How To Make The Best Use Of Live Sessions

- Please login on time
- Please do a check on your network connection and audio before the class to have a smooth session
- All participants will be on mute, by default. You will be unmuted when requested or as needed
- Please use the “Questions” panel on your webinar tool to interact with the instructor at any point during the class
- Ask and answer questions to make your learning interactive
- Please have the support phone number (US : 1855 818 0063 (toll free), India : +91 90191 17772) and raise tickets from LMS in case of any issues with the tool
- Most often logging off or rejoining will help solve the tool related issues

COURSE OUTLINE

MODULE 4

INTRODUCTION TO LINUX

INSTALLATION AND INITIALISATION

USER ADMINISTRATION

BOOT AND PACKAGE MANAGEMENT

NETWORKING

LINUX OVERVIEW AND SCRIPTING

LINUX FOR SOFTWARE DEVELOPMENT

SECURITY ADMINISTRATION

Objectives

After completing this module, you should be able to:

- Understand Boot Management System
- Configure services to run at boot
- Perform Package Management – installing and removing Packages
- Verify dependencies on packages and resolve them
- Understand kernel configuration
- Shut down the system

edureka!

Boot And Package Management

Kernel Configuration

edureka!

/proc

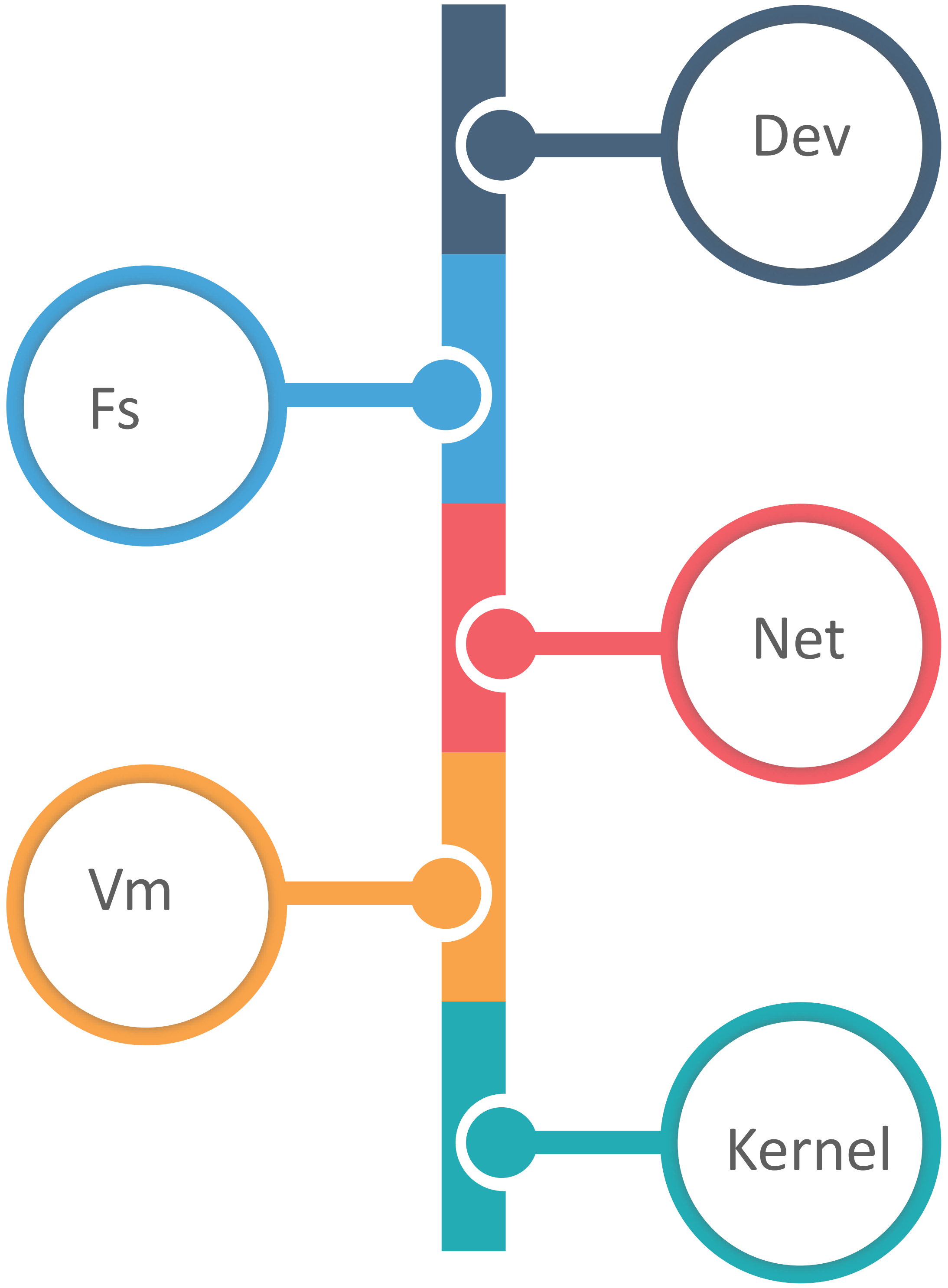
In Linux, the directory “/proc” represents the default method for handling kernel and other memory related information to handle the system

“/proc/sys” is where one can find information about the device, driver and kernel features

Before changing any parameter in /proc, one should understand the usage of it

/proc

The structure varies from kernel to kernel but generally has these directories –



Dev – specific devices connected to system.

Fs – filesystem configuration.

Net – network related configuration.

Vm – kernel’s virtual memory usage.

Kernel – kernel-specific configuration.

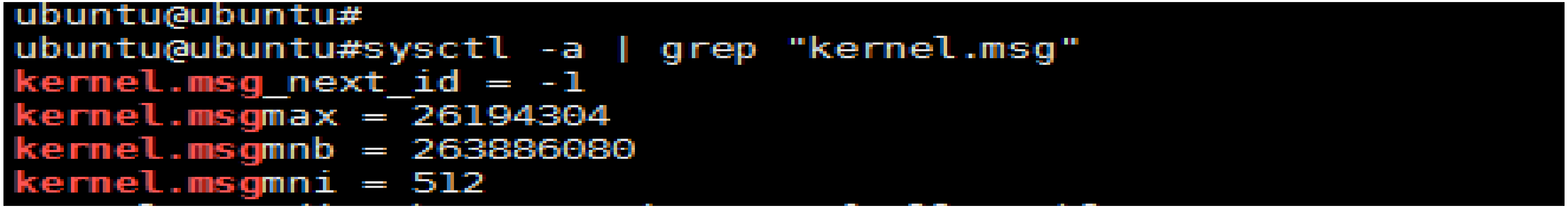
sysctl

- The file “/etc/sysctl.conf” contains a list of kernel parameters
- We can manually edit this file or use system commands to change it
- The commands to be used from CLI -

```
# sysctl –a                                : list down all these configurations.

# sysctl <parameter_name>                  : to view a particular parameter

# sysctl –w <paramater_name and value>     : to modify a parameter value

# sysctl –p                                : to ensure modified value persist after a reboot
```

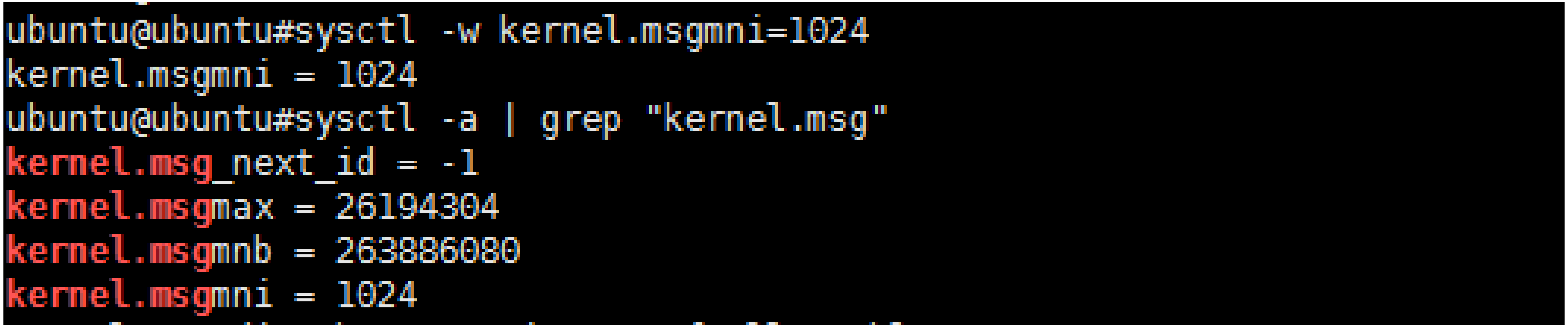
DEMO – sysctl

Demo - sysctl



```
ubuntu@ubuntu#
ubuntu@ubuntu#sysctl -a | grep "kernel.msg"
kernel.msg_next_id = -1
kernel.msgmax = 26194304
kernel.msgmnb = 263886080
kernel.msgmni = 512
```

Now, we will modify one the parameters



```
ubuntu@ubuntu#sysctl -w kernel.msgmni=1024
kernel.msgmni = 1024
ubuntu@ubuntu#sysctl -a | grep "kernel.msg"
kernel.msg_next_id = -1
kernel.msgmax = 26194304
kernel.msgmnb = 263886080
kernel.msgmni = 1024
```

Run A Script During Boot

Step-1
“/etc/init.d/” : the path you should put your script to be executed at boot

Step-2
Sudo needs to be used for this or the user access permission needs to be elevated

Step-3
Add “#!/bin/sh” at the top of file to let OS know about the shell to be used. If started by some other shell, it may not be able to recognize few commands

Step-4
cp /etc/init.d/skeleton /etc/init.d/your_app : get a template and modify it to your usage, provided by the Linux

Step-5
chmod 755 <filename> : to make the file have executable permissions

Step-6
Now, reboot the system

Boot Management

edureka!

Safe Mode

01 In safe mode, only essential system services are allowed to boot

02 It is primarily used to fix the issues during startup or with OS

03 Some newly running program may have infected the OS and running in safe-mode doesn't start those process helping us to resolve the issue

04 If system restore is enabled, we can trigger it in safe mode to restore to a point where a normal boot-up was working fine

05 The recovery mode in Ubuntu is safe mode while Windows uses the same name

Single User Mode

01 In single user mode a multi-user OS boots as a single super-user.

02 It doesn't have a graphical interface with a bare minimum daemon running and only command line interface supported for the user.

03 It is primarily used to maintain server networks.

04 It is required to have exclusive access to otherwise shared resource in a multi-user system.

05 In Linux, run-level 1 boots into a single user mode.

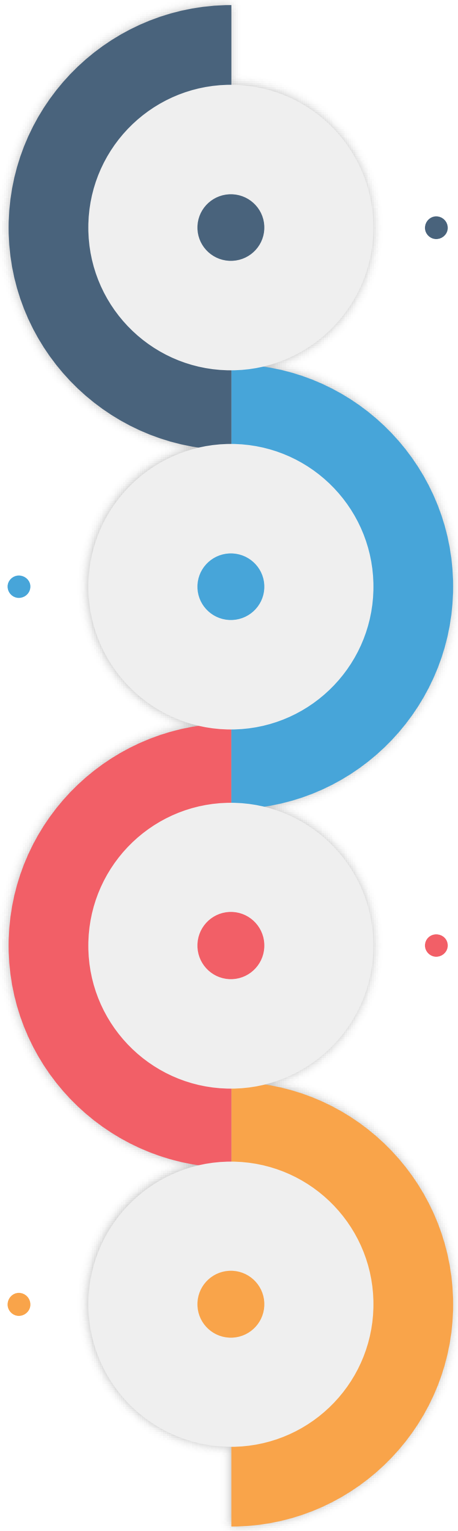
Protecting Single User Mode

- Allowing single user mode may lead to multiple security risks
- One should secure it with root password before booting
- Append the following line in /etc/inittab

```
su:S:wait:/sbin/sulogin
```

Shutdown

Why shutdown is required?



Maintenance or OS upgrade

Hardware changes

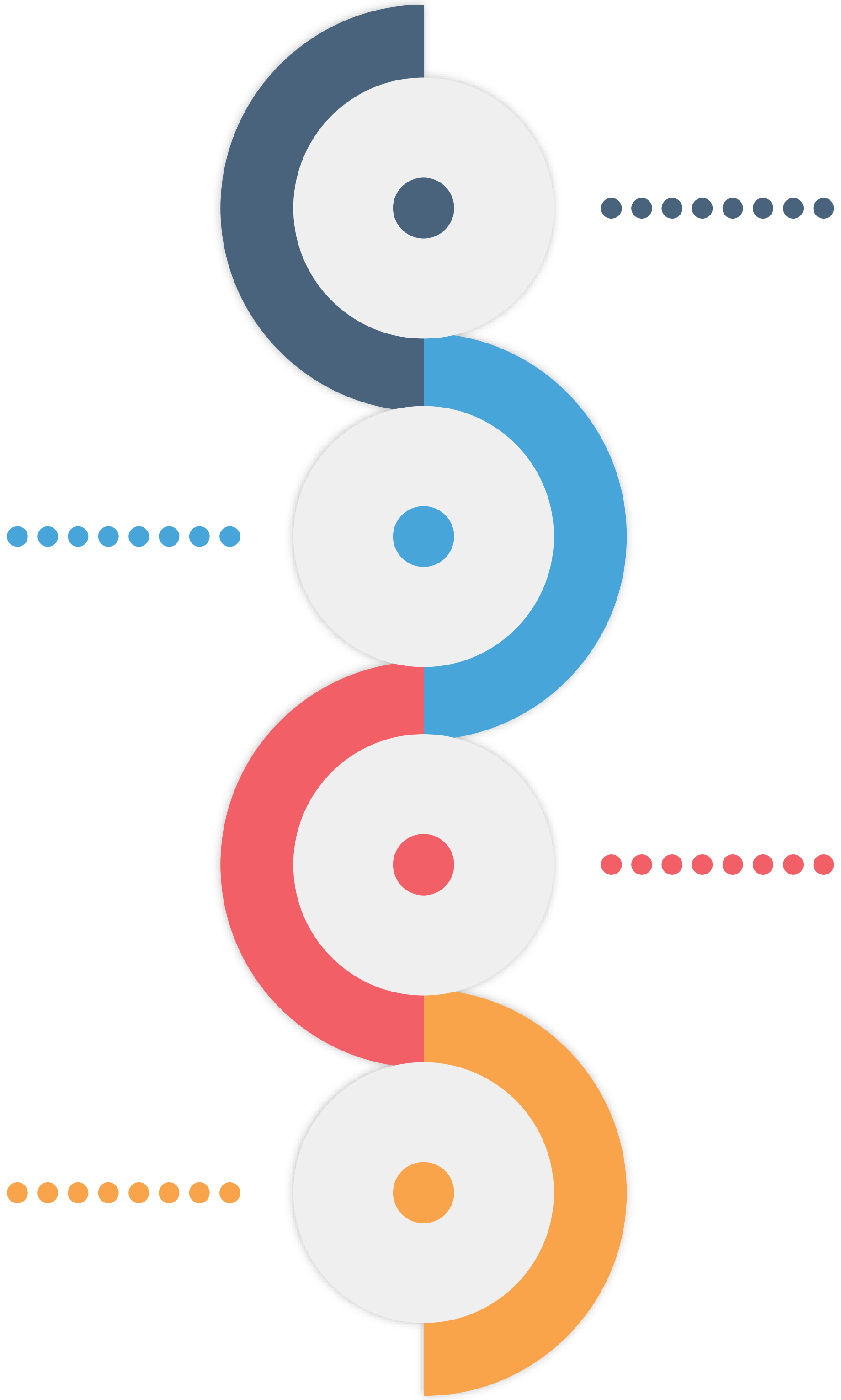
System not responding

Running diagnostics or Performance testing

edureka!

Shutdown

Ways to shutdown



Turn off power

Use shutdown command/button from GUI

Send a term signal

Telinit to change run-levels

System Shutdown Steps

01 All users are notified , with some reasonable warning.

02 All running process are sent signal to exit gracefully.

03 All subsystem are shutdown.

04 File System integrity is maintained.

05 The run-levels are changed and process is halted.

System Commands

```
# service <service_name> stop  : to stop a service

# sevice <service_name> start        : to start a service

# reboot                             : to reboot the system

# ps –ef | grep <service_name>       : to get process-id for service

# kill -9 <process-id>               : to kill a particular process with its process Id

# ssh <user_name>@<server_ip         : to login in a particular server with user_name

# service <service_name> status      : to check the current status of the service
```

Grub Bootloader Configurations

01 The configuration file of grub is located at /boot/grub/grub.cfg

02 Running update-grub commands create it, so not recommended to change this

03 Grub Settings are stored in /etc/default/grub

04 Scripts are located at /etc/grub.d for various functionalities

05 Run the update-grub command every time you make a change to either of the two above mentioned locations.

Grub Settings File

One can open the /etc/default/grub with a text editor

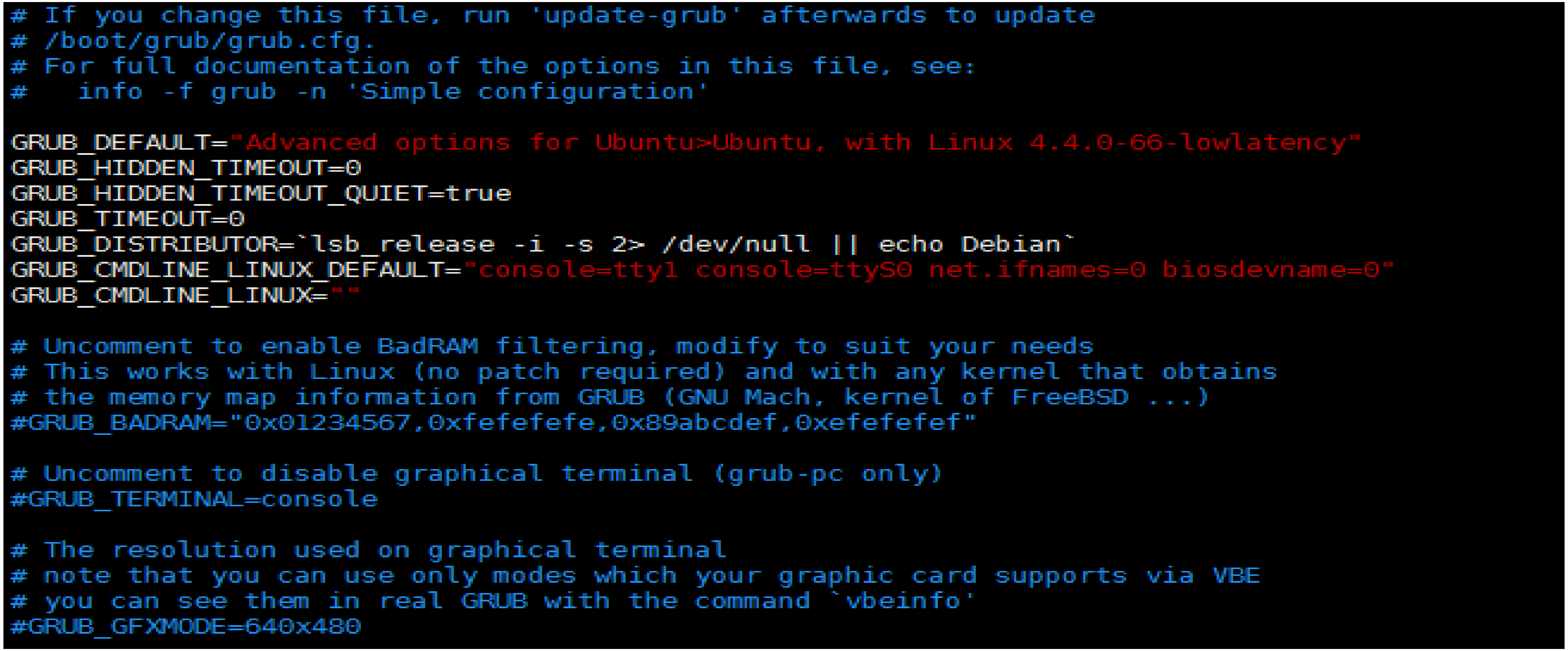
GRUB_DEFAULT :can provide 0..n as integer to load OS at that entry. Can provide the values as saved which will load the last chosen OS as default.

GRUB_TIMEOUT :Number of seconds to wait for keyboard entry before booting.

GRUB_HIDDEN_TIMEOUT :This notifies that grub will be hidden and automatically boot default OS.

GRUB_BACKGROUND :This specifies the background for the grub instead of monochrome screen.

Demo - Bootloader Configurations



```
# If you change this file, run 'update-grub' afterwards to update
# /boot/grub/grub.cfg.
# For full documentation of the options in this file, see:
#   info -f grub -n 'Simple configuration'

GRUB_DEFAULT="Advanced options for Ubuntu>Ubuntu, with Linux 4.4.0-66-lowlatency"
GRUB_HIDDEN_TIMEOUT=0
GRUB_HIDDEN_TIMEOUT_QUIET=true
GRUB_TIMEOUT=0
GRUB_DISTRIBUTOR=`lsb_release -i -s 2> /dev/null || echo Debian`
GRUB_CMDLINE_LINUX_DEFAULT="console=tty1 console=ttyS0 net.ifnames=0 biosdevname=0"
GRUB_CMDLINE_LINUX=""

# Uncomment to enable BadRAM filtering, modify to suit your needs
# This works with Linux (no patch required) and with any kernel that obtains
# the memory map information from GRUB (GNU Mach, kernel of FreeBSD ...)
#GRUB_BADRAM="0x01234567,0xfefefefe,0x89abcdef,0xefefefef"

# Uncomment to disable graphical terminal (grub-pc only)
#GRUB_TERMINAL=console

# The resolution used on graphical terminal
# note that you can use only modes which your graphic card supports via VBE
# you can see them in real GRUB with the command `vbeinfo'
#GRUB_GFXMODE=640x480
```

RPM

RPM - Red Hat Package Manager

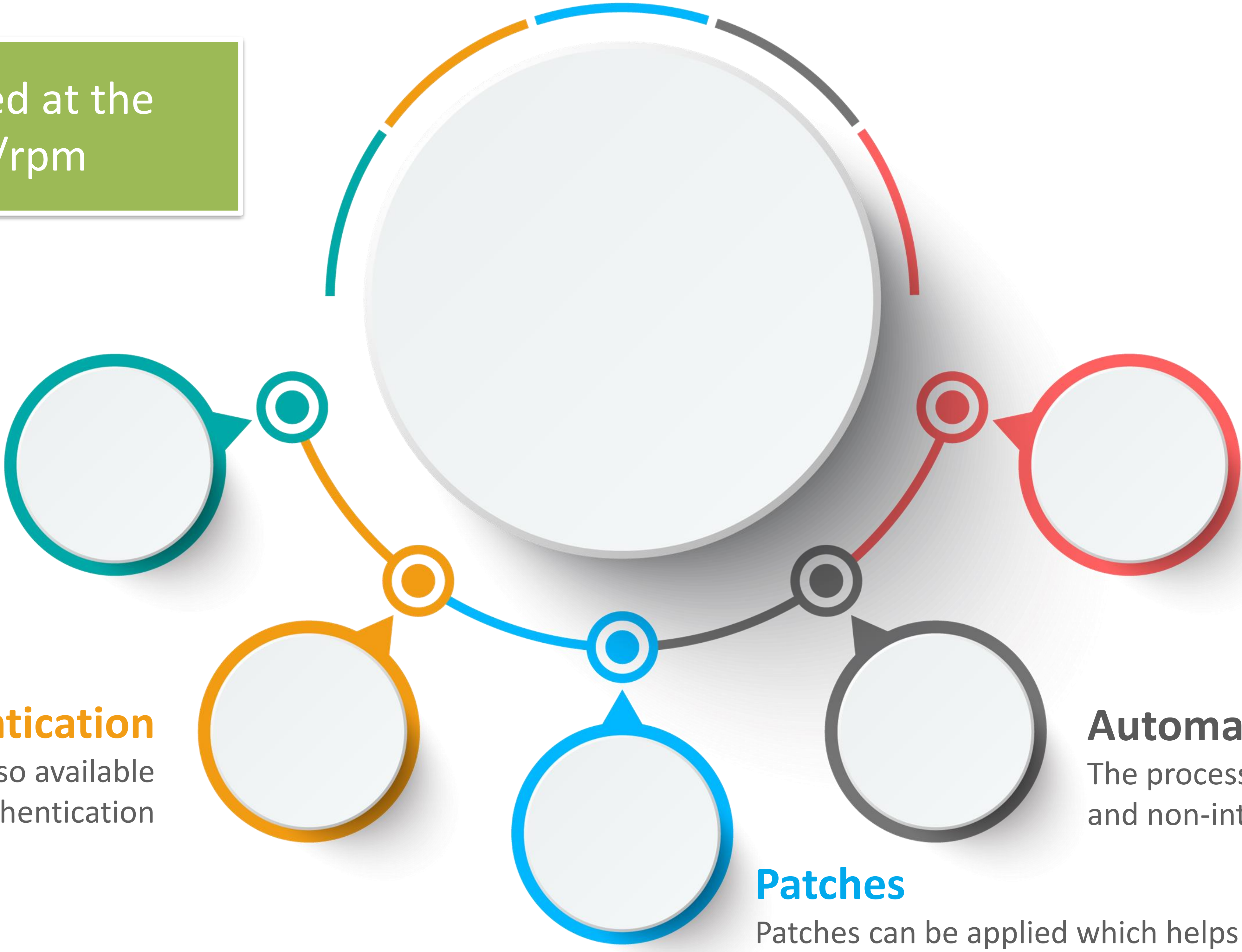
- The files used by this program have an extension .rpm
- RPM was originally created in 1997
- RPM is free and released under GPL
- It is default packaging tool for RHEL, CentOS, Fedora, etc.

Red Hat Package Manager

To install and manage softwares packages in Linux.

RPM Features

The package is stored at the location /var/lib/rpm



Crypto
The packages in rpm can be verified cryptographically by md5 and GPG

Authentication
Source archive is also available which helps in authentication

Patches
Patches can be applied which helps in updating process faster and easier

Automated
The process is automated and non-interactive

Installation time verification
Done for dependency

Basic Tasks For RPM

The primary features of this package management tool are:

Installing	: to install a particular package.
Updating	: update the existing package.
Uninstalling	: remove the currently installed package.
Query	: get information about the package.
Authentication	: verify the package for security reasons.

Finding RPM Packages

- Red Hat developed RPM package can be found at
  - Red Hat Enterprise Linux CD ROMs
  - Red Hat Network
  - Red Hat errata page having list of packages
- It can be found on the Internet. Some of the websites are:
  - http://rpmfind.net
  - http://www.redhat.com
  - http://rpm.pbone.net

Installing

- Login as root or get elevated permissions for the user
- Options
  - -i : install a package
  - -v : verbose for a nicer display
  - -h : print hash marks as the package archive is unpacked.

Syntax

```
rpm -i <options> <package_name>
```

Example : # rpm –ivh MySQL-client-5.5.30-1.e16.x86_64.rpm

edureka!

Check RPM Signature

- Check the PGP signature before installing any package.
- If integrity and origin is OK then one can go ahead and install that package.

Syntax

```
rpm - -checksig <package_name>
```

Example :

```
# rpm - -checksig MySQL-client-5.5.30-1.e16.x86_64.rpm
```

edureka!

Check Dependency Of RPM Package

- Check the dependency of the package
- Options
  - -q : Query a package
  - -p : List capabilities the package provides
  - -R: List capabilities on which this package depends
- To ignore these dependencies use '–nodeps' before installing package

Syntax

```
rpm –q <options> <package name>
```

Example :

```
# rpm -qpR MySQL-client-5.5.30-1.e16.x86_64.rpm
```

edureka!

Check An Installed Package

- One can check if a particular package is already installed or not
- To view files of this installed package add –l option
  - # rpm –ql MySQL

Syntax

rpm –q <package_name>

Example : # rpm –q MySQL

View Installed RPM Packages

- One can list all the recently installed rpm packages.
- One can shorten the list to check for recently installed ones by adding –last
  - # rpm –qa - -last

Syntax

rpm –qa

Upgrade a RPM Package

One can upgrade a rpm package based on requirements:

Syntax

rpm –U <option> <package_name>

Example : # rpm –Uvh MySQL-client-5.5.30-1.e16.x86_64.rpm

Remove RPM Package

- To remove a package use ‘-e’ option.
- In case you don’t want to remove the dependent packages use ‘—nodeps’ option.
  - # rpm –e - -nodeps MySQL-client-5.5.30-1.e16.x86_64.rpm

Syntax

```
rpm –e <option> <package_name>
```

Example : # rpm –e MySQL-client-5.5.30-1.e16.x86_64.rpm

Query RPM Packages

To find the package to which a particular file belongs to use ‘-qf’

Syntax

```
rpm –qf <file_name>
```

```
Example : # rpm –qf passwd
```

Query RPM Packages

To find details about a particular installed package

Syntax

rpm –qi <package_name>

Example : # rpm –qi MySQL

Verify RPM Package

- To verify a package, use ‘-Vp’ option

Syntax

```
rpm –Vp <package_name>
```

Example : # rpm –Vp MySQL-client-5.5.30-1.e16.x86_64.rpm

- To verify all rpm packages, use the following command:

Syntax

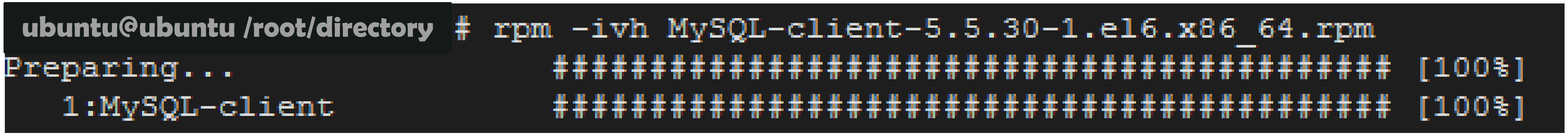
```
Syntax : rpm –Va
```

DEMO - RPM

edureka!

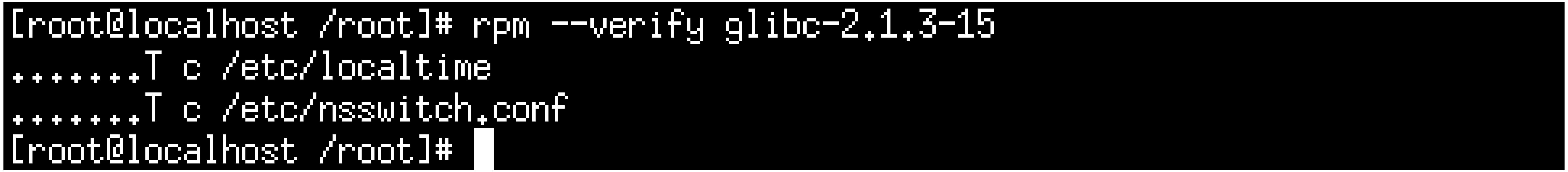
Demo - RPM

- Installing a new package



```
ubuntu@ubuntu /root/directory # rpm -ivh MySQL-client-5.5.30-1.el6.x86_64.rpm
Preparing...                ########################################### [100%]
   1:MySQL-client           ########################################### [100%]
```

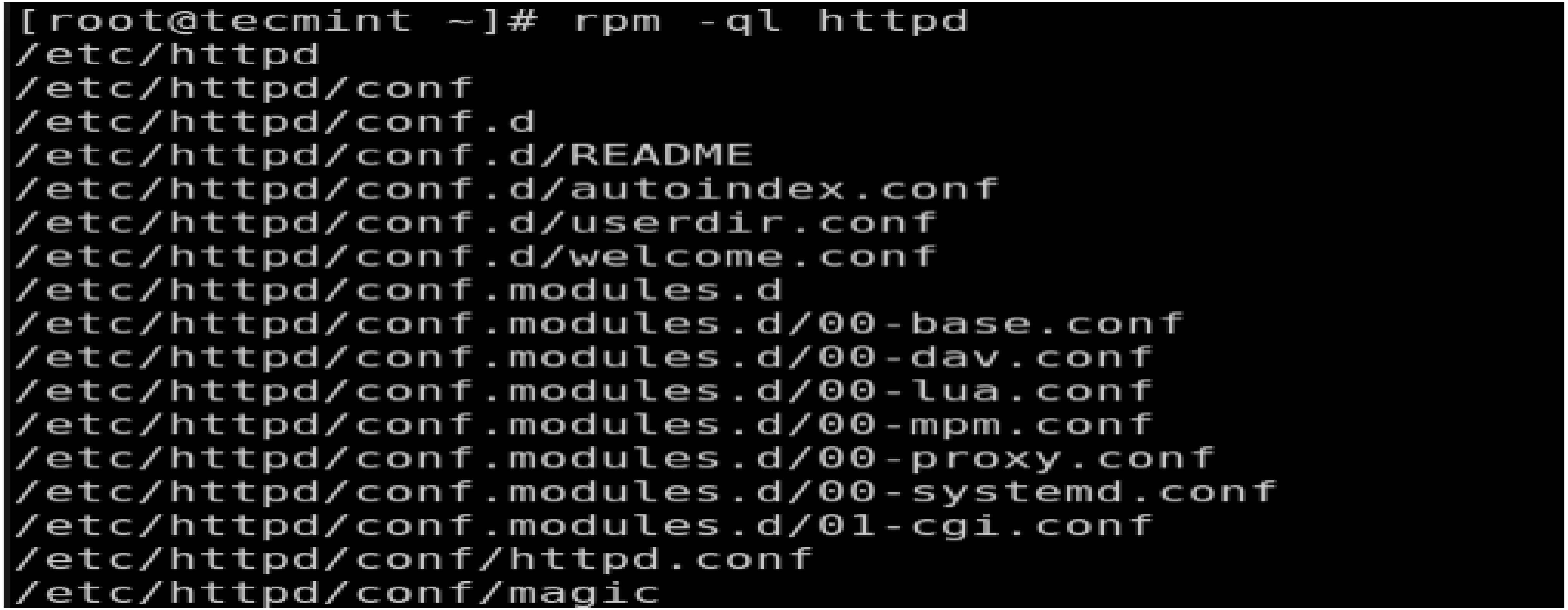
- Verifying a package



```
[root@localhost /root]# rpm --verify glibc-2.1.3-15
.......T c /etc/localtime
.......T c /etc/nsswitch.conf
[root@localhost /root]# 
```

Demo - RPM

- Query a particular package



```
[root@tecmint ~]# rpm -ql httpd
/etc/httpd
/etc/httpd/conf
/etc/httpd/conf.d
/etc/httpd/conf.d/README
/etc/httpd/conf.d/autoindex.conf
/etc/httpd/conf.d/userdir.conf
/etc/httpd/conf.d/welcome.conf
/etc/httpd/conf.modules.d
/etc/httpd/conf.modules.d/00-base.conf
/etc/httpd/conf.modules.d/00-dav.conf
/etc/httpd/conf.modules.d/00-lua.conf
/etc/httpd/conf.modules.d/00-mpm.conf
/etc/httpd/conf.modules.d/00-proxy.conf
/etc/httpd/conf.modules.d/00-systemd.conf
/etc/httpd/conf.modules.d/01-cgi.conf
/etc/httpd/conf/httpd.conf
/etc/httpd/conf/magic
```

YUM

edureka!

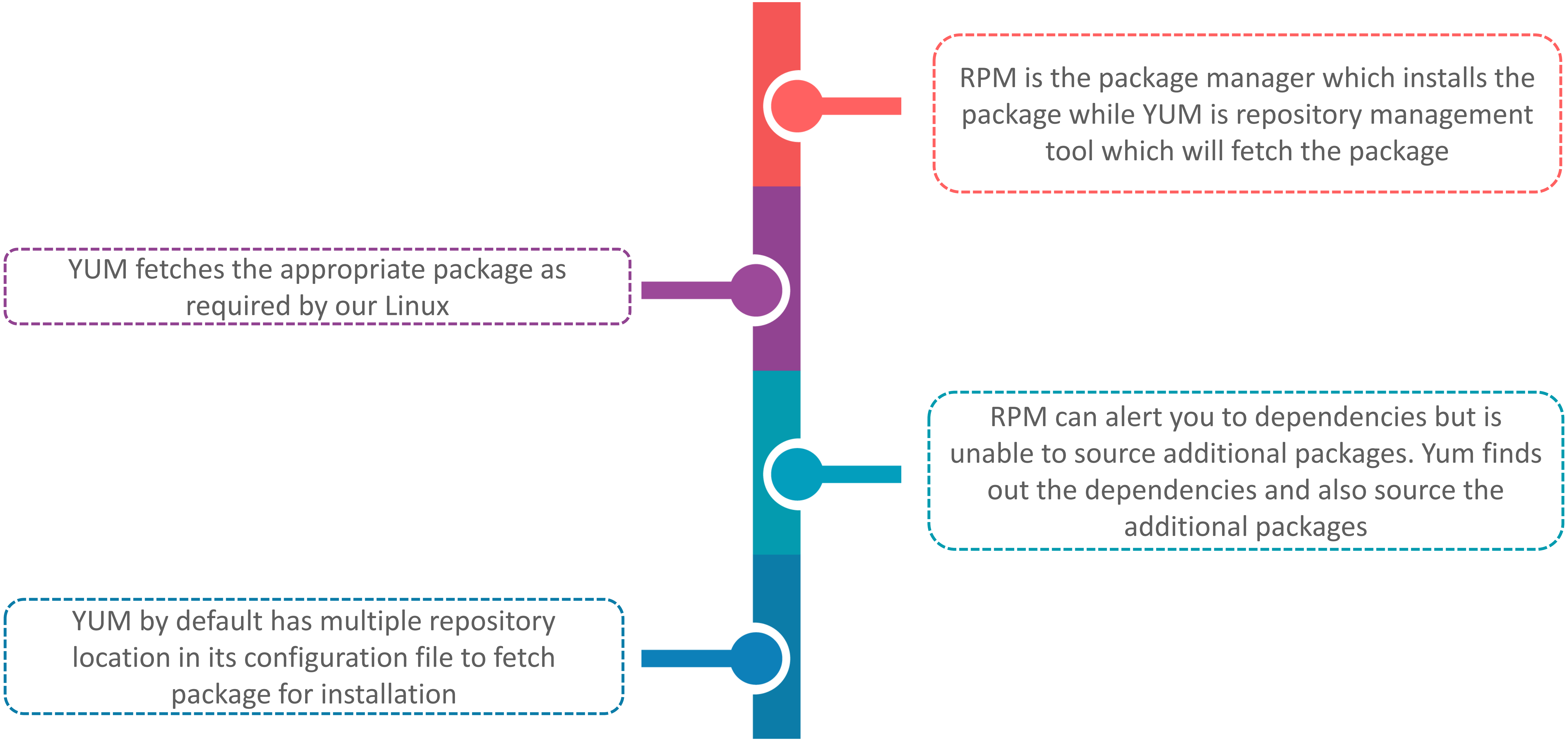
YUM

- YUM was created in 2003 and is the primary choice for RPM based distros.
- Installing and updating of packages are simpler.
- Software dependencies are taken care of and installed along with it.
- Yum is primarily in command line interface but GUI based wrappers also exist.
- It is the official package manager for Red Hat and CentOS.

YUM (YellowDog, Updater, Modifier)

Package management which is interactive and based on rpm

RPM and YUM



RPM is the package manager which installs the package while YUM is repository management tool which will fetch the package

YUM fetches the appropriate package as required by our Linux

RPM can alert you to dependencies but is unable to source additional packages. Yum finds out the dependencies and also source the additional packages

YUM by default has multiple repository location in its configuration file to fetch package for installation

Install using YUM

- Use command ‘install’ to install a package using YUM

Syntax

```
yum install <package_name>
```

Command will show dependencies and will ask for confirmation.

Example : # yum install python

Syntax

```
yum –y install <package_name>
```

To suppress confirmation and install software automatically use ‘-y’.

Example : # yum –y install python

Remove Package

- Use command ‘remove’ to remove package from the system

Syntax

```
yum remove <package_name>
```

Command will ask for a confirmation to remove the package.

Example : # yum remove python

Syntax

```
yum –y remove <package_name>
```

To suppress confirmation and remove program automatically use ‘-y’.

Example : # yum –y remove python

List Package

- Use Command ‘list’ to find a specific package

Syntax

yum list <package_name>

Example : # yum list openssh

To list all available package installed use keyword “installed”

```
# yum list installed
```

To list all available package in YUM database don’t mention the package name

```
# yum list
```

Search Package

- Use command 'search' to find all available package to match the name of package specified.

Syntax

yum search <package_name>

Example : # yum search openssh

Syntax

yum provides <filename>

To find which package provides a particular file, use command 'provides'

Example : # yum provides resolv.conf

edureka!

Update Package

- Use command ‘update’ to update a package

Syntax

```
yum update <package_name>
```

Example : # yum update openssh

To update the whole system, don’t provide a package name

```
# yum update
```

To check which packages have update available use ‘check-update’

```
# yum check-update
```

Yum Repository Related Options

- To list all enabled repositories in YUM use ‘repolist’.

Syntax

```
# yum repolist
```

To view both enabled and disabled append ‘all’.

```
# yum repolist all
```

To install from a specific repository use ‘- -enablerepo’ option.

```
yum - -enablerepo=<repository_name>
<package_name>
```

Example : # yum -enablerepo=extras install mysql

edureka!

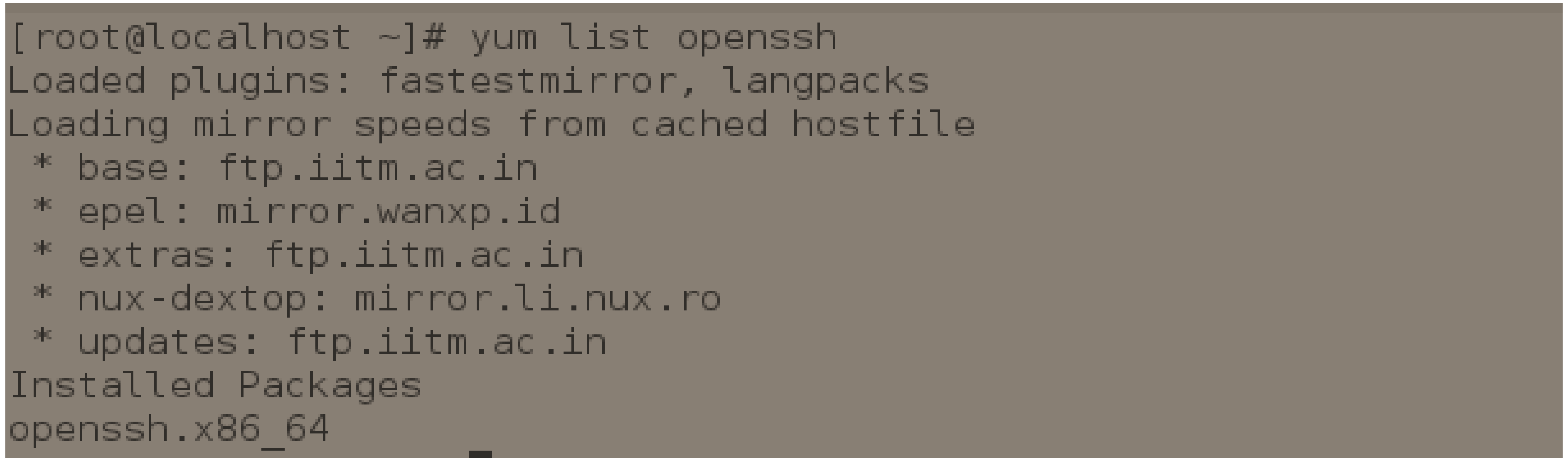
Other YUM Options

history	: to view all past executed YUM commands.
clean all	: removes cached packages and header created to resolve dependencies.
grouplist	: lists YUM groups.
groupinfo	: lists information about the YUM group.
groupinstall	: installs the packages in the YUM group.
groupremove	: to remove installed group from the system.

DEMO - YUM

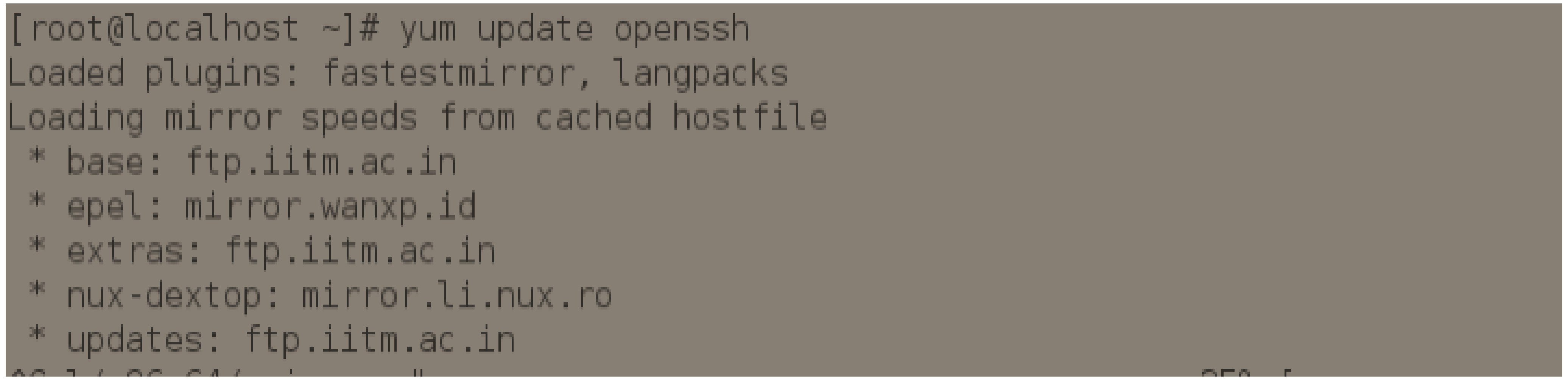
Demo : YUM

- List a package



```
[root@localhost ~]# yum list openssh
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
 * base: ftp.iitm.ac.in
 * epel: mirror.wanxp.id
 * extras: ftp.iitm.ac.in
 * nux-dextop: mirror.li.nux.ro
 * updates: ftp.iitm.ac.in
Installed Packages
openssh.x86_64
```

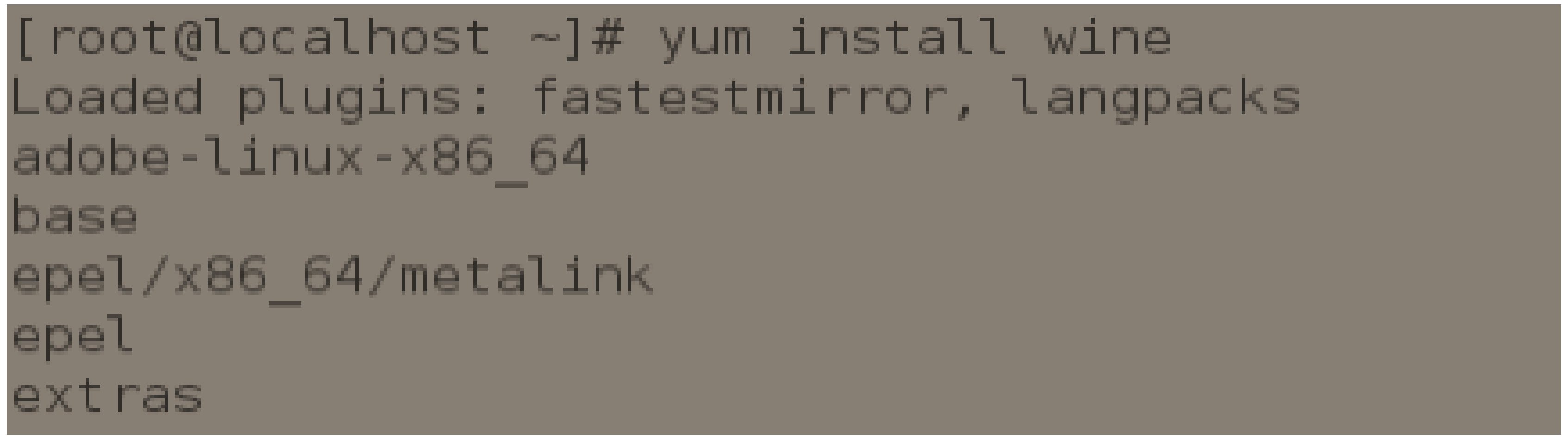
- Update a package



```
[root@localhost ~]# yum update openssh
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
 * base: ftp.iitm.ac.in
 * epel: mirror.wanxp.id
 * extras: ftp.iitm.ac.in
 * nux-dextop: mirror.li.nux.ro
 * updates: ftp.iitm.ac.in
```

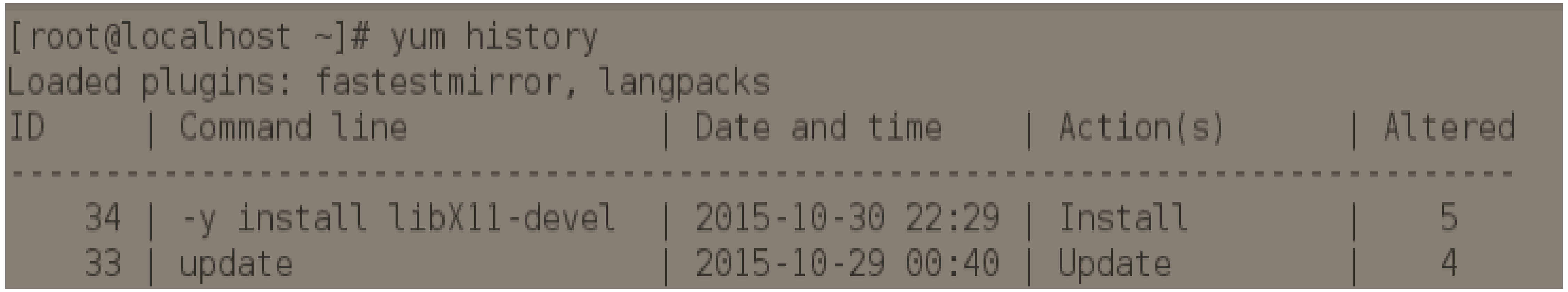
Demo : YUM (continued)

- Install a Package



```
[root@localhost ~]# yum install wine
Loaded plugins: fastestmirror, langpacks
adobe-linux-x86_64
base
epel/x86_64/metalink
epel
extras
```

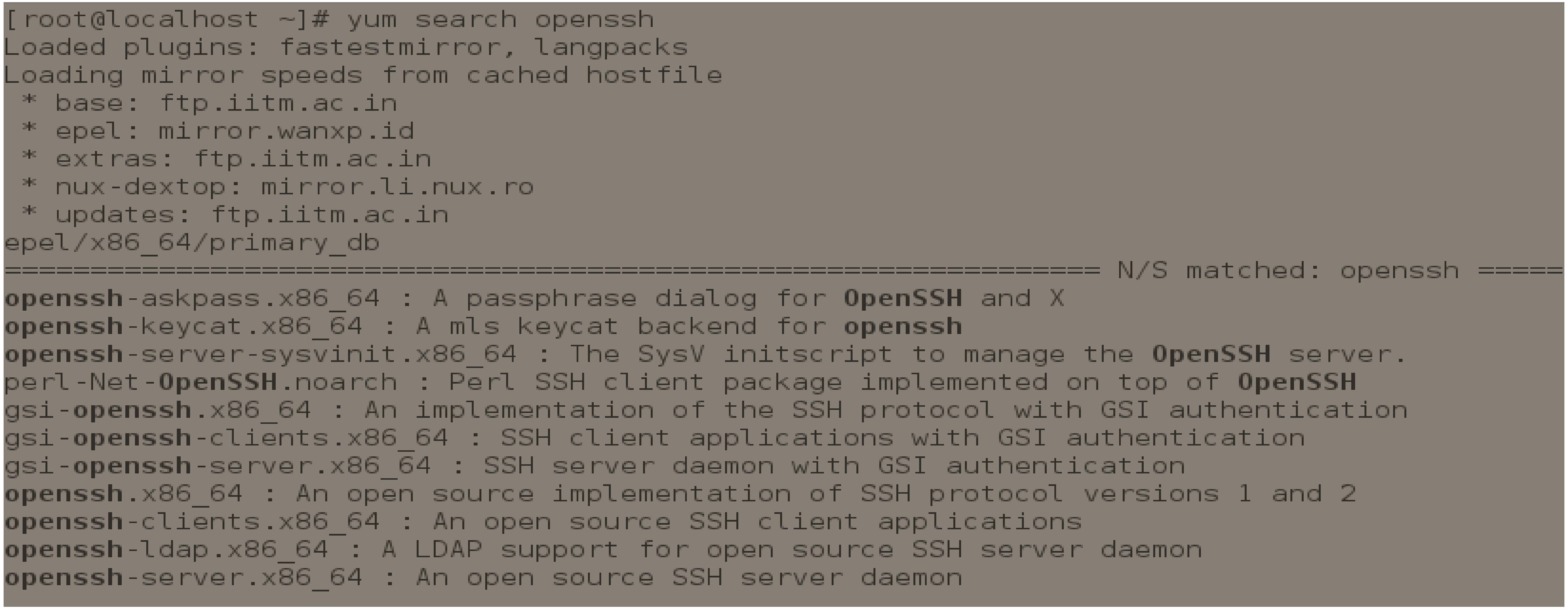
- Check history



```
[root@localhost ~]# yum history
Loaded plugins: fastestmirror, langpacks
ID     | Command line             | Date and time    | Action(s)      | Altered
-------------------------------------------------------------------------------
    34 | -y install libX11-devel  | 2015-10-30 22:29 | Install        |    5
    33 | update                   | 2015-10-29 00:40 | Update         |    4
```

Demo : YUM (continued)

Search a package



```
[root@localhost ~]# yum search openssh
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
 * base: ftp.iitm.ac.in
 * epel: mirror.wanxp.id
 * extras: ftp.iitm.ac.in
 * nux-dextop: mirror.li.nux.ro
 * updates: ftp.iitm.ac.in
epel/x86_64/primary_db
=============================================================== N/S matched: openssh =====
openssh-askpass.x86_64 : A passphrase dialog for OpenSSH and X
openssh-keycat.x86_64 : A mls keycat backend for openssh
openssh-server-sysvinit.x86_64 : The SysV initscript to manage the OpenSSH server.
perl-Net-OpenSSH.noarch : Perl SSH client package implemented on top of OpenSSH
gsi-openssh.x86_64 : An implementation of the SSH protocol with GSI authentication
gsi-openssh-clients.x86_64 : SSH client applications with GSI authentication
gsi-openssh-server.x86_64 : SSH server daemon with GSI authentication
openssh.x86_64 : An open source implementation of SSH protocol versions 1 and 2
openssh-clients.x86_64 : An open source SSH client applications
openssh-ldap.x86_64 : A LDAP support for open source SSH server daemon
openssh-server.x86_64 : An open source SSH server daemon
```

Dpkg

dpkg

01 Dpkg is the main package management system in Debian and similar OSes

02 It is used to install, build, remove, and manage packages

03 The package for it has an extension of .deb at the end

04 Dpkg is a low level tool and APT is the commonly used high level tool as it can deal with complex tasks involved in package management

05 The dpkg database is located under /var/lib/dpkg

Install Package

Use command ‘-i’ to install a package

Syntax

```
dpkg –i <package name>
```

Example : # dpkg –i python2.7.deb

Syntax

```
# dpkg –s python
```

To check if a package is installed or not use ‘s’ option.

List Package

- Use command ‘-l’ to list a package with dpkg.

Syntax

dpkg –l <package_name>

Example : # dpkg –l python

To list all packages, don’t add a package name.

```
# dpkg –l
```

To view content of a package, use ‘-c’ option.

```
# dpkg –c python2.7.deb
```

Remove Package

- To remove a package we must use package name and not the original one with .deb extension.

Syntax

```
dpkg –r <package name>
```

```
Example : # dpkg –r python
```

Package Install From Directory

- To install from a specified directory specify the directory name.
- Use command ‘-R’ to recursively iterate it.

Syntax

```
dpkg –R - -install <directory_name>
```

Example : # dpkg –R - -install debpackage

Syntax

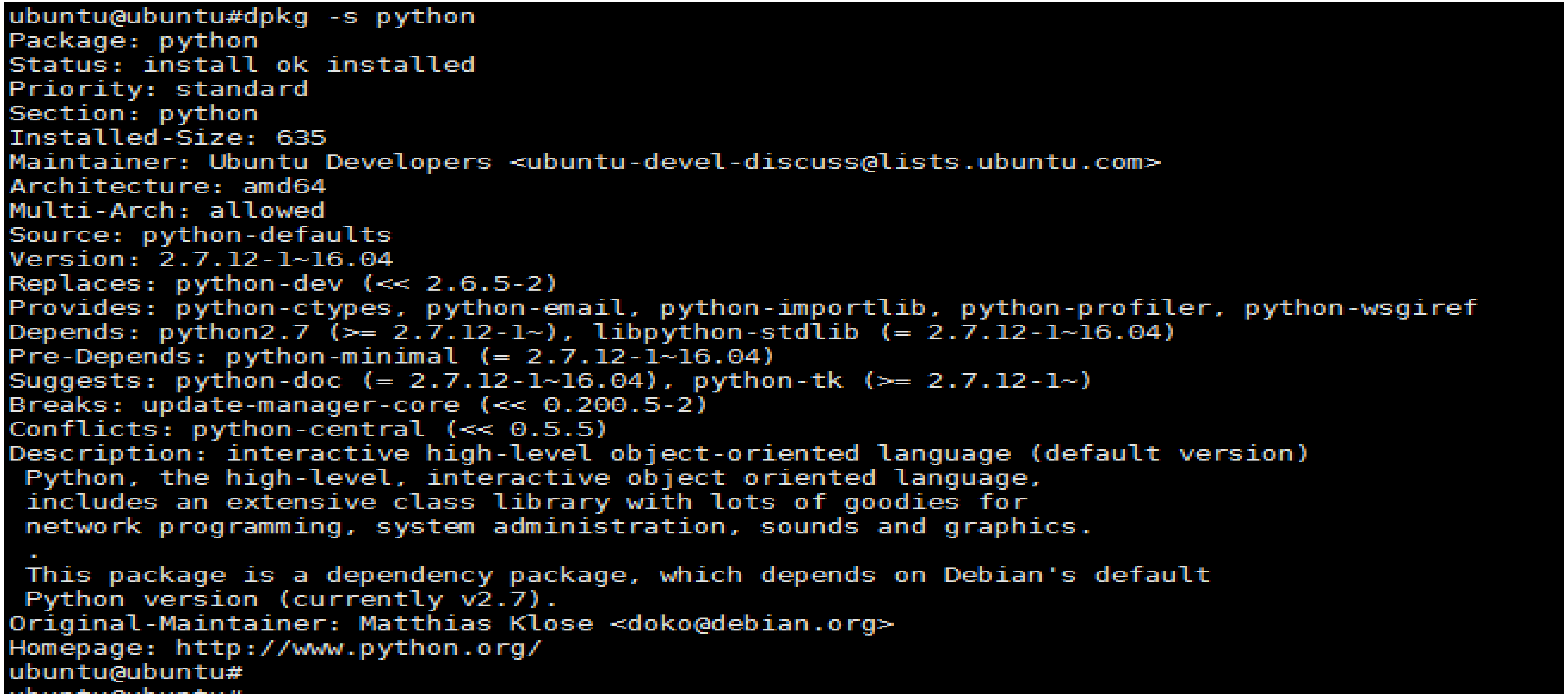
```
# dpkg -u
```

To update a package use ‘-u’ option.

DEMO - dpkg

edureka!

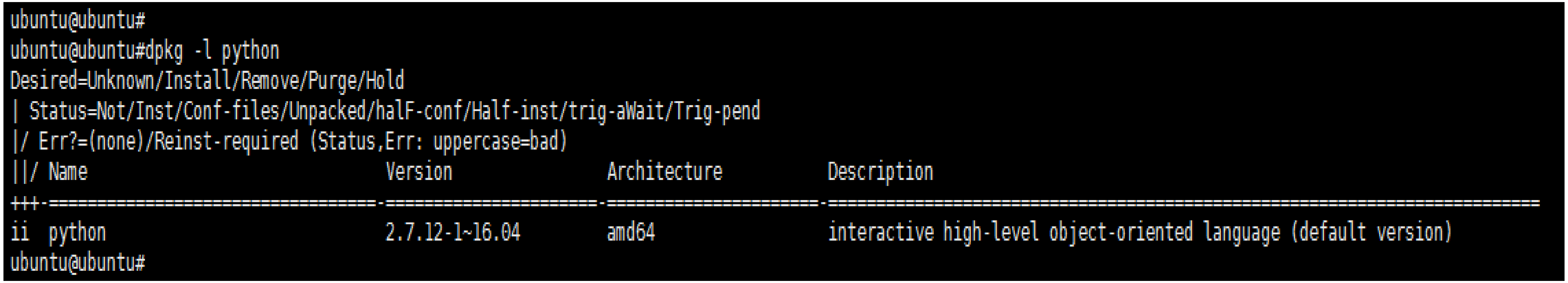
Demo - dpkg



```
ubuntu@ubuntu#dpkg -s python
Package: python
Status: install ok installed
Priority: standard
Section: python
Installed-Size: 635
Maintainer: Ubuntu Developers <ubuntu-devel-discuss@lists.ubuntu.com>
Architecture: amd64
Multi-Arch: allowed
Source: python-defaults
Version: 2.7.12-1~16.04
Replaces: python-dev (<< 2.6.5-2)
Provides: python-ctypes, python-email, python-importlib, python-profiler, python-wsgiref
Depends: python2.7 (>= 2.7.12-1~), libpython-stdlib (= 2.7.12-1~16.04)
Pre-Depends: python-minimal (= 2.7.12-1~16.04)
Suggests: python-doc (= 2.7.12-1~16.04), python-tk (>= 2.7.12-1~)
Breaks: update-manager-core (<< 0.200.5-2)
Conflicts: python-central (<< 0.5.5)
Description: interactive high-level object-oriented language (default version)
 Python, the high-level, interactive object oriented language,
 includes an extensive class library with lots of goodies for
 network programming, system administration, sounds and graphics.
 .
 This package is a dependency package, which depends on Debian's default
 Python version (currently v2.7).
Original-Maintainer: Matthias Klose <doko@debian.org>
Homepage: http://www.python.org/
ubuntu@ubuntu#
ubuntu@ubuntu#
```

edureka!

Demo – dpkg (continued)



```
ubuntu@ubuntu#
ubuntu@ubuntu#dpkg -l python
Desired=Unknown/Install/Remove/Purge/Hold
| Status=Not/Inst/Conf-files/Unpacked/halF-conf/Half-inst/trig-aWait/Trig-pend
|/ Err?=(none)/Reinst-required (Status,Err: uppercase=bad)
||/ Name                               Version                Architecture           Description
+++-==================================-======================-======================-==========================================================================
ii  python                             2.7.12-1~16.04         amd64                  interactive high-level object-oriented language (default version)
ubuntu@ubuntu#
```

edureka!

Apt

edureka!

apt-get

Apt-get is the command line interface to handle package using APT library.

It is the default package management system for Debian-like distro like Ubuntu.

It is an efficient way of handling packages in your system.

Dependencies are managed automatically.

Upgrades and removal are handled carefully to maintain the stability of the system.

It has an external GUI support with tools like synaptic, aptitude, etc.

apt-cache

Apt-cache is the command line interface to search apt software packages.

This tool is used to search software packages and get information about them.

One can search for a package without having exact name of the package.

The data is fetched from different sources listed in sources.list file.

/var/cache/apt/archives/ contains already downloaded packages to avoid downloading them again if one needs to re-install a package after removing it.

edureka!

List & Search Package

- Use command ‘pkgnames’ to list packages starting with a particular string.

Syntax

apt-cache pkgnames <package_name>

Example : # apt-cache pkgnames python

Syntax

# apt-cache search python

Use command ‘search’ to search for a package with a particular name.

edureka!

Check Package Information

- Use command ‘show’ to get details about a package.

Syntax

```
apt-cache show <package_name>
```

Example : # apt-cache show python

Syntax

```
# apt-cache showpkg python
```

To check dependencies of a package use ‘showpkg’ option.

Update Package

- Use command 'update' to update a package.

Syntax

apt-get update <package_name>

Example : # apt-get update python

To update the whole system, don't provide package name.

```
# apt-get update
```

To install a package but prevent from upgrading if already installed use '- -no-upgrade' option.

```
# apt-get install python - -no-upgrade
```

Install Package

- Use command ‘install’ to install a package.

Syntax

apt-get install <package_name>

Example : # apt-get install python

To install multiple packages together, provide multiple package name after install.

# apt-get install python mysql

To install multiple package having a particular string, use wildcard.

# apt-get install ‘*name’

edureka!

Upgrade Package

- Use command ‘upgrade’ to upgrade the system. It may remove or update the installed packages.

Syntax

```
apt-get upgrade
```

Example : # apt-get upgrade

Syntax

```
# apt-get install python - -only-upgrade
```

To upgrade only specific package without installation of any new packages use ‘- -only-upgrade’ option.

Remove Package

- Use command ‘remove’ to remove a particular package.

Syntax

```
apt-get remove <package_name>
```

Example : # apt-get remove python

Syntax

```
# apt-get remove - -purge python
```

Removing a package doesn’t remove its configuration file. To remove configuration files along with it, append with ‘purge’ option.

edureka!

Download Package

- Use command ‘download’ to download a package without installing it.

Syntax

apt-get download python

Example : # apt-get download python

Syntax

# apt-get source python

To download and unpack source code of a package use ‘source’ option.

edureka!

Check Dependencies

- Use command ‘check’ to check for dependencies.

Syntax

```
apt-get check
```

Example : # apt-get download python

Syntax

```
# apt-get build-dep python
```

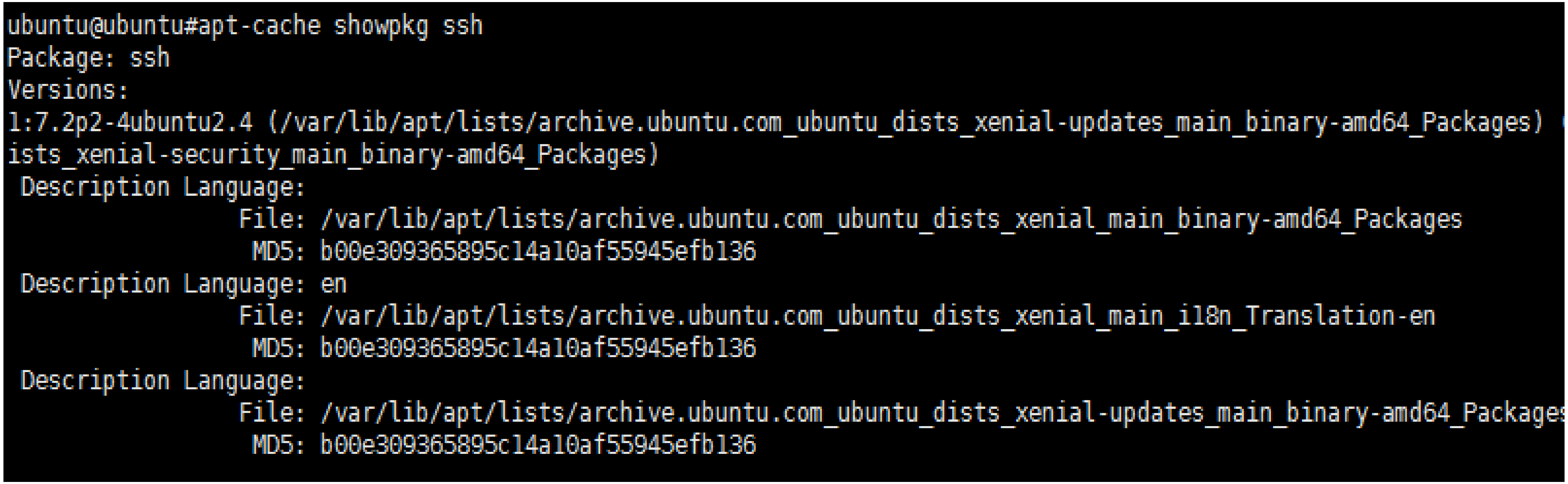
To install build dependencies use ‘build-dep’ option.

DEMO : apt

edureka!

Demo - apt

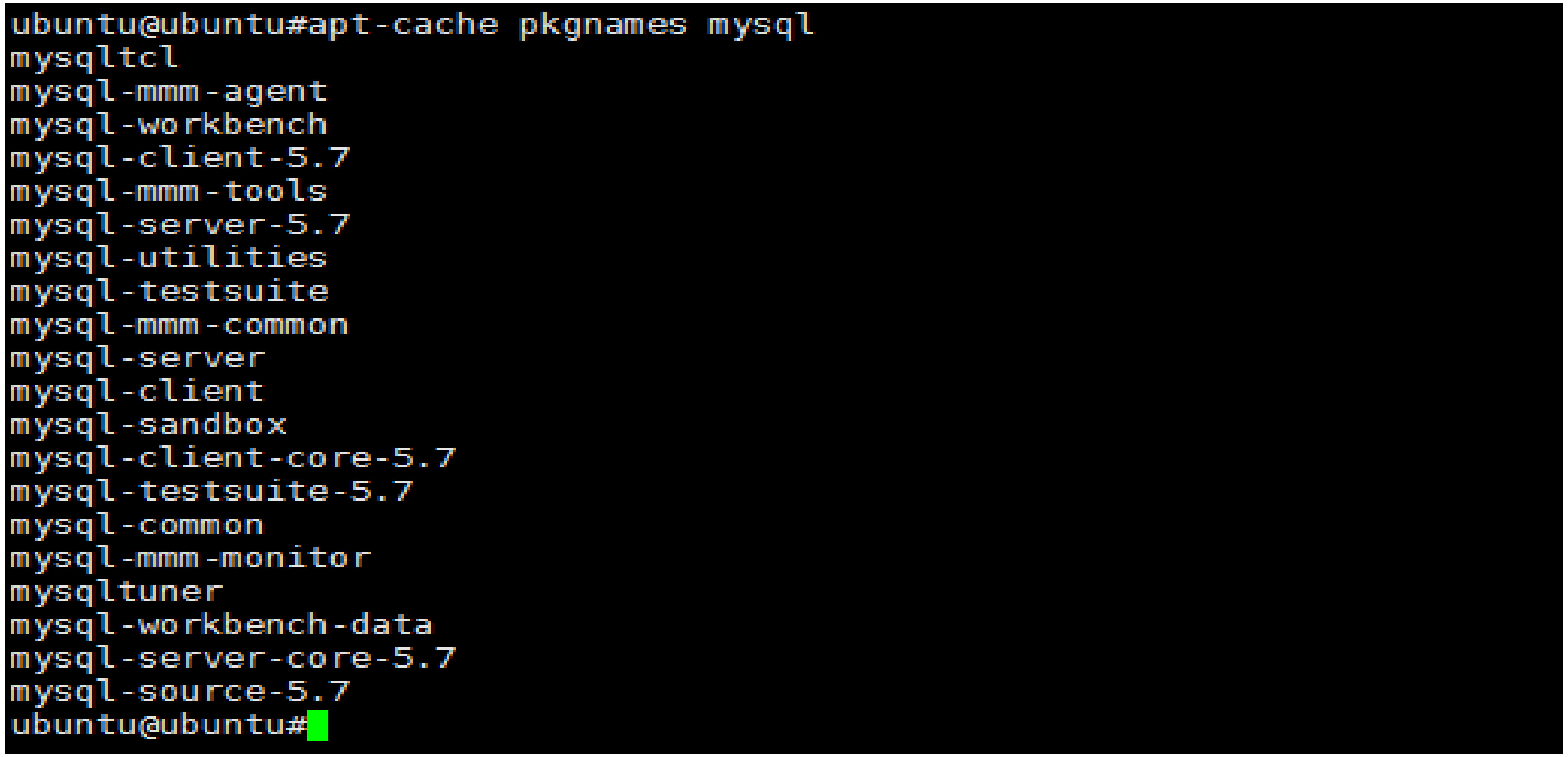
- Show package details
- Install package with no-upgrade option



```
ubuntu@ubuntu#apt-cache showpkg ssh
Package: ssh
Versions:
1:7.2p2-4ubuntu2.4 (/var/lib/apt/lists/archive.ubuntu.com_ubuntu_dists_xenial-updates_main_binary-amd64_Packages) (
ists_xenial-security_main_binary-amd64_Packages)
 Description Language:
                 File: /var/lib/apt/lists/archive.ubuntu.com_ubuntu_dists_xenial_main_binary-amd64_Packages
                  MD5: b00e309365895c14a10af55945efb136
 Description Language: en
                 File: /var/lib/apt/lists/archive.ubuntu.com_ubuntu_dists_xenial_main_i18n_Translation-en
                  MD5: b00e309365895c14a10af55945efb136
 Description Language:
                 File: /var/lib/apt/lists/archive.ubuntu.com_ubuntu_dists_xenial-updates_main_binary-amd64_Packages
                  MD5: b00e309365895c14a10af55945efb136
```

Demo - apt (continued)

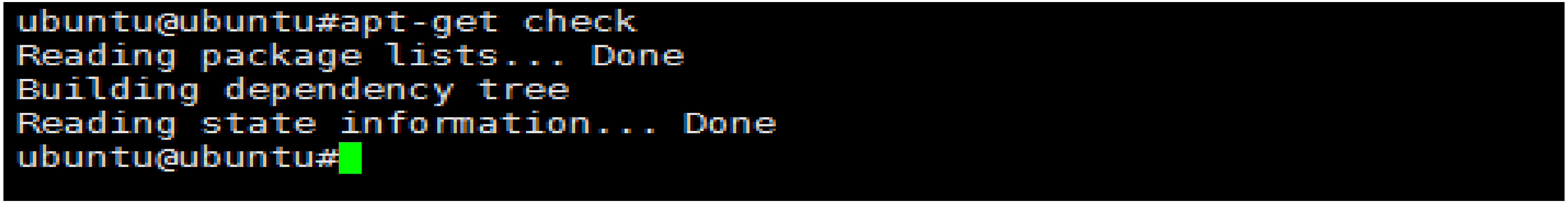
- Get packages with mysql



```
ubuntu@ubuntu#apt-cache pkgnames mysql
mysqltcl
mysql-mmm-agent
mysql-workbench
mysql-client-5.7
mysql-mmm-tools
mysql-server-5.7
mysql-utilities
mysql-testsuite
mysql-mmm-common
mysql-server
mysql-client
mysql-sandbox
mysql-client-core-5.7
mysql-testsuite-5.7
mysql-common
mysql-mmm-monitor
mysqltuner
mysql-workbench-data
mysql-server-core-5.7
mysql-source-5.7
ubuntu@ubuntu#
```

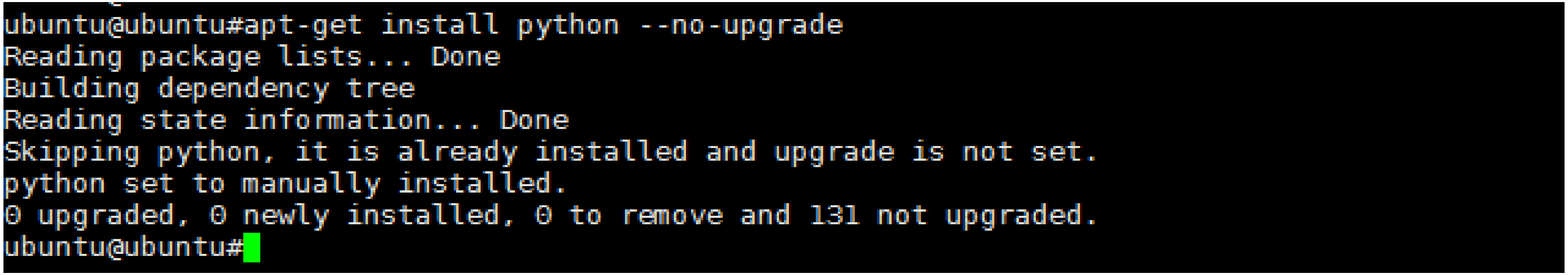
Demo - apt (continued)

- Check for dependencies



```
ubuntu@ubuntu#apt-get check
Reading package lists... Done
Building dependency tree
Reading state information... Done
ubuntu@ubuntu#
```

- Install a package with no-upgrade option



```
ubuntu@ubuntu#apt-get install python --no-upgrade
Reading package lists... Done
Building dependency tree
Reading state information... Done
Skipping python, it is already installed and upgrade is not set.
python set to manually installed.
0 upgraded, 0 newly installed, 0 to remove and 131 not upgraded.
ubuntu@ubuntu#
```

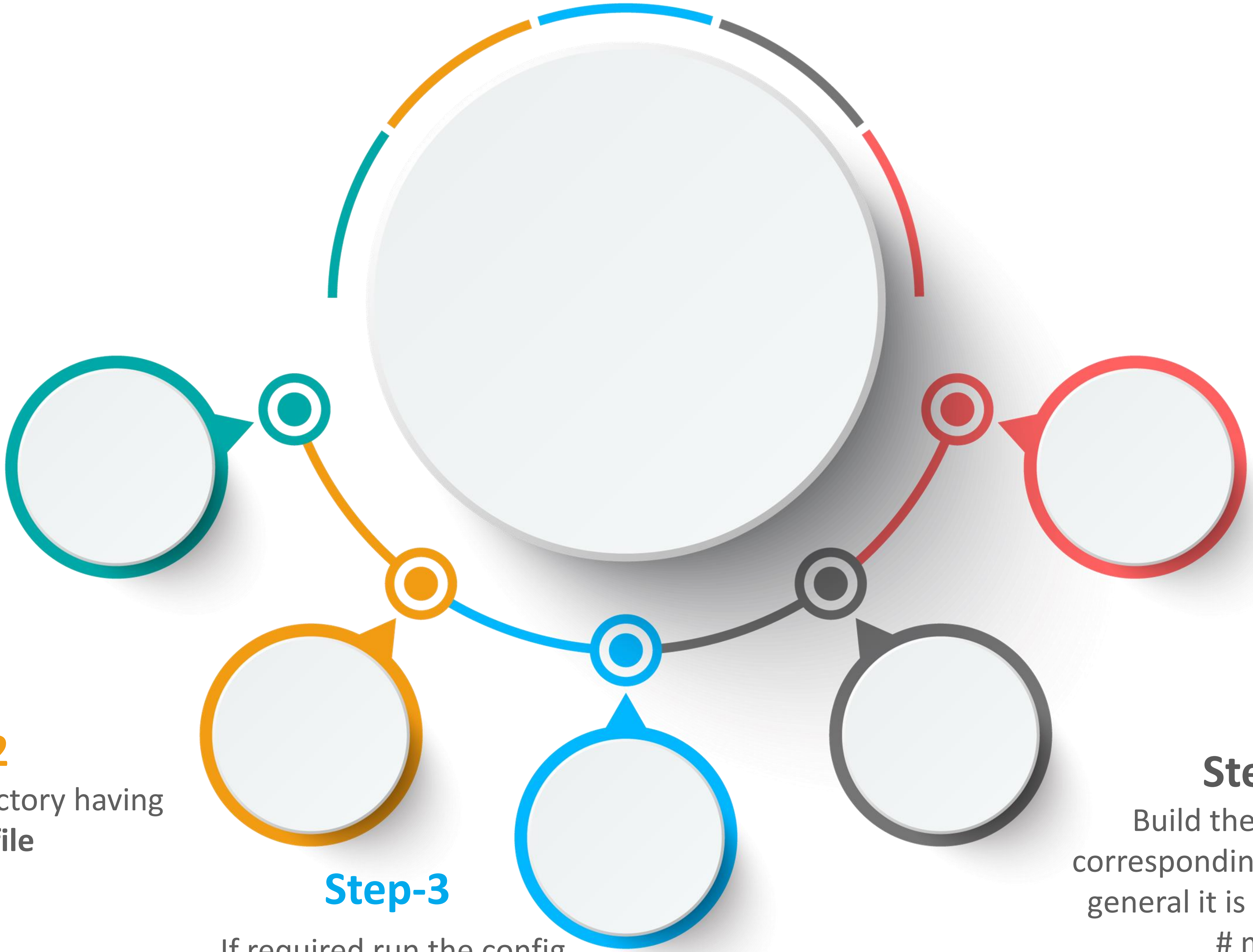
Build From Source Code

edureka!

Build From Source Code

- Download the software, compile it to generate binaries or libraries.
- This is useful when installing non-official releases.
- It helps in installing on embedded devices or devices without internet.
- The download file is generally compressed in zip, tar, etc formats.
- One should read the README file which generally has the instruction to install and usage.
- It should also outline any dependencies regarding the same.

Steps



Step-1

Perform any configuration changes if required for the system. May need to change the directory name or IP address in the script.

Step-2

Traverse to the directory having the **Makefile**

Step-3

If required run the config
# make config

Step-4

Build the code with corresponding command. In general it is simple ‘make’
# make

Step-5

Install binary and library at suitable place mentioned in README.
# make install

edureka!

Libraries

- The source code often comes along with some pre-compiled libraries
- These are common functionalities that are used by multiple programs in the code
- They are generally found in ‘lib’ folder in any source code
- The libraries are of two types :

Static Library

- During compilation library is packed in binary

Dynamic Library

- Binary is linked to library during runtime.

Quiz

1. The command to end the process (pid – 555) before it exits itself is _________ ?

   a. kill- -9 555

   b. Kill 555

   c. Kill -2 555

   d. exit

Answers

1. The command to end the process (pid – 555) before it exits itself is _________ ?

   a. kill- -9 555

   b. Kill 555

   c. Kill -2 555

   d. exit

Answer A: to kill a particular process with its process Id

Quiz

2. Which package system does Ubuntu use?

   a. rpm

   b. deb

   c. tgz

   d. rhp

Answers

2. Which package system does Ubuntu use?

   a. rpm

   b. deb

   c. tgz

   d. rhp

Answer B: Ubuntu and other Debian-like distro use deb packages.

Summary

- In this module, you should have learnt to:
  - Configure services to run at boot
  - Perform Package Management – installing and removing Packages
  - Verify dependencies on packages and resolve them
  - Understand kernel configuration
  - Shut down the system

Questions

FEEDBACK
RESPONSE
OPINION
SURVEY
COMMENT
IDEA
ADVICE
RESULT
RATING

Thank You

For more information please visit our website
www.edureka.co