**Linux Lab - Redirection and wildcards**

**Introduction :** Topics to be covered in this lab include:

* **Output Redirection :** Output redirection means that instead of displaying the output of a command to the display window, it will get captured and redirected (usually to a file).
* **Appending Output:** adds (concatenates) to the end of a file. This can be used when you don’t want to overwrite the contents of the existing file.
* **Sort command** is used to sort in alphabetically or numerically order a list of items.
* **Redirecting Input and Output using the sort command.** As well as output being redirected, input can be redirected also. Instead of requiring the user to supply the input at keyboard, the input could come from a file.
* **Cat command** – concatenates files and prints to the standard output.
* **who command –** lists who is logged on.
* **grep command** - You can use **grep** to search for a word or pattern in a file.
* **Wildcards** (? and \*) : used to search for groups of files.
* **Pipes :** Pipe is a symbol used to allow the output of one command to be the input to another command.

**You should use the linux303 machine for this labsheet.**

**Connect to linux303 by using the Secure Shell Client (SSH).**

**Task 1 – Output Redirection**

When you run a command where does the output usually get displayed? ..................................

Output redirection means that instead of displaying the output of a command to the display window, it will get captured and redirected (usually to a file).

 ls > Files.txt

The output of the listing command will be redirected and stored in file Files.txt.

* Key in the command

ls > fred.txt

The > symbol redirects the **output** of the ls command and sends it to a file called fred.txt.

**NB:** It **overwrites** what is in the file if the file already exists. It creates the file if it doesn’t exist.

You can take the output of any command that sends its output to the terminal window and redirect it.

* Write down the command to do a **long** directorylisting and send the output to a file called **listing.txt (**overwriting the file)

………………………………………………………………………………………………

**Task 2 – Appending output**

How do you **append** output to a file, rather than overwriting what is already there?

………………………………………………………………………………………………

What does the term **append** mean?

………………………………………………………………………………………………

Write down the command to

* do a long directory listing of the **/bin** directory and **append** the output to the file listing.txt

………………………………………………………………………………………………

* display the current working directory and **append** this to the file listing.txt

………………………………………………………………………………………………

Run the **cal** command – what does it display?

………………………………………………………………………………………………

Redirect this to a file called **diary.txt**

………………………………………………………………………………………………

**Task 3 – Redirecting both input and output (using the sort command)**

**Sort**

This command will sort text either alphabetically or numerically. Sort on its own will allow you to enter a number of lines of text (ended by Ctrl+D) and will then display the lines in sorted order. E.g.

 *sort*

 *beetroot*

 *carrot*

 *artichoke*

 *Ctrl+D*

will output

 *artichoke*

 *beetroot*

 *carrot*

# Use of Redirection symbols ( Input and Output)

As well as output being redirected, input can be redirected also. It’s not as common as output redirection as there are not many commands that need input supplied by user.

|  |  |
| --- | --- |
| **Redirection symbol** | **Purpose** |
| **<** | Redirects input (e.g. to a file). |
| **>** | Redirects output (e.g. to a file). |

# Create a file with 6 names and give it the name sample.txt.

Using redirection you can sort files. The < symbol allows you to redirect input to the sort command from a file. The result will be displayed on the screen. E.g.

 sort < sample.txt

You can also use redirection to send the output from a file to another file. This can be done using the > symbol. The sorted version of the first file will be saved in the second file. E.g.

sort < sample.txt > sortedlist.txt

**Exercises:**

1. Use the sort command on its own. Type a list of the subjects you do (in non-alphabetical order) and finish with Ctrl+D.

………………………………………………………………………………………………

1. Use **cat command** to create a file called names.txt that contains the names of 4 people in your class. Then append another 2 names onto that file.

………………………………………………………………………………………………

1. Use the sort command to sort names.txt and display the result on the screen.

………………………………………………………………………………………………

1. Use the sort command to sort names.txt and store the result in a file called sortednames.txt

………………………………………………………………………………………………

**Task 4 – Cat command**

The cat command is considered as one of the most frequently used commands in **Linux.** It can be used for the following purposes under Linux:

* Display text files on screen.
* Copy what you type to a file.
* Combine the text files and displays to the standard output.
* Create new text files.

## cat command Syntax

The syntax is as follows:

**cat** filename **// Displays the file on screen.**

**cat** > filename **// It copies what you type to a file. To save and exit press the CONTROL and d keys (CTRL+D).**

**cat** file1 file2 **// Combines the two files and displays them to the standard output (screen). The files are not changed.**

**cat** file1 file2 > newcombinedfile

**// Combines the two files and saves it to a new file called newcombinedfile.**

**What do the following commands do?**

**cat reportlist.txt**

………………………………………………………………………………………………

**cat > inputfile.txt**

………………………………………………………………………………………………

**cat inputfile.txt reportlist.txt > newfile.txt**

………………………………………………………………………………………………

Note: >>

**cat inputfile.txt reportlist.txt >> newfile.txt**

………………………………………………………………………………………………

**Task 5: The who command**

To use this command just type **who** on its own.

*who*

What is this command used for?

………………………………………………………………………………………………

You can use redirection to write the list of users to a file, e.g.

w*ho > loggedon.txt*

**Task 6: The grep command**

Syntax: grep [option(s)] pattern [file(s)]

You can use **grep** to search for a word or pattern in a file. For example to find the word ‘science’ in a file called file1.txt, you would type

*grep Science file1.txt*

**Important:** the file called file1.txt is in the CFY folder. Copy this to your home directory and type in the command above.

**Grep** is case sensitive so the above example would not find lines that contain ‘Science’.

 To stop it being case sensitive, what switch should you use? …………

Write the full command: …………………………………………………………………

Search for the phrase **bubbling beakers** in the file.

………………………………………………………………………………………………

**What switches do you use with grep to do the following?**

a) Displays those lines that do **NOT** match

………………………………………………………………………………………………

b) Precedes each matching line with the line number

………………………………………………………………………………………………

c) Print only the total count of matched lines

………………………………………………………………………………………………

**Note:** You can use **multiple** switches with the one grep command.

**In your home directory:**

Create a file called file2.txt that contains the help information about the ls command.

………………………………………………………………………………………………

Use grep which is **not** case sensitive to find the word “use” or “Use” in the file.

………………………………………………………………………………………………

Find the phrase “list directory” in the file file2.txt

………………………………………………………………………………………………

Try each of the switches from a) b) and c) above when searching for the string “list directory”

………………………………………………………………………………………………

………………………………………………………………………………………………

………………………………………………………………………………………………

Delete the file: file2.txt.

………………………………………………………………………………………………

**Task 7: Using Wildcards**

Any time a filename is used, you can replace parts of the filename or all of it with wildcards. Any sensible combination you can think of can be used. You can also use wildcards to specify directories.

**Warning:**

When copying (cp), moving (mv), or removing files (rm) with wildcards, it's often wise to include the -i (interactive) option, to avoid mishaps.

|  |  |
| --- | --- |
| Wildcard | Description |
| \* | Represents any amount of characters, including none. |
| ? | Each '?' represents any single character. |

**Exercises on wildcards**

**Change to the bin directory in Linux /bin and carry out the following exercises:**

Write the Linux command to:

List all files that start with the letter f.

………………………………………………………………………………………………

List all files whose file name ends in the letter r and where the files extension starts with the letter b. For example files like tiger.bmp, manager.bmp

………………………………………………………………………………………………

List all files whose second letter in the name is ‘a’ and whose last letter in the name is e. For example files like dave.txt, manage.sxc

………………………………………………………………………………………………

List all files which have an extension. For example files like temp.txt, report.xls

………………………………………………………………………………………………

**Task 8: Pipes**

A very useful Linux feature is named pipes which enable different processes to communicate.

One of the fundamental features that makes Linux useful is the “pipe”. Pipes allow separate processes to communicate without having been designed explicitly to work together. This allows tools quite narrow in their function to be combined in complex ways.

* Pipe is a symbol used to allow the output of one command as input to another command. Commands are processed from left to right.

A simple example of using a pipe is the command:

  **ls | grep x**

When bash examines the command line, it finds the vertical bar character | that separates the two commands. Bash and other shells run both commands, connecting the output of the first to the input of the second. The ls program produces a list of files in the current directory, while the grep program reads the output of ls and prints only those lines containing the letter x.

**What will the following commands do?**

who | grep ‘cfy’

………………………………………………………………………………………………

who | grep ‘cfy’ | sort

………………………………………………………………………………………………

**Write the commands to do the following:**

1) Do a long directory listing of your home directory for any files that you created today. (Where the date matches todays date)

………………………………………………………………………………………………

2) Do the same as for number 1. This time save the results in a file called **res1.txt**.

………………………………………………………………………………………………

3) Do a **short** directory listing of all the first year accounts. Sort the output in **descending order.**

………………………………………………………………………………………………

4) Do the same as for number 3. This time save the results in a file called **res2.txt**.

………………………………………………………………………………………………

5) Do a **short** directory listing of all the first year accounts which are between cfy030 and cfy039.

Sort the output in **ascending order** and output to a file called **res3.txt**

………………………………………………………………………………………………

6) Do a **short** directory listing of all files and folders beginning with **n** in the **/bin** directory and save it to a file called **listing.txt** in your home account. Run this command from your home directory.

………………………………………………………………………………………………

**Revision ( taken from past exam papers)**

1. Write the command to do a short directory listing of files in the **/etc** folder whose last letter in the file name is ‘s’ and which are .conf files. (3 marks)

For example files like sensors.conf or ts.conf

………………………………………………………………………………………………

 Write the command to do a short directory listing of files in the home **/etc** folder whose second letter is ‘t’ and have 4 letters in filename. (3 marks)

 For example files like motd.tail

………………………………………………………………………………………………

1. Create a file called **rhyme.txt** (in your home directory) with the following data

 **THREE blind mice**

 **three blind MICE**

 **see how they run**

 Find the word MICE in the file rhyme.txt whilst ignoring the CASE. (3 marks)

………………………………………………………………………………………………

Find all lines in the file **rhyme.txt** that **do** **not** contain the word mice and output the number of these lines. (3 marks)

..............................................................................................................................................

1. View a sorted list of users who are logged into the system. (2 marks)

………………………………………………………………………………………………

1. View a sorted list of users who are logged into the system in **descending order** and **save** the output to a file called log.txt (3 marks)

………………………………………………………………………………………………

1. List all of the users who are logged on and search in the list for the user “**cfy073**”, output the result to a file loggedon.txt (Use one command to do this!)

You should test this in your linux303 account (4 marks)

..............................................................................................................................................