OpenPrinting

Also you can make printing just work!

Interactive Workshop

Till Kamppeter – OpenPrinting

Opportunity Open Source 4.0, August 29, 2026

What you will learn

- To **download CUPS** from GitHub, **build** and **test** it
- To do **modifications** on the source code
- To **clone the GitHub repository** of CUPS, **push** your modifications and have the **CI tests** running
- To prepare a **Pull Request** (PR)
- To troubleshoot using **CUPS' log files**
- To **emulate an IPP printer**, by the simple command line utility **`ippeveprinter`** but also by **`go-mfp`**, a comprehensive simulator of an IPP multi-function printer

Prepare your laptop

You will need a **developer** setup

- **Operating system**
  - Highly recommended: **Linux**, like **Ubuntu**, **Fedora**, or **Debian**
  - **Linux under WSL** on a Windows box should also work
  - You can install a **virtual machine (VM)** to not mess up your system (if you have an **immutable** distribution, you **need a VM** for developing).
- **Tools**
  - **Plain text file editor**
    - Standard keyboard control (**`Ctrl+C`**, **`Ctrl+V`**, ...): GNOME/KDE/COSMIC text editor, VS Code, VS Codium, Zed, micro, ...
    - App-specific control: vi/vim, emacs, nano, ...

Prepare your laptop

- **Tools**
  - **Compilers**
    - **`gcc`** (for C) and **`g++`** (for C++)
  - Build system tools
    - **`automake`**, **`autoconf`**
  - Command line interface for **GIT**
    - **`git`**
  - On **Debian** or **Ubuntu** install these tools with
    `sudo apt install build-essential git`

Download the source code

- **Go to https://github.com/OpenPrinting/cups**
- In the right column of the screen you find a "**Releases**" section, this leads you to the latest release. **Download** the **`cups-2.4.19.tar.gz`** file.
- Uncompress it and enter its main directory:

```
tar -xvzf cups-2.4.19.tar.gz
cd cups-2.4.19
```

- If you have a **project-oriented editor** (Integrated Development Environment, **IDE**, like **VS Code/Codium**, **Zed**, ...), **open this source directory** with it.
- Read the files **`README.md`** and **`INSTALL.md`**, the latter tells you which components (dependencies) you have to install and how to build the source code

Install the needed components/ Dependencies

- According to **`INSTALL.md`**, use the following commands:
- On **Debian** or **Ubuntu**:

```
sudo apt-get install autoconf build-essential libavahi-client-dev \
     libgnutls28-dev libkrb5-dev libnss-mdns libpam-dev \
     libsystemd-dev libusb-1.0-0-dev zlib1g-dev
```

- On **Fedora**:

```
sudo dnf install autoconf make automake gcc gcc-c++ krb5-devel avahi-devel \
     gnutls-devel krb5-libs nss-mdns pam-devel \
     systemd-devel libusb1-devel zlib-devel
```

- On any other distro make sure not to only install the required libraries themselves but also their **development (header) files** (**`-dev`** or **`-devel`** packages).

Compile the source code

- Continuing in **`INSTALL.md`**, use the following commands:
- Configuring the build:
  `./configure`
- Build (compile) all the components:
  `make`
- Run the unit tests to check for regressions (otional):
  `make test`
- Preview what gets installed where:
  `make DESTDIR=`pwd`/tmp install`
- Actually install into your system:
  `make install`

Compile the source code

- By default, **`make install`** overwrites your system's CUPS, to let it install to somewhere else do:
  `./configure –prefix=/opt`
- To run your CUPS then, make your libcups being found:
  `export LD_LIBRARY_PATH=/opt/lib`
- Run executables with full path:
  `/opt/bin/lpstat`
- Either stop your system's cupsd or let your cupsd run on a port other than 631.

GitHub: Fork, clone, modify, push, PR

- **Fork** the CUPS repo to your GitHub
- **Clone** your copy with the git clone ... command
- You have CUPS source of **current development**, not the last release
- You can now build, test, and **modify** the code
- **See** your changes with **`git diff`**
- While modifying commit your changes in useful pieces and tell in the commit messgae what you have done (use **`git commit`**).
- **Upload** your changes to your repo copy with **`git push`**
- **Create a PR** for your changes based on your repo

Debugging: Use error_log

- **CUPS logs** all its activity (not necessarily only errors) in the **`/var/log/cups/error_log`** file
- Standard logging is not very verbose, you need to switch to debug mode: **`cupsctl –debug-logging`**
- Reproduce the bug and have a look into the log (or attch it to your bug report)
- Check for suspicious actions

Debugging: Emulate a test printer

- **`ippeveprinter`**: Minimalistic IPP Everywhere printer emulator
- **`go-mfp`**: Detailed emulation of a modern multi-function printer, soon including the scanner part.
  https://github.com/OpenPrinting/go-mfp

Debugging: Emulate a test printer

- **Google Summer of Code:**
  https://github.com/LinuxFoundationGSoC/ProjectIdeas/wiki/GSoC-2026-OpenPrinting
- OpenPrinting **GitHub**: https://github.com/OpenPrinting/
  Michael Sweet's GitHub: https://github.com/michaelrsweet/
- OpenPrinting **mailing list**: printing-architecture@lists.linux.dev
  https://lore.kernel.org/printing-architecture/
  https://subspace.kernel.org/lists.linux.dev.html
- OpenPrinting **News**: https://openprinting.github.io/news
- **LinkedIn**: https://www.linkedin.com/company/openprinting/
- **Mastodon**: #OpenPrinting