Introduzione ai Moduli del Kernel di Linux

E.Mumolo, DEEI

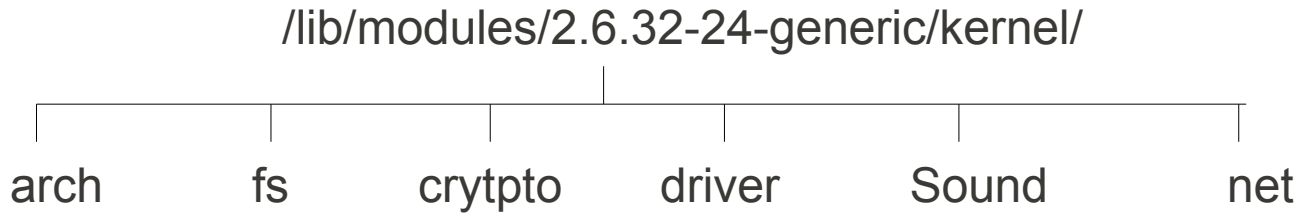
mumolo@units.it

I moduli di Linux

- Introducono la possibilità di compilare parti del kernel sotto forma di moduli
- I MODULI DEL KERNEL SONO CARICATI DINAMICAMENTE!!
  - insmod - carica in ordine i moduli: A richiede B richiede C: insmod C B A
  - rmmod – scarica in ordine: rmmod A B C
- Suffissi:
  - .c sorgenti in C
  - .o file oggetto
  - .mod.o file che contiene informazioni aggiuntive sulla compilazione
  - .ko Kernel Object (dal Kernel 2.6): è la combinazione del file .o e di un file .mod.o

Uno sguardo ai moduli

- Contenuti in `/lib/modules`
- Cenno all'albero dei moduli:



cpuid.ko fpu.ko .. reiserfs.ko ...

- Nel file `modules.dep` c'è la lista di tutti i moduli
- modules.dep è ottenuta con il comando `depmod -a`

I moduli di Linux

- I moduli sono utilizzati per realizzare:
  - device driver -- gestisce l'interazione a basso livello con un particolare dispositivo;
  - filesystem driver --- interpreta le informazioni memorizzate su un media come file e directory;
  - system call --- gestiscono nuove system call o soppiantano quelle originali;
  - network driver software --- interpreta uno specifico protocollo di rete;
  - executable interpreter software --- interpreta gli script secondo una determinata sintassi propria del linguaggio di programmazione considerato;

I moduli di Linux

- Programmi per la gestione dei moduli:
  - insmod Carica manualmente i moduli del kernel (NB: rispettare le dipendenze!).
  - rmmod Scarica manualmente i moduli del kernel (NB: rispettare le dipendenze!).
  - `rmmod -a` scarica tutti i moduli inutilizzati
  - lsmod Elenca i moduli caricati nel kernel.
  - depmod Rigenera il file modules.dep delle dipendenze tra i moduli. Qualche esempio delle dipendenze:

    ...
    kernel/drivers/char/pc8736x_gpio.ko: kernel/drivers/char/nsc_gpio.ko
    kernel/drivers/char/hangcheck-timer.ko:
    kernel/drivers/char/tpm/tpm.ko: kernel/drivers/char/tpm/tpm_bios.ko
    kernel/sound/core/snd-page-alloc.ko:
    kernel/sound/core/snd-rawmidi.ko: kernel/sound/core/seq/snd-seq-device.ko
    kernel/sound/core/snd.ko

I moduli di Linux

- Programmi per la gestione dei moduli:
  - modprobe Carica o rimuove un modulo rispettando le dipendenze.
  - /etc/modules.conf Configurazione dei moduli utilizzati.
  - kerneld Programma demone per il carico e lo scarico automatico dei moduli.

Mappa del kernel

SystemInterfaces

DeviceModel

SystemRun

Generic HW access

DeviceAccess

BusDrivers

Processi (fork,exec...)

Threads

Synchronization

Scheduler

Interrupt

CPU

MemoryAccess

VirtualMemory

MemoryMapping

LogicalMemory

PageAllocator

PhysicalMemory Operation

FileDirectory Access

VirtualFileSystem

PageCache Swap

LogicalFileSystem

BlockDevices

DiskController

Socket Access

Protocol Families

Networking Storage

Protocols

VirtualNetwork Device

Network Device Driver

Compilazione del kernel

- Sorgenti del kernel : `/usr/src/linux`
- Sorgente del nuovo kernel: `/home/name`
- Per configurare e creare il kernel:

```
cd /usr/src/linux
make O=/home/name/build/kernel menuconfig
make O=/home/name/build/kernel
# kernel compresso:/vmlinuz, /boot/vmlinuz, /bzImage o
/boot/bzImage
make O=/home/name/build/kernel  # non comprime
```

- L'opzione `'O=output/dir'` serve ad indicare la directory del kernel
- Pulizia della directory sorgente: `make mrproper`
- `Menuconfig serve per modificare il makefile`

I moduli di Linux

- Codice codificato in C

- Ogni modulo del kernel ha un'entry point (init_module) e un exit point (cleanup_module)

- In definitiva: la struttura utilizza 3 parti principali scritte dall'utente
  - Funzione che inizializza il sistema, definisce le caratteristiche dei vari task e IPC (**init_module**)
  - Funzione che realizza lo scopo del modulo
  - Funzione che rilascia le risorse (**cleanup_module**)

Alcune API del Kernel di Linux

http://gnugeneration.com/books/linux/2.6.20/kernel-api/

- Gestione della memoria
  - kmalloc — alloca memoria
  - kzalloc — alloca e azzera memoria
  - kmem_cache_create — creazione di una cache
- Schedulazione
- Temporizzazione
- Funzioni interne
  - kthread_create – crea un kernel thread
- Accesso allo spazio utente
  - copy_to_user — copia un blocco dati dallo spazio kernel allo spazio utente
  - copy_from_user — copia un blocco dati dallo spazio utente allo spazio kernel
- IPC, Networking
- Gestione dispositivi a blocchi
- Gestione dispositivi a carattere
  - register_chrdev — registra un major number per dispositivi a carattere.
- Funzioni di utilità
  - printk — stampa un messaggio del kernel

I moduli di Linux

- Primo esempio:

```
#include <linux/kernel.h>
#include <linux/module.h>
MODULE_LICENSE("GPL");

int init_module(void)  //entry point
{
      printk("Hello world!\n");
// printk = scrive in /var/log/messages
      return 0;
}

void cleanup_module(void)     //exit point
{
      printk("Goodbye world!\n");
      return;
}
```

I moduli di Linux

- Compilazione sorgenti:
  - Il comando base per compilare il modulo è:

```
make -C /lib/modules/2.6.32-41-generic/build M=/home/em/corso/driver hello.ko
```

Opzione per passare nella directory indicata e usare il Makefile ivi contenuto

Opzione per tornare nella directory di partenza

Kernel object da costruire

  - Concisamente (makefile):

Variabile obj-m

Costruisce i kernel object contenuti nella variabile obj-m

```
obj-m = hello.o
KVERSION = $(shell uname -r)
all:
   make -C /lib/modules/$(KVERSION)/build M=$(PWD) modules
clean:
   make -C /lib/modules/$(KVERSION)/build M=$(PWD) clean
```

- Per eseguire il modulo: `insmod <modulo.ko>`
- Per terminare : `rmmod <modulo.ko>`
- Attenzione (1): printk scrive nel ‘kernel ring buffer
- Attenzione (2): printk può non essere predicibile: rt_printk versione RT

I moduli di Linux

- Logging in Linux: metodo per salvare eventi speciali su vari file contenuti in /var/log
- Configurazione del logging: **/etc/syslog.conf**.
- Alcuni Log file importanti
  - **/var/log/messages**: file principale, memorizza eventi di boot, stato, errori, servizi in esecuzione …
  - **/var/log/XFree86.0.log**: risultati della esecuzione del server di Xwindows Xfree86
  - **/var/log/secure** – messaggi di autentificazione, servizi xinetd
  - **/var/log/vsftpd.log** – transazioni FTP
  - **/var/log/maillog** – transazioni di posta.
  - **syslogd** – riceve messaggi da vari demoni
  - **klogd** – messaggi del kernel

I moduli di Linux

- Attenzione: i log prima di essere memorizzati nel file si trovano nel ‘Kernel ring Buffer’
- Strumenti per analizzare i log: Comandi di shell

dmesg:

```
dmesg [-c][-s bufsize]print or control the kernel ring buffer
dmesg - c  Clear the ring buffer contents after printing.
```

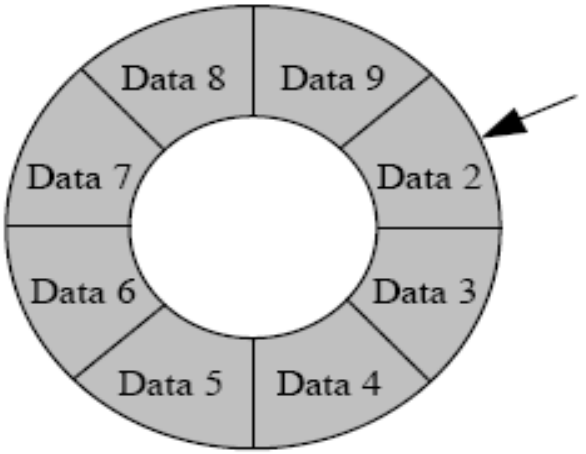
tail: tail -f /var/log/messages (Lo switch –f continua a visualizzare ulteriori eventi quando arrivano)

more: more /var/log/XFree86.0.log

less: less /var/log/messages

logger: comando per inserire messaggi nel log file

- Kernel Ring Buffer:



I moduli di Linux

Comando `printk/rt_printk :` Scrive sul kernel Ring buffer.

Per leggere/cancellare il buffer:

o mediante chiamate di sistema:

```
syslog,  klogctl - read and/or clear kernel message ring buffer;
set  console_loglevel
```

o mediante comandi Linux

```
$tkadmin dump ringbuffer ringbuffer.out
```

In definitiva:

```
dmesg -c
insmod hello.ko
dmesg
rmmod hello.ko
dmesg
```

Oppure: `tail -f /var/log/messages`
Visualizza il ring buffer quando si verifica un evento