中国科学技术大学软件学院
SCHOOL OF SOFTWARE ENGINEERING OF USTC

Build & Run Linux System

Based on VMware&Ubuntu10.04
linux-3.2.1

中国科学技术大学软件学院
孟宁
mengning@ustc.edu.cn

Agenda

- 编译Linux在QEMU模拟器上运行
  - 1. 准备模拟器
  - 2. 编译Linux内核
  - 3. 准备根文件系统
  - 4. 运行
- 制作完整的PC-Linux系统
- 构建Linux内核的gdb调试环境

准备模拟器qemu

- 1. sudo apt-get install qemu
  - 这个源http://mirror-fpt-telecom.fpt.net/ubuntu/有qemu
- 2. 有的源中不带qemu，则需要自己编译（未实际验证）
  - 获得qemu源代码http://wiki.qemu.org/download/qemu-1.0.tar.gz，并解压缩配置、编译，并安装到指定的目录下
  - ./configure --prefix=/usr/bin --target-list=i386-softmmu
  - make
  - su -c “make install”
- 编译安装完成之后，如何使用qemu?
  - 可以通过指定路径的方式使用qemu，此时qemu在安装目录下的bin目录中
  - 可以将安装目录/bin加入到PATH环境变量中，此时可以在任何目录下直接使用qemu

编译Linux内核

- 获得http://www.kernel.org/pub/linux/kernel/vx.y/linux-x.y.z.tar.gz，解压缩得到目录linux-x.y.z，不妨称之为Linux源代码根目录（以下简称源码根目录）
  - tar -zvxf linux-3.2.1.tar.gz （以3.2.1版本为例）
- 进入源代码根目录
- 可以使用make help得到一些编译内核的帮助信息
- 我们采用i386的缺省编译
  - make i386_defconfig
  - make
- 可以观察一下编译过程中的输出信息，特别是编译最后阶段的输出信息。

准备根文件系统（方法一）

- 1）建立目标根目录映像
  - dd if=/dev/zero of=myinitrd4M.img bs=4096 count=1024
  - mkfs.ext3 myinitrd4M.img
  - mkdir rootfs
  - sudo mount -o loop myinitrd4M.img rootfs

准备根文件系统（方法一）

- 2）准备一个helloworld应用程序，使用静态链接的方法编译成可执行文件，命名为init，并将init拷贝到目标根目录下
  - （linux启动后期会在根目录中寻找一个应用程序来运行，在根目录下提供init是一种可选方案）
  - gcc -static -o init helloworld.c
  - cp init rootfs/

```
#include<stdio.h>

int main()
{
        printf("hellworld!\n");
}
```

准备根文件系统（方法一）

- 3）准备dev目录
  - sudo mkdir rootfs/dev
  - linux启动过程中会启用console设备
    - sudo mknod rootfs/dev/console c 5 1
  - 另外需要提供一个linux根设备，我们使用ram
    - sudo mknod rootfs/dev/ram b 1 0
- 4）sudo umount rootfs
- 至此，一个包含简单应用程序的根目录映像myinitrd4M.img就准备好了

运行

- 使用方法一手工创建的根文件系统
- qemu -kernel linux-3.2.1/arch/x86/boot/bzImage -initrd myinitrd4M.img -append "root=/dev/ram init=/init"

准备根文件系统（方法二）

- 1）编译BusyBox
  - 下载busybox的源代码，解压缩
    - http://busybox.net/downloads/busybox-1.19.3.tar.bz2
  - make help可以得到一些编译busybox的帮助信息
  - make defconfig
  - make menuconfig修改如下配置：
    - enable：busybox settings –> build options –> build busybox as a static binary（no share libs）
    - sudo apt-get install libncurses5-dev libncurses5-dbg libncurses5 （执行make menuconfig需要的库）
  - make

准备根文件系统（方法二）

- 2）准备根目录映像，并安装busybox到根目录映像中
  - dd if=/dev/zero of=busyboxinitrd4M.img bs=4096 count=1024
  - mkfs.ext3 busyboxinitrd4M.img
  - mkdir rootfs
  - sudo mount -o loop busyboxinitrd4M.img rootfs
  - 在busybox目录下
    - sudo make CONFIG_PREFIX=../rootfs/ install
    - sudo umount rootfs

qemu -kernel linux-3.2.1/arch/x86/boot/bzImage -initrd myinitrd4M.img -append "root=/dev/ram init=/init noapic"

运行

- 使用方法二BusyBox创建的根文件系统
- qemu -kernel linux-3.2.1/arch/x86/boot/bzImage -initrd busyboxinitrd4M.img -append "root=/dev/ram init=/bin/ash"
- 此时可以进入busybox提供的shell环境

```
[    2.808733] EXT3-fs (ram0): mounted filesystem with writeback
[    2.811979] VFS: Mounted root (ext3 filesystem) readonly on de
[    2.813057] Freeing unused kernel memory: 452k freed
[    2.839521] Write protecting the kernel text: 6196k
[    2.840029] Write protecting the kernel read-only data: 2084k
/bin/ash: can't access tty; job control turned off
/ # ls
bin      linuxrc  sbin     usr
[    6.389190] ls used greatest stack depth: 6612 bytes left
/ #
```

制作完整的PC-Linux系统

- 制作带grub启动的磁盘映像
  - 1. 获得grub并制作grub启动软盘
  - 2. 准备磁盘映像
  - 3. 将磁盘映像升级为带grub启动的
  - 4. 运行

获得grub并制作grub启动软盘

- 下载grub，解压缩，查看解压缩得到的目录
  - ftp://alpha.gnu.org/gnu/grub/grub-0.97-i386-pc.tar.gz
- 建立启动软盘映像
  - dd if=/dev/zero of=a.img bs=512 count=2880
- 添加grub启动功能
  - sudo losetup /dev/loop3 a.img
  - sudo dd if=./grub-0.97-i386-pc/boot/grub/stage1 of=/dev/loop3 bs=512 count=1
  - sudo dd if=./grub-0.97-i386-pc/boot/grub/stage2 of=/dev/loop3 bs=512 seek=1
  - sudo losetup -d /dev/loop3
- 测试是否能进入grub界面
  - qemu -fda a.img

准备磁盘映像

- dd if=/dev/zero of=32M.img bs=4096 count=8192
- sudo losetup /dev/loop3 32M.img
- 在磁盘映像上建立一个活动分区
  - sudo fdisk /dev/loop3 （根据帮助新建一个主分区）
- sudo losetup -d /dev/loop3
- 将活动分区格式化成ext3fs，并mount到rootfs目录上
  - sudo losetup -o 32256 /dev/loop3 32M.img
  - 其中，32256是分区的起始位置，为63 512
  - 其中，63是通过file 32M.img得到的startsector信息
  - sudo mkfs.ext3 /dev/loop3
- 将前面制作的bzImage和busyboxinitrd4M.img拷贝到rootfs中

将磁盘映像升级为带grub启动的

- 准备相关目录，并拷贝一些必要的文件
  - sudo mkdir rootfs/boot
  - sudo mkdir rootfs/boot/grub
  - sudo cp ./grub-0.97-i386-pc/boot/grub/* rootfs/boot/grub
- 在rootfs/boot/grub中编写menu.lst，具有如下内容
  - default 0
  - timeout 30
  - title linux on 32M.img
  - root (hd0,0)
  - kernel (hd0,0)/bzImage root=/dev/ram init=/bin/ash
  - initrd (hd0,0)/myinitrd4M.img

运行

- 注意别忘了做一些清理工作
  - sudo umount rootfs
  - sudo losetup -d /dev/loop3
- 利用grub启动软盘，在硬盘映像上添加grub功能
  - qemu -boot a -fda a.img -hda 32M.img
  - 进入grub界面后
  - root (hd0,0)
  - setup (hd0)
- 测试从磁盘启动进入grub界面
  - qemu -hda 32M.img
- 如果QEMU上显示像正常机器一样启动并最后停在/bin/ash的命令界面下，说明您成功模拟了PC-Linux系统，请您分析对比一下这个Linux系统与您正使用linux发行版还有哪些差异和不足？

构建Linux内核的gdb调试环境

1. 在qemu中启动gdb server
2. 建立gdb与gdbserver之间的连接
3. 加载vmlinux中的符号表，设置断点
4. 重新配置编译Linux使之携带调试信息

在qemu中启动gdb server

- qemu -kernel linux-3.2.1/arch/x86/boot/bzImage -initrd busyboxinitrd4M.img -append "root=/dev/ram init=/bin/ash" -s -S
- 可以看到在新打开的qemu虚拟机上，整个是一个黑屏，此时qemu在等待gdb的连接
- 关于-s和-S选项的说明
  - -S freeze CPU at startup (use ’c’ to start execution)
  - -s shorthand for -gdb tcp::1234 若不想使用1234端口，则可以使用-gdb tcp:xxxx来取代-s选项

建立gdb与gdbserver之间的连接

- 在另外一个终端运行gdb，然后在gdb界面中运行如下命令
  - target remote：1234 则可以建立gdb和gdbserver之间的连接
  - 按c 让qemu上的Linux继续运行
- 假如在前面使用-gdb tcp::xxxx，则这里的1234也要修改为对应的xxxx
- 问题：此时没有加载符号表，无法根据符号设置断点

加载vmlinux中的符号表，设置断点

- 在gdb界面中targe remote之前加载符号表
  - file linux-3.2.1/vmlinux
- 在gdb界面中设置断点
  - break start_kernel 断点的设置可以在target remote之前，也可以在之后
- 在设置好start kernel处断点并且target remote之后可以继续运行，则在运行到start kernel的时候会停下来，等待gdb调试命令的输入
- 此后可以继续设置新的断点，...
- 问题：此时尽管有符号表，但是无法显示源代码

重新配置编译Linux使之携带调试信息

- 在原来配置的基础上，重新配置Linux，使之携带调试信息
  - kernel hacking—>
  - [*] compile the kernel with debug info
- make重新编译（时间较长）
- 此时，若按照前面相同的方法来运行，则在start kernel停下来后，可以使用list来显示断点处相关的源代码

中国科学技术大学软件学院
SCHOOL OF SOFTWARE ENGINEERING OF USTC

谢谢大家！

References

《深入理解Linux内核》第三版
独辟蹊径品内核：Linux内核源代码导读
http://219.219.220.231/wiki/LinuxStart
http://staff.ustc.edu.cn/~xlanchen/2011FallULK/ULK2011Fall.htm
http://www.kernel.org/
http://wiki.qemu.org
http://busybox.net
http://www.gnu.org/software/grub/