Introduction to qemu-debootstrap

Sat 21 January 2017
Tags qemu linux aarch64 arm64
Posted by Logan

User-mode QEMU translates the instructions and the system calls. To run a **static executable**, you may wrap the command line with `qemu-${arch}`. For example, you may wrap an ARM64 static executable with `qemu-aarch64`:

```
$ qemu-aarch64 /path/to/static-executable
```

However, running a **dynamically linked executable** requires more efforts. To run a dynamically linked executable, a complete user space environment (including dynamic linker, shared libraries, etc) must be set up.

Fortunately, `qemu-debootstrap` can help us set up an Ubuntu (or Debian) user space environment. In this post, the instructions to set up an ARM64 Ubuntu environment will be presented.

Prerequisites

Three packages are required for this post:

- **debootstrap** is a Debian (or Ubuntu) bootstrap tool. It will download several deb packages and set up a minimal Debian (or Ubuntu) user space.
- **qemu-user-static** is the deb package for user-mode QEMU static executables. Don't confuse this package with *qemu-user*. The static version must be installed because these QEMU executables will be copied into chroot environments. This package also includes `qemu-debootstrap`, which is a wrapper for cross-architecture debootstrap.
- **schroot** is a chroot wrapper. Besides changing the root directory, schroot will run several hooks before entering and after leaving the chroot so that many functionalities, including network, can work without problems.

Run the command below to install these prerequisites:

```
$ sudo apt-get install debootstrap qemu-user-static schroot
```

Create an Environment

To create an Ubuntu 16.04 (Xenial Xerus) ARM64 user space in the `arm64-ubuntu` directory, run the following command [1]:

```
$ sudo qemu-debootstrap --arch=arm64 xenial arm64-ubuntu
```

And then, create a schroot configuration at `/etc/schroot/chroot.d/arm64-ubuntu`:

```
$ echo "[arm64-ubuntu]
description=Ubuntu 16.04 Xenial (arm64)
directory=$(pwd)/arm64-ubuntu
root-users=$(whoami)
users=$(whoami)
type=directory" | sudo tee /etc/schroot/chroot.d/arm64-ubuntu
```

Finally, enter the schroot with:

```
$ schroot -c arm64-ubuntu
```

Now, we are in an environment that can run ARM 64-bit binaries. For example, we can check the architecture with:

```
> uname -m
aarch64
```

BTW, to be a root user in the chroot, you may pass `-u root`:

```
$ schroot -c arm64-ubuntu -u root
```

And then, you may install packages with `apt-get`. For example:

```
> apt-get install vim
```

Epilogue

In this post, I have covered `qemu-user-static`, `qemu-debootstrap` and `schroot`. These tools significantly boosted my productivity on cross-architecture development. I hope this article is helpful. Thanks for reading.

[1] By default, `qemu-debootstrap` will download deb packages from http://ports.ubuntu.com/ubuntu-ports. If you would like to change the apt repository, specify the URL after the directory name.