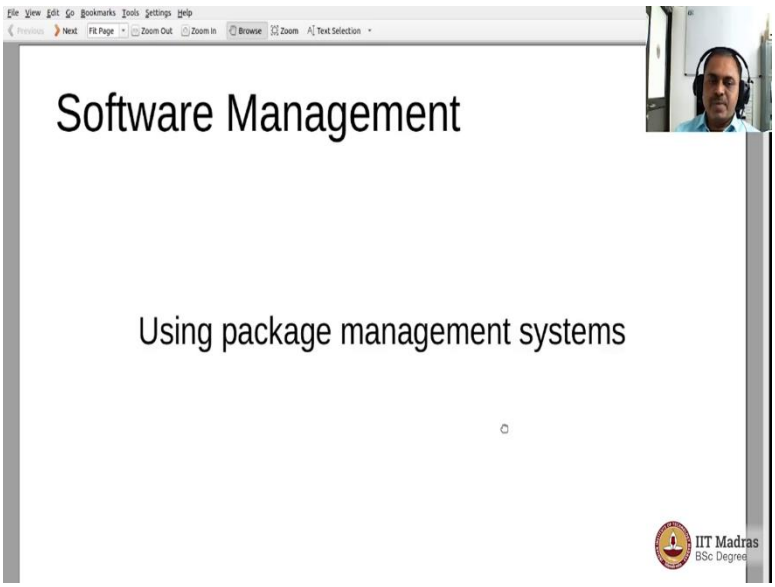
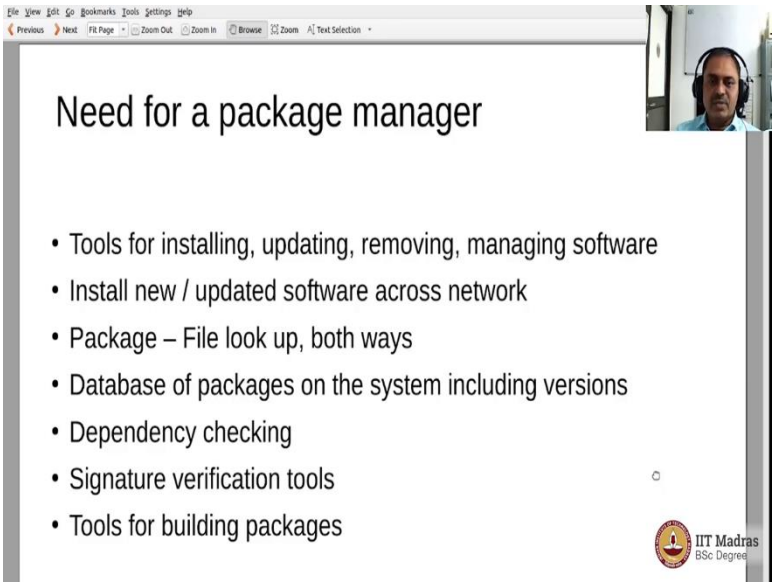
System Commands
Professor Gandham Phanikumar
Metallurgical and Material Engineering
Indian Institute of Technology, Madras
Software Management Part 01

(Refer Slide Time: 00:14)





Welcome to the lesson on Software Management. In the last two decades, managing software on Linux operating systems has become very convenient. Thanks to the package management systems. In the past, one had to download the source code compile, and place the binaries and libraries in their respective locations, and in the process also handle any conflicts with existing libraries, etc. But today, this has been made very convenient, because the package management system would take care of all those details for you.

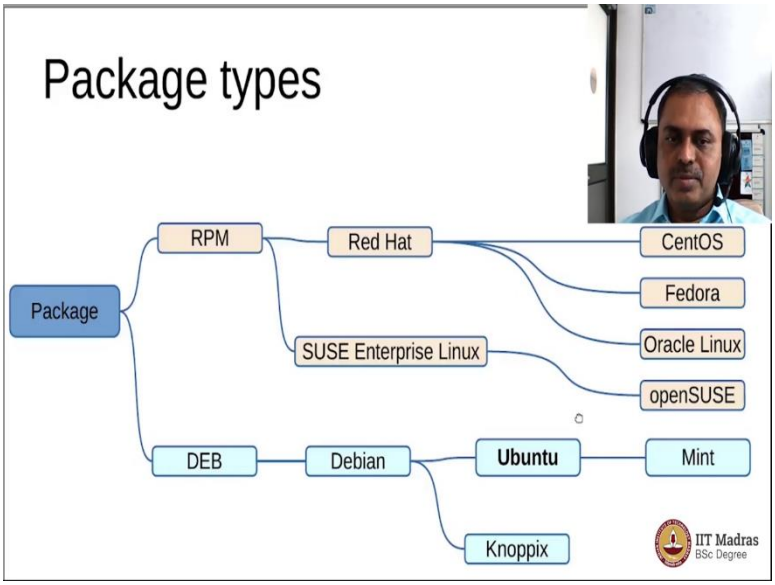
Why do we need to have a package manager? It is very simple. We need tools to install, update, remove and manage the software, and doing it manually would be quite painful, particularly, with the number of packages runs into several thousands, which is typical for a fully installed Linux operating system today. We also want to be able to install the new and updated software across network that is, we may not have the software available as a disk or as a USB drive with us readily, and we may want to do the updating over the Internet.

Sometimes we are interested in knowing what are the various files that are supplied by a given package. And sometimes we are also wondering, which package has supplied a particular executable that we have become familiar using it. So, both ways a package manager would let you look up the database and answer your queries. So, the database of packages on the Linux operating system would include the versions, and their compatibility and requirements, so that it makes your life quite easy in upgrading or adding new packages without disturbing the existing setup.

And how do you verify whether the software tools that you are installing are coming from the original software vendor or the authentic Linux source? There are ways by which signatures are placed along with the software tools, which help you verify the authenticity of the source of these tools, so it is a very good idea to pick up software tools for any Linux operating system from the respective source of that particular flavor of Linux, and not as a binary from any arbitrary website on the Internet.

There are some packages which come as a source code, and you need tools to build executable versions of those tools, and package manager also would take care of such aspects. This is particularly true of kernel modules, so that whenever the kernel gets updated, certain modules need to be compiled for that particular version and made available.

(Refer Slide Time: 03:03)



There are two major package types that are available. One is RPM and the other is DEB or Debian. The RPM format is followed originally from the Red Hat Linux operating system. Derived from the Red Hat operating system our CentOS, Fedora and Oracle Linux. The RPM format is also followed by the SUSE Enterprise Linux firmware the OpenSUSE operating system has been derived.

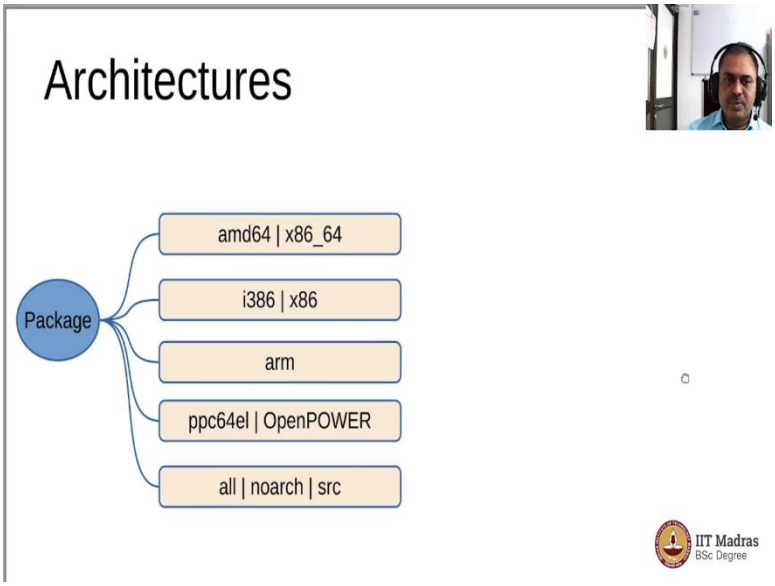
The Debian format of package type is from the Debian Linux operating system from where the Ubuntu Linux has been derived. The Mint Linux is also a derivative of Ubuntu and therefore it is also a Debian type of Linux. Topics is another flavor, which is used to run Linux off a media drive such as a CD drive or a DVD drive, and that is also a derivative from Debian.

As you have noticed, Ubuntu is the most popular desktop operating system including its derivatives, and therefore the package type that we must be familiar is Debian package. Whereas those who are maintaining Linux servers, they should be familiarized with the RPM package type.

(Refer Slide Time: 04:28)

Let us check what type of an operating system we have on the desktop that we are using. I would just open the shell to check what is that let us see what type of an operating system we have. So, we can run the command lsb release with an option minus a and you see that the desktop operating system that is right now running in my machine is Ubuntu version 20.04 LTS, and the nickname for this operating system with the version 20.04 is focal. So, when you look for some packages for this particular version of Ubuntu, you could actually search by the nickname which makes it a little easier to identify those.

(Refer Slide Time: 05:04)



Linux operating system is available for a wide variety of architectures. So, in the past, we had the 32-bit hardware, so we would have Linux for i386 architecture or x86 architecture.

Today we have 64-bit hardware, and therefore, we would actually have the operating system for AMD 64 or x86 underscore 64, as the architecture. And Linux is now also available on arm architecture which is popularly used by the mobile phones. Linux is also available for power PC architecture.

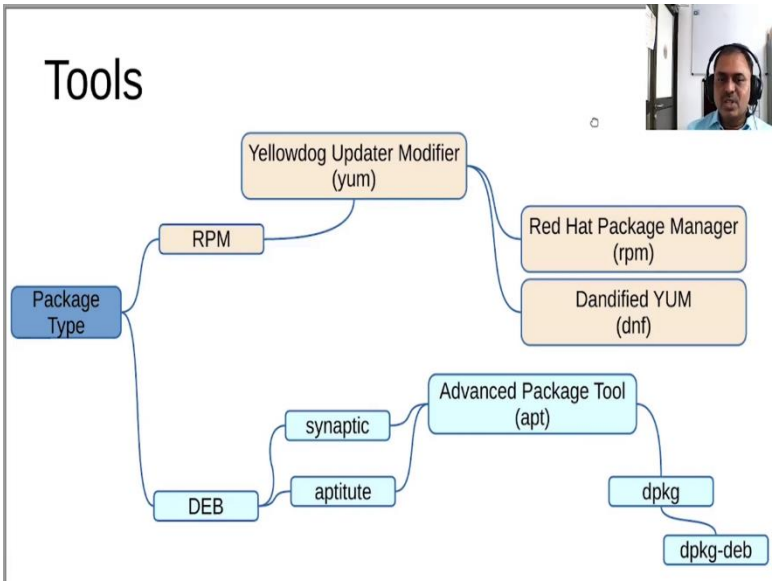
You may have heard of the RISC fifth generation risk five architecture. For example, in the context of Shakti Processor, those architectures are also readily supported by the Linux operating system. And sometimes packages come, which are not tied to any particular architecture, because they may be a source code are just pieces of files that are coming as a bundle, so they would be denoted with an architecture as noarch or all or src.

So, look out for the string, which indicates the architecture for which the particular package has been built, and also which architecture your desktop mission is, so that you would install a compatible package. Usually, this is taken care by the package manager, so you do not have to worry much.

(Refer Slide Time: 06:45)

Let us look at what type of an architecture we see. You type a command, uname minus a, and you will see what type of a kernel is running right now, so my desktop is running kernel 5.11, and you could see some, the string that is a x86 64 architecture. So, once you have identified which version of the Linux and what architecture then by and large, you can actually pick up a package and install it yourself, but it is a good idea to go through the package management.

(Refer Slide Time: 07:16)



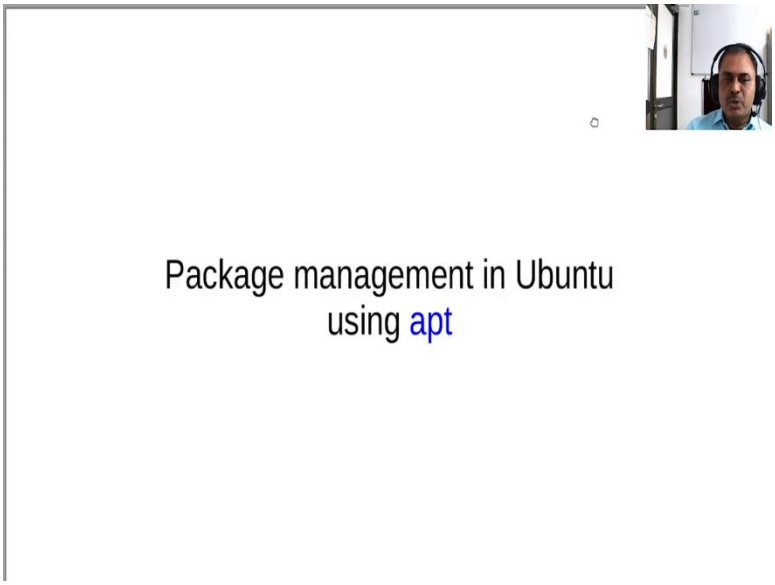
So, here are some package management systems that are widely used across both the RPM type of package as well as Debian type of package. So, the so called YUM software, which expands to Yellowdog Updater Modifier is popularly used to manage the RPM packages. On Red Hat, you would have the RPM, as application which would help you manage the packages, and the YUM package management system would actually call RPM to install, update or remove the RPM packages.

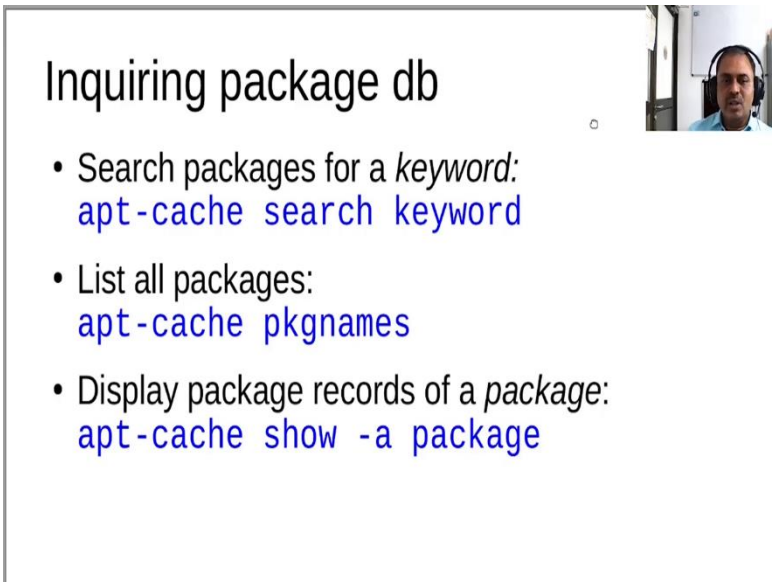
In the recent past the YUM package management system has been modified, and it is now called as dnf which expands to Dandified YUM. So, if you have a Red Hat open system, then you would use dnf to manage the packages. Most of the options are quite compatible, so you should be able to readily use those comments.

However, we are going to learn in depth how to manage the Debian packages, which can be done both using a graphical user interface, as well as command line. So, synaptic is a graphical user interface, and aptitude is a command line interface available to manage packages of the type Debian, and at the backend is the advanced package tool, APT, which does the package management.

And there is a tool called dpkg, which is called by APT to actually install the packages. So, we would also use some of the commands directly using dpkg, which further makes calls to dpkg hyphen, Deb, which we normally do not have to call directly ourselves. So, let us go and explore further Debian package management on Ubuntu our similar operating system using a APT and dpkg.

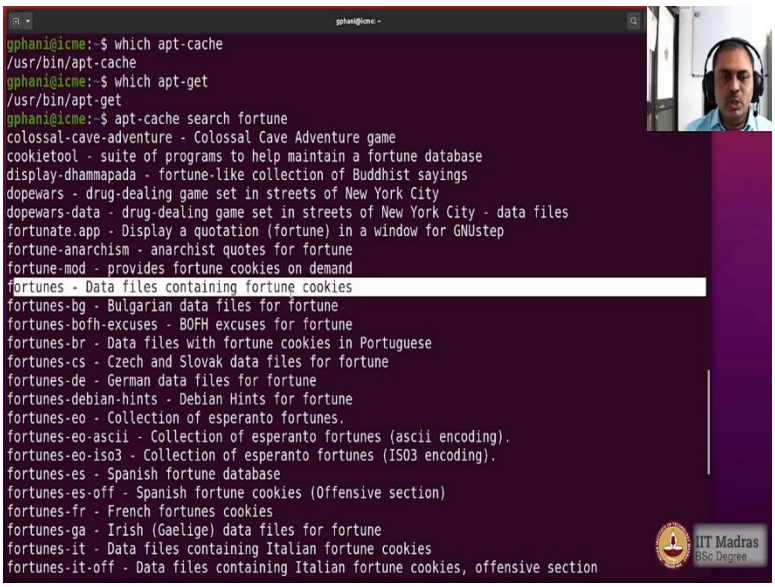
(Refer Slide Time: 09:11)

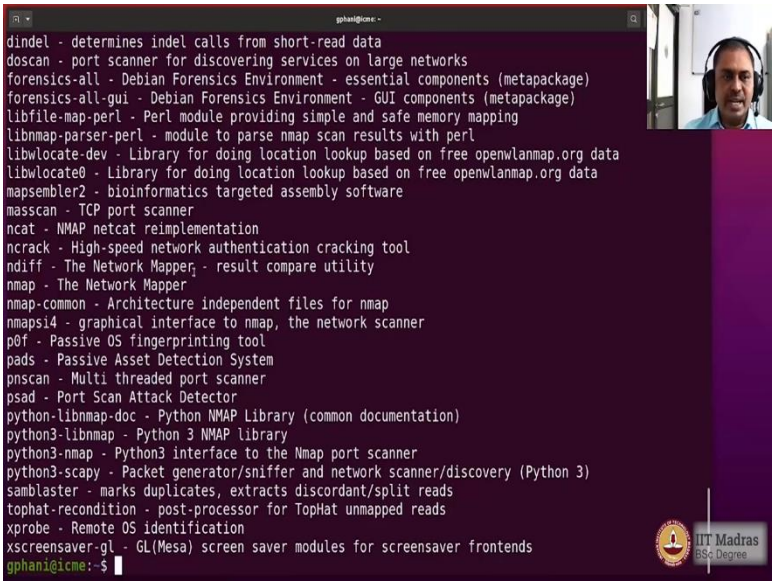


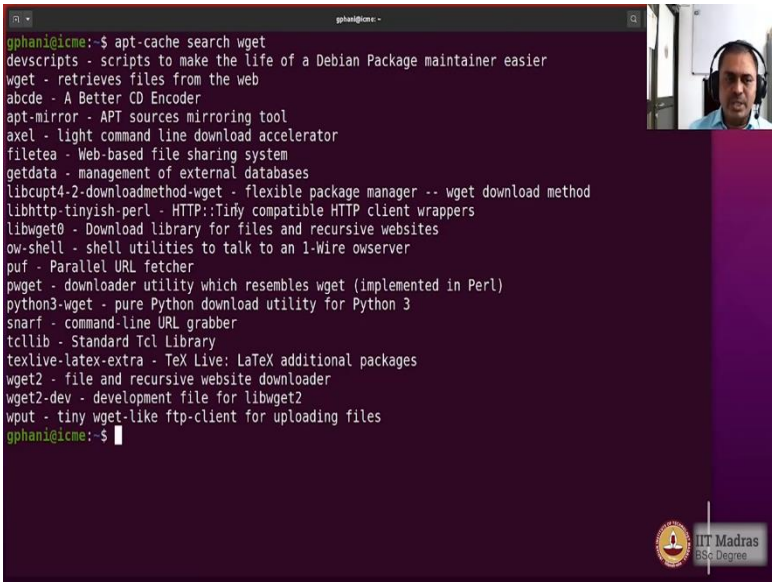


So, we will now use the APT package management system to explore the packages on an Ubuntu operate system. So, here are a few ways by which you can use the apt cache command, so you could search for a keyword, you could ask for what are the packages that have been installed on the operating system, and you could also ask for the detailed information of a particular package or a list of packages.

(Refer Slide Time: 09:53)







So, let us look at these commands by actually running them on the command line. So, before we proceed, we should first check whether these comments are available, so that we would do by using the which command which apt. So, there is a file that is already available, so the apt is available for us to explore.

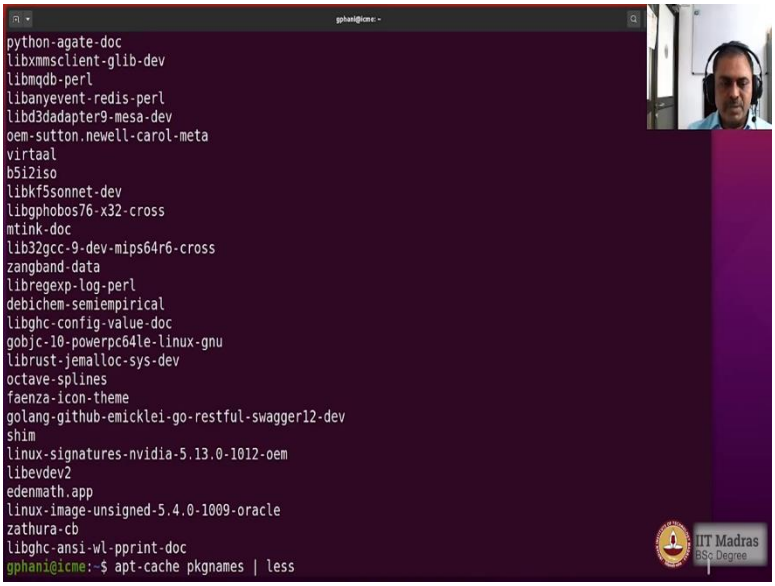
So similarly, we would try apt cache that is available, and apt-get, so that is also available. So, we will now first explore the apt cache command, which does not require the super user permissions so directly, we can start using that. So apt, and then search, and you can give a keyword to search the packages that may have that particular keyword in either the description or the title of that particular package.

So, let us search for a package called fortune. And you will see that there is a lot of packages which have that word. And you see that there is a package here called fortunes, which is

basically a data file containing a fortune cookies. So, it is something funny that it would display on the screen, if you are under command fortune.

Let us search for a string like nmap to see what are the packages that have that particular keyword, and you will see that there are many such packages, but the one which we may have in our mind is the network mapper called nmap. And let us say somebody mentioned to us wget and we want to know what is it about, and you will see, that there is definitely a package by the name wget which retrieves files from the Internet. So, you could actually like that search for packages using a keyword and then identify the specific package that you may be interested in knowing. Now, we have identified some packages now, so let us go and see whether we have those installed or not.

(Refer Slide Time: 11:38)

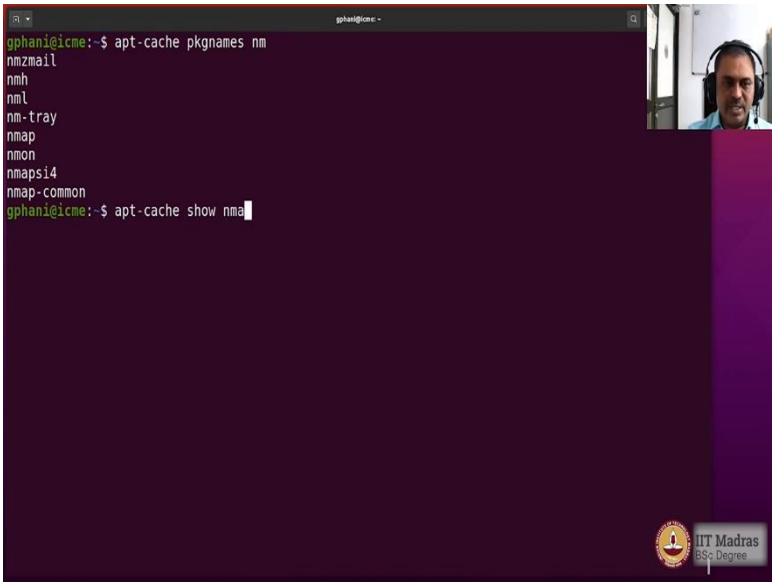


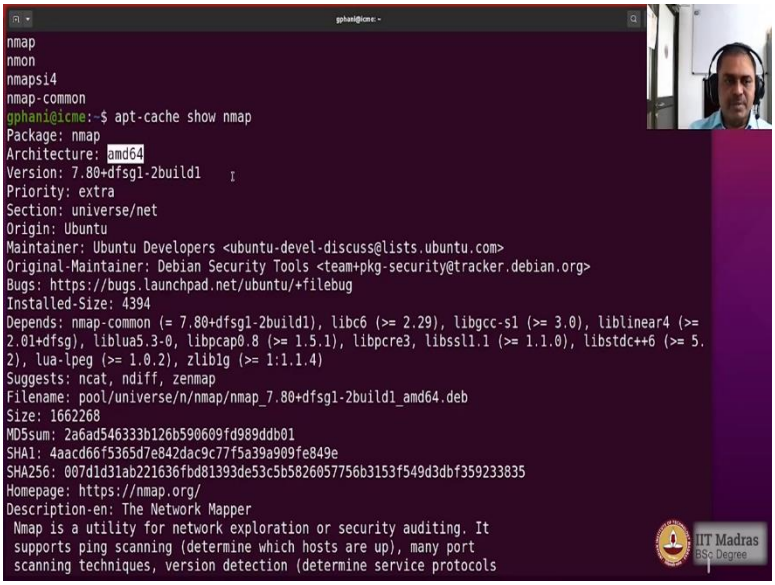

gphani@icme: ~
ruby-net-ntp
libghc-stmonadtrans-dev
sdparm
lib64gfortran-10-dev-x32-cross
libcudf-dev
fusiondirectory-plugin-dhcp
libboost-python1.67.0
python3-qpack
maskprocessor
perceptualdiff
libofx-doc
libjsonpath-java
libdbd-sybase-perl
librust-fern-dev
piuparts
python3-csvkit
libghc-maths-dev
libjalali-dev
libdeflate0
libxmlrpc-epi-dev
python3-dfdatetime
libqtav-dev
linux-image-5.4.0-1051-gke
myspell-el-gr
libplack-app-proxy-perl
stgit-contrib
libgtkdsv-3-dev
libsundials-cvodes3
:
IIT Madras
BSc Degree

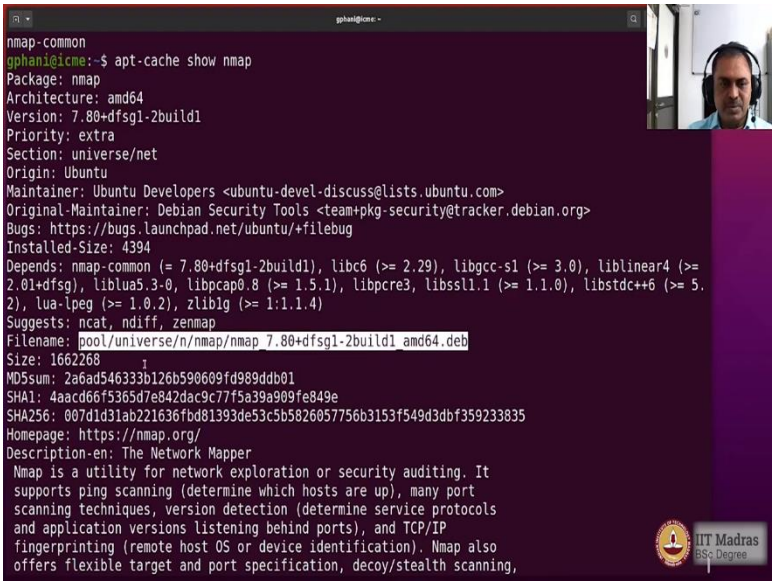

gphani@icme: ~
0ad
0ad-data
0ad-data-common
0install
0install-core
0xffff
2048-qt
2ping
2to3
2vcard
3270-common
389-ds
389-ds-base
389-ds-base-dev
389-ds-base-libs
3dchess
3depict
3dldf-doc
4g8
4pane
4store
4ti2
4ti2-doc
64tass
6tunnel
7kaa
7kaa-data
9base
:
IIT Madras
BSc Degree


gphani@icme: ~
gphani@icme:~$
IIT Madras
BSc Degree


gphani@icme: ~
gphani@icme:~$
IIT Madras
BSc Degree


gphani@icme: ~
gphani@icme:~$ apt-cache pkgnames nm
nmzmail
nmh
nml
nm-tray
nmap
nmon
nmapsi4
nmap-common
gphani@icme:~$ apt-cache show nma
IIT Madras
BSc Degree


gphani@icme: ~
nmap
nmon
nmapsi4
nmap-common
gphani@icme:~$ apt-cache show nmap
Package: nmap
Architecture: amd64
Version: 7.80+dfsg1-2build1
Priority: extra
Section: universe/net
Origin: Ubuntu
Maintainer: Ubuntu Developers <ubuntu-devel-discuss@lists.ubuntu.com>
Original-Maintainer: Debian Security Tools <team+pkg-security@tracker.debian.org>
Bugs: https://bugs.launchpad.net/ubuntu/+filebug
Installed-Size: 4394
Depends: nmap-common (= 7.80+dfsg1-2build1), libc6 (>= 2.29), libgcc-s1 (>= 3.0), liblinear4 (>=
2.01+dfsg), liblua5.3-0, libpcap0.8 (>= 1.5.1), libpcre3, libssl1.1 (>= 1.1.0), libstdc++6 (>= 5.
2), lua-lpeg (>= 1.0.2), zlib1g (>= 1:1.1.4)
Suggests: ncat, ndiff, zenmap
Filename: pool/universe/n/nmap/nmap_7.80+dfsg1-2build1_amd64.deb
Size: 1662268
MD5sum: 2a6ad546333b126b590609fd989ddb01
SHA1: 4aacd66f5365d7e842dac9c77f5a39a909fe849e
SHA256: 007d1d31ab221636fbd81393de53c5b5826057756b3153f549d3dbf359233835
Homepage: https://nmap.org/
Description-en: The Network Mapper
 Nmap is a utility for network exploration or security auditing. It
 supports ping scanning (determine which hosts are up), many port
 scanning techniques, version detection (determine service protocols
IIT Madras
BSc Degree



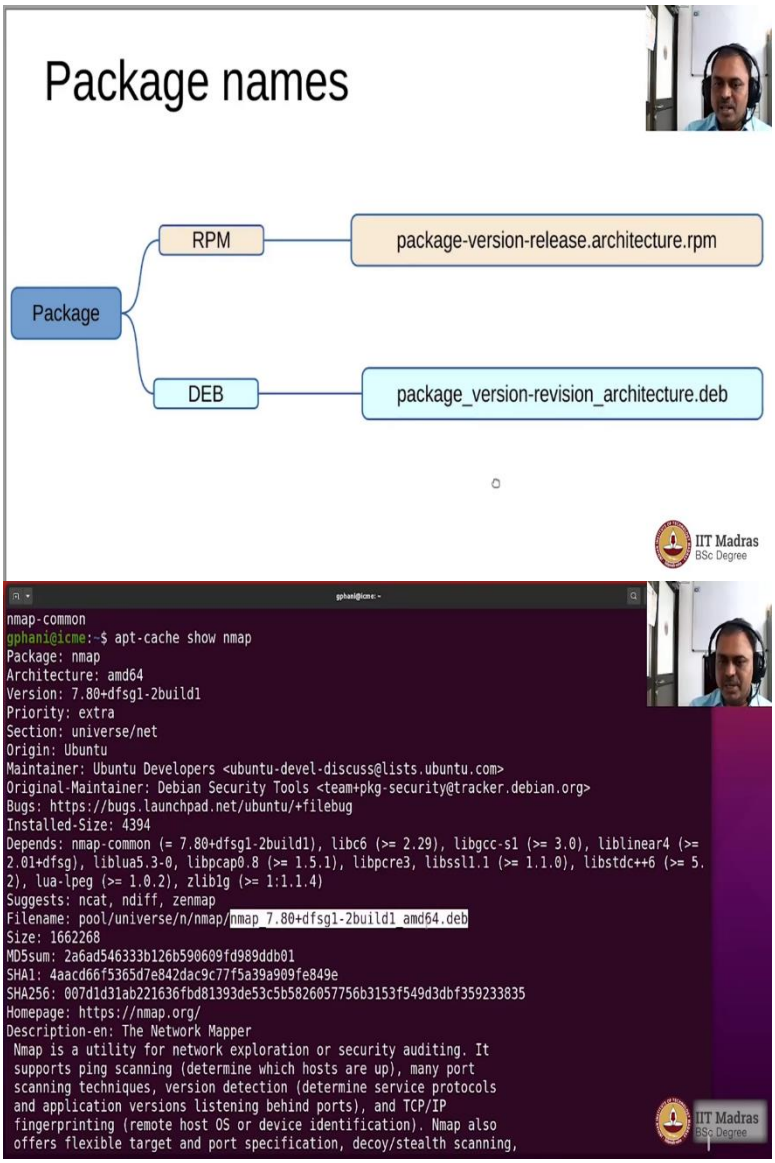
Let us see what are the packages that are installed on this particular system. So, we will do that by passing a keyword to the apt cache command pkgnames. And it would actually give a long list of packages that are installed, so we could actually then pipe it to the less command so that we could see the screen output page by page. So, you could see now that you could scroll through the list.

Now, you see that this list does not seem to have any sorting order, so we could also then use a sort to look at those page by page. So, you can now see that it has been sorted, so initially numerically after that alphabetically, so there are a lot of packages that are installed on this particular system. If you want to avoid looking through the entire list, but only those packages that are starting with a particular character or a set of characters, you could also do that apt-cache pkgnames and then after that, you could start with let us say, nm, and you will see that all those packages, which have the first two characters as nm will be displayed, and

then nmap is one of those packages that has been installed. So, we want to know a little bit more about that package, so let us go ahead and ask apt-cache to show the details of that particular package nmap.

So, you can now see that the details are being displayed here. So, it says that this package is part of the AMD 64 architecture, which is compatible with what we have. And this particular package has a name for the file from the file that it has come from. The file that this package came from is listed here. And if you look at the name, here, you can make out that it has a particular format. So, let us go and explore a little bit more about this format now.

(Refer Slide Time: 13:35)



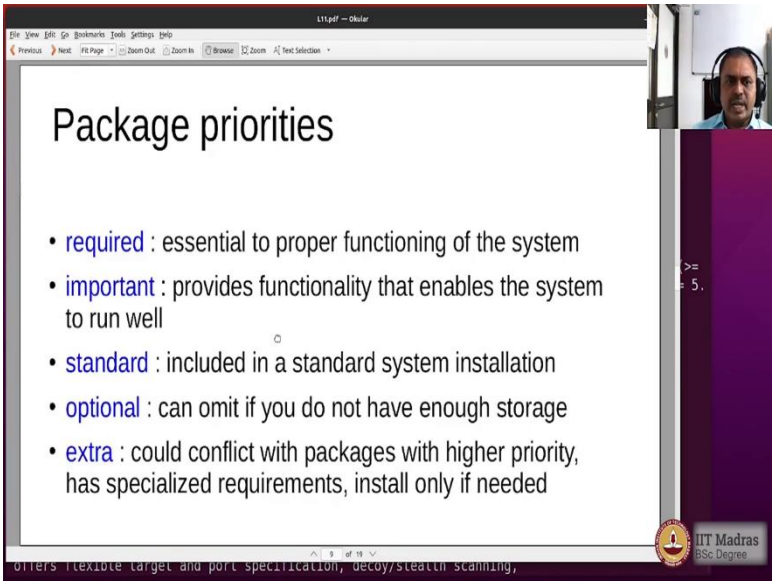
So, here you see the package name is given in a particular format, but the, and it is slightly different between the RPM and DEB. So, for the DEB format, you can see that the name of

the package is followed by an underscore after that there is a version and a revision, and then there is an underscore and then the architecture dot deb. So, let us compare this with the file name that we just saw.

So, the file name that we just saw is having two underscores. So, you can see that the string before the first underscore is the name of the package nmap and that you can already confirm here, and then you can see that the version is here so 7.80 plus df sg1 that is version 7.8 you can say that is a version, and after the hyphen is the revision.

So, 2build1 is a revision so there are many revisions of a particular version that would come. And then after the second underscore is the architecture name so AMD 64 is architecture. So, you can see that the file name of the package already gives you three pieces of information. The name of the package, the version and the revision, as well as, the architecture for which that particular package has been built. So, there are some more pieces of information that would come along, which we would explore now.

(Refer Slide Time: 14:53)

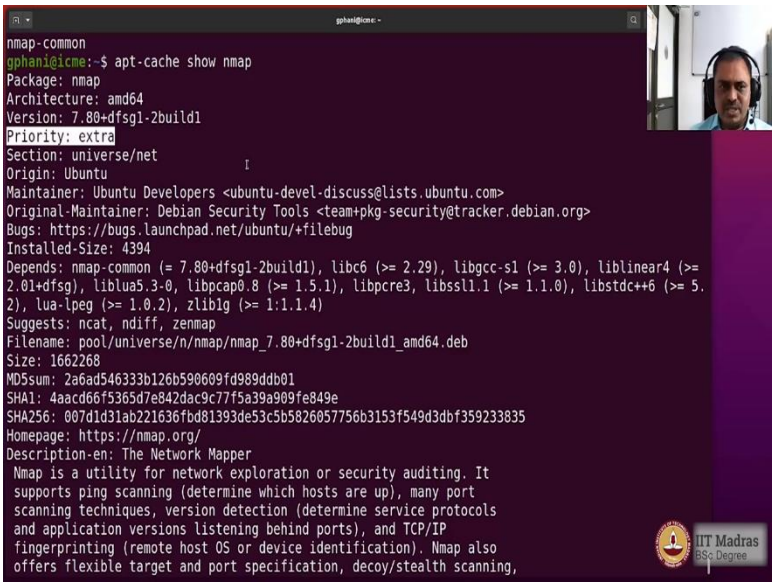


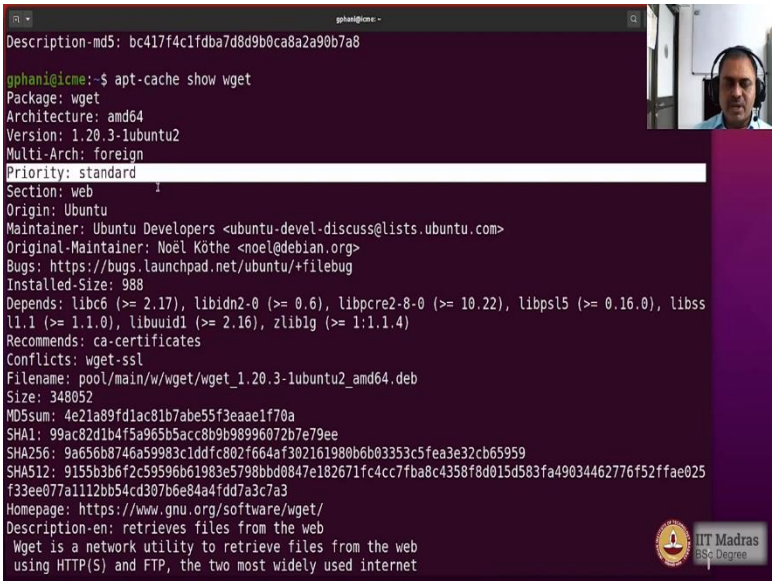
So, each package also lists what kind of a priority that particular package is, and top was priority is required, which means, it is essential to proper functioning of the system. So, you should not remove that package unless you know what you are doing. And the second level of priority is important where it provides functionality, which enables a system to run smoothly. It is a good idea to leave packages that are categorized as important also on the system.

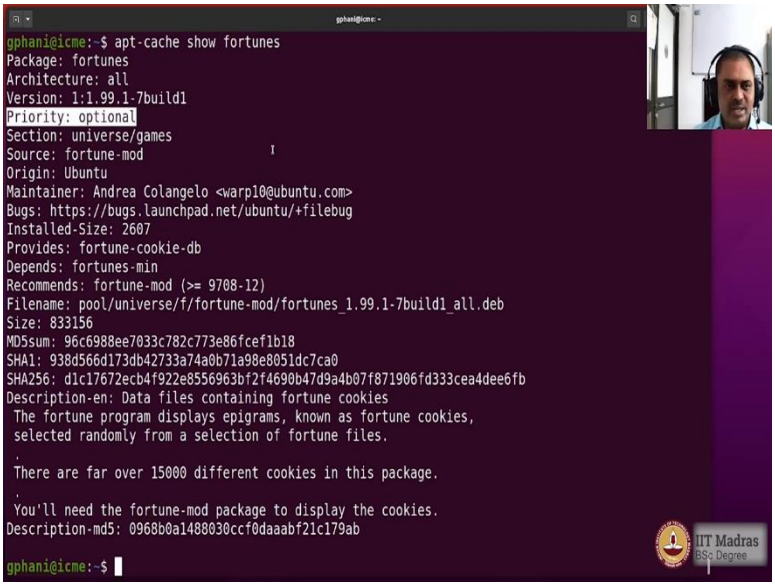
The third level of priority is standard, which means that these are the packages which came along with the standard system installation, but typically, if you have not done any specific customization to the installation then all operating systems of that particular version would contain those packages, which are at the level of priority called standard or higher than that namely important required.

And then there are some packages, which are at the level of priority called optional, which, if you are running short of the storage, hard disk space is not enough, you could actually skip installing them or you can also delete them if you do not need them. And then there are some packages, which are called extra, so one has to pay a little attention to installing these software, because these are known to have certain specialized required requirements or could have conflict with packages, which are actually of higher priority, so you should only install them if you really need them, and in the process do not actually change the versions of the packages, which are actually at a higher priority. So, one has to pay attention by installing these packages under the category extra. So, let us see some packages, and identify where this particular piece of information is located.

(Refer Slide Time: 16:42)




gphani@icme: ~
Description-md5: bc417f4c1fdba7d8d9b0ca8a2a90b7a8
gphani@icme:~$ apt-cache show wget
Package: wget
Architecture: amd64
Version: 1.20.3-1ubuntu2
Multi-Arch: foreign
Priority: standard
Section: web
Origin: Ubuntu
Maintainer: Ubuntu Developers <ubuntu-devel-discuss@lists.ubuntu.com>
Original-Maintainer: Noël Köthe <noel@debian.org>
Bugs: https://bugs.launchpad.net/ubuntu/+filebug
Installed-Size: 988
Depends: libc6 (>= 2.17), libidn2-0 (>= 0.6), libpcre2-8-0 (>= 10.22), libpsl5 (>= 0.16.0), libss
l1.1 (>= 1.1.0), libuuid1 (>= 2.16), zlib1g (>= 1:1.1.4)
Recommends: ca-certificates
Conflicts: wget-ssl
Filename: pool/main/w/wget/wget_1.20.3-1ubuntu2_amd64.deb
Size: 348052
MD5sum: 4e21a89fd1ac81b7abe55f3eaae1f70a
SHA1: 99ac82d1b4f5a965b5acc8b9b98996072b7e79ee
SHA256: 9a656b8746a59983c1ddfc802f664af302161980b6b03353c5fea3e32cb65959
SHA512: 9155b3b6f2c59596b61983e5798bbd0847e182671fc4cc7fba8c4358f8d015d583fa49034462776f52ffae025
f33ee077a1112bb54cd307b6e84a4fdd7a3c7a3
Homepage: https://www.gnu.org/software/wget/
Description-en: retrieves files from the web
Wget is a network utility to retrieve files from the web
using HTTP(S) and FTP, the two most widely used internet
IIT Madras
BSc Degree


gphani@icme: ~
gphani@icme:~$ apt-cache show fortunes
Package: fortunes
Architecture: all
Version: 1:1.99.1-7build1
Priority: optional
Section: universe/games
Source: fortune-mod
Origin: Ubuntu
Maintainer: Andrea Colangelo <warp10@ubuntu.com>
Bugs: https://bugs.launchpad.net/ubuntu/+filebug
Installed-Size: 2607
Provides: fortune-cookie-db
Depends: fortunes-min
Recommends: fortune-mod (>= 9708-12)
Filename: pool/universe/f/fortune-mod/fortunes_1.99.1-7build1_all.deb
Size: 833156
MD5sum: 96c6988ee7033c782c773e86fcef1b18
SHA1: 938d566d173db42733a74a0b71a98e8051dc7ca0
SHA256: d1c17672ecb4f922e8556963bf2f4690b47d9a4b07f871906fd333cea4dee6fb
Description-en: Data files containing fortune cookies
The fortune program displays epigrams, known as fortune cookies,
selected randomly from a selection of fortune files.
.
There are far over 15000 different cookies in this package.
.
You'll need the fortune-mod package to display the cookies.
Description-md5: 0968b0a1488030ccf0daaabf21c179ab
gphani@icme:~$
IIT Madras
BSc Degree


gphani@icme: ~
gphani@icme:~$ fortune
I know the disposition of women: when you will, they won't; when
you won't, they set their hearts upon you of their own inclination.
-- Publius Terentius Afer (Terence)
gphani@icme:~$
IIT Madras
BSc Degree

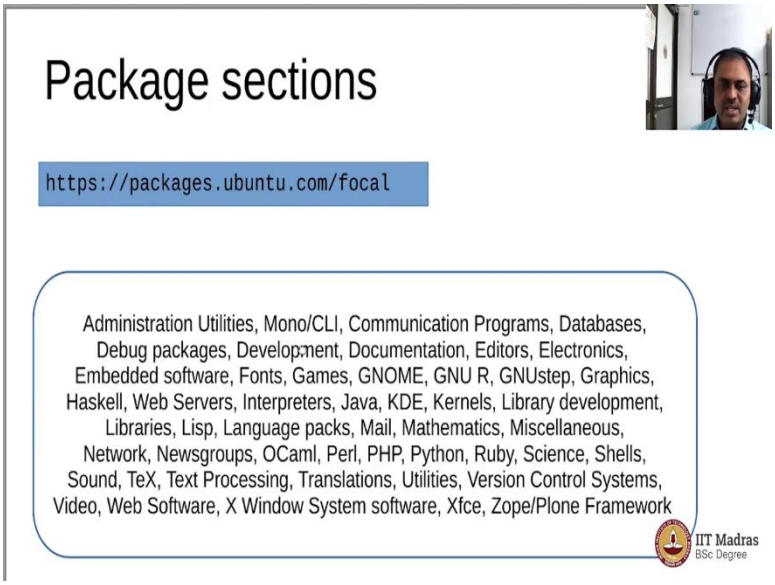
So, the output of the nmap package coming from the apt-cache shows the priority here. So, it says that it is an extra, that means, it is not very important. If you want you can actually remove the package, and that does not hurt your system. So let us look at some other package names and see what would it show.

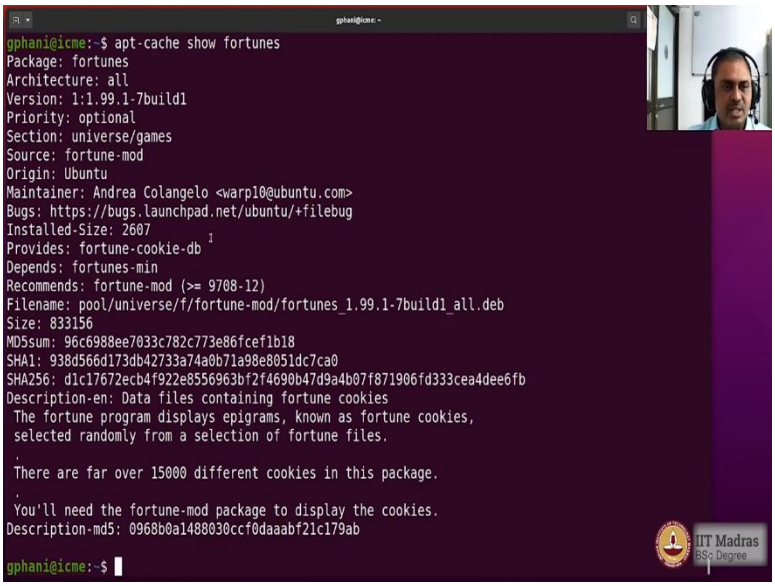
So here is the information about the wget package, and it comes with a priority called standard, which means, that it is at a higher priority than nmap. And it is a good idea to keep the wget command because it is a package that comes along with a standard installation of the operating system.

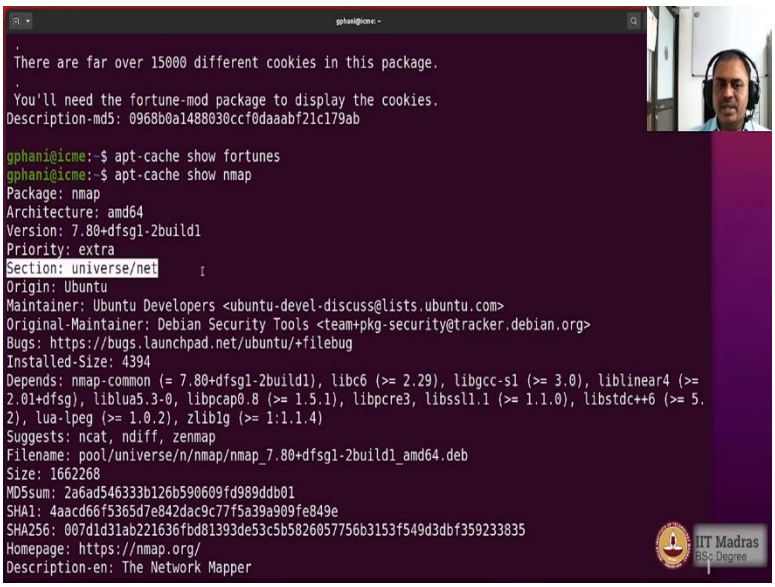
Let us look at the information about the fortune package. So, this package is under the optional category. So, which means that it does not have any conflict with the higher priority packages, so you can install it without having to pay extra attention, but it is an optional package so if you want, you can delete it, in case your hard disk is running out of space.

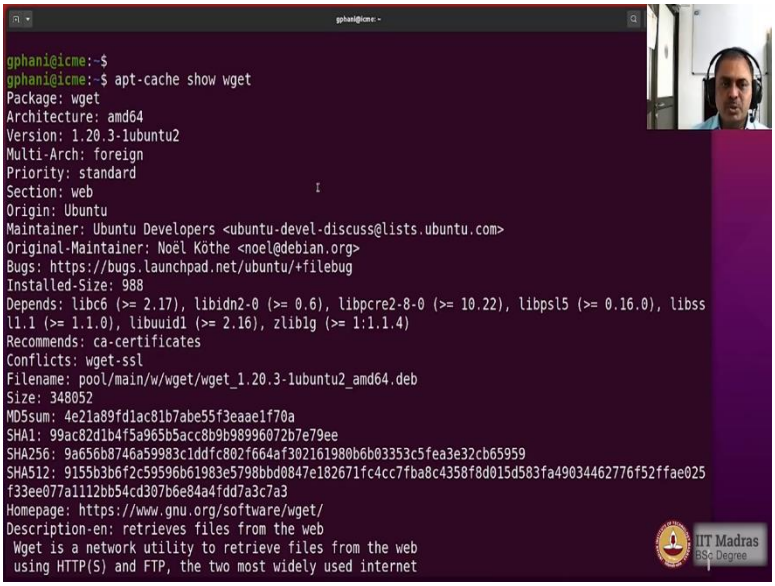
So, what is this command for sure, let us just check it out and it then generally display something funny, some quotation or sometimes it could be even harsh. So do not be offended. It is an entertainment for the programmer at the terminal and it will be different for every time that you type the same command. It is a randomly picked fortune cookie that that you can display onto the screen.

(Refer Slide Time: 18:07)









The Ubuntu packages also come categorized under package sections, and the list of sections will change depending upon the version. So, for the focal Linux, which is Ubuntu 20.04 lts here are the packet sections that are listed as per the website on ubuntu dot com. And you can see that the large number of categories under which the software packages have been grouped. So, we can actually see which package comes under which package section by looking at the output of the apt-cache show command.

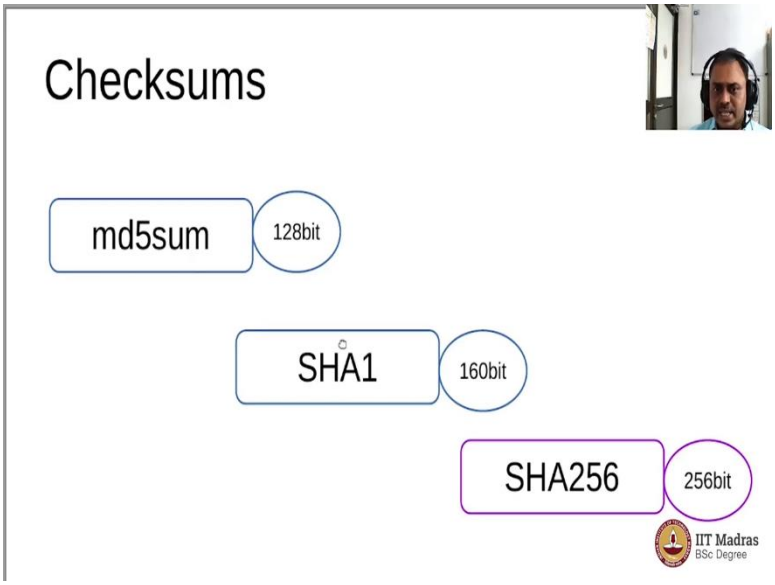
So, let us have a look at that now. So, the fortunes package comes under the section called universe slash games, so it a game so that is a category. You could also look at the category for nmap so it says that it comes under the universe by net so it is coming under the package group called net.

And let us say wget that would be coming under the section called web. So, like that each of these packages comes under a particular section. So sometimes you may want to generally look at one of the packages in a particular section to even explore packages that you may not have heard of.

(Refer Slide Time: 19:28)

So, you can look at the list of package sections and there are some sections you may want to go thoroughly because you work in that particular topic or domain. For example, if you are fascinated by mathematics, then you may want to explore all the packages that are grouped under the package section called mathematics, and see that some of them actually may be quite interesting for you, you may not have come across those packages until you have a looked at the listing here on the ubuntu dot com website.

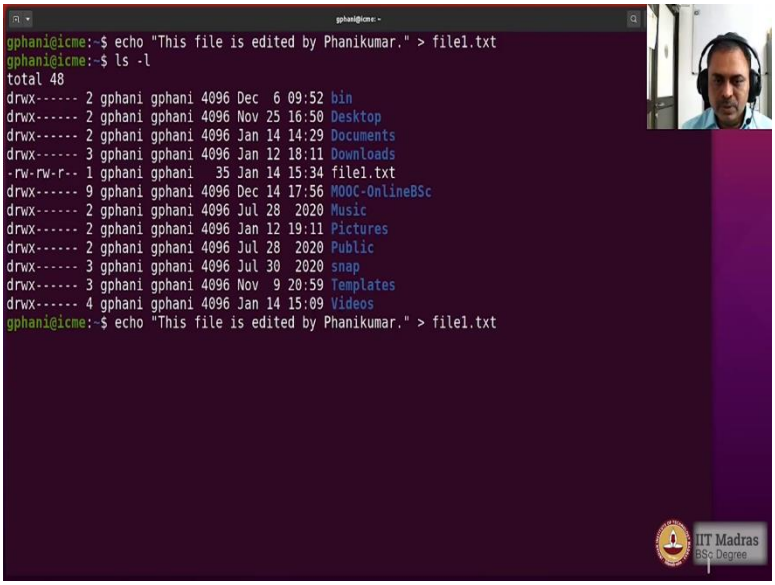
(Refer Slide Time: 20:00)

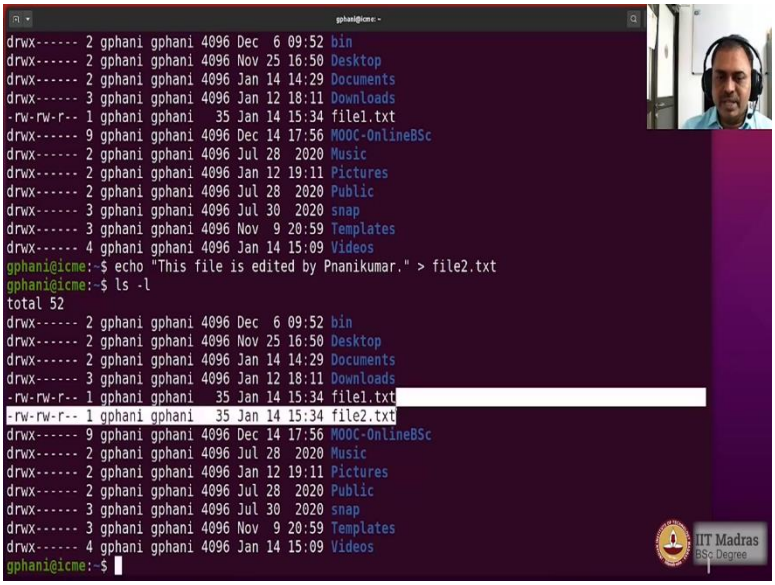


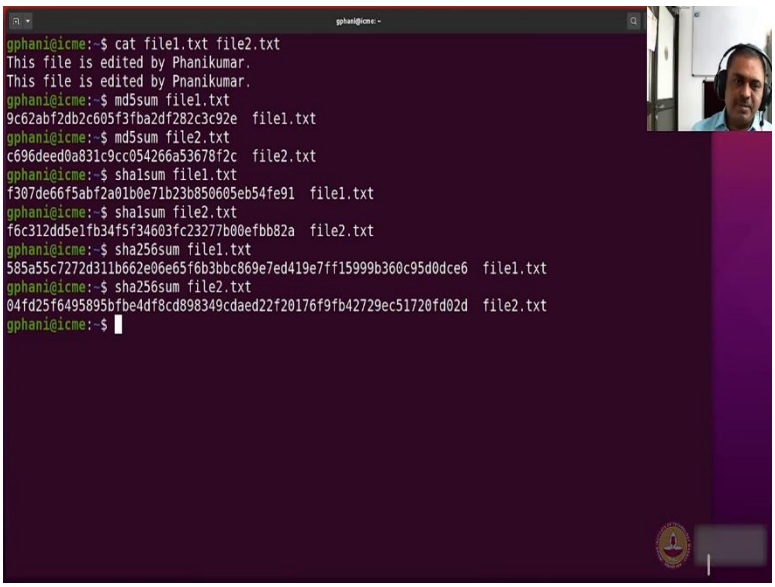
Now, how do we know that the package that we are installing on our operating system is an authentic one. So, the way to verify is by looking at what is called a checksum. So, there are three major ways by which the checksums are computed. The first one is md5sum, which gives you a 128-bit string, irrespective of the size of the input file and this would be very different even if there is a small change in the input string.

And then there is a SHA1 checksum, which is 160-bit string and then, the most advanced that is being used today is the SHA256 checksum, which is a 256-bit string. So, one can actually compare the check sums of any package with what is actually listed on the ubuntu dot com website to see whether the package that has been installed or being installed is authentic or not. In fact, there are more ways by which we can verify the authenticity, but this is a very good opportunity for us to see what do these checksums mean.

(Refer Slide Time: 21:24)







So, let us explore that in a moment, let us explore the concept of checksums by using it on some files that we can create right on the command line and we would use a simple string, as the only content of the file to illustrate how the checksums can be very different for a small change done to the content of the file.

So, we would echo a string to a file, so let us do that. This file is edited by, and then I give my name and then I redirect the string to a file, file1 dot t txt and let me look at the file here. So, file1 dot txt has 35 bytes. Now, what I do is that I make a small change to this string, so what I would do is, instead of h I put n, and then write to another file called file2 dot txt. And now if you look at the files, the two files, file1 dot txt and file2 dot txt are identical in their size.

And if we look at the contents, cat file1 dot txt and file2 dot txt, you could see that if you do not pay attention they look identical. Except for the h end and n they look identical. But when we run the md5sum or any other sum that we have just not mentioned, you would see what is the difference.

So, we now run md5sum of the file1 dot txt and you see that you would get 128-bit string and then we run it also for file2 dot txt and you will see that the two strings are very different. So, it actually shows you that for a very small change in the bit sequence of the input stream the output checksum is very different. And this is one way by which we can actually identify if the original file has been tampered or not.

And there are other ways of evaluating the checksum, so let us look at those also. So, sha1sum file1 dot txt and sha1sum file2 dot txt. You could also see that the 160-bit string is very different between the two files and then you also have sha256sum file1 dot txt and sha256sum file2 dot txt.

So, this would be a 256-bit string, which would also be very different if the input string is different or input stream file stream is different even in just a few bits. So, these are the three major mechanisms by which the checksums are compared to ensure that the file that is being installed is authentic and the same as what has been supplied by the original vendor of the software. And one has to pay attention to these while copying and to ensure that in the process of downloading, nothing has gone wrong to the contents of the file.