Operating Systems EE5012 - Laboratory

Graham Claffey

*18296661*

*16/Feburary/2019*

# *SDip in Embedded Systems Engineering*

**Laboratory Assignment #2**

# **Assignment objectives**

1. Learn some of the basic bash shell commands
2. Learn how to write and run a very simple shell script program
3. Learn how to extract some key system specifications and report them in a script

**Description of solution**

The first step to completing this assignment was to review the basic bash shell commands. I first undertook a series of exercises with the list of utilities that would be needed in the system\_info.sh script. To complete the basic exercises I used the man and info pages for the following list of utilities:

echo | whoami | id | hostname | uname | find |wc

ls | grep | set | sed | df | awk | sort | head | cat

These are listed in order of appearance in the system\_info.sh script.

**echo –** used this to output text to the screen from the stdout. In the system\_info.sh script, the escape sequence characters are used for formatting of the output.

**whoami –** used to display the currently logged in user to the screen.

**id –** used to print the user id. *~$ id - u*

**hostname –** used to print the name of the current machine.

**uname –** used to print information about the type of system being used. e.g. architecture, OS type etc.

**find –** used for multiple purposes in our script, the most common instance is using it to list a number of files or directories separately. *~$ find location -type f*

**wc – word count -** used to count word, lines or characters. We use it in system\_info.sh to count the number of line outputs with. *~$ wc -l*

**ls – list structure –** used to list the structure of a specified directory. Can list hidden files, list recursively through the file system. *~$ ls -lRa*

**grep – general regular expression(regex) print –** used to search files using regex with the used of metacharacters. *~$ grep -i “^string”*

**set –** used to print the system environment variables. e.g. SHELL, PATH, HOME, PPID

**sed – stream editor –** used to manipulate text from files one line at a time. Quite an extensive and powerful command can also filter undesirable characters from strings of text. *~$ sed -e ‘s/find/replace/g’ -e ‘s/multiple/operators/g’*

**df – disk filesystem –** used to report the file system disk space usage. We can use this to see the free space, available blocks, mount points. *~$ df -h*

**awk –** a programming language used for text retrieval and text processing on data files. Another extremely powerful utility that can format the output of data(anything inside the {} with a $ is a variable). *~$ awk ‘{print $4, $2}’*

**sort –** used to sort the output from another utility via piping | or from a text file using the redirection methods < input TEXTFILE output >. *~$ sort -grk5*

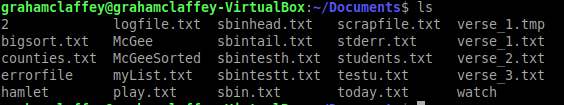
**head –** used to print the top specified number of lines from a text file or other command. *~$ head -10*

**cat – concatenate –** used to concatenate output, files, multiple files etc and print on the standard output. *~$ cat file1.txt file2.txt > bothfilescombined.txt*

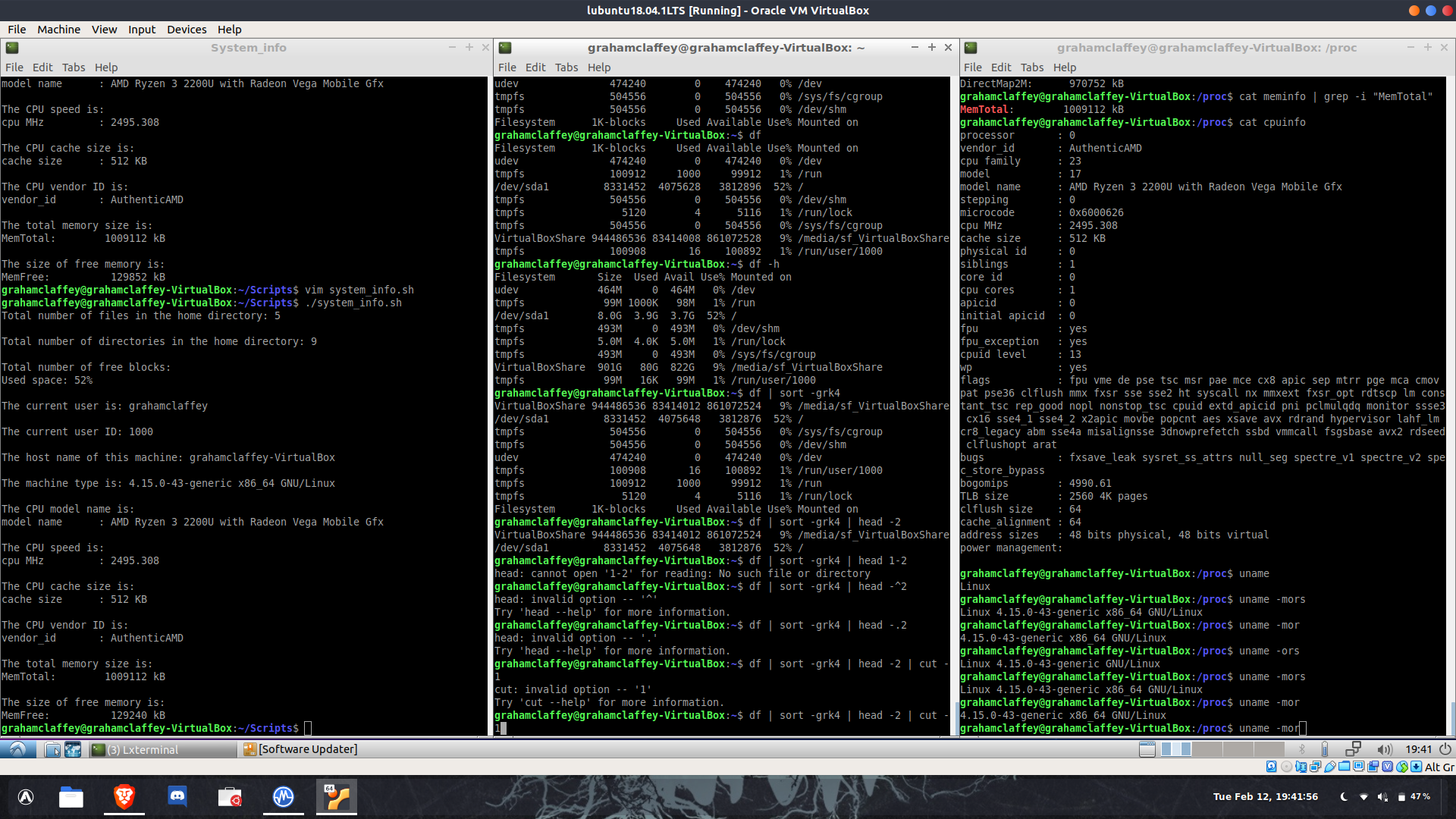
In addition to these commands, there is a necessity to learn about redirection and pipping, so that we can string these commands together within the script. Once finished reviewing and practising the commands that were needed for this script, I proceeded to systematically add functionally to my script.

**Testing and results**

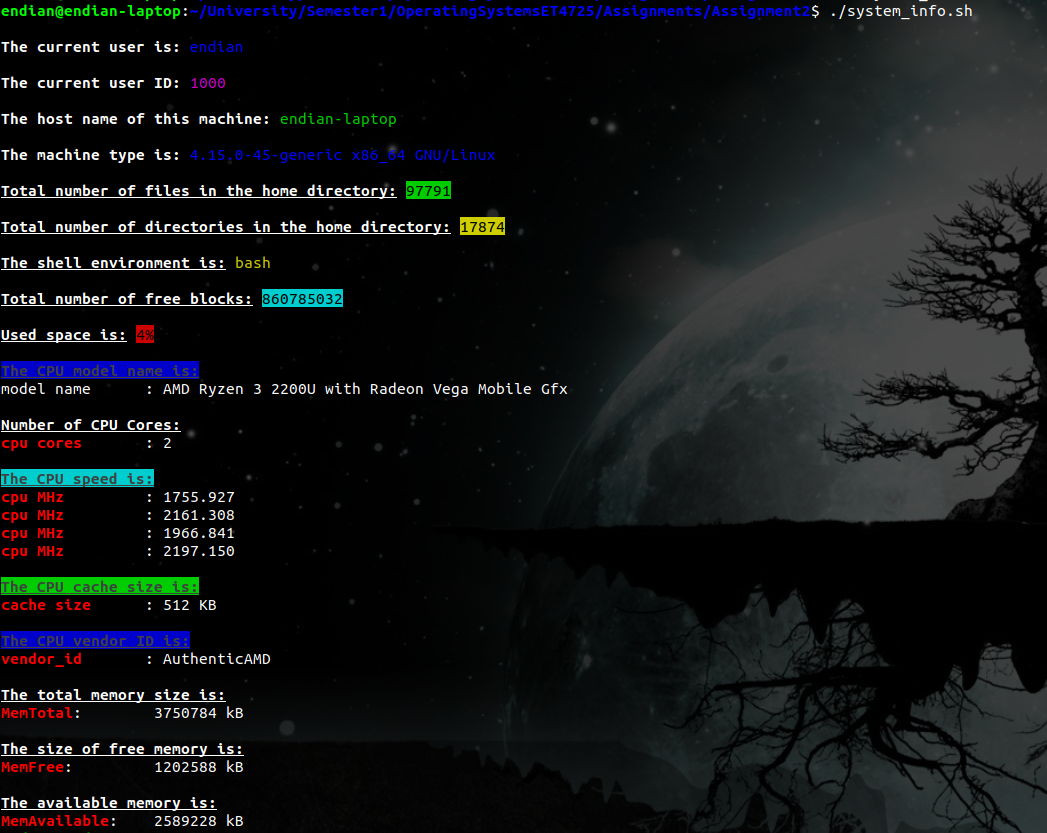
The testing phase was to create quite lengthy test files in multiple directories, as to test the functionality of the commands used in system\_info.sh. Then it was a matter of going through each segment that will eventually make the script.

****

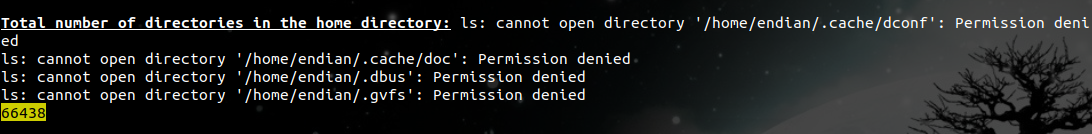
The next step was to test out the different components of the script in the terminal emulator when I achieved the desired result I would add those utilities to the script.

****

Once the script was completed, I then worked on the formatting using the echo command. With the echo command, I used the escape sequences to customize the output.

****

*Problem:*

A problem I ran into after the script was fully completed. On my lubuntu virtual machine the script worked as intended, but when I transferred the script to my main host OS, I was receiving "ls: Permission denied" errors.

I didn't want to use any sudo commands for this script, so to solve this problem I used the &n operator to merge stderr with its own stdout using, *~$ 2>&1*

*Solution:*

*ls -lRa ~ 2>&1 | egrep -c '^d'*

**Statement of completion**

With this assignment, I managed to use most of the utilities that were listed on week 3’s pdf file from the class. One of the only utilities that I didn’t find a use for was the *tail* command. I've learned more in depth about the commands from class because of this assignment. I have improved greatly in the area of data manipulation using the bash command line.

In total, I would estimate around 12 – 15 hours of study and work in this assignment, due to the extra exercises I laid out before beginning the script.

**Source code**

*#! /bin/bash*

*#Title: system\_info*

*#Description: This is a script used to display various information on the current system,*

*# and displays the system information to the user, in an easy to read and neatly formatted fashion.*

*#Format info: The formatting that is used is using the Bash escape sequences. If you wish to know more about*

*# these escape sequences visit: https://misc.flogisoft.com/bash/tip\_colors\_and\_formatting*

*#Commands: echo, cat, ls, grep, awk, sed(redundant-ish), sort, head, wc, uname, id, hostname, df, whoami, find*

*#Tools: Text editor: Vim, Terminal Emulator: Terminator*

*#Author: Graham Claffey*

*#ID: 18296661*

*#----------------USER Details---------------------------------------------*

*echo -ne "\n\e[1mThe current user is:\e[0m "*

*#print current user*

*echo -e "\e[94m`whoami`\e[0m"*

*echo -ne "\n\e[1mThe current user ID:\e[0m "*

*#print user id*

*echo -e "\e[35m`id -u`\e[0m"*

*echo -ne "\n\e[1mThe host name of this machine:\e[0m "*

*#print the name of the host machine*

*echo -e "\e[32m`hostname`\e[0m"*

*echo -ne "\n\e[1mThe machine type is:\e[0m "*

*#display machine type*

*echo -e "\e[94m`uname -mor`\e[0m"*

*#-----------------------SYSTEM INFO---------------------------------------*

*echo -ne "\n\e[1m\e[4mTotal number of files in the home directory:\e[0m "*

*#print home directory files*

*echo -e "\e[42m\e[30m`find $HOME -type f 2>&1 | wc -l`\e[0m" # result: 97704 5 files difference*

*# ls ~ lRa -1 2>&1 | grep '^-' | wc -l result: 97699*

*# find $HOME -type f 2>&1 | grep -v "Permission denied" | wc -l result: 97700*

*# ls ~ lRa 2>&1 | grep '^-' | grep -v "Permission denied" | wc -l result: 97700*

*# Seems to me like in this situation the grep -v is just omitting the Permission denied files*

*# which leaves a difference of 5 files in my $HOME, ~*

*# I will test this more soon...*

*# CONCLUSION: ls -lRa -1 2>&1 | egrep -c '^l|^-' # result: 977704*

*# Seems like I was missing the files with the symbolic link (l) identifier.*

*# With the help of egrep I got the same result as the find utility.*

*echo -ne "\n\e[1m\e[4mTotal number of directories in the home directory:\e[0m "*

*#print home directory, directory files*

*echo -e "\e[43m\e[30m`ls -lR ~ | grep '^d' | wc -l`\e[0m"*

*echo -ne "\n\e[1m\e[4mThe shell environment is:\e[0m "*

*echo -e "\e[33m`set | grep "SHELL=" | sed -e 's|/||g' -e 's/SHELL=//g' -e 's/bin//g'`\e[0m"*

*#--------------------DISK INFO--------------------------------------------*

*echo -ne "\n\e[1m\e[4mTotal number of free blocks:\e[0m "*

*#show free blocks*

*echo -e "\e[46m\e[30m`df | awk '{print $6, $4}' | sort -k1 | head -1 | awk '{print $2}'`\e[0m"*

*echo -ne "\n\e[1m\e[4mUsed space is:\e[0m "*

*#Show used space*

*#(sed is redundant for my purposes here, but if I want to use the Use% value as a variable, I can do so without worry)*

*echo -e "\e[41m\e[30m`df | awk '{print $6, $5}' | sort | head -1 | awk '{print $2}' | sed 's/%//g'`%\e[0m"*

*#---------------------------CPU INFO--------------------------------------*

*echo -e "\n\e[1m\e[4m\e[44m\e[30mThe CPU model name is:\e[0m "*

*#print cpu model*

*cat /proc/cpuinfo | head -10 | grep -i "model name"*

*echo -e "\n\e[1m\e[4mNumber of CPU Cores:\e[0m "*

*cat /proc/cpuinfo | head -15 | grep --color "cpu cores"*

*echo -e "\n\e[1m\e[4m\e[46m\e[30mThe CPU speed is:\e[0m "*

*#make a list of the cpu core speeds*

*cat /proc/cpuinfo | grep -i --color "cpu MHz"*

*echo -e "\n\e[1m\e[4m\e[42m\e[30mThe CPU cache size is:\e[0m "*

*#display the cpu cache size*

*cat /proc/cpuinfo | head -15 | grep --color "cache size"*

*echo -e "\n\e[1m\e[4m\e[44m\e[30mThe CPU vendor ID is:\e[0m "*

*#print the vendor id*

*cat /proc/cpuinfo | head -10 | grep --color "vendor\_id"*

*#----------------------------MEMORY INFO----------------------------------*

*echo -e "\n\e[1m\e[4mThe total memory size is:\e[0m "*

*#print total system memory*

*cat /proc/meminfo | grep -i --color "MemTotal"*

*echo -e "\n\e[1m\e[4mThe size of free memory is:\e[0m "*

*#print total free memory*

*cat /proc/meminfo | grep -i --color "MemFree"*

*echo -e "\n\e[1m\e[4mThe available memory is:\e[0m"*

*#print available memory*

*cat /proc/meminfo | grep --color "MemAvailable"*