UNIX Bash Shell Scripting

Week 1

Lecturer: Shahdad Shariatmadari

May 2020

Let’s start

- Please introduce yourself
  - Tell me about your background in IT, programming, development
  - Please introduce yourself at BB-> “discussion board”-> Introduction.

Agenda

- Course introduction
- Obtaining your Seneca accounts Changing passwords
- The Matrix server
- The role of an operating system
- File system in Unix
  - Basic file operation

UNX510/DPS918

- BlackBoard
- Course website

Assessment

Lab	10%
Class Activity	15%
Quizzes	10%
Assignments	30%
Midterm	15%
Exam	20%

Opening Question

- What is an Operating System?

Definition - Operating Systems

- A computer can't do anything useful without a program — a program is defined as data and a list of instructions to follow.
- An operating system (OS) is a collection of programs which manage and control the basic operation of the computer, including:
  - Allocating resources (memory, disk space, network bandwidth, access to devices)
  - Managing files
  - Starting, stopping, and controlling other programs
  - Enforcing basic system security

Evolution – Unix OS

Unix is an operating system originally developed at Bell Labs starting in 1969.

Unix is:

- a *portable, interactive, multitasking, multiuser* operating system.
- *written in a language that can be used on different types of computers* (the C language)
- *interactive* – i.e. users can use the computer directly, and immediately see the results of their actions
- *a multitasking environment* - the operating system creates the illusion of performing multiple tasks at the same time by rapidly switching between them

Linux and GNU

- Although Unix source code was widely distributed at low cost to educational institutions, it was still controlled by AT&T and was therefore never completely free (cost) nor free (unrestricted freedom to modify and use it).

- Richard Stallman published the *GNU Manifesto* in 1984, which described the need for *Free Software* ("Free in the sense of free speech, not free beer"). The resultant GNU project developed free, open source replacements for most of the Unix programs, but not for the Unix kernel (the core program that interacted with and controlled the hardware).

Linux and GNU

- These programs were released under the *GNU General Public License* (GPL), which permits anyone to copy, use, and modify the software, as long as these rights are preserved for anyone receiving a subsequent copy of the software.

The GNU Operating System

Free as in Freedom

Linux and GNU

- In 1991, Linus Torvalds, a Finnish computer programmer, released the *Linux* kernel, eventually placing it under the GPL. The Linux kernel, GNU software, and some other components can be combined into a powerful, Unix-like operating system (it can't technically be called Unix, because it has never been certified to *be* Unix, but virtually everyone in the industry regards it as such).
- The combined GNU and Linux system is called *GNU/Linux* by some but just *Linux* by others (much to the dismay of Richard Stallman, who feels that the simple name *Linux* downplays the tremendous contribution made by the GNU Project).

Using Linux/Unix at Seneca

- Throughout your studies at Seneca you will use a variety of Unix/Linux systems, including:

  MATRIX – Practice & perform Linux commands

  ZENIT – Used for some advanced courses

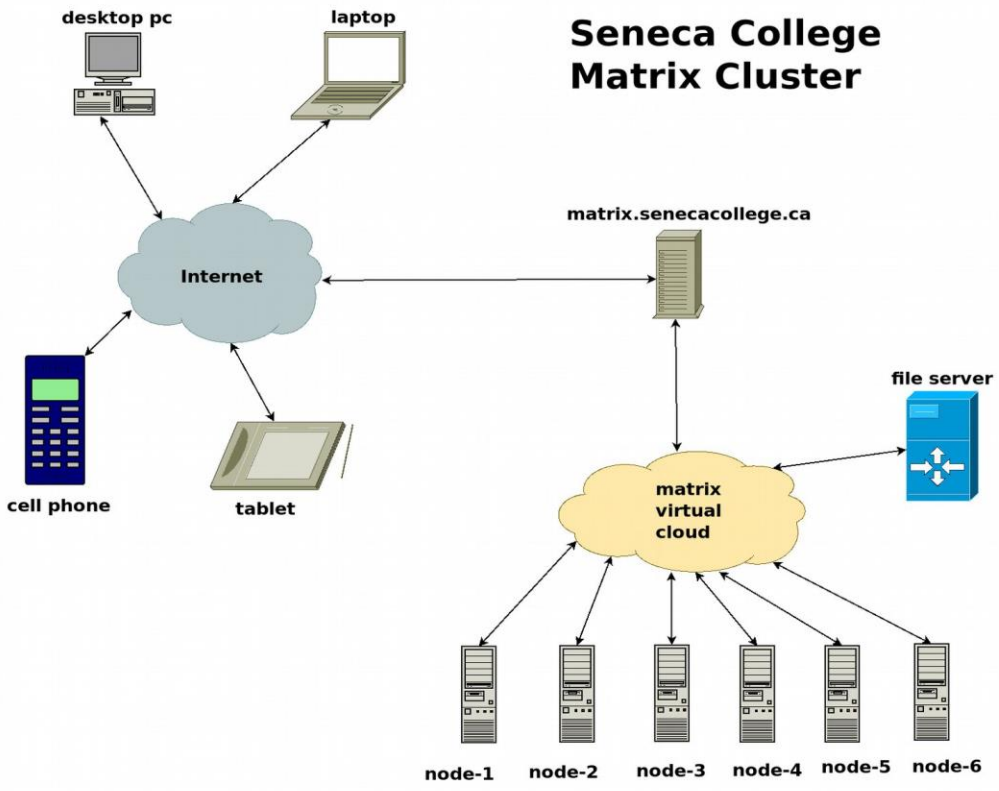
  ICT – Main ICT web server

  MY.SENECACOLLEGE.CA (Blackboard) – College LMS (Learning Management System)

- Most college servers are available under servername.senecac.on.ca and/or servername.senecacollege.ca

Matrix Server

The Matrix server consists of several Virtual Computers running CentOS all connected together to form a cluster. A cluster is a cost effective alternative to larger servers.



Matrix Server

- Note that the workstations in the labs can connect to the Matrix cluster. When you boot (startup) a PC in a Seneca lab, you can use the ssh or Putty app in Windows to open a connection to Matrix.
- An alternative method to connect to Matrix from the Windows desktop is to use the Knoppix VM. Knoppix is a Linux virtual machine. Once you start it you can open a terminal window and issue the
- ssh command like this

  ssh username@matrix.senecacollege.ca

Interacting with Unix/Linux

- • You will be shown the use of a telnet application (such as ssh or putty) to connect to your Matrix account.
- • When the telnet application runs and connects to the server, it acts like a terminal that is physically connected to the computer. In this case, the monitor is the telnet window on your PC, and accepts keyboard entry.

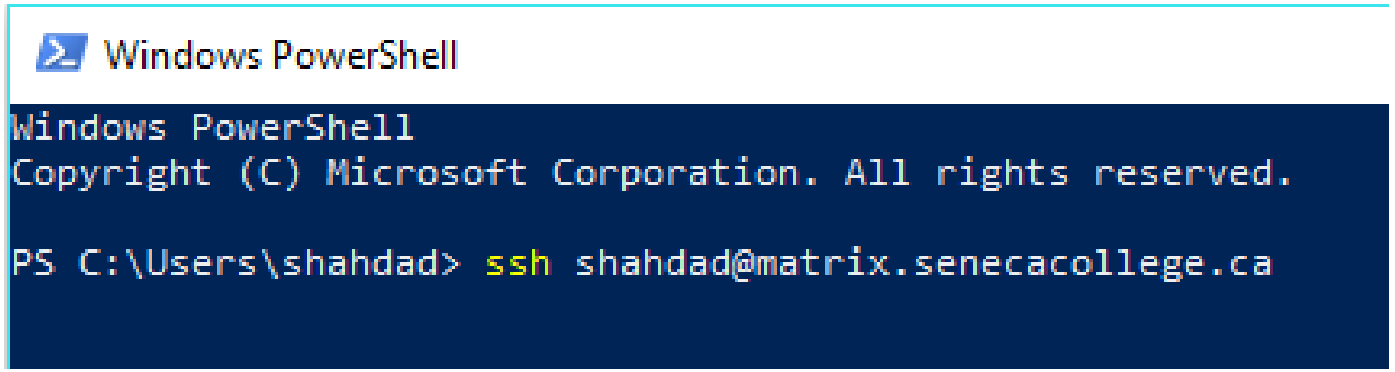
Secure Communication

- SSH or “secure shell” application allows data (i.e. keystrokes) to be encrypted to prevent other people intercepting this information.
  - Other than that, the functionality is similar to telnet
- In the Seneca Labs in MS Windows, there is an application on the desktop called SSH Client. You are advised to use this application. You can download SSH applications for your PC at home.
- On a Unix/Linux host ssh is available on the command line, for example:

```
ssh user@matrix.senecac.on.ca
```

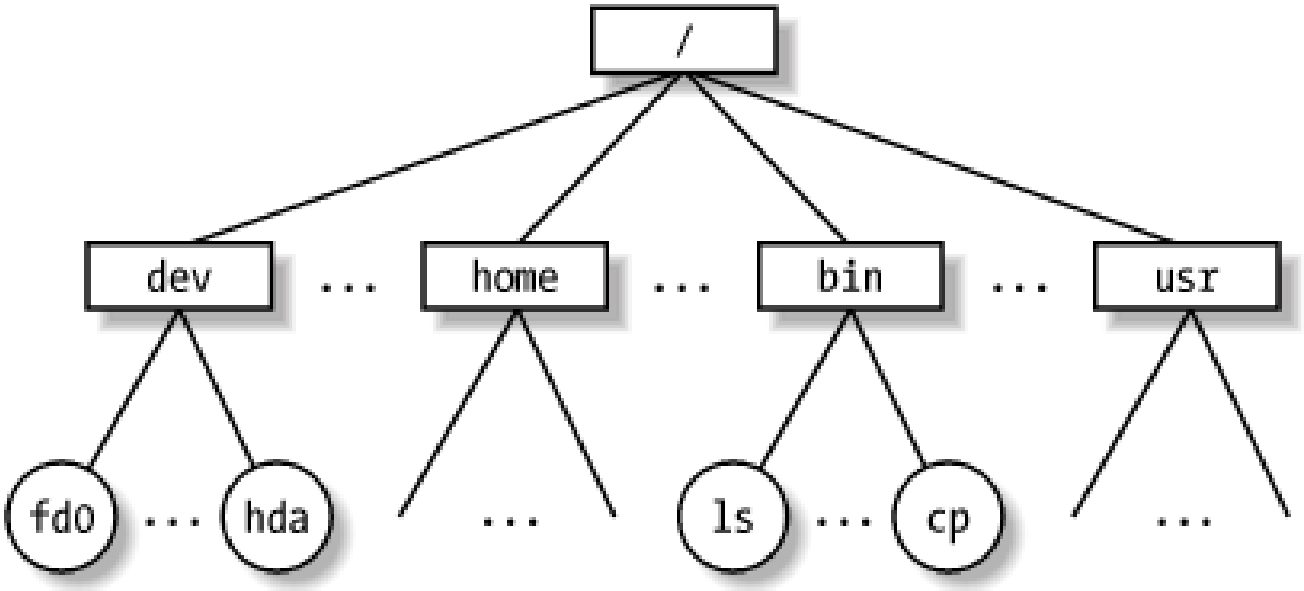
How to connect to Matrix

- Using Putty
- Using **Windows power shell** :
  - ssh username@matrix.senecacollege.ca



What is filesystem?

- In OS, a file system or filesystem, controls how data is stored and retrieved.
- From the user's point of view, files are organized in a tree-structured

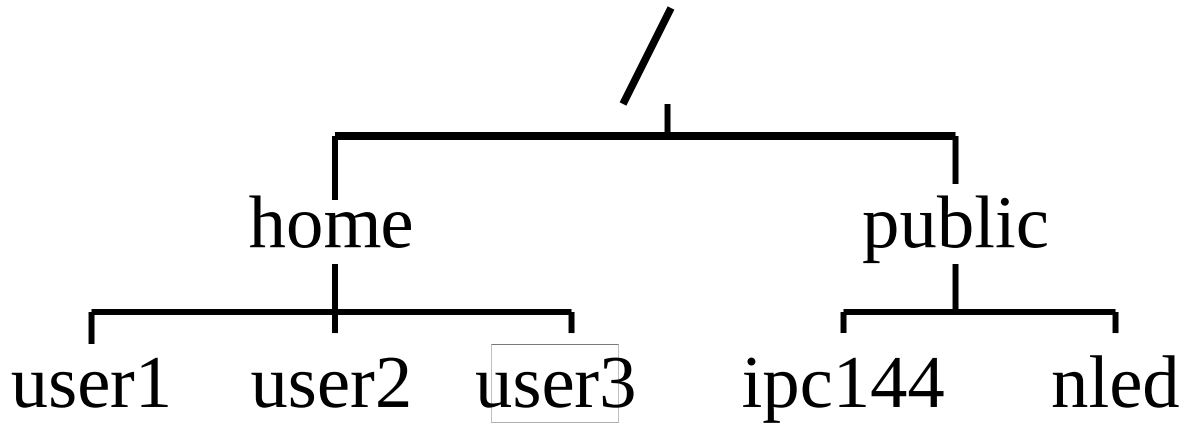


Unix File System

- The Unix/Linux file system is hierarchical, similar to other operating systems today
  - Files are organized in directories
  - Directories may contain sub-directories
- What is different (from Windows) is that there are no drive letters (such as C:, or D:)
  - All files and directories appear under a single root, even if multiple storage devices are used

Hierarchical File System

- In the Linux (Unix) OS, the "root directory" / is the starting directory, and other "child directories", "grandchild directories", etc. are created
- The hierarchical structure resembles an "upside-down tree". There is actually a command called tree that can display a "tree diagram"!



Typical Unix/Linux Directories

/	Root directory (ancestor to all directories).
/home	Used to store users' home directories.
/bin	Common system binaries (commands).
/usr/bin	Common utilities (commands) for users.
/usr/sbin	Common utilities for system administration.
/etc	System administration files (eg. passwd)
/var	Dynamic files (log and mail files)
/tmp, /var/tmp	Temporary files for programs
/dev	Device driver files (terminals, printers, etc.)

Basic Commands

pwd

- Used to display the user's present working directory. A user may need to know where they are located on the computer system in order to build directories, copy files, etc…

cd *directorypath*

- Used to change to a directory. Entering the cd command without a directory name will change to the user's home directory.

Home directory

- Every user when receiving an account has a "home" directory created
- This is where you keep your personal files
- ~ represents your home
  - You can use the ~ symbol in pathnames
- A cd command without any argument will get you directly to your home directory
- Remember to keep your files private

Basic Commands

mkdir *directorypath*

- Used to create a directory. Multiple arguments can be used to create multiple directories. The option –p (parent) allows multiple directory levels to be created.

rmdir *directorypath*

- Used to remove only empty directories (i.e. directories that contain no subdirectories or regular files). A user cannot remove a directory from within the directory itself.

Basic Commands

ls

- Used to display the contents of a directory (eg. regular files or sub-directories). By default, the ls command displays non-hidden filenames only.

- The following are common options available with the ls command:
  - **-a** short display of hidden & non-hidden files
  - **-l** detailed display of files (excl. hidden files)
  - **-d** combined with -l option, displays info about the directory itself instead of the files within it
- Options can be combined, for example: ls -la (or ls -l -a)

Types of Files

- On a Unix/Linux file system a "file" can be anything
    - To an average computer user a file is a text document, video, music, photo etc.
- A directory is really an index file, containing references to file locations on the physical disc and other related information
- Devices such as the terminal or printer are also files
    - You will learn more details about this later in this course
- Any file (or directory) name starting with a period is considered to be a hidden file

Types of Files

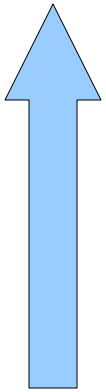
- You can use the ls -l command to determine the type of file.

For Example:

```
ls -l /dev/tty
crw-rw-rw-    1 root  root  5, 0 2003-03-14 08:07 /dev/tty

ls -l monday.txt w1.c
-rw-r--r--    1 someuser users 214 2006-01-23 14:20 monday.txt
-rw-r--r--    1 someuser users 248 2005-10-12 13:36 w1.c

ls -ld uli101
drwxr-xr-x    2 someuser users 4096 2006-01-17 16:43 uli101
```



You can determine file type from looking at first character in detailed listing:

- `-` indicates a regular file
- `b` or `c` indicates a device file
- `d` indicates a directory

Note: you can use the **`-d`** option with the **detailed listing command** to get information for just the directory itself, not the filenames within it

eg. **`ls -ld /home/myacct`**

Hidden Files

- A file is hidden if its name starts with a .
- For example: .profile
- ls –a will show all files including hidden
- . and .. directories are hidden
  - ls –A will show "Almost" all files – not including . and ..
- Why make files hidden?
  - To clean up directories
  - To hide backups
  - To protect important files from accidental deletion
- Remember: directories are really files, you can hide them as well

How to create files?

- Using **touch** command
  - How to use it?
- Using **vi** command
  - How to use it?

Activity

- Create a new folder in your HOME folder, name it May19-Act1
  - Create another folder inside May19-Act1, name it “invoice”
  - Create another folder inside May19-Act1, name it “report”
  - Using touch command create three files inside invoice folder: Jan.txt, Feb.txt, Mar.txt
  - Using vi, create a new file in report directory, name is summary.txt
    - Add you name, your courses in this semester inside the file
  - Show list of all files/directories in “May19-Act1” (all details)
  - Show the tree view of “May19-Act1”
  - Using vi, Create a txt file and add your name and today date as its content.

File Naming

- Unix/Linux is case sensitive!
- Adopt a consistent file naming scheme – this will help you find your files later
- Make your file and directory names meaningful
- Avoid non alphanumeric characters, as they may have a special meaning to the system that will make your work more difficult
- Avoid using spaces in file names – consider periods, hyphens, and underscores instead
- Feel free to use file name extensions to describe a file's purpose

Working With The File System

- Be very careful when working with files on the command line, as there is no undo command or a Trash/Recycling Bin
  - A single command can wipe out all your files
  - Changes are instant and permanent
- Make backups of important files, preferably outside of your account – USB storage is a good option
- You will learn later additional ways to control file access through file permissions which will help you prevent accidental file damage or deletion

Getting Help with Commands

A comprehensive online manual for common UNIX/Linux commands exists on your server

The online manual is available by using the command man

Command Structure:

man [options] *command*

Options:

-k provides short (one-line) explanations relating to the commands matching the character string. This can be useful if the user doesn't remember the name of a command, eg. man -k calendar

vi (Visual) Editor

vi is a powerful, interactive, visually-oriented text editor

Features:

- Efficient editing by using keystrokes instead of mouse.
- Use of regular expressions.
- Possibility to recover files after accidental loss of connection.
- Features for programmers (eg. line numbering, auto-indent, etc.)

Although you may prefer to use other editors (such as nano or nled), knowing vi is very useful, as this is one editor that is present on all Unix-like systems

Starting vi Session

There are two ways to start an editing session with vi:

- Enter **`vi filename`** -recommended since filename has already been assigned and changes will be saved to that filename when saving within vi, for example **`:w<ENTER>`**
- If the filename exists, it will be edited. If the filename doesn't exist, it will be created.
- Enter **`vi`** - filename is not assigned, therefore user has to type **`:w filename<ENTER>`** in order to save the file.

Modes

- There are three operational modes while using the vi editor:
  - Command Mode (default mode when starting)
    - User presses letter(s) for a command – for example to input text, delete text, append text, etc.
    - Does NOT require <ENTER> key, the keystrokes are used individually.
  - Input Mode
    - Input Mode allows user to enter or edit text. Press <ESC> to return to command mode.
  - Last-line Mode
    - Pressing colon ":" opens a prompt at the bottom of the screen to enter more complex commands, such as search and replace. Requires <ENTER> key to execute command.

Moving in Command Mode

- You can move around to text in the screen by using the following keys:
  - **h (left), j (down), k (up), and l (right).**
  - **w (right one word or to special characters),**
  - **W (right one word including special characters)**
  - **b (left one word or to special characters),**
  - **B (left one word including special characters)**
  - **0 (zero) (beginning of line)**
  - **$ (end of line)**
  - **G (go to last line in file)**
  - **237G (go to line 237 in file)**
- You may be able to move around by using the arrow keys (depends on version of vi).

Getting into Input Mode

While in command mode, you can issue the following commands to input text:

i – insert to left of cursor
o – insert line below current line
a - append to right of cursor
r - replace character under cursor

I – insert at beginning of line
0 – insert line above current line
A - append at end of current line
R – overwrite text character-by-character

Don't forget to hit <ESC> to return to command mode.

Common Editing Commands

- x – Delete the single character under the cursor
- d – Delete
  - eg. **dw** - delete from the current position to the next word or special character
  - eg. **d$** - delete from the current position to the end of the line
  - eg. **dd** - delete the entire current line
- c – Change
  - eg. **cw** - change from the current position to the next word or special character
  - eg. **c$** - change from the current position to the end of the line
  - eg. **cc** - change the entire current line
- y – Yank (copy)
  - eg. **yw** - copy from the current position to the next word or special character
  - eg. **y$** - copy from the current position to the end of the line
  - eg. **yy** - copy the entire current line

Common Editing Commands

p – paste deleted or copied text after or below cursor

P – paste deleted or copied text before or above cursor

u – undo previous edit

. – repeat previous edit

Most editing commands can be preceded with a repetition factor, for example:

3x = delete 3 characters

2u = undo the last 2 edits

12dd = delete 12 lines

Searching

- Search for text (in command mode)

– /pattern	Search forward for pattern
– ?pattern	Search backwards for pattern
– n	Display next match

Saving Edited File

- Work performed during vi session is stored in a Work Buffer (temporary storage) until the user saves their work.

- To save your vi session, make sure you are in command mode by pressing <ESC>

- To save your changes and exit, type ZZ (two capital z's). You can also use either :x<ENTER> or :wq<ENTER>

- You can save without exiting by typing :w<ENTER>

Aborting Editing Session

- If you make a mistake in your editing session that undo cannot easily solve, you can abort your session without modifying the contents of your file by using the following last-line command:

:q!<ENTER>