



# Introduzione a Linux

Linux teoria  
Parte 2

**Ruggero Donida Labati**

**Laboratorio di Sistemi Operativi**  
Università degli Studi di Milano  
Dipartimento di Informatica  
A.A. 2022/2023

R. DONIDA LABATI – INTRODUZIONE A LINUX – LINUX TEORIA

1

## Panoramica della lezione

- Verranno descritti i meccanismi di gestione della memoria in Linux
- Verranno trattati i file system Linux
- Verrà descritta la gestione dell'I/O in Linux
- Verrà trattata la comunicazione tra processi in Linux
- Verranno descritti aspetti di sicurezza e gestione della rete in Linux



R. DONIDA LABATI – INTRODUZIONE A LINUX – LINUX TEORIA

2

## Sommario (1/2)

1. Gestione della memoria
  - Mappatura della memoria fisica
  - Divisione della memoria per un buddy heap
  - Allocatore slab in Linux
  - Memoria virtuale
  - Swapping e paginazione
  - Memoria virtuale del kernel
  - Esecuzione e caricamento di programmi
  - Schema della memoria per programmi ELF
  - Linking statico e dinamico



R. DONIDA LABATI – INTRODUZIONE A LINUX – LINUX TEORIA

3

## Sommario (2/2)

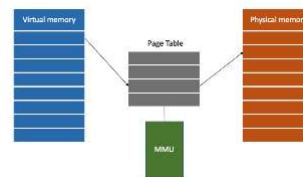
2. File system
  - Il file system ext3
  - Politiche di allocazione dei blocchi in ext3fs
  - Journaling
  - Il file system proc
3. Input e output
  - Dispositivi a blocchi
  - Dispositivi a caratteri
4. Comunicazione tra processi
  - Scambio di dati tra processi
5. Gestione della rete
6. Sicurezza

R. DONIDA LABATI – INTRODUZIONE A LINUX – LINUX TEORIA

4

## 1. Gestione della memoria (1/2)

- Il gestore della memoria Linux si occupa di allocare e liberare
  - Pagine
  - Gruppi di pagine
  - Piccoli blocchi di memoria
- Integra meccanismi per la gestione della memoria virtuale



R. DONIDA LABATI – INTRODUZIONE A LINUX – LINUX TEORIA

5

## 1. Gestione della memoria (2/2)

- Divide la memoria in 3 zone, in accordo alle caratteristiche hardware

Divisione della memoria per x86

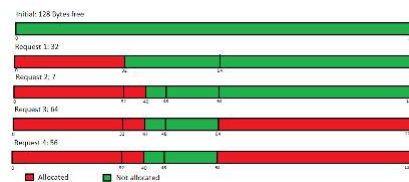
zone	physical memory
ZONE_DMA	< 16 MB
ZONE_NORMAL	16 .. 896 MB
ZONE_HIGHMEM	> 896 MB

R. DONIDA LABATI – INTRODUZIONE A LINUX – LINUX TEORIA

6

## Mappatura della memoria fisica (1/4)

- L'allocatore della memoria si occupa di allocare e rilasciare tutte le pagine fisiche e può allocare gruppi di pagine fisiche contigue su richiesta
- Usa l'algoritmo buddy-heap per mantenere traccia delle pagine libere

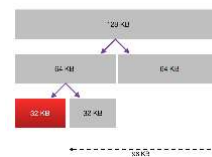


R. DONIDA LABATI – INTRODUZIONE A LINUX – LINUX TEORIA

7

## Mappatura della memoria fisica (2/4)

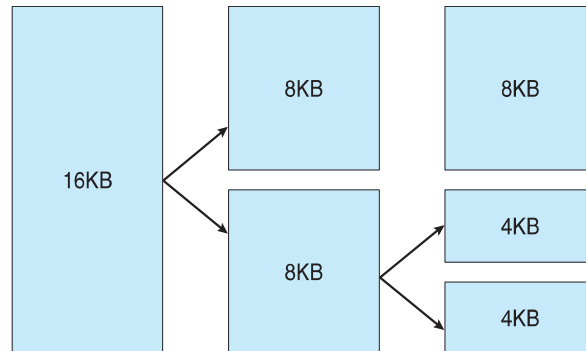
- Buddy-heap
  - Ogni regione della memoria allocabile viene aggregata con un partner adiacente
  - Quando due regioni partner allocate vengono liberate, vengono combinate per creare una regione più grande
  - Se una richiesta di allocazione di una piccola regione di memoria non può essere soddisfatta, una regione libera più grande viene suddivisa in due partner per soddisfare la richiesta



R. DONIDA LABATI – INTRODUZIONE A LINUX – LINUX TEORIA

8

## Divisione della memoria per un buddy heap

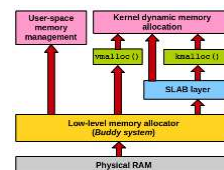


R. DONIDA LABATI – INTRODUZIONE A LINUX – LINUX TEORIA

9

## Mappatura della memoria fisica (3/4)

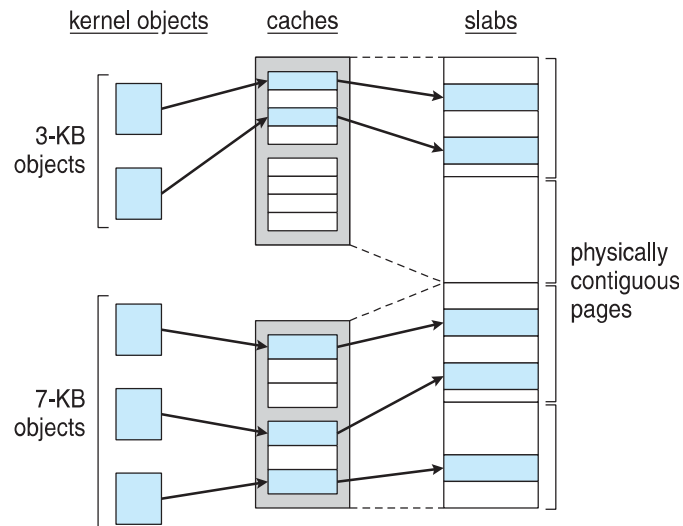
- L'allocazione della memoria per il kernel Linux può essere
  - Statica (i driver riservano aree contigue di memoria al boot del sistema)
  - Dinamica (tramite l'allocatore delle pagine)
- Linux utilizza anche l'allocatore slab per la memoria del kernel



R. DONIDA LABATI – INTRODUZIONE A LINUX – LINUX TEORIA

10

## Allocatore slab in Linux

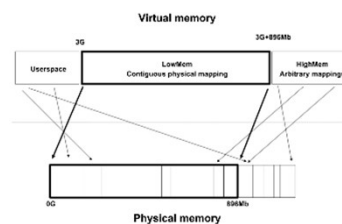


R. DONIDA LABATI – INTRODUZIONE A LINUX – LINUX TEORIA

11

## Mappatura della memoria fisica (4/4)

- Vengono inoltre utilizzate la cache delle pagine e la memoria virtuale per gestire la memoria fisica
  - La cache delle pagine è una cache principale del kernel per i file e i principali dispositivi di I/O a blocchi
  - La cache delle pagine salva intere pagine di file per la gestione dell'I/O in locale e in rete



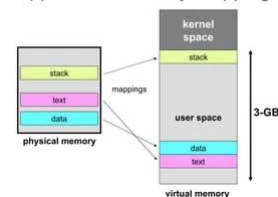
R. DONIDA LABATI – INTRODUZIONE A LINUX – LINUX TEORIA

12

## Memoria virtuale (1/5)

- Il sistema di gestione della memoria virtuale (VM) mantiene lo spazio degli indirizzi visibile ad ogni processo
  - Crea pagine di memoria virtuale su richiesta
  - Gestisce il caricamento delle pagine dal disco e la loro scrittura su disco (swapping) su richiesta

Application memory-mapping

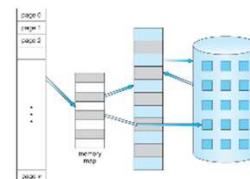


R. DONIDA LABATI – INTRODUZIONE A LINUX – LINUX TEORIA

13

## Memoria virtuale (2/5)

- Il gestore della VM mantiene due viste separate dello spazio degli indirizzi del processo
  - Una vista logica che descrive le istruzioni riguardanti il layout dello spazio degli indirizzi
    - Lo spazio degli indirizzi consiste in un insieme di regioni non sovrapposte, uno sottoinsieme di pagine allineate dello spazio degli indirizzi
  - Una vista fisica dello spazio degli indirizzi in cui ogni spazio degli indirizzi viene mappato nelle tabelle delle pagine hardware dei processi



R. DONIDA LABATI – INTRODUZIONE A LINUX – LINUX TEORIA

14

## Memoria virtuale (3/5)

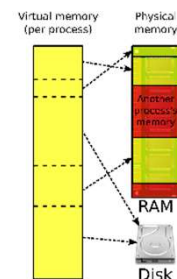
- Caratteristiche delle regioni della VM
  - Il backing store descrive da dove vengono le pagine di una regione; le regioni possono essere descritte da un file o da nessun file (demand-zero memory)
  - La reazione della regione in caso di scrittura (condivisione della pagina o copia alla scrittura)
- Il kernel crea un nuovo spazio degli indirizzi virtuale
  1. Quando un processo esegue un programma che richiama la chiamata a sistema `exec()`
  2. Alla creazione di un processo tramite la chiamata a sistema `fork()`

R. DONIDA LABATI – INTRODUZIONE A LINUX – LINUX TEORIA

15

## Memoria virtuale (4/5)

- All'esecuzione di un programma, al processo viene assegnato un nuovo spazio degli indirizzi virtuale completamente vuoto
- La lettura delle routine del programma popola lo spazio degli indirizzi con regioni di memoria virtuale



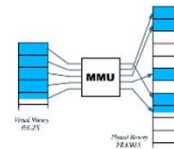
R. DONIDA LABATI – INTRODUZIONE A LINUX – LINUX TEORIA

16



## Memoria virtuale (5/5)

- La creazione di un processo tramite `fork()` implica la creazione di una copia completa dello spazio degli indirizzi *virtuali* del processo chiamante
  - Il kernel copia i descrittori VMA del padre e crea un nuovo insieme di tabelle delle pagine per il figlio
  - Le tabelle delle pagine del padre vengono direttamente copiate per il figlio, con il reference count di ogni pagina incrementato
  - Dopo una `fork`, il padre e il figlio condividono le stesse pagine di memoria fisica nei loro spazi degli indirizzi

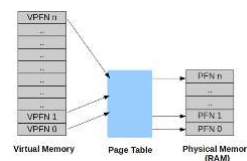


R. DONIDA LABATI – INTRODUZIONE A LINUX – LINUX TEORIA

17

## Swapping e paginazione (1/2)

- Il sistema di paginazione rialloca la memoria dalla memoria fisica al disco quando è necessaria memoria per altre attività
- Il sistema di paginazione può essere diviso in due passi
  1. L'algoritmo pageout-policy decide quali pagine scrivere su disco e quando scriverle
  2. Il meccanismo di paginazione effettua effettivamente il trasferimento e riporta le pagine nella memoria fisica quando necessario

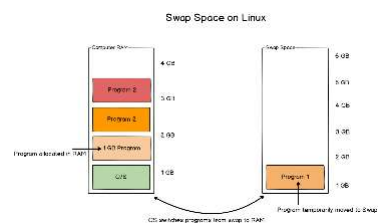


R. DONIDA LABATI – INTRODUZIONE A LINUX – LINUX TEORIA

18

## Swapping e paginazione (2/2)

- Altre caratteristiche
  - Le pagine di memoria virtuale possono essere salvate in un dispositivo di swap in normali file
  - La mappa dei bit utilizzata per tracciare i blocchi dello spazio di swap viene mantenuta nella memoria fisica
  - Viene utilizzato l'algoritmo di allocazione next-fit per provare a scrivere esecuzioni contigue



R. DONIDA LABATI – INTRODUZIONE A LINUX – LINUX TEORIA

19

## Memoria virtuale del kernel

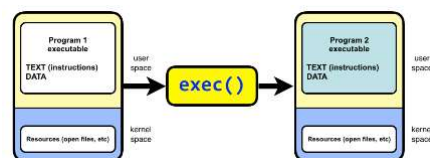
- Il kernel Linux riserva spazi di indirizzi virtuali per ogni suo processo, in modo dipendente dall'architettura
- L'area di memoria virtuale del kernel consiste in due regioni
  1. Un'area statica che contiene i riferimenti alle tabelle delle pagine per ogni pagina di memoria fisica disponibile nel sistema, fornendo quindi una traduzione da indirizzi fisici a virtuali quando viene eseguito codice kernel
  2. La restante parte della sezione riservata non è utilizzata per uno scopo specifico; i campi delle sue tabelle delle pagine possono essere modificati per puntare a qualsiasi altra area di memoria

R. DONIDA LABATI – INTRODUZIONE A LINUX – LINUX TEORIA

20

## Esecuzione e caricamento di programmi (1/2)

- Linux mantiene una tabella di funzioni per il caricamento dei programmi; fornisce ad ogni funzione l'opportunità di provare a caricare un dato file quando viene effettuata una chiamata `exec()`
- La registrazione di routine di caricamento multiple consente di supportare i formati binari ELF e a.out

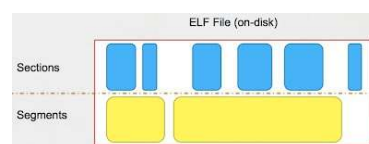


R. DONIDA LABATI – INTRODUZIONE A LINUX – LINUX TEORIA

21

## Esecuzione e caricamento di programmi (2/2)

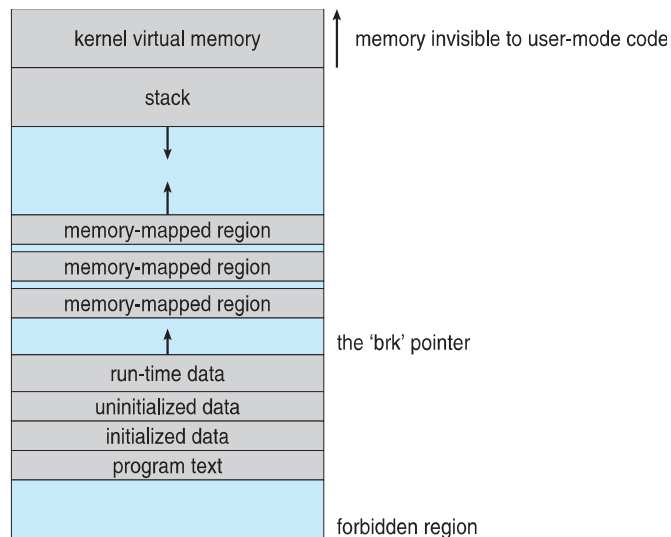
- Inizialmente, le pagine dei file binari vengono mappate nella memoria virtuale
  - Solo quando un programma prova ad accedere ad una pagina ottenendo un page fault, la pagina viene letta in una pagina fisica
- Il formato binario ELF consiste in un header seguito da diverse sezioni di pagine allineate
  - Il loader ELF legge l'header e mappa le sezioni del file in regioni separate della memoria virtuale



R. DONIDA LABATI – INTRODUZIONE A LINUX – LINUX TEORIA

22

## Schema della memoria per programmi ELF

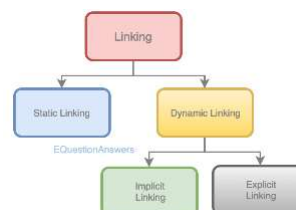


R. DONIDA LABATI – INTRODUZIONE A LINUX – LINUX TEORIA

23

## Linking statico e dinamico (1/3)

- Un programma in cui tutte le librerie necessarie sono memorizzate all'interno dei binari dell'eseguibile è linkato in modo statico alle sue librerie
  - Il principale svantaggio del linking statico consiste nel fatto che ogni programma deve contenere la copia esatta di librerie di funzioni di sistema utilizzate con frequenza

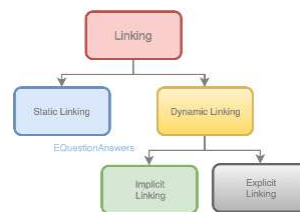


R. DONIDA LABATI – INTRODUZIONE A LINUX – LINUX TEORIA

24

## Linking statico e dinamico (2/3)

- Il linking dinamico è più efficiente in termini di memoria fisica e spazio disco perché legge le librerie di sistema solo una volta



R. DONIDA LABATI – INTRODUZIONE A LINUX – LINUX TEORIA

25

## Linking statico e dinamico (3/3)

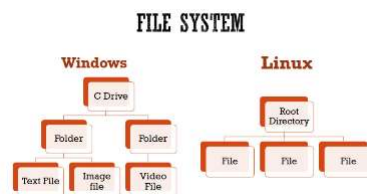
- Linux implementa il linking dinamico in modalità utente tramite specifiche librerie del linker
  - Ogni programma linkato dinamicamente contiene piccole e specifiche funzioni linkate staticamente, che vengono chiamate quando il processo inizia
  - Mappa la link library all'interno della memoria
  - La link library determina le librerie dinamiche richieste dal processo e nomina le variabili e funzioni necessarie
  - Mappa le librerie nella memoria virtuale e risolve i riferimenti ai simboli contenuti nelle librerie
  - Le librerie condivise compilate sono il position-independent code (PIC) e possono quindi essere lette ovunque

R. DONIDA LABATI – INTRODUZIONE A LINUX – LINUX TEORIA

26

## 2. File system (1/3)

- In Linux, il file system appare agli utenti come un albero di directory organizzato con una gerarchia con semantica UNIX
- Internamente, il kernel nasconde i dettagli implementativi e gestisce file system multipli ed eterogenei tramite un livello di astrazione, il virtual file system (VFS)

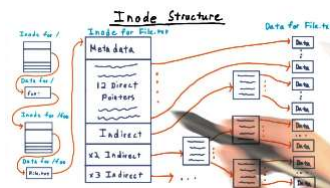


R. DONIDA LABATI – INTRODUZIONE A LINUX – LINUX TEORIA

27

## 2. File system (2/3)

- IL VFS Linux è progettato con principi di programmazione ad oggetti ed è composto da 4 componenti principali
  - La struttura *inode* dell'oggetto, che rappresenta un singolo file
  - Il *file object*, che rappresenta un file aperto
  - Il *superblock object*, che rappresenta l'intero file system
  - Il *dentry object*, che rappresenta una singola directory

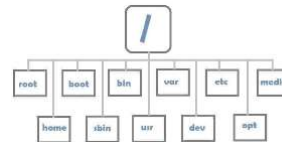


R. DONIDA LABATI – INTRODUZIONE A LINUX – LINUX TEORIA

28

## 2. File system (3/3)

- Esempi di operazioni per l'oggetto file  
(da struct file\_operations in `/usr/include/linux/fs.h`)
  - `int open( )` ← apre un file
  - `ssize_t read( )` ← legge da un file
  - `ssize_t write( )` ← scrive in un file
  - `int mmap( )` ← mappa un file in memoria

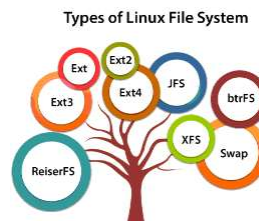


R. DONIDA LABATI – INTRODUZIONE A LINUX – LINUX TEORIA

29

## Il file system ext3 (1/3)

- ext3 è il file system standard per dischi in Linux
  - Usa un meccanismo simile a BSD Fast File System (FFS) per posizionare i blocchi di dati di un file specifico
  - Evoluzione dei file system extfs ed ext2
  - ext4 aggiunge caratteristiche aggiuntive ad ext2, come gli extents
  - File system differenti possono essere utilizzati da diverse distribuzioni Linux



R. DONIDA LABATI – INTRODUZIONE A LINUX – LINUX TEORIA

30

## Il file system ext3 (2/3)

- Le principali differenze tra ext3fs e FFS riguardano le loro politiche di allocazione del disco
  - In FFS il disco viene allocato utilizzando blocchi di 8 Kb, con i blocchi suddivisi in frammenti di 1 Kb per salvare file piccoli o blocchi parzialmente riempiti alla fine del file
  - ext3 non utilizza frammenti, ma alloca in unità più piccole
  - In ext3, la dimensione di default dei blocchi varia in base alla dimensione totale del file system (1, 2, 4 o 8 blocchi)

R. DONIDA LABATI – INTRODUZIONE A LINUX – LINUX TEORIA

31

## Il file system ext3 (3/3)

- Le principali differenze tra ext3fs e FFS riguardano le loro politiche di allocazione del disco
  - ext3 usa politiche di allocazione dei cluster progettate per posizionare blocchi di un file adiacenti a livello logico in blocchi fisici del disco adiacenti, in modo da poter eseguire richieste di I/O per diversi blocchi del disco tramite una singola operazione su un block group
  - Mantiene la mappa dei bit dei blocchi liberi in un block group, cercando i byte liberi per allocare un minimo di 8 blocchi

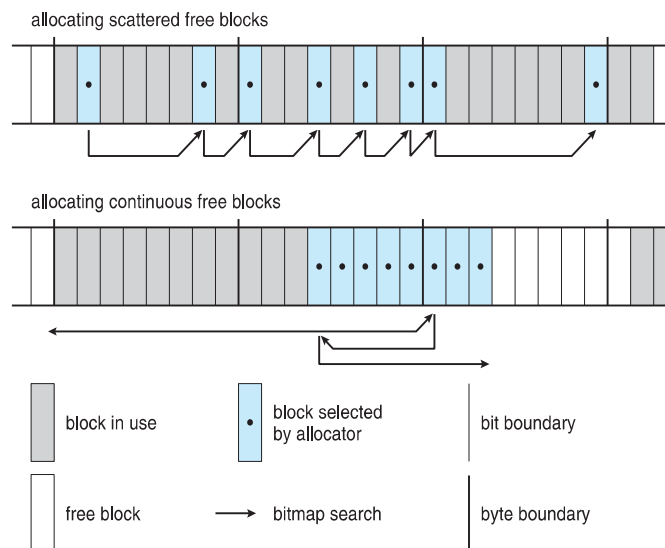
The logo for the ext3 file system, featuring the word "ext" in a grey, lowercase, sans-serif font, followed by a large, bold, orange number "3".

R. DONIDA LABATI – INTRODUZIONE A LINUX – LINUX TEORIA

32



## Politiche di allocazione dei blocchi in ext3fs

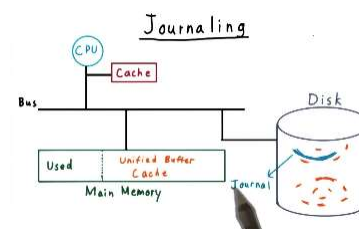


R. DONIDA LABATI – INTRODUZIONE A LINUX – LINUX TEORIA

33

## Journaling (1/2)

- ext3 implementa il journaling scrivendo le transazioni in un file di log per ogni aggiornamento del file system
  - Una volta scritto nel file di log, la transazione viene considerata committed
  - Durante il tempo, il file di log delle transazioni viene aggiornato dal file system per applicare le modifiche

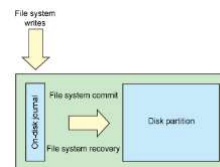


R. DONIDA LABATI – INTRODUZIONE A LINUX – LINUX TEORIA

34

## Journaling (2/2)

- In caso di crash del sistema, alcune transazioni potrebbero essere presenti nel journal, ma non essere state applicate al file system
  - Devono essere completate quando il sistema è stato recuperato
  - Non sono necessari ulteriori controlli della consistenza a seguito di un crash (molto più veloce di altri metodi)
- Migliora le performance di scrittura su hard disk trasformando l'I/O casuale in I/O sequenziale

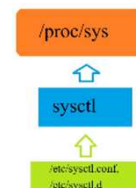


R. DONIDA LABATI – INTRODUZIONE A LINUX – LINUX TEORIA

35

## Il file system Linux proc

- File system creato e mantenuto a run-time dal Kernel di linux
  - Tenere traccia dei vari processi che stanno funzionando sulla macchina e del loro stato
  - `/proc` legge direttamente dalla memoria del sistema le sue informazioni in "tempo reale"
- Vi sono varie utility come *lscpi*, *scanpci* che ottengono i loro dati proprio leggendo `/proc` ed interpretandone le informazioni.
  - Altre utility simili sono ps e top



R. DONIDA LABATI – INTRODUZIONE A LINUX – LINUX TEORIA

36

### 3. Input e output (1/2)

- Il file system Linux orientato ai dispositivi accede al disco mediante 2 cache
  - I dati sono gestiti nella page cache, che è unificata a quella del gestore della memoria virtuale
  - I metadati sono gestiti nella buffer cache, una cache separata e indicizzata tramite il blocco disco fisico

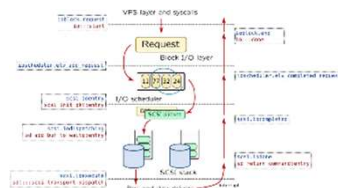


R. DONIDA LABATI – INTRODUZIONE A LINUX – LINUX TEORIA

37

### 3. Input e output (2/2)

- Linux divide i dispositivi in 3 classi
  1. Dispositivi a blocchi, che consentono un accesso casuale completamente indipendente, usando blocchi di dati di dimensione fissa
  2. Dispositivi a caratteri, che includono la maggior parte degli altri dispositivi e non necessitano di supportare funzionalità dei normali file
  3. Dispositivi di rete, che sono interfacciati tramite il sottosistema di rete del kernel

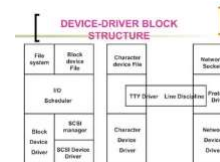


R. DONIDA LABATI – INTRODUZIONE A LINUX – LINUX TEORIA

38

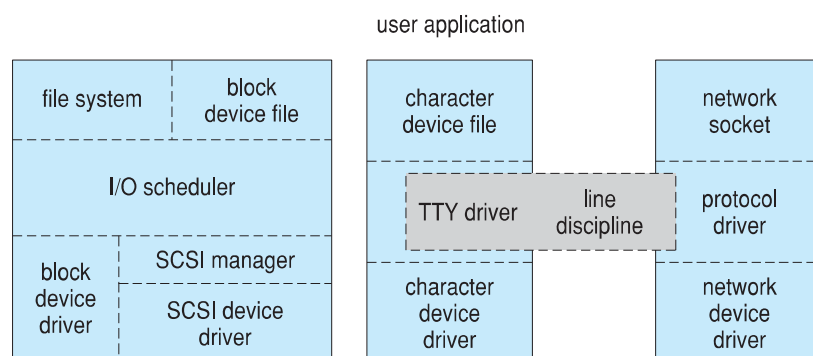
## Dispositivi a blocchi (1/3)

- Forniscono l'interfaccia principale a tutti i dispositivi a dischi del sistema
- La block buffer cache viene utilizzata per due scopi principali
  - Agisce come un insieme di buffer per l'I/O attivo
  - Serve per completare l'I/O
- Il request manager gestisce la lettura e scrittura del contenuto del buffer da e verso il dispositivo a blocchi



39

## Dispositivi a blocchi (2/3)



40

## Dispositivi a blocchi (3/3)

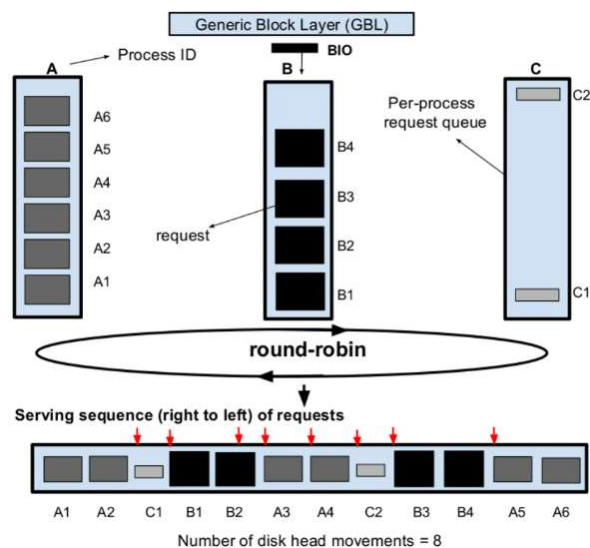
- Il kernel 2.6 ha introdotto il Completely Fair Queueing (CFQ)
  - Molto diverso dall'algoritmo dell'ascensore
  - Mantiene un insieme di liste, di default una per processo
  - Usa l'algoritmo C-SCAN, con round robin tra i processi richiedenti I/O



R. DONIDA LABATI – INTRODUZIONE A LINUX – LINUX TEORIA

41

## Completely Fair Queueing (CFQ)

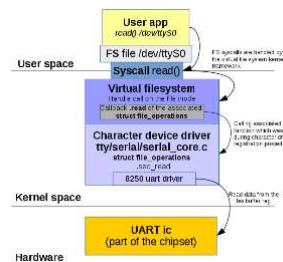


R. DONIDA LABATI – INTRODUZIONE A LINUX – LINUX TEORIA

42

## Dispositivi a caratteri (1/3)

- Un dispositivo a caratteri non offre accesso casuale a blocchi di data fissi
- Il driver di un dispositivo a caratteri deve registrare un insieme di funzioni che vengono implementate dai driver di varie operazioni di file I/O

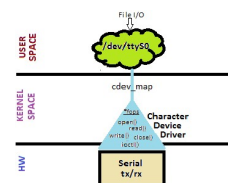


R. DONIDA LABATI – INTRODUZIONE A LINUX – LINUX TEORIA

43

## Dispositivi a caratteri (2/3)

- Il kernel effettua una preelaborazione quasi nulla di una richiesta di lettura o scrittura da parte di un dispositivo a caratteri, passando semplicemente la richiesta al dispositivo
- L'eccezione principale a questa regola consiste in un sottoinsieme speciale di driver per dispositivi a caratteri che implementano i dispositivi terminale, per cui il kernel mantiene un'interfaccia standard



R. DONIDA LABATI – INTRODUZIONE A LINUX – LINUX TEORIA

44

### Dispositivi a caratteri (3/3)

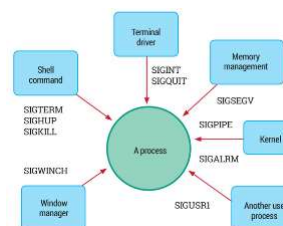
- *Line discipline* è un interprete delle informazioni ottenute dal dispositivo terminale
  - La disciplina più comune la *tty discipline*, che unisce il flusso di dati del terminale negli stream standard input e standard output del processo utente, consentendo ai processi di comunicare direttamente con il terminale dell'utente
  - Numerosi processi potrebbero essere eseguiti contemporaneamente, la *tty line discipline* è responsabile dell'aggregazione e disaggregazione degli ingressi e uscite del terminale per i vari processi collegati ad esso, anche per la sospensione o il risveglio dei processi da parte dell'utente

R. DONIDA LABATI – INTRODUZIONE A LINUX – LINUX TEORIA

45

### 4. Comunicazione tra processi (1/2)

- In modo simile a UNIX, Linux informa i processi di un evento tramite segnali
- Esiste un numero limitato di segnali ed essi non trasmettono informazione addizionale, ma i processi possono solo ricevere l'arrivo di un segnale

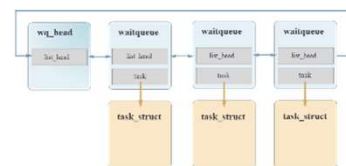


R. DONIDA LABATI – INTRODUZIONE A LINUX – LINUX TEORIA

46

## 4. Comunicazione tra processi (2/2)

- Il kernel Linux non usa segnali per comunicare con processi eseguiti in modalità kernel, ma effettua la comunicazione utilizzando strutture *wait\_queue*
- Implementa anche i semafori System V Unix
  - Un processo può attendere per un segnale o un semaforo
  - I semafori offrono maggiore scalabilità
  - Le operazioni per semafori multipli possono essere atomiche

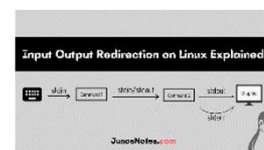


R. DONIDA LABATI – INTRODUZIONE A LINUX – LINUX TEORIA

47

## Scambio di dati tra processi

- La pipe è il meccanismo che consente al processo figlio di ereditare il canale di comunicazione dal processo padre, consentendo la lettura dei dati alla fine della pipe
- La memoria condivisa offre un metodo di comunicazione estremamente veloce: un dato scritto da un processo in memoria condivisa può essere immediatamente letto da un altro processo che mappa la stessa regione nel suo spazio di indirizzi



R. DONIDA LABATI – INTRODUZIONE A LINUX – LINUX TEORIA

48



## 5. Gestione della rete (1/2)

- La rete è una delle funzionalità chiave di Linux
  - Supporta i protocolli internet standard per la comunicazione UNIX to UNIX
  - Implementa anche protocolli nativi di sistemi non-UNIX, come quelli usati per reti di PC (ad esempio Appletalk e IPX)
- Internamente, la gestione della rete del kernel Linux è implementata tramite tre livelli software
  - L'interfaccia socket
  - I driver dei protocolli
  - I driver dei dispositivi di rete



R. DONIDA LABATI – INTRODUZIONE A LINUX – LINUX TEORIA

49

## 5. Gestione della rete (2/2)

- Il gruppo di protocolli più rilevanti del sistema di rete Linux è l'Internet Protocol suite
  - Implementa il routing tra i diversi host, ovunque nella rete
  - Sopra al protocollo di routing, implementa i protocolli UDP, TCP e ICMP
- I pacchetti passano anche dal firewall management per filtraggi basati sulle regole delle firewall chains



R. DONIDA LABATI – INTRODUZIONE A LINUX – LINUX TEORIA

50

## 6. Sicurezza (1/2)

- Linux utilizza il sistema Pluggable Authentication Modules (PAM)
- PAM è basato su librerie condivise che possono essere utilizzate da ogni componente del sistema che necessita di autenticare gli utenti

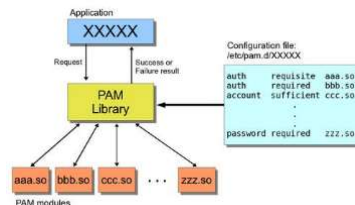


R. DONIDA LABATI – INTRODUZIONE A LINUX – LINUX TEORIA

51

## 6. Sicurezza (2/2)

- Il controllo degli accessi nei sistemi UNIX, incluso Linux, viene effettuato utilizzando identificatori numerici univoci (*uid* e *gid*)
- Il controllo degli accessi viene effettuato assegnando oggetti a una protection mask, che specifica le modalità di accesso (*r*, *w*, *x*) che devono essere garantite a utente, gruppo, o altri



R. DONIDA LABATI – INTRODUZIONE A LINUX – LINUX TEORIA

52

## In sintesi

1. Gestione della memoria
2. File system
3. Input e output
4. Comunicazione tra processi
5. Gestione della rete
6. Sicurezza

