



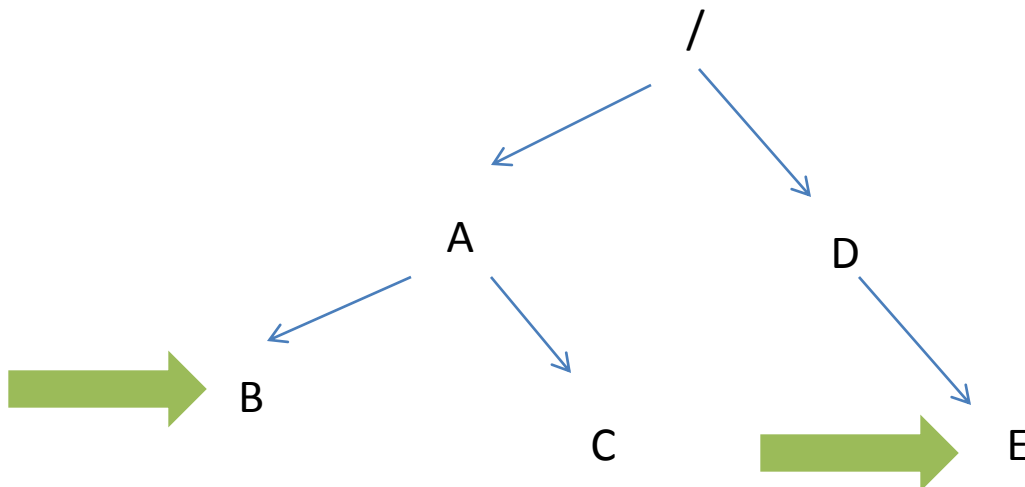
Linux Basics – Lecture 2

Lecture 1 - Recap

- History of Linux and UNIX
 - Why is UNIX most popular?
- Main parts of Linux Operating System
 - Kernel
 - Shell
 - File system
- How to use the shell
- Commands that you can use in Linux
 - Ls
 - cd

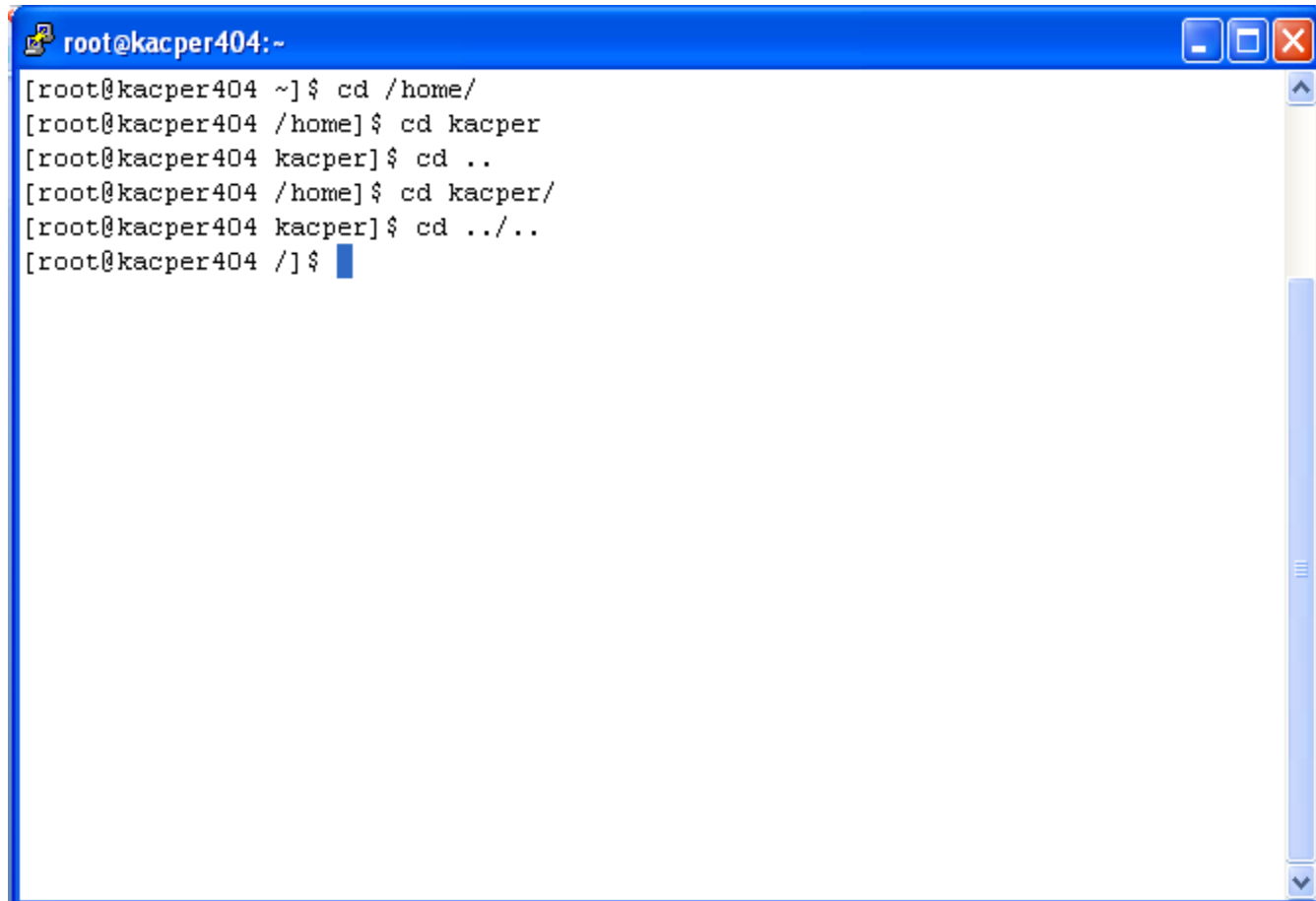
Lecture 1 – Activities

- Type “man bash” and “man tcsh” to understand the two shells
- Write 2 different “cd” commands to go to directory E from directory B (hint – One is using absolute method and the other using relative method)



LINUX Commands

cd command

A terminal window with a blue title bar and standard window controls. The title bar text is 'root@kacper404:~'. The terminal content shows a sequence of 'cd' commands being executed to navigate between directories: from '~' to '/home/', then to 'kacper', then to '..', then to 'kacper/', then to '../..', and finally to '/'. Each command is preceded by a prompt indicating the current directory and user. A blue cursor is visible at the end of the final prompt.

```
root@kacper404:~  
[root@kacper404 ~]$ cd /home/  
[root@kacper404 /home]$ cd kacper  
[root@kacper404 kacper]$ cd ..  
[root@kacper404 /home]$ cd kacper/  
[root@kacper404 kacper]$ cd ../..  
[root@kacper404 /]$
```

LINUX Commands

Command: **mkdir**

✓ To create a new directory use “**mkdir**”

✓ Syntax: **\$ mkdir directoryname**

✓ **\$mkdir -p dir1/dir2/dir3.**

It will create the directory tree.dir3 will created under dir2 and dir2 is created under dir1

✓ **\$ mkdir dir5**

\$ cd !\$.

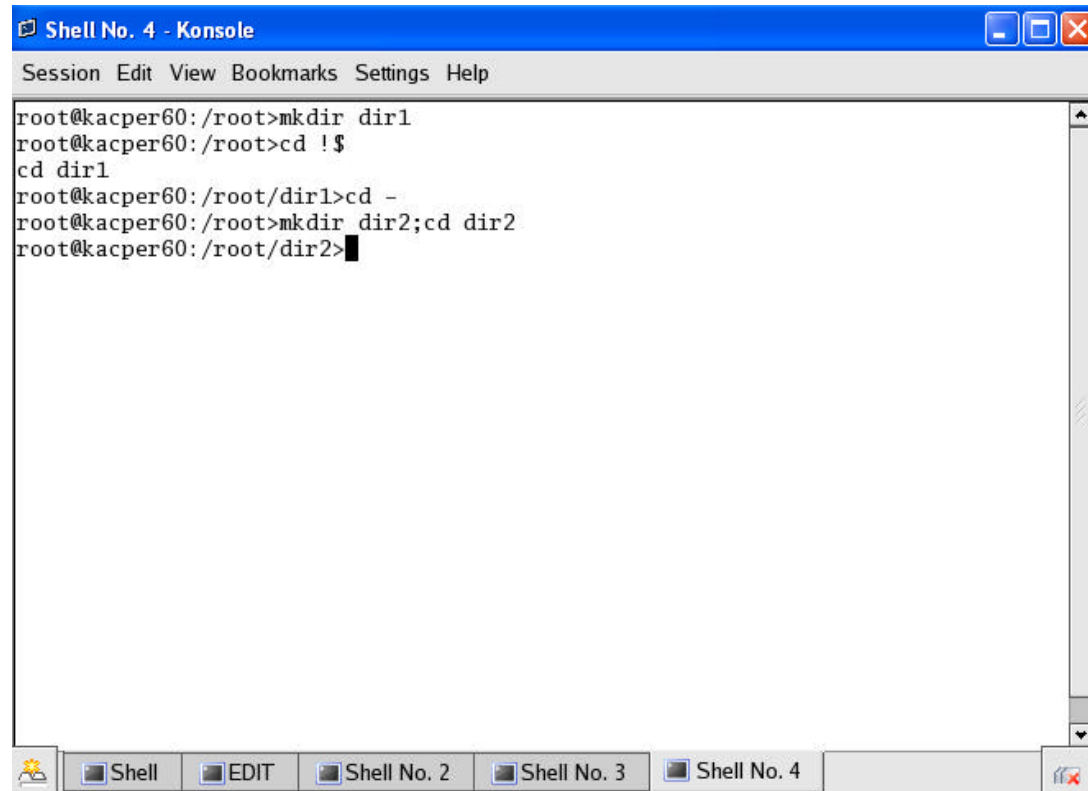
It will point the location of dir5

LINUX Commands

✓\$ **mkdir dir5; cd dir5.**

It will create dir5 first and then point the location of dir5.

Concatenation of above two commands.(';' called **command separator, explained later**)



```
Shell No. 4 - Konsole
Session Edit View Bookmarks Settings Help

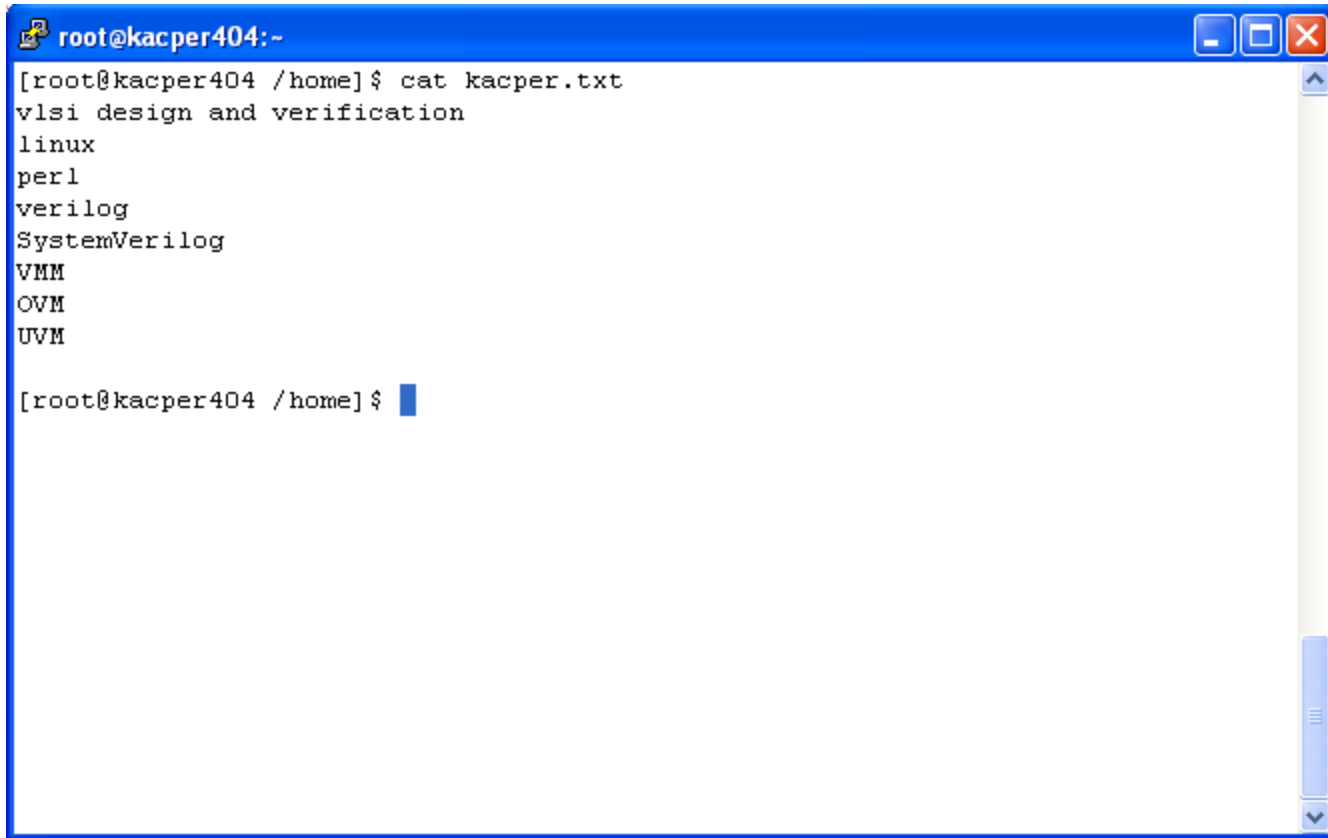
root@kacper60:/root>mkdir dir1
root@kacper60:/root>cd !$
cd dir1
root@kacper60:/root/dir1>cd -
root@kacper60:/root>mkdir dir2;cd dir2
root@kacper60:/root/dir2>
```

LINUX Commands

Command: **cat**

✓ \$ cat filename

It will display the contents of the file filename

A terminal window with a blue title bar and standard window controls. The title bar text is 'root@kacper404:~'. The terminal content shows the command '[root@kacper404 /home]\$ cat kacper.txt' followed by its output: 'vlsi design and verification', 'linux', 'perl', 'verilog', 'SystemVerilog', 'VMM', 'OVM', and 'UVM'. The prompt '[root@kacper404 /home]\$' is followed by a blue cursor block.

```
root@kacper404:~  
[root@kacper404 /home]$ cat kacper.txt  
vlsi design and verification  
linux  
perl  
verilog  
SystemVerilog  
VMM  
OVM  
UVM  
[root@kacper404 /home]$
```

LINUX Commands

Command: **cat**

✓ \$cat >file1

Success is not a destination.

[Ctrl+d]

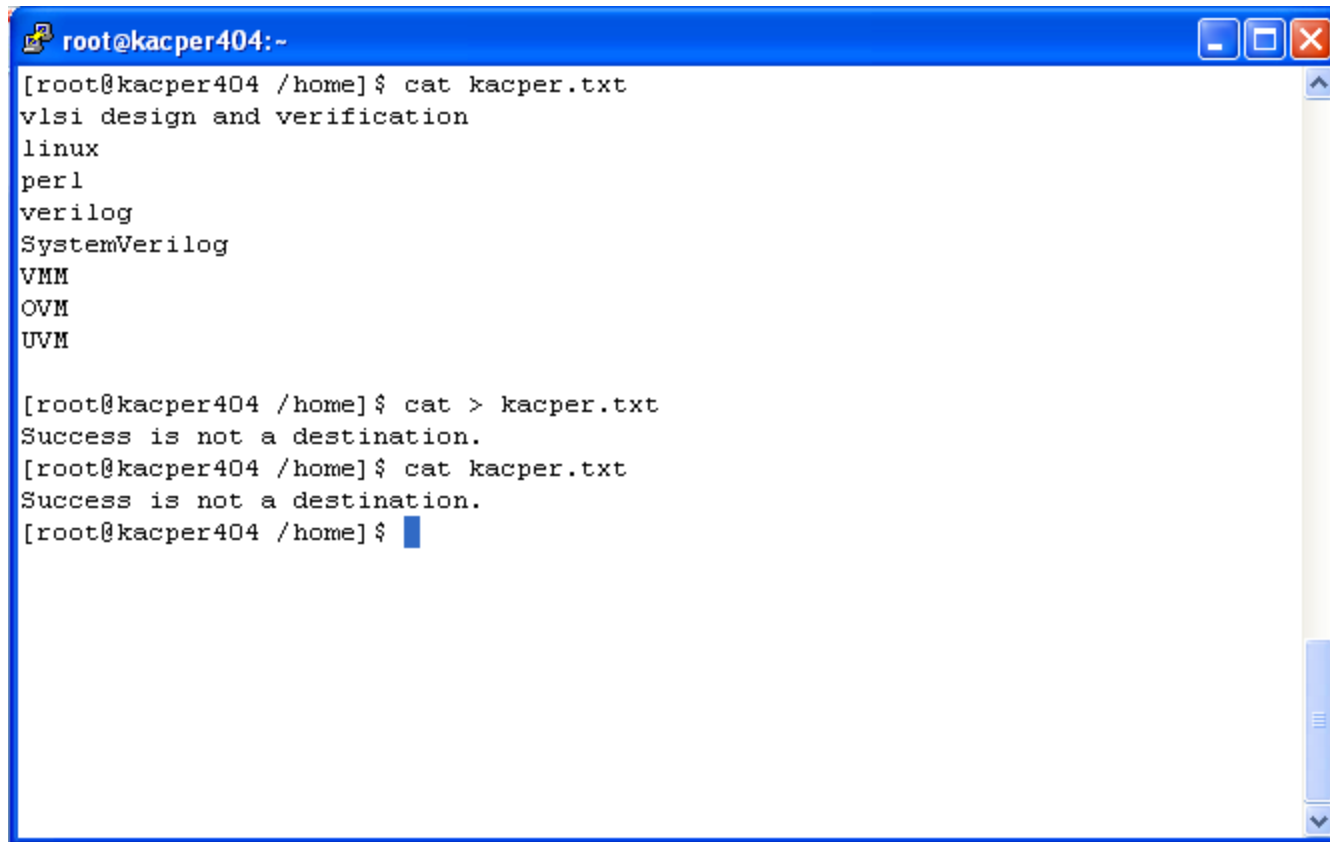
- The above command creates the file called file1 and you can enter the text there only. After finishing your work press Ctrl+d (Press Enter after the last line of your character to denote the end of the file)
- If file1 already exists then it over writes the contents of the file1
- ">" is called Redirection Operator

LINUX Commands

✓ \$cat >file1

Success is not a destination.

[Ctrl+d]

A terminal window titled 'root@kacper404:~' with standard window controls. The terminal shows the following sequence of commands and output:

```
[root@kacper404 /home]$ cat kacper.txt
vlsi design and verification
linux
perl
verilog
SystemVerilog
VMM
OVM
UVM

[root@kacper404 /home]$ cat > kacper.txt
Success is not a destination.
[root@kacper404 /home]$ cat kacper.txt
Success is not a destination.
[root@kacper404 /home]$
```

LINUX Commands

✓ `$cat >>flie1`

It's a progressive journey.

[Ctrl+d]

`$ cat flie1`

Success is not a destination. It's a progressive journey.

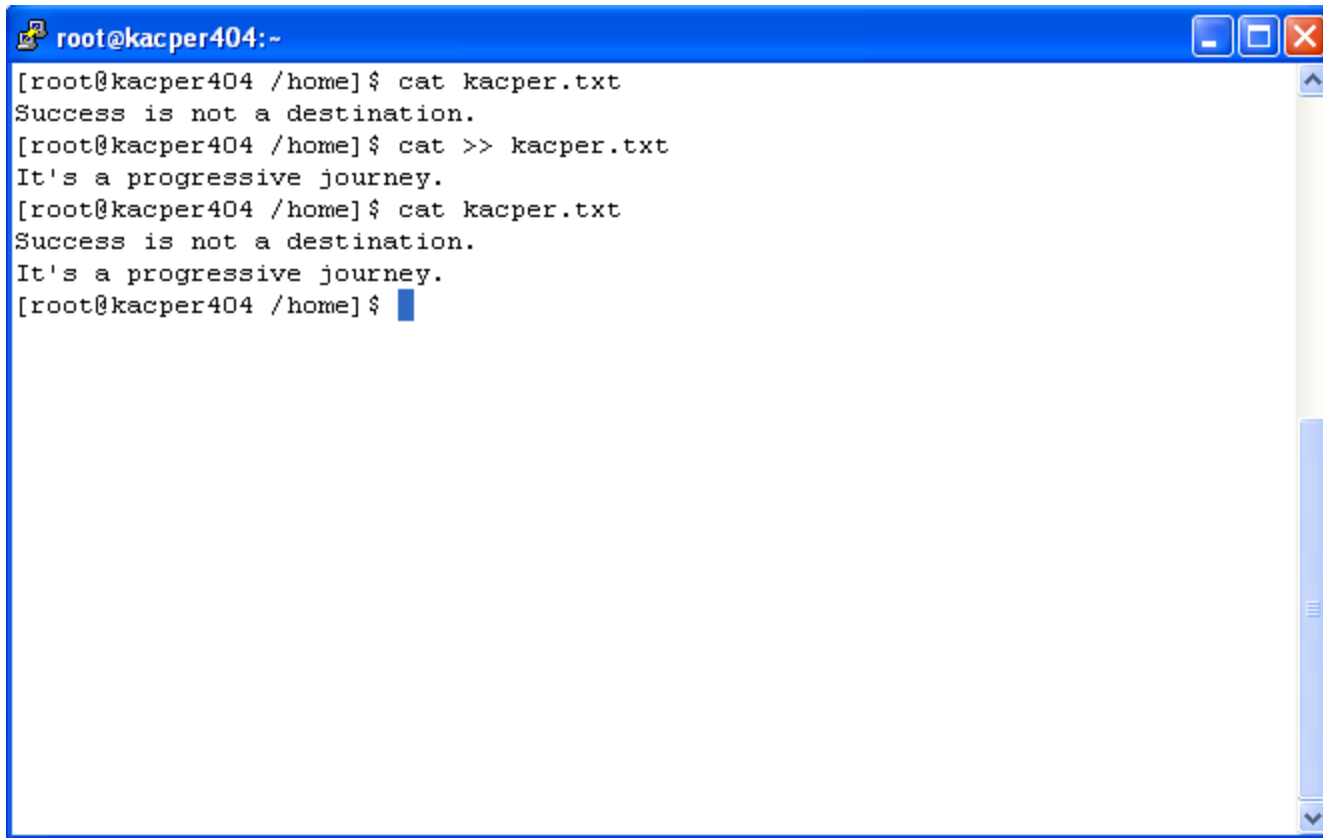
- The above command is used to append more text to already existing file.

✓ `$cat flie1 file2 >file3`

- The above command is used to write contents of the file1 and file 2 into file3

LINUX Commands

\$cat >>file1

A terminal window with a blue title bar and standard window controls. The prompt is root@kacper404:~. The user enters 'cat kacper.txt' and sees 'Success is not a destination.' and 'It's a progressive journey.'. Then they enter 'cat >> kacper.txt' and see the same output again. The cursor is at the end of the last prompt line.

```
root@kacper404:~  
[root@kacper404 /home]$ cat kacper.txt  
Success is not a destination.  
It's a progressive journey.  
[root@kacper404 /home]$ cat >> kacper.txt  
Success is not a destination.  
It's a progressive journey.  
[root@kacper404 /home]$
```

LINUX Commands

- ✓ Command: **cp**
- ✓ Syntax \$ cp [options] Source Destination
Copies Source into Destination
- ✓ \$ cp file1 file2
Copies file1 into file2
- ✓ \$ cp -prf /home/kacper/ /hdd/backup/
Copies all files, directories, and subdirectories inside kacper into backup

LINUX Commands

✓ Command: **cp**

Options:

Options	Descriptions
-i	Interactive prompts before overwriting
-f	Force if an existing destination file cannot be opened, remove it and try again
-p	Preserve preserve mode, ownership, and timestamps
-R, -r	Recursive copy directories recursively
-u	Update copy only when the SOURCE file is newer than the destination file or when the destination file is missing

LINUX Commands

Command: **mv**

- ✓ To move a file to different location use “mv”
- ✓ `$ mv [options] Source Destination`
- ✓ mv can also be used to rename a file
- ✓ `$ mv filename1 filename2 (Rename file)`
- ✓ `$mv /home/kacper/top.v /hdd/kacper/backup/`
Moves the top.v into backup directory
- ✓ `$mv -i /home/kacper/top.v /hdd/kacper/backup/`
Asks before over writing the file

LINUX Commands

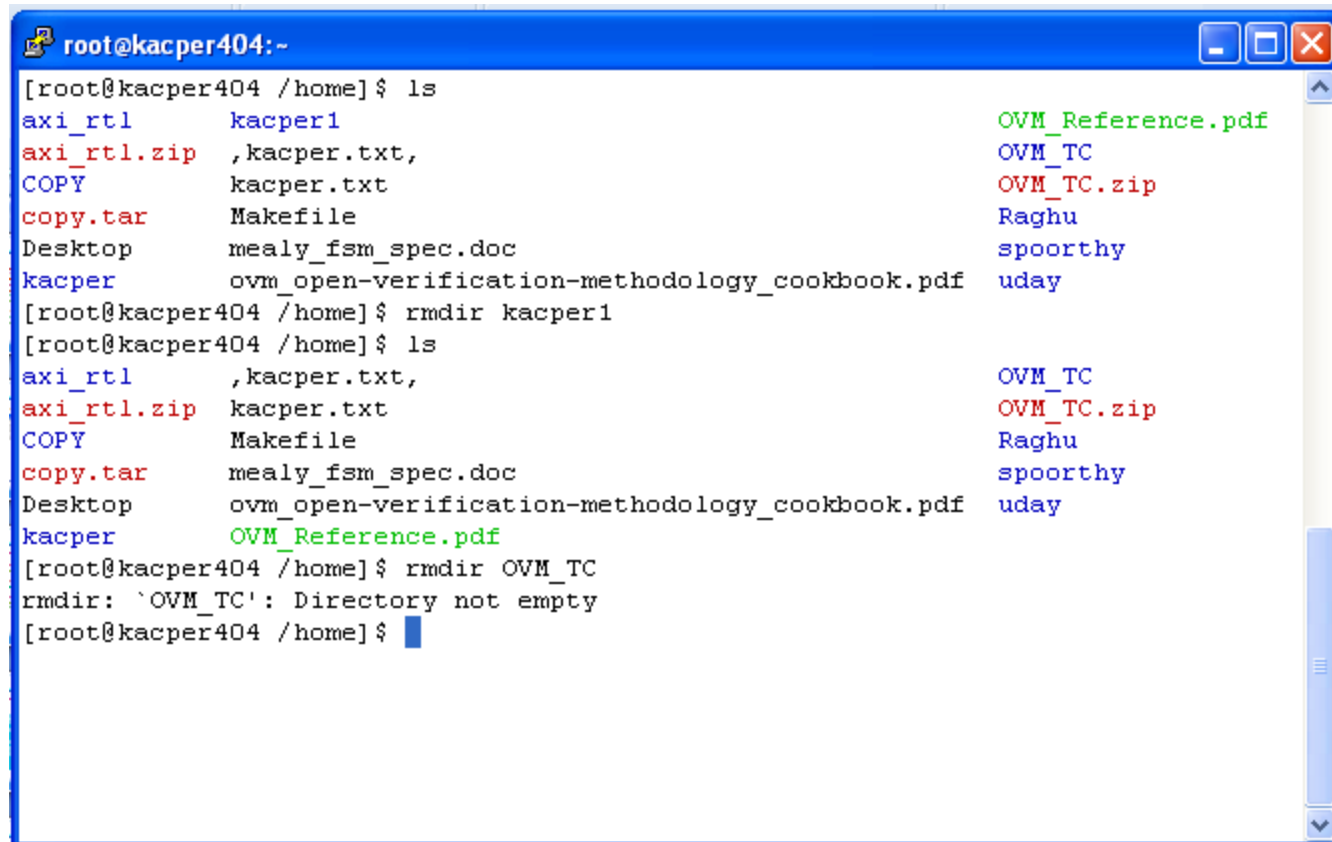
Command: **rm**

- ✓ To remove a file use “rm”
- ✓ Syntax: \$ rm filename
- ✓ `rm -i *` prompts you before deleting a file. The “i” stands for interactive
- ✓ `rm -rf *` recursively removes all files and subdirectories in your current directory, without prompting to delete files
- ✓ Be very careful, deletions are permanent in UNIX/LINUX

LINUX Commands

Command: **rmdir**

- ✓ To remove a empty directory use “rmdir”
- ✓ Syntax :\$ rmdir directoryname

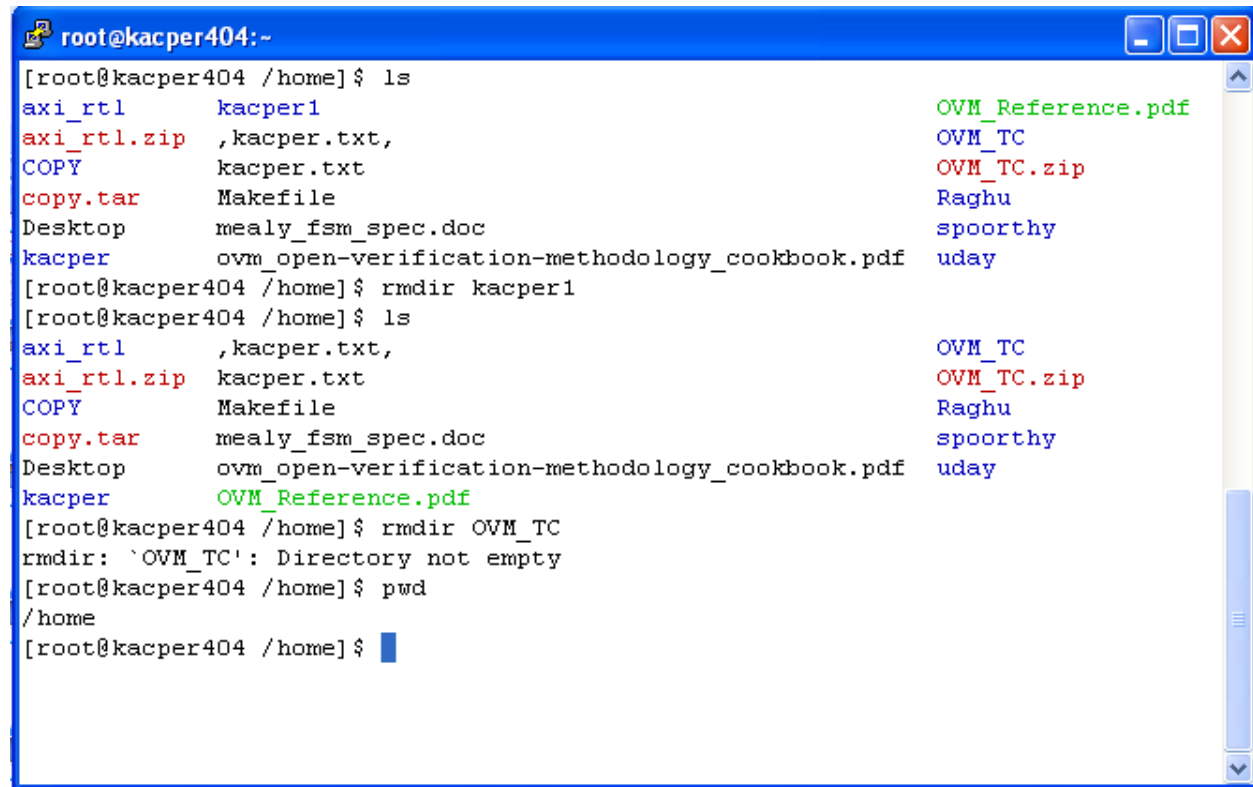


```
root@kacper404:~  
[root@kacper404 /home] $ ls  
axi_rtl      kacper1      OVM_Reference.pdf  
axi_rtl.zip  ,kacper.txt, OVM_TC  
COPY        kacper.txt   OVM_TC.zip  
copy.tar    Makefile     Raghu  
Desktop     mealy_fsm_spec.doc spoorthy  
kacper      ovm_open-verification-methodology_cookbook.pdf uday  
[root@kacper404 /home] $ rmdir kacper1  
[root@kacper404 /home] $ ls  
axi_rtl      ,kacper.txt, OVM_TC  
axi_rtl.zip  kacper.txt   OVM_TC.zip  
COPY        Makefile     Raghu  
copy.tar    mealy_fsm_spec.doc spoorthy  
Desktop     ovm_open-verification-methodology_cookbook.pdf uday  
kacper      OVM_Reference.pdf  
[root@kacper404 /home] $ rmdir OVM_TC  
rmdir: `OVM_TC': Directory not empty  
[root@kacper404 /home] $
```


LINUX Commands

Command: pwd

To find your current path use “pwd”



```
root@kacper404:~  
[root@kacper404 /home]$ ls  
axi_rtl      kacper1      OVM_Reference.pdf  
axi_rtl.zip  ,kacper.txt, OVM_TC  
COPY        kacper.txt   OVM_TC.zip  
copy.tar    Makefile     Raghu  
Desktop     mealy_fsm_spec.doc spoorthy  
kacper      ovm_open-verification-methodology_cookbook.pdf uday  
[root@kacper404 /home]$ rmdir kacper1  
[root@kacper404 /home]$ ls  
axi_rtl      ,kacper.txt, OVM_TC  
axi_rtl.zip  kacper.txt   OVM_TC.zip  
COPY        Makefile     Raghu  
copy.tar    mealy_fsm_spec.doc spoorthy  
Desktop     ovm_open-verification-methodology_cookbook.pdf uday  
kacper      OVM_Reference.pdf  
[root@kacper404 /home]$ rmdir OVM_TC  
rmdir: `OVM_TC': Directory not empty  
[root@kacper404 /home]$ pwd  
/home  
[root@kacper404 /home]$
```

LINUX Commands

Display Commands:

Command: **less**

- “less” displays a file, allowing forward/backward movement within it
- Use “/” to search for a string in the file
- Press “q” to quit
- Syntax: \$ less [options]filename

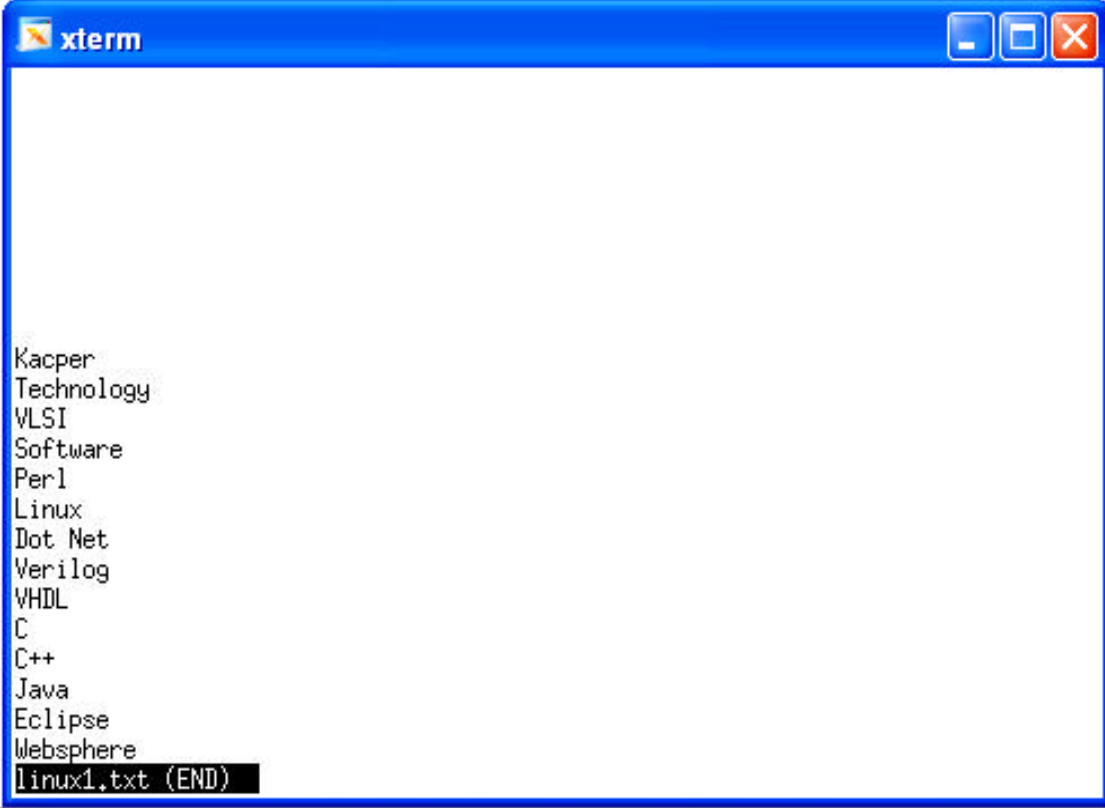
Options

- -c clears the screen before displaying.
- +n starts printing from nth line

LINUX Commands

Display Commands:

Command: **less**

A screenshot of an xterm window with a blue title bar and standard window controls. The terminal displays a list of topics: Kacper, Technology, VLSI, Software, Perl, Linux, Dot Net, Verilog, VHDL, C, C++, Java, Eclipse, Websphere, and linux1.txt (END). The last line is highlighted with a black background.

```
xterm  
  
Kacper  
Technology  
VLSI  
Software  
Perl  
Linux  
Dot Net  
Verilog  
VHDL  
C  
C++  
Java  
Eclipse  
Websphere  
linux1.txt (END)
```

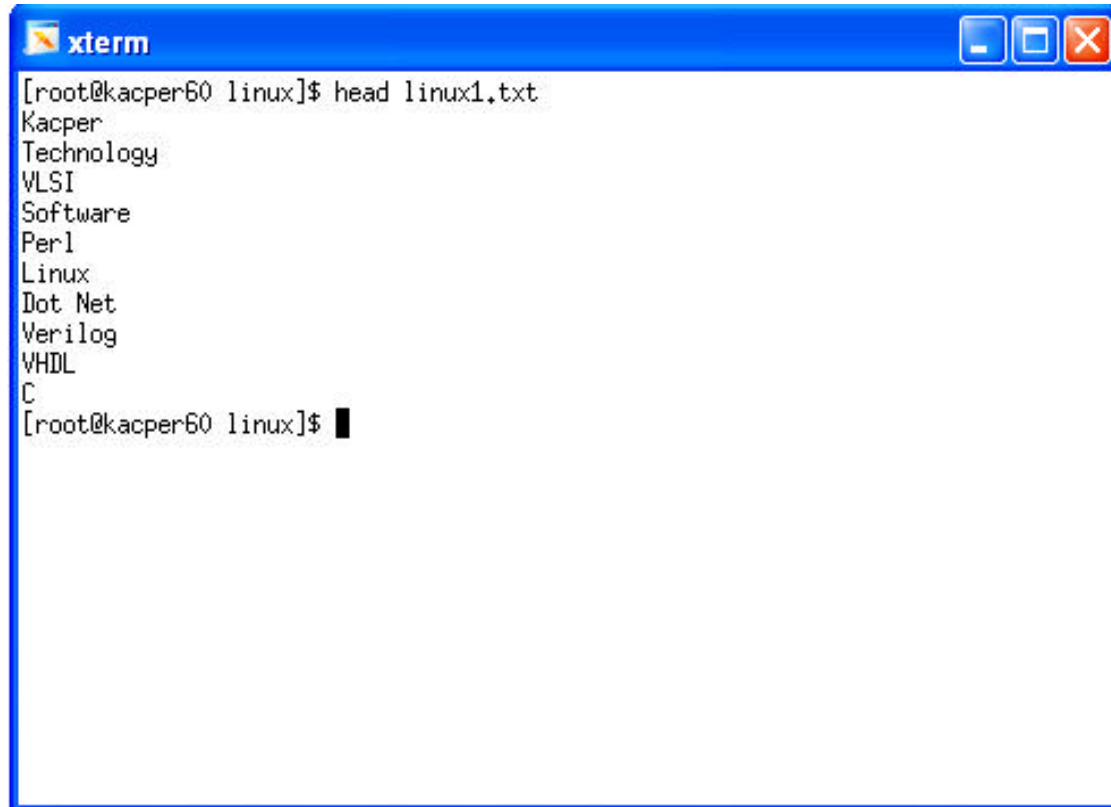
LINUX Commands

Command: **head**

- ✓ “head” displays the top part of a file
- ✓ By default it shows the first 10 lines
- ✓ -n allows you to change the number of lines to be shown
- ✓ Syntax: `$ head [options]filename`
- ✓ Example:
 - “head -n50 file.txt” displays the first 50 lines of the file.txt
- ✓ `$head -18 filename`
 - Displays the first 18 lines of the file called filename

LINUX Commands

Command: **head**

A screenshot of an xterm window with a blue title bar and standard window controls. The terminal shows the command 'head linux1.txt' being executed, which outputs the first ten lines of the file 'linux1.txt'. The output lists various technologies: Kacper, Technology, VLSI, Software, Perl, Linux, Dot Net, Verilog, VHDL, and C. The prompt returns to the shell after the command is executed.

```
xterm
[root@kacper60 linux]$ head linux1.txt
Kacper
Technology
VLSI
Software
Perl
Linux
Dot Net
Verilog
VHDL
C
[root@kacper60 linux]$
```

LINUX Commands

Command: **tail**

- ✓ Displays last 10(by default) lines of a file
- ✓ Same as head command
- ✓ Syntax:\$ tail filename
Displays the last 10 lines from the ending
- ✓ \$tail -12 filename
Displays the last 12 lines from the ending

LINUX Commands

Command: **tail**

A screenshot of an xterm window with a blue title bar and standard window controls. The terminal shows the command 'tail linux1.txt' being executed, which outputs the last lines of the file: Perl, Linux, Dot Net, Verilog, VHDL, C, C++, Java, Eclipse, and Websphere. The prompt returns to the user.

```
xterm
[root@kacper60 linux]$ tail linux1.txt
Perl
Linux
Dot Net
Verilog
VHDL
C
C++
Java
Eclipse
Websphere
[root@kacper60 linux]$
```

LINUX Commands

Command: **more**

✓ Read files and displays the text one screen at a time

✓ Syntax:

\$ more [options] filename

✓ Options:

-c clears the screen before displaying.

-n displays the first n lines of the file. We can also
 see next lines by pressing [Enter]

+n displays the lines from nth line.

LINUX Commands

Command: **more**

```
xterm
[root@kacper60 linux]$ ls -la / | more
total 95424
drwxr-xr-x 31 root root 4096 Jun  4 15:38 ./
drwxr-xr-x 31 root root 4096 Jun  4 15:38 ../
-rw-r--r--  1 root root    0 Jun  4 15:30 .autofsck
drwxr-xr-x  2 root root 4096 Oct 13  2004 .automount/
drwxr-xr-x  2 root root 4096 May 22 04:04 bin/
drwxr-xr-x  3 root root 4096 Sep 14  2007 boot/
-rw-r--r--  1 root root   813 Nov 20  2008 .cshrc
dr-xr-xr-x  3 root root 4096 Feb 22 15:59 depot/
drwxr-xr-x  8 root root 5680 Jun  4 15:30 dev/
-rw-r--r--  1 root root  1247 Jul 31  2009 Div3chk
drwxr-xr-x 106 root root 12288 Jun  6 16:08 etc/
-rw-r--r--  1 root root 97280000 Sep 30  2009 file.out
-rw-r--r--  1 root root 27750 Sep 14  2007 .fonts.cache-1
drwxr-xr-x 19 root root 4096 Sep 23  2009 home/
drwxr-xr-x  2 root root 4096 Aug 12  2004 initrd/
drwxr-xr-x 11 root root 4096 Jul  4  2009 lib/
drwxr-xr-x  6 root root 4096 Aug  7  2009 LOCAL_SCRATCH/
drwx----- 2 root root 16384 Sep 14  2007 lost+found/
drwxrwxrwx  5 root root 4096 Jun  4 15:30 media/
drwxr-xr-x  2 root root 4096 Jul  2  2009 mnt/
drwxr-xr-x  5 root root 4096 Jul 31  2009 opt/
--More--
```