Quick Guide to Unix commands #2--Draft

(or .2% of unix)

You are the 005289 visitor to this page.

Introduction

This Guide is for those that have learned the VERY VERY BASICS of unix and want to know how to do a few more interesting functions. These commands are quite useful and increase the amount you can do with unix significantly. The commands are divided between information gathering/searching, file modification, communication, and a few other helpful "clean-up" commands. This guide was written as a compliment to the "Quick Guide to Unix Commands #1." If you haven't mastered the commands in the first Guide, it is not recommended that you start on this guide.

Remember that all of these commands are case sensitive.

IF THEY ARE IN LOWER CASE LETTERS, THAT MEANS THEY MUST BE IN LOWER CASE LETTERS.

Information Gathering and Search Commands

finger

This command is used to find out information about others on the net. It can be used to find out if someone is logged on, how long they have been on, and if they have any unread mail. It may also give you some more information about the person depending on whether they have a .plan file(To create this file, use an editor to make a file named .plan. It can contain anything. The permissions on this file must be 644--these permissions are detailed later in this guide in the chmod section). The finger command looks people up by their email address. The format of the command is also dependent on "where" the person you are looking for is located. For example, if you have your account on ux1, and the person you are looking for is also on ux1 (kundert for example), then you would type:

ux1> finger kundert

The results of this command would look like this:

```
Login name: kundert                    In real life: Joshua Kevin Kundert
Directory: /students/homeg/kundert     Shell: /bin/csh
```

```
On since Jul 18 11:06:37 on ttytw from rcstaff25
12 seconds Idle Time
No Plan.
```

However, if you are on ux1, and you wanted to finger someone (kundert@students for example) on the students cluster, then you would type:

ux1> finger kundert@students

and the output would be:

```
[students.uiuc.edu]

Login name: kundert                      In real life: Joshua Kevin Kundert
Directory: /students/homeg/k/kundert     Shell: /usr/local/bin/tcsh
Last seen at Thu Nov  2 14:37:36 1995 on ux4.

No plan.
```

If they are on a totally different machine (for example at a different university or at a business), then you would have to type in the full address after finger. If you wanted to finger jasha@magic.nowhere, it would look like this:

ux1> finger jasha@magic.nowhere

It is important to note that the finger function will not always work, because some machines have it turned off.

grep

This command stands for "get regular expression print." It is a search command that will allow you to search files for particular expressions which it will then print to the screen. The normal format for grep is:

```
ux1> grep (options) expression filename (filename2 filename3 ....)
```

It should also be noted that it could be more than one filename, because you can search more than one file at a time. As an example of how to use grep, assume that you have a file named "text", which has the following contents:

```
>Java, a new object-oriented programming language developed by Sun
>Microsystems, Inc., can be used to develop Java "applets,"
>mini-applications that are embedded into the HyperText Markup Language
>(HTML) code currently used to create World Wide Web pages. Applets add a
>revolutionary new level of interactivity to the Web that goes far beyond
>the simple forms and buttons currently available on most Web sites.
>Features of applets written in Java range from simple text manipulation
>and 3-D animated graphics to systems or network management tasks--and best
>of all, users can instantaneously access and run these applications just by
>clicking on them from their Web browsers.
```

If you wanted to find all the lines that the word "Java" is in, you would type:

ux1> grep Java text

and these lines would appear on the screen.

```
>Java, a new object-oriented programming language developed by Sun
>Microsystems, Inc., can be used to develop Java "applets,"
>Features of applets written in Java range from simple text manipulation
```

There are also a list of options that you can put in the command to get various other information. Some of these options are:

1. -v All lines but those matching are printed.
2. -x (Exact) only lines matched in their entirety are printed (fgrep only).
3. -c Only a count of matching lines is printed.
4. -n Each line is preceded by its relative line number in the file.
5. -i The case of letters is ignored in making comparisons - that is, upper and lower case are considered identical. This applies to grep and fgrep only.

To see some of the results of these options. If you wanted to know the line numbers of the lines with the word "Java" in them, you would type:

ux1> grep -n Java text

and the result would be:

```
3:>Java, a new object-oriented programming language developed by Sun
4:>Microsystems, Inc., can be used to develop Java "applets,"
9:>Features of applets written in Java range from simple text manipulation
```

One specific option that should be noticed is the "-i" option. If you don't use it, then grep is case sensitive. For example, if you wanted to find all uses of the word "applets" in the file text, and you typed:

ux1> grep applets text

then your output would be

```
>Microsystems, Inc., can be used to develop Java "applets,"
>Features of applets written in Java range from simple text manipulation
```

However, If you typed:

ux1> grep -i applets text

then your output would be:

```
>Microsystems, Inc., can be used to develop Java "applets,"
>(HTML) code currently used to create World Wide Web pages. Applets add a
>Features of applets written in Java range from simple text manipulation
```

Those are the basics of grep. The best way to learn about it is to experiment with it.

ph

The ph command is another way to obtain information about students and staff here at the University of Illinois and certain other places around the net. ph can be used through your unix account to find out information about students at other universities (assuming they have ph servers there), but it is much easier to use gopher to find out this information. To find out information with ph, you would type:

ux1> ph firstname lastname

You should then get some basic information about the person, such as address, curriculum, and their email address. The phone number will not always be there. The person must add this field in ph for the phone number to appear. To find out more information about a person, then you can type:

ux1> ph firstname lastname return all

If you are not sure of the spelling of the person's name, you can use wildcard characters. For example, if you wanted to ph a person named James Kersting, but you did not remember the person's full last name, you would type:

ux1> ph james ke*

and it might find the person. If it can't find it, it will give you the notice:

502:Too many entries to print.

This means that the search isn't specific enough and you need to add more information. Other bits of information can also be used to find people. Using a person's curriculum is a common way of finding people. If you knew james was a junior in engineering, you could type:

ux1> ph james k* curriculum=eng3

and it would give you a listing of all the james's who are juniors in engineering (if it wasn't to large to print). These functions can also be combined.

File Permission Modification

chmod

This command is essential if you ever want to have your own homepage. It stands for 'change mode.' It is the command that allows you to change the permissions (also called file access modes) on your files and directories. What **THAT** means takes a little explaining. In unix, all files and directories have permissions on them. There three sets of permissions on each file that correspond to a owner, a group, and other. If you have ever typed ls -l at your unix prompt, you may have have seen a line like this:

```
drwxr-xr-x  2 kundert       512 Jul 14 18:13 public_html/
```

The "drwxr-xr-x" lines are the permissions--file access modes-- of that particular item in your account. The differences between owner, group, and other break down as follows:

```
         drwx     r-x       r-x
         owner    group     others
```

These permissions are usually set so that only the owner (the creator of them) can read, write, or execute them. Permissions can be set, however, so that certain groups of others, or everyone else in the world can read, write, or execute your files and/or directories. IT IS NOT ADVISED THAT YOU SET YOUR PERMISSIONS SO THAT OTHERS CAN WRITE TO YOUR DIRECTORY. To see the permissions of your files, you type:

ux1> ls -l

This will give you a listing like this:

```
drwx------  2 kundert        512 May 25 09:18 Mail/
drwx------  4 kundert        512 Jul 12 11:59 News/
drwx------  2 kundert        512 Jun 15 11:18 cyber/
-rw-------  1 kundert        705 Jul 18 09:19 end
drwx------  2 kundert       1024 Jun 27 12:01 features/
drwxr-xr-x  2 kundert        512 Jul 14 18:13 public_html/
-rwxr-xr-x  1 kundert       1565 Aug 14  1993 sigswitch*
```

The "drwxr-xr-x" portion of the sixth line tells you the file access modes of your the public_html directory. The meaning of the specific line, "drwxr-xr-x" is as follows:

1. d--------- means that it is a directory
2. -rwx------ means that it is read,write, and executable by you(owner).
3. ----r-x--- means that it is read and executable by your group.
4. -------r-x means that it is read and executable by everyone(others).

(This setting "drwxr-xr-x" is somewhat common, because it is the setting needed for directories used in setting up personal homepages)

Knowing this, chmod is the command used to change these values. If you type:

ux1> man chmod

It will give you these numbers as a means of explaining chmod:

```
400      read by owner
200      write by owner
100      execute (search in directory) by owner
070      read, write, execute (search) by group
007      read, write, execute (search) by others
```

This result may appear rather cryptic. To use our previous example with these numbers, it would appear as follows:

1. d--------- Has no chmod number associated with it, being either a directory or a file is determined during its creation.

2. -rwx------ =700, because it is read(400)+write(200)+execute(100)by owner.
3. ----r-x--- =050, because it is read(040)+execute(010) by group.
4. -------r-x =005, because it is read(004)+execute(001) by others.

Therefore, if you had a directory that looked like this:

drwx------ 2 kundert 512 Jun 15 11:18 cyber/

and you also wanted to make it readable and executable by your group and everyone else, you would type the command:

ux1> chmod 755 cyber

and it would change from

drwx------ 2 kundert 512 Jun 15 11:18 cyber/

to:

drwxr-xr-x 2 kundert 512 Jun 15 11:18 cyber/

If you had typed the command:

ux1>chmod 744 cyber

it would have changed to:

drwxr--r-- 2 kundert 512 Jun 15 11:18 cyber/

This is all there is to it. There is also another method using letters, but it is more complicated and will not be covered here.

Communication

talk

The talk function may allow you to "talk" to your friends or associates who have internet connections and who are logged on at the same time you are. If you know someone is logged on at the other address (zack@magic.nowhere for example), then to talk to them you would type:

ux1> talk zack@magic.nowhere

They would then get a message flashing on the screen saying you (your email address) wanted to talk to them and it would tell them what to type. Once the two of you are connected, you are given a split screen, in which you get one half to type in and your friend gets the other. To get out of a talk session, you type: **^c** (Which means you hold down the control key and then hit the C key.) The one downside to this function is that not all places have it.

Another important concern about talk is that it should only be used to talk to people you know. If you think you **might** know the person then extreme courtesy and politeness should be used while talking the person. Otherwise the use of talk to bother people you don't know is considered harrassment and can lead to the revoking of your account.

Some Clean-Up Commands

ps

This command is often used in conjunction with the kill command. The ps command will give you a listing of all the recent processes that your account has run and which pid (process ID) number each process has. Here is an example of the ps command:

```
ux1>ps
 PID TT STAT  TIME COMMAND
 7268 qm I     0:00 elm
 7312 qm I     0:00 sleep 60
15497 qm I     1:00 sigswitch -f sigswitch
 7530 qM S     0:02 -tcsh[kundert] (tcsh)
 7625 qM R     0:00 ps

 ux1>
```

On the students cluster here, you should use the command ps -fu *username*--where username is your login to the students cluster. Here is an example of the command on students:

```
ux4> ps -fu kundert
     UID   PID  PPID  C    STIME TTY      TIME COMD
 kundert  4978  4397 80 14:24:31 pts/406  0:03 ps -fu kundert
 kundert  4397  4362 80 14:23:46 pts/406  0:03 -tcsh

ux4>
```

The pid number is a way of referencing every process that it is started on a machine. It is a basic accounting feature. Once you know the pid you can use the kill command to stop the process.

kill

This command is used in conjunction with the ps command. With the kill command you can kill a process(such as elm or nn) which is running on your account. An example of when you might have to kill a process is when you are running elm and are kicked out of the internet for some reason. If you login again and try to run elm, it says that another copy of elm is running and that you can't start it up. Using the ps command you can get the pid number of the "old" elm and then use kill to stop the "old" elm from running anymore. For example if were in this situation and you typed:

ux1> ps

and got the result:

```
PID TT STAT  TIME COMMAND
 7268 qm I     0:00 elm
 7530 qM S     0:02 -tcsh[kundert] (tcsh)
 7625 qM R     0:00 ps

ux1>
```

You would know that the pid number of the "old" elm was 7268. To get rid of this process, you would type:

ux1> kill 7268

To be really sure to kill it (because sometimes unix doesn't think that you really do want to kill it), you would type:

ux1> kill -9 7268

The -9 will tell it to kill it and kill it hard.

Further "Help"

For some further help, there are always the man pages. It must be mentioned, however, that most of the man pages were written in the "expert- friendly" mode and may be confusing to new users. To use the man pages on a command (chmod for example), you would type:

ux1> man chmod

and it would give you what was in the online manuel. You can always go to these pages for more information.

---

last updated 11/02/95