Managing Software with APT, DNF, and Snap

Software management on modern Linux distributions

Speakers:

- 22120368 - Phan Thanh Tiến
- 22120375 - Lưu Thái Toàn
- 22120383 - Nguyễn Đăng Trí

Objectives

- Understand the role of package managers
- Add and remove a repository
- Search / install / remove packages
- Compare APT, DNF, and Snap
- Demo

Background: Package managers

- A tool for **discovery**, **install**, **upgrade**, and **removal** of system package (including apps and its libraries to run). *(Not limited to Linux systems! e.g., Homebrew on macOS, Chocolatey on Windows)*
- They perform dependency resolution, manage metadata and caches, and apply updates

  > Inspect dependencies using `apt` : `apt-cache depends vlc`

- They integrate with the OS packaging format (.deb / .rpm) and can influence system state (services, configs)
- Important for security (timely patches), reproducibility, and disk/resource sharing

- Types of package managers:
  - Imperative: modify system state directly (e.g., APT, DNF)
  - Application package manager: bundle app + dependencies, isolated from system (e.g., Snap, Flatpak)
  - Declarative: define desired state, system converges to it (e.g., Nix Package Manager)

Background: Repositories and packages

- Repository: a remote source (URL) that exposes package metadata and files
- Packages: versioned bundles (binary + metadata) built for the distro format
- Repo metadata (indexes) enable search, dependency resolution, and fast installs via caching
- Trust model: repositories are usually signed with GPG keys — verify before adding

Imperative package formats

`apt` and `dnf`

When install a package...

- Signature & integrity checks
- Resolve dependencies
- Install dependencies and the package
- Triggers and post-install integration
- Implications: DEB/RPM modify global filesystem and system state; upgrades can require config migration and service restarts

Why DEB cannot run on RPM-based systems (and vice versa)?

- Package archive layout (technical):
    - DEB: ar archive containing `debian-binary`, `control.tar.*` (control files & scripts), and `data.tar.*` (payload)
    - RPM: header metadata + cpio payload; header contains file lists, dependencies, provides, and scriptlets

- Both deb and rpm packages are contains of the binary file, metadata and scripts. They are different in the structure of the package and metadata, so the low level tools (dpkg and rpm) are different.
- But, if you have the binary file inside the package, you can extract it and run it on any system if the dependencies are satisfied.

linuxvox.com/blog/linux-deb-file/

man7.org/linux/man-pages/man5/deb.5.html

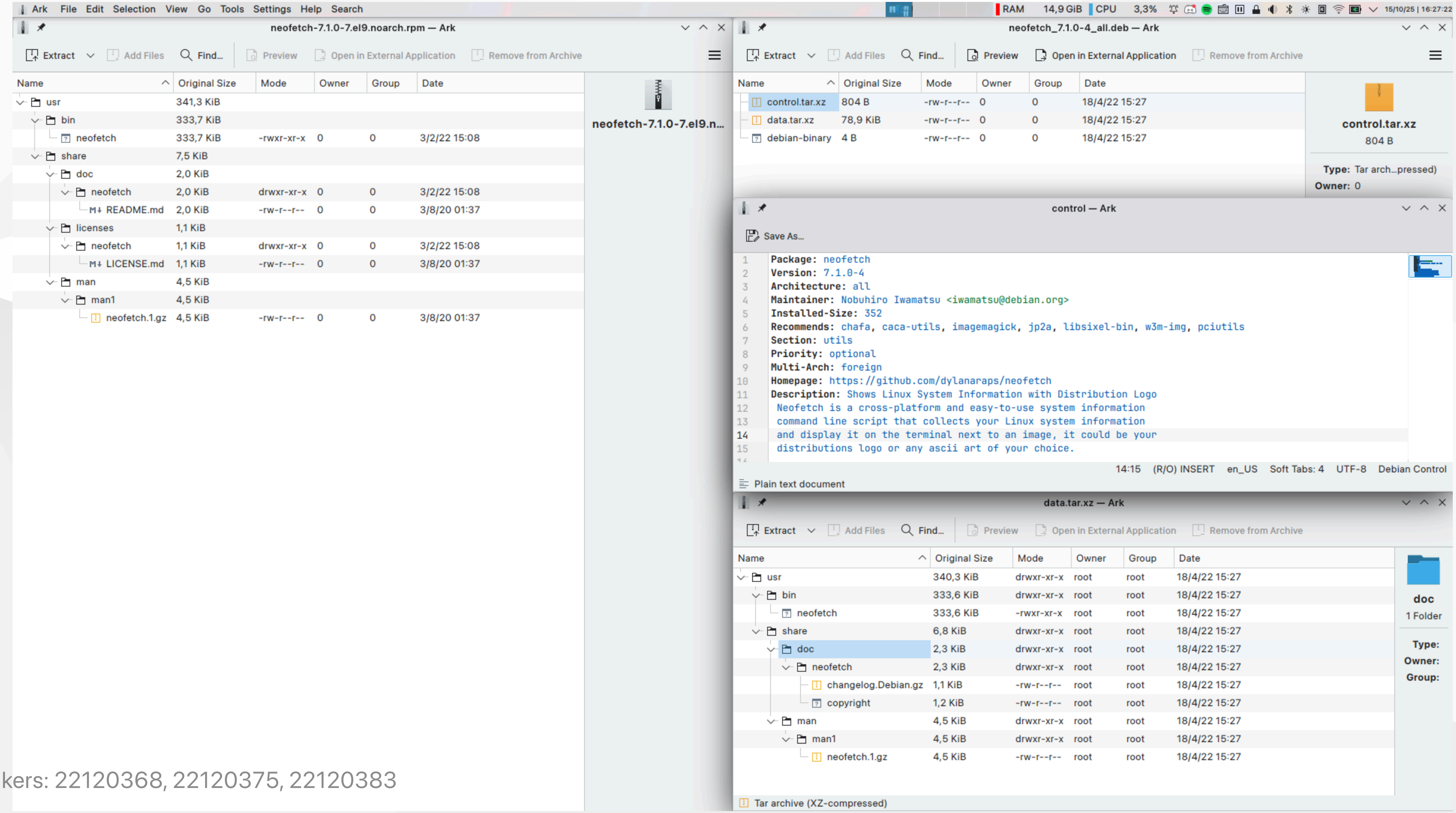
jfearn.fedorapeople.org/en-US/RPM/4/html/RPM_Guide/ch-package-structure.html

ftp.rpm.org/max-rpm/s1-rpm-file-format-rpm-file-format.html

Inspecting a .deb vs .rpm package

https://rhel.pkgs.org/9/epel-x86_64/neofetch-7.1.0-7.el9.noarch.rpm.html

https://debian.pkgs.org/12/debian-main-arm64/neofetch_7.1.0-4_all.deb.html



neofetch-7.1.0-7.el9.noarch.rpm — Ark

Name	Original Size	Mode	Owner	Group	Date
usr	341,3 KiB				
bin	333,7 KiB				
neofetch	333,7 KiB	-rwxr-xr-x	0	0	3/2/22 15:08
share	7,5 KiB				
doc	2,0 KiB				
neofetch	2,0 KiB	drwxr-xr-x	0	0	3/2/22 15:08
README.md	2,0 KiB	-rw-r--r--	0	0	3/8/20 01:37
licenses	1,1 KiB				
neofetch	1,1 KiB	drwxr-xr-x	0	0	3/2/22 15:08
LICENSE.md	1,1 KiB	-rw-r--r--	0	0	3/8/20 01:37
man	4,5 KiB				
man1	4,5 KiB				
neofetch.1.gz	4,5 KiB	-rw-r--r--	0	0	3/8/20 01:37

neofetch-7.1.0-7.el9.n...

neofetch_7.1.0-4_all.deb — Ark

Name	Original Size	Mode	Owner	Group	Date
control.tar.xz	804 B	-rw-r--r--	0	0	18/4/22 15:27
data.tar.xz	78,9 KiB	-rw-r--r--	0	0	18/4/22 15:27
debian-binary	4 B	-rw-r--r--	0	0	18/4/22 15:27

control.tar.xz
804 B
Type: Tar arch...pressed)
Owner: 0

control — Ark

```
Package: neofetch
Version: 7.1.0-4
Architecture: all
Maintainer: Nobuhiro Iwamatsu <iwamatsu@debian.org>
Installed-Size: 352
Recommends: chafa, caca-utils, imagemagick, jp2a, libsixel-bin, w3m-img, pciutils
Section: utils
Priority: optional
Multi-Arch: foreign
Homepage: https://github.com/dylanaraps/neofetch
Description: Shows Linux System Information with Distribution Logo
 Neofetch is a cross-platform and easy-to-use system information
 command line script that collects your Linux system information
 and display it on the terminal next to an image, it could be your
 distributions logo or any ascii art of your choice.
```

14:15 (R/O) INSERT en_US Soft Tabs: 4 UTF-8 Debian Control

data.tar.xz — Ark

Name	Original Size	Mode	Owner	Group	Date
usr	340,3 KiB	drwxr-xr-x	root	root	18/4/22 15:27
bin	333,6 KiB	drwxr-xr-x	root	root	18/4/22 15:27
neofetch	333,6 KiB	-rwxr-xr-x	root	root	18/4/22 15:27
share	6,8 KiB	drwxr-xr-x	root	root	18/4/22 15:27
doc	2,3 KiB	drwxr-xr-x	root	root	18/4/22 15:27
neofetch	2,3 KiB	drwxr-xr-x	root	root	18/4/22 15:27
changelog.Debian.gz	1,1 KiB	-rw-r--r--	root	root	18/4/22 15:27
copyright	1,2 KiB	-rw-r--r--	root	root	18/4/22 15:27
man	4,5 KiB	drwxr-xr-x	root	root	18/4/22 15:27
man1	4,5 KiB	drwxr-xr-x	root	root	18/4/22 15:27
neofetch.1.gz	4,5 KiB	-rw-r--r--	root	root	18/4/22 15:27

doc
1 Folder
Type:
Owner:
Group:

- Low-level tools and databases:
  - `dpkg` unpacks DEBs and updates `/var/lib/dpkg/` (status database); APT performs repo management and dependency solving
  - `rpm` manages the RPM DB under `/var/lib/rpm`; DNF/YUM orchestrate transactions using rpm metadata and repo data
- Maintainer scripts / scriptlets:
  - DEB: `preinst`, `postinst`, `prerm`, `postrm` — run at install/upgrade/remove
  - RPM: `%pre`, `%post`, `%preun`, `%postun` — run during package lifecycle
  - Scripts run as root and may create users, set permissions, enable services, perform migrations

Cross platform package managers

Snap and Flatpak

Snap packages

- Bundle the app and most of its dependencies into a single compressed file (squashfs).
- Some snaps use **shared content snaps** (like GNOME or KDE runtimes) to avoid duplicating large libraries.
- Managed by the `snapd` service, which handles installing, updating, and running snaps.
- When you run a snap:
  - `snapd` mounts the snap package as a virtual filesystem.
  - The app runs in a sandbox, isolated from the rest of the system.
  - Access to system resources is controlled by interfaces (permissions).

- Cross-distro compatibility:
  - Snaps do not rely on the host’s package manager or libraries (no .deb/.rpm needed).
  - As long as `snapd` is installed, snaps work the same way on any Linux distribution.
- Sandboxing:
  - Snaps run in a confined environment, limiting access to system resources for security.
  - Permissions are managed via interfaces that can be connected or disconnected.

snapcraft.io/docs/system-architecture

APT (Debian / Ubuntu)

- Frontends: apt, apt-get, apt-cache
- Repo files: `/etc/apt/sources.list` and `/etc/apt/sources.list.d/`
- Common commands:
    - Update metadata: `sudo apt update`
    - Search: `apt search <name>` or `apt-cache search <name>`
    - Install: `sudo apt install <package>`
    - Remove: `sudo apt remove <package>` (keep config) or `sudo apt purge <package>` (remove config)

DNF (Fedora / RHEL / CentOS / AlmaLinux)

- Successor to yum on many RPM-based distros
- Repo files: `/etc/yum.repos.d/*.repo`
- Common commands:
  - Update metadata: `sudo dnf makecache` or `sudo dnf check-update`
  - Search: `dnf search <name>`
  - Install: `sudo dnf install <package>`
  - Remove: `sudo dnf remove <package>`

Adding repositories

- Why: access newer versions, vendor packages, or 3rd-party software
- Debian/Ubuntu:
  - Add a PPA or a `.list` file in `/etc/apt/sources.list.d/`, import GPG key, then `sudo apt update`
- RHEL/CentOS/AlmaLinux:
  - Add a `.repo` file to `/etc/yum.repos.d/`, import GPG key, then `sudo dnf makecache`
  - Use `dnf config-manager --add-repo <repo-url>`
- Security: always verify repository GPG keys and prefer HTTPS where available

Snap (Canonical)

- Find: `snap find <name>`
- Install: `sudo snap install <snap-name>`
- Remove: `sudo snap remove <snap-name>`
- List installed: `snap list`

https://snapcraft.io/docs/snap-howto

Comparison: APT | DNF | Snap

APT	DNF	Snap
Package format: .deb	Package format: .rpm	Package format: snap bundle
Repo files: /etc/apt/sources.list(.d)	Repo files: /etc/yum.repos.d/*.repo	Managed by snapd (no distro repo files)
Strong dependency resolution (dpkg backend)	Dependency resolution with rich metadata, plugins	Bundles dependencies, isolated runtime (larger size)
System-focused packages, integrates with system services	System-focused packages, plugin ecosystem	App-focused, sandboxed, transactional installs

APT (Debian/Ubuntu)	DNF (Fedora/RHEL/AlmaLinux)	Snap (cross-distro)
Typical use: system packages, servers, libraries	Typical use: system packages, enterprise RHEL ecosystems	Typical use: desktop and some server apps for cross-distro delivery
Pros: mature, fast, small packages	Pros: modern metadata, modular repos	Pros: cross-distro, sandboxed, easy packaging
Cons: tied to Debian ecosystem	Cons: tied to RPM ecosystem	Cons: larger disk usage, sometimes slower start, requires snapd

Demo

Install a package that not exists in default repositories by adding a new repository

https://www.sublimetext.com/docs/linux_repositories.html

Debian/Ubuntu

Repository structure

```
deb [repository_url] [distribution] [component]
```

- `distribution` : Specified the distribution name of the Debian (e.g., stable, buster, focal) or Ubuntu (e.g., focal, jammy)
- `component` : Define the component which can be main, contrib, and non-free

Debian/Ubuntu

Add a new repository

Edit `/etc/apt/sources.list` | Use GUI software center

```
# add GPG key
wget -qO - https://download.sublimetext.com/sublimehq-pub.gpg | \
  sudo tee /etc/apt/keyrings/sublimehq-pub.asc > /dev/null

# .list file
echo "deb [signed-by=/etc/apt/keyrings/sublimehq-pub.asc] https://download.sublimetext.com/ apt/stable/" \
| sudo tee /etc/apt/sources.list.d/sublime-text.list >/dev/null

# .source file
echo -e \
"Types: deb
URIs: https://download.sublimetext.com/
Suites: apt/stable/
Architectures: amd64
Components: main
Signed-By: /etc/apt/keyrings/sublimehq-pub.asc" | sudo tee /etc/apt/sources.list.d/sublime-text.source

# apt-add-repository
sudo apt-add-repository "deb [signed-by=/etc/apt/keyrings/sublimehq-pub.asc] \
        https://download.sublimetext.com/ apt/stable/"
```

itslinuxfoss.com/add-debian-repository/

Debian/Ubuntu

Installing a software from added repositories

```
sudo apt update
apt search sublime-text | head -n 20
sudo apt install sublime-text
apt list --installed | grep sublime-text || true
sudo apt remove sublime-text

sudo add-apt-repository -r "deb https://download.sublimetext.com/ apt/stable/"
```

- GUI: Any software center (e.g., Ubuntu Software/GNONE Software/KDE Discover).
- Or Synaptic Package Manager.

CentOS/RHEL/AlmaLinux

- Adding/Disabling/Removing repo: https://gist.github.com/aelkz/0dc6864cd7f3665a2780b2a111ad1a49

```
sudo rpm -v --import \
  https://download.sublimetext.com/sublimehq-rpm-pub.gpg
sudo dnf config-manager --add-repo \
  https://download.sublimetext.com/rpm/stable/x86_64/sublime-text.repo
```

```
sudo dnf makecache
sudo dnf install -y sublime-text
dnf list installed | grep sublime-text || true
sudo dnf remove -y sublime-text
sudo dnf config-manager --remove-repo \
  https://download.sublimetext.com/rpm/stable/x86_64/sublime-text.repo
```

Snap

- Snap is a centralized app store, managed by Canonical. So there are no way to add/remove repository like APT or DNF.

```
sudo apt install -y snapd
snap find hello-world | head -n 10
sudo snap install hello-world
snap list | head -n 20
sudo snap remove hello-world
```

Trends & alternatives

- App distribution alternatives: Flatpak and AppImage (app sandboxing, desktop apps)
- Functional/declarative package managers: Nix / NixOS (focus on reproducibility, rollbacks)
- Containers change distribution of applications, but package managers remain important for system maintenance and shared libraries

References

- man apt, man apt-get, man dnf, man snap
- https://snapcraft.io/docs
- Flatpak docs: https://flatpak.org
- Nix/NixOS introduction: https://nixos.org

Thank you

Questions?