owner, size in bytes, and time of last modification for each file.

The mode printed under the `-l` option contains 11 characters, which are interpreted as follows: the first character is as follows:

- d if the entry is a directory
- if the entry is a plain file

The next 9 characters are interpreted as three sets of three bits each. The first set refers to owner permissions; the next refers to permissions to others in the same user-group; and

the last to all others. Within each set, the three characters indicate permission to read, write, or execute the file as a program. For a directory, execute permission means permission to search the directory. The permissions are indicated as follows:

- r if the file is readable;
- w if the file is writable;
- x if the file is executable;
- if the indicated permission is not granted.

Examples:

```
% ls
```

- will print an alphabetized list of file names in the current directory.

```
% ls *.c
```

- will list all of the files in the current directory whose names end in ".c".

```
% ls -l
```

- will print an alphabetized list in long format in the current directory.

```
% ls -la
```

- will print an alphabetized list in long format including "." and ".." and all other files with names starting with ".".

```
% ls -l temp
```

- will print long format information for the file having name "temp" in the current directory. output could be

```
-rw-rw-rw- 1 zhao nobody 1740 Sep 6 13:18 temp
----- |--- file-name
| | | | | |----- Date/Time
| | | | |----- Size
| | |----- group-id
| |----- owner-id
|----- number of links
----- file permissions
```

```
% ls subdir1
```

- list the contents of the subdirectory ./subdir1

1.2.7 Catenate, print, and merge

`cat filename` – Catenate and print

`cat file1 file2 ... > file` – Merging files *file1 file2 ...* to a new file *file*.

Description

Cat reads each file in sequence and displays it on the standard output or merges them into a new file.

1.2.8 Copies file to new file (fname remains)

`cp fname newname` - Copies file to new file

1.2.9 Display file on the screen

`more filename` - Display file on the screen page by page

Description

`more` is a filter which allows examination of a continuous text one screenful at a time on a soft-copy terminal. It normally pauses after each screenful, printing `-More-` at the bottom of the screen. If the user then types a carriage return, one more line is displayed. If the user hits a space, another screenful is displayed.

1.2.10 Move or rename files or directories

`mv oldname newname` - will rename the file (could be a directory) "oldfile" to "newfile"

1.2.11 Print a hard copy of the file

a) `lp [-Pprinter] [-#num] [name ...]`

Description

`lp` can be used to print the named files on any printer connected to the SCL machines.

Options:

`-P` force output to a specific printer. The default printer is SP. The other choice is NeWSprinterCL(Color Printer.)

`-#n` prints *n* copies of the file.

b) `a2ps [-p] [-nn] [-ns] [-nh] [-fn] [filename]|lpr`

Description

`a2ps` is for printing output of SAS, MINITAB, or others statistical output. The command `txt2ps` is same as `a2ps`.

Options:

- p force output to portrait style, the default style is landscape.
- nn does not print the number of line.
- ns does not print the frame.
- nh does not print the title.
- fn the font size is *n*.

Examples:

To print a postscript file use the following command:

```
lp -dsolo file .ps
```

To print a dvi file use the following command:

```
dvips -f file .dvi lp -dsolo—
```

To print a textfile use the following command:

```
txt2ps filename lp -dsolo—
```

1.2.12 Send/read an electronic mail

`mail recipient address`

Description

`mail` command can be used to electronically send a mail to another user (on the same or remote machine). In case no recipient is specified in the command line, `mail` reads all of the pending mails received by the system for the user.

All the recipients are specified through their e-mail address which is a combination of their login-id and their machine name. For example, all users having UNIX account in SCL have e-mail addresses of the form,

```
lastname@stat.wmich.edu
```

After you finish, at the new line enter “.”, or at the new line typing `^d` - control + d, then your mail will be sent to recipient.

Example

To send an e-mail to user manish who has an account in the SCL, you can use following command,

```
~> mail manish<CR>
```

To send an electronic mail to user having login id jignesh on the machine `string.cs.wisc.edu` (machine at University of Wisconsin, Madison) you should use the command,

```
~> mail jignesh@string.cs.wisc.edu<CR>
```

If you want to send file to another user, you should use the command,

```
~> mail jignesh@string.cs.wisc.edu < fname<CR>
```

Notes: e-mail addresses are globally unique.

1.2.13 Transfer files to/from remote site

`ftp hostname` - connect to *hostname*

Description

The `ftp` command is the user interface to the Internet standard File Transfer Protocol(FTP). `ftp` transfers files to and from a remote network site.

Example

`ftp math-stat.wmich.edu` - connect to math-stat system. You will be asked the username of remote system(math-stat system) and password. The screen will display the following message:

```
Connected to math-stat.wmich.edu.
220 ibis FTP server (SunOS 5.5) ready.
Name (math-stat:zhao):zhao<CR>
Password:*****<CR>
230 User zhao logged in.
ftp>
```

At prompt `ftp` , you input your command,

`ftp> get remote filename local filename` - get file from remote system to your account.

`ftp> put local filename remote filename` - put your file to remote system.

`ftp> mget r*.*` - multiple file transfers, transfers files with names beginning with a letter r from remote system to your account.

`ftp> mput r*.*` - multiple file transfers, transfers files with names beginning with a letter r from your account to remote system.

When you use `mget` or `mput`, after finishes each file transformation, the prompt will ask you whether or not you want to transfer the next file, if you want, just enter `RETURN` key, otherwise, enter `n`.

1.2.14 Login on remote terminal

`rlogin remote system address`

Description

`rlogin` establishes a remote login session from your terminal to the remote machine named *hostname*.

Example

`rlogin grog.lab.wmich.edu -l username` - remote login to grog (UCC UNIX account). In same time specify a different username for the remote login. If you do not use this option `-l`, the remote username used is the same as your local username. The system will ask you the remote account password, then your current window will become the terminal of remote system.

1.2.15 Dial-In general information

What you need to be able to use VAX and/or UNIX by phone:

1. a **valid VAX account** on the UCC VAX system
2. a **personal computer** in your home. The rest of this information assumes you are working with a microcomputer.
3. a **modem**. This is a device which manages the telephone signals between the computers. There are modems which are in separate boxes, and modems which are cards to install inside the microcomputer. Either kind will work.
4. a **telephone line**. You need to have a private line.
5. a **communications program** which runs on the microcomputer, such as `terminal.exe` (you will find this program in Accessories Icon in MS-Window).

The telephone number you can reach to WMU is **387-2040, 387-2070 or 387-2080**. Once connected the `host:` prompt will appear. You can type host address which you want to login, such as `stat.wmich.edu` or `tigger.cc.wmich.edu` and then you will be asked login username (UCC VAX account username) and password. If your VAX account is valid, then you enter Michnet system and begin to login your account. The follow message is to login SCL through dial-in.

```
host: watson.stat.wmich.edu
Access Controlled
login: username@wmich.edu
Password:*****
```

```
UNIX(r) System V Release 4.0 (student)
```

```
login: zhao
Password: *****
Last login: Tue Oct 31 20:15:55 on console
Sun Microsystems Inc.   SunOS 5.4           Generic July 1994
/home/student/zhao :
```


Chapter 2

Text edit - vi

The vi editor is a screen-oriented editor. You can use it for preparing of the programs and documentation. What is described here is a quick introduction to vi.

2.1 Starting and Quitting the Editor

To edit a file, enter the command `vi` at the shell prompt:

```
vi filename
```

Here, `filename` is the specification for the file to be edited; it can refer to either an existing file or one which you wish to be created. If the file exists, the file will be read into the editor and the beginning of the file will be displayed; otherwise, a screen with a column of tilde characters will be displayed.

2.1.1 A skeleton vi session

```
vi filename  start vi with cursor at first line
i           switch to insert mode[this i will not show]

           [type in your text at this point]

Q           move cursor to the command line (bottom line)
:wq         leave vi, save the file
or
:q!         leave vi, do not save the changes
```

2.1.2 Entering command-mode and ex-mode commands

The `<ESC>` key switches you from insert mode to command mode. Commands which affect the entire file or parts of the file specified by line numbers or marks, such as write or search

commands, require that you type a colon(from command mode) to go to the command line with the `:` prompt before you enter the command; this is ex mode. You can see the command and change it until you press `<RETURN>`. Commands which affect the characters in the file at the cursor position let you enter the command itself with no colon, and the command does not echo on the screen.The command action happens immediately - no `<RETURN>` is necessary. The cursor-moving commands are letters typed in command mode.

2.1.3 Entering text

The following commands will all take you from command mode to insert mode.

<code>i</code>	insert text before the cursor
<code>I</code>	insert text at the beginning of the line
<code>a</code>	append insert text following the cursor
<code>A</code>	append; insert text at the end of the line
<code>o</code>	open a new line below the current line
<code>O</code>	open a new line above the current line
<code>r</code>	replace the current character with a new character
<code>R</code>	replace characters with new ones until <code><ESC></code> is pressed
<code>cw</code>	replace word

2.1.4 Moving the cursor

If you terminal has arrow keys, moving the cursor is very easy, otherwise, you will use characters keys.

<code>j</code> or <code>↓</code>	move down one character
<code>k</code> or <code>↑</code>	move up one character
<code>l</code> or <code>→</code>	move right one character
<code>h</code> or <code>←</code>	move left one character

2.1.5 Joining two lines into one

<code>J</code>	join two lines; deleting the carriage return will not join lines
----------------	--

2.1.6 Moving by larger chunks

CTRL-D	scroll down(forward) half a scrn
CTRL-U	scroll up(backward) half a scrn
CTRL-F	scroll forward almost a whole screen
CTRL-B	scroll backward almost a whole screen
H	home position(top left corner)
M	middle line
L	lower line
<i>n</i> G	go to line <i>n</i>
G	go to the end of text

2.1.7 Deleting text

x	delete current character
<i>nx</i>	delete <i>n</i> characters, beginning at current one
dd	delete the current line
<i>n</i> dd	delete <i>n</i> lines
dw	delete one word

2.1.8 Undo

u	undo the last change to this line
U	Undo all changes to this line

2.1.9 Searching and replacing

<i>/findme</i>	go to the next instance of <i>findme</i>
? <i>findme</i>	go to the previous instance of <i>findme</i>
N	repeat the previous search, but in the opposite direction
<i>n</i>	repeat the previous search, in the same direction(find next)
<i>/^function</i>	find the next line starting with <i>function</i>
<i>:range s/old/new/g</i>	find and replace all instances of the <i>old</i> string in <i>range</i> with the <i>new</i> string. Omitting the <i>/g</i> will cause it to replace only the first instance on each line. See Range specifiers section.

2.1.10 Range specifiers

These are used in copy, move, write, and substitute commands.

n line n
 n_1, n_2 lines n_1 through n_2
 $\%$ the whole buffer (the whole file you are editing)
 $.$ the current line
 $., \$$ the current line through the end of the buffer
 $., +n$ the current line through the n line following the current line

2.1.11 Copying and moving text

yy yank the current line
 P put the contents of the general purpose buffer before the current position.
 p put the contents of the general purpose buffer after the current position.
 $"bdd$ delete this line, putting it in named buffer b
 $"bp$ put a copy of the material in buffer b at the current position

2.1.12 Writing text

$:w \textit{fname}$ write the entire file to \textit{fname}
 $:n, \$w \textit{fname}$ write lines n to the end to \textit{fname}