Linux Kernel Programming

Exploring & debugging the kernel using Qemu

Pierre Olivier

Systems Software Research Group @ Virginia Tech

February 18, 2017

Outline

Outline

Qemu quick presentation

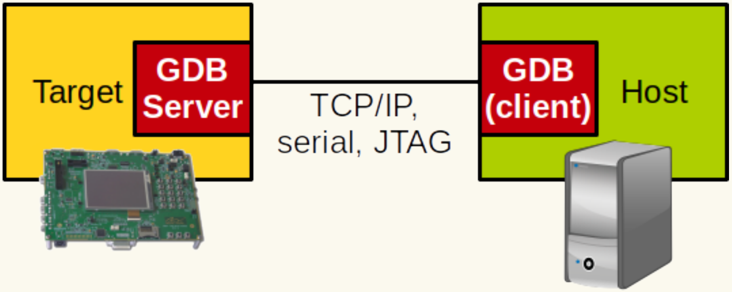
- **Full system emulator:** emulates an entire virtual machine
  - Using a software model for the CPU, memory, devices
  - Emulation is slow
- Can also be used in conjunction with hardware virtualization extensions to provide high performance virtualization
  - **KVM**
    - In-kernel support for virtualization + extensions to Qemu

Outline

Qemu and kernel development

GDB server

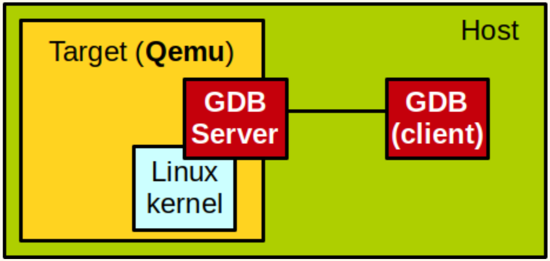
- **GDB server**
  - Originally used to debug a program executing on a remote machine
  - For example when GDB is not available on that remote machine
    - Ex: low performance embedded systems



Qemu and kernel development

Qemu & GDB sever

- Qemu is capable of running a kernel in an emulated machine with an associated root file system ...
- ... and **act as a GDB server for the kernel itself**



Qemu and kernel development

Qemu & GDB server: benefits

- Benefits:
  1. **Debugging**
  2. **Runtime code exploration**

Outline

Using Qemu and GDB server
Requirements

- **Requirements**:
  - Linux should be compiled with debug symbols:
    - `make menuconfig` > Kernel hacking > Compile the kernel with debug info (old kernels)
    - `make menuconfig` > Kernel hacking > Compile-time checks and compiler options > Compile the kernel with debug info
  - Qemu options:
    - **`-kernel path/to/bzImage`**: path to the `bzImage` of the kernel we want to execute and debug
    - **`-s`**: enable the GDB server
    - **`-S`**: (optional): pause on the first kernel instruction waiting for a GDB client connection order to continue
- **Usage (client side)**:

```
gdb path/to/vmlinux
(gdb) target remote:1234
```

Using Qemu and GDB server
Requirements (2)

- GDB usage:
  - `http://www.dirac.org/linux/gdb/`

Demo.

Outline

Miscellaneous information

Remote GDB bug on 64 bits

- Error when connecting to the remote target: `Remote 'g' packet reply is too long`
  - You need to patch GDB (client)
  - Patch for old versions of GDB sources: `http://www.cygwin.com/ml/gdb-patches/2012-03/msg00116.html`
  - Last version (7.11+): `https://github.com/olivierpierre/gdb-remote-patch`
- Compiling GDB:
  1. Grab the sources: `https://www.sourceware.org/gdb/download/`
  2. Patch it using `patch -p1 < patch-name.patch`
  3. Then:

```
./configure # Might notify for missing dependencies
make
sudo mv /usr/bin/gdb /usr/bin/gdb.old # Backup the old version
make install
```

Miscellaneous information

Optimized values

```
(gdb) p some_variable
$1 = <value optimized out>
```

- **It is not possible to disable optimization for the entire kernel**
- Needs to be done on a per-file basis
  1. Identify the file containing the variable declaration
  2. Update the corresponding makefile (example with fs/ext4/Makefile):

```
obj-$(CONFIG_EXT4_FS) += ext4.o

CFLAGS_bitmap.o = -O0

ext4-y  := balloc.o bitmap.o dir.o file.o fsync.o ialloc.o inode.o page-io.o \
ioctl.o namei.o super.o symlink.o hash.o resize.o extents.o \
ext4_jbd2.o migrate.o mballoc.o block_validity.o move_extent.o \
mmp.o indirect.o extents_status.o xattr.o xattr_user.o \
xattr_trusted.o inline.o readpage.o sysfs.o
# ...
```

Miscellaneous information
Mounting a virtual disk

- With Qemu, the root filesystem is generally present on a virtual disk (disk image)
- What if there is a crash at boot time that prevent the emulated machine from booting?
- You can mount the virtual disk on the host to try to fix the problem from there
  - **Mounting depends on the image format**
- Check the format using `file`:

```
file <disk image file path>

# QCOW2 format:
debian7.qcow2: QEMU QCOW Image (v3), 21474836480 bytes

# RAW format:
hd.img: x86 boot sector; partition 1: ID=0x83, active, starthead 32, startsector 2048,
      40134656 sectors; partition 2: ID=0x5, starthead 254, startsector 40138750,
      1802242 sectors, code offset 0x63
```

Miscellaneous information

Mounting a virtual disk (2)

- **Qcow2 format:**

```
sudo modprobe nbd max_part=63
sudo qemu-nbd -c /dev/nbd0 image.qcow2
sudo mount /dev/nbd0p1 /mnt/image

# work on the mounted filesystem ...

sudo umount /mnt/image
sudo qemu-nbd -d /dev/nbd0
sudo rmmod nbd
```

- **Raw format:**

```
file hd.img
hd.img: x86 boot sector; partition
      1: ID=0x83, active, starthead
      32, startsector 2048, 40134656
       sectors; partition 2: ID=0x5,
       starthead 254, startsector
      40138750, 1802242 sectors,
      code offset 0x63

# 1048576 == 2048 * 512
sudo mount -o loop,offset=1048576 hd
      .img /mnt/image

# work on the mounted filesystem ...

sudo umount /mnt/image
```

- **Do not launch the VM while the root filesystem is mounted on the host**

Miscellaneous information

Additional info

- Cursor disappears in qemu window?
  - Ctrl + Alt (right)
- **Do not close Qemu without issuing the `halt` command to the (Qemu) VM**
  - Risks leading to inconsistent filesystem state (data loss, VM unable to boot ...)
  - This is true for all VMs
- Qemu is too slow
  - Update Qemu version
  - Try KVM but you need a native installation