内核模块

内核模块可以在系统不重启的情况下动态加载或卸载。模块可以是内置或者动态加载，内置的模块在编译内核时需要特别设置，从而编译成内核功能。需要注意的是不能跨内核版本使用内核模块。

1.模块保存目录

```
/lib/modules/kernel_release/
```

2.查看当前已经加载的所有模块

```
lsmod
```

3.查看某一模块详细信息

```
modinfo modules_name
```

4.显示模块的依赖信息

```
modprobe --show-depends module_name

#保存模块间的依赖关系的文件
/lib/kernel_release/modules.dep
#手动生成模块依赖关系，执行一下命令
depmod
```

5.装载/卸载模块

```
#装载
modprobe module_name
insmod /lib/modules/kernel_release/modules_name.ko

#卸载
modprobe -r module_name
rmmod modules_name
```

内核参数

系统管理员可以通过用户空间访问和监控内核。`/proc`目录是一个伪文件系统，此目录下是内核的映射文件。`/proc/sys`目录下是内核的参数映射文件，这些文件是可读写的。

修改内核参数

立即生效，但重启系统会失效

```
#方法一
echo Parameter_value > /proc/sys/dir_name/Parameter_name
#示例
echo 1 > /proc/sys/net/ipv4/icmp_echo_ignore_all

#方法二
sysctl -w dir_name.{...}.Parameter_name=Parameter_value
#示例
sysctl -w net.ipv4.icmp_echo_ignore_all=1
```

永久生效，但不立即生效

```
#修改内核参数配置文件
/etc/sysctl.conf
#如果需要立即生效，则执行如下命令，让内核重新读取配置文件
sysctl -p
#显示所有内核参数及其值
sysctl -a
```

内核编译

源码目录

```
[xuekaixin@root /usr/src/linux-2.6.38.1]#ls -l
drwxrwxr-x. 26 root root   4096 Jan  5 18:27 arch #硬件体系结构相关的代码
```

```
drwxrwxr-x.  2 root root   4096 Jan  5 18:27 block
-rw-rw-r--.  1 root root  18693 Mar 24  2011 COPYING
-rw-rw-r--.  1 root root  93910 Mar 24  2011 CREDITS
drwxrwxr-x.  3 root root  20480 Jan  5 18:27 crypto
#常用加密和散列算法（如AES、SHA等），还有一些压缩和CRC校验算法
drwxrwxr-x. 90 root root  12288 Jan  5 18:27 Documentation #说明文档
drwxrwxr-x. 91 root root   4096 Jan  5 18:27 drivers #系统所有的设备驱动程序
drwxrwxr-x. 37 root root   4096 Jan  5 18:27 firmware
drwxrwxr-x. 70 root root  12288 Jan  5 18:27 fs #支持的文件系统
drwxrwxr-x. 23 root root   4096 Jan  2 22:01 include #编译所需要的头文件
drwxrwxr-x.  2 root root   4096 Jan  5 18:27 init
drwxrwxr-x.  2 root root   4096 Jan  5 18:27 ipc #进程间通信相关代码
-rw-rw-r--.  1 root root   2464 Mar 24  2011 Kbuild
-rw-rw-r--.  1 root root    252 Mar 24  2011 Kconfig
drwxrwxr-x.  8 root root  12288 Jan  5 18:31 kernel
#主要的核心代码，此目录下的文件实现了大多数linux系统的内核函数
drwxrwxr-x.  8 root root  16384 Jan  5 18:27 lib #目录包含了核心库代码，不过与处理器结构相关的库代码被放在arch/*/lib/目录下
-rw-rw-r--.  1 root root 191716 Mar 24  2011 MAINTAINERS
-rw-rw-r--.  1 root root  52126 Mar 24  2011 Makefile
drwxrwxr-x.  2 root root  12288 Jan  5 18:27 mm #这个目录包括所有独立于cpu体系结构的内存管理代码
-rw-r--r--.  1 root root      0 Jan  5 18:29 Module.symvers
drwxrwxr-x. 53 root root   4096 Jan  5 18:27 net #网络协议相关代码
-rw-rw-r--.  1 root root  17512 Mar 24  2011 README
```

```
-rw-rw-r--.  1 root root   3371 Mar 24  2011 REPORTING-BUGS
drwxrwxr-x.  9 root root   4096 Mar 24  2011 samples
drwxrwxr-x. 13 root root   4096 Jan  5 18:31 scripts #用于配置内核文件的脚本文件
drwxrwxr-x.  8 root root   4096 Jan  5 18:27 security #SElinux模块代码
drwxrwxr-x. 21 root root   4096 Jan  5 18:27 sound #音频设备驱动程序
drwxrwxr-x.  9 root root   4096 Mar 24  2011 tools
drwxrwxr-x.  3 root root   4096 Jan  5 18:27 usr
drwxrwxr-x.  3 root root   4096 Mar 24  2011 virt
```

完整编译

1.获取源码

www.kernel.org、http://vault.centos.org

2.解压内核文件到指定目录

```
#如果是linux-kernel_release.tar.{bz2|xz}格式的包
tar -{x|J}f linux_kernel_release.tar.xz -C /usr/src

#如果是kernel-kernel_release.src.rpm格式的包
useradd mockbuild
rpm -ivh kernel_kernel_release.src.rpm
cd ~/rpmbuild/SOURCES/
tar -xf linux-kernel_release.tar.bz2 -C /usr/src
```

3.在当前目录创建一个软链接

```
ln -s linux-kernel-release linux
```

4.生成编译配置文件

可以将当前系统内核编译时的配置文件文件复制到当前目录下并以`.config`命名，以这个为模板进行修改。

```
cp /boot/config-kernel_release .config
```

进入配置窗口选择进行选择编译的模块及内核功能，可以用以下命令，选择完成后会在目录下生成`.config`配置文件

```
make menuconfig #命令行界面，打开一个选择菜单
make nconfig #新版的命令行界面

make gconfig #Gnome桌面环境使用，需要安装图形开发库
make kconfig #KDE桌面环境使用，需要安装图形开发库

make allyesconfig #编译所有模块

[*] #编译进内核功能
[M] #编译成模块功能
[] #不编译此功能模块
```

5.编译安装

```
make
make modules_install
make install
```

6.测试

内核编译安装后，会在`/boot`目录下生成对应版本的`initamfs`和`vmlinux`文件，并自动在grub的配置文件`grub.conf`文件内添加一个title，且默认新内核是首启动内核。重启，测试编译好的内核是否可以正常运行。

7.二次编译

如果编译失败，需要重新编译

```
make clean #清理编译好的二进制文件
make mrproper #清理文件，连同配置文件删除，执行此命令前应备份配置文件
make disclean #重置源码目录树
```

部分编译

1.只编译某个目录下的代码

```
make SUBDIR=arch/
```

2.只编译部分模块

```
make M=drivers/net/
```

3.只编译某一个模块

```
 make dirvers/net/pcnet32.ko
```

4.将编译结果放置于其他目录

```
make O=/usr/src/tmp/kernel
```

交叉编译

交叉编译简单的说就是从一个平台上生成另一个平台上可执行的代码。当目的主机不具备编译环境或者由于资源所限，无法完成编译时，就需要在其他平台进行交叉编译。

```
make ARCH=ARCHNAME
#当编译需要在某平台运行的的内核时，可以使用默认的编译配置文件
#默认配置文件在arch/arch_name/configs/
make ARCH=ARCHNAME arch_defconfig
```

screen工具

screen可以在当前窗口虚拟一个窗口出来，用来运行需要长时间执行的任务，在虚拟出来的窗口中执行的任务不会因为远程连接中指而中止。例如编译内核时，需要较长的时间，所以可以在screen打开一个窗口，进行编译。

```
screen #新建一个窗口
screen -ls #列出当前打开的窗口
screen -r screen_id #恢复指定的一个窗口
```