# Guide d'Installation d'un Cross-Compiler pour Développement d'OS

Fiakx

## Table des matières

# 1 Introduction

Développer un système d'exploitation nécessite des outils spécifiques, notamment un cross-compiler. Ce document explique comment installer et configurer un cross-compiler pour architecture x86_64.

# 2 Prérequis

— Distribution Linux (Ubuntu/Debian ou Arch Linux)
— Accès internet
— 10 Go d'espace disque libre
— Droits administrateur (sudo)

# 3 Installation des dépendances

## 3.1 Sur Ubuntu/Debian

```
sudo apt update
sudo apt install -y build-essential bison flex libgmp3-dev libmpc-dev libmpfr-dev texinfo libisl-dev nasm grub2 xorriso qemu qemu-system-x86
```

## 3.2 Sur Arch Linux

```
sudo pacman -Syu --noconfirm base-devel bison flex gmp libmpc mpfr texinfo isl nasm grub xorriso qemu qemu-arch-extra
```

# 4 Construction du Cross-Compiler

## 4.1 Configuration initiale

```
export PREFIX="$HOME/opt/cross"
export TARGET=x86_64-elf
export PATH="$PREFIX/bin:$PATH"
mkdir -p $HOME/src
cd $HOME/src
```

### 4.2 Installation de Binutils

```
wget https://ftp.gnu.org/gnu/binutils/binutils-2.38.tar.xz
tar xf binutils-2.38.tar.xz
mkdir build-binutils
cd build-binutils
../binutils-2.38/configure --target=$TARGET --prefix="$PREFIX" --with-sysroot --disable-nls --disable-werror
make -j$(nproc)
make install
cd ..
```

### 4.3 Installation de GCC

```
wget https://ftp.gnu.org/gnu/gcc/gcc-11.2.0/gcc-11.2.0.tar.xz
tar xf gcc-11.2.0.tar.xz
mkdir build-gcc
cd build-gcc
../gcc-11.2.0/configure --target=$TARGET --prefix="$PREFIX" --disable-nls --enable-languages=c,c++ --without-headers
make all-gcc -j$(nproc)
make all-target-libgcc -j$(nproc)
make install-gcc
make install-target-libgcc
cd ..
```

## 5 Configuration de l'environnement

```
echo 'export PATH="$HOME/opt/cross/bin:$PATH"' >> ~/.bashrc
source ~/.bashrc
```

## 6 Vérification

```
x86_64-elf-gcc --version
```

# 7 Outils supplémentaires

## 7.1 NASM

```
sudo apt install nasm  # Ubuntu/Debian
sudo pacman -S nasm   # Arch Linux
```

## 7.2 QEMU

```
sudo apt install qemu-system-x86  # Ubuntu/Debian
sudo pacman -S qemu qemu-arch-extra  # Arch Linux
```

## 7.3 GRUB

```
sudo apt install grub2  # Ubuntu/Debian
sudo pacman -S grub    # Arch Linux
```

# 8 Projet de base

```
mkdir -p myos/src/{boot,kernel}
touch myos/src/kernel/main.c
touch myos/src/boot/boot.asm
```

## 8.1 Makefile

```
CC=x86_64-elf-gcc
ASM=nasm
CFLAGS=-ffreestanding -nostdlib -Wall -Wextra

all: myos.iso

build/kernel.o: src/kernel/main.c
  $(CC) $(CFLAGS) -c $< -o $@

build/boot.o: src/boot/boot.asm
  $(ASM) -f elf64 $< -o $@

myos.bin: build/boot.o build/kernel.o
  $(CC) -T linker.ld $^ -o $@ -nostdlib -lgcc
```

```
15
16 myos.iso: myos.bin
17   mkdir -p isodir/boot/grub
18   cp myos.bin isodir/boot/
19   echo 'menuentry "MyOS" { multiboot /boot/myos.bin }' >
      isodir/boot/grub/grub.cfg
20   grub-mkrescue -o myos.iso isodir
21
22 clean:
23   rm -rf build/*.o myos.bin myos.iso isodir
24
25 .PHONY: all clean
```

# 9 Conclusion

Pour approfondir :

— https://wiki.osdev.org/
— https://gcc.gnu.org/onlinedocs/
— https://sourceware.org/binutils/docs-2.38/