

Linux Systems and Open Source Software

Package Management

Chia-Heng Tu
Dept. of Computer Science and Information Engineering
National Cheng Kung University
Fall 2022



Outline

- Introduction
- Package Management
  - **rpm**
  - **dpkg** and **apt**
  - GNOME Software: App. Store in Linux
- Solutions to Handle Dependencies for
  - Ubuntu Packages: Snap (Snappy)
  - Python Packages: Uncompiled Package Management

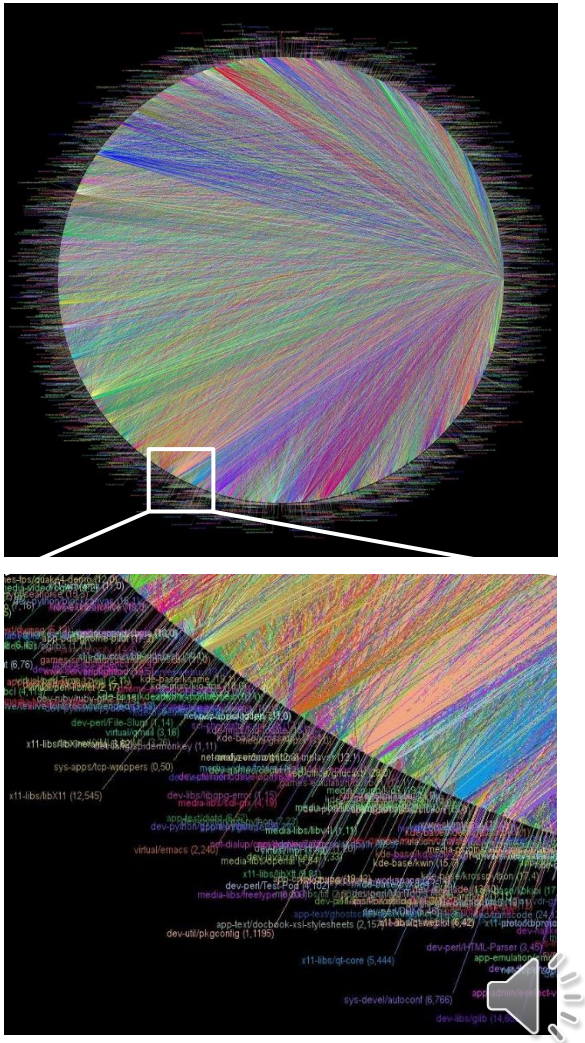


INTRODUCTION



Status Quo of Open Source Software

- Open source software is everywhere
  - Do not have to reinvent the wheel
  - Are able to make the most of the tools at your disposal
- However, **dependencies** among software are extremely complex
  - The package dependency graph (right) shows the dependencies of Gentoo Linux packages
  - The most common dependencies are development tools (below)



63,988 dependencies among 14,319 software packages

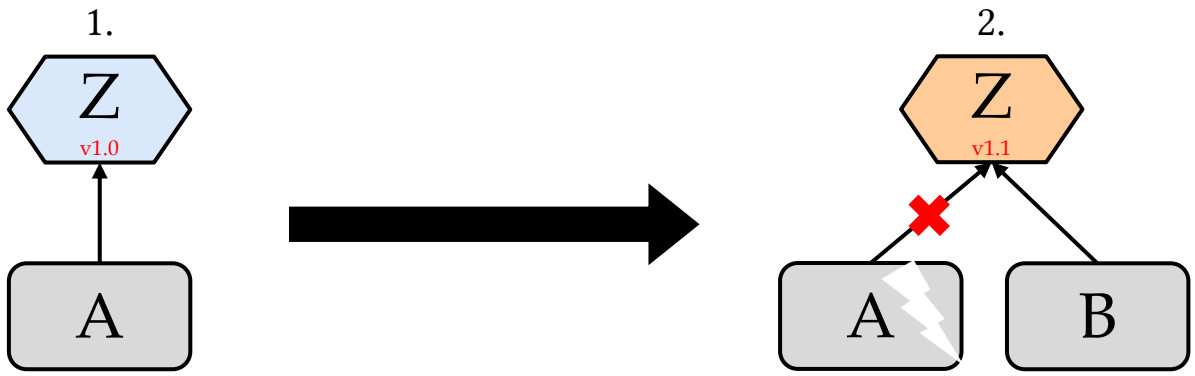
Package	# of reverse dependencies
dev-lang/perl	1559
dev-util/pkgconfig	1195
dev-lang/python	1047
x11-libs/gtk+	1042

Courtesy of https://disfunksioneel.blogspot.com/2011/04/



A Common Package Conflict Example

1. You download software A, which uses library Z
   - Software A's installer puts a copy of library Z in your system directory
2. You then download software B, which also uses library Z, but a *slightly different version*
   - It replaces the original copy of library Z in your system directory with the other version
   - But that version doesn't work right with program A, so A does not work properly



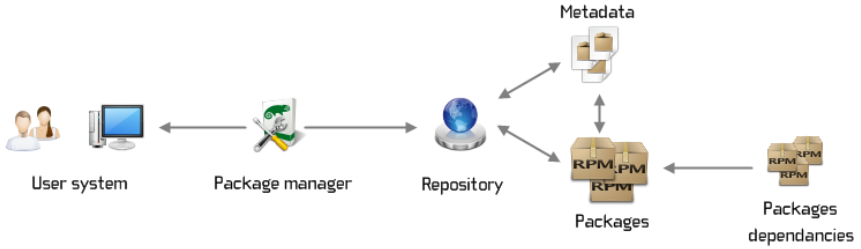
Dependency Hell

Courtesy of https://superuser.com/a/1000388

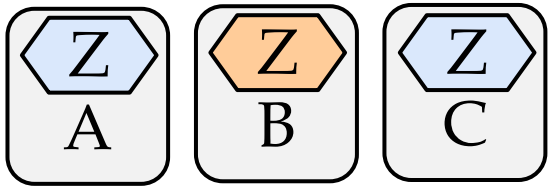


A Common Package Conflict Example (Cont'd)

- Possible solution 1
  - Smart **package managers** can perform smart upgrades: interdependent software components are upgraded at the same time

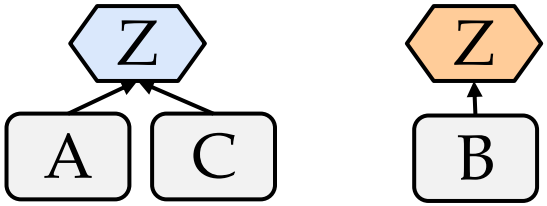


- Possible solution 2
  - Each program include its library Z
  - E.g., "Private DLLs" in Windows, copies of libraries per application in the directory of the application



✓ dependency-free
✘ more traffic
✘ more disk space
✘ more ram

- Possible solution 3
  - The best solution, but it is difficult to implement
  - OS allows an application to request a module/library by a unique name and version number (E.g., Windows Vista and Gentoo Linux)



- Visit this page for more

Courtesy of https://en.wikipedia.org/wiki/Dependency_hell; https://devopedia.org/package-manager



PACKAGE MANAGEMENT



Package Management

- Is a method of **installing** and **maintaining** software on the system
  - It also includes **updating** and probably **removing** SW as well
- In the early days of Linux
  - software were only distributed as **source code**, along with the required man pages, the necessary configuration files, and more
- Nowadays
  - most Linux distributors use ***pre-built*** machine codes called ***packages***,
  - which are presented to users ready for installation on that distribution
  - One of the wonders of Linux is still the possibility to obtain source code of a program to be studied, improved, and compiled

Courtesy of https://www.tecmint.com/linux-package-management/



A Packaging System

- A collection of software tools
  - automates the process of installing, upgrading, configuring, and removing SW (in the form of packages)
  - Metadata within packages helps the above things
    - E.g., the software's name, description of its purpose, version number, vendor, checksum, and *a list of dependencies* necessary for the software to run properly
  - **Example of the functionalities of the tools:**
    - Working with file archivers to extract package archives
    - Ensuring the integrity and authenticity of the package by verifying their digital certificates and checksums
    - Looking up, downloading, installing, or updating existing software from a software repository or app store
    - Grouping packages by function to reduce user confusion
    - Managing dependencies to ensure a package is installed with all packages it requires, thus avoiding "Dependency Hell"
- Each Linux distribution family uses a distinct packaging system
  - Debian: *.**deb**; CentOS: *.**rpm**; openSUSE: *.**rpm** built specially for openSUSE
  - A package intended for one distribution will not be compatible with another distribution
  - However, most distributions are likely to fall into one of the three distribution families

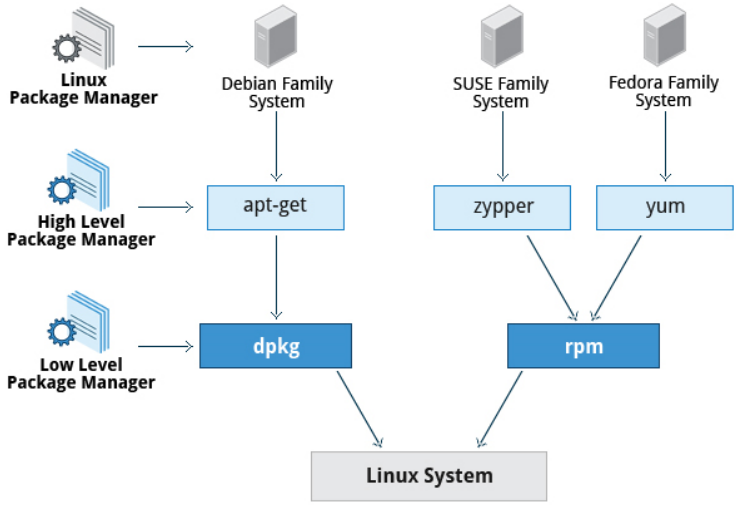
Courtesy of https://en.wikipedia.org/wiki/Package_manager



High and Low-level Package Tools

- Two levels of utilities are available for package management
  - **High-level tools**: ensure dependency resolution and search metadata
  - **Low-level tools**: actual installation, upgrade, and removal of package files

Distribution	Low-level	High-level
Debian-based	dpkg	apt / aptitude
RPM-based	rpm	yum
openSUSE	rpm	zypper

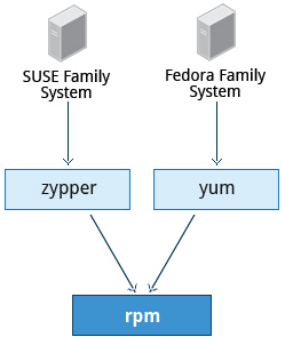


Courtesy of https://www.tecmint.com/linux-package-management/
Courtesy of https://borosan.gitbook.io/lpic1-exam-guide/1024-use-debian-package-management



RPM (RPM Package Manager)

- RPM (originally *Red Hat Package Manager*)
  - Refers to **.rpm** file format and package manager program itself
  - Is the package management system used by Linux Standard Base (LSB) compliant distributions for low-level handling of packages
  - Is used to query, install, verify, upgrade, and remove packages
  - Is more frequently used by RPM-based distributions, such as Fedora, RHEL, and CentOS



Courtesy of https://en.wikipedia.org/wiki/RPM_Package_Manager



dpkg (Debian Package)

- Is the base (low-level tools) of the package management system
  - of the free operating system Debian and its numerous derivatives
- Is used to install, remove, provide information about and build ***.deb** packages
  - but it can't automatically download and install their corresponding dependencies
- Provides several other programs necessary for run-time functioning
  - including **dpkg-deb, dpkg-split, dpkg-query, dpkg-statoverride, dpkg-divert** and **dpkg-trigger'**

Courtesy of https://en.wikipedia.org/wiki/Dpkg

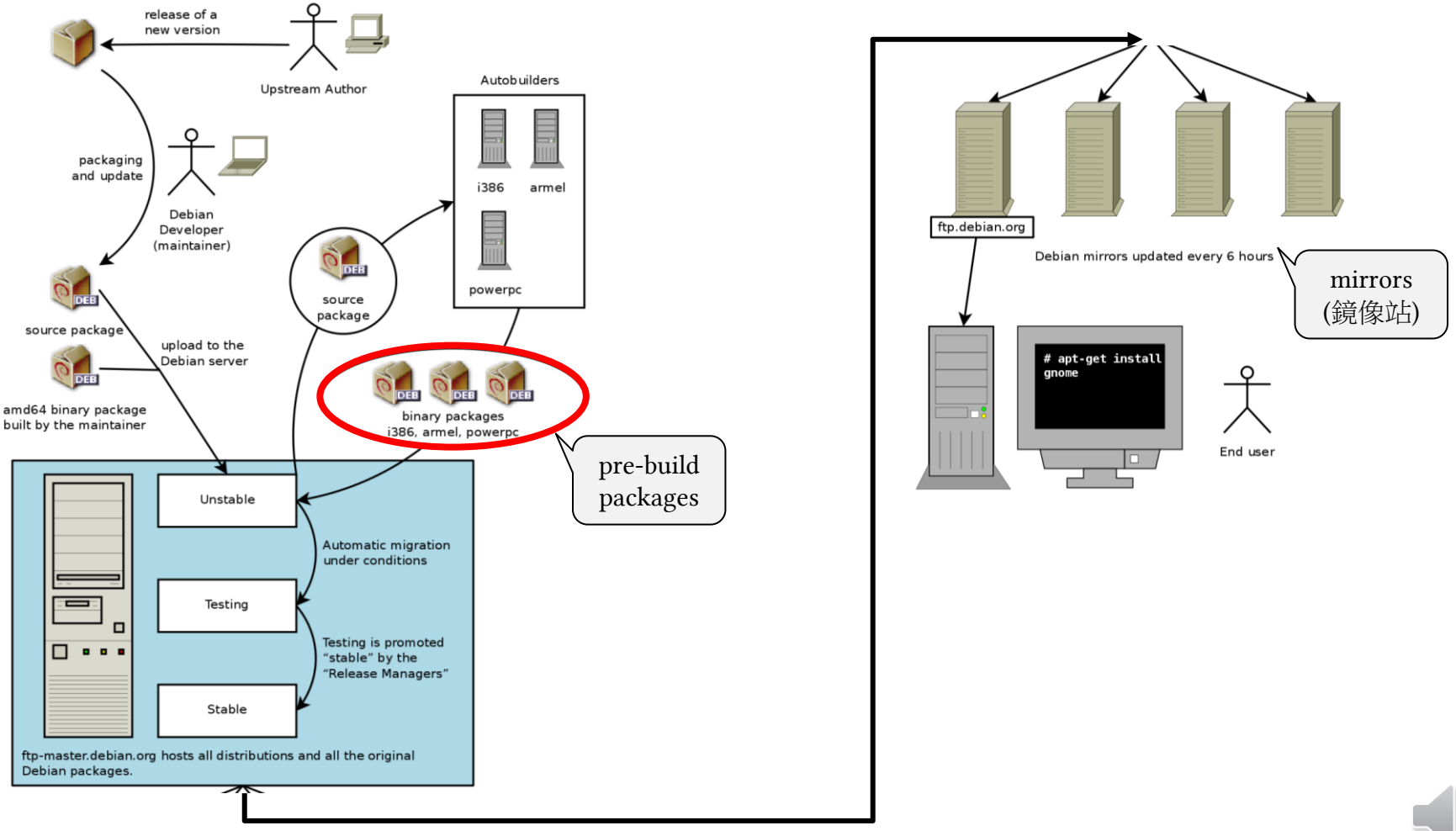


Debian Binary Package File

- A Debian package is a file that ends in **.deb** and contains software for your Debian system
- **deb** Package filename format and example:
  - <package-name>_<epoch>:<upstream-version>-<debian.version>-<architecture>.deb
    - epoch and debian.version are optional
  - e.g., vim_8.0.1453-1ubuntu1_amd64.deb

Courtesy of https://www.debian.org/doc/manuals/debian-reference/ch02.en.html#_debian_package_file_names

Development and Usage Flow of deb



Courtesy of https://www.debian.org/doc/manuals/debian-handbook/sect.release-lifecycle.ro.html



dpkg Commands (Query)

- $/> dpkg -l, --list {package}
  - List packages that match the given pattern

```
$ dpkg -l python3
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  python3        3.6.7-1~18.04    amd64            interactive high-level object-oriented language
```

Link

- $/> dpkg -s, --status {package}
  - Report status of specified package

```
$ dpkg -s python3
Package: python3
Status: install ok installed
Priority: important
Section: python
Installed-Size: 187
Maintainer: Ubuntu Developers <ubuntu-devel-discuss@lists.ubuntu.com>
Architecture: amd64
Multi-Arch: allowed
Source: python3-defaults
Version: 3.6.7-1~18.04
Replaces: python3-minimal (<< 3.1.2-2)
Provides: python3-profiler
Depends: python3.6 (>= 3.6.7-1~), libpython3-stdlib (= 3.6.7-1~18.04)
Pre-Depends: python3-minimal (= 3.6.7-1~18.04)
```

The various front-ends to dpkg (such as apt-get, aptitude, and dselect) use these fields to facilitate package management

Courtesy of http://man7.org/linux/man-pages/man1/dpkg-query.1.html
https://www.debian.org/doc/debian-policy/ch-relationships.html



dpkg Commands

- **$/> dpkg -i, --install {package-file}**
  - Install the package
- **$/> dpkg -r, --remove {package}**
  - Remove an installed package, except *conffiles*
- **$/> dpkg -P, --purge {package}**
  - This removes everything, including *conffiles*

Note: you should use high-level tools by default

All configurations files managed by dpkg are called “conffiles”

Courtesy of http://man7.org/linux/man-pages/man1/dpkg.1.html



APT (Advanced Package Tool)

- The **apt** command is a high-level command-line interface for package management
  - It is basically a wrapper of **apt-get**, **apt-cache** and similar commands
  - originally intended as an end-user interface and enables some options better suited for interactive usage by default

$/> apt install …

```
Processing triggers for libglib2.0-0:i386 (2.39.92-2) ...
Processing triggers for libglib2.0-0:amd64 (2.39.92-2) ...
Processing triggers for shared-mime-info (1.2-0ubuntu2) ...

Progress: [ 60%] [##################################.....................]
```

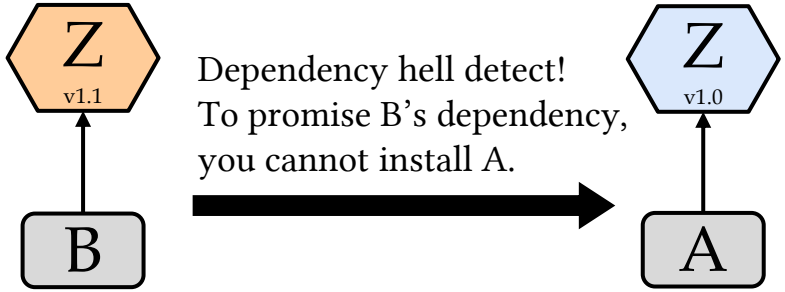
- Functionality
  - Provide a friendly progress bar when installing packages
  - Will remove cached **.deb** packages by default after successful installation of downloaded packages
- You can find more about, apt, apt-get, aptitude

Courtesy of https://www.debian.org/doc/manuals/debian-reference/ch02.en.html#_literal_apt_literal_vs_literal_apt_get_literal_literal_apt_cache_literal_vs_literal_aptitude_literal
Courtesy of https://www.omgubuntu.co.uk/2014/04/how-to-enable-apt-terminal-progress-bar



How Does APT "Solve" Dependency Hell?

- It reports the dependency error!
- Example:
  - **B** has been installed and want to install **A**
  - **B** needs **Z** (version 1.1) to work
  - **A** needs **Z** (version 1.0) to work




```
$ sudo apt install A
Reading package lists... Done
Building dependency tree
Reading state information... Done
You might want to run 'apt-get -f install' to correct these:
The following packages have unmet dependencies:
 libncurses5 : Depends: Z (= 1.1) but 1.0 is to be installed
E: Unmet dependencies. Try 'apt-get -f install' with no packages (or specify a solution).
```

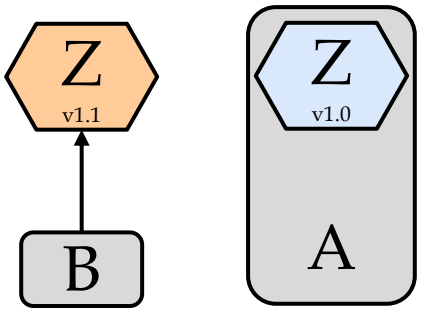
  - APT avoided installation failure due to missing dependencies
  - It is not possible to have two versions of the same package in Linux, because there will be conflicting files

Courtesy of https://unix.stackexchange.com/questions/350192/apt-get-not-properly-resolving-a-dependency-on-a-fixed-version-in-a-debian-ubunt
Courtesy of https://askubuntu.com/q/758502



What If We Still Want to Install Both?

- Though this situation (to keep both versions) rarely appear, there are some possible schemes can be applied:
  1. Wait the developers solve it
  2. Build **A** or **B** yourself from the source code
  3. Use special software (e.g., snap as described in the next section)



Ubuntu Packages Search

- Use the tool to search for packages to be installed with APT

https://packages.ubuntu.com/

Search the contents of packages

This search engine allows you to search the contents of Ubuntu distributions for any files a full list of files in a given package.

Keyword: vim Search Reset

Display:

- packages that contain files named like this
- packages that contain files whose names end with the keyword
- packages that contain files whose names contain the keyword

Distribution: bionic Architecture: amd64

List of files within the vim_8.0.1453-1ubuntu1.3_amd64.deb:

/usr/bin/vim.basic
/usr/share/bug/vim/presubj
/usr/share/bug/vim/script
/usr/share/doc/vim/NEWS.Debian.gz
/usr/share/doc/vim/changelog.Debian.gz
/usr/share/doc/vim/copyright
/usr/share/lintian/overrides/vim

ubuntu packages — Search — package names

» Ubuntu » Packages » bionic (18.04LTS) » editors » vim

[Source: vim] [xenial] [xenial-updates] [bionic] [bionic-updates] [disco] [disco-updates] [eoan] [eoan-updates] [focal]

Package: vim (2:8.0.1453-1ubuntu1.3 and others) [security]

Vi IMproved - enhanced vi editor

Packages providing vim

vim-athena
Vi IMproved - enhanced vi editor - with Athena GUI
vim-gtk
Vi IMproved - enhanced vi editor - with GTK2 GUI
vim-gtk3
Vi IMproved - enhanced vi editor - with GTK3 GUI
vim-nox
Vi IMproved - enhanced vi editor - with scripting languages support

GUI

Other Packages Related to vim

depends · recommends · suggests · enhances

- libacl1 (>= 2.2.51-8) — Access control list shared library
- libc6 (>= 2.15) [not arm64, ppc64el] — GNU C Library: Shared libraries; also a virtual package provided by libc6-udeb
- libc6 (>= 2.17) [arm64, ppc64el]
- libgpm2 (>= 1.20.7) — General Purpose Mouse - shared library
- libpython3.6 (>= 3.6.4~rc1) [not amd64, i386] — Shared Python runtime library (version 3.6)
- libpython3.6 (>= 3.6.5) [amd64, i386]
- libselinux1 (>= 1.32) — SELinux runtime shared libraries
- libtinfo5 (>= 6) — shared low-level terminfo library for terminal handling
- vim-common (= 2:8.0.1453-1ubuntu1) [not amd64, i386] — Vi IMproved - Common files
- vim-common (= 2:8.0.1453-1ubuntu1.3) [amd64, i386]
- vim-runtime (= 2:8.0.1453-1ubuntu1) [not amd64, i386] — Vi IMproved - Runtime files
- vim-runtime (= 2:8.0.1453-1ubuntu1.3) [amd64, i386]
- ctags — virtual package provided by exuberant-ctags
- vim-doc — Vi IMproved - HTML documentation
- vim-scripts — plugins for vim, adding bells and whistles

Core lib

Download vim

Architecture	Version	Package Size	Installed Size	Files
amd64	2:8.0.1453-1ubuntu1.3	1,126.0 kB	2,789.0 kB	[list of files]
arm64	2:8.0.1453-1ubuntu1	956.1 kB	2,591.0 kB	[list of files]
armhf	2:8.0.1453-1ubuntu1	962.6 kB	1,815.0 kB	[list of files]
i386	2:8.0.1453-1ubuntu1.3	1,179.4 kB	3,048.0 kB	[list of files]
ppc64el	2:8.0.1453-1ubuntu1	1,322.9 kB	4,308.0 kB	[list of files]
s390x	2:8.0.1453-1ubuntu1	1,038.0 kB	3,027.0 kB	[list of files]

Links for vim

Ubuntu Resources:

- Bug Reports
- Ubuntu Changelog
- Copyright File

Download Source Package vim:

- [vim_8.0.1453-1ubuntu1.3.dsc]
- [vim_8.0.1453.orig.tar.gz]
- [vim_8.0.1453-1ubuntu1.3.debian.tar.xz]

Maintainer:

- Ubuntu Developers (Mail Archive)

Please consider filing a bug or asking a question via Launchpad before contacting the maintainer directly.

Original Maintainers (usually from Debian):

- Debian Vim Maintainers (Mail Archive)
- James McCoy

It should generally not be necessary for users to contact the original maintainer.

External Resources:

- Homepage [vim.sourceforge.io]

Similar packages:

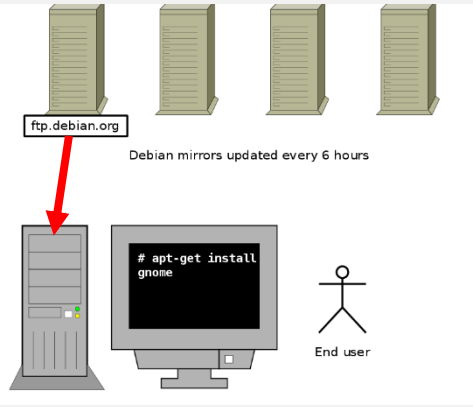
- vim-gnome
- vim-gtk
- vim-athena
- vim-gtk3
- vim-athena-py2
- vim-gnome-py2
- vim-gtk-py2
- vim-gtk3-py2
- vim-nox
- vim-nox-py2
- vim-common



APT Commands

- **$/> sudo apt install {package}**
  - Install a single package
- **$/> sudo apt update**
  - Update Repository Index
  - Only update packages' information

List of hosts (mirrors) are recorded in **/etc/apt/sources.list**



- **$/> sudo apt install --only-upgrade {package}**
  - Upgrade single package
- **$/> sudo apt upgrade**
  - Upgrade entire system packages to latest version

- Be careful with this!!!
- Doing this could lead to failures of program executions due to dependency failures

Courtesy of https://www.2daygeek.com/apt-command-examples-manage-packages-debian-ubuntu-systems/



APT Commands (Cont'd)

- **$/> sudo apt remove {package}**
  - Remove a single package
- **$/> sudo apt purge {package}**
  - Remove a package with all files
- **$/> sudo apt autoremove {package}**
  - removes an installed package and dependencies

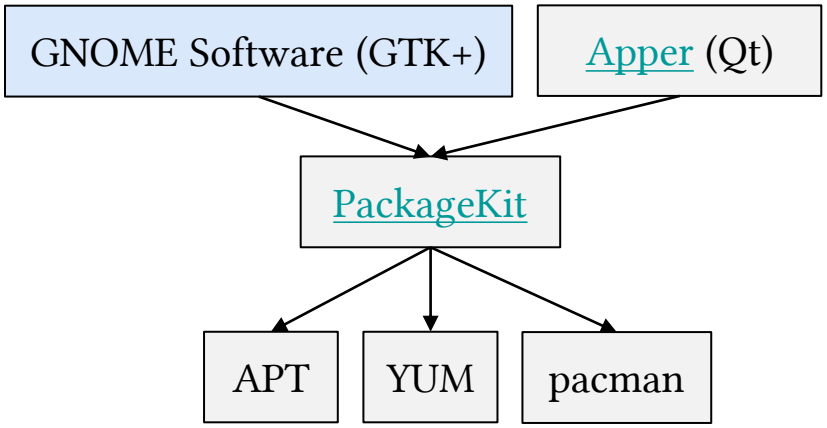
- **$/> sudo apt add-apt-repository ppa:jonathonf/vim**
  - Add PPA (Personal Package Archive) to system
    (Add third party's host to **/etc/apt/sources.list.d/)**

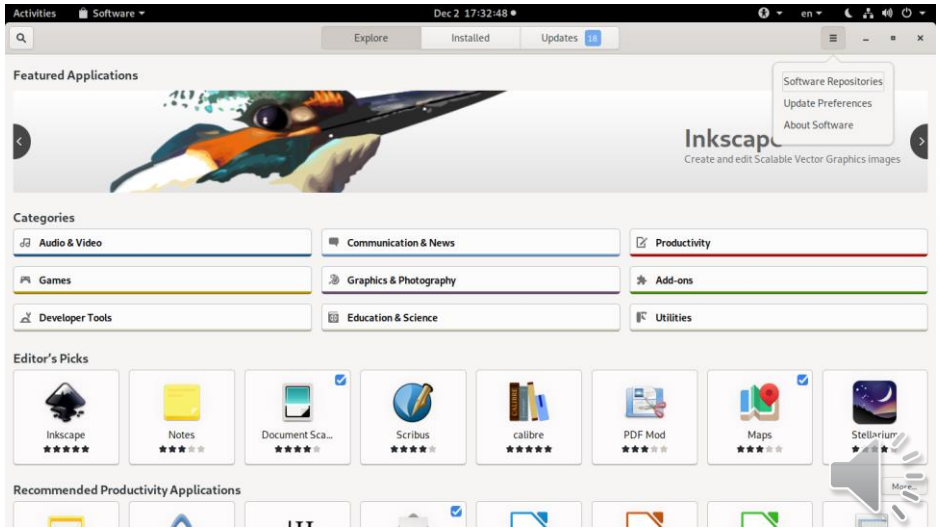
Courtesy of https://www.2daygeek.com/apt-command-examples-manage-packages-debian-ubuntu-systems/



GNOME Software

- GNOME Software is a utility for application and updates installation on the Linux (e.g., app. store for Linux)
  - It is the GNOME front-end to the PackageKit,
  - in turn a front-end to several package management systems, these include systems based on both rpm and deb (see the bottom-left figure)
- Ubuntu replaced its previous Ubuntu Software Center program with GNOME Software starting with Ubuntu 16.04 LTS





Courtesy of https://www.wikiwand.com/en/GNOME_Software



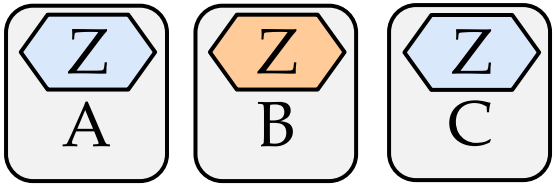
Snap (for Linux applications)
Virtualenv (for Python applications)

SOLUTIONS TO HANDLE DEPENDENCIES



Snap

- A Snap is a bundle of an app and its dependencies
  - that works without modification across many different Linux distributions



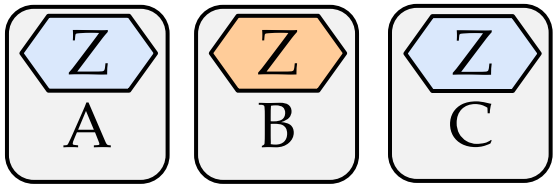
- more traffic
- more disk space
- more ram

- Unlike traditional Linux package management approaches,
  - which require specifically adapted packages for each Linux distribution (such as APT or YUM)

Courtesy of https://anglehit.com/ubuntu-snap-just-made-installing-new-apps-and-dependencies-on-16-04-lts-much-easier/
Courtesy of https://snapcraft.io/docs

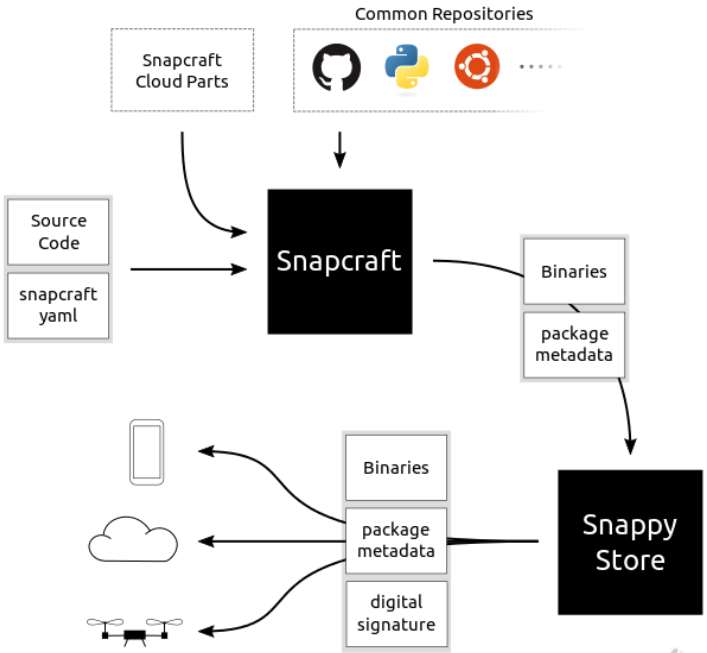


Snap/Snappy/Snapcraft



- more traffic
- more disk space
- more ram

- Snap is both the command line interface and the application package format
- **snapd** is the background service that manages and maintains your snaps
- **snapcraft** is the command and the framework used to build your own snaps
- Snap Store provides a place to upload your snaps, and for users to browse and install

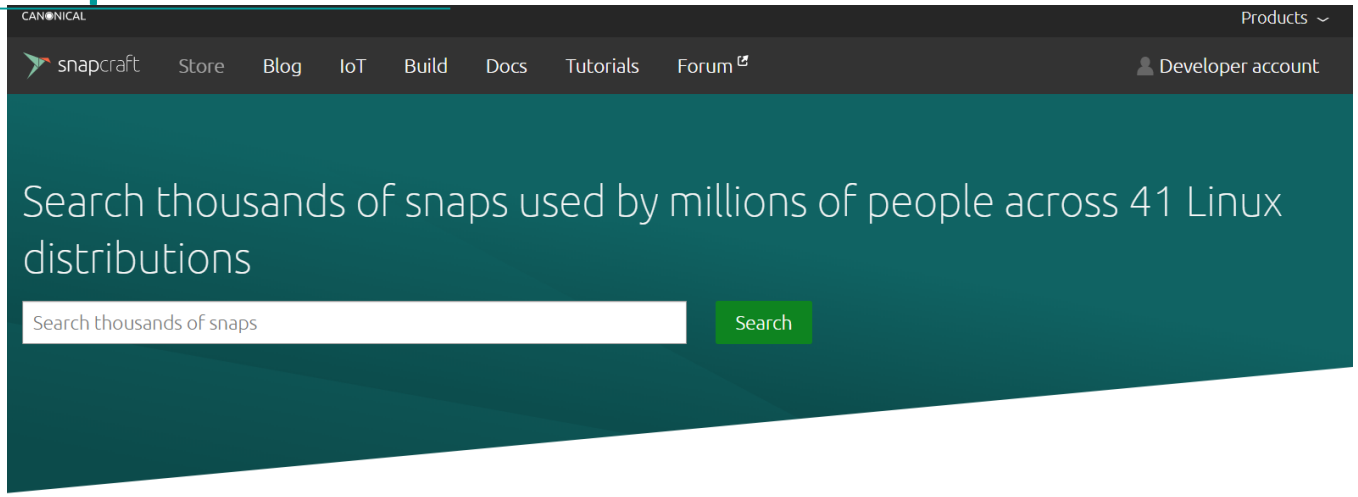


Courtesy of https://anglehit.com/ubuntu-snap-just-made-installing-new-apps-and-dependencies-on-16-04-lts-much-easier/
Courtesy of https://snapcraft.io/docs



Snap Store Screenshot

https://snapcraft.io/store




Riot
Alan Pope
Communicate the way you want with Riot


Mindustry
Alan Pope
A sandbox tower defense game


Inkscape
Inkscape Project
Vector Graphics Editor



Uncompiled Package Management (Python)

- Python is highly portable
  - E.g., it runs across different platforms with its interpreter

- To run Python programs, we also need a tool to maintain the **dependencies between python packages**
- **pip** is standard package-management system
  - Is used to install and manage Python packages
  - Python Package Index (PyPI) is a repository of software for the Python programming language

Courtesy of https://runestone.academy/runestone/books/published/thinkcspy/GeneralIntro/ThePythonProgrammingLanguage.html
Courtesy of https://en.wikipedia.org/wiki/Pip_(package_manager)



pip Commands

- Python package installation
  - **$/> pip install SomePackage** # latest version
  - **$/> pip install SomePackage==1.0.4** # specific version
  - **$/> pip install 'SomePackage>=1.0.4'** # minimum version
- List installed packages
  - **$/> pip list**
    docutils (0.10)
    Jinja2 (2.7.2) ...
- Check compatible dependencies of installed packages
  - **$/> pip check**
    No broken requirements found.
    $/> echo $?
    0

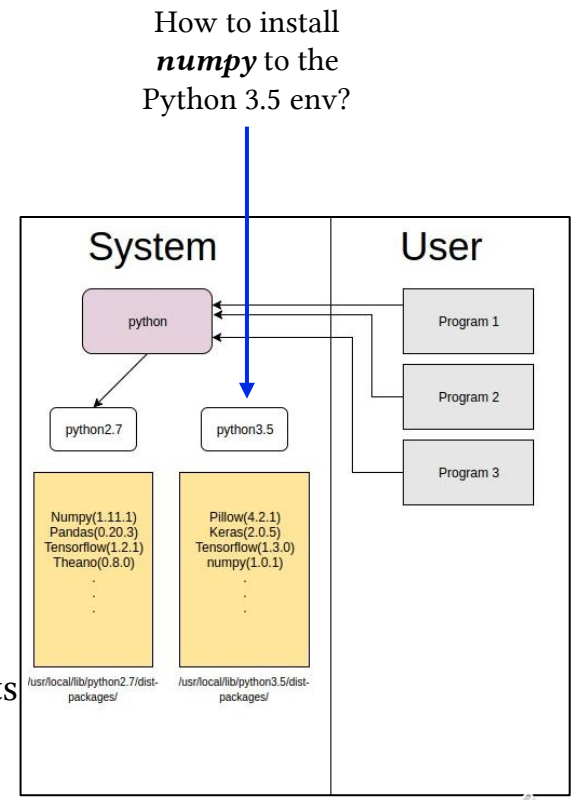
Courtesy of https://pip.pypa.io/en/stable/reference/



Dependency between Python Packages

- Sometimes, Python environments are installed concurrently (co-exist) on a Linux system
  1. Python 2.7 and 3.5
  2. Python 3.1 and 3.5
- What can you do if you were about to install ***numpy*** package on the two different Python environments?

1. Python 2.7 and Python 3.5 (illustrated in the right figure)
   - ***numpy*** 1.11.1 has been installed in Python 2.7
   - You may change your ***default python interpreter and pip*** to python3, so that you can install ***numpy*** 1.0.1 to it
   - by appending **alias python=python3** and **alias pip=pip3** to your **.bashrc** file
2. Python 3.1 and Python 3.5
   - ***numpy*** has been installed in Python 3.1
   - **The above solution does not work** when the Python environments have **the same major version**
   - If you choose to install this new updated *numpy* you risk breaking your existing projects in Python 3.1 as your previous projects depend on the previous API

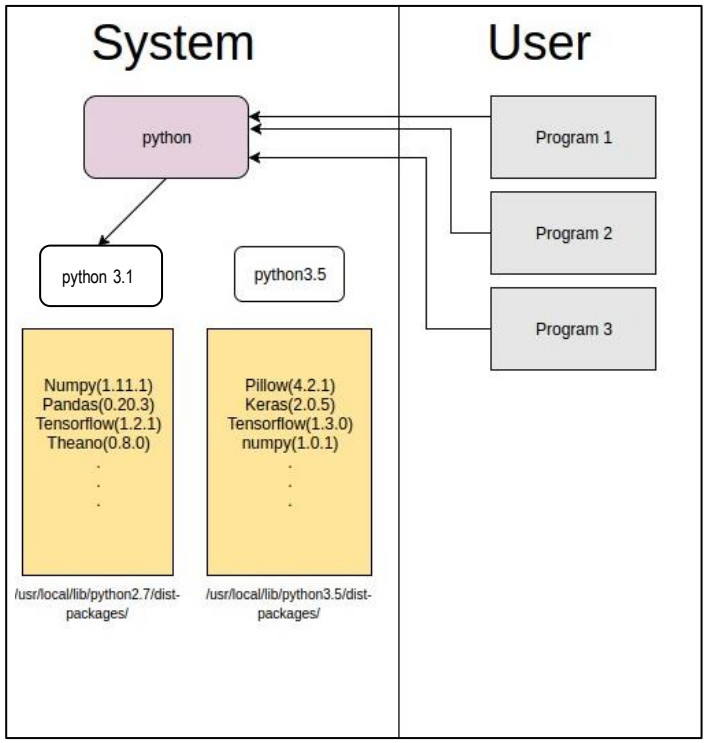


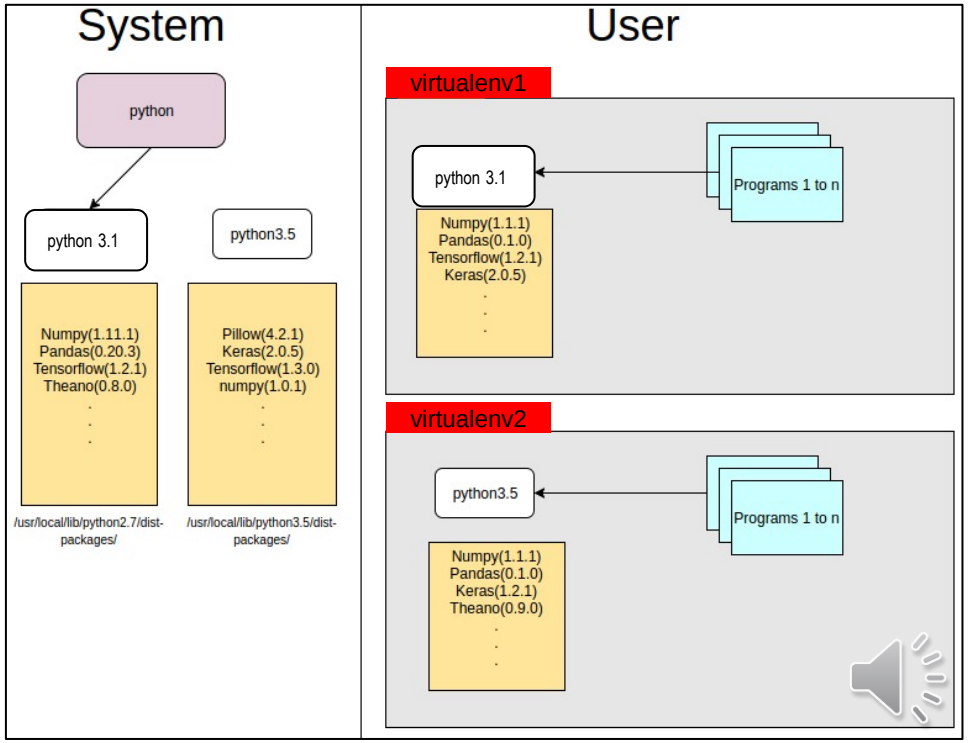
Courtesy of https://solution44.com/2017/08/15/Ideal-python-setup.html



Handle Python-level Dependency

- virtualenv enables users to create multiple mini-python environments
  - which are isolated from **the global python environment (System)** and from **each other (User)**
  - Can be installed easily with, for example, pip





Courtesy of https://solution44.com/2017/08/15/Ideal-python-setup.html



References

- Debian
  - Debian Packaging Tutorial [PDF]
  - Debian New Maintainers' Guide
- APT
  - APT (Advanced Packaging Tool) : Advanced Package Management tool for Debian Based Systems
- Snap
  - https://snapcraft.io/docs/getting-started
  - https://itsfoss.com/use-snap-packages-ubuntu-16-04/