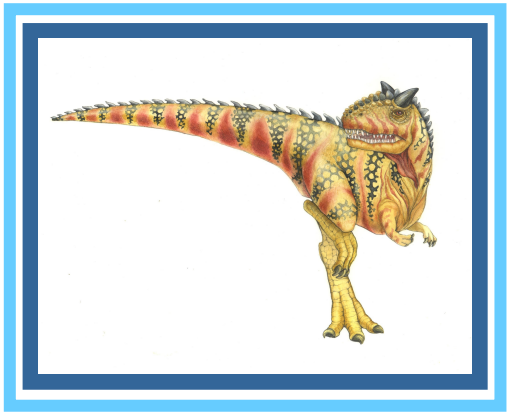
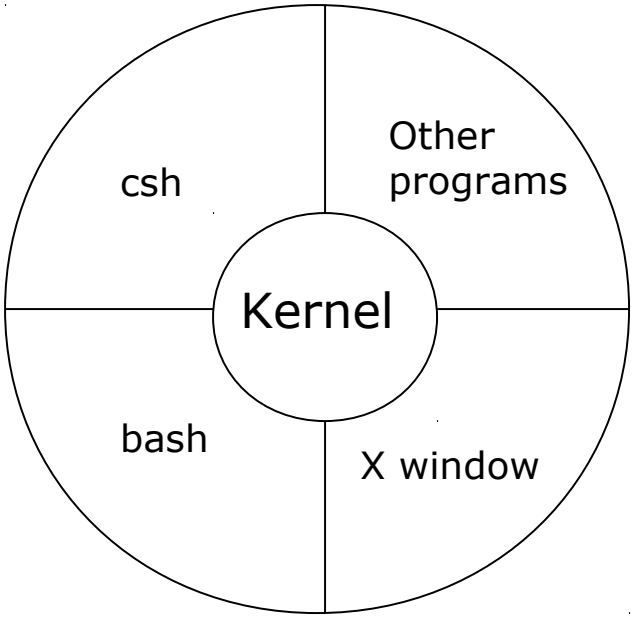
Chapter 2: Scripting



What is Shell?

- Shell is the interface between end user and the Linux system, similar to the commands in Windows
- Bash is **NOT ALWAYS** installed as in /bin/sh
- Check the version

```
% /bin/sh --version
% /bin/bash --version
```



Pipe and Redirection

- Redirection (< or >)
  - `% ls -l > lsoutput.txt` (save output to lsoutput.txt)
  - `% ps >> lsoutput.txt` (append to lsoutput.txt)
  - `% more < killout.txt` (use killout.txt as parameter to more)
  - `% kill -l 1234 > killouterr.txt 2 >&1` (redirect to the same file)
  - `% kill -l 1234 >/dev/null 2 >&1` (ignore std output)
- Pipe (|)
  - Process are executed *concurrently*
  - `% ps | sort | more`
  - `% ps -xo comm | sort | uniq | grep -v sh | more`
  - `% cat mydata.txt | sort | uniq | > dummy.txt` (generates an empty file !)

Some Command-Lines

```
echo        print out a string
            echo "$HOME is where I want to be"

cat         Output specified files in sequence
            cat file1 file2 file3

whereis     Show where a file can be found

printenv    Display all environment variables

grep        Get Regular Expression and Print

head        first few lines of output
            head -5 filename

tail        last few lines of output
            tail -8 filename
```

More Commands

```
$ pwd
$ ls
$    ls file; ls directory ; ls -a ; ls -l ; ls -R
$ cd
$   cd ..
$   cd /home/tim/projects
$   cd ~/projects
$   cd ~tim/projects
$   cd $HOME/projects
$ mkdir make-a-directory
$ rmdir a-directory
$ mv
$   mv oldfilename newfilename
$   mv file1 file2 file3 newtargetdirectory
$ cp
$   cp -r dir1 dir1copy
$ rm
$ pushd
$ pop
$ find
$   find . -ls
$   find . -type d -print
$   find . -type f -exec "echo" "{}" ";"
```

Shell as a Language

- We can write a script containing many shell commands
- Interactive Program:
  - grep files with POSIX string and print it

```
% for file in *
> do
> if grep -l POSIX $file
> then
> more $file
> fi
> done
Posix
There is a file with POSIX in it
```

  - '*' is wildcard

```
% more `grep -l POSIX *`
% more $(grep -l POSIX *)
```

Writing a Script

- Use text editor to generate the “first” file

```
#!/bin/sh
# first
# this file looks for the files containing POSIX
# and print it
for file in *
do
    if grep –q POSIX $file
    then
        echo $file
    fi
done
exit 0
```

```
% /bin/sh first
% chmod +x first
%./first (make sure . is include in PATH parameter)
```

Variables

- Variables needed to be declared, note it is case-sensitive (e.g. foo, FOO, Foo)
- Add '$' for storing values

```
% salutation=Hello
% echo $salutation
Hello
% salutation=7+5
% echo $salutation
7+5
% salutation="yes dear"
% echo $salutation
yes dear
% read salutation
Hola!
% echo $salutation
Hola!
```

Quoting

- Edit a “vartest.sh” file

```
#!/bin/sh

myvar=“Hi there”

echo $myvar
echo “$myvar”
echo `$myvar`
echo \$myvar

echo Enter some text
read myvar

echo ‘$myvar’ now equals $myvar
exit 0
```

Output

```
Hi there
Hi there
$myvar
$myvar
Enter some text
Hello world
$myvar now equals Hello world
```

Environment Variables

- $HOME home directory
- $PATH path
- $SHELL which shell
- $PS1 The first layer prompt (normally %)
- $PS2 The second layer prompt (normally >)
- $0 name of the script file
- $n argument n
- $$ process id of the script (current PID)
- $! last PID
- $? exit code
- $# number of input parameters
- $* all arguments as one list
- $@ all arguments as separated lists
- $IFS separation character (white space)
- Use ‘env’ to check the value

Condition

- test or ‘ [‘

```
if test -f fred.c
then
...
fi
```

```
if [ -f
fred.c ]
then
...
fi
```

```
if [ -f fred.c ];then
...
fi
```

Example 1

```
$ vi coba1.sh
#! /bin/bash
for VAR in Satu Dua Tiga Empat
do
     echo $VAR
done
exit 0

$ chmod +x coba1.sh
$./coba1.sh
Satu
Dua
Tiga
Empat
```

Example 2

```
$ vi coba2.sh
$ chmod 755 coba2.sh
$ ./coba2.sh 1 2 3

#!/bin/bash
echo "Ini PID[$$] dengan $# ARGUMEN YAITU:"
echo "(satu list)  $*"
for VAR in "$*" ; do echo $VAR ; done

echo "(satu per satu)  $@"
for VAR in "$@"
do
    echo $VAR
done
exit 0
```

coba.sh

```
$ ./coba2.sh 1 2 3

Ini PID[4857] dengan 3 ARGUMEN YAITU:
(satu list)  1 2 3
1 2 3
(satu per satu)  1 2 3
1
2
3
```

End of Scripting