Xen Hypervisor Setup on Odroid XU4

Ryan Cowart
with Armen Dzhagaryan and Aleksandar Milenkovic
The LaCASA Laboratory
http://lacasa.uah.edu

The following document explains the setup process for building and running the Xen hypervisor on the Odroid XU4 computing platform. It will detail how to configure a kernel image with Xen compatibility, configure a dom0 kernel, and configure and launch domU's. The host PC environment that the following procedure was executed on consisted of:

– Ubuntu 14.04 LTS installed on a Laptop
– SD card device name of /dev/mmcblk0

*NOTE: The majority of these steps were followed from the Odroid Wiki page located at http://odroid.com/dokuwiki/doku.php?id=en:xu4_xen

I. The first step is to download and install the ARM toolchains on the host PC.
To complete this step, execute the following commands.

```
$ mkdir toolchains
$ cd toolchains
$ wget
http://releases.linaro.org/14.09/components/toolchain/binaries/
gcc-linaro-arm-none-eabi-4.9-2014.09_linux.tar.xz/
$ wget http://dn.odroid.com/ODROID-XU/compiler/arm-eabi-
4.6.tar.gz
```

Untar the files in the toolchains directory

Then update your ~/.bashrc file with the following lines.

```
export ARCH=arm
export PATH=${PATH}:${PATH_TO_TOOLCHAINS_DIR}/
gcc-linaro-arm-none-eabi-4.9-2014.09_linux/bin
export PATH=${PATH}${PATH_TO_TOOLCHAINS_DIR}/arm-eabi-4.6/bin
```

Then source your bashrc with

```
$ source ~/.bashrc
```

II. The next step is to obtain the kernel image for the basic platform preparation.

```
$ wget http://odroid.in/ubuntu_14.04lts/
ubuntu-14.04lts-server-odroid-xu3-20150725.img.xz
```

OR

```
$ wget http://odroid.in/ubuntu_14.04lts/
ubuntu-14.04.1lts-lubuntu-odroid-xu3-20150212.img.xz
$ unxz IMAGE_FILE.xz
```

```
$ sudo dd if=IMAGE_FILE.img of=/dev/mmcblk0 bs=1M conv=fsync
$ sync
```

III. The next step is to download the Odroid bootloader that has the HYP mode enabled which is needed to support virtualization.

```
$ git clone https://github.com/hardkernel/u-boot.git -b
odroidxu3-v2012.07
```

Then enter the u-boot directory and compile the bootloader.

```
$ cd u-boot
$ export CROSS_COMPILE=arm-none-eabi-
$ make odroid_config
$ make ARCH=arm CROSS_COMPILE=arm-none-eabi- -j8
```

Then change directories into the sd_fuse/hardkernel directory and run the given script

```
$ cd sd_fuse/hardkernel
$ sudo ./sd_fusing.sh /dev/mmcblk0
$ sync
```

IV. The boot.ini file in the original kernel image needs to be replaced with one that supports virtualization.

```
$ wget http://git.io/vCu3c -O boot.ini
$ sudo cp -f boot.ini /media/USERNAME/boot/
$ sync
```

V. Next, download, build, and install the Xen source code:

```
$ git clone https://github.com/bkrepo/xen.git
$ cd xen
$ make dist-xen CROSS_COMPILE=arm-none-eabi-  \
XEN_TARGET_ARCH=arm32 debug=y  \
CONFIG_EARLY_PRINTK=exynos5250 -j8
$ sudo cp xen/xen /media/USERNAME/boot/
$ sync
```

VI. Build and install the Domain 0 Linux Kernel:

```
$ git clone https://github.com/bkrepo/linux-dom0.git
$ cd linux-dom0
$ make odroidxu3_xen_defconfig ARCH=arm
$ make ARCH=arm CROSS_COMPILE=arm-none-eabi- -j8
$ sudo make modules_install ARCH=arm  \
INSTALL_MOD_PATH=/media/USERNAME/rootfs/
$ sudo cp -f arch/arm/boot/zImage arch/arm/boot/dts/exynos5422-
odroidxu3.dtb  \
/media/USERNAME/boot/
$ sync
```

*NOTE: Boot the current SD card configuration on the Odroid (id: 'root/odroid', password: 'odroid') and complete steps VII through X on the Odroid platform.

VII. Get the Xen source code:

```
$ git clone  https://github.com/bkrepo/xen.git
$ cd xen
$ sudo su
```

VIII. Installation packages for xen tools build:

```
$ apt-get update
$ apt-get build-dep xen
$ apt-get install libpixman-1-dev
$ apt-get install xtightvncviewer
```

IX. Building Xen tools:

```
$ ./configure --disable-xen --disable-docs
$ make dist-tools -j4
$ make install-tools
$ update-rc.d xencommons defaults 19 18
$ update-rc.d xendomains defaults 21 20
$ update-rc.d xen-watchdog defaults 22 23
$ ldconfig
```

X. DHCP bridge network configuration:

```
$ apt-get install bridge-utils
$ vim /etc/network/interfaces
```

Insert the following lines into the /etc/network/interfaces file:

```
auto eth1
iface eth1 inet manual
up ip link set eth1 up

auto xenbr0
iface xenbr0 inet dhcp
bridge_ports eth1
```

*NOTE: Steps XI through XVI are executed on the host PC.

XI. Domain U kernel build preparation:

```
$ wget https://www.kernel.org/pub/linux/kernel/v4.x/linux-
4.2.tar.xz
$ tar Jxvf linux-4.2.tar.xz
$ cd linux
$ make exynos_defconfig ARCH=arm
$ make menuconfig ARCH=arm
```

1. Kernel Features → Xen guest support on ARM
2. Device Drivers → Block devices → Xen virtual block device support.
3. Device Drivers → Network device support → Xen network device frontend
4. Device Drivers → Xen driver support → Select all.
5. System Type → ARM system type → Allow multiple platforms to be selected.

6. System Type → Multiple platform selection → ARMv7 based platforms
7. System Type → Dummy Virtual Machine.
8. Device Drivers → Input Device support → Miscellaneous devices → Xen virtual keyboard and mouse support.

XII. Patching Domain U kernel source code:

```
$ wget http://git.io/vCV9E -O xen_blkif.patch
$ patch -p1 < xen_blkif.patch
```

XIII. Build & Installation:

```
$ export CROSS_COMPILE=arm-eabi-
$ make ARCH=arm CROSS_COMPILE=arm-eabi- zImage -j8
$ sudo cp arch/arm/boot/zImage /media/USERNAME/rootfs/root/
$ sync
```

XIV. Create the Domain U image:

```
$ sudo kpartx -v -a ubuntu-14.04lts-server-odroid-xu3-
20150725.img
```

OR

```
$ sudo kpartx -v -a ubuntu-14.04.1lts-lubuntu-odroid-xu3-
20150212.img
$ sudo dd if=/dev/mapper/loop0p2 of=domU.img
```

XV. Modifying Domain U root file system:

```
$ mkdir domU-root
$ sudo mount domU.img domU-root
$ cd domU-root
$ vim etc/fstab
```

Replace everything in the etc/fstab file with the following line:

```
/dev/xvda   /   ext4   errors=remount-ro   0   1
```

```
$ vim etc/init/console.conf
```

Replace everything in the etc/init/console.conf file with the following line:

```
start on stopped rc RUNLEVEL=[2345]
stop on runlevel [!2345]
respawn
exec /sbin/getty -8 38400 hvc0
```

```
$ cd ..
$ sudo umount domU-root
$ sudo cp domU.img /media/USERNAME/rootfs/root/
$ sync
```

XVI. Create the Xen configuration file:

```
$ vim domU.cfg
```

Insert the following line into the /etc/fstab file:

```
kernel = "/root/zImage"
memory = 256
```

```
maxmem = 256
name = "Domain U"
vcpus = 4
cpus = ['1', ‘2’,’3’,’4’]
serial="pty"
disk = [ 'phy:/root/domU.img,xvda,w' ]
vif = ['bridge=xenbr0']
extra = 'console=hvc0 xencons=tty root=/dev/xvda rw'
vfb = [‘type=vnc’]
```

```
$ sudo cp domU.cfg /media/USERNAME/rootfs/root/
$ sync
```

XVII. Boot the SD on the Odroid and run the virtual machine (On ODROID-XU3/4, id: 'root/odroid', password: 'odroid'):

```
$ sudo su
$ cd /root
$ xl create -c domU.cfg
```

*NOTE: If you are booting the workstation version of Ubuntu, follow the following steps to connect to the desktop environment of the domU.

XVIII. Login to the domU with same login as that for dom0.

```
$ sudo apt-get update
$ sudo apt-get install tightvncserver
$ sudo vncserver -geometry 800x600 :1
```

XIX. Connect to the vnc server on dom0 with the following command:

```
$ sudo xvncviewer DOMU_IP_ADDRESS:1
```