Linux software packaging formats

Ready… Fight!

Quest for software management

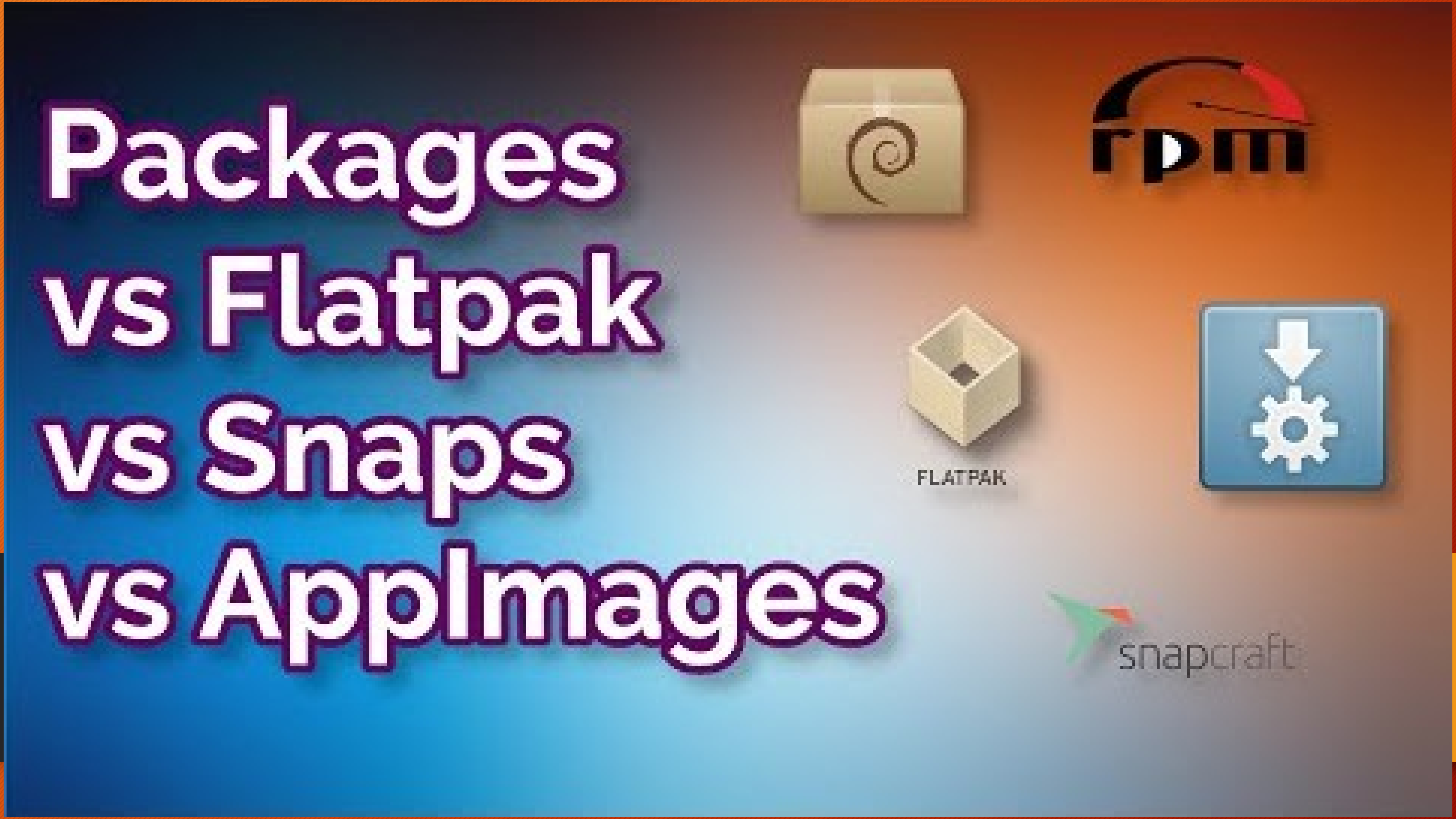
- Linux can run programs from anywhere, as long as all files and dependencies are available
  - Tarball anyone? (.tar.gz)
- You can download the source code for open-source application and compile and run on your system
- BUT:
  - How do you patch and update the software?
  - How do you keep track of what application is installed and where?
  - What about security? Are your apps running as root or as your privileged user account?
  - How do you uninstall an application?
    - Find all the files, change all the config files back, remove all links manually?

The battle between package formats

- Traditional formats:
  - .rpm, .deb, .yum, etc…
- Flatpack:
  - https://en.wikipedia.org/wiki/Flatpak
- Snaps (proprietor is Canonical):
  - https://en.wikipedia.org/wiki/Snap_(package_manager)
- Appimage:
  - https://en.wikipedia.org/wiki/AppImage

Watch this video:

https://www.youtube.com/watch?v=9HuExVD56Bo



Deb and rpm packages

- Binary format
  - Compiled for a particular system architecture (x86, x86_64, arm, etc.)
  - Maybe packaged for a specific distro
- Usually pulled from repository, made for specific distro
  - User can also download the .deb or .rmp file from internet and use the dpkg or rpm commands to install manually
- Has a descriptor that lists the dependencies and libraries it needs (not included in the package itself)
  - If not using a package manager to install, user should install all dependencies manually in order for the software to function

Deb and rpm packages

- Pros:
  - Fast (binary)
  - Space efficient
  - Can be pulled from repositories
- Cons:
  - Dependency issues (if not installed from repository for specific distro)
  - Developer needs to package application for various distros specifically
  - Developers need to maintain repository, or use the distro repository
    - Costly, time consuming, and distro may or may not accept the package into their repository

Flatpack

- Binary format
- Ship with their own libraries
    - Can also reference specific libraries instead of shipping very big packages
    - Solve dependency issues by shipping the right version of libraries
- Still space efficient, although less than native distro repository packages
- Have a repository system (called remote)
    - Example: flathub
- Sandboxed
- Usually GUI software is delivered in this format

Flatpack

- Pros
  - Fast
  - Solves dependency issue
  - Can run on many systems
- Cons
  - Since developer specifies the version of libraries it can ship with outdated, flawed, or vulnerable libraries
  - Use more disk space compared to native repository apps
    - Different apps my install their own version of the same library (duplicates)

Snaps

- Binary
- Self-contained with their dependencies and libraries
- Can ship CLI and server applications
- Can do delta updates (only download and update the files that have changed)
  - Updates can be applies while snap is running
  - Snap needs to be restarted to start using the new versions

Snaps

- Cons:
  - Do not support the main GUI desktop theme in most cases
  - Can’t make use of 3rd party repositories
    - All snaps are available via Snapcraft or SnapStore that are controlled by Canonical.
      - Not Open, or community based
    - Mainly available on Ubuntu
    - Other distros are deliberately avoiding or removing support
  - They are larger in size
  - Take longer to load

AppImage

- Appimage contains the entire package that includes:
  - Application binary
  - Libraries
  - Any runtime or other dependencies
- Portable: just click on it and it will run
  - Can run from anywhere on any system or distro
    - Example: put it on a usb drive and run it on any computer without having to install anything
- No repository, downloaded from internet
  - Appimage hub is a site dedicated to listing these application packages

Appimage

- Cons:
    - Can get very large
    - Can’t auto-update
    - Can’t do partial updates (the entire new version needs to be downloaded)
    - Is not aware of system settings and themes

.tar.gz (source code)

- Need to follow developers instructions in the package to install
  - Developer may include scripts to “make” and “make install” the application
- Need to install all other dependencies on the system manually
- Some Linux distros (such as Arch, Gentoo) they only ship source code
  - These distros provide tools that assists in obtaining, compiling, installing, and otherwise managing applications
- Advantages:
  - Can be compiled on any system given that the dependencies are also present for that system
  - Small (only text, and compressed using gunzip format)