**Linux Shell Scripting: A project based approach to learning**

27 April 2021

01:00

FIRST SCRIPT:

1. Creates new accounts
2. Checks for proper privileges
3. Reports if account creation failed

Vim commands:

* To edit: i[Enter]
* To save and exit: :wq!
* To exit without saving: :q!

#!/bin/bash - interpreter which the user selects to execute the file

# - Sharp

! - Bang

#! - Shebang

If you do not specify #!/bin/bash, the script will execute with your current shell as the interpreter

-rwxrwxrwx 1 vagrant vagrant 3090 Apr 27 06:24 Vagrantfile

* Rwx: read, write, execute permission for the vagrant user
* Rwx: read, write, execute permission to all the users residing inside "vagrant" group
* Rwx: read, write, Execute permission to all the rest of the users

To change file permission modes, use the chmod command. For eg: chmod 755 filename.sh

r: 4

w: 2

x: 1

755: rwx [(4+2+1)(4+1)(4+1)]

. - this directory

.. - parent directory

/ - directory separator

* For command, 'chmod 755 luser-demo01.sh' means 7(421-rwx) for the vagrant user of the file
* 5(4+1) is assigned to the users residing in the 'vagrant' group
* 5(4+1) is assigned to the rest of the users
* "touch" - creates an empty file if one doesn’t exist or updates the last timestamp of the file
  + touch filename. Extension
* "echo" - is a built-in shell. In other words, it's a command built-in the shell. It's part of bash
* "type" - is a built-in shell command. Eg: type(echo) - give description
* "type -a echo" gives all the variant of echo
* "/usr/bin/echo 'Hello' " - will print hello
* "help echo |less"- will give the entire description of the command which in this case is echo. '|less' depicts the description in a brief manner in a paged manner without scrollable feature that we can exit using the :q [ENTER]
* "uptime" - how long the system is up or online
* "man" - help manual for non-shell built-in commands
* "help" is for built in shell commands and "man" is for non-built in shell commands like online references
* # - comment
* Variables can start with a letter or underscore but not digits. From special characters only '\_' can be used for variable definition.
* Echo "variable\_name" - will get converted to a value
* Echo "variable\_name" - will not get converted to a value
* There is a difference between echo "This is $WORDing" and echo "This is ${WORD}ing". The former won't work since, the interpreter won't be able to determine when is $WORD is ending and if maybe, $WORDing is also a variable. Whereas, for the latter command, it will work splendidly.

SECOND SCRIPT:

1. Special variables created by shell
2. Output of a command in a variable
3. If statement

* Using the built-in shell variables with pseducodes in the form of comment statements for e.g.: #Display uid command: echo "Your UID is ${UID}"
* For navigating or searching inside the man pages, use the vim code editor key bindings
* For searching inside man page use: /name\_of\_the\_search\_word
* Click on 'n' for displaying the next occurrence of the searched keyword
* The only time EUID and UID will be different would in the case of SET UID script. Otherwise, it mostly remains the same and is read-only
* UID represents the current user, it is initialized at shell startup
* If a script contains SET UID = some\_value enabled then, no matter which user is executing the script, it will be executed with the some\_value permissions Rarely, seen that too in propreitary UNIX systems.
* '?some\_value' [ENTER] - will reverse search inside the man page
* Inside the man pages, if any word is written inside [] that means, its optional. For e.g. id [OPTION]…[USER]
* To store the output of a command inside a variable. Use the following command:
  + VARIABLE\_NAME=$(shell command)
  + Echo "printing the output of the command inside the variable: ${VARIABLE\_NAME}"
* Using if condition:
  + Ifspace[[space"$UID" -eq 0space] ]

Then

Echo "you are root user"

Else

Echo "you are not root user"

* Fi
* Variables inside [[ ]] always need to be represented inside double quotation marks
* Command separators in shell: ';', 'ENTER'
* [[ - shell keyword. (can check using type -a [[) - this is bash specific and might not work for other shells
* Another older way to include expressions inside if command is to use: if [some\_condition] but the single bracket usage is for an older version and the double brackets usage is the newer version
* For file, string, logical, comparison operations we can use the shell built-command called 'test'
* Root always has the UID is 0
* Sudo command - super user doer