

## SEER AKADEMI Linux Basics-Lecture 4

Once again welcome to this class for on Linux basics this is the ongoing classes this is the class on Linux and all the programming languages today is our 4<sup>th</sup> session we will be continuing our discussion on the Linux ,we started talking about the basics as for the kernel, the file system and then the shell ,then we went into more details about the command and then last week . We saw behind last lecture, we saw the commands, how to do some, managing the file access and system resources.

(Refer Slide Time: 00:49)

### Lecture 3 - Recap

- Manage file access
  - chmod
  - chown
  - chgrp
- Manage System resources
  - ps
  - top
  - kill
- Managing resources
  - Du
  - Df
  - Free

And managing general disk usage or disk resources to be specific we saw the commands like change mode to modify the file functions ,we had the different sections in the file permission lists ,the first one is for the user ,and the second one is for the group ,and the third one is for everyone else , we saw how to set these access codes or actually like a binary number. So it is basically read ,write and execute ,so for each of these things ,so maximum that you can allocate a number is 7 which is read ,write ,execute each one is represented by 1 bit ,so if all of them are on that is 1, 1,1 which stands for digital number 7,so you can say like 7,7,7 that means wanted for bigger table for all the books and then based on how, where you made the because 0 ,and you can control the access.

For example you want ,so you want all the read, write, execute and yourself 7 , and then the group just wants to read and execute, then you make it as 1 0 1 which is 5 and then for all of them, we just want them to read then you make it as 1 0 0 which goes to 4 ,so like that you can control ,so 7 or any combination of both and then use to control boxes.

Then you also we saw the how to change the ownership of given file ,from yourself some of the owner, some of the user and then and also you the change group to change the group functions of the group, itself of people but this comes in handy because look at like various real-life situations, a particular file access may be restricted because of subgroups or particular directory and the access only by certain group .

And you may have permission, but you do not know like in which group, you belong to and you may have like permissions of multiple groups, we want to make sure that particular file corresponds at least that one group ,which will enable you to do more stuff we also saw like how to manage the system resources, in particular we saw some of the definition for processes ,how do we what we call processes in Linux .

And how Linux assigns process time for each of the processes and then we also saw how to report those processes, using the ps command or the top commands and then we also saw how to kill a particular process by using the kill and in which we also saw like the various priorities, there were the interrupts can be as high as - 9 which means that instant kill of the process.

We also saw some commands related to the resources, the disk usage in particular we saw like Du command , the Du various options and which actually enabled us to get how much disk space is remaining and then we also like we saw the DF command which is essentially again another way of estimating the usage here ,the DF reports on particular file system as to mount point and how much memory and then we also saw the command free to also understand how much memory is needed.

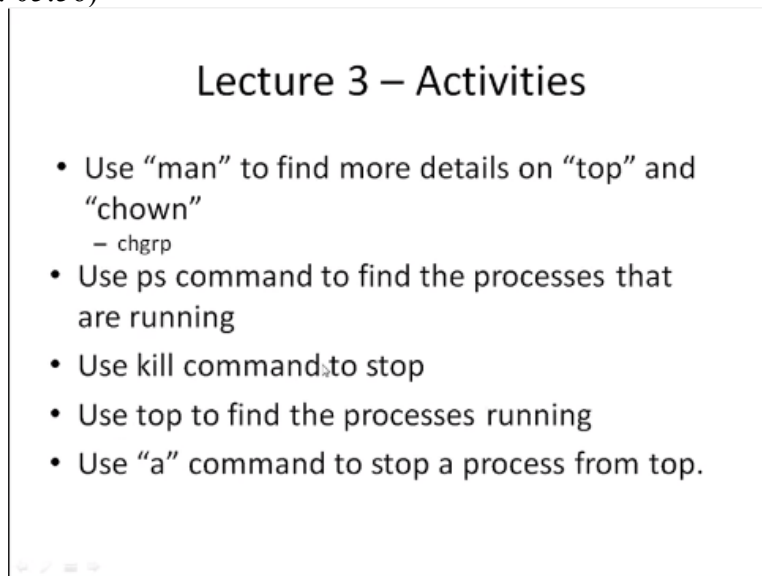
We also use some other commands to kind of map analogy ,they want to find out like which, who is the user ,so there are who command gives you all the users in the system , or who am I specific to which user name ,you are using and accessing system . So today we will be looking at some more commands.

(Refer Slide Time: 05:54)

## Lecture 3 - Recap

- Use “man” to find more details on “top” and “chown”
  - chgrp
- Use ps command to find the processes that are running
- Use kill command to stop
- Use top to find the processes running
- Use “a” command to stop a process from top.

(Refer Slide Time: 05:56)



Lecture 3 – Activities

- Use “man” to find more details on “top” and “chown”
  - chgrp
- Use ps command to find the processes that are running
- Use kill command to stop
- Use top to find the processes running
- Use “a” command to stop a process from top.

And before we start about today's lecture ,I want to assign you some more activities, so just like what we did in the last lecture today I want you to use man command to find more details on top is where the commands that we study in last lecture to get more details on top and then chown and find out and see like more of options with more detail especially the top will vary widely is used to read in the dynamic way as to what is happening with all the processes in the system. And so particular interest is how do you kill with the top and also how do you exit from top, those kind of things, become useful then use the PS command ,to find all the processes that are running and use the kill command to stop, also use top to find the processes ,which is running and then use the a command ,use at least one command to stop the process from top document itself. So these are just fun activities and so in today's topic.

(Refer Slide Time: 07:19)

## Lecture 4 - Topics

- File system commands
  - Grep, fgrep,
  - Locate
  - Uniq, sort
  - touch
  - diff, tee

We will be dealing with more in depth file system commands specifically we talk about grep, fgrep, locate, unique ,sort, touch ,find another command like diff and tee I have kind of group them here we will see why and also like some of the commands that are also like other commonly used for flow link , how to link a file, the linking is a very easy way of actually accessing a file ,without actually copying the file and in turn a long time as well.

(Refer Slide Time: 08:19)

### Commands in File System:

- File system commands
  - Grep, fgrep,
  - Locate
  - Uniq, sort
  - touch
  - diff, tee

So let us start with these commands .

(Refer Slide Time: 08:24)

## Commands in File System:

Command: `fgrep`

- ✓ `fgrep` is used to search for exact strings in text files
- ✓ The `fgrep` contains various options
- ✓ They are
  - i for ignore case
  - v for displaying the lines that don't match
  - n for displaying the line number with the line where the match was found
- ✓ Syntax: `$ fgrep [options] textfiles`

The first command that we will talk about is for `fgrep` and the `fgrep` command is mainly used to search a keyword inside a file, so the keyword is specified as a string and the `fgrep` provides a lot of options for essentially, for doing various things, so here are some options are given which is - I for ignore case, - v for the lines which do not match, so basically it excludes the lines that match and it gives you only the lines, that do not match from line and then also you can say like the - n and the following specify some number .

So that the command actually displays the lines ,then it found the match and then after that n number of lines will specify here, in that time ,so these are like very powerful and useful commands ,so typically we try to get and use the `fgrep` to find a particular keyword that is there in the file and then once we find it then we can do all kinds of operation forms, keywords and then we can use it in other programs.

Even the string that we specify for matching that you can specify it as a regular expression and you can actually go into the man `grep` or `fgrep` to get more information about this form of itself there are also like other nifty features like - L which only shows all the files, there the keyword is present it does not output the exactly the line itself ,so these are all like some things you can do too gather information.

So again these are like the information getting amongst themselves they do not modify the file or one of the features that we talked about earlier was with multitasking, so even within a window you can do multitasking which is essentially ,we saw this pipe command you can type a command output of a command, into another command, so similarly you can use web command in that fashion.

For example if you want to search multiple keywords and you want to see like I mean which file all the particular keywords are present, so you can actually grab one keyword at a time , and then type those outputs into the subsequent problems ,so it will be like grep a from star.txt type grep B to type grep E , so now the output of that this entire command will be one file which contains all .

So it is again a useful phenomenon then you can you can put it in multiple orders and it can give you some exercise electron .  
(Refer Slide Time: 11:50)

**Commands in File System:**

Command: fgrep

- ✓ fgrep is used to search for exact strings in text files
- ✓ By using the find command we can find the files by date and also we can specify the range of times
- ✓ Example: `$ find /user/bin -type f -atime +100 -print`
- ✓ We can also use find to show the postscript files in our directory
- ✓ Syntax: `$ fgrep [options] textfiles`

Page 1 of 2

Copyright: 2010, Kacper Technologies Pvt.Ltd. All Rights Reserved

The other command that we will talk about is find the find command is again used to find particular file in a hard drive so this is like mostly like the file name rather than using really a grep which is the more evil ,so using this one you can find files by date and you can specify also the range of times, then modify .

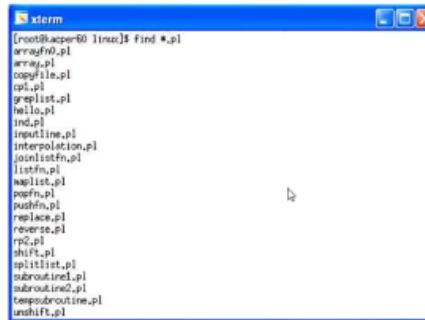
So here this is an example find \$ user, \$ bin type is F and then - a time +100 - print then you can actually like get this particular files so using find command you can also use find to show the postscript files in our directory then one example is like find \*.pearl here you will see all the commands that end in pearl, so again the syntax follows the general syntax of the Linux command ,only thing is here the directory is also an argument , which is essentially the find and argument then the options and then followed by what is action .

For example here the action is to print ,but all the time support for both the perform in the example come on.

(Refer Slide Time: 13:28)

## Commands in File System:

Command: find



```
xterm
[root@kasper60 linux]# find *.pl
array.pl
copyfile.pl
grep.pl
hello.pl
ind.pl
inoutline.pl
interpolation.pl
joinlistfn.pl
listfn.pl
mlist.pl
popfn.pl
pushfn.pl
replace.pl
reverse.pl
rpc.pl
shift.pl
splittest.pl
subroutine1.pl
subroutine2.pl
testsubroutine.pl
unshift.pl
```

So here there is more stuff basically because you just do not specify anything it assumes that it is actually the current entry, so find start perl will give you ,list all the PL extended files, this is same as you can say think of right now as LS \*.pl that also gives you ,so the find is also used when particularly directories or particular file is embedded in multiple that is apparently and somewhere below the hierarchy where this file is then. You can use find to actually let them find the file.

(Refer Slide Time: 14:13)

## Commands in File System:

Command: locate

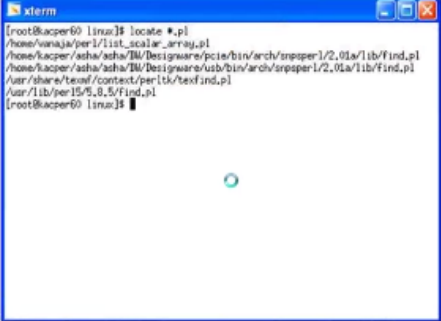
- ✓ The locate command is much faster than the find command
- ✓ Finding a file using locate is faster when compared to the find command
- ✓ The files are printed with the path if we use this command
- ✓ Syntax:\$ locate [search string]

The other command is locate, locate is much faster than the find command that is the reason why you chose to located file ,which is online, so again the files are printed with the fgrep , you use the command, so the syntax is just locate and then the search string, so it finds the file and then you print the whole path, where exactly file is located for that string.

(Refer Slide Time:14:46)

## Commands in File System:

Command: locate



```
xterm
[root@kacper60 linux]# locate *.pl
/home/vanaja/perl/11st_scalar_array.pl
/home/kacper/asha/asha/BI/Designware/pcie/bin/arch/snoper1/2_01a/11b/Find.pl
/home/kacper/asha/asha/BI/Designware/usb/bin/arch/snoper1/2_01a/11b/Find.pl
/usr/share/teem/context/perl/teemfind.pl
/usr/lib/perl5/5.8.5/Find.pl
[root@kacper60 linux]#
```

So here is an example again locate \*pl it gives you like the entire path and then followed by where it is and then it goes under all the directories, so from the current directory goes under each hierarchy and then finds if that particular file exists, in those hierarchies and includes all the entire path actually like I just wanted to even the previous one, that is exactly the same thing. So even though it does not print out the path name, this set of files is not just limited to current directly to actually perform current directly onwards it goes all the way under and then find subtypes ,so that becomes the principle of LS command on the find, whereas LS only restricts to the particular hierarchy that gives, for example LS \*pl it only finds all the Perl files in current directory, if you do LS ./Perl it goes under one directly below and then.

Let us find such as across all the entire tree that is under that particular directory and gives you all the matters , so this is a principle okay .

(Refer Slide Time:16:17)



## Commands for showing file details:

Command: wc

- ✓ **\$wc[options] filename**  
Gives the number of lines, words and characters in a file called filename
- ✓ **\$wc -l filename**  
Gives the number of lines
- ✓ **\$wc -w filename**  
Gives the number of words
- ✓ **\$wc -c filename**  
Gives the number of characters

And then the other command that we will learn is wc, it is not water closet, so it is actually stands for word count, so here the key thing is you can actually determine how many lines are there in a file using this WC, even though it is the word, actually you can give you the number of line, words or characters all of them.

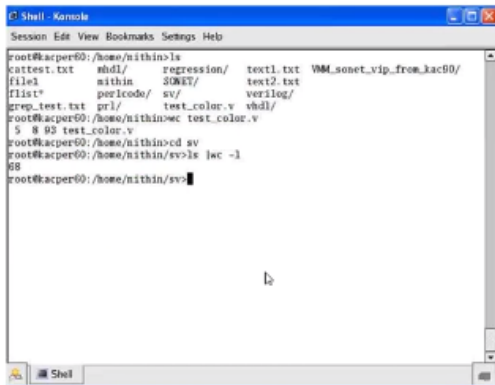
So again there are multiple options - L gives you the lines and if you just specify WC without any options and then the file name it gives you all three it, gives you the number of lines, at the first column, number of words at the second column and then you can specifically go there going over more WC - L filename that gives you the number of lines W 3 -W the file name.

The number words again the words, means the non whitespace characters, group of characters separated by white space capital or a new line, so those are the words that is the definition of work and so that collection will be treated as a word and then the WC - 4 will give you the number of characters, character is typically non whitespace characters, so all the way to dispensable and not above it.

(Refer Slide Time: 17:57)

## Commands for showing file details:

Command: wc



```
root@kacper00: /home/nithin/ls
cat1.txt      nhd1/      regression/  text1.txt   VM_sonet_vip_from_kac00/
file1        nithin     30W67/      text2.txt
!list*       pericoda/  sv/         verilog/
grep_test.txt gr1/       test_color.v vhd1/
root@kacper00: /home/nithin# wc test_color.v
 5  8 93 test_color.v
root@kacper00: /home/nithin# cd sv
root@kacper00: /home/nithin/sv# ls | wc -l
68
root@kacper00: /home/nithin/sv#
```

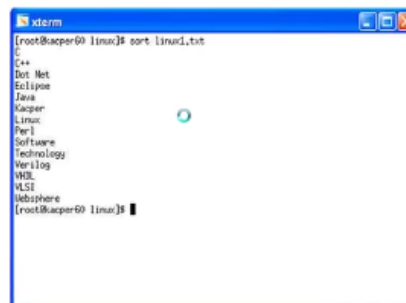
Here are some examples so here is a default 1 w test color B you see the 5 lines 8 characters 8 words 93 chapters and then here you can also see like somebody is typing the LS command into wc , so then it gives you the since it is LS basically like it gives you how many lines are there in this output of the LS command again as I said like we will use the pipe the commands output gets piped.

So in this one you can think that basically there are 68 files in that in this directory you know like the regular LS actually gives the files, multiple files in the same line when you type the LS to the WC moment L it gives each file as 1 line., so that is why I like a maximum of 5 to 16 . (Refer Slide Time: 19:17)

## Commands for showing file details:

Command: sort

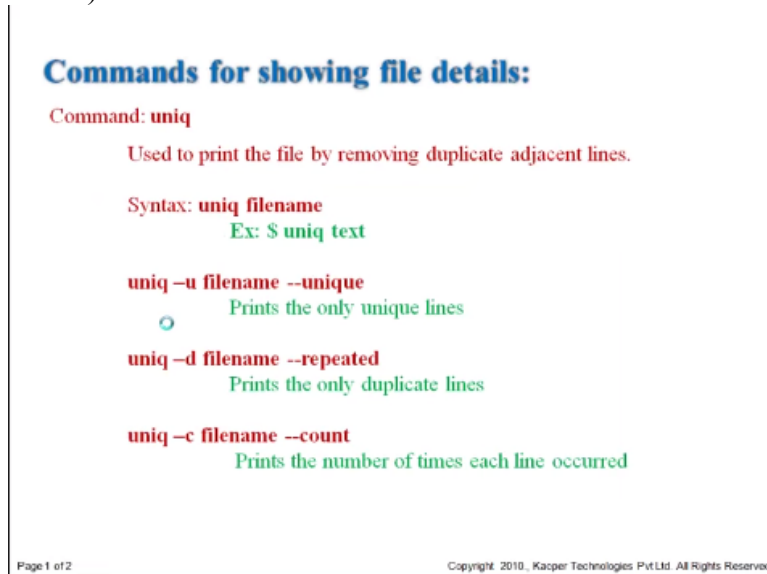
- ✓ It prints the lines of the file in sorted order.
- ✓ Syntax: \$ sort filename



```
[root@kacper00 linux]# sort linux1.txt
C++
Dot Net
Eclipse
Java
Kacper
Linux
Perl
Software
Technology
Verilog
VHD
VLSI
WebSphere
[root@kacper00 linux]#
```

So the next command is sort, the command is used to sort the files in the alphabetical order, sort file name, sort and then the file name this takes all the lines and basically sees which line to be

come in first and then it presents that way, for example in this case Linux, C,C ++ and Eclipse ,Java so you can see that actually and if this goes so inside the content may be in any other order but when you do this it sorts immediately into a book model .  
So again this is another useful command if you are collecting some data and then the data is all some form and you want to present it in a much better form ,so you can use this command to present it in an alphabetical order , but now let us move on this command unique .  
(Refer Slide Time: 20:38)



**Commands for showing file details:**

**Command: `uniq`**  
Used to print the file by removing duplicate adjacent lines.

**Syntax: `uniq filename`**  
Ex: `$ uniq text`

**`uniq -u filename --unique`**  
Prints the only unique lines

**`uniq -d filename --repeated`**  
Prints the only duplicate lines

**`uniq -c filename --count`**  
Prints the number of times each line occurred

Page 1 of 2 Copyright © 2010, Kasper Technologies Pvt.Ltd. All Rights Reserved

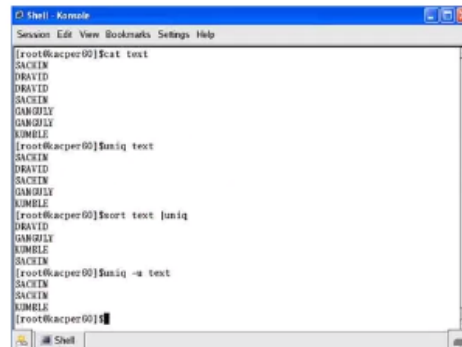
This command prints out the file, removing all the duplicate adjacent lines , so again a unique file name , it gives you all the lines that are appearing only once and then there are options to it so for example unique - u file name - prints all the unique lengths and then if you say like repeated with - d it only prints out the what are the duplicated so again this command is used many ways for very useful .

If you have a particular file name which you copied from multiple sources and we don't know whether something is repeated in time one way to find out is just use and you can also say like - c which gives you the number of form times a particular line receptor ,so for the unique lines it prints out 1 and then how many more times from that particular line has been occurring inside that line. So let us look at some examples.

(Refer Slide Time: 22:12)

## Commands for showing file details:

Command: **uniq**



```
Shell - Konsole
Session Edit View Bookmarks Settings Help
[root@kacper00]#cat text
SACHIN
DRAVID
DRAVID
SACHIN
GANGULY
KUMBLE
[root@kacper00]#uniq text
SACHIN
DRAVID
SACHIN
GANGULY
KUMBLE
[root@kacper00]#sort text |uniq
DRAVID
GANGULY
KUMBLE
SACHIN
[root@kacper00]#uniq -u text
SACHIN
SACHIN
KUMBLE
[root@kacper00]#
```

Here create the test we create this one in feedback Sachin ,Sachin is actually repeated but become patient and Ganguly is repeated twice only so if you just do like this delete text now it gives you Sachin twice but Dravid and Ganguly and generally so one way to eliminate this kind of situation is to use the sort ,so when you do a sort the Sachin and Sachin to position actually like they get together .

So in this case it will be Dravid ,Kumble or Dravid , Ganguly and then Kumble,Sachin so now if you say like sort ,text and then pass that to the unique command then you get really disagree because go to such insult coming now together and that gets eliminated and then you can say like unique the text then it gives you like Sachin, Sachin and Kumble so we will illustrate how the unique command is getting used.

Actually this never fall the unique - view is actually only friends of the unique lines in this case like if you have one they are sorted this way then we will print out the Oh actually like because the text is still did this one so that is why such an expensive twice because it thinks still they are unique as they appear in different mind the entire Droved is born and the gondola is gone because they are repeated so that is why the unique - you filing is to set in and one so now let us go to the next one which is touch.

(Refer Slide Time: 24:48)

## Commands for showing file details:

### Command: touch

- ✓ Used to create empty file
- ✓ Syntax `touch file2`  
It will create the file called file2 of size zero bytes, if file2 doesn't exist
- ✓ Used to change the time stamps.(i.e. dates and times of the recent modification or access)

↳

That is a way to create empty file so when you say like touch and then the file name an empty file with that finding it will be created you have a size of 0 bytes it does not exist if that particular file exists then it updates the timestamp so that it appears as if like that file was updated just recently ok so again this is useful in the comp or some particular commands that may work on will not run if the file is not updated to the lattes. And time steps or you may have a requirement that this particular file needs to be updated much later than another portable form so and but the fat file may be created later than the file. That you want to access in that sense you can just do a patch and then the file name which updates the timestamp and then you can do as here as an example.

(Refer Slide Time: 26:06)

## Commands for showing file details:

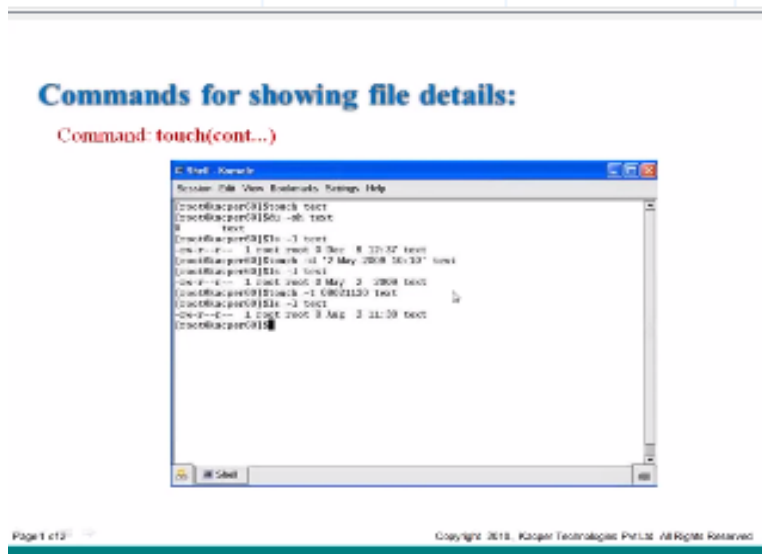
### Command: touch(cont...)

- ✓ For example to change the last access time of file6 to 10:10 a.m. May 2, 2009, it can be done in the following way
- ✓ `$ touch -d '2 May 2009 10:10' file6`  
It will change the time stamp of the file6.It can be verified by the following way
- ✓ `$ ls -l file6`  
`-rw-r--r-- 1 root root 120 May 2 2009 file6`
- ✓ `$ touch -t 08021130 file7`  
It can change the date and time. The expression 08021130 denotes the Month, day, hour and minute. In general the expression type is MMDDhhmm

We want to change the access time by six 10:10 in vain so here we can actually give a particular date and then we can say like touch so touch - be second May and then followed by the time and then it will change the timestamp to that particular thing so you can use an LS - L to verify whether the timestamp change as so another way to do it is - team touch - team the touch - T takes an expression which is one or by day followed by hour followed by move so it is shown here one month day our means once you specify this form corner it actually updates the timestamp to that and that particular time.

So that that is another way to change the 1-3 before the dating second quotation or disk specificity and then strain and then it is a big student one typically we do not use within the options in the stay attached and then the file name then it puts that the latest timestamp into that one so it just gets the timestamp from the date command then puts that particular date and all that but file so that it appears as if this work model and then if it is not there then that it produces the file with real hope so here are again some more examples.

(Refer Slide Time: 28:01)



So they may say like patch and then we see the what is a disk usage for that particular with reports zero bytes and if you do an LS - L still reported zero by bassist Einstein and then we say okay now we want to change the timestamp and then provide this time stamp and then say like this one for text you see and older things so again you can advance the date or move it forward you cannot move it forward more than the system paper so that kind of thing make sure so here it is oscillating - the option to change the date on the 13th time in film and sting so as I mentioned earlier one there are few commands.

(Refer Slide Time: 29:00)

## Commands for showing file details:

Command: tee

✓ Sends the output in two directions at a time

✓ Syntax: tee [options ] filename

✓ Ex : \$ls -l | tee file2

It will list all the files in the current directory and as well stores in the file called file2

\$ls -l | tee -a file2

It will append the list to file2

To actually alter the programming behavior or managing the system or fulfill one thing that we saw was the pipe command then we also saw the public processes can be all in the other command is what is called T the T command actually sends an output of a program or in our case it is a process opera process to two different directions at the same time so you can specify T and then I will mean options and then you can actually send it to two different locations so what are the two locations here one is it stores into the particulars file so here we have a know example LS - L by T filing / 2.

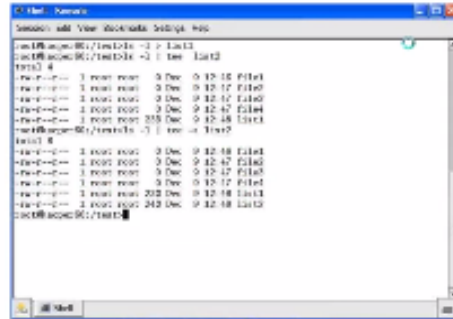
So now let us say like I mean how this works so LS - L can return output which is the list of all the five of the details inside the directory and then that is actually pipes to T and then I will - so when it is five tackle eight is actually going to the output file in our case it is the terminal then displays what is the output of that element custom and but we say that we also do at P to file - that means that it actually writes out.

The same information into PI to end zone and the - demons a is an option to append to a particular file same as for the double arrow when we say double arrow it only liked us just that portion of the test writing the files biting into PI R as the T will write into the file and also output into the kernel let us see an example.

(Refer Slide Time: 31:11)

## Commands for showing file details:

Command: tee



```
Windows [cmd] Powershell
C:\Users\user> ls -l | tee list1.txt
-rw-rw-r-- 1 user user 0 Dec  9 12:16 file1
-rw-rw-r-- 1 user user 0 Dec  9 12:17 file2
-rw-rw-r-- 1 user user 0 Dec  9 12:17 file3
-rw-rw-r-- 1 user user 255 Dec  9 12:18 list1
-rw-rw-r-- 1 user user 0 Dec  9 12:18 list2
-rw-rw-r-- 1 user user 0 Dec  9 12:18 list3
-rw-rw-r-- 1 user user 242 Dec  9 12:18 list4
-rw-rw-r-- 1 user user 0 Dec  9 12:18 list5
```

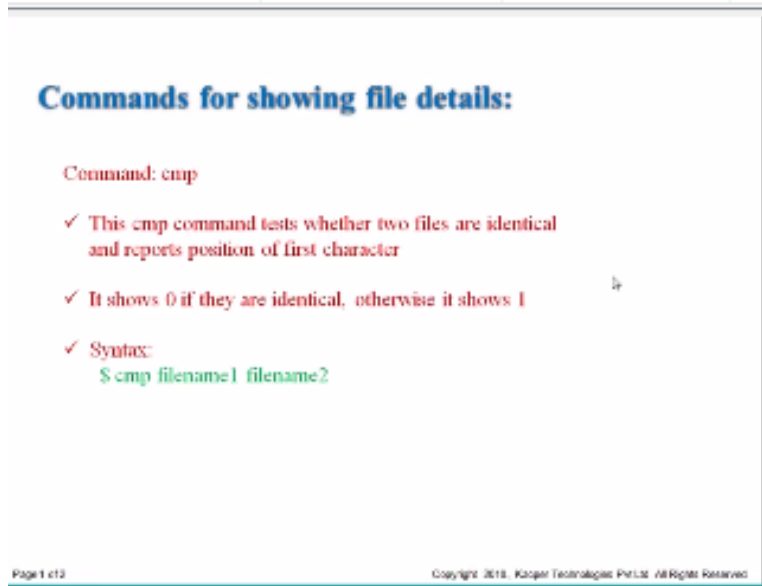
So in this example we initially we store the output of LS minus L using rather than arrow into list 1 now the second command is interesting with pain management form but now 5 - t list - so now let us see what is output so when you do the arithmetic cell it is actually there are all these for zero size files PI 1 PI 2 pi 4 and then there is some pipe or mobile form size which is this one so now what will happen when we do the list too so one thing is lowered to notice is when you specify this lf- and pipe TL this - it displays the output of the LS command in the terminal. This you can see like the for fear of size files and then the list 1 which we created earlier or form for send to the list 1 but now you see that basically produces this output at the output terminal but also now it writes this information in on the pipe for loops - and that list - is not shown here because at the time of keying basically there are two parallel processes so the list - was still getting updated so it does not even know that that particular file existed at that one so the LS minus L produces normal output now the next command.

Is the T with - a which means that we want to append the list - with the same LS minus L command so now if you look at it now the list to come a list - is shown and pretty much you can see that actually go they have like similar type of number of Mohawks so again you can see that actually there are two more some more information that gets written into the list - but in general you can now see the how to work again in this case the T is still it folks has two processes on those processes of executing dependently so if the still does not know the existence of this particular file list -.

So once we finish the execution that when the X we are going to go the directory and then that is when we move up okay this file exists until the second commands will start using the form you I think so now we will go into a few more commands which are quite important because these are



the commands that you use more often in current in the current course as well as go out in the real world we'll be using all the common form much more details.  
(Refer Slide Time: 34:44)



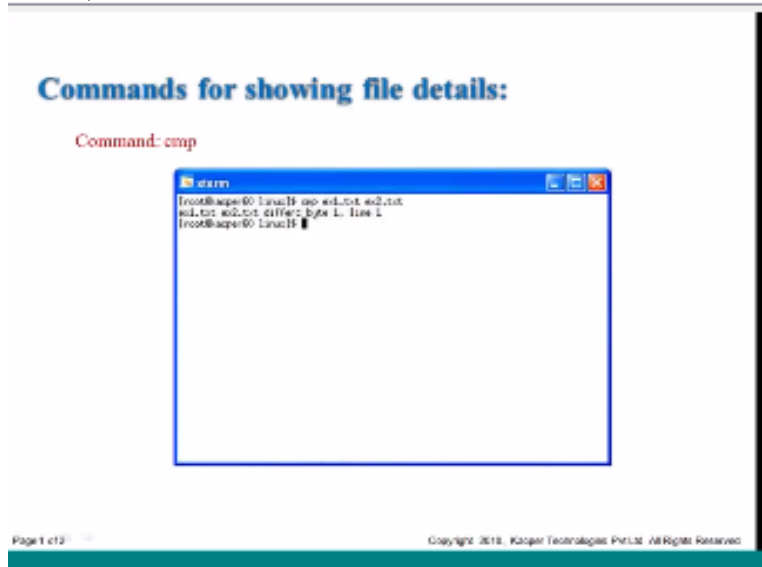
**Commands for showing file details:**

Command: `cmp`

- ✓ This `cmp` command tests whether two files are identical and reports position of first character
- ✓ It shows 0 if they are identical, otherwise it shows 1
- ✓ Syntax:  
`$ cmp filename1 filename2`

Page 1 of 2 Copyright 2018, Kasper Technologies Pvt.Ltd. All Rights Reserved

So the next amount is C MP stands for compare this actually compares to Wiles on reports if they are identical and is report the position of the first character the program itself generated zero output yes they are identical otherwise it produces a one octave and then choose the difference so the syntax is the CMP PI 1 PI 2 you.  
(Refer Slide Time: 35:26)



**Commands for showing file details:**

Command: `cmp`

```
root@kali:~# cmp a1.txt a2.txt
a1.txt a2.txt differ: byte 1, line 1
root@kali:~#
```

Page 1 of 2 Copyright 2018, Kasper Technologies Pvt.Ltd. All Rights Reserved

So here we give a CMP X 1 perfected with two defects here it says essentially like I mean differs by because in white one and that is language so after it agree Goods this it stops and we observe that motion so again CMP rather useful to understand how to what are this what are the

differences between two files and then out of business, understand how to what are this what are the differences between two files and then out of business.  
(Refer Slide Time: 36:16)

### Commands for showing file details:

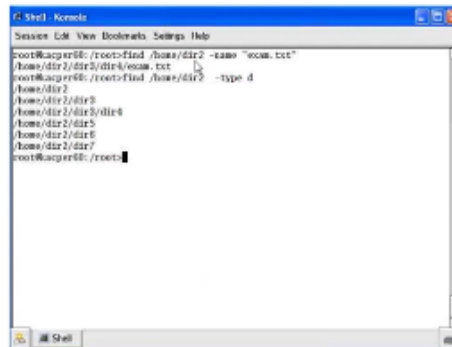
#### Command: **find**

- ✓ This command is used to find the location of a file
- ✓ Syntax: **find path selection criteria**
- ✓ Ex: **\$find . -name "top.v"**  
It will look for top.v in the current directory
- \$find . -type d**  
Finds all directories.
- \$find /home/kacper -type d**  
Finds all directories and subdirectories inside the kacper
- \$find . -type f -name ".\*"**   
Finds all hidden files

You so we already saw the final command actually this long should have gone earlier one and we know that actually basically define the path selection criteria since it is a fat horrible section so they find dot which means that start on the parent directory and then the selection criteria and this my name is chopper, so it looks at look for the top dot me the current directory and all the way under then you can also say like find dot and then flash type d which is then the type to the directory and then it finds all the directories.  
Under and under your current and anything like dot star as we know the hidden files start with dot into a zapping so again like when you do the final part with type at and name is dot star then it finds all the hidden kind of again this all should be followed by also the second item which is what to do with this paper to make – print.  
(Refer Slide Time: 37:42)

## Commands for showing file details:

Command: **find**



```
root@kasper06: /root# find /home/dir2 -name "ocsa.txt"
/home/dir2/dir3/dir4/ocsa.txt
root@kasper06: /root# find /home/dir2 -type d
/home/dir2
/home/dir2/dir3
/home/dir2/dir3/dir4
/home/dir2/dir5
/home/dir2/dir6
/home/dir2/dir7
root@kasper06: /root#
```

So here in the comb there to exam dot text is what we would not find and then we see that the basically and then it also gives you all the income to the home go to is the to go to their for their fall success of our towns .  
(Refer Slide Time: 38:05)

## Commands for showing file details:

Command: **diff**

- ✓ **\$ diff filename1 filename2**  
It compare two files for differences
- ✓ **\$vimdiff filename1 filename2 or sdiff filename1 filename2**  
It will display the two files side by side
- ✓ **Other forms of diff:**  
Xdiff and tkdiff

So now we come to another important command or the diff actually this is used to get the differences between two files so it compares the two file from reports what are the differences so there is a simple disk which is it is basically compares the two pipes and then displays on a running actually it shows running display of what are the differences, and it tags one file by less than and then the other file right there go to distinguish the difference between those two and then move .

It reports like all the lines of all then there are specialized two commands then this file name 1 column 2 or as this filename component to these commands display the files side-by-side it's the part of this bin this you can also have a ticketed which is using the tickle TA interface to create a UI ,so that you can compare those two points, and the good thing about all these commands are you can display the differences and then you can merge the differences you will merge file differences to  $\pi$  B or  $\pi$  base differences 2  $\pi$  here.

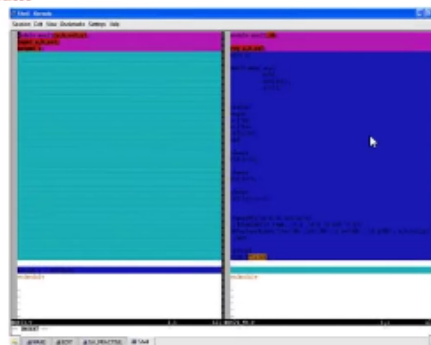
In whichever way that you want go to and then you can do several other things essentially you can even make both the files exactly so basically the differences between in the waterways in a will go into P and deep in both a things like that so again this is another fucking command for wanting to use this in realize, the disk around is often used to find differences between two files either two scripts to run a particular tool or things like that and then it also helps us to find where what we changed this past certain shoes so you can compare.

The current version with the previous version to see like watching and so then you can take comparative options against it or you can go the other way like to see that actually they are different then you can say like okay prove a my old files and then, I will throw in my new fungal value things like that also he will do so it is a very helpful to man.

(Refer Slide Time: 41:01)

### Commands for showing file details:

Command: vimdiff



And you have been using it and lock the elbow rim this again it shows side-by-side and uses the editor all the di we will talk about this in the programming context. In the next term but there are these editors to edit files once a editor is them more for visual editor bi there are other editors like Emacs on the one of the most affordable path yeah, it is very common because there is actually fairly easy to the strength-based so we will like that a lot couple of other things that I

wanted to highlight one is for the use of form tab that you can use tab to complete a particular command or complete a particular file given size.

So this again is run ,I will type everything it is shaping new faces way you can stop at some point and then get to continue and complete that particular so that's also provided . I (Refer Slide Time: 42:22)

**Commands for showing file details:**

**Command: ln**

- ✓ **\$ln target\_filename link\_name**  
Will create a "hard" link to target\_filename called link\_name. Hard links need to be deleted explicitly to remove a file.
- ✓ **\$ln -s target\_name link\_name**  
This command creates a soft link or symbolic link. If you delete the original file, softlink will generate an error.

Page 1 of 2  
Copyright © 2010, Kacooz Technologies Pvt.Ltd. All Rights Reserved

I also wanted to briefly touch upon one additional command before Ln or link so imagine you have a file form of all five which runs it to like couple of gigabytes or even more these days like I mean 50 hundred gigabytes is nothing and then, now you have a another directory their particular program that we want to run and which also need this file as one of these boots now one way to do it is you can just copy that file over.

To the new location and then this particular fire this particular program will access that pine and do whatever it wants and then it finishes position instead of that you can also think of another way here you do not copy the whole file to the new location which can lead to actually like to think one is copy the file over to a new location, since it is running into like multiple gigabyte probably too so the whole system down or take a long time for or the protecting file to be popping the second thing that it has is basically each of those memories.

They are having now another lead you are having like two puppies of the same information and as a result you just occupying the disk space with that and as a result some other important program will want the end times to work because you are blocking the system it is your tools so a better way is to actually link a particular file linking a file means essentially you are creating a pointer to that file in that direction so that the program that wants to access can chase that pointer

and then go back to the links can be created in two ways one is called hardening and then the other one from a soft link or symbolic move.

So let us look at what is hard link is when we specify the command followed by the target file name and then the link so here both of them are arguments and the first argument is what the link that you want to create, and then the second one is what sorry first one is the file name which you want to create a link form and then the second one is the name of the meal that you want to it so if you specify `Ln target file name link name` then it creates the hard link to target file name or link .

So why what is the why is it called a hard link so the hard link is because even though if you remove the original program the program contains still lives under this targeting or essentially the link so if you want to delete a file it is not just enough weight that particular file type system you also need to delete all the hard link references to that side so because otherwise I can still keep the data for the link to work and it will make it work the other command is `Ln -M` and then target name and then the link name.

This creates a soft link or a symbolic link to the particular find target that you want also see it so your issue if you have a soft Ling and then you can say like okay delete my original file then next time when you run the program generates an error saying that hey that particular file is not found even though like it exists in the zymology , so you need to pretty much chase through symbolically find out there exactly things and then add it in place but the key thing is again the symbolic link.

You want the program to fail if it does not find the file so that is the reason why symbolically world are doing emotion problems .

(Refer Slide Time: 47:20)

## Lecture 4 - Topics

- File system commands

- Grep, fgrep,
- Locate
- Uniq, sort
- touch
- diff, tee,

You so in this lecture we covered a number of topics all very important and very useful comments we grep a diff locate unique sought touch if tee, also like the link amounts so I think this should give you a good flavor of the entire Linux system. We will be start starting to talk about the Linux networking in the next session but if this actually like ,I mean we cover all the basics of Linux so and we will also have some more assignments that will be coming your way but you will also do a recap of this lecture in the next one we will start the Linux ,so again thanks a lot thank you very much and Good luck.