Virtualizing an OpenWrt CPE

by Vaibhav Bajpai `v.bajpai@jacobs-university.de`

OpenWrt buildroot environment can be used to cross-compile source code of a measurement test for a Customer Premises Equipment (CPE) such as a SamKnows box. The compiled test binary can later be remotely installed on the CPE. However the CPE, usually does contain the development tools required to introspect a binary. It's also not wise to try to build the development tools on the CPE hardware anyway, because it would just diverge off the production CPE build. As such, once a measurement test is pushed to the box there is little one can do but hope that the test works.

It would be nice to be able to virtualize a CPE on an x86 machine and install the development tools there. The OpenWrt distribution comes bundled with a lot of development tools. These development tools are available through the `opkg` package manager. The CPE hardware, however, usually run on a Microprocessor without Interlocked Pipeline Stages (MIPS) instruction set. Therefore, the challenge is to emulate a MIPS processor using an emulator and then virtualize the CPE on top of it. `qemu` appears to be a reasonable emulator for this task.

Introspect qemu MIPS emulation capability

Install `qemu`

```
$ sudo apt-get install qemu
```

Introspect the CPU on the SamKnows probe. This is only for informational purposes, since access to a SamKnows probe may not be available.

```
(samsbox) # cat /proc/cpuinfo
...
machine                 : TP-LINK TL-WR741ND v4
cpu model               : MIPS 24Kc V7.4
```

Introspect `qemu`.

```
$ qemu-system-mips -cpu '?'
MIPS '4Kc'
MIPS '4Km'
MIPS '4KEcR1'
MIPS '4KEmR1'
MIPS '4KEc'
MIPS '4KEm'
MIPS '24Kc'
MIPS '24Kf'
MIPS '34Kf'
```

It appears qemu supports mips 24Kc emulation.

```
$ qemu-system-mips -M '?'
Supported machines are:
mipssim    MIPS MIPSsim platform
malta      MIPS Malta Core LV (default)
magnum     MIPS Magnum
pica61     Acer Pica 61
mips       mips r4k platform
```

On further investigation it appears, `qemu` does not have firmwares for these architechtures (1). However, OpenWrt seems to provide a firmware for `mips malta` (2).

Virtualize OpenWrt on an emulated MIPS processor using default user-mode networking

The following instructions were run on Debian Squeeze x86-64. The `mips malta` image was generated using OpenWrt buildroot. However, the generation is only required to be done once, so the final kernel image has been uploaded to github for future use. The process of kernel image generation is in the appendix.

Get the `mips malta` (big endian) image for `qemu`

```
$ wget http://cnds.eecs.jacobs-university.de/users/vbajpai/aims-tutorial-2013/openwrt-malta-
be-vmlinux-adjustment.elf
```

Run the image under `qemu`

```
$ qemu-system-mips -kernel openwrt-malta-be-vmlinux-adjustment.elf \
                   -serial tcp::4000,server \
                   -m 64 -nographic
```

In a separate session, connect to the headless console over the network

```
$ nc 0.0.0.0 4000
...
[   18.060000] device eth0 entered promiscuous mode
[   18.110000] br-lan: port 1(eth0) entered forwarding state
[   18.120000] br-lan: port 1(eth0) entered forwarding state
[   20.120000] br-lan: port 1(eth0) entered forwarding state
```

Hit ENTER at this stage to fall into the prompt

```
BusyBox v1.19.4 (2013-01-07 16:44:17 CET) built-in shell (ash)
Enter 'help' for a list of built-in commands.

  _______                     ________        __
 |       |.-----.-----.-----.|  |  |  |.----.|  |_
 |   -   ||  _  |  -__|     ||  |  |  ||   _||   _|
 |_______||   __|_____|__|__||________||__|  |____|
          |__| W I R E L E S S   F R E E D O M
 -----------------------------------------------------
 BARRIER BREAKER (Bleeding Edge, r35037)
 -----------------------------------------------------
  * 1/2 oz Galliano         Pour all ingredients into
  * 4 oz cold Coffee        an irish coffee mug filled
  * 1 1/2 oz Dark Rum       with crushed ice. Stir.
  * 2 tsp. Creme de Cacao
 -----------------------------------------------------
```

Request an IP form the DHCP server

```
(openWrt) # udhcpc -i br-lan
```

Ping the default gateway

```
(openWrt) # ping -c 3 10.0.2.2
64 bytes from 10.0.2.2: seq=0 ttl=255 time=7.059 ms
```

However you cannot connect to the external network

```
(openWrt) # ping -c 3 8.8.8.8
```

We are going to use `vde2` with `tun/tap` to enable external network.

Enable external network using VDE2

Install `vde2`

```
$ sudo apt-get install vde2
$ sudo apt-get install uml-utilities
```

Initiate the `tun` kernel module

```
$ sudo modprobe tun
$ sudo echo "tun" >> /etc/modules
```

Create a `tun/tap` interface

```
$ sudo vim /etc/network/interfaces
auto tap0
iface tap0 inet static
    address 10.0.3.1
    netmask 255.255.255.0
    vde2-switch -t tap0
```

Setup `dnsmasq`

```
$ sudo apt-get install dnsmasq
$ sudo vim /etc/dnsmasq.conf
...
dhcp-range=tap0,10.0.3.10,10.0.3.20,4h
```

Restart the network

```
$ sudo service networking restart
$ sudo service dnsmasq restart
$ ifconfig tap0
tap0    ...
        inet addr:10.0.3.1
```

Add your user to the `vde2-net` group

```
$ newgrp - vde2-net
```

Check if you're added to the group

```
$ groups
... vde2-net
```

Add the group manually if it's still not listed:

```
$ sudo vim /etc/group
...
vde2-net:x:105:<username>
```

Logout and Log back in, check if you're added to the group

```
$ groups
... vde2-net
```

Enable IPv4 forwarding as a root user

```
# echo "1" > /proc/sys/net/ipv4/ip_forward
# iptables -t nat -A POSTROUTING -s 10.0.3.0/24 -o eth0 -j MASQUERADE
```

Run the image under `qemu`

```
$ qemu-system-mips -kernel openwrt-malta-be-vmlinux-adjustment.elf \
                   -serial tcp::4000,server \
                   -net nic -net vde,sock=/var/run/vde2/tap0.ctl \
                   -m 64 -nographic
```

In a separate sesssion, connect to the headless console over the network

```
$ nc 0.0.0.0 4000
...
[   18.060000] device eth0 entered promiscuous mode
[   18.110000] br-lan: port 1(eth0) entered forwarding state
[   18.120000] br-lan: port 1(eth0) entered forwarding state
[   20.120000] br-lan: port 1(eth0) entered forwarding state
```

Hit ENTER at this stage to fall into the prompt

```
BusyBox v1.19.4 (2013-01-07 16:44:17 CET) built-in shell (ash)
Enter 'help' for a list of built-in commands.

  _______                     ________        __
 |       |.-----.-----.-----.|  |  |  |.----.|  |_
 |   -   ||  _  |  -__|     ||  |  |  ||   _||   _|
 |_______||   __|_____|__|__||________||__|  |____|
          |__| W I R E L E S S   F R E E D O M
 -----------------------------------------------------
 BARRIER BREAKER (Bleeding Edge, r35037)
 -----------------------------------------------------
  * 1/2 oz Galliano         Pour all ingredients into
  * 4 oz cold Coffee        an irish coffee mug filled
  * 1 1/2 oz Dark Rum       with crushed ice. Stir.
  * 2 tsp. Creme de Cacao
 -----------------------------------------------------
```

Request an IP using DHCP, and set a password for `root`

```
(OpenWrt) udhcpc -i br-lan
...
Lease of 10.0.3.20 obtained, lease time 14400
(OpenWrt) passwd
(OpenWrt) CTRL-C
```

SSH into `qemu` VM from the host. Give it a minute

```
$ ssh root@10.0.3.20
(OpenWrt) # ping 8.8.8.8
64 bytes from 8.8.8.8: seq=0 ttl=61 time=311.755 ms
```

Add a DNS server

```
(OpenWrt) # echo "nameserver 8.8.8.8" > /tmp/resolv.conf.auto
(OpenWrt) # ping www.google.com
64 bytes from 173.194.69.104: seq=0 ttl=61 time=14.350 ms
```

It would be better to allow host `dnsmasq` to set the nameserver value for the guest though.

Enable Installation of Packages

Configure `opkg`

```
(OpenWrt) # cat /etc/opkg.conf
src/gz attitude_adjustment http://downloads.openwrt.org/attitude_adjustment/12.09-rc1/malta/
be/packages
...
```

This location does not exist, and gives a HTTP 404 response. However, `AR71xx` package location used by SamKnows probe can also be used instead. This change needs to be explicilty told to `opkg`. The architecture is followed by a priority value.

```
(OpenWrt) # vim /etc/opkg.conf
arch ar71xx 100
arch malta_mips 200
arch all 300
src/gz attitude_adjustment http://downloads.openwrt.org/attitude_adjustment/12.09-rc1/ar71xx
/generic/packages/
...
```

Update the packages

```
(OpenWrt) # opkg update
```

However we still cannot install any package, because we do not have a `/overlay` mounted to a disk. We tried preparing a `qcow2` image and supplying it as a `-hda` option argument when starting `qemu`, but the kernel does not appear to recognize it as a block device. As a workaround we can always mount `overlay_root` to `/tmp` and use it a RAM disk for installing the development tools.

```
(OpenWrt) # vim /etc/opkg.conf
- option overlay_root /overlay
+ option overlay_root /tmp
```

Install a package

```
(OpenWrt) opkg install file
(OpenWrt) file /bin/busybox
/bin/busybox: ELF 32-bit MSB executable, MIPS, MIPS32 rel2 version
1, dynamically linked (uses shared libs), with unknown capability
0x41000000 = 0xf676e75, with unknown capability 0x10000 = 0x70403, not
stripped
```

The ramdisk is only limited by the amount of memory supplied by qemu during runtime and can be changed using `-m` flag.

Secure Remote Access

Use Public-Key Authentication

```
$ scp ~/.ssh/id_dsa.pub root@10.0.3.20:/tmp
(OpenWrt) # cat /tmp/id_dsa.pub >> /etc/dropbear/authorized_keys
(OpenWrt) # chmod 600 /etc/dropbear/authorized_keys
```

Disable Password-based logins.

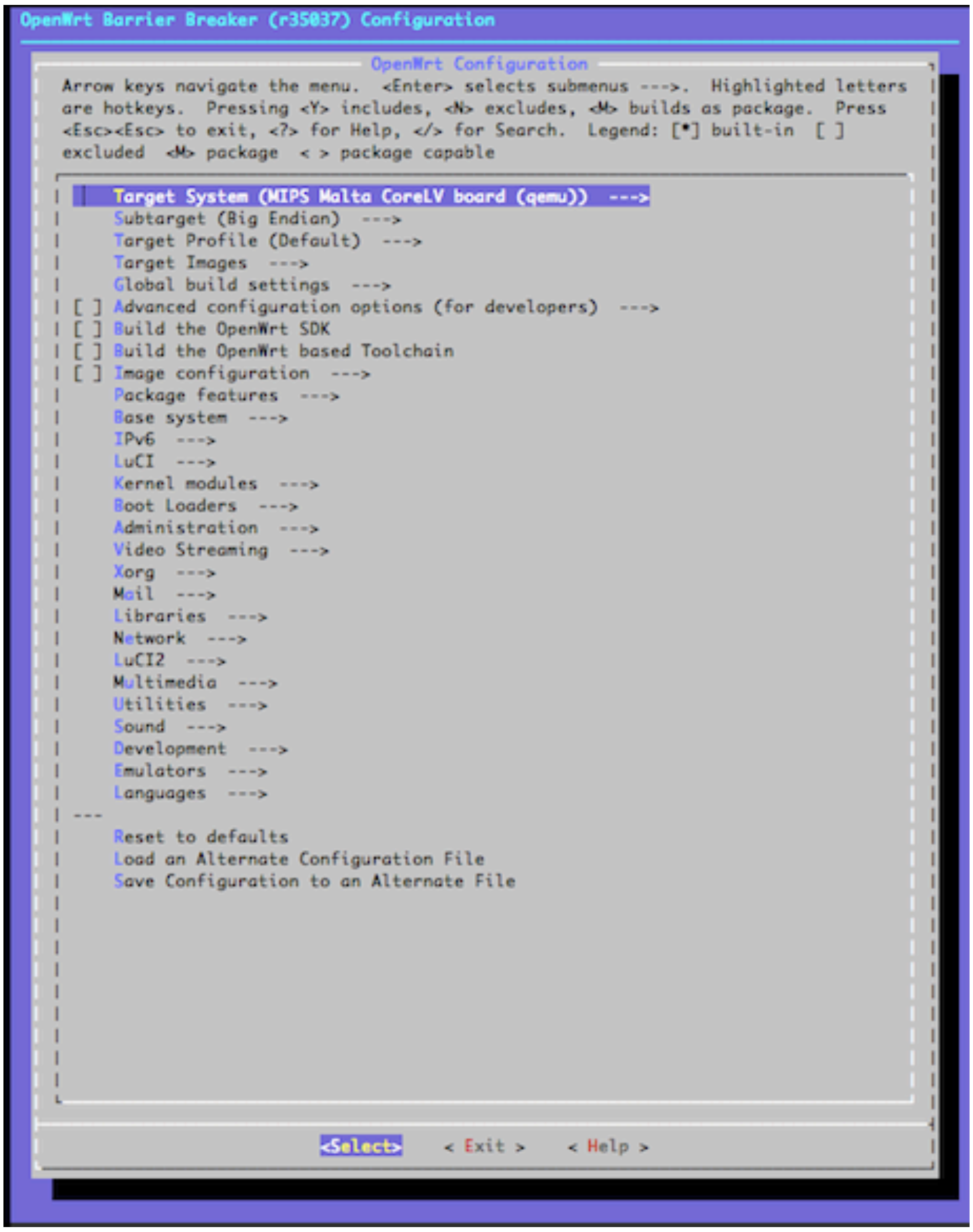
```
(OpenWrt) # uci set dropbear.@dropbear[0].PasswordAuth=off
(OpenWrt) # uci commit dropbear
```

It's good to be able to login using your signature

Appendix

The OpenWrt buildroot environment provides an option to prepare a `mips malta` kernel image that can be used by `qemu`.

```
$ git clone git://nbd.name/openwrt.git openWrt/
[OpenWrt] $ make package/symlinks
```



```
OpenWrt Barrier Breaker (r35037) Configuration

                 OpenWrt Configuration
 Arrow keys navigate the menu.  <Enter> selects submenus --->.  Highlighted letters
 are hotkeys.  Pressing <Y> includes, <N> excludes, <M> builds as package.  Press
 <Esc><Esc> to exit, <?> for Help, </> for Search.  Legend: [*] built-in  [ ]
 excluded  <M> package  < > package capable

        Target System (MIPS Malta CoreLV board (qemu))  --->
        Subtarget (Big Endian)  --->
        Target Profile (Default)  --->
        Target Images  --->
        Global build settings  --->
    [ ] Advanced configuration options (for developers)  --->
    [ ] Build the OpenWrt SDK
    [ ] Build the OpenWrt based Toolchain
    [ ] Image configuration  --->
        Package features  --->
        Base system  --->
        IPv6  --->
        LuCI  --->
        Kernel modules  --->
        Boot Loaders  --->
        Administration  --->
        Video Streaming  --->
        Xorg  --->
        Mail  --->
        Libraries  --->
        Network  --->
        LuCI2  --->
        Multimedia  --->
        Utilities  --->
        Sound  --->
        Development  --->
        Emulators  --->
        Languages  --->
    ---
        Reset to defaults
        Load an Alternate Configuration File
        Save Configuration to an Alternate File

              <Select>    < Exit >    < Help >
```

Imgur

```
[OpenWrt] $ make -j 3
```

The kernel image is generated at `openWRT/bin/malta/` as `openwrt-malta-be-vmlinux.elf`. This is the image that is uploaded to github.

References

(1) QEMU MIPS Emulation →
(2) OpenWRT MALTA under QEMU →