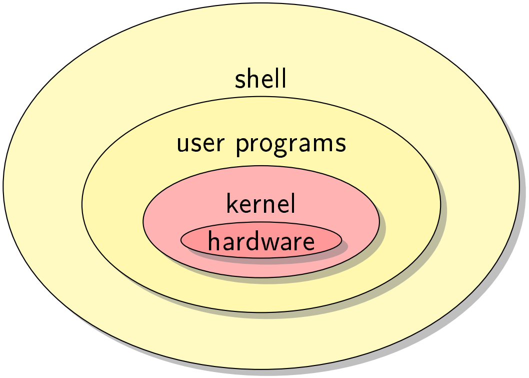
Fundamentals of Computing: Lecture 37

Piyush P Kurur
Office no: 224
Dept. of Comp. Sci. and Engg.
IIT Kanpur

November 9, 2009

The Unix system



The Shell

The Shell

- The command processor

The Shell

- The command processor
- In non-graphical mode is the first that is executed on login.

The Shell

- The command processor
- In non-graphical mode is the first that is executed on login.
- Is programmable.

The Shell

- The command processor
- In non-graphical mode is the first that is executed on login.
- Is programmable.
- Collection of shell commands forms a shell program.

The Shell

- The command processor
- In non-graphical mode is the first that is executed on login.
- Is programmable.
- Collection of shell commands forms a shell program.

Which shell?

The Shell

- The command processor
- In non-graphical mode is the first that is executed on login.
- Is programmable.
- Collection of shell commands forms a shell program.

Which shell?

There are many shells available, each slightly different syntax.
We will use the `bash` shell. Other common shells are `csh`, `zsh` etc.

Simplest shell program

The starting $ is the prompt. You dont have to type it.

```
$ programname arg1 arg2 arg2
```

e.g.

```
$ echo foo bar biz
```

Redirecting a file to input and output to file

Remember that every program has three open files, `stdin`, `stdout`, `stderr`

- If we write `$ cmd <foo` then the `stdin` of `cmd` is the file `foo` and not the keyboard.
- If we write `$ cmd >foo` then the `stdout` of `cmd` is the file `foo` and not the monitor.

```
$ cat > foo
I am redirecting the output of cat to foo.
$ cat < foo
I am redirecting the output of cat to foo.
$ exit
```

One can also redirect `stderr`

One can also redirect `stderr`

- Redirecting `stderr` to a file `cmd 2>foo` (no space between 2 and >)

One can also redirect `stderr`

- Redirecting `stderr` to a file `cmd 2>foo` (no space between 2 and >)
- Redirecting `stderr` to `stdout` `cmd 2>&1`(no space between & and 1)

One can also redirect stderr

- Redirecting stderr to a file cmd 2>foo (no space between 2 and >)
- Redirecting stderr to stdout cmd 2>&1(no space between & and 1)

```
$ gcc badCFile.c
# Too much data  see in pages
$ gcc badCFile 2>&1 | less
```

One can also redirect `stderr`

- Redirecting `stderr` to a file `cmd 2>foo` (no space between 2 and >)
- Redirecting `stderr` to `stdout` `cmd 2>&1`(no space between & and 1)

```
$ gcc badCFile.c
# Too much data  see in pages
$ gcc badCFile 2>&1 | less
```

This also tells us about piping.

```
$ cmd1 | cmd2 | cmd3 | cmd4
$
```

Some useful programs

- `echo` prints its command line arguments
- `less` shows int `stdin` in pages
- `grep pattern` shows only those lines of its stdin that matches the pattern.