



Módulos do Kernel do Linux

Versão 2.6.x

Walter Fetter Lages

`w.fetter@ieee.org`

Universidade Federal do Rio Grande do Sul

Escola de Engenharia

Departamento de Engenharia Elétrica

ENG04008 Sistemas de Tempo Real



Introdução

- O kernel do Linux pode ser modularizado
 - Módulos do kernel podem ser carregados e descarregados da memória sem boot
 - Normalmente são utilizados para implementar
 - Drivers
 - Filesystems
 - Chamadas de sistema
 - Disciplinas de linha TTY
 - Interpretadores de executáveis
- O código nos módulos executa a nível de kernel
 - Podem comprometer todo o sistema

Carga/Descarga de Módulos



- `insmod [options] module`
- `rmmmod [options] module`
- `lsmod`
- `modinfo [options] module`
 - Insere, remove, lista módulos carregados e informações sobre um módulo
- `depmod [options] module1.o module2.o...`
 - Gera dependência entre módulos
 - Cria `/lib/modules/*/modules.dep`
- `modprobe [options] module`
 - Insere módulo e os demais módulos necessários



Programação de Módulos

- Módulos são arquivos objeto
- As constantes `__KERNEL__` e `MODULE` devem ser definidas antes de qualquer outra coisa, inclusive includes
 - Usualmente isto é feito no início do programa fonte ou na linha de comando do compilador
 - Módulos devem definir as funções
 - `int init_module(void)`
 - Deve retornar 0 para sinalizar sucesso
 - Chamada quando o módulo é carregado
 - `void cleanup_module(void)`
 - Chamada quando módulo é descarregado



Macros

- `#include <linux/module.h>`
- `MODULE_INFO (tab, info)`
- `MODULE_ALIAS (alias)`
- `EXPORT_SYMBOL (symbol)`
- `MODULE_AUTHOR (autor_str)`
- `MODULE_LICENSE (license)`
 - "GPL", "GPL v2", "GPL and additional rights", "Dual BSD/GPL", "DUAL MPL/GPL", "Proprietary"
- `MODULE_VERSION (description_str)`
- `MODULE_DESCRIPTION (description_str)`
- `MODULE_SUPPORTED_DEVICE (device_str)`
- **Ainda não implementada**



Passagem de Parâmetros

- Os módulos podem receber parâmetros ao serem carregados
- Parâmetros são variáveis globais no módulo que podem ser inicializadas na carga
- `module_param(name, type, perm)`
 - `byte, short, ushort, int, uint, long, ulong, charp, bool, invbool`
- `module_param_array(name, type, nump, p`
- `MODULE_PARM_DESC(parm, desc_str)`



Exemplo

```
#include <linux/module.h>  
#include <linux/kernel.h>  
#include <asm/io.h>  
#include <asm/segment.h>
```

```
static int lpt=0x378;  
static char data=0;
```

```
module_param(data, byte, 0);  
MODULE_PARM_DESC(data, "dado p/porta");
```



Exemplo

```
int init_module(void)
{
    printk(KERN_INFO "init\n");
    outb(data, lpt);
    return 0;
}

void cleanup_module(void)
{
    printk(KERN_INFO "cleanup\n");
}
```



Compilação

- A partir da versão 2.6.0 do kernel o normal é os módulos serem compilados na árvore do kernel
- Exisem *hooks* para compilação fora da árvore do kernel

```
KERNEL_SOURCE=/lib/modules/`name`-r/build
EXTRA_CFLAGS+=-I/usr/include -D__KERNEL__ -DMODULE

obj-m += exemplo.o

all: exemplo.ko

exemplo.ko: exemplo.c
    $(MAKE) -C $(KERNEL_SOURCE) O=. SUBDIRS=$(PWD) modules
```



Exemplo

```
insmod exemplo.ko data=0xff  
rmmod exemplo.ko
```