Getting QEMU to run ARM vm-image on Windows

For use with the OpenSecurityTraining.info/IntroARM.html class

By Jeff Tam

Environments

Running QEMU on Windows
Building ARM vm-image on Linux environment

Setting up QEMU (Windows)

Download latest QEMU binaries: http://wiki.qemu.org/Main_Page
Download GTK+: http://www.gtk.org/download/win32.php

Creating the ARM vm-image (Linux)

The following is a summary of & application of the instructions from: https://developer.mozilla.org/en-US/docs/Developer_Guide/Virtual_ARM_Linux_environment

Environment Setup

```
# Add Linaro's repository, containing their tools and more recent versions of qemu
sudo add-apt-repository ppa:linaro-maintainers/tools
sudo apt-get update
# Install linaro tools and qemu
sudo apt-get install linaro-image-tools qemu-user-static qemu-system

# If you want to be able to cross-compile on the host, install these as well
Sudo apt-get install gcc-arm-linux-gnueabi g++-arm-linux-gnueab

# Add Linaro's repository, containing their tools and more recent
# versions of qemu (you need at least qemu 0.15*).
sudo add-apt-repository ppa:linaro-maintainers/tools
# Install linaro tools and qemu
sudo apt-get install linaro-image-tools qemu-user-static qemu-system

# If you want to be able to cross-compile on the host, install these as well
sudo apt-get install gcc-arm-linux-gnueabi g++-arm-linux-gnueab
# or you can install CodeSourcery/ARM-Linux toolchain
```

Download a Linaro release & hardware pack

```
wget http://releases.linaro.org/platform/linaro-n/nano/alpha-3/linaro-natty-nano-tar-20110302-0.tar.gz
```

wget http://releases.linaro.org/platform/linaro-n/hwpacks/alpha-3/hwpack_linaro-vexpress_20110302-0_armel_supported.tar.gz

Create the image

linaro-media-create --image_file vexpress.img --dev vexpress --binary linaro-natty-nano-tar-20110302-0.tar.gz --hwpack hwpack_linaro-vexpress_20110302-0_armel_supported.tar.gz

Extracting the kernel and initrd

The created image contains the needed kernel and initrd of the ARM vm

The following mounts the image to "/mnt/tmp"

sudo mount -o loop,offset="$(file vexpress.img | awk 'BEGIN { RS=";"; } /partition 2/ { print $7*512; }')" -t auto vexpress.img /mnt/tmp

The wanted files are linked to by "/mnt/tmp/vmlinuz" (kernel) & "/mnt/tmp/initrd.img" (initrd)

Transfer vexpress.img, vmlinuz, & initrd to the Windows box/host

Renamed to vmlinuz & initrd.img

Starting QEMU (Windows)

qemu-system-arm -M vexpress-a9 -cpu cortex-a9 -kernel ./vmlinuz -initrd ./initrd.img –redir tcp:2200::22 -m 512 -append "root=/dev/mmcblk0p2 vga=normal mem=512M devtmpfs.mount=0 rw" -drive file=vexpress.img,if=sd,cache=writeback

Notes:

- The "-redir tcp:2200::22" redirects TCP traffic on the host port 2200 to the guest machine (QEMU) port 22. This will allow us to SSH into the machine later by connecting to localhost on 2200.
- The "-m 512" specifies that we want 512 MB of RAM. You can adjust this, but make sure you also change it in the "-append" string.
- The "-drive file=vexpress.img,if=sd,cache=writeback" attaches our images as an SD card. (Supposedly provides faster I/O)

Setting up APT-repositories

If you are using an older version of Ubuntu as the vm (out of support, etc)

edit "sources.list" to use "old-releases.ubuntu.com/ubuntu"

vi /etc/apt/sources.list

Setting up SSH on the vm (VM)

Enable a network connection

ifconfig eth0 up

dhclient eth0

Install SSH
apt-get install openssh-server

Persist network changes
auto eth0
iface eth0 inet dhcp

Notes:

- Set a password for root (“passwd” cmd)
- SSH via “ssh –p2200 root@localhost” on the host (Windows) machine

Installing needed packages

apt-get install libpthread-stubs0
apt-get install gcc
apt-get install g++
apt-get install make

References

QEMU: http://wiki.qemu.org/Main_Page
GTK+: http://www.gtk.org/download/win32.php
ARM vm-image & QEMU startup: https://developer.mozilla.org/en-US/docs/Developer_Guide/Virtual_ARM_Linux_environment