Linux Core Features

...

DAY 1

Shells n' such

Introduction to Linux (1)

What is Linux?

What is Linux not?

Common Shells (1)

What is a shell?

- A program that users employ to run commands
- A command language interpreter that allows Linux to understand your commands

Common Shells (2)

Common Shells:

- Bourne (sh)
- Bourne Again Shell (bash)
- CSH
- TCSH

Common Shells (3)

Bourne Shell (sh):

- Minimal Features
- No job control, aliasing, history, filename completion, etc

Common Shells (4)

Bourne Again Shell (bash):

- Improvement on sh

Features:

- Job control, aliasing, command history, command line editing with arrow keys, filename completion, etc

Does not have built-in list data type

Common Shells (5)

CSH and TCSH

- Shells are similar in functionality to bash
- Cannot follow symbolic links invisibly like bash
- Do have built-in list data type for scripting

Common Shells (6)

Finding your shell

- Echo $SHELL <- shell variable stores location of shell binary
- Cat /etc/shells

Login Shell

First process spawned that executes under your UID during an interactive session

- Process displayed with ‘-’ character
- Shell gained from text console
- Not created during GUI login (session || windows manager instead)

non-Login Shell

Shell invoked from another shell

- Shell inside another shell (upgrading from sh with bash)
- Opening terminal in GUI session
- When a shell runs a script or a command passed on its command line

Interactive vs non-Interactive

Interactive

- Reads commands from user input

non-Interactive

- Shell is running scripts (i.e. cron)

Shells

Demos:

Determine BASH mode

Determine Shell Type

Why do Shells matter

/etc/bash.bashrc # file applies only to interactive BASH shells

/etc/profile # file applies only to Bourne and BASH compatible shells; SETS $PATH VAR: echo $PATH | tr ':' '\n'

~/.bashrc # file applies only to BASH non-login shells

~/.profile # file applies only to login shells

Commands, Arguments

Demo

Standard Input

STDIN: Standard input

Input into a program, file descriptor 0

Represented as 0< or < when redirecting standard input from a file

Standard Output

STDOUT: Standard output

Output of a program, file descriptor

Represented as 1> or > when redirecting standard output to a file

Standard Error

STDERR: Standard error

Output of a programs error handler

Represented as 2> when redirecting standard error to a file

Redirection

Write to a file “>”

Append a file “ >>”

Merge streams example: 2&>1

Pipes (1)

Unnamed pipe: |

- Redirect output of one command to input of another command
- Uni-directional
- Opened at time of creation

Pipes (2)

Named pipe / FIFO

- Created with mkfifo command or mknod p command
- Exist on filesystem with a name
- Can be accessed by unrelated processes
- bi-directional

Pipes (3)

Demos

- Unnamed
- Named
- MKNOD