®

OpenPrinting

Designing and Packaging Printer and Scanner Drivers

Till Kamppeter – OpenPrinting

September 20, 2021

What we had

- **Printer drivers**
  - PPD files
  - Filters, perhaps also backends
  - All has to be in CUPS-specific directories
- **Scanner drivers**
  - Shared libraries with SANE ABI in SANE-specific directories
- **Packaging**
  - Binaries were built specific to destination distro and packaged in DEB or RPM packages
  - For each distro drivers need to be built, packaged, and tested separately
  - As files need to be in specific directories drivers cannot be installed with CUPS in a Snap or with scanning user applications in Snaps

What we want

- **Sandboxed packaging – Snaps**
  - Distribution-independent: Install from Snap Store on any distro
  - More security: Every package with all its libraries and files in its own sandbox, fine-grained control for communication between packages
  - All-Snap distributions
- **But**
  - You cannot drop driver files into directories of a snapped CUPS or snapped user applications, Snaps do not see the system's files
  - Snaps only communicate via IP, D-Bus, domain socket (Snap interfaces)
- **Also**
  - CUPS is deprecating support for PPD files, working by itself only in driverless IPP mode.

The New Architecture

- Printer/Scanner Applications **emulating** an (driverless) **IPP device**
  - Easily snappable: Communicates only via IP
  - Multi-function device support, Printing, Scanning, Fax Out in single Snap
  - Web admin interface for vendor/device-specific GUI
  - Behaves like a network printer/scanner/multi-function device
- **CUPS for printing** (and fax out)
  - CUPS discovers and uses all driverless IPP printers it finds
  - CUPS spools jobs, does page management, converts job formats
- **IPP Scan/eSCL for scanning**
  - User apps scan on IPP scanners/Scanner Applications via IPP Scan/eSCL
  - Retro-fit user app Snap with sane-airscan backend later direct IPP scan

Development Tools

- **PAPPL**
  - **libpappl**: Library providing everything what Printer/Scanner Applications have in common
    - Daemon
    - Web admin interface
    - IPP server emulation
    - Job handling
    - Answering all IPP requests, especially get-printer-attributes
    - Printer discovery and setup
  - Only what is specific to the supported devices needs to be implemented

Developemnt Tools

- **cups-filters 2.x**
  - **libcupsfilters** (mostly done)
    - Filter functions
    - To convert data formats during print/scan job execution
    - Re-using the code of the CUPS filters: pdftopdf, pdftops, pstops, rastertops, rastertopdf, ...
    - Chaining filter function when conversion cannot be done with a single filter
    - All filter functions have the same interface, taking input/output streams, job attributes/options, printer capabilities, log function, and filter-specific parameters
    - Auxiliary functions, for IPP attribute handling, calling filter functions in chains, with pipes, embedding classic CUPS filters/backends …

Developemnt Tools

- **cups-filters 2.x**
  - **libppd** (Mostly done)
    - All PPD handling functions of libcups and some more
      - IPP attributes ↔ PPD option conversion, fully automatic
      - Find PPD for discovered printer, list available PPDs
      - Apply PostScript/PJL code in PPD to jobs
    - PPDs are deprecated in CUPS and everything PPD-supporting will be removed soon
    - For retro-fitting existing classic printer drivers without need of rewriting
  - **Customized build options** for the individual Snap (Planned)
    - No libppd, no libqpdf, Raster-only, no Ghostscript/Poppler, ...

Development Tools

- CUPS-driver-retro-fit library – **libpappl-retrofit**
  - **Encapsulate classic CUPS drivers** in a Printer Application Snap
  - Supports classic CUPS drivers: **PPDs**, **CUPS filters**, **CUPS backends**
  - Lists PPDs in a human-readable way, normalizing make/model, applying reg-exp for driver name, …
  - PPD auto-selection for given device ID by make, model, driver, PDLs, also using reg-exps.
  - Find best-suited PPD option settings for given job IPP attributes
  - Support CUPS extensions in PPDs: String, password, numeric, … options
  - Supports back- and side channel and discovery mode of CUPS backends
  - **Easy Printer Application creation** with minimum of C code

Development Tools

- **snapcraft**
  - Printer/Scanner Application packaged as a Snap → **Distro-independent**
  - Upload to Snap Store → **Easily available for everyone**
  - Snapcraft building is similar to RPM/DEB building:
    - Instruction file (snapcraft.yaml)
    - snapcraft tool builds the package according to this
  - In contrary to RPM/DEB all dependencies (libraries, …) included in Snap
  - **Advanced Security**: Snaps are isolated from each other and from the host system, communication only through defined interfaces: network, usb-raw, avahi-control, …

Development Tools

- **snapcraft**
  - Client (CUPS, SANE frontend) communicates only via IPP, Printer Application Snap communicates also with device
  - **Planned: snapcraft plugins and extensions**
    - To simplify snapping Printer/Scanner Applications
    - To avoid re-including common instructions in snapcraft.yaml
    - For constant quality
    - Easy maintenance
  - **Planned: Finding Snaps in the Snap store by hardware signature**
    - Driver auto-installation
    - Perhaps with help of OpenPrinting database?

Design Guidelines

- **CUPS driver/PPD retro-fit only for old, unmaintained drivers**
- 1 Printer/Scanner Application = 1 Snap
- Printer/Scanner/Fax support in a single Application, for multi-function devices
- Recommended: 1 Printer/Scanner Application per project or manufacturer/product line: Gutenprint, HPLIP, SANE, foo2zjs, Epson, Canon, Ricoh, …
- NOT 1 Application per device → A lot of clutter and code duplication
- 1 Printer/Scanner Application = 1 Port
- For more than 1 device on 1 Application use URI: **`ipp://localhost:<PORT>/ipp/print/<NAME>`**
- DNS-SD service names must be always the same, independent of order Application start at boot or of device discovery
- Web admin interface: Auto-setup, manual setup of additional devices/instances, configuration of options not accessible via IPP

Further Activity

- Despite of the incredible work of our whole team we did not finish on the tools and resources yet
- Continued development of the tools
  - PAPPL: Scanner support, String option support, Human-readable names
  - Finalization of cups-filters 2.x
  - Spin out cups-browsed in own project (and Printer Application Snap)
  - Gutenprint, HPLIP as native Printer Applications (not PPD-based retro-fit)
  - chroot jail to retro-fit closed-source classic printer/scanner drivers
  - Plug-in and extension for snapcraft for quickly and easily snapping Printer Applications
  - Finalizing the CUPS Snap (mainly waiting for work of snapd team, ETA: October 2021)

Questions / Comments