OpenPrinting

# Distribution-independent packaging of the printing stack and printer/scanner drivers

**Till Kamppeter – OpenPrinting, Soumyadeep Ghosh - Snapcrafters**

**Linux App Summit 2025, April 26, 2025**

# Motivation

- **Classic packaging** by distros (DEB,RPM, ...) produces packages which are **specific to the distro (and its version)** they are designed for.
- They are built exactly for the resources (libraries, desktop environment, ...) which the distro contains.
- The packages are usually **created by the devlopers of the distro**.
- Once a release of the distro is made, no new upstream versions or new upstream packages are done any more.
- **No easy access to the newest software**, especially when staying in LTS cyvles.
- Hardware enablement updates (new drivers) are also very limited.
- Immutable distros usually only allow adding desktop apps (Flatpak) and not system components or hardware drivers

# Motivation

- **At OpenPrinting we want to allow the delivery of**
  - Always **newest printer/scanner/MFP drivers**, also from manufacturers and for all distros
  - **Newest printing stack**, also for systems which come without printing support
  - **Secure installation of drivers from third parties** (manufacturers) without needing approval from distro maintainers
- **Snaps** for many desktop distros
- **OCI-containers** for servers, (immutable) desktop distros without Snap support, or Snap haters.
- Note that with Flatpak we (still) cannot package daemons and that AppImage has no security concept.

# Snap - Introduction

- **Sandboxed packaging**
- **OS-distribution-independent**
  - **You package and test once**, put your **Snap** into the **Snap Store**, and users of **any distro** (Ubuntu, Debian, SUSE, Red Hat, Windows, ...) can use it.
  - **All libraries and other dependencies** come with your Snap
- Your app runs in a **security shell** isolated from the host system
  - Communication to outside only via **well-defined interfaces**
  - **Snap Store has control**, has to explicitly permit "dangerous" interfaces
  - This way we can **trust third-party apps**
  - We are not dependent any more on distro maintainers for secure packages
- **User experience as with smartphone apps**

# Snap - Introduction

- Compressed and **GPG-signed read-only squashfs images** (immutable apps)
- Includes **metadata** in a ***.yaml** file
- Installed Snap has a **writable file system** area inside its confinement
- Come in **5 types** (app, os core, gadget, kernel, desktop session)
  - **Don’t fear the daemons, we snap them, too!**
- Support **transactional (atomic) updates** and **rollback**
- Can handle **binary diffs** for smaller download on upgrades
- **Available on multiple distros** and supported by default on all Ubuntu installs since Ubuntu 14.04

# Snap - Introduction

- **Confinement via**:
  - **AppArmor** (File system access rules)
  - **seccomp** (System call restrictions)
  - **Namespaces** (Separate resource spaces: PIDs, users, network, ...)
- **“root-safe”**
  - Applications can **run as root** but can not break out of the package confinement, **no need for specific user or group setup** to maintain security.
  - Example: **Daemon Snaps**
- **Storage-efficient**
  - Image stays compressed after install
  - **Core Snaps** and **content provider Snaps** hold common libraries and data files

# Snap - Introduction

- Communication is possible via **well-defined interfaces**: "network", "cups", "dbus", ...
- A "**plug**" has to be connected with a "**slot**" of the system or of another Snap in order to communicate
  - **“Safe” interfaces**
    - Ex.: “cups” which allows listing available printers and printing
    - **are auto-connected** when installing from Snap Store
  - **“Dangerous” interfaces**
    - Ex.: “cups-control” which allows creating/removing printers, delete all jobs ...
    - **need manual connection** or **permission** from Snap Store team for auto-connection

# Snap – CUPS

- **Complete immutable printing stack in one Snap**
- **Current upstream releases** of all components
  - CUPS
  - cups-filters
  - Ghostscript
  - QPDF
  - cups-browsed
- Provides interface **slots**: “cups”, “cups-control”
- **Plugs** interfaces: “network”, “network-bind”, “network-manager-observe”, “avahi-control”, “raw-usb”
- **System user/group** “snap_daemon” instead of “lp”

# Snap – CUPS

- “**cups-control**” interface: **Full admin** access to CUPS
  - **Snap mediation**: cupsd allows admin access from a Snap only it plugs “cups-control”
  - Considered “**dangerous**”, needs permission for auto-connect
  - For **printer setup tools**
- “**cups**” interface: **Printing-only** access to CUPS
  - **Requires Snap mediation** to work, therefore **we force use of CUPS Snap,** using Snap’s domain socket
  - Considered “**safe**”, so it gets auto-connected
  - For **applications which print**

# Snap – CUPS

- **CUPS 3.x**
  - **Sharing server**
    - Current CUPS Snap (or Printer Application Snap) will be the base
  - **Local server**
    - User daemon
    - Trigger by D-Bus?
    - Run permanently from login to logout?
  - **Snapping (snapcraft.yaml, scripts, …) in cups-sharing and cups-local repos**
    - Move snapping already into CUPS 2.5.x repo?

# Snap – CUPS

- **What is needed to use the CUPS Snap as print environment (claasic distro)?**
  - **Drivers for legacy/specialty printers** must be available as **Printer Applications**
    - **Classic drivers and PPDs** cannot get installed into the **immutable file system**
    - **All drivers** available on Debian are retro-fitted into **Printer Applications** available in the Snap Store (only Braille embossers missing)
    - **`legacy-printer-app`** of pappl-retrofit maps **classically installed CUPS printer drivers** into a **Printer Application**, especially also proprietary drivers.
    - **Look-up service for Printer Applications** on OpenPrinting web site
    - **“Add Printer”** of current printer setup tools and CUPS web interface **does not work**
  - Snap of CUPS 2.x contains **cups-browsed**, therefore **no special requirements for print dialogs**

# Snap – CUPS

- **What is needed for printing on an all-Snap distro (Ubuntu Core)?**
  - **CUPS Snap + ipp-usb Snap + CPDB CUPS backend Snap**
  - **Driverless (IPP)** printing
  - **Printer Applications Snaps** as drivers for legacy/specialty printers
  - **Applications**
    - Plug **“cups”** interface
    - Use **Common Print Dialog Backends (CPDB)**
    - Use **xdg-desktop-portal** (not all desktops/toolkits)
  - **Printer setup tools**
    - Plug **“cups-control”** interface

# Snap – Printer/Scanner Applications, ipp-usb

- Way of **distribution-independent packaging for printer/scanner driver** distribution
- **Plugs** interfaces: “avahi-control”, “home”, “network”, “network-bind”, “raw-usb”, “hardware-observe”
- **Kept up-to-date** with Snap update and versioning automation, applied to Debian’s packaging GitLab for drivers which are not maintained any more
- **ipp-usb** uses shell script working as “**UDEV observation daemon**” to launch ipp-usb when printer appears
  - Snap does not support UDEV rules
  - Script is based on “**`udevadm`**” command line tool, especially “**`udevadm monitor`**”
  - We will **investigate running ipp-usb permanently** instead

# OCI container images – Introduction

- Software running in a **container/sandbox**
  - **System software**
  - **Server/cloud** applications
  - **Restricted access** to other containers and host system
- Most well-known platform/tool is **Docker**
  - Has the container image “store” **Docker Hub**
- **ROCKs/rockcraft**
  - Easy **container image build similar to snapcraft**, works with Docker and others
  - **Ubuntu is base distro** for build and runtime, as with Snap
  - At runtime also **distroless** is supported

# OCI container images – Introduction

- **Why OCI containers?**
  - **Immutable desktop distributons**
    - Most immutable desktop distros do not support Snap
    - Many of them allow adding system software as OCI containers
    - Desktop apps are added as Flatpaks
  - **Server/cloud**
    - OCI containers are a standard format here

# OCI container images – Introduction

- **Tons of CUPS Images in Docker Hub**
  - All from **third-parties**, none of them from OpenPrinting
  - Can one **trust** these people?
  - Images can be **highly specialized**, only for a very restricted use case
- **=> We need general-purpose, “official” images from OpenPrinting**
  - **GSoC project by Rudra Pratap Singh** to create OCI images for OpenPrinting done in 2024
    - CUPS
    - Printer Applications
  - Winter-of-Code (2025) project by Vishal Patel
    - ipp-usb

# OCI container images – Introduction

- **All OCI images of OpenPrinting** created with **rockcraft**
  - **All build instructions in the `rockcraft.yaml` file**, same syntax as **`snapcraft.yaml`** of Snap, only what is specific to Snap or OCI image is different
  - **Image contains complete file system**, including the base distro and all dependencies, does not require any other images to be installed
  - Image can be created “**distroless**” without base distro, with only explicitly selected files
  - **Image is not immutable**, once installed any file can be modified
  - Image does not contain info about how to communicate with outside, **all permissions, mounts, … have to be supplied on the docker command line**

# OCI container images – Pseudo-immutable

- Images usually contain **system daemons, permanently running**
- The **daemons are running as root**, but have no free access to the outside
- **Not being immutable the containers are difficult to update to new software versions**
- For the container images of the **Printer Applications** and **ipp-usb** we have Prs to make them **pseudo-immutable**:
  - Image only contains the file system for the software itself
  - Software is running as unprivileged user, so not able to modify the file system of the image
  - As writable area a volume is mounted

# OCI container images – Invocation

- In contrary to Snaps, **OCI containers are not automatically correctly started** after installation
- They have to be started with a **command line specifying the way of communication with the outside**
- **Example `ipp-usb` (after PR for pseudo-immutability merged):**

```
sudo docker run -d --network host \          Same network as host
     -v /dev/bus/usb:/dev/bus/usb:ro \       Read access to USB device files
     -v ipp-usb-state:/var/ipp-usb \         Persistent storage
     -v ipp-usb-conf:/etc/ipp-usb \          Persistent config files
     --device-cgroup-rule='c 189:* rmw' \    Full access to USB printers
     --name ipp-usb \
     ipp-usb:latest
```

The container is running its own **`avahi-daemon`**, therefore no D-Bus permission needed

# Flatpak ?!

- It is well known that **Flatpak is designed only for desktop applications**, one cannot package system software/daemons with it.
- **All software from OpenPrinting** (CUPS, ipp-usb, Printer Applications) are **daemons**, so flatpaking them is not possible (at least for now).
- **Christian Hergert** has worked on a way to flatpak daemons and wrote in his blog: https://blogs.gnome.org/chergert/2024/05/07/system-extensions-from-flatpak/
- There was also discussion on the GUADEC 2024 in Denver, in a Flatpak BoF: https://openprinting.github.io/OpenPrinting-News-July-2024/#guadec-2024-in-denver
- **Changes on systemd are needed** to make it possible.
- If this gets solved we will start flatpaking at OpenPrinting ...

# Questions / Comments