

# 第4章 内核编译与管理



- 内核概述
- 内核版本
- 内核编译与安装
- 查看内核模块
- 内核模块加载与卸载
- 修改内核参数



# 目录页

Part 1

Part 2

Part 3

## Linux内核



## 内核编译安装



## 内核模块管理





## 4.1.1 内核概述



**内核** (Kernel) 是**操作系统的核心**，它是一个系统软件，负责管理系统中的进程、内存、设备驱动程序、文件和网络等。在操作系统中，内核独立于普通应用程序，它工作在**内核空间**。内核既要管理应用程序的运行，又要管理硬件设备的运作。



## 4.1.1 内核概述



进程管理

中断管理

模块管理

文件系统

进程间通信

系统启动

定时器

内存管理

虚拟文件系统接口

设备驱动程序

网络管理



## 4.1.1 内核概述

**单内核**是个独立的大进程，通常以单个静态二进制文件的形式存放于磁盘上。

**微内核**的功能被划分为多个独立的进程，所有的进程都保持独立并运行在各自的地址空间上。

单内核

微内核

Linux内核：  
单/微内核



## 4.1.2 内核的开发与更新

内核的开发遵循GPL协议，为了确保开发过程有序进行，Linux采用双树系统管理内核版本。



一些新特性、实验性改进等都在属于开发树的内核中进行。



新特性在开发树中完成测试之，在稳定树中进行相同的改进。



## 4.1.3 内核版本

**Linux内核版本命名在不同的时期有其不同的规范，自第一个内核版本发布至今，内核版本的命名大约经历了四个阶段。**

**1**

从内核第一个0.01版本发布到1.0版本，这期间从0.12版本开始遵循GPL规范。

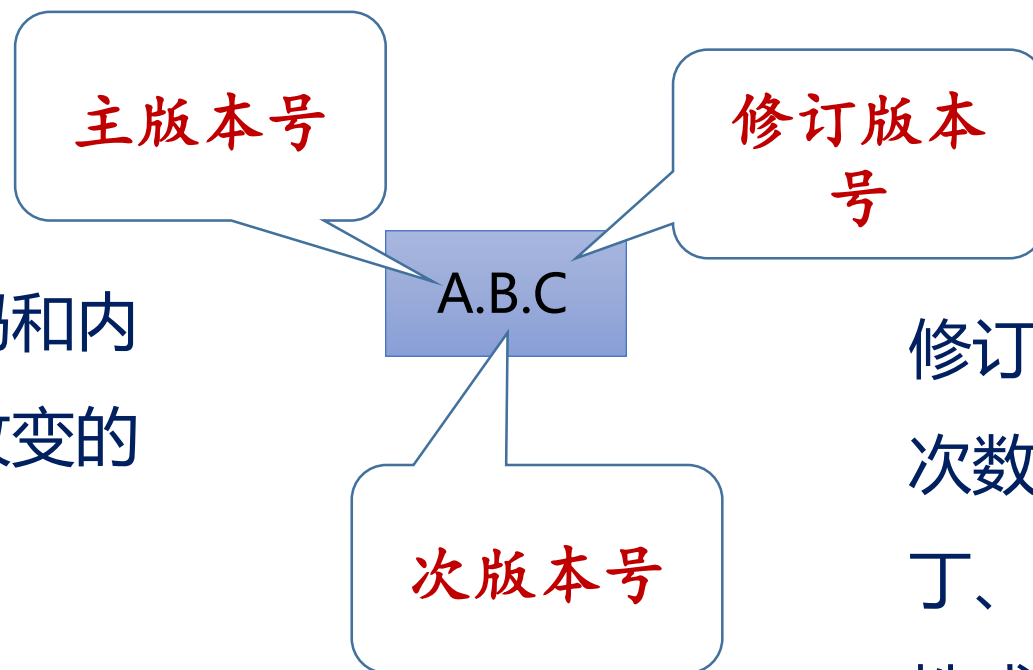
**2**

1.0版本发布之后，直到2.6版本，内核版本的命名格式为**A.B.C**。



## 4.1.3 内核版本

版本号只有在代码和内核的概念有重大改变的时候才会改变。



修订版本号表示内核修改的次数，它在内核增加安全补丁、修复Bug、实现新的特性或驱动时都会改变

次版本号根据传统的“奇-偶”系统版本号来分配：奇数为开发版本，偶数为稳定版本。





## 4.1.3 内核版本

Linux内核版本命名在不同的时期有其不同的规范，自第一个内核版本发布至今，内核版本的命名大约经历了四个阶段。

3

2004年2.6版本发布之后，直到2011年，这一段时间内核版本比较稳定。

4

2011年5月，Linus宣布为了纪念Linux发布20周年，在2.6.39版本发布之后，将内核版本提升到了3.0。



## 4.1.3 内核版本

### uname命令

uname -r 用于查看内核信息。

**uname -r**



# 目录页

Part 1

Part 2

Part 3

## Linux内核



## 内核编译安装



## 内核模块管理





## 4.2.1 获取内核源码

内核源码可以从内核的官方网站 (<https://www.kernel.org/>) 获取。

Protocol Location

<https://www.kernel.org/pub/>  
<https://git.kernel.org/>  
<https://rsync.kernel.org/pub/>

Latest Stable Kernel:



5.3.2

长期支持版

	rc1	2019-09-30	[tarball]	[patch]	[view diff]	[browse]
stable:	5.3.2	2019-10-01	[tarball]	[pgp] [patch] [inc. patch]	[view diff]	[browse] [changelog]
stable:	5.2.18	2019-10-01	[tarball]	[pgp] [patch] [inc. patch]	[view diff]	[browse] [changelog]
longterm:	4.19.76	2019-10-01	[tarball]	[pgp] [patch] [inc. patch]	[view diff]	[browse] [changelog]
longterm:	4.14.146	2019-09-21	[tarball]	[pgp] [patch] [inc. patch]	[view diff]	[browse] [changelog]
longterm:	4.9.194	2019-09-21	[tarball]	[pgp] [patch] [inc. patch]	[view diff]	[browse] [changelog]
longterm:	4.4.194	2019-09-21	[tarball]	[pgp] [patch] [inc. patch]	[view diff]	[browse] [changelog]
longterm:	3.16.74	2019-09-23	[tarball]	[pgp] [patch] [inc. patch]	[view diff]	[browse] [changelog]
linux-next:	next-20191004	2019-10-04				[browse]



## 4.2.2 编译安装

### 编译安装步骤

1

➤ 解压：包括解压、清理、安装必要的依赖。

2

➤ 内核环境配置

3

➤ 编译安装



## 4.2.2 编译安装

### 1 解压

- `sudo tar -xf linux-4.19.76.tar.xz`

### 2 内核环境配置

#### (1) 安装必要的依赖

```
sudo apt-get install gcc make libncurses5-dev openssl libssl-dev
```

```
sudo apt-get install build-essential
```

```
sudo apt-get install pkg-config
```

```
sudo apt-get install libc6-dev
```

```
sudo apt-get install bison
```

```
sudo apt-get install flexsudo apt-get install libelf-dev
```



## 4.2.2 编译安装

### 2 内核环境配置

(2) 复制本机的内核配置文件到新内核的目录下

```
sudo cp /boot/config-4.15.0-62-generic .config
```



## 4.2.2 编译安装

### 2 内核环境配置

#### (3) 配置菜单

sudo make menuconfig

```
.config - Linux/x86 5.3.0 Kernel Configuration

Linux/x86 5.3.0 Kernel Configuration
Arrow keys navigate the menu. <Enter> selects submenus ---> (or empty
submenus ----). Highlighted letters are hotkeys. Pressing <Y>
includes, <N> excludes, <M> modularizes features. Press <Esc><Esc> to
exit, <?> for Help, </> for Search. Legend: [*] built-in [ ]

*** Compiler: gcc (Ubuntu 7.4.0-1ubuntu1~18.04.1) 7.4.0 ***
General setup --->
[*] 64-bit kernel
Processor type and features --->
Power management and ACPI options --->
Bus options (PCI etc.) --->
Binary Emulations --->
Firmware Drivers --->
[*] Virtualization --->
General architecture-dependent options --->
└(+)
```

<Select>   <Exit>   <Help>   **<Save>**   <Load>





## 4.2.2 编译安装

### 3.编译安装

#### (1) 编译

`sudo make`

#### (2) 安装模块

`sudo make modules_install`

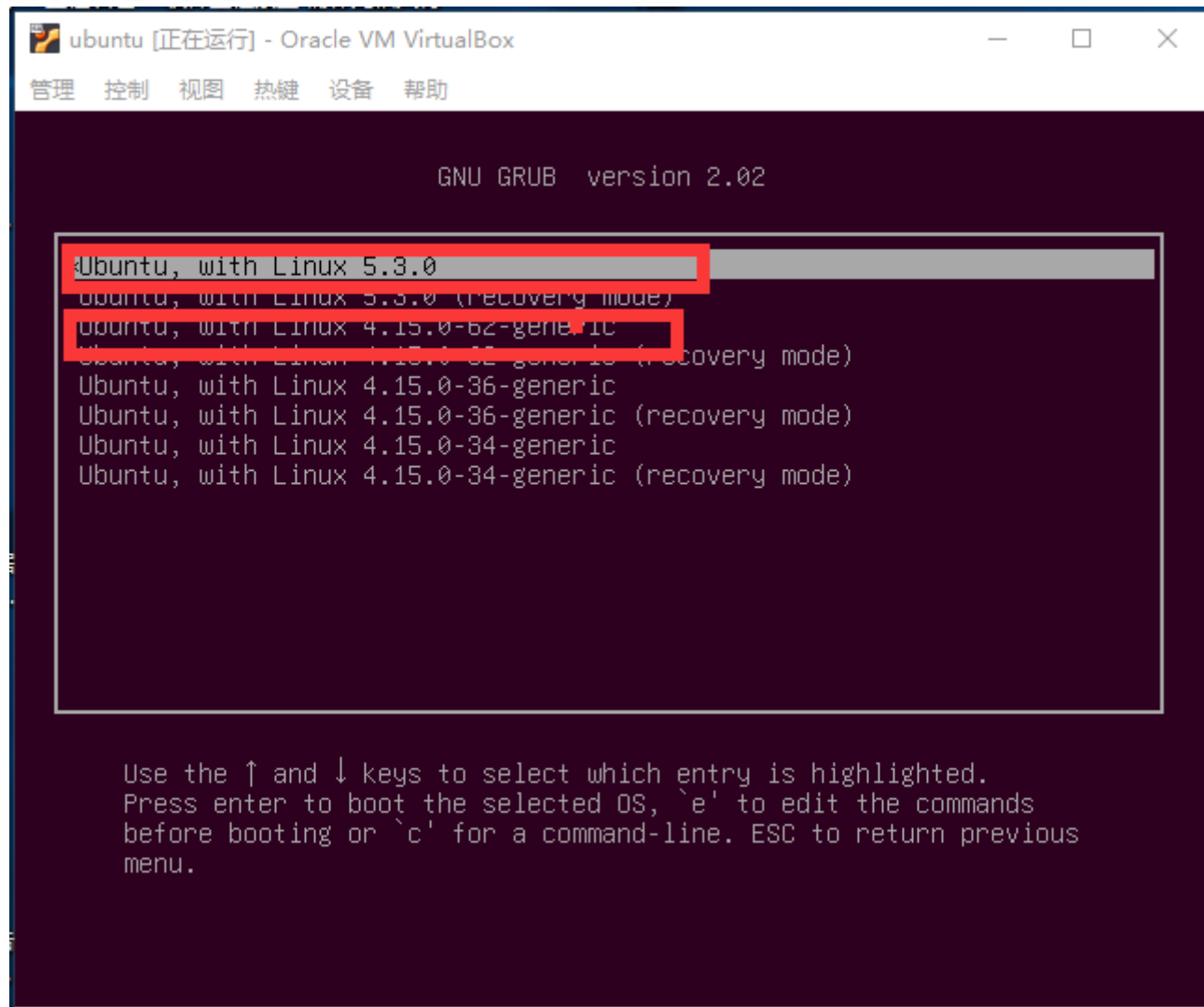
#### (3)安装内核

`sudo make install`



## 4.2.2 编译安装

### 3.编译安装 引导界面





# 目录页

Part 1

Part 2

Part 3

## Linux内核



## 内核编译安装



## 内核模块管理





## 4.3.1 内核模块概述

**内核模块**是Linux对外部提供的一个接口，它的全称是动态可加载内核模块（Loadable Kernel Module, LKM），简称模块。Linux之所以提供模块机制，是因为Linux内核是一个单内核，它把所有的内容集成在一起，虽然效率高，但可扩展性较差，模块机制正好为了弥补这一设计缺陷。

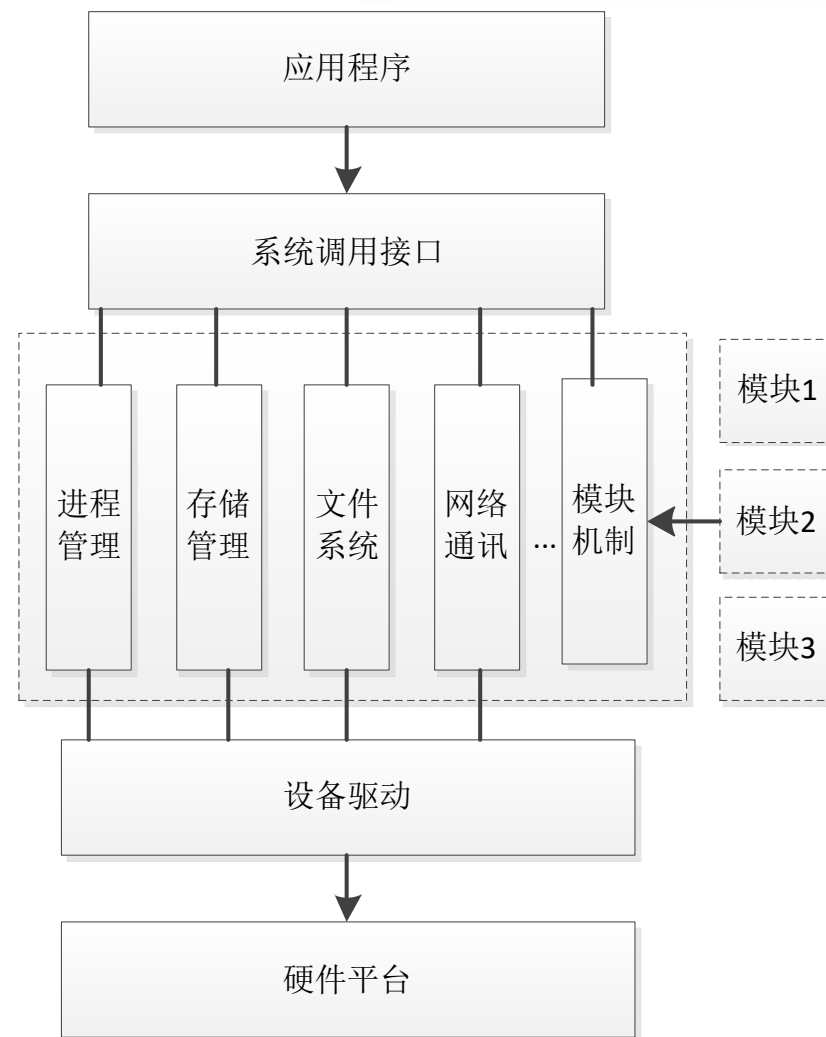


## 4.3.1 内核模块概述

模块是具有独立功能的程序，它可以被单独编译，但不能独立运行，因为它只有初始化函数，没有单独的main()函数。动态加载模块时，模块被链接到内核并作为内核的一部分在**内核空间**运行。

内核模块存放的位置：

`/lib/modules/`uname -r`/kernel`





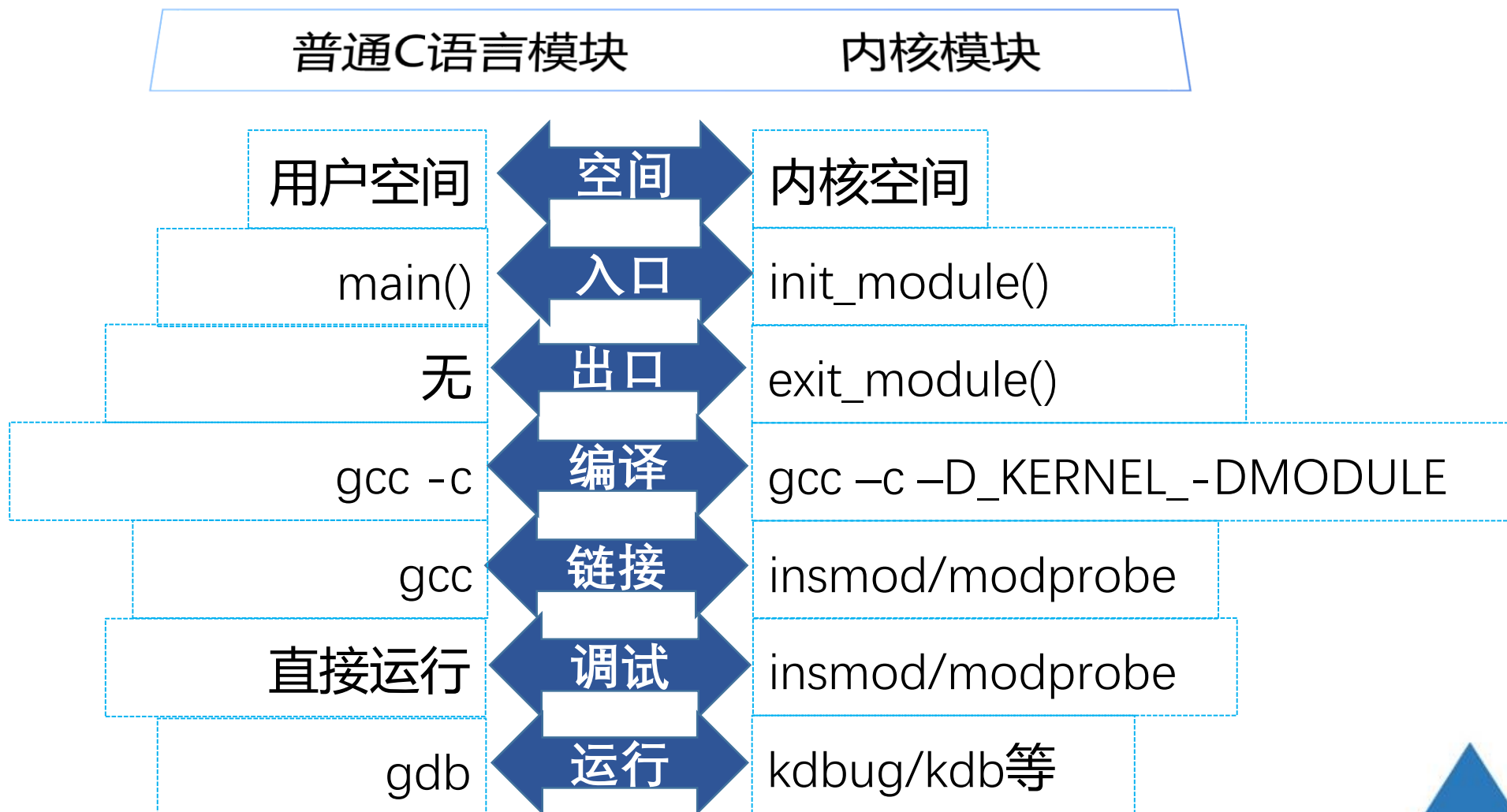
## 4.3.1 内核模块概述

### 模块程序结构





## 4.3.1 内核模块概述





## 4.3.1 内核模块概述

### 优点

- 1 扩展了内核的功能
- 2 具有良好的可移植性
- 3 被链接后，作用域同内核

### 缺点

- 1 消耗内核性能
- 2 使用不当造成系统崩溃
- 3 需要维护内核符号表



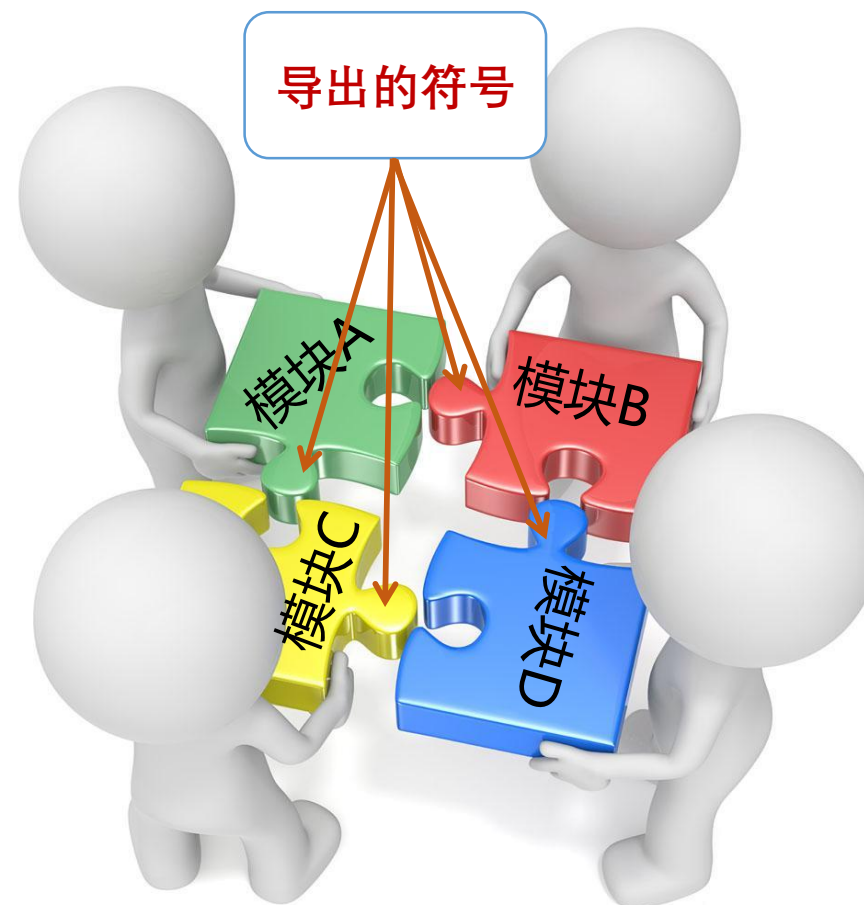


## 4.3.1 内核模块概述

### 模块间的依赖关系

如果一个内核模块A要引用另一个内核模块B中的全局变量或函数（符号），就说模块B被模块A引用，或者说模块A依赖模块。

加载模块A之前需要先加载模块B，用 **EXPORT\_SYMBOL()/EXPORT\_SYMBOL\_GPL()** 将符号导出。





## 4.3.2 查看内核模块

### 1 lsmod命令

lsmod命令可以列出目前系统中已经加载的所有模块，这个命令没有任何参数，直接执行即可。

### 2 modinfo命令

modinfo命令用于查看某一个内核模块具体信息。

**modinfo [选项] 模块名**



## 4.3.2 查看内核模块

modinfo常用选项：

- -a选项：显示模块开发人员信息。
- -d选项：显示模块的使用说明。
- -l选项：显示模块遵守的规范，都是遵守GPL规范。
- -h选项：显示帮助信息。
- -p选项：显示模块所支持的参数。
- -V选项：显示模块版本信息。



## 4.3.3 加载与卸载

### 1 modprobe命令

modprobe命令不仅可以动态的加载模块，还可以卸载模块，通过modprobe命令加载、卸载模块时。

**modprobe [选项] 模块名**

modprobe命令加载卸载模块时，不需要带模块的路径与扩展名。



## 4.3.3 加载与卸载

### 2 insmod命令

insmod命令的功能和modprobe类似，它也可实现加载模块的功能，但insmod命令在加载模块时需要**带上模块的绝对路径**，并且一定**要有后缀名.ko**。

### 3 rmmod命令

rmmod命令用来卸载模块，modprobe命令和insmod命令加载的模块都可以使用这个命令完成卸载。

**rmmod [选项] 模块名**



## 4.3.4 修改内核参数

(1) 通过命令进行修改

/proc/sys/目录下存放着绝大部分的内核参数，且这些参数可以在系统运行时被修改。

系统重启后无效。

(2) 通过/etc/sysctl.conf配置文件修改内核参数

/etc/sysctl.conf配置文件用于保存或记录内核参数的存放位置，系统在启动时会通过此文件去读取内核参数，因此可以将内核参数写在此文件中以达到修改的目的。

系统重启后有效。



## 4.4 本章小结



本章主要讲解了与**内核**相关的知识，包括内核的**概念**、如何**获取内核源码**、**内核的编译安装**、**内核模块的管理**。通过本章的学习，读者应对Linux内核有一个整体的了解，并掌握**内核安装**、**内核模块管理**的方法。