CUPS 2.4/2.5/3.0

Michael R Sweet, Lakeside Robotics Corporation
msweet@lakesiderobotics.ca
Agenda

• Whose CUPS is This?
• CUPS 2.4: Summary, Planning
• CUPS 2.5: Summary, Planning
• CUPS 3.0: Transition, Architecture, Breaking Up the Band, Library Cleanup, Challenges/Needs, Planning
Whose CUPS is This?

- After I left Apple in December 2019, development of CUPS at Apple slowed and many issues and pull requests were not resolved.
- In September 2020, I approached the OpenPrinting maintainers to fork Apple CUPS so that we could restart development.
- In March 2021, Apple contracted me to merge many of the OpenPrinting CUPS bug fixes back into Apple CUPS for a 2.3.5 release.
  - Apple's focus is currently only on bug fixes for CUPS 2.3.x.
  - OpenPrinting has taken over new CUPS feature development starting with CUPS 2.4.
Organization

• CUPS is a project of OpenPrinting whose core developers consist of the packagers and maintainers of CUPS on different operating systems/distributions

• Currently no explicit leadership structure, operating by consensus in email discussions

• Managing the project as a whole by consensus is OK, but has issues

• CUPS also depends on several other OpenPrinting projects (cups-filters in particular) to provide a complete printing solution for Linux, *BSD, etc., so some decisions about CUPS need a broader view
Leadership and Roles

• I have been the de-facto leader/release manager of CUPS for 23 years...
  • This has to change!

• Proposal: The core developers can take turns as the release manager for a single minor release cycle
  • Typically a minor release cycle lasts two years - one for development and one for maintenance
  • Development and maintenance overlap, so there will always be at least two release managers active (in case of emergencies)
Release Manager Role

- Responsibilities are:
  - Management of a feature release ("vM.N") milestone in Github
  - Coordination with/coaching of developers for bug fixes and features
  - Coordination with other release managers for bug fixes
  - Monitoring of CI builds
  - Creation of release tarballs and announcements as needed
  - A release manager's job is not to do all of the coding!
CUPS 2.4
Summary

• Over 90 issues fixed/PRs merged in CUPS 2.4 since CUPS 2.3.3op2
• Adds official AirPrint™/Mopria printer sharing support
• Adds `cupsSetOAuthCallback` API to support OAuth challenges (RFC 6750)
• Adds explicit container support (snapcraft for 2.4.0)
• Adds `pkg-config` support
• Deprecated (but doesn't remove) Kerberos and `cups-config`
I'm close to releasing the first beta of CUPS 2.4

Proposed schedule:

- 2.4-b1: September 24, 2021
- 2.4-rc1: October 8, 2021
- 2.4.0: October 22, 2021
- 2.4.1: January 2022
- 2.4.2: March 2022
After releasing 2.4.0 I'll create a v2.4.x branch from master

Bug fixes then get merged to master and v2.4.x

Features are only merged to master

Who will take over as the 2.4 release manager?
CUPS 2.5
Summary

• A few features are currently queued up:
  • OAuth support in cupsd (Issue #246)
  • Default OAuth callback for desktop (D-BUS API?)
  • TLS/X.509 improvements (Issue #99)
  • Centralized CUPS localization (Issue #216)
  • Docker, AppImage, other container technologies?
  • Who will be the 2.5 release manager?
Planning

• Proposed schedule:
  • 2.5-b1: September 2022
  • 2.5-rc1: October 2022
  • 2.5.0: November 2022
  • 2.5.1: January 2023
  • 2.5.2: March 2023
Planning

• Probably the last of the 2.x series

  • Should we commit to doing 2.5.x updates after 3.0.0 is out? If so, how long?

  • Important to coordinate X.509/OAuth functionality with 3.0 development to minimize differences and provide a clear migration path

• Need to enlist desktop developer(s) for OAuth UI

• Create separate project as needed for UI and D-Bus service that libcups can talk to
CUPS 3.0
Transition

User Applications

CUPS

PPD Files

Printer Drivers

IPP Everywhere™ PostScript Raster

User Applications

CUPS

PostScript Printer App

Printer Applications

IPP Everywhere™ PostScript Raster
Architecture

• Commands: lp, lpr, lpstat, cancel, etc.

• Local Server: runs as user, IPP domain sockets/D-BUS/XPC, only temporary queues, basic spooling and filtering/rasterization, limited job history

• Sharing Server: runs as root, IPP domain/TCP sockets, only permanent queues, advanced spooling and filtering/rasterization, job history and accounting, push/pull/release printing, authentication, web interface, configurability

• Library: libcups, as exists today
Architecture

User-Level

- User Applications
- CUPS Commands
- CUPS Local Server

System-Level

- CUPS Sharing Server
- Printer Applications

CUPS Library
Architecture

- Handles all discovery and communications with printers
- Handles authentication, authorization, consent, and notification UI
- Converts to/from PDF/raster as needed for printers
- Job history is limited to the current session/login
- No web interface
- Configuration limited to listing specific printers or servers that cannot be discovered via DNS-SD ("profiles")
Architecture

- Handles all communications with printers
- Authorization/consent/notification UI needs to be handled by client
- Converts to/from PDF/raster as needed for printers
- Push (to printer) and pull/release (from printer/proxy)
- Job history is configurable with accounting interface
- Web interface
- Configuration similar to existing cupsd
- OAuth token introspection (RFC 7662) and scopes for ACLs
Breaking Up the Band

• Split CUPS up into four projects?
  • CUPS library
  • Command-line tools
  • Local server
  • Sharing server

• Advantages: More closely matches current packaging of CUPS, decouples server and library features/bug fixes, path for libcups 2.x?

• Disadvantages: Complicates CI testing and user builds
Library Cleanup

- Major release gives us the opportunity to dump obsolete APIs in the CUPS library
  - `cupsGetClasses`, `cupsGetPrinters`, `httpMD5Password`, PPD APIs, etc.
- Can also take the opportunity to clean up/normalize the APIs
  - `cupsGetDest2s` renamed to `cupsGetDest`, etc.
- This would mean bumping the major version of the library, which will cause binary compatibility issues for existing executables
  - Do we care? If so, important to have a story for libcups 2.x support (how long, packaging guidance, etc.)
Challenges/Needs

• Much broader scope and integration than the original CUPS work

• Desktop support - need to uplift GNOME/KDE/XFCE desktops to new D-Bus API for printing, authorization, consent UI

• Need developers to work on the local and sharing servers, desktop UI/services
  • Can probably use/adapt PAPPL code for the core server bits
  • Much of the print dialog work can be repurposed
  • Probably have existing authorization/notification UI we can use
Challenges/Needs

• Graphics libraries - current PDF tools/libraries have problematic licenses or other limitations

  • My PDFio library (https://www.msweet.org/pdfio) can handle PDF filtering but doesn't support rasterization

  • Xpdf/Poppler are GPL2 but provide command-line programs we can run (not ideal but workable)

  • MµPDF (https://mupdf.com) is AGPL and doesn't offer a stable API

  • PDFium (https://opensource.google/projects/pdfium) is BSD-licensed but requires the Chromium build system/tools
Planning

• Release Manager: Me?

• Proposed schedule:
  • January 2022: New projects (as needed) and/or branching
    • Developing 2.5 and 3.0 in parallel, lots of common work
  • October 2022: 2.5.0 out, focus on 3.0 development
  • March 2023: 3.0-b1
  • October 2023: 3.0.0
Open Discussion