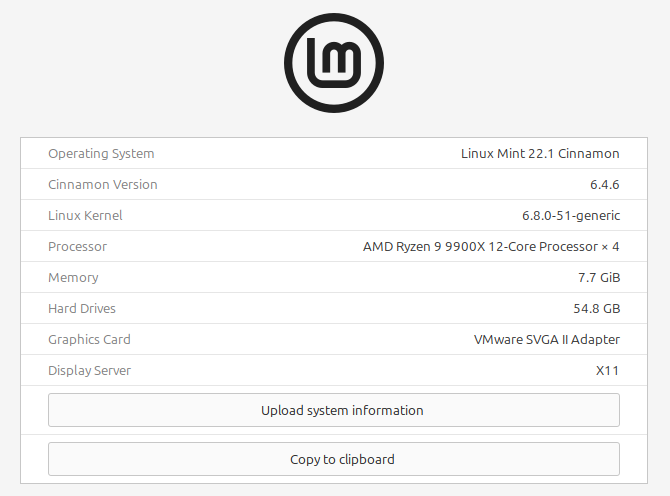
# Compilation guide for loki2 on Linux

Target OS: Linux Mint VM (Ubuntu-based)



**Requires:** Internet access; full binaries for OpenSSL and YARA (download instructions provided in this guide); the Rust toolchain (rust and cargo); and all dependencies specified in Cargo.toml.

Requirements:

* Internet access
* Full binaries for **OpenSSL** and **YARA** (build steps included below)
* Rust toolchain (rust, cargo)

## Steps to compile

### Step 1: Install Rust

user@user-virtual-machine:~/Documents$ curl --proto '=https' --tlsv1.2 -sSf https://sh.rustup.rs | sh

When prompted, select the **default installation option**.

After installation, reload the environment:

user@user-virtual-machine:~/Documents$ source $HOME/.cargo/env

### Step 2: Download loki2 and signature base

user@user-virtual-machine:~/Documents$ sudo apt install git

user@user-virtual-machine:~/Documents$ git clone https://github.com/Neo23x0/Loki2.git ./loki2

user@user-virtual-machine:~/Documents$ git clone https://github.com/Neo23x0/signature-base signature-base/

Create a symbolic link inside the Loki2 directory pointing to the signature-base:

user@user-virtual-machine:~/Documents$ cd loki2

user@user-virtual-machine:~/Documents/loki2$ ln -s ../signature-base/ ./signatures

### Step 3: Set Up Environment for musl Static Compilation

Export environment variables for musl builds:

user@user-virtual-machine:~/Documents/loki2$ export CC=musl-gcc

export TARGET\_PREFIX=/usr/x86\_64-linux-musl

export CFLAGS="-static -I/usr/x86\_64-linux-musl/include"

export YARA\_INCLUDE\_DIR=/usr/x86\_64-linux-musl/include

export LIBRARY\_PATH=/usr/x86\_64-linux-musl/lib

export PKG\_CONFIG\_PATH=/usr/x86\_64-linux-musl/lib/pkgconfig

### Step 4: Install Build Tools and Compile Dependencies

To prepare the build environment for musl and ensure proper linking of YARA/OpenSSL/etc., install the required toolchain and set up missing symlinks for musl compatibility.

Install the required build tools:

user@user-virtual-machine:~/Documents/loki2$ sudo apt-get update

sudo apt-get install automake libtool make gcc pkg-config flex bison clang musl-tools linux-libc-dev wget

Create required symlinks for musl to access kernel headers:

user@user-virtual-machine:~/Documents/loki2$ sudo mkdir -p /usr/include/x86\_64-linux-musl

sudo ln -s /usr/include/x86\_64-linux-gnu/asm /usr/include/x86\_64-linux-musl/asm

sudo ln -s /usr/include/asm-generic /usr/include/x86\_64-linux-musl/asm-generic

sudo ln -s /usr/include/linux /usr/include/x86\_64-linux-musl/linux

cd ..

### Step 5: Build and Install OpenSSL (Static)

user@user-virtual-machine:~/Documents$ wget -nv -O openssl.tar.gz https://www.openssl.org/source/openssl-1.1.1p.tar.gz

mkdir openssl && cd openssl

tar --strip=1 -xzf ../openssl.tar.gz

./config --prefix=$TARGET\_PREFIX no-afalgeng no-async no-capieng no-dso no-shared no-sock no-ui

make

sudo make install\_sw

cd ..

### Step 5: Build and Install YARA (Static)

user@user-virtual-machine:~/Documents$ wget -nv -O yara.tar.gz https://github.com/VirusTotal/yara/archive/v4.2.3.tar.gz

mkdir yara && cd yara

tar --strip=1 -xzf ../yara.tar.gz

./bootstrap.sh

LDFLAGS="$(pkg-config --static --libs libcrypto)" ./configure --with-crypto --disable-shared --prefix=$TARGET\_PREFIX

make

sudo make install

cd ..

### Step 7: Compile Loki2

user@user-virtual-machine:~/Documents/loki2$ export RUSTFLAGS="-L/usr/x86\_64-linux-musl/lib -lcrypto -lssl"

rustup target add x86\_64-unknown-linux-musl

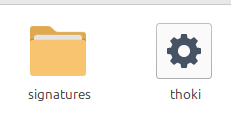
user@user-virtual-machine:~/Documents/loki2$ cargo build --target x86\_64-unknown-linux-musl --release --verbose

The portable loki ELF file is located at /home/user/Documents/loki2/target/x86\_64-unknown-linux-musl/release

## Notes for Deployment on Fresh Systems:

On the system, place the following in the same directory:

* The statically built loki ELF binary
* The signatures folder (with the content of the signature-base inside)



Give the loki ELF file the execute permission (chmod +x ./loki)

This is sufficient to run loki without system-wide dependencies.