Installation manual: Snort 3 + LibDAQ + LibML (SnortML) on Ubuntu 24.04

Reproducible, from-scratch setup for building Snort 3 with SnortML support in an isolated prefix under your home directory.

Assumptions

- Ubuntu 24.04 installed in a VirtualBox VM.
- You want all compiled artifacts under: ~/snortml-research/install
- You use GitHub SSH (your user: iRootTsar).
- You build from source: snort3/snort3, snort3/libdaq, snort3/libml.

0) Create project root folder structure

Run:

```
mkdir -p ~/snortml-research/{repos,build,install,datasets,pcaps,scripts,results,docs}
```

Meaning:

- repos: clones (snort3/libml/libdaq + thesis repo)
- install: isolated prefix (binaries + libs)
- datasets, pcaps: inputs
- scripts: helper scripts (verify, extraction, etc)
- results: outputs
- docs: documentation (put this guide and/or PDFs here)

1) Fix apt “file:///cdrom” repository

If sudo apt update shows a cdrom error, disable that source line:

```
sudo sed -i.bak '/file:\/\/\/cdrom/s/^[[:space:]]*deb[[:space:]]/# deb /' /etc/apt/sources.list
head -n 5 /etc/apt/sources.list
sudo apt update
```

Expected: sudo apt update completes with no file:/cdrom errors.

Also recommended (VirtualBox): Settings -> Storage -> Optical Drive -> set to “Empty” (remove ISO).

2) Install build dependencies (Ubuntu 24.04)

Enable Universe and install required tools/libs.

```
sudo apt install -y software-properties-common
sudo add-apt-repository -y universe
sudo apt update
```

Install packages:

```
sudo apt install -y \
  git ca-certificates curl wget \
  build-essential pkg-config \
  cmake ninja-build \
  autoconf automake libtool \
  bison flex \
  python3 python3-venv python3-pip \
  libpcap-dev \
  libdumbnet-dev \
  libluajit-5.1-dev luajit \
  libssl-dev openssl \
  zlib1g-dev liblzma-dev \
  libhwloc-dev \
  libsqlite3-dev \
  uuid-dev \
  libpcre2-dev \
  libhyperscan-dev \
  libmnl-dev libnetfilter-queue-dev \
  tcpdump
```

Sanity checks:

```
git --version
cmake --version
luajit -v
dpkg -l | grep -E 'libpcap-dev|libdumbnet-dev|libluajit|libhyperscan-dev|libpcre2-dev' || true
```

Notes: Snort’s “DNET” requirement on Ubuntu is satisfied by libdumbnet-dev (library libdumbnet).

3) Configure GitHub SSH in the VM (clone/push without passwords)

Generate key:

```
ssh-keygen -t ed25519 -C "irootts-snortml-thesis"
eval "$(ssh-agent -s)"
ssh-add ~/.ssh/id_ed25519
cat ~/.ssh/id_ed25519.pub
```

Add the printed public key to GitHub: GitHub -> Settings -> SSH and GPG keys -> New SSH key

Test:

```
ssh -T git@github.com
```

Expected: “Hi iRootTsar! You’ve successfully authenticated…”.

4) Fork model (recommended)

On GitHub (web):

- Fork snort3/snort3 -> iRootTsar/snort3
- Fork snort3/libdaq -> iRootTsar/libdaq
- Fork snort3/libml -> iRootTsar/libml
- Create a separate repo for your thesis work (docs/scripts/results), for example: iRootTsar/snortml-nonhttp-thesis

Why: forks keep your patches clean and easy to PR later. Thesis repo holds documentation, scripts, datasets metadata, experiment configs.

5) Clone repos into ~/snortml-research

```
cd ~/snortml-research/repos

git clone git@github.com:iRootTsar/libdaq.git
git clone git@github.com:iRootTsar/libml.git
git clone git@github.com:iRootTsar/snort3.git

cd ~/snortml-research
git clone git@github.com:iRootTsar/snortml-nonhttp-thesis.git thesis
```

Add upstream remotes and create working branches from upstream/master:

```
cd ~/snortml-research/repos/libdaq
git remote add upstream https://github.com/snort3/libdaq.git
git fetch upstream
git checkout -b thesis/non-http-ml upstream/master

cd ~/snortml-research/repos/libml
git remote add upstream https://github.com/snort3/libml.git
git fetch upstream
git checkout -b thesis/non-http-ml upstream/master

cd ~/snortml-research/repos/snort3
git remote add upstream https://github.com/snort3/snort3.git
git fetch upstream
git checkout -b thesis/non-http-ml upstream/master
```

If you want to reset to upstream/master later (safe way):

```
cd ~/snortml-research/repos/libdaq
git fetch upstream
git reset --hard upstream/master

cd ~/snortml-research/repos/libml
git fetch upstream
git reset --hard upstream/master

cd ~/snortml-research/repos/snort3
git fetch upstream
git reset --hard upstream/master
```

6) Define the isolated install prefix (critical)

For every build terminal, export these:

```
export ROOT="$HOME/snortml-research"
export PREFIX="$ROOT/install"

export PATH="$PREFIX/bin:$PATH"
export LD_LIBRARY_PATH="$PREFIX/lib:$PREFIX/lib/x86_64-linux-gnu:${LD_LIBRARY_PATH:-}"
export PKG_CONFIG_PATH="$PREFIX/lib/pkgconfig:$PREFIX/lib/x86_64-linux-gnu/pkgconfig:${PKG_CONFIG_PATH:-}"
export CMAKE_PREFIX_PATH="$PREFIX:${CMAKE_PREFIX_PATH:-}
```

Optional: persist in ~/.bashrc (append once):

```
cat >> ~/.bashrc <<'EOF'
export ROOT="$HOME/snortml-research"
export PREFIX="$ROOT/install"
export PATH="$PREFIX/bin:$PATH"
export LD_LIBRARY_PATH="$PREFIX/lib:$PREFIX/lib/x86_64-linux-gnu:${LD_LIBRARY_PATH:-}"
export PKG_CONFIG_PATH="$PREFIX/lib/pkgconfig:$PREFIX/lib/x86_64-linux-gnu/pkgconfig:${PKG_CONFIG_PATH:-}"
export CMAKE_PREFIX_PATH="$PREFIX:${CMAKE_PREFIX_PATH:-}"
EOF

source ~/.bashrc
```

7) Build and install (order matters)

Build order: LibDAQ, LibML, Snort 3.

7.1) Build LibDAQ

```
cd ~/snortml-research/repos/libdaq
git clean -xfd

./bootstrap
./configure --prefix="$PREFIX"
make -j"$(nproc)"
make install
```

Quick check:

```
ls -la "$PREFIX/lib" | grep daq || true
pkg-config --modversion libdaq || true
```

7.2) Build LibML

```
cd ~/snortml-research/repos/libml
git clean -xfd

./configure.sh --prefix="$PREFIX"
cd build
make -j"$(nproc)"
make install
```

Quick check:

```
ls -la "$PREFIX/lib" | grep -i ml || true
```

7.3) Build Snort 3

```
cd ~/snortml-research/repos/snort3
git clean -xfd
rm -rf build

./configure_cmake.sh --prefix="$PREFIX"
cd build
make -j"$(nproc)"
make install
```

8) Verification (must pass before you start thesis engineering work)

8.1) Verify version + linked components

```
"$PREFIX/bin/snort" -V
```

8.2) Verify dynamic links

```
ldd "$PREFIX/bin/snort" | grep -E 'daq|ml|dumbnet|dnet|luajit|pcap|ssl|pcre|hs' || true
```

8.3) Verify SnortML inspector is present

```
"$PREFIX/bin/snort" --help | grep -i snort_ml || true
```

9) Create a reusable verification script (recommended)

Create:

```
cat > ~/snortml-research/scripts/verify_install.sh <<'EOF'
#!/usr/bin/env bash
set -euo pipefail

ROOT="$HOME/snortml-research"
PREFIX="$ROOT/install"

echo "=== snort -V ==="
"$PREFIX/bin/snort" -V

echo
echo "=== ldd (filtered) ==="
ldd "$PREFIX/bin/snort" | grep -E 'daq|ml|dumbnet|dnet|luajit|pcap|ssl|pcre|hs' || true

echo
echo "=== snort_ml presence (help output) ==="
"$PREFIX/bin/snort" --help | grep -i snort_ml || true
EOF
```

Run:

```
chmod +x ~/snortml-research/scripts/verify_install.sh
~/snortml-research/scripts/verify_install.sh
```

10) Where to place documentation + how to push

Recommended:

```
Put this file at: ~/snortml-research/thesis/docs/installation_manual.md
```

Example:

```
cd ~/snortml-research/thesis
mkdir -p docs
nano docs/installation_manual.md
```

Then commit and push:

```
cd ~/snortml-research/thesis
git add docs/installation_manual.md
git commit -m "Add installation manual"
git push origin main
```

If you later add a PDF in docs:

```
git add docs/Installation_Manual.pdf
git commit -m "Add installation manual PDF"
git push origin main
```

11) Clean rebuild procedure (when you want a clean slate for builds only)

This resets build outputs without deleting your clones.

```
rm -rf ~/snortml-research/install/*
cd ~/snortml-research/repos/libdaq && git clean -xfd
cd ~/snortml-research/repos/libml  && git clean -xfd
cd ~/snortml-research/repos/snort3 && git clean -xfd && rm -rf build
```

Then rebuild in the normal order (LibDAQ -> LibML -> Snort 3).