# Script pour construire un kernel XEN Ubuntu

Ce script construit Xen, un noyau Linux 64 bits de l'arborescence Raspberry Pi, et intègre un rootfs Ubuntu 20.04 64 bits minimal pour le Raspberry Pi 4. Une version récente d'Ubuntu est requise pour exécuter le script de construction. Une connexion Internet est requise. 8 Go de RAM ou plus sont recommandés et 10 Go d'espace disque libre.

Usage:

```
./rpixen.sh
```

Lorsque le script est terminé, flasher l'image sur la carte SD avec (par exemple) :

```
umount /dev/sdX1
umount /dev/sdX2
sudo dd if=rpixen.img of=/dev/sdX bs=8M
sync
```

Pour se connecter utiliser le nom d'utilisateur Dornerworks et le mot de passe Dornerworks.

Pour installer un bureau graphique, exécuter la commande `sudo apt install ubuntu-desktop` et redémarrer.

```
#!/bin/bash -Eeux
# -E Si défini, l'interruption ERR est héritée par les fonctions shell.
# -u Traite les variables non définies comme une erreur lors de la
substitution.
# -x Affiche les commandes et leurs arguments au fur et à mesure de leur
exécution.
# -e Quitte immédiatement si une commande se termine avec un état différent
de zéro.
# # Construction d'une Image XEN ubuntu
#
set -o pipefail

fail_handler () {
    BUILDLINE="$1"
}

trap 'fail_handler ${LINENO}' ERR

WRKDIR=$(pwd)/
SCRIPTDIR=$(cd $(dirname $0) && pwd)/
```

```
USERNAME=dornerworks
PASSWORD=dornerworks
SALT=dw
HASHED_PASSWORD=$(perl -e "print crypt(\"${PASSWORD}\",\"${SALT}\");")
HOSTNAME=ubuntu
UBUNTUVERSION="20.04.3"

BUILD_ARCH=${1:-arm64}

sudo apt install device-tree-compiler tftpd-hpa flex bison qemu-utils kpartx
git curl qemu-user-static binfmt-support parted bc libncurses5-dev libssl-
dev pkg-config python acpica-tools wget gettext

# la variable $BASH_SOURCE est une variable BASH initialisée avec le path
d'exécution du script
ARM64_TOOLCHAIN_WRKDIR=$(pwd)/
ARM64_TOOLCHAIN_SCRIPTDIR=$(cd $(dirname ${BASH_SOURCE}) && pwd)/

if ! which ccache > /dev/null; then
    sudo apt install ccache
fi

if [ ! -h /usr/lib/ccache/aarch64-none-linux-gnu-gcc ]; then
    sudo ln -s ../../bin/ccache /usr/lib/ccache/aarch64-none-linux-gnu-gcc
fi

if [ ! -h /usr/lib/ccache/aarch64-none-linux-gnu-g++ ]; then
    sudo ln -s ../../bin/ccache /usr/lib/ccache/aarch64-none-linux-gnu-g++
fi


if [ -z "${ARM64_TOOLCHAIN_VERSION-}" ]; then

    ARM64_TOOLCHAIN_VERSION="9.2-2019.12"
    ARM64_TOOLCHAIN_FILENAME=gcc-arm-${ARM64_TOOLCHAIN_VERSION}-x86_64-
aarch64-none-linux-gnu

    if [ ! -d ${ARM64_TOOLCHAIN_SCRIPTDIR}${ARM64_TOOLCHAIN_FILENAME} ];
then
        cd ${ARM64_TOOLCHAIN_SCRIPTDIR}
        if [ ! -s ${ARM64_TOOLCHAIN_FILENAME}.tar.xz ]; then
            wget https://developer.arm.com/-/media/Files/downloads/gnu-
a/${ARM64_TOOLCHAIN_VERSION}/binrel/${ARM64_TOOLCHAIN_FILENAME}.tar.xz
        fi
        tar -xf ${ARM64_TOOLCHAIN_FILENAME}.tar.xz
        cd ${ARM64_TOOLCHAIN_WRKDIR}
    fi

PATH=/usr/lib/ccache:${ARM64_TOOLCHAIN_SCRIPTDIR}${ARM64_TOOLCHAIN_FILENAME}
/bin:$(echo ${PATH} | sed 's|/usr/lib/ccache:||g')
```

```
fi



DTBFILE=bcm2711-rpi-4-b.dtb
if [ "${BUILD_ARCH}" == "arm64" ]; then
    DTBXENO=pi4-64-xen
else
    ARM32_TOOLCHAIN_WRKDIR=$(pwd)/
    ARM32_TOOLCHAIN_SCRIPTDIR=$(cd $(dirname ${BASH_SOURCE}) && pwd)/

    if ! which ccache > /dev/null; then
        sudo apt install ccache
    fi

    if [ ! -h /usr/lib/ccache/arm-none-linux-gnueabihf-gcc ]; then
        sudo ln -s ../../bin/ccache /usr/lib/ccache/arm-none-linux-
gnueabihf-gcc
    fi

    if [ ! -h /usr/lib/ccache/arm-none-linux-gnueabihf-g++ ]; then
        sudo ln -s ../../bin/ccache /usr/lib/ccache/arm-none-linux-
gnueabihf-g++
    fi

    if [ -z "${ARM32_TOOLCHAIN_VERSION-}" ]; then

        ARM32_TOOLCHAIN_VERSION="9.2-2019.12"
        ARM32_TOOLCHAIN_FILENAME=gcc-arm-${ARM32_TOOLCHAIN_VERSION}-x86_64-
arm-none-linux-gnueabihf

        if [ ! -d ${ARM32_TOOLCHAIN_SCRIPTDIR}${ARM32_TOOLCHAIN_FILENAME} ];
then
            cd ${ARM32_TOOLCHAIN_SCRIPTDIR}
            if [ ! -s ${ARM32_TOOLCHAIN_FILENAME}.tar.xz ]; then
                wget https://developer.arm.com/-/media/Files/downloads/gnu-
a/${ARM32_TOOLCHAIN_VERSION}/binrel/${ARM32_TOOLCHAIN_FILENAME}.tar.xz
            fi
            tar -xf ${ARM32_TOOLCHAIN_FILENAME}.tar.xz
            cd ${ARM32_TOOLCHAIN_WRKDIR}
        fi

PATH=/usr/lib/ccache:${ARM32_TOOLCHAIN_SCRIPTDIR}${ARM32_TOOLCHAIN_FILENAME}
/bin:$(echo ${PATH} | sed 's|/usr/lib/ccache:||g')

    fi
    DTBXENO=pi4-32-xen
fi
XEN_ADDR=0x00200000
```

```
# Clone sources
if [ ! -d firmware ]; then
    mkdir -p firmware/boot
    cd firmware/boot
    wget
https://raw.githubusercontent.com/raspberrypi/firmware/master/boot/fixup4.da
t
    wget
https://raw.githubusercontent.com/raspberrypi/firmware/master/boot/fixup4cd.
dat
    wget
https://raw.githubusercontent.com/raspberrypi/firmware/master/boot/fixup4db.
dat
    wget
https://raw.githubusercontent.com/raspberrypi/firmware/master/boot/fixup4x.d
at
    wget
https://raw.githubusercontent.com/raspberrypi/firmware/master/boot/start4.el
f
    wget
https://raw.githubusercontent.com/raspberrypi/firmware/master/boot/start4cd.
elf
    wget
https://raw.githubusercontent.com/raspberrypi/firmware/master/boot/start4db.
elf
    wget
https://raw.githubusercontent.com/raspberrypi/firmware/master/boot/start4x.e
lf
    cd ${WRKDIR}
fi

if [ ! -d xen ]; then
    git clone --depth=1 --branch RELEASE-4.15.1
git://xenbits.xen.org/xen.git
fi

if [ ! -d linux ]; then
    git clone --depth 1 --branch rpi-5.10.y
https://github.com/raspberrypi/linux.git linux
    cd linux
    cat > 0001-Add-Xen-overlay-for-the-Pi-4.patch << EOF
From: Corey Minyard <cminyard@mvista.com>
Date: Wed, 17 Nov 2021 12:41:32 -0500
Subject: [PATCH] Add Xen overlay for the Pi 4

Signed-off-by: Corey Minyard <cminyard@mvista.com>
---
 arch/arm/boot/dts/overlays/Makefile           |  4 +-
 .../boot/dts/overlays/pi4-32-xen-overlay.dts  | 21 ++++++++++
 .../boot/dts/overlays/pi4-64-xen-overlay.dts  | 39 ++++++++++++++++++++
```

```
 3 files changed, 63 insertions(+), 1 deletion(-)
 create mode 100644 arch/arm/boot/dts/overlays/pi4-32-xen-overlay.dts
 create mode 100644 arch/arm/boot/dts/overlays/pi4-64-xen-overlay.dts

diff --git a/arch/arm/boot/dts/overlays/Makefile
b/arch/arm/boot/dts/overlays/Makefile
index a6b0d9ea0385..4cfe5885eca1 100644
--- a/arch/arm/boot/dts/overlays/Makefile
+++ b/arch/arm/boot/dts/overlays/Makefile
@@ -244,7 +244,9 @@ dtbo-$(CONFIG_ARCH_BCM2835) += \
    w1-gpio-pullup.dtbo \
    w5500.dtbo \
    wittypi.dtbo \
-   wm8960-soundcard.dtbo
+   wm8960-soundcard.dtbo \
+   pi4-32-xen.dtbo \
+   pi4-64-xen.dtbo

 targets += dtbs dtbs_install
 targets += $(dtbo-y)
diff --git a/arch/arm/boot/dts/overlays/pi4-32-xen-overlay.dts
b/arch/arm/boot/dts/overlays/pi4-32-xen-overlay.dts
new file mode 100644
index 000000000000..47afa6ac24b7
--- /dev/null
+++ b/arch/arm/boot/dts/overlays/pi4-32-xen-overlay.dts
@@ -0,0 +1,21 @@
+// Xen configuration for Pi 4
+/dts-v1/;
+/plugin/;
+
+/ {
+    compatible = "brcm,bcm2711";
+
+    fragment@0 {
+        target-path = "/chosen";
+        __overlay__ {
+            #address-cells = <0x1>;
+            #size-cells = <0x1>;
+            xen,xen-bootargs = "console=dtuart dtuart=/soc/serial@7e215040
sync_console dom0_mem=512M dom0_max_vcpus=1 bootscrub=0";
+
+            dom0 {
+                compatible = "xen,linux-zimage", "xen,multiboot-module";
+                reg = <0x00400000 0x01800000>;
+            };
+        };
+    };
+};
diff --git a/arch/arm/boot/dts/overlays/pi4-64-xen-overlay.dts
b/arch/arm/boot/dts/overlays/pi4-64-xen-overlay.dts
```

```
new file mode 100644
index 000000000000..6c499a831db6
--- /dev/null
+++ b/arch/arm/boot/dts/overlays/pi4-64-xen-overlay.dts
@@ -0,0 +1,39 @@
+// Xen configuration for Pi 4
+/dts-v1/;
+/plugin/;
+
+/ {
+    compatible = "brcm,bcm2711";
+
+    fragment@0 {
+        target-path = "/chosen";
+        __overlay__ {
+            #address-cells = <0x1>;
+            #size-cells = <0x1>;
+            xen,xen-bootargs = "console=dtuart dtuart=/soc/serial@7e215040
sync_console dom0_mem=512M bootscrub=0";
+
+            dom0 {
+                compatible = "xen,linux-zimage", "xen,multiboot-module";
+                reg = <0x00480000 0x01780000>;
+            };
+        };
+    };
+
+    /* Introduce a dummy device node to make Xen map the framebuffer */
+    fragment@1 {
+        target-path = "/";
+        __overlay__ {
+            fb_bus {
+                compatible = "simple-bus";
+                ranges;
+                #address-cells = <2>;
+                #size-cells = <1>;
+                fb_mem {
+                    compatible = "dummy";
+                    status = "okay";
+                    reg = <0x0 0x3b400000 0x04c00000>;
+                };
+            };
+        };
+    };
+};
--
2.34.0
EOF

    git am 0001-Add-Xen-overlay-for-the-Pi-4.patch
```

```
    cd ${WRKDIR}
fi

# Build xen
if [ ! -s ${WRKDIR}xen/xen/xen ]; then
    cd ${WRKDIR}xen
    if [ ! -s xen/.config ]; then
        cat > xen/arch/arm/configs/rpi4_defconfig << EOF
CONFIG_DEBUG=y
CONFIG_SCHED_ARINC653=y
CONFIG_EARLY_UART_CHOICE_8250=y
CONFIG_EARLY_UART_BASE_ADDRESS=0xfe215040
CONFIG_EARLY_UART_8250_REG_SHIFT=2
EOF
        make -C xen XEN_TARGET_ARCH=arm64 CROSS_COMPILE=aarch64-none-linux-
gnu- rpi4_defconfig
    fi
    make XEN_TARGET_ARCH=arm64 CROSS_COMPILE=aarch64-none-linux-gnu- dist-
xen -j $(nproc)
    cd ${WRKDIR}
fi

# Build Linux
cd ${WRKDIR}linux
if [ "${BUILD_ARCH}" == "arm64" ]; then
    if [ ! -s ${WRKDIR}linux/.build-arm64/.config ]; then
        # utilize kernel/configs/xen.config fragment
        make O=.build-arm64 ARCH=arm64 CROSS_COMPILE=aarch64-none-linux-gnu-
bcm2711_defconfig xen.config
    fi
    make O=.build-arm64 ARCH=arm64 CROSS_COMPILE=aarch64-none-linux-gnu- -j
$(nproc) broadcom/${DTBFILE}
    make O=.build-arm64 ARCH=arm64 CROSS_COMPILE=aarch64-none-linux-gnu- -j
$(nproc) overlays/${DTBXENO}.dtbo
    if [ ! -s ${WRKDIR}linux/.build-arm64/arch/arm64/boot/Image ]; then
        echo "Building kernel. This takes a while. To monitor progress, open
a new terminal and use \"tail -f buildoutput.log\""
        make O=.build-arm64 ARCH=arm64 CROSS_COMPILE=aarch64-none-linux-gnu-
-j $(nproc) > ${WRKDIR}buildoutput.log 2> ${WRKDIR}buildoutput2.log
    fi
elif [ "${BUILD_ARCH}" == "armhf" ]; then
    if [ ! -s ${WRKDIR}linux/.build-arm32/.config ]; then
        # utilize kernel/configs/xen.config fragment
        make O=.build-arm32 ARCH=arm CROSS_COMPILE=arm-none-linux-gnueabihf-
bcm2711_defconfig xen.config
    fi
    make O=.build-arm32 ARCH=arm CROSS_COMPILE=arm-none-linux-gnueabihf- -j
$(nproc) ${DTBFILE}
    make O=.build-arm32 ARCH=arm CROSS_COMPILE=arm-none-linux-gnueabihf- -j
$(nproc) overlays/${DTBXENO}.dtbo
    if [ ! -s ${WRKDIR}linux/.build-arm32/arch/arm/boot/zImage ]; then
```

```
        echo "Building kernel. This takes a while. To monitor progress, open
a new terminal and use \"tail -f buildoutput.log\""
        make O=.build-arm32 ARCH=arm CROSS_COMPILE=arm-none-linux-gnueabihf-
-j $(nproc) zImage modules dtbs > ${WRKDIR}buildoutput.log 2>
${WRKDIR}buildoutput2.log
    fi
fi
cd ${WRKDIR}


if [ ! -d bootfiles ]; then
    mkdir bootfiles
fi

cp ${WRKDIR}firmware/boot/fixup4*.dat ${WRKDIR}firmware/boot/start4*.elf
bootfiles/

mkdir -p bootfiles/overlays
if [ "${BUILD_ARCH}" == "arm64" ]; then
    cp ${WRKDIR}linux/.build-arm64/arch/arm64/boot/dts/broadcom/${DTBFILE}
bootfiles/
    cp ${WRKDIR}linux/.build-
arm64/arch/arm64/boot/dts/overlays/${DTBXENO}.dtbo bootfiles/overlays
elif [ "${BUILD_ARCH}" == "armhf" ]; then
    cp ${WRKDIR}linux/.build-arm32/arch/arm/boot/dts/${DTBFILE} bootfiles/
    cp ${WRKDIR}linux/.build-
arm32/arch/arm/boot/dts/overlays/${DTBXENO}.dtbo bootfiles/overlays
fi

cat > bootfiles/cmdline.txt <<EOF
console=hvc0 clk_ignore_unused root=/dev/mmcblk0p2 rootwait
EOF

# https://www.raspberrypi.org/documentation/configuration/config-txt/boot.md
# the boot image must be named kernel8.img for the fsbl to load it in 64-bit
mode
# Xen must be placed on a 2M boundary
cat > bootfiles/config.txt <<EOF
kernel=kernel8.img
arm_64bit=1
kernel_address=${XEN_ADDR}
dtoverlay=${DTBXENO}
enable_gic=1
#disable_overscan=1
# Enable audio (loads snd_bcm2835)
dtparam=audio=on
[pi4]
max_framebuffers=2
[all]
enable_jtag_gpio=1
```

```
enable_uart=1
init_uart_baud=115200
EOF

# 26MiB worth of zeros
dd if=/dev/zero of=bootfiles/kernel8.img bs=1024 count=26624

# Assuming xen is less than 2MiB in size
dd if=${WRKDIR}xen/xen/xen of=bootfiles/kernel8.img bs=1024 conv=notrunc

if [ "${BUILD_ARCH}" == "arm64" ]; then
    # Assuming linux is less than 23.5MiB in size
    # Image is offset by 2.5MiB from the beginning of the file
    dd if=${WRKDIR}linux/.build-arm64/arch/arm64/boot/Image
of=bootfiles/kernel8.img bs=1024 seek=2560 conv=notrunc
elif [ "${BUILD_ARCH}" == "armhf" ]; then
    # Assuming linux is less than 24MiB in size
    # Image is offset by 2MiB from the beginning of the file
    dd if=${WRKDIR}linux/.build-arm32/arch/arm/boot/zImage
of=bootfiles/kernel8.img bs=1024 seek=2048 conv=notrunc
fi

if [ -d /media/${USER}/boot/ ]; then
    cp -r bootfiles/* /media/${USER}/boot/
    sync
fi

ROOTFS=ubuntu-base-${UBUNTUVERSION}-base-${BUILD_ARCH}-prepped.tar.gz
if [ ! -s ${ROOTFS} ]; then
  WRKDIR=$(pwd)/
  ARCH=${1:-arm64}
  UBUNTUVERSION=${2:-20.04.3}

  sudo apt install qemu-user-static

  # Download Ubuntu Base file system (https://wiki.ubuntu.com/Base)
  ROOTFSURL=http://cdimage.ubuntu.com/ubuntu-
base/releases/${UBUNTUVERSION}/release/
  ROOTFS=ubuntu-base-${UBUNTUVERSION}-base-${ARCH}.tar.gz
  if [ ! -s ${ROOTFS} ]; then
      curl -OLf ${ROOTFSURL}${ROOTFS}
  fi

  MNTRAMDISK=/mnt/ramdisk/
  MNTROOTFS=${MNTRAMDISK}qemu-${ARCH}-rootfs/

  IMGFILE=ubuntu-base-${UBUNTUVERSION}-base-${ARCH}-prepped.tar.gz

  if [ -s ${IMGFILE} ]; then
      ROOTFS=${IMGFILE}
  fi
```

```
  IMGFILE=${MNTRAMDISK}${IMGFILE}

  sudo mkdir -p ${MNTRAMDISK}
  sudo mount -t tmpfs -o size=3g tmpfs ${MNTRAMDISK}

  sudo mkdir -p ${MNTROOTFS}
  sudo tar -C ${MNTROOTFS} -xf ${ROOTFS}

  sudo mkdir -p ${MNTROOTFS}
  sudo mount -o bind /proc ${MNTROOTFS}proc
  sudo mount -o bind /dev ${MNTROOTFS}dev
  sudo mount -o bind /dev/pts ${MNTROOTFS}dev/pts
  sudo mount -o bind /sys ${MNTROOTFS}sys
  sudo mount -o bind /tmp ${MNTROOTFS}tmp

  if [ "${ARCH}" == "arm64" ]; then
      sudo cp $(which qemu-aarch64-static) ${MNTROOTFS}usr/bin/
  elif [ "${ARCH}" == "armhf" ]; then
      sudo cp `which qemu-arm-static` ${MNTROOTFS}usr/bin/
  fi

# /etc/resolv.conf is required for internet connectivity in chroot. It will
get overwritten by dhcp, so don't get too attached to it.
  sudo chroot ${MNTROOTFS} bash -c 'echo "nameserver 8.8.8.8" >
/etc/resolv.conf'
  sudo chroot ${MNTROOTFS} bash -c 'echo "nameserver 2001:4860:4860::8888"
>> /etc/resolv.conf'

  sudo sed -i -e "s/# deb /deb /" ${MNTROOTFS}etc/apt/sources.list
  sudo chroot ${MNTROOTFS} bash -Eeuxc "DEBIAN_FRONTEND=noninteractive apt-
get update"

# Install the dialog package and others first to squelch some warnings
  sudo chroot ${MNTROOTFS} bash -Eeuxc "DEBIAN_FRONTEND=noninteractive apt-
get -y install dialog apt-utils"
  sudo chroot ${MNTROOTFS} bash -Eeuxc "DEBIAN_FRONTEND=noninteractive apt-
get -y upgrade"
  sudo chroot ${MNTROOTFS} bash -Eeuxc "DEBIAN_FRONTEND=noninteractive apt-
get -y install systemd systemd-sysv sysvinit-utils sudo udev rsyslog kmod
util-linux sed netbase dnsutils ifupdown isc-dhcp-client isc-dhcp-common
less nano vim net-tools iproute2 iputils-ping libnss-mdns iw software-
properties-common ethtool dmsetup hostname iptables logrotate lsb-base lsb-
release plymouth psmisc tar tcpd libsystemd-dev symlinks uuid-dev libc6-dev
libncurses-dev libglib2.0-dev build-essential bridge-utils zlib1g-dev patch
libpixman-1-dev libyajl-dev libfdt-dev libaio-dev git libusb-1.0-0-dev
libpulse-dev libcapstone-dev libnl-route-3-dev openssh-sftp-server qemu-
system-x86 python3-dev linux-firmware-raspi2"
  sudo chroot ${MNTROOTFS} bash -Eeuxc "DEBIAN_FRONTEND=noninteractive apt-
get clean"
```

```
  sudo cat > ${MNTROOTFS}etc/systemd/system/regenerate_ssh_host_keys.service
<< EOF
[Unit]
Description=Regenerate SSH host keys
Before=ssh.service
ConditionFileIsExecutable=/usr/bin/ssh-keygen

[Service]
Type=oneshot
ExecStartPre=-/bin/dd if=/dev/hwrng of=/dev/urandom count=1 bs=4096
ExecStartPre=-/bin/sh -c "/bin/rm -f -v /etc/ssh/ssh_host_*_key*"
ExecStart=/usr/bin/ssh-keygen -A -v
ExecStartPost=/bin/systemctl disable regenerate_ssh_host_keys

[Install]
WantedBy=multi-user.target
EOF

  sudo chroot ${MNTROOTFS} systemctl enable regenerate_ssh_host_keys.service
  cd ${WRKDIR}
  sudo sync
  sudo umount ${MNTROOTFS}proc || true
  sudo umount ${MNTROOTFS}dev/pts || true
  sudo umount ${MNTROOTFS}dev || true
  sudo umount ${MNTROOTFS}sys || true
  sudo umount ${MNTROOTFS}tmp || true
  sudo tar -czf ${IMGFILE} -C ${MNTROOTFS} $(ls ${MNTROOTFS}) || true
  sudo chown $USER:$USER ${IMGFILE} || true
  sudo rm -rf ${MNTROOTFS} || true
  mv ${IMGFILE} . || true
  sudo umount ${MNTRAMDISK} || true
  sudo rmdir ${MNTRAMDISK} || true
fi

MNTRAMDISK=/mnt/ramdisk/
MNTROOTFS=/mnt/rpi-arm64-rootfs/
MNTBOOT=${MNTROOTFS}boot/
IMGFILE=${MNTRAMDISK}rpixen.img

sudo mkdir -p ${MNTRAMDISK}
sudo mount -t tmpfs -o size=3g tmpfs ${MNTRAMDISK}

qemu-img create ${IMGFILE} 2048M
/sbin/parted ${IMGFILE} --script -- mklabel msdos
MB_TO_SECTOR=2048
/sbin/parted ${IMGFILE} --script -- mkpart primary fat32 $(( 1 *
${MB_TO_SECTOR} ))s $(( 129 * ${MB_TO_SECTOR} - 1 ))s
/sbin/parted ${IMGFILE} --script -- mkpart primary ext4 $(( 129 *
${MB_TO_SECTOR} ))s -1025s

LOOPDEVS=$(sudo kpartx -avs ${IMGFILE} | awk '{print $3}')
```

```
LOOPDEVBOOT=/dev/mapper/$(echo ${LOOPDEVS} | awk '{print $1}')
LOOPDEVROOTFS=/dev/mapper/$(echo ${LOOPDEVS} | awk '{print $2}')

sudo mkfs.vfat ${LOOPDEVBOOT}
sudo mkfs.ext4 ${LOOPDEVROOTFS}

sudo fatlabel ${LOOPDEVBOOT} boot
sudo e2label ${LOOPDEVROOTFS} RpiUbuntu

sudo mkdir -p ${MNTROOTFS}
sudo mount ${LOOPDEVROOTFS} ${MNTROOTFS}

sudo tar -C ${MNTROOTFS} -xf ${ROOTFS}

sudo mkdir -p ${MNTROOTFS}
if ! mount | grep ${LOOPDEVROOTFS}; then
  sudo mount ${LOOPDEVROOTFS} ${MNTROOTFS}
fi
sudo mkdir -p ${MNTBOOT}
sudo mount ${LOOPDEVBOOT} ${MNTBOOT}
sudo mount -o bind /proc ${MNTROOTFS}proc
sudo mount -o bind /dev ${MNTROOTFS}dev
sudo mount -o bind /dev/pts ${MNTROOTFS}dev/pts
sudo mount -o bind /sys ${MNTROOTFS}sys
sudo mount -o bind /tmp ${MNTROOTFS}tmp

sudo cp -r bootfiles/* ${MNTBOOT}

cd ${WRKDIR}linux
if [ "${BUILD_ARCH}" == "arm64" ]; then
    sudo --preserve-env PATH=${PATH} make O=.build-arm64 ARCH=arm64
CROSS_COMPILE=aarch64-none-linux-gnu- INSTALL_MOD_PATH=${MNTROOTFS}
modules_install > ${WRKDIR}modules_install.log
elif [ "${BUILD_ARCH}" == "armhf" ]; then
    sudo --preserve-env PATH=${PATH} make O=.build-arm32 ARCH=arm
CROSS_COMPILE=arm-none-linux-gnueabihf- INSTALL_MOD_PATH=${MNTROOTFS}
modules_install > ${WRKDIR}modules_install.log
fi
cd ${WRKDIR}

# Build Xen tools

if [ "${BUILD_ARCH}" == "arm64" ]; then
    LIB_PREFIX=aarch64-linux-gnu
    CROSS_PREFIX=aarch64-none-linux-gnu
    XEN_ARCH=arm64
elif [ "${BUILD_ARCH}" == "armhf" ]; then
    LIB_PREFIX=arm-linux-gnueabihf
    CROSS_PREFIX=arm-none-linux-gnueabihf
    XEN_ARCH=arm32
```

```
fi

# Change the shared library symlinks to relative instead of absolute so they
play nice with cross-compiling
sudo chroot ${MNTROOTFS} symlinks -c /usr/lib/${LIB_PREFIX}/

cd ${WRKDIR}xen

# TODO: --with-xenstored=oxenstored

# Ask the native compiler what system include directories it searches
through.
SYSINCDIRS=$(echo $(sudo chroot ${MNTROOTFS} bash -c "echo | gcc -E -Wp,-v -
o /dev/null - 2>&1" | grep "^ " | sed "s|^ /| -isystem${MNTROOTFS}|"))
SYSINCDIRSCXX=$(echo $(sudo chroot ${MNTROOTFS} bash -c "echo | g++ -x c++ -
E -Wp,-v -o /dev/null - 2>&1" | grep "^ " | sed "s|^ /| -
isystem${MNTROOTFS}|"))

CC="${CROSS_PREFIX}-gcc --sysroot=${MNTROOTFS} -nostdinc ${SYSINCDIRS} -
B${MNTROOTFS}lib/${LIB_PREFIX} -B${MNTROOTFS}usr/lib/${LIB_PREFIX}"
CXX="${CROSS_PREFIX}-g++ --sysroot=${MNTROOTFS} -nostdinc ${SYSINCDIRSCXX} -
B${MNTROOTFS}lib/${LIB_PREFIX} -B${MNTROOTFS}usr/lib/${LIB_PREFIX}"
LDFLAGS="-Wl,-rpath-link=${MNTROOTFS}lib/${LIB_PREFIX} -Wl,-rpath-
link=${MNTROOTFS}usr/lib/${LIB_PREFIX}"

PKG_CONFIG_LIBDIR=${MNTROOTFS}usr/lib/${LIB_PREFIX}/pkgconfig:${MNTROOTFS}us
r/share/pkgconfig \
PKG_CONFIG_SYSROOT_DIR=${MNTROOTFS} \
LDFLAGS="${LDFLAGS}" \
PYTHON=${MNTROOTFS}/usr/bin/python3 \
./configure \
    --with-system-qemu=/usr/bin/qemu-system-i386 \
    --enable-systemd \
    --disable-xen \
    --enable-tools \
    --disable-docs \
    --disable-stubdom \
    --disable-golang \
    --prefix=/usr \
    --with-xenstored=xenstored \
    --build=x86_64-linux-gnu \
    --host=${CROSS_PREFIX} \
    CC="${CC}" \
    CXX="${CXX}"

PKG_CONFIG_LIBDIR=${MNTROOTFS}usr/lib/${LIB_PREFIX}/pkgconfig:${MNTROOTFS}us
r/share/pkgconfig \
PKG_CONFIG_SYSROOT_DIR=${MNTROOTFS} \
LDFLAGS="${LDFLAGS}" \
make dist-tools \
    CROSS_COMPILE=${CROSS_PREFIX}- XEN_TARGET_ARCH=${XEN_ARCH} \
```

```
    CC="${CC}" \
    CXX="${CXX}" \
    -j $(nproc)

sudo --preserve-env PATH=${PATH} \
PKG_CONFIG_LIBDIR=${MNTROOTFS}usr/lib/${LIB_PREFIX}/pkgconfig:${MNTROOTFS}us
r/share/pkgconfig \
PKG_CONFIG_SYSROOT_DIR=${MNTROOTFS} \
LDFLAGS="${LDFLAGS}" \
make install-tools \
    CROSS_COMPILE=${CROSS_PREFIX}- XEN_TARGET_ARCH=${XEN_ARCH} \
    CC="${CC}" \
    CXX="${CXX}" \
    DESTDIR=${MNTROOTFS}

sudo chroot ${MNTROOTFS} systemctl enable xen-qemu-dom0-disk-backend.service
sudo chroot ${MNTROOTFS} systemctl enable xen-init-dom0.service
sudo chroot ${MNTROOTFS} systemctl enable xenconsoled.service
sudo chroot ${MNTROOTFS} systemctl enable xendomains.service
sudo chroot ${MNTROOTFS} systemctl enable xen-watchdog.service

cd ${WRKDIR}

# It seems like the xen tools configure script selects a few too many of
these backend driver modules, so we override it with a simpler list.
# /usr/lib/modules-load.d/xen.conf
sudo bash -c "cat > ${MNTROOTFS}usr/lib/modules-load.d/xen.conf" <<EOF
xen-evtchn
xen-gntdev
xen-gntalloc
xen-blkback
xen-netback
EOF

# /etc/hostname
sudo bash -c "echo ${HOSTNAME} > ${MNTROOTFS}etc/hostname"

# /etc/hosts
sudo bash -c "cat > ${MNTROOTFS}etc/hosts" <<EOF
127.0.0.1   localhost
127.0.1.1   ${HOSTNAME}
# The following lines are desirable for IPv6 capable hosts
::1     ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
EOF

# /etc/fstab
```

```
sudo bash -c "cat > ${MNTROOTFS}etc/fstab" <<EOF
proc            /proc           proc    defaults          0       0
/dev/mmcblk0p1  /boot           vfat    defaults          0       2
/dev/mmcblk0p2  /               ext4    defaults,noatime  0       1
EOF

# /etc/network/interfaces.d/eth0
sudo bash -c "cat > ${MNTROOTFS}etc/network/interfaces.d/eth0" <<EOF
auto eth0
iface eth0 inet manual
EOF
sudo chmod 0644 ${MNTROOTFS}etc/network/interfaces.d/eth0

# /etc/network/interfaces.d/xenbr0
sudo bash -c "cat > ${MNTROOTFS}etc/network/interfaces.d/xenbr0" <<EOF
auto xenbr0
iface xenbr0 inet dhcp
    bridge_ports eth0
EOF
sudo chmod 0644 ${MNTROOTFS}etc/network/interfaces.d/xenbr0

# Don't wait forever and a day for the network to come online
if [ -s ${MNTROOTFS}lib/systemd/system/networking.service ]; then
    sudo sed -i -e "s/TimeoutStartSec=5min/TimeoutStartSec=15sec/"
${MNTROOTFS}lib/systemd/system/networking.service
fi
if [ -s ${MNTROOTFS}lib/systemd/system/ifup@.service ]; then
    sudo bash -c "echo \"TimeoutStopSec=15s\" >>
${MNTROOTFS}lib/systemd/system/ifup@.service"
fi

# User account setup
sudo chroot ${MNTROOTFS} useradd -s /bin/bash -G adm,sudo -l -m -p
${HASHED_PASSWORD} ${USERNAME}
# Password-less sudo
sudo chroot ${MNTROOTFS} /bin/bash -euxc "echo \"${USERNAME} ALL=(ALL)
NOPASSWD:ALL\" > /etc/sudoers.d/90-${USERNAME}-user"

df -h | grep -e "Filesystem" -e "/dev/mapper/loop"

cd ${WRKDIR}
sudo sync

sudo umount ${MNTROOTFS}proc || true
sudo umount ${MNTROOTFS}dev/pts || true
sudo umount ${MNTROOTFS}dev || true
sudo umount ${MNTROOTFS}sys || true
sudo umount ${MNTROOTFS}tmp || true
sudo umount ${MNTBOOT} || true
sudo umount ${MNTROOTFS} || true
sudo kpartx -dvs ${IMGFILE} || true
```

```
sudo rmdir ${MNTROOTFS} || true
if [ "$STATUS" == "0" ]; then
  mv ${IMGFILE} . || true
  echo "=== BUILD SUCCEEDED ==="
else
  rm ${IMGFILE} || true
  echo "=== BUILD ERROR on line ${BUILDLINE} ==="
fi
sudo umount ${MNTRAMDISK} || true
sudo rmdir ${MNTRAMDISK} || true
```

From:
http://www.ouarte.garden/ - **dwndoc**

Permanent link:
**http://www.ouarte.garden/doku.php?id=cookbooks:xen-rpi4-builder-ubuntu**

Last update: **2025/02/19 10:59**