Introduction to UNIX

EECS 2031

Summer 2014

Przemyslaw Pawluk

June 10, 2014

What we will discuss today

- Introduction
- File System
- Files
- Commands
- Permissions
- Homework

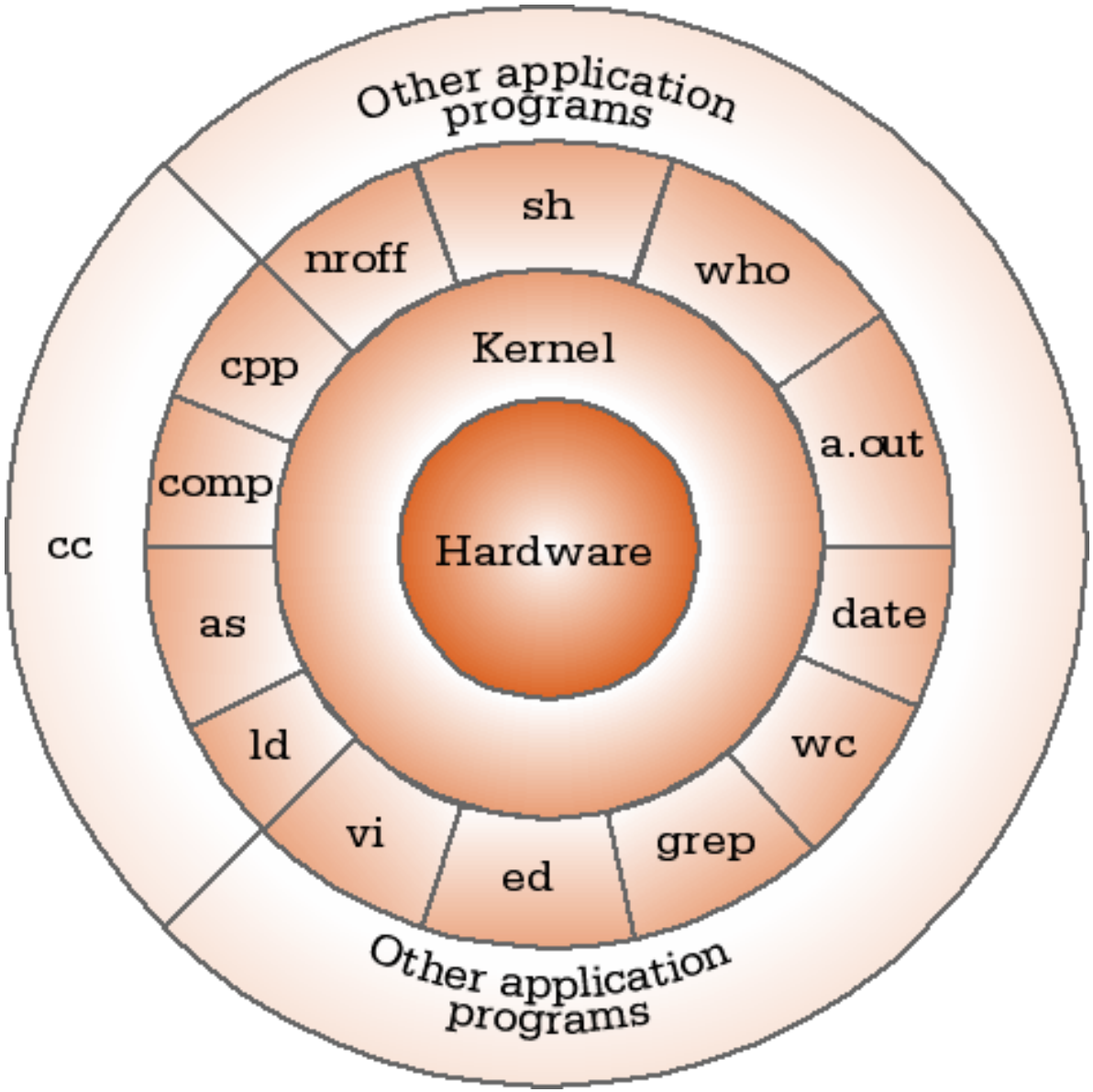
Table of Contents

What is UNIX?

- An Operating System (OS)
- Mostly coded in C
- It provides a number of facilities:
  - Management of hardware resources
  - Directory and file system
  - Execution of programs

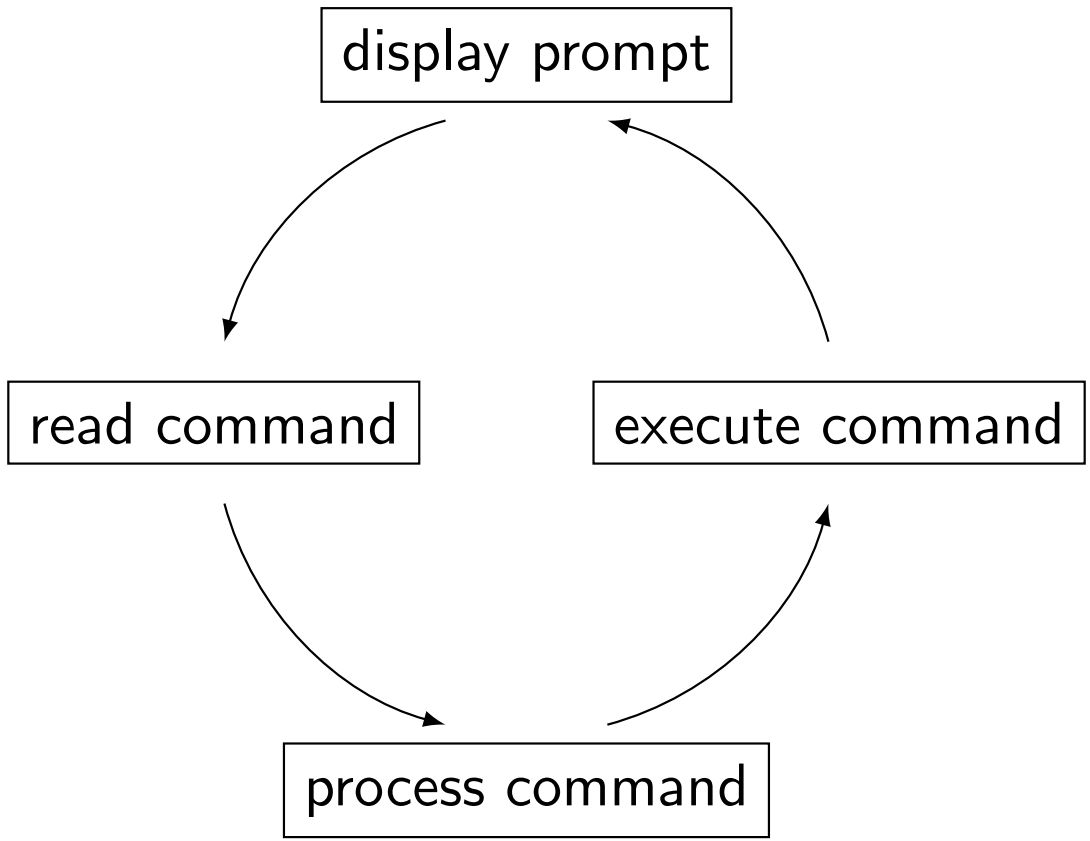
UNIX®

Kernel-Shell Relationship



The Shell

The shell does 4 jobs repeatedly:



Unix Commands

Various commands available

We will see some of the most useful ones

Selected commands that you already know

`ls`, `cp`, `mv`, `rm`, `cd`, `pwd`, `mkdir`, `rmdir`, `man`

Sample commands

- `date` Gives time and date
- `cal` Calendar
- `passed` Changes your password

System

- `uptime` Machines up time
- `hostname` Name of the machine
- `whoami` Your name
- `who`

echo

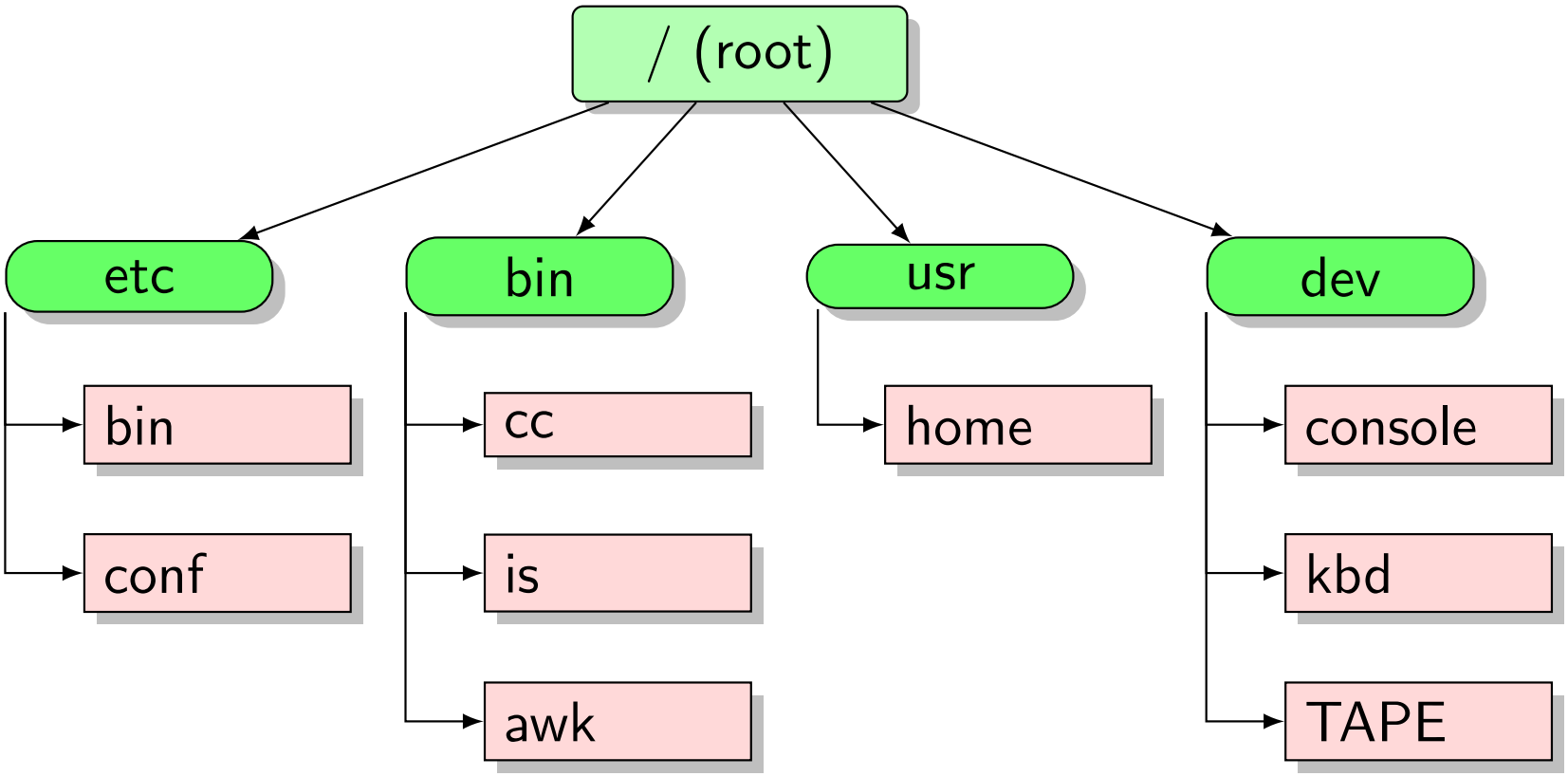
- When one or more strings are provided as arguments, `echo` by default repeats those strings on the screen
- It is not necessary to surround the strings with quotes, as it does not affect what is written on the screen
- If quotes (either single or double) are used, they are not repeated on the screen
- To display single or double quotes, use `\'` or `\"`

Table of Contents

The File System

- Directory structure
- Current working directory
- Path names
- Special notations

Directory Structure



Current Working Directory

- In a shell, the command `is` shows the contents of the current working directory
- `pwd` shows the current working directory
- `cd` changes the current working directory to another

Path names

- A path name is a reference to something in the file system.
- A path name specifies the set of directories you have to pass through to find a file.
- Directory names are separated by '/' in UNIX.
- Path names beginning with '/' are absolute path names.
- Path names that do not begin with '/' are relative path names (start search in current working directory).

Special Characters

- `.` means the current directory
- `..` means the parent directory
- `~` means the home directory e.g.: `cat ~/lab3.c`
- To go directly to your home directory, type `cd`

Wildcards (File Name Substitution)

- Allow user to refer to several files in one go.
- How to list all files in the current directory that start with `'e'`?

"?" – Matches single character

```
ls a?.txt
```

"*" – Matches multiple characters

```
ls e*
```

"[...]" – Matches all listed characters

```
ls lab[123].pdf
```

Table of Contents

cat, more, tail, head

cat

```
% cat phone_book
Yvonne 416-987-6543
Amy 416-123-4567
William 905-888-1234
John 647-999-4321
Annie 905-555-9876
```

more

```
% more phone_book
```

Similar to cat, except that the file is displayed one screen at a time

head and tail

```
% tail myfile.txt
```

Display the last 10 lines

Word count – wc

wc

Print byte, word, and newline counts for each FILE, and a total line if more than one FILE is specified. With no FILE, or when FILE is -, read standard input.

cmp, diff

```
% cat phone_book
Yvonne 416-987-6543
Amy 416-123-4567
William 905-888-1234
John 647-999-4321
Annie 905-555-9876

% cat phone_book2
Yvonne 416-987-6543
Amy 416-111-1111
William 905-888-1234
John 647-999-9999
Annie 905-555-9876
```

```
% cmp phone_book phone_bool
phone_book phone_book2
differ: char 9, line 2

% diff phone_book
phone_book2
2c2
< Amy 416-123-4567
---
> Amy 416-111-1111
4c4
< John 647-999-4321
---
> John 647-999-9999
```

Stdin / Stdout

- Each Unix command reads input from standard input (stdin) and produces output to standard output (stdout)
- By default, stdin is the keyboard, and stdout is the screen
- But this can change

Input / Output Redirection

- Redirect output to a file (overwriting): `command > file`
- Append output to a file: `command >> file`
- Read input from a file: `command < file`

Pipes

Pipe is ...

... a way to connect the output of one program to the input of another program without a temporary file.

```
ls -l | wc -l          # count number of files
who | sort             # sort user list
who | wc -l            # count number of users
```

tee

tee copies its input to a file as well as to standard output (or to a pipe).

```
% date | tee date.out
Tue Nov 9 13:51:22 EST 2010
% cat date.out
Tue Nov 9 13:51:22 EST 2010
% date | tee date.out | wc
  1 6 29
% cat date.out
Tue Nov 9 13:52:49 EST 2010
```

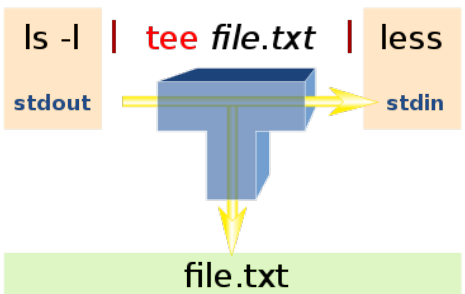


Table of Contents

Command Terminators and Comments

- New line or ; - to execute in order
- & - do not wait for command to complete (run in background)
- If a shell word begins with #, the rest of the line is ignored.

Quotes

Single Quotes

The single quotes should be used when you want the text left alone. If you are using the C shell, the "!" character may need a backslash before it.

Double Quotes

Double quotes doesn't expand meta-characters like "*" or "?," but does expand variables and does command substitution.

Back Quotes

To use the output of a command `X` as the argument of another command `Y`, enclose `X` in back quotes: `` `X` ``

Comparison of quotes

```
% echo The time now is `date`
The time now is Tue Nov 9 13:11:03 EST 2010

% echo "The time now is `date`"
The time now is Tue Nov 9 13:11:15 EST 2010

% echo 'The time now is `date`'
The time now is `date`
```

Table of Contents

File permissions

Show

`ls l`

Each file will come with a 10-character string e.g.: `-rwxr--r--`

How to read it?

[Owner, Group, Others]X[read, write, execute]

`-rwxr--r--`

The owner of this file can read, write, and execute this file, but everybody else can only read it

Note

The first character is `file type` and is not related to permission. It can take following values: - for regular file, d-directory, l-link, p-pipe, c-character device, b-block device, D-Door (Sun only)

chmod permissions

Letter	Meaning
u	The user (owner of the file)
g	The group the file belongs to
o	The other users
a	all of the above (an abbreviation for ugo

Letter	Meaning
r	Permission to read the file
w	Permission to write the file
x	Permission to execute the file

chmod Command

```
chmod who+permissions filename # or dirname
chmod who-permissions filename # or dirname
```

Examples

```
chmod u+x my_script  # make file executable
chmod a+r index.html # for web pages
chmod a+rx Notes     # for web pages
chmod a-rx Notes
chmod a-r index.html
```

chmod with Binary Numbers

```
chmod u+x my_script
chmod a+r index.html

chmod a+rx Notes
chmod a-rx Notes

chmod a-r index.html
.
```

```
chmod 700 my_script
chmod 644 index.html

chmod 755 Notes
chmod 700 Notes
chmod 750 Notes
chmod 600 index.html
chmod 640 index.html
```

chgrp Command

```
chgrp grp_name filename # or dirname
```

Examples

```
chgrp submit lab1
chgrp labtest lab9
```

Note

To display the group(s) a user belongs to, use id command:

```
% id cse12345
uid=12695(cse12345) gid=10000(ugrad) groups=10000(ugrad)
```

Table of Contents

Homework

Create a simple C program that:

- Reads a list of integers from the a file `input.txt` (one per line)
- Sorts the integers using a binary tree
- Writes the sorted numbers to the file called `sorted.out`