Bash Shell

- Many shell programs exist.
- Most popular shell is the “Bash” (Bourne Again Shell).
- Users interact with a system by executing *commands* which are interpreted by the shell and transformed into actions by the kernel.

```
sysadmin@localhost:~$ ls
Desktop Documents Downloads Music Pictures Public Templates Videos
```

- The Bash shell has numerous built-in commands and features including: aliases, re-executing commands, wildcard matching, input/output redirection, pipes and background processing.

What is a Command?

- A program executed on the command line.
- Sources of commands include:
  - Internal (built-in shell) commands
  - External commands stored in binary files
  - Aliases
  - Functions
  - Scripts

Aliases

- An *alias* can be used to map longer commands to shorter key sequences.
- To determine what aliases are set on the current shell use the `alias` command.
- New aliases can be created using the following format:

```
alias name=command
```

- Aliases created this way only persist while the shell is open. Once the shell is closed, the new aliases are lost.

Basic Command Syntax

- Command syntax:

```
command [options…] [arguments…]
```

- Commands, options and arguments are all case-sensitive.
- To execute a command, the first step is to type the name of the command.

```
sysadmin@localhost:~$ ls
Desktop  Documents  Downloads Music  Pictures Public Templates  Videos
```


Specifying Options

- *Options* can be used with commands to expand or modify the way a command behaves.

```
command [options] [arguments]
```

- *Short options* are specified with a hyphen – followed by a single character (ie `-a`).
- *Long options* for commands are preceded by a double hyphen `--` (i.e. `--all`).
- The *lone double hyphen* `--` option can be used to indicate the end of all options for the command.
- BSD style options do not use hyphens, just a single character (i.e. `a`)

Display System Information

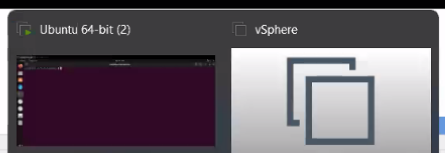
- The `uname` command displays useful system information.
- There are many options available for the `uname` command. For example:
  - `-a`, `--all` - displays all information about the system
  - `-s`, `--kernel-name` - displays Kernel name
  - `-n`, `--node-name` - displays network node name
  - `-r`, `--kernel-release` - displays Kernel release
  - `-v`, `--kernel-version` - displays Kernel version

Command Completion

- The Bash shell provides the ability to complete commands and their arguments automatically.

- Type a few characters of a command (or its file name argument) and then press the **Tab** key twice, this will provide a list of files that match.

```
sysadmin@localhost:~$ ca
cal                 capsh               cat                 cautious-launcher
calendar            captoinfo           catchsegv
caller              case catman
```



Sections Within Man Pages

- The format of each man page is broken into sub-sections:
  - NAME = Brief description.
  - SYNOPSIS = How command is executed.
  - DESCRIPTION = Provides a more detailed description of the command.
  - OPTIONS = The options for the command.
  - FILES = Which files are used for the command.
  - AUTHOR= Provides the name of the person who created the man page and (sometimes) how to contact the person.

Searching by Name or Keyword

- To return all man pages that match a *name*:

```
man -f name
```

```
sysadmin@localhost:~$ man -f passwd
```

- To return all man pages that match a *keyword*:

```
man -k keyword
```

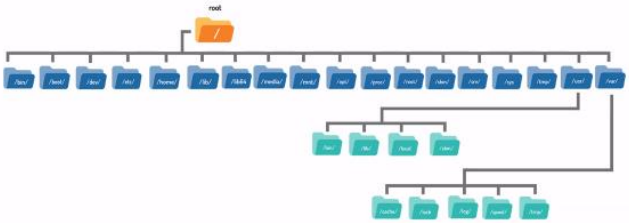
```
sysadmin@localhost:~$ man -k password
```

Filesystem Hierarchy Standard

- The Filesystem Hierarchy Standard (FHS) is standard that specifies standard directories and their content for use with a filesystem.
- Learning FHS helps you know what directories to expect to find and what to find in them.
- FHS allows programmers to write programs that will be able to work across a wide variety of systems that conform to this standard.

Filesystem Hierarchy Standard

- The FHS details many important directories.
- Administrators should know the directories on the next slides.



Important Directories

Directory	Purpose
/	The root of the primary filesystem hierarchy
/bin	Contain essential user executables
/boot	Contain the kernel and bootloader files
/dev	Populated with files representing attached devices
/etc	Configuration files specific to the host
/home	Common location for user home directories
/lib	Essential libraries to support /bin and /sbin executables
/mnt	Mount point for temporarily mounting a filesystem

Important Directories

Directory	Purpose
/opt	Optional third party add-on software
/root	Home directory for the root user
/sbin	Contains system or administrative executables
/usr/share/doc	Documentation for software packages
/usr/share/info	Information pages for software packages
/usr/share/locale	Locale information
/usr/share/man	Location for man pages
/usr/share/nls	Native language support files

Filesystem Hierarchy Standard

- A *shareable* directory, typically does not contain anything that would be unique to a particular system like a configuration file.
- A *static* directory usually doesn't change and may suggest that it might be mounted read-only.
- A *variable* directory is likely to change and would have to be available for both read and writes.

Finding Files and Commands

- A GUI typically provides a search tool that makes it possible to find files and applications.
- The CLI provides the `locate` and `find` commands which are useful for searching for a file within the filesystem.

locate Command

```
locate [OPTION]... PATTERN...
```

- The `locate` command searches a database that contains the location of the files on the filesystem.
- The `locate` command accepts a search string as an argument.

```
sysadmin@localhost:~$ locate passwd
/etc/passwd
/etc/passwd-
/etc/pam.d/chpasswd
/etc/pam.d/passwd
/etc/security/opasswd
```

- The `locate` command depends on a database which is updated using the `updatedb` command.

locate Command

- Advantages:
  - Fast because it searches a database of all files on the computer.
- Disadvantages:
  - New files are not in the database if it hasn't been updated.
  - You can only search for files by name versus other search criteria.

find Command

```
find [OPTIONS]... [starting-point...] [expression]
```

- The `find` command searches a live filesystem for specified files.
- The `find` command supports different search criteria options. The following table illustrates some examples of criteria:

`-iname FILE`	Case insensitive search by name.
`-mtime -3`	Files modified less than three days ago.
`-size +1M`	Files larger than 1 megabyte.
`-user jane`	Files owned by the user `jane`.

whereis Command

```
whereis [OPTION]... NAME...
```

- The `whereis` command displays the directory location and man page for the specified command.
- Searches only the directories defined by the `$PATH` variable.

```
sysadmin@localhost:~$ whereis grep
grep: /bin/grep /usr/share/man/man1/grep.1.gz /usr/share/info/grep.info.gz
```

- The `-s` option can be used to find source code that has been installed for a given command.
- The `-u` option can be used to identify commands that do not have an entry for a requested attribute.

which Command

```
which [OPTION]…. FILENAME…
```

- The `which` command displays the directory location(s) of a specified command or script.

```
sysadmin@localhost:~$ which bash
/bin/bash
```

- The `which` command returns the location of the real command.
- The `which` command searches only the directories defined by the `$PATH` variable.

type Command

```
type [OPTION]... NAME...
```

- The `type` command displays information about various commands.

```
sysadmin@localhost:~$ type echo
echo is a shell builtin
```

- Using the `-a` option can reveal the path of a command.

```
sysadmin@localhost:~$ type -a echo
echo is a shell builtin
echo is /bin/echo
```

- The `type` command supports other options and can lookup multiple commands simultaneously.