Qemu 简述

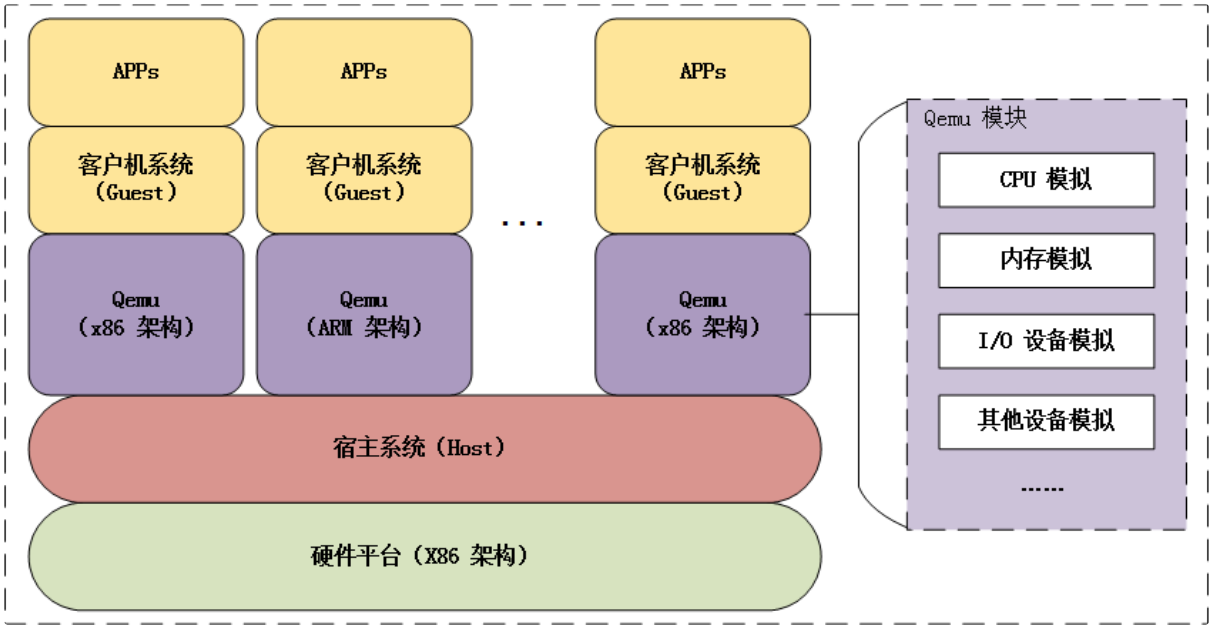
> 本文首发于我的公众号 **Linux云计算网络（id: cloud_dev）** ，专注于干货分享，号内有 **10T** 书籍和视频资源，后台回复
> 取，欢迎大家关注，二维码文末可以扫。

CONTENTS

Qemu 架构

Qemu 是纯软件实现的虚拟化模拟器，几乎可以模拟任何硬件设备，我们最熟悉的就是能够模拟一台能够独立运行操作系统
认为自己和硬件打交道，但其实是和 Qemu 模拟出来的硬件打交道，Qemu 将这些指令转译给真正的硬件。

正因为 Qemu 是纯软件实现的，所有的指令都要经 Qemu 过一手，性能非常低，所以，在生产环境中，大多数的做法都是
虚拟化工作，因为 KVM 是硬件辅助的虚拟化技术，主要负责 比较繁琐的 CPU 和内存虚拟化，而 Qemu 则负责 I/O 虚拟化，两者合作各自发挥自身的优势，相得益彰。



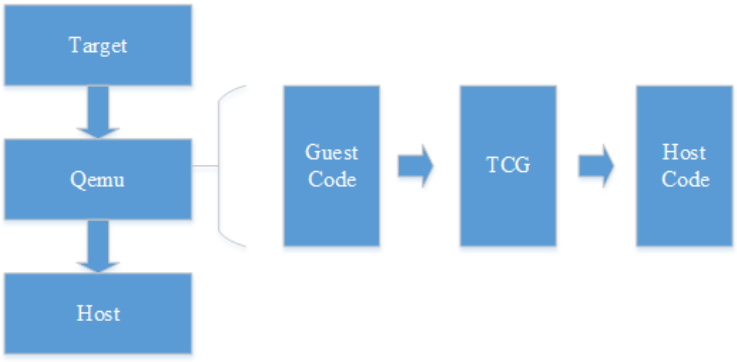
从本质上看，虚拟出的每个虚拟机对应 host 上的一个 Qemu 进程，而虚拟机的执行线程（如 CPU 线程、I/O 线程等）对应 Qemu 进程的一个线程。下面通过一个虚拟机启动过程看看 Qemu 是如何与 KVM 交互的。

Copy

```
// 第一步，获取到 KVM 句柄
kvmfd = open("/dev/kvm", O_RDWR);
// 第二步，创建虚拟机，获取到虚拟机句柄。
vmfd = ioctl(kvmfd, KVM_CREATE_VM, 0);
// 第三步，为虚拟机映射内存，还有其他的 PCI，信号处理的初始化。
ioctl(kvmfd, KVM_SET_USER_MEMORY_REGION, &mem);
// 第四步，将虚拟机镜像映射到内存，相当于物理机的 boot 过程，把镜像映射到内存。
// 第五步，创建 vCPU，并为 vCPU 分配内存空间。
ioctl(kvmfd, KVM_CREATE_VCPU, vcpuid);
vcpu->kvm_run_mmap_size = ioctl(kvm->dev_fd, KVM_GET_VCPU_MMAP_SIZE, 0);
// 第五步，创建 vCPU 个数的线程并运行虚拟机。
ioctl(kvm->vcpus->vcpu_fd, KVM_RUN, 0);
// 第六步，线程进入循环，并捕获虚拟机退出原因，做相应的处理。
for (;;) {
 ioctl(KVM_RUN)
 switch (exit_reason) {
    case KVM_EXIT_IO:  /* ... */
    case KVM_EXIT_HLT: /* ... */
 }
}
// 这里的退出并不一定是虚拟机关机，
// 虚拟机如果遇到 I/O 操作，访问硬件设备，缺页中断等都会退出执行，
// 退出执行可以理解为将 CPU 执行上下文返回到 Qemu。
```

Qemu 源码结构

Qemu 软件虚拟化实现的思路是采用二进制指令翻译技术，主要是提取 guest 代码，然后将其翻译成 TCG 中间代码，最后再将中间代码翻译成 host 指定架构的代码，如 x86 体系就翻译成其支持的代码形式，ARM 架构同理。



CONTENTS

所以，从宏观上看，源码结构主要包含以下几个部分：

- /vl.c：最主要的模拟循环，虚拟机环境初始化，和 CPU 的执行。
- /target-arch/translate.c：将 guest 代码翻译成不同架构的 TCG 操作码。
- /tcg/tcg.c：主要的 TCG 代码。
- /tcg/arch/tcg-target.c：将 TCG 代码转化生成主机代码。
- /cpu-exec.c：主要寻找下一个二进制翻译代码块，如果没有找到就请求得到下一个代码块，并且操作生成的代码块。

其中，涉及的主要几个函数如下：

函数	路径	注释
main_loop	{/vl.c}	很多条件的判断，如电源是否断等
qemu_main_loop_start	{/cpus.c}	分时运行 CPU 核
struct CPUState	{/target-xyz/cpu.h}	CPU 状态结构体
cpu_exec	{/cpu-exec.c}	主要的执行循环
struct TranslationBlock	{/exec-all.h}	TB（二进制翻译代码块） 结构体
cpu_gen_code	{translate-all.c}	初始化真正代码生成
tcg_gen_code	{/tcg/tcg.c}	tcg 代码翻译成 host 代码

知道了这个总体的代码结构，再去具体了解每一个模块可能会相对容易一点。

Qemu 的使用

1. 源码下载

Copy

```
centos：
sudo apt-get install qemu
ubuntu：
sudo yum install qemu -y
安装包：
$wget http://wiki.qemu-project.org/download/qemu-2.0.0.tar.bz2
$tar xjvf qemu-2.0.0.tar.bz2
Git：
$git clone git://git.qemu-project.org/qemu.git
```

2. 编译及安装

```
$cd qemu-2.0.0 //如果使用的是git下载的源码，执行cd qemu
$./configure --enable-kvm --enable-debug --enable-vnc --enable-werror  --target-list="x86_64-softmmu"
$make -j8
```

sudo make install

configure 脚本用于生成 Makefile，其选项可以用 ./configure --help 查看。

这里使用到的选项含义如下：

- --enable-kvm：编译 KVM 模块，使 Qemu 可以利用 KVM 来访问硬件提供的虚拟化服务。
- --enable-vnc：启用 VNC。
- --enalbe-werror：编译时，将所有的警告当作错误处理。
- --target-list：选择目标机器的架构。默认是将所有的架构都编译，但为了更快的完成编译，指定需要的架构即可。

安装好之后，会生成如下应用程序：

- ivshmem-client/server：这是一个 guest 和 host 共享内存的应用程序，遵循 C/S 的架构。
- qemu-ga：这是一个不利用网络实现 guest 和 host 之间交互的应用程序（使用 virtio-serial），运行在 guest 中。
- qemu-io：这是一个执行 Qemu I/O 操作的命令行工具。
- qemu-system-x86_64：Qemu 的核心应用程序，虚拟机就由它创建的。
- qemu-img：创建虚拟机镜像文件的工具，下面有例子说明。
- qemu-nbd：磁盘挂载工具。

下面通过创建虚拟机操作来对这些工具有个初步的认识。

3. 创建虚拟机

- 使用qemu-img创建虚拟机镜像

虚拟机镜像用来模拟虚拟机的硬盘，在启动虚拟机之前需要创建镜像文件。

```
qemu-img create -f qcow2 test-vm-1.qcow2 10G
```

Copy

-f 选项用于指定镜像的格式，qcow2 格式是 Qemu 最常用的镜像格式，采用来写时复制技术来优化性能。test-vm-1.qcow2 是镜像文件的名字，10G是镜像文件大小。镜像文件创建完成后，可使用 qemu-system-x86 来启动x86 架构的虚拟机.

- 使用 qemu-system-x86 来启动 x86 架构的虚拟机

```
qemu-system-x86_64 test-vm-1.qcow2
```

Copy

因为 test-vm-1.qcow2 中并未给虚拟机安装操作系统，所以会提示 “No bootable device”，无可启动设备。

- 启动 VM 安装操作系统镜像

```
qemu-system-x86_64 -m 2048 -enable-kvm test-vm-1.qcow2 -cdrom ./Centos-Desktop-x86_64-20-1.iso
```

Copy

-m 指定虚拟机内存大小，默认单位是 MB， -enable-kvm 使用 KVM 进行加速，-cdrom 添加 fedora 的安装镜像。可在弹出的窗口中操作虚拟机，安装操作系统，安装完成后重起虚拟机便会从硬盘 (test-vm-1.qcow2) 启动。之后再启动虚拟机只需要执行：

```
qemu-system-x86_64 -m 2048 -enable-kvm test-vm-1.qcow2
```

Copy

qemu-img 支持非常多种的文件格式，可以通过 qemu-img -h 查看.
其中 raw 和 qcow2 是比较常用的两种，raw 是 qemu-img 命令默认的，qcow2 是 qemu 目前推荐的镜像格式，是功能最多的格式。这些知识后面会有文章来专门讲述。

CONTENTS

CONTENTS

stay hungry stay foolish ----jobs 希望多多烧香!

分类: 云计算, 虚拟化
标签: 虚拟化, 云计算

推荐 2 赞赏 收藏 反对 0

« 上一篇: 虚拟化技术总览
» 下一篇: CPU 虚拟化

posted @ 2017-11-19 12:22 CloudDeveloper 阅读(32767) 评论(0) 编辑 收藏