# Linux and vi

Lab 2 has three parts to it. The first part is to get the student familiar with the more of the basics of using a Linux operating system. The second part will be an exercise using the *vi* editor. The third part is less structured will require some look-ups for commands, etc.

Items noted with a \*\* are the key concepts for this lab. Pay special attention to what is done at these steps and why.

## Part 1: More Linux basics

### Password

Log onto the system using your system userid and password. Your userid should be the same as your 49er id. The initial password will be **have been sent to you in an email from TSO.**

Change your password if prompted. If not prompted use the *passwd* command to change it. Just follow the prompts.

### Linux II

#### Piping

Sometimes you would like the output of one program to be the input data to another program. That is piping, signified by the verical bar (|).

**Example:** you what to know what services use the UDP protocol. The text file *services* has that information. Cat, more and less can be used to look at the contents of text files. You could do more services to view the contents and scroll down to manually scan all lines than have “udp” in them. Or you could try less services, and scroll up and down looking for all “udp” lines or search for the string with /upd.

An easier way is to have the utility *grep* look for the data (pattern) for you:

Try the following: \*\*

First switch to the etc directory and then list the contents of *services* with *cat*:

cd /etc

cat services

and observe the output.

First let’s have *grep* look for lines with “asp” in them:

cat services | grep asp

The above will take the output of *cat* and send (pipe) it to *grep* to be processed. *Grep* then looks for lines with “asp” in them and lists those lines to the terminal.

Next see if there is a service with “80” in the name:

cat services | grep 80

Now let’s look for lines with the udp potocol:

cat services | grep udp

Note there are a lot of lines with “udp” in them. If there are lots of lines from grep you can then pipe the output of *grep* to another program such as *less* or *more*. To do that try:

cat services | grep udp | less

Summarize what you saw from the above command.\*\*

#### Redirection

By default, the output of a program goes to standard out (std out) which is usually the terminal or monitor. The output of a program can also be sent to a file. This file can then be read or analysed later. There are two ways to send data to another file: > and >> as documented in the following table:

Table 1: Use of the > and >> directives

|  |  |
| --- | --- |
| program > file\_name | Replace the contents of *file\_name* with the output of *program* |
| program >> file\_name | Append the output of *program* to the end of the file *file­\_name* |

To write to or create a file in a directory you must have permissions to do so. The above commands will work because everyone has read access to */etc*. But only root can write to files in */etc*. Your userid can write to your home directory (~) however.

Go to your home directory:

cd ~

Check you’re at your home directory with the following print working directory command:

pwd

You should see an answer similar to */home/yourid*

Now to view *services* file you need to type the complete directory path to get to it, but now you can now save the results directly to a file in your home directory.

Now we can save the output of the*grep* search for udp lines to a file:

cat /etc/services | grep udp > udpentries.txt

If the file *udpentries.txt* exists in your home directory its contents will be replaced with the output from *grep*.

Here is an example of saving two different searches to one file:

cat /etc/services | grep asp > selectentries.txt

cat /etc/services | grep 80 >> selectentries.txt

The first line replaces *selectentries.txt* with the “asp” matches. The second line adds the matches for “80” to the end of the file. Try the above and document the results in your report.\*\*

Lastly, be aware you can also redirect the contents of a file as the input for another program with the < symbol. We won’t go into the details of that now, but feel free to experiment if you are curious.

## Part 2: vi

### Exercise summary

Use vi to create and edit files

### Start vi

**Note:** *vi* is the original standard editor for UNIX systems. Many systems have an updated or improved versions such as *vim* or *vim-tiny*. You will need to find which is installed on your workstation (and on your VMs in the future). The labs will be using the term *vi* generically. Substitute the name vi for the version installed on your machine.

* Open a terminal
* Start *vi* by typing ***vi*** and hitting <enter>
  + you are now in *vi* in command mode
* Add some text:
  + Type *a* then hit the <enter> key
    - you have switched from the command mode to an insert mode
  + Enter your name, home town (but misspell it), and hair color – each on a separate line
* Try to quit
  + Hit <esc>, then type *:q* and comment on the results \*\*
* Save the file with a specific name:
  + Hit <esc>, then type *:w your\_filename* and note the results this time
  + Type *:q* to quit and note the results this time
  + Note: the two commands can be combined as *:wq* if you want to save the file to the same name and quit.
* List the files and note the new file in your *pwd* (present working directory)
* Reopen the file: *vi your\_filename*
* Fix the misspelling of youre home town
* After your home town on a new line add your home state
  + Try the *i, a, r* commands when inserting, appending, and replacing text. Note the difference between the upper and lower case commands for each.
* Change your hair color (in the text, not in real life)
* Save your file and exit
  + Note what directory the file was saved in.\*\*

In your report show the end result, e.g. the contents of the edited file. \*\*

### Edit a sample network configuration file

Get a copy of the of the network configuration file *interfaces* to your home directory. On the Debian lab workstations the *interfaces* file is in the */etc/network* directory. Copy the *interfaces* file to your home directory (~ is the notation for everyone’s home directory.):  
 cp /etc/network/interfaces ~

Side note: the above command works regardless of your pwd since you have explicitly noted the source and destination directories.

# This file describes the network interfaces available on your system

# and how to activate them. For more information, see interfaces(5).

# The loopback network interface

auto lo

iface lo inet loopback

# The primary network interface

allow-hotplug eth0

iface eth0 inetdhcp

auto eth0

#

By default the *interfaces* file should look similar to but not identical to this:

Note that # is the comment character. When changing a configuration file is it a good idea to not delete the old data, but to comment them out so the file can be easily restored to the original condition if needed. In this example we are changing the configuration file for the Ethernet card to use a static IP address instead of getting an addressed assigned by DHCP. Use vi to edit the file to the example on the next page.

The underscored lines are the ones you will have either edited or added. Note that the line with *dhcp* was commented out instead of deleted. That makes it easy to go back if an error was made or you want to use dhcp again. Make sure to replace *your\_name* with your name and put the current date in for *date*.

In your report show the end result.\*\*

# This file describes the network interfaces available on your system

# and how to activate them. For more information, see interfaces(5).

#

# Edited by your\_name on date

# The loopback network interface

auto lo

iface lo inet loopback

# The primary network interface

allow-hotplug eth0

# iface eth0 inetdhcp

iface eth0 inet static  
address 192.168.1.5  
netmask 255.255.255.0  
gateway 192.168.1.1

auto eth0

#

## Part 3: On Your Own

Use man to get details for the following commands: *mv*, *cp* and *vm*. What error did you get for one of them?\*\*

Try the following commands:

Note: for the following exercise the files you move and copy is your choice. The files you edited practicing vi are safe choices.

* mkdir
  + Create a new directory in your home directory (~). The name is up to you but remember it for later use.
* mv
  + Rename a file
  + Move a file to the directory you created above
* cp
  + Copy multiple files to the directory you created above. The files you copy are up to you.
* jobs, ps, and kill  
  In the following *xload* and *gcalctool* are utilities that are used as examples of programs that need to be terminated. If the & is entered as a parameter after the program to be run the program is started as a background process and control is returned to the terminal.
  + Start *xload* as a background job (xload &)
    - the ampersand starts xload as a background process
  + use jobs to list the current jobs \*\*
  + kill xload: kill %n
    - substitute n with the job number
  + start *xload* as a separate background job again
  + use ps to list the processes \*\*
  + kill xload using the process number e.g. kill 1234
  + Start both xload and the calculator (*gcalctool*) with the & option
    - kill one then the other by using the jobs number
    - repeat starting both programs but this time kill them in reverse order
  + Last try starting *xload* without the &
    - Note the result in the terminal \*\*
  + Note: if a job or process won’t kill use the “extreme” version: kill -9 name
    - the -9 switch kills anything, so use with care

Read the man file for *ps* to find how to list all the processes and try that. Summarize what you saw.\*\*

* *du* and *df*
  + Try the commands and see what they report
  + Compare the default to using the –h option for both commands
  + Which one of these is best viewed by piping the output to the *less* program and why? \*\*
* *chmod*
  + Use man to get details for chmod
  + Experiment with the *interfaces* file you edited in Part 2
    - Do a long listing for *interface*s: ls –l interfaces
    - Comment on the permissions, owner and group for the file\*\*
    - Change the permissions so the owner can no longer edit the file: chmod u-w interfaces
    - Do a long listing again to verify the change worked
    - Use vi to edit the file. Make a simple change to anything in the file
    - Try to save and quit the file with :wq
    - Comment on the result\*\*

# Appendix

### Gathering Data

Whenever possible, data must be gathered and reported as text, not as screenshots. This includes no pictures taken with your phone. There are two basics methods for capturing text in the Linux terminal:

* Cutting from the screen and pasting into a text file
  + Note: the short cuts are similar to but subtly different from Windows
* Redirecting output to a file with the “>” directive, e.g. ls –la > dirlisting.text
  + Note: > replaces a file, >> appends to a file

### WH302 Lab structure oddity

Regardless of the ID you are using you will notice something unusual about the current users Home directory and the Networks storage directory.

Some general notes:

* In a network file system the users Home directory would follow the user from system to system
* The workstations in this lab will need to be occasionally disconnected from the network
* Access to the home directory is needed at alltimes to prevent the machine from freezing
* Students need a network file to store data that can be accessed from any workstation

The 302 lab has an “odd” configuration:

* When a user logs on to a alb workstation new home directory is created for that ID on the local machine if one does not currently exist
* The user is connected to their network storage space (network\_storage on the network file server)
* The users “home” directory, which is normally placed in the network file system is redirect to the local storage version.
* All data saved to the home directory (and Desktop in the home directory) are saved on the local workstation
* Only specialnetwork\_storage directory is available from any workstation connected to and logged onto the hades lab.
* All date in the various users home directories is erased overnight on all workstations
  + The user’s private network\_storage directory is not effected