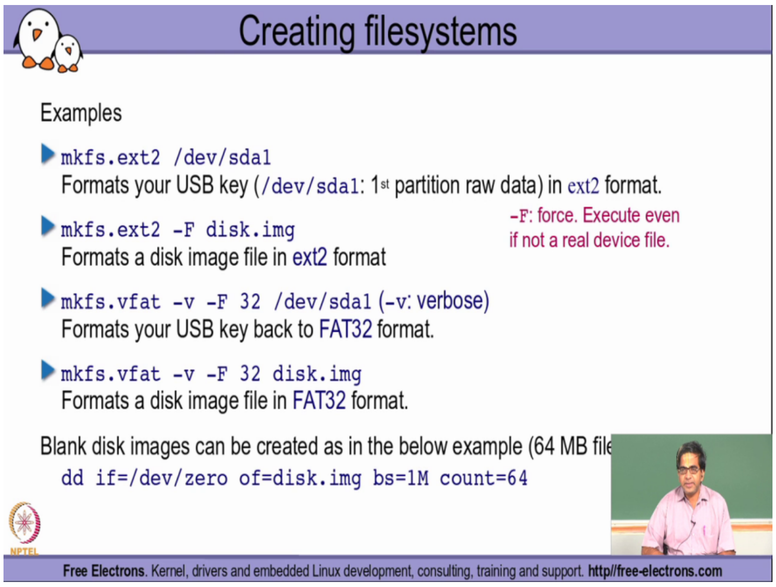
Information Security
Sri Vasan V S Principal Consultant
Department of Computer Science and Engineering
Indian Institute of Technology, Madras
Mod04lec26
Basic Commands

So in this module we will just take a very quick look at some of the basic commands that we do make use of for administering a local Linux System.

(Refer Slide Time: 00:24)


Creating filesystems

Examples

- `mkfs.ext2 /dev/sda1`
  Formats your USB key (`/dev/sda1`: 1st partition raw data) in ext2 format.
- `mkfs.ext2 -F disk.img`
  Formats a disk image file in ext2 format
  `-F`: force. Execute even if not a real device file.
- `mkfs.vfat -v -F 32 /dev/sda1` (`-v`: verbose)
  Formats your USB key back to FAT32 format.
- `mkfs.vfat -v -F 32 disk.img`
  Formats a disk image file in FAT32 format.

Blank disk images can be created as in the below example (64 MB fil

```
dd if=/dev/zero of=disk.img bs=1M count=64
```

Free Electrons. Kernel, drivers and embedded Linux development, consulting, training and support. http//free-electrons.com

So the first command that we will typically require after we have actually partitioned a disk or we have received the partition disk is, for us for us to make use of a command called mkfs. Now what exactly this mkfs command does is that, mkfs basically creates the file system.

So in our initial set of modules we had a very brief look at what a file system is all about, typically without creating the file system, we wouldn't be able to store files in the disk partition that we have unless and until we want to access it as a raw partition like just like typically how databases make use of, but other than the database is part of it if you want to really store data in 8 in terms of logical files one of the first requirements for us to have this kind of utility is that we need to first create a file system on the disk partition.

So the mkfs command is the way by which we will be able to create a file system so that the disk partition is having a very clear meta data information stored inside that exactly tells where the different data blocks are available and where for a particular file, the corresponding data is actually stored. Right? So we had discussed about the I nodes and the data blocks inside the I nodes in our previous module and on a typical system if you take any kind of a Unix flavour you have different types of file system that is actually supported.

So you have something like ext2 you have ext3,ext4,riser fs, xfs. So many different file systems that are possible and most of the times we try to differentiate the file system decision by trying to understand what kind of data we are actually trying to store on that. So there are different kind of functionalities that are provided by different file systems so depending on our specific requirement of what type of data we intend to store on a file system, we basically decide what is the type of file system that we want to create on that particular partition.

So one way which we will take a look at is, I want to create for example A file system called as a ext2 file system. So mkfs dot ext2 is the name of the command that I will make use of and then I specify slash dev slash sda 1. So dev sda 1, is basically the first raw partition of my first disk. Right ?

And I am basically saying that I want to create a ext2 type of a file system on this particular first partition in my first hard disk that has been detected by my OS. Right? Now, other way by which I could actually create a file system is if I want to basically create a disk image file so I am sort of mimicking a file system on a local disk file that I have I can also use minus F option for doing that wherein I can run the mkfs dot ext2 command and use minus F and then say disk dot img. Right?

So the minus F option basically will go ahead and create the file system even when the file that has been given is not a real physical device but a sort of a soft file that is actually available on in on an existing file system itself so that's why I was just mentioning it as a file that will that will sort of mimic a file system or disk partition at the lower level.

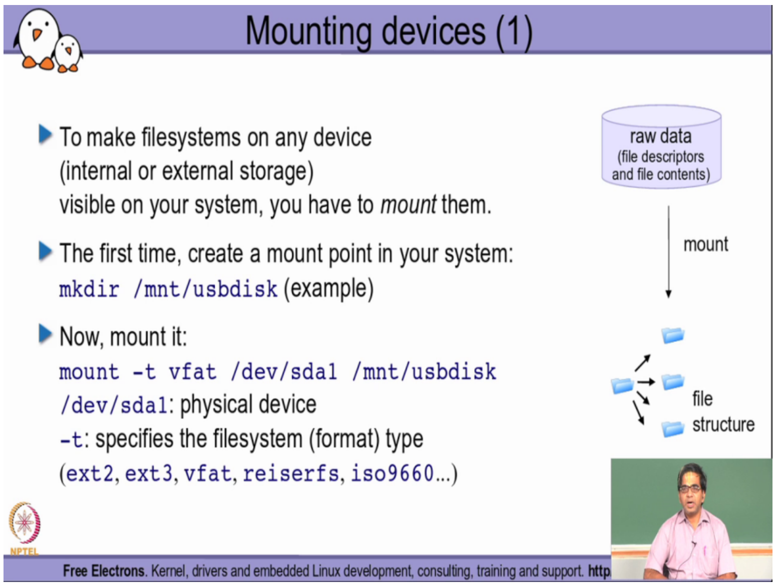
Suppose if I want to basically have a different type of file system created so I could say mksf.Vfat. So Vfat so Vfat is originally from the Windows world. It is a type of file system that was actually available in the windows and those file systems that are supported only on Windows is also you have the support of those file systems in our linux distributions. Right? So I could say mksf.Vfat and then do appropriately create a Vfat fs on whatever disk partitions that has been mentioned, Right?

So likewise if I want to for example create an xfs file system, so xfs is another type of file system I would similarly run a command called mksf dot xfs Right? Pretty much, the arguments will continue to be turning out to be the same, but you can have more details of it if you run the man command and give mksf as the argument to the man command, where it will display you the complete details of how this particular mksf command has actually got to be made use of, Right? So for example if I want to create a blank file which I want to use in my mkfs creation like how we have used here with the minus F option called disk dot IMG. There is a command called dd, so dd basically stands for disk device or device down where I specify IF, IF is basically the input file what is the input file that I should use for creating a file and then OF is basically the output file. So if I say slash dev slash zero as the input file, slash dev slash zero is a basically a very special file.

One of the special files which Linux exposes which will basically give a stream of zeros as the bites whenever that file is read, so whenever I open this file Dev zero which is basically what the IF option to DD command is going to do, it will continuously the driver will basically give it a stream of zero bytes and that stream of zero bytes will be dumped onto this OF that is the output file here, Right? So disk dot img will basically be getting created with all the every bite initialises a zero bite with a block size of 1 MB.

So every input and output on this particular file will be done in the block size of 1 MB and this will be done 64 times, so the file size will be 64 into 1 MB, Right? So how many times it will be done into what is the size every time I do it so that basically becomes 64 into 1 MB which is basically what is a 64 MB file that will be created by running a dd command in this form. So dd command is actually is typically used to create a disk file of a very large size and you could actually have all bites in that files that you have created to be also initialized with zeros rather than having some junk characters by trying to have the IF argument as slash dev slash zero.

(Refer Slide Time: 07:49)



Mounting devices (1)

- To make filesystems on any device (internal or external storage) visible on your system, you have to *mount* them.
- The first time, create a mount point in your system:
  `mkdir /mnt/usbdisk` (example)
- Now, mount it:
  `mount -t vfat /dev/sda1 /mnt/usbdisk`
  `/dev/sda1`: physical device
  `-t`: specifies the filesystem (format) type
  (`ext2`, `ext3`, `vfat`, `reiserfs`, `iso9660`...)

raw data
(file descriptors and file contents)
mount
file structure

Free Electrons. Kernel, drivers and embedded Linux development, consulting, training and support. http

Once I create a file system, how do I actually mount it? Because unless and until we mount that file system we will not be able to access the data that is there inside the file system and neither create a new data inside the file system, Right? So, for that we typically use the Mount command. Whenever we use the mount command, one of the first requirements is that we should have a Mount point on the system right? So the mount point is basically the point in my entire file system hierarchy where this particular raw partition or my removable media is going to get mounted on, Right? So, first I use mkdir command and create the mount point. If for example the mount point is not already available, Right?

So once I have created that mount point whenever it is required so this is basically a one time activity that we will do till such time that I don't have the mount point. So once I do that creation with this mkdir. So creation of the mount point with this mkdir, subsequently every time for me to when I mount it, I don't need to keep creating it again and again, Right? As long as I created on the disk and not let’s say, something under that my slash tmp directory because anything that is created under slash tmp directory, will not be available to you across the (Reboche). So as long as I don’t create my mount point under slash tmp for a particular system, the mount point creation with the mkdir command is typically an activity that will be done only once. Right?

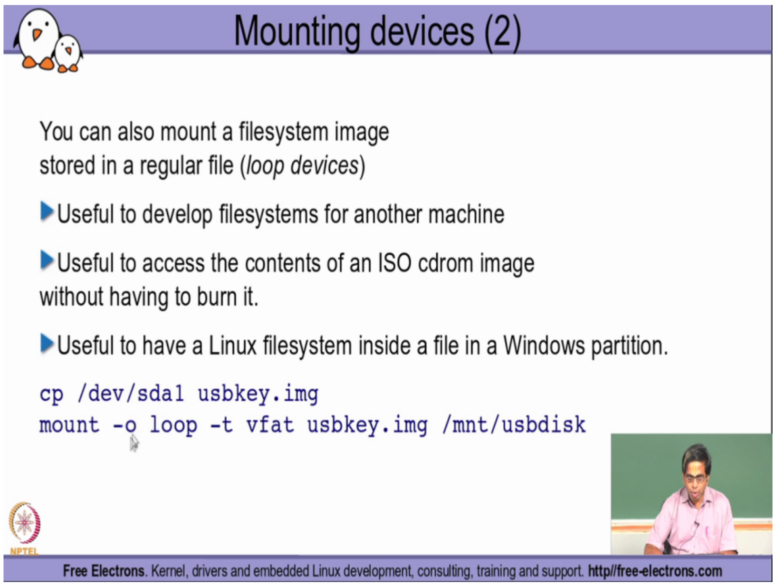
So once I run the mkdir command once on my system and create this mount point, I will basically go ahead and try to mount the mount command, so the mount I specify the minus T option, so the minus T is basically the option to specify what is the file system format that is actually available on that particular disk partition that is getting mounted so if for example it is a Vfat file system I say Mount minus tv fat followed by slash dev slash sda 1. So that is basically the disk partition that is mentioned that is right now required to be mounted and I also specify what is the mount point on which it has to be mounted, Right?

So this raw partition slash dev slash sda 1 which is containing a v fat file system which I have created before with mkfs command that we saw in the last slide, now I am telling with the Mount command this particular file system that is available on dev sda 1 should be mounted onto this mount point, Right? So if I had for example created Let say an ext3 file system or ext4 file system on this mount point, I would have appropriately mentioned that particular file system type as an argument to minus T here, Right?

And if that was really the file system that had been created on the partition, then the mount command will be successful and I will be able to mount it. So once I mounted, I will now typically do a changed directory into this particular mount point, Right? So I will say CD slash mnt slash slash usb disk and whatever data was actually available in this particular raw disk partition, will now be available for me to access under this particular directory. Right?

So that’s basically why we call this argument as the Mount point. So what is this mount point? In my entire file system hierarchy in which the data that I have created originally here on this RAW partition is going to be available to me right now. So that's basically what the mount command is actually helping us with, because without the mount command of that particular disk partition, it is not possible for us to access the data inside that partition.

(Refer Slide Time: 12:00)



Mounting devices (2)

You can also mount a filesystem image stored in a regular file (*loop devices*)

- Useful to develop filesystems for another machine
- Useful to access the contents of an ISO cdrom image without having to burn it.
- Useful to have a Linux filesystem inside a file in a Windows partition.

```
cp /dev/sda1 usbkey.img
mount -o loop -t vfat usbkey.img /mnt/usbdisk
```

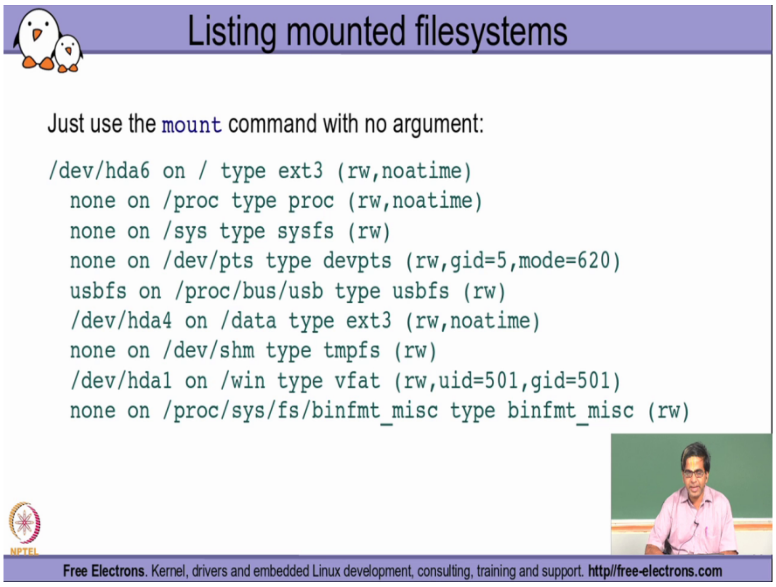
Free Electrons. Kernel, drivers and embedded Linux development, consulting, training and support. **http//free-electrons.com**

So I could also make use of what is called as a loop type of device for mounting especially if I have a Cd-rom image, Right? So we call it as an ISO file system ,so any kind of Cd-rom image that I want to mount it locally as like a normal file system, I use the minus O option, Right? Minus O to the mount command is basically stands for option so the minus O and then I say loop, so I am basically telling that this a kind of a loop device that is internally containing the entire file system image, Right?

And this file system image I am now telling is basically a V fat file system type and this is the file system image file that is available and I want this file system image file to be mounted as this particular on this particular mount point. Right?

So whenever I have a CD image we basically try to use this kind of mount option to have the contents of the file system on the CD mounted as if it is like a file system available on my local hard disk right? So Linux allows me to actually mount them as a special device called as a loop device using this small customisation or a small additional option specified with my existing Mount command.

(Refer Slide Time: 13:36)



Listing mounted filesystems

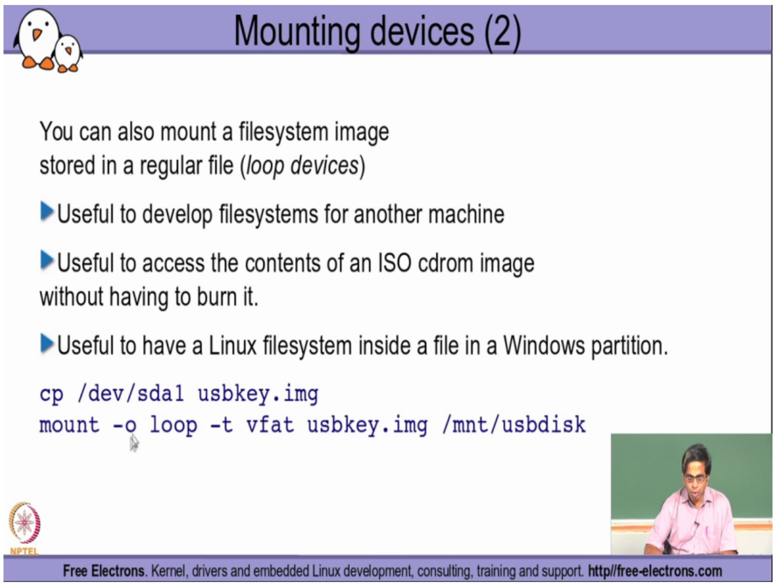
Just use the mount command with no argument:

```
/dev/hda6 on / type ext3 (rw,noatime)
  none on /proc type proc (rw,noatime)
  none on /sys type sysfs (rw)
  none on /dev/pts type devpts (rw,gid=5,mode=620)
  usbfs on /proc/bus/usb type usbfs (rw)
  /dev/hda4 on /data type ext3 (rw,noatime)
  none on /dev/shm type tmpfs (rw)
  /dev/hda1 on /win type vfat (rw,uid=501,gid=501)
  none on /proc/sys/fs/binfmt_misc type binfmt_misc (rw)
```

Free Electrons. Kernel, drivers and embedded Linux development, consulting, training and support. http//free-electrons.com

So if I basically want to find out what kind of file systems are available or mounted on my system currently, I just run the mount command and then it displays me all the mounted file systems right? So for example it says slash dev slash hda6 is mounted on slash, so that means slash that is the route is the mount point for this partition and it is of the type of ext3 and what kind of options have been used it is specifying RW and no access time. So these are like different options that are possible to be specified whenever we run the mount command.

(Refer Slide Time: 14:20)



Mounting devices (2)

You can also mount a filesystem image stored in a regular file (*loop devices*)

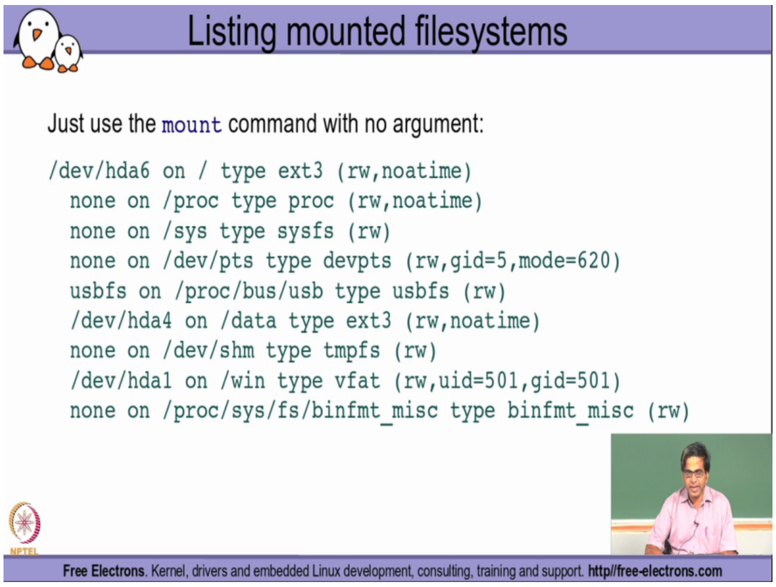
- Useful to develop filesystems for another machine
- Useful to access the contents of an ISO cdrom image without having to burn it.
- Useful to have a Linux filesystem inside a file in a Windows partition.

```
cp /dev/sda1 usbkey.img
mount -o loop -t vfat usbkey.img /mnt/usbdisk
```

Free Electrons. Kernel, drivers and embedded Linux development, consulting, training and support. http//free-electrons.com

So very similar to the minus O that we specify here so I could for example say minus ORW.

(Refer Slide Time: 14:26)



Listing mounted filesystems

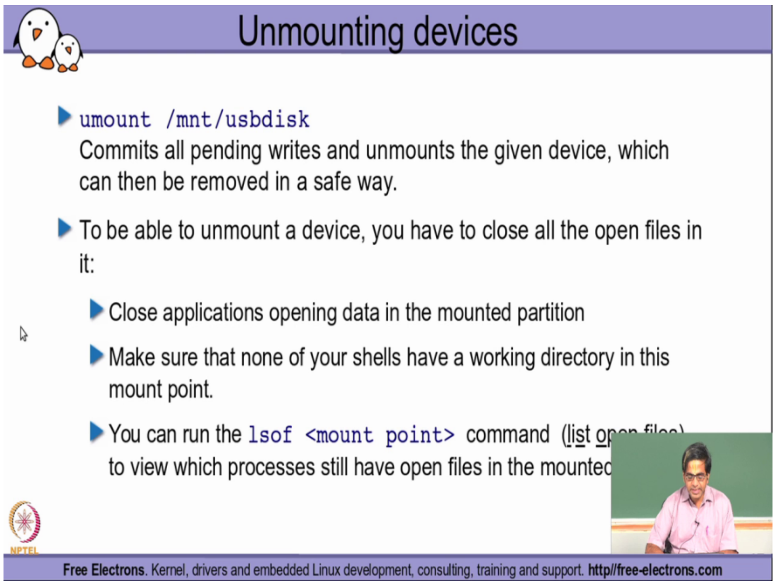
Just use the mount command with no argument:

```
/dev/hda6 on / type ext3 (rw,noatime)
  none on /proc type proc (rw,noatime)
  none on /sys type sysfs (rw)
  none on /dev/pts type devpts (rw,gid=5,mode=620)
  usbfs on /proc/bus/usb type usbfs (rw)
  /dev/hda4 on /data type ext3 (rw,noatime)
  none on /dev/shm type tmpfs (rw)
  /dev/hda1 on /win type vfat (rw,uid=501,gid=501)
  none on /proc/sys/fs/binfmt_misc type binfmt_misc (rw)
```

Free Electrons. Kernel, drivers and embedded Linux development, consulting, training and support. **http//free-electrons.com**

And then it will mean that this particular file system has to be mounted as read write as well and not just as read only. So if I wanted to do mounting as read only any kind of modification that I tried to do on the file system, so whether I am trying to create a new file, whether I am trying to modify an existing file, all these kind of operations will be not allowed at that particular point in time and the file system has been mounted as read only. So it has to be mounted as read write only then any kind of modification operations on the file system, error creating a file, modifying existing file, all those things will be allowed.

(Refer Slide Time: 15:05)



Unmounting devices

- `umount /mnt/usbdisk`
  Commits all pending writes and unmounts the given device, which can then be removed in a safe way.
- To be able to unmount a device, you have to close all the open files in it:
  - Close applications opening data in the mounted partition
  - Make sure that none of your shells have a working directory in this mount point.
  - You can run the `lsof <mount point>` command (list op
    to view which processes still have open files in the mounted

Free Electrons. Kernel, drivers and embedded Linux development, consulting, training and support. **http//free-electrons.com**

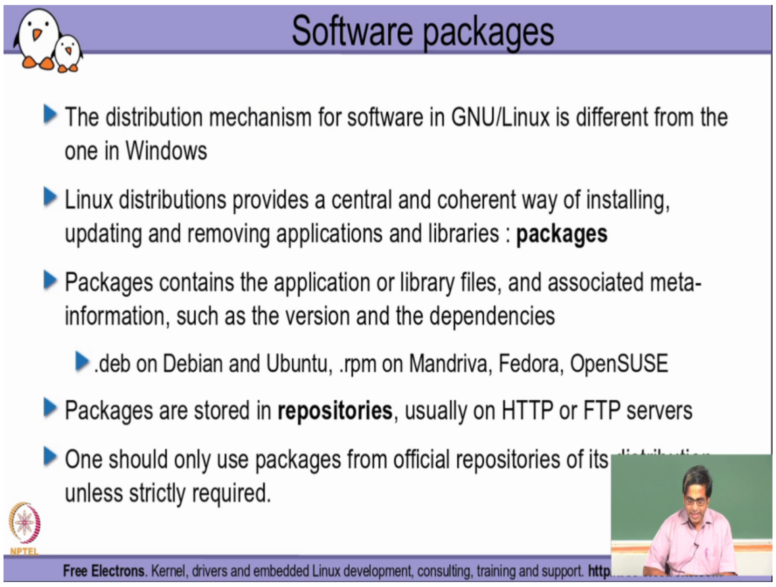
So if I want to un mount basically a file system, I could either specify the file system, mount point, or I could specify the corresponding partition and then run the command called un mount. So that is basically like Umount. So U mount stands for un mount so U mount I can specify the corresponding partition which has been mounted or I could also specify the mount point, both are accepted as arguments by umount .

So unless and until all processes that are using this particular file system has stopped using it, so suppose I have a shell process in which I have gone inside a directory which is available in that particular file system, right?, If I have done a CD into directory inside the file system and I on another shell prompt, I try to un mount it, that I basically try to u mount it, Unix system will not allow me to do that because my shell processes mark this particular file system as busy, Right ? Because one shell process is right now the current directory is inside this particular file system.

So to be able to unmount the device I have to basically have all the open files within the file system to be closed, so how can I do it so I have to be close all the applications that have data opened in this particular mounted file system and then as I was telling, I should not have any of my shells have the working directory in this mount point. And if I want to basically find out which is actually using any kind of data inside this file system, I have a command called lsof. So lsof basically stands for list open files, Right?

So, it basically will tell which process still has open files in that particular mounted partition, Right? So on some distributions this might not be available by default but it is definitely a downloadable and then installed which can, which could come in handy for the system administrator whenever he has to basically, sort of, dynamically mount and unmount filesystems especially when lot of removable media is used as USB, pen-drive and those kind of media stuff.

(Refer Slide Time: 17:24)



Software packages

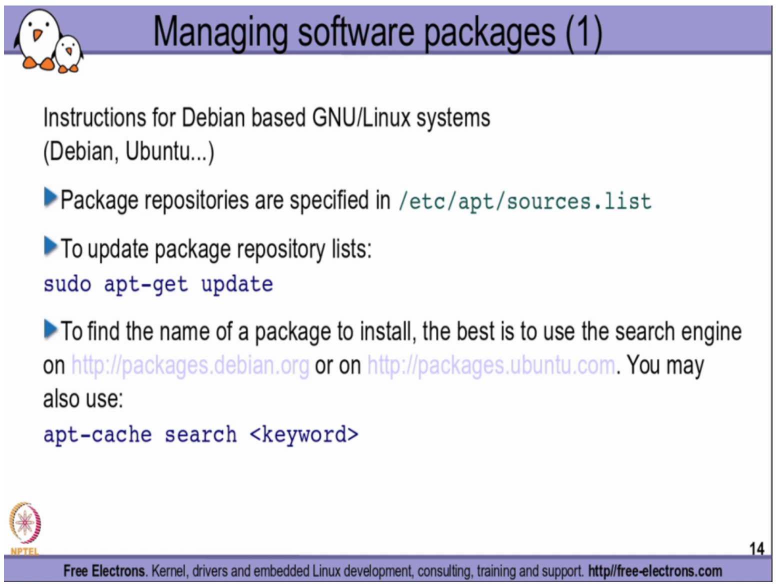
- The distribution mechanism for software in GNU/Linux is different from the one in Windows
- Linux distributions provides a central and coherent way of installing, updating and removing applications and libraries : **packages**
- Packages contains the application or library files, and associated meta-information, such as the version and the dependencies
  - .deb on Debian and Ubuntu, .rpm on Mandriva, Fedora, OpenSUSE
- Packages are stored in **repositories**, usually on HTTP or FTP servers
- One should only use packages from official repositories of its unless strictly required.

Free Electrons. Kernel, drivers and embedded Linux development, consulting, training and support. http

So any kind of software is installed with a command called as an apt get. So these are what is called as packages so any kind of an application from the Linux point of view, is referred to as a package which is typically stored in Linux distribution repositories. So there will be different repositories which are accessible either over HTTP or over FTP depending on what are the command line tools that we are using to download a package and then install it on our local system, Right?

So by default whenever we try to install a Linux distribution, there will be a basic set of packages that comes in with the default installation but over and above that there are thousands of packages that are available, thousands of different applications and tools that are available which would be,

(Refer Slide Time: 18:20)



Managing software packages (1)

Instructions for Debian based GNU/Linux systems
(Debian, Ubuntu...)

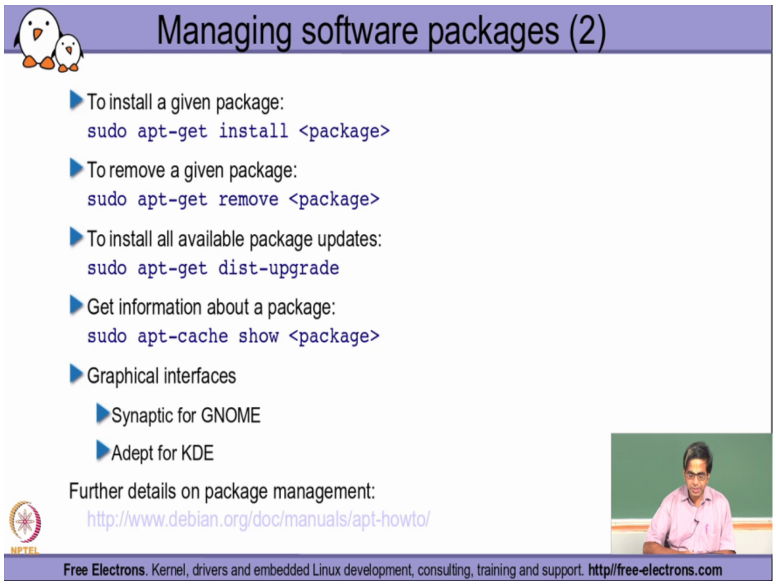
- Package repositories are specified in `/etc/apt/sources.list`
- To update package repository lists:
  `sudo apt-get update`
- To find the name of a package to install, the best is to use the search engine on http://packages.debian.org or on http://packages.ubuntu.com. You may also use:
  `apt-cache search <keyword>`

14

Free Electrons. Kernel, drivers and embedded Linux development, consulting, training and support. http//free-electrons.com

We could optionally download and install and on a distribution like a Debian or Ubuntu distribution, you can actually run the command called Sudo apt-get update to update my list of repositories on my local system which is what will be referred whenever I tried to do a installation of a new package, Right?

(Refer Slide Time: 18:56)



Managing software packages (2)

- To install a given package:
  `sudo apt-get install <package>`
- To remove a given package:
  `sudo apt-get remove <package>`
- To install all available package updates:
  `sudo apt-get dist-upgrade`
- Get information about a package:
  `sudo apt-cache show <package>`
- Graphical interfaces
  - Synaptic for GNOME
  - Adept for KDE

Further details on package management:
http://www.debian.org/doc/manuals/apt-howto/

Free Electrons. Kernel, drivers and embedded Linux development, consulting, training and support. http//free-electrons.com

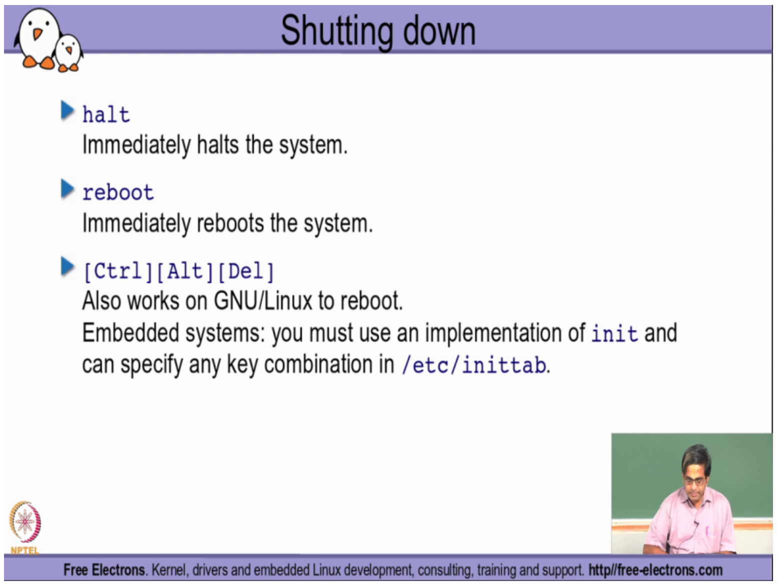
So once I do the repository update, it will go ahead and update the entire set of repository list that is available for this particular distribution version, after which I could actually go ahead and do a apt- get installed, or whatever package I want to download and install on my local system, Right?

This command will download the package, check for the dependencies to be available on the local system, if there are independencies, it will query you, whether you want to install those dependencies also. If you give a concurrence for that it will first and install the dependencies and then go and install this particular package, Right? Now another most commonly used command is once we are done using the package, for any reason we want to remove the package, we also say sudo apt-get remove package.

So sudo is basically the command that was discussed in one of the previous modules which could be made use of, to temporarily run a command as a super user, Right ? So apt-get is a command that will work only as a super user, so even when I have logged in as a normal user, if I want to run the apt-get command I will have to run with sudo and then I will be queried for my super user password that is the root password of my local system. Once I type in the password correctly, and the system has authenticated me as the correct root user, it will go and execute the apt-get command which is basically going to either download the package from the network and then do the complete installation process because installation process requires Super user privileges for it to be successful that is basically the reason why we do all the apt-get commands under the sudo as an argument, Right?

So sudo will be able will help the argument when whatever command has been given as an argument to be run as a super user and not as a normal user. If you try to run the apt-get command as a normal user, it will tell an error message back saying that you don't have enough permission or a permission denied message will come back on your terminal window.

(Refer Slide Time: 20:55)



Shutting down

- `halt`
  Immediately halts the system.
- `reboot`
  Immediately reboots the system.
- `[Ctrl][Alt][Del]`
  Also works on GNU/Linux to reboot.
  Embedded systems: you must use an implementation of `init` and can specify any key combination in `/etc/inittab`.

Free Electrons. Kernel, drivers and embedded Linux development, consulting, training and support. http//free-electrons.com

So some of the other system administration command that you would possibly to use very frequently is halt or reboot so halt will basically halt the system, reboot will basically halt the system, but also bring up the system after the halt. So if you want to basically shut down and then restart the system, you could run the reboot command or alternatively in a GNU/ Linux based system you could also press control alt delete, if this key is actually mapped to the reboot command. So that's again another way by which you could actually do. So this basically brings us to the end of this module.

Thank you!