

LPIC-1

Versione 5.0
Italiano

101

Table of Contents

ARGOMENTO 101: L'ARCHITETTURA DI SISTEMA	1
101.1 Determinare e Configurare le Impostazioni dell'Hardware	2
101.1 Lezione 1	3
Introduzione	3
L'Attivazione delle Periferiche	4
Il Controllo delle Periferiche in Linux	4
I File Informativi e i File dei Dispositivi	11
I Dispositivi di Archiviazione	12
Esercizi Guidati	14
Esercizi Esplorativi	15
Sommario	16
Risposte agli Esercizi Guidati	17
Risposte agli Esercizi Esplorativi	18
101.2 Avviare il sistema	19
101.2 Lezione 1	21
Introduzione	21
BIOS o UEFI	22
Il Bootloader	23
Inizializzazione del Sistema	25
Ispezionare l'inizializzazione	26
Esercizi Guidati	30
Esercizi Esplorativi	31
Sommario	32
Risposte agli Esercizi Guidati	33
Risposte agli Esercizi Guidati	34
101.3 Modificare runlevel / target di avvio e spegnere o riavviare il sistema	35
101.3 Lezione 1	37
Introduzione	37
SysVinit	38
systemd	41
Upstart	44
Spegnimento e Riavvio	45
Esercizi Guidati	48
Esercizi Esplorativi	49
Sommario	50
Risposte agli Esercizi Guidati	51
Risposte agli Esercizi Guidati	52
ARGOMENTO 102: INSTALLAZIONE DI LINUX E GESTIONE DEI PACCHETTI	53

102.1 Progettare il layout del disco rigido	54
102.1 Lezione 1	55
Introduzione	55
Punti di Montaggio	56
Tenere le Cose Separate	57
Lo Swap	59
LVM	60
Esercizi Guidati	62
Esercizi Esplorativi	63
Sommaro	64
Risposte agli Esercizi Guidati	65
Risposte agli Esercizi Guidati	66
102.2 Installare un boot manager	67
102.2 Lezione 1	68
Introduzione	68
GRUB Legacy e GRUB 2	69
Dove è il Bootloader?	69
La Partizione /boot	70
GRUB 2	71
GRUB Legacy	78
Esercizi Guidati	82
Esercizi Esplorativi	83
Sommaro	84
Risposte agli Esercizi Guidati	85
Risposte agli Esercizi Guidati	86
102.3 Gestire le librerie condivise	88
102.3 Lezione 1	89
Introduzione	89
Le Librerie Condivise	89
Convenzioni di Denominazione dei File Oggetto Condivisi	90
Configurazione dei Percorsi delle Librerie Condivise	91
Ricerca delle Dipendenze di un Eseguibile Specifico	94
Esercizi Guidati	96
Esercizi Esplorativi	97
Sommaro	98
Risposte agli Esercizi Guidati	100
Risposte agli Esercizi Esplorativi	101
102.4 Utilizzare la gestione dei pacchetti Debian	102
102.4 Lezione 1	103
Introduzione	103

Lo Strumento di Gestione Pacchetti in Debian (dpkg)	104
Advanced Package Tool (apt)	108
Esercizi Guidati	118
Esercizi Esplorativi	119
Sommario	120
Risposte agli Esercizi Guidati	122
Risposte agli Esercizi Esplorativi	123
102.5 Utilizzare la gestione dei pacchetti RPM e YUM	125
102.5 Lezione 1	126
Introduzione	126
Il Gestore di Pacchetti RPM (rpm)	127
YellowDog Updater Modified (YUM)	132
DNF	137
Zypper	139
Esercizi Guidati	146
Esercizi Esplorativi	147
Sommario	148
Risposte agli Esercizi Guidati	149
Risposte agli Esercizi Esplorativi	150
102.6 Linux come sistema virtualizzato	151
102.6 Lezione 1	152
Introduzione	152
Concetti di Virtualizzazione	152
Tipi di Macchine Virtuali	153
Lavorare con i Template di Macchine Virtuali	160
Distribuire Macchine Virtuali in Cloud	162
I Container	164
Esercizi Guidati	166
Esercizi Esplorativi	167
Sommario	168
Risposte agli Esercizi Guidati	169
Risposte agli Esercizi Esplorativi	170
ARGOMENTO 103: GNU E UNIX COMMANDS	172
103.1 Lavorare con la Command Line	173
103.1 Lezione 1	175
Introduzione	175
Ottenere Informazioni di Sistema	175
Ottenere Informazioni sui Comandi	176
Utilizzare l'History dei Comandi	179
Esercizi Guidati	181

Esercizi Esplorativi	182
Sommario	183
Risposte agli Esercizi Guidati	184
Risposte agli Esercizi Esplorativi	185
103.1 Lezione 2	186
Introduzione	186
Trovare le variabili di ambiente	186
Creare Nuove Variabili d'Ambiente	187
Cancellare una Variabile d'Ambiente	188
Usare il Virgolettato per Escludere i Caratteri Speciali	189
Esercizi Guidati	191
Esercizi Esplorativi	192
Sommario	193
Risposte agli Esercizi Guidati	194
Risposte agli Esercizi Esplorativi	195
103.2 Elaborare flussi di testo utilizzando i filtri	196
103.2 Lezione 1	198
Introduzione	198
Una Rapida Rassegna su Reindirizzamenti e Pipe	198
Elaborare Flussi di Testo	201
Esercizi Guidati	212
Esercizi Esplorativi	214
Sommario	216
Risposte agli Esercizi Guidati	218
Risposte agli Esercizi Esplorativi	223
103.3 Eseguire la gestione di base dei file	229
103.3 Lezione 1	231
Introduzione	231
Manipolazione dei File	232
Creare e Rimuovere Directory	237
Manipolazione Ricorsiva di File e Directory	238
File Globbing e Wildcards	241
Tipi di Wildcard	242
Esercizi Guidati	246
Esercizi Esplorativi	248
Sommario	249
Risposte agli Esercizi Guidati	250
Risposte agli Esercizi Esplorativi	252
103.3 Lezione 2	254
Introduzione	254

Come Ricercare File	254
Archiviazione di File	258
Esercizi Guidati	264
Esercizi Esplorativi	265
Sommaro	266
Risposte agli Esercizi Guidati	267
Risposte agli Esercizi Esplorativi	268
103.4 Utilizzare flussi, pipe e reindirizzamenti	270
103.4 Lezione 1	271
Introduzione	271
Reindirizzamenti	272
Here Document e Here String	275
Esercizi Guidati	277
Esercizi Esplorativi	278
Sommaro	279
Risposte agli Esercizi Guidati	280
Risposte agli Esercizi Esplorativi	281
103.4 Lezione 2	282
Introduzione	282
Le Pipe	282
Sostituzione dei Comandi	284
Esercizi Guidati	287
Esercizi Esplorativi	288
Sommaro	289
Risposte agli Esercizi Guidati	290
Risposte agli Esercizi Esplorativi	292
103.5 Creare, controllare e terminare i processi	293
103.5 Lezione 1	295
Introduzione	295
Controllare i Job	295
Controllare i Processi	300
Esercizi Guidati	311
Esercizi Esplorativi	313
Sommaro	315
Risposte agli Esercizi Guidati	317
Risposte agli Esercizi Esplorativi	320
103.5 Lezione 2	323
Introduzione	323
Caratteristiche dei Terminali Multiplexer	323
GNU Screen	324

tmux	331
Esercizi Guidati	340
Esercizi Esplorativi	344
Sommario	346
Risposte agli Esercizi Guidati	347
Risposte agli Esercizi Esplorativi	352
103.6 Modificare le priorità di esecuzione del processo	354
103.6 Lezione 1	355
Introduzione	355
Il Linux Scheduler	356
Leggere le Priorità	357
La Niceness del processo	358
Esercizi Guidati	360
Esercizi Esplorativi	362
Sommario	363
Risposte agli Esercizi Guidati	364
Risposte agli Esercizi Esplorativi	366
103.7 Cercare file di testo utilizzando espressioni regolari	367
103.7 Lezione 1	368
Introduzione	368
Espressioni tra Parentesi Quadre	369
Quantificatori	371
Limiti	371
Rami e Riferimenti all'Indietro	372
Ricerca con Espressioni Regolari	372
Esercizi Guidati	374
Esercizi Esplorativi	375
Sommario	376
Risposte agli Esercizi Guidati	377
Risposte agli Esercizi Esplorativi	378
103.7 Lezione 2	379
Introduzione	379
Il "Cercatore" di Pattern: grep	379
L'Editor di Stream: sed	383
Combinare grep e sed	387
Esercizi Guidati	391
Esercizi Esplorativi	392
Sommario	394
Risposte agli Esercizi Guidati	395
Risposte agli Esercizi Esplorativi	396

103.8 Modifica base di un file	398
103.8 Lezione 1	399
Introduzione	399
Insert Mode	400
Normal Mode	400
Comandi dei Due Punti	403
Editor Alternativi	404
Esercizi Guidati	406
Esercizi Esplorativi	407
Sommario	408
Risposte agli Esercizi Guidati	409
Risposte agli Esercizi Esplorativi	410
ARGOMENTO 104: DISPOSITIVI, IL FILE SYSTEM LINUX, IL FILESYSTEM HIERARCHY	
STANDARD	411
104.1 Creare partizioni e filesystem	412
104.1 Lezione 1	413
Introduzione	413
Comprendere MBR e GPT	414
Creare un File System	421
Gestire le Partizioni con GNU Parted	432
Creare Partizioni di Swap	439
Esercizi Guidati	441
Esercizi Esplorativi	442
Sommario	444
Risposte agli Esercizi Guidati	445
Risposte agli Esercizi Esplorativi	446
104.2 Mantenere l'integrità dei filesystem	448
104.2 Lezione 1	449
Introduzione	449
Controllare l'Utilizzo dei Dischi	450
Controllo dello Spazio Libero	452
Manutenzione dei filesystem ext2, ext3 e ext4	456
Esercizi Guidati	464
Esercizi Esplorativi	465
Sommario	466
Risposte agli Esercizi Guidati	467
Risposte agli Esercizi Esplorativi	469
104.3 Verificare il montaggio e lo smontaggio dei filesystem	471
104.3 Lezione 1	472
Introduzione	472

Montare e Smontare i Filesystem	472
Montare un Filesystem all'Avvio	476
Utilizzare UUID ed Etichette	478
Montare Dischi con Systemd	480
Esercizi Guidati	484
Esercizi Esplorativi	485
Sommario	486
Risposte agli Esercizi Guidati	487
Risposte agli Esercizi Esplorativi	489
104.5 Gestire le autorizzazioni e la proprietà dei file	491
104.5 Lezione 1	492
Introduzione	492
Richiedere Informazioni Riguardo File e Directory	492
E Riguardo le Directory?	494
Visualizzare i File Nascosti	494
Comprendere i Tipi di File	495
Comprendere i Permessi	496
Modificare i Permessi sui File	498
Modificare la Proprietà di un File	501
Interrogare i Gruppi	502
Permessi Default	502
Permessi Speciali	505
Esercizi Guidati	508
Esercizi Esplorativi	510
Sommario	511
Risposte agli Esercizi Guidati	512
Risposte agli Esercizi Esplorativi	515
104.6 Creare e modificare collegamenti hard e soft	518
104.6 Lezione 1	519
Introduzione	519
Comprendere i Link	519
Esercizi Guidati	524
Esercizi Esplorativi	525
Sommario	528
Risposte agli Esercizi Guidati	529
Risposte agli Esercizi Esplorativi	530
104.7 Trovare i file di sistema e collocarli nella posizione corretta	534
104.7 Lezione 1	535
Introduzione	535
Il Filesystem Hierarchy Standard	535

Ricerca di File	538
Esercizi Guidati	547
Esercizi Esplorativi	548
Sommario	549
Risposte agli Esercizi Guidati	550
Risposte agli Esercizi Esplorativi	552
Imprint	554



**Linux
Professional
Institute**

Argomento 101: L'Architettura di Sistema



101.1 Determinare e Configurare le Impostazioni dell'Hardware

Obiettivi LPI di riferimento

LPIC-1 v5, Exam 101, Objective 101.1

Peso

2

Aree di Conoscenza Chiave

- Abilitare e disabilitare le periferiche integrate.
- Distinguere tra i vari tipi di dispositivi di archiviazione di massa.
- Determinare le risorse hardware per i dispositivi.
- Strumenti e utilità per elencare varie informazioni sull'hardware (per esempio `lsusb`, `lspci`, ecc.).
- Strumenti e utilità per manipolare i dispositivi USB.
- Comprensione concettuale di `sysfs`, `udev` e `dbus`.

Di seguito è riportato un elenco parziale dei file, dei termini e dei comandi utilizzati

- `/sys/`
- `/proc/`
- `/dev/`
- `modprobe`
- `lsmod`
- `lspci`
- `lsusb`



101.1 Lezione 1

Certificazione:	LPIC-1
Versione:	5.0
Argomento:	101 L'Architettura di Sistema
Obiettivo:	101.1 Determinare e Configurare le Impostazioni dell'Hardware
Lezione:	1 di 1

Introduzione

Sin dagli albori dell'informatica, i produttori di personal computer integrano una gran varietà di parti hardware nei loro sistemi, che a loro volta devono essere supportate dal sistema operativo. Questa eterogeneità rischia di essere troppo impegnativa dal punto di vista dello sviluppatore del sistema stesso, a meno che l'industria non stabilisca standard per i set di istruzioni e la comunicazione dei dispositivi. Analogamente al livello di astrazione standardizzato fornito dal sistema operativo a un'applicazione, questi standard semplificano la scrittura e la manutenzione di un sistema operativo non legato a un modello di hardware specifico. Tuttavia, la complessità dell'hardware richiede a volte aggiustamenti su come le risorse debbano essere interfacciate con il sistema operativo, in modo che possa essere installato e funzionare correttamente.

Alcune di queste regolazioni possono essere eseguite anche senza un sistema operativo installato. La maggior parte delle macchine offre un'utilità di configurazione che può essere eseguita all'accensione della macchina stessa. Fino alla metà degli anni 2000, l'utilità di configurazione era implementata nel BIOS (*Basic Input / Output System*), lo standard per il firmware contenente le routine di configurazione di base presente nelle schede madri x86. Dalla fine del primo decennio degli anni 2000, le macchine basate sull'architettura x86 hanno iniziato a sostituire il BIOS con

una nuova implementazione chiamata UEFI (*Unified Extensible Firmware Interface*), che ha funzionalità più sofisticate per l'identificazione, il test, la configurazione e gli aggiornamenti del firmware. Nonostante la modifica, non è raro chiamare ancora l'utilità di configurazione con il nome di BIOS, poiché entrambe le implementazioni soddisfano lo stesso scopo di base.

NOTE

Ulteriori dettagli sulle somiglianze e le differenze tra BIOS e UEFI saranno trattati in una lezione successiva.

L'Attivazione delle Periferiche

L'utilità di configurazione del sistema viene visualizzata dopo aver premuto un tasto specifico all'accensione del computer. Quale tasto premere varia da produttore a produttore, ma di solito è **Canc** o uno dei tasti funzione, come **F2** o **F12**. La combinazione di tasti da utilizzare viene spesso visualizzata nella schermata di accensione.

Nell'utilità di configurazione del BIOS è possibile abilitare e disabilitare le periferiche integrate, attivare la protezione degli errori di base e modificare le impostazioni hardware come IRQ (Interrupt ReQuest) e DMA (Direct Memory Access). La modifica di queste impostazioni è raramente effettuata sulle macchine moderne, ma potrebbe essere necessario apportare modifiche per risolvere problemi specifici. Esistono tecnologie RAM, per esempio, compatibili con velocità di trasferimento dati più elevate rispetto ai valori predefiniti, quindi si consiglia di modificarlo con i valori specificati dal produttore. Alcune CPU offrono funzionalità che potrebbero non essere necessarie per l'installazione specifica e che possono essere disattivate. Le funzioni disabilitate riducono il consumo di energia e possono aumentare la protezioni del sistema, disabilitando per esempio funzioni della CPU contenenti bug noti.

Se la macchina è dotata di molti dispositivi di archiviazione, è importante definire quale abbia il bootloader corretto e dovrebbe essere la prima voce nell'ordine di avvio. Il sistema operativo potrebbe non caricarsi se il dispositivo errato viene visualizzato per primo nell'elenco dell'avvio del BIOS.

Il Controllo delle Periferiche in Linux

Una volta identificati correttamente i dispositivi, spetta al sistema operativo associare i componenti software corrispondenti richiesti da essi. Quando una funzionalità hardware non opera come previsto, è importante identificare dove si stia verificando esattamente il problema. Quando un componente hardware non viene rilevato dal sistema operativo, è molto probabile che esso — o la porta a cui è collegato — sia difettoso. Quando il componente hardware viene rilevata correttamente, ma continua a non funzionare, potrebbe esserci un problema sul fronte sistema operativo. Pertanto, uno dei primi passi, quando si affrontano i problemi relativi all'hardware, è

verificare se il sistema operativo stia rilevando correttamente il dispositivo. Esistono due modi di base per identificare le risorse hardware su un sistema Linux: usare comandi specifici o leggere file specifici all'interno di filesystem speciali.

I Comandi di Ispezione

I due comandi essenziali per identificare i dispositivi collegati in un sistema Linux sono:

`lspci`

Mostra tutti i dispositivi attualmente collegati al bus PCI (*Peripheral Component Interconnect*). I dispositivi PCI possono essere un componente collegato alla scheda madre, come un controller del disco, o una scheda di espansione inserita in uno slot PCI, come una scheda grafica esterna.

`lsusb`

Elenca i dispositivi USB (*Universal Serial Bus*) attualmente collegati alla macchina. Sebbene esistano dispositivi USB per quasi tutti gli scopi immaginabili, l'interfaccia USB è ampiamente utilizzata per collegare dispositivi di input — tastiere, dispositivi di puntamento — e supporti di archiviazione rimovibili.

L'output dei comandi `lspci` e `lsusb` consiste in un elenco di tutti i dispositivi PCI e USB identificati dal sistema operativo. Tuttavia, il dispositivo potrebbe non essere ancora completamente operativo, poiché ogni parte hardware richiede un componente software per controllare il dispositivo corrispondente. Questo componente software è chiamato *kernel module* e può far parte del kernel Linux ufficiale o essere aggiunto separatamente da altre fonti.

I moduli del kernel Linux relativi ai dispositivi hardware sono anche chiamati *drivers*, come in altri sistemi operativi. I driver per Linux, tuttavia, non sono sempre forniti dal produttore del dispositivo. Mentre alcuni produttori forniscono i propri driver binari da installare separatamente, molti altri sono scritti da sviluppatori indipendenti. Storicamente, i componenti che funzionano su Windows, per esempio, potrebbero non avere un modulo kernel controparte per Linux. Al giorno d'oggi, i sistemi operativi basati su Linux hanno un forte supporto hardware e la maggior parte dei dispositivi funziona senza sforzo.

I comandi direttamente correlati all'hardware richiedono spesso i privilegi di root per essere eseguiti o mostreranno informazioni limitate se eseguiti da un utente normale, quindi potrebbe essere necessario effettuare il login come root o eseguire il comando con `sudo`.

Il seguente output del comando `lspci`, per esempio, mostra alcuni dispositivi identificati:

```
$ lspci
01:00.0 VGA compatible controller: NVIDIA Corporation GM107 [GeForce GTX 750 Ti] (rev a2)
```

```
04:02.0 Network controller: Ralink corp. RT2561/RT61 802.11g PCI
04:04.0 Multimedia audio controller: VIA Technologies Inc. ICE1712 [Envy24] PCI Multi-
Channel I/O Controller (rev 02)
04:0b.0 FireWire (IEEE 1394): LSI Corporation FW322/323 [TrueFire] 1394a Controller (rev 70)
```

L'output di tali comandi può essere lungo decine di righe, quindi gli esempi precedenti e successivi contengono solo le sezioni di interesse. I numeri esadecimali all'inizio di ogni riga sono gli indirizzi univoci del corrispondente dispositivo PCI. Il comando `lspci` mostra maggiori dettagli su un dispositivo specifico se il suo indirizzo è fornito con l'opzione `-s`, accompagnato dall'opzione `-v`

```
$ lspci -s 04:02.0 -v
04:02.0 Network controller: Ralink corp. RT2561/RT61 802.11g PCI
    Subsystem: Linksys WMP54G v4.1
    Flags: bus master, slow devsel, latency 32, IRQ 21
    Memory at e3100000 (32-bit, non-prefetchable) [size=32K]
    Capabilities: [40] Power Management version 2
    kernel driver in use: rt61pci
```

L'output ora mostra molti più dettagli del dispositivo sull'indirizzo `04: 02.0`. È un controller di rete, il cui nome interno è `Ralink corp. RT2561 / RT61 802.11g PCI`. `Subsystem` è associato alla marca e al modello del dispositivo `Linksys WMP54G v4.1` e può essere utile a fini diagnostici.

Il modulo del kernel può essere identificato nella riga `kernel driver in use`, che mostra il modulo `rt61pci`. Da tutte le informazioni raccolte, è corretto presumere che:

1. Il device è stato identificato.
2. Un modulo del kernel corrispondente è stato caricato.
3. Il device dovrebbe essere pronto all'uso.

Un altro modo per verificare quale modulo del kernel è in uso per il dispositivo specificato è fornito dall'opzione `-k`, disponibile nelle versioni più recenti di `lspci`:

```
$ lspci -s 01:00.0 -k
01:00.0 VGA compatible controller: NVIDIA Corporation GM107 [GeForce GTX 750 Ti] (rev a2)
    kernel driver in use: nvidia
    kernel modules: nouveau, nvidia_drm, nvidia
```

Per il dispositivo prescelto, una scheda NVIDIA GPU, `lspci` dice che il modulo in uso è chiamato `nvidia`, alla riga `kernel driver in use: nvidia` e tutti i corrispondenti moduli del kernel

sono elencati nella riga `kernel modules`: `nouveau`, `nvidia_drm`, `nvidia`.

Il comando `lsusb` è simile a `lspci`, ma elenca esclusivamente le informazioni USB:

```
$ lsusb
```

```
Bus 001 Device 029: ID 1781:0c9f Multiple Vendors USBtiny
Bus 001 Device 028: ID 093a:2521 Pixart Imaging, Inc. Optical Mouse
Bus 001 Device 020: ID 1131:1001 Integrated System Solution Corp. KY-BT100 Bluetooth Adapter
Bus 001 Device 011: ID 04f2:0402 Chicony Electronics Co., Ltd Genius LuxeMate i200 Keyboard
Bus 001 Device 007: ID 0424:7800 Standard Microsystems Corp.
Bus 001 Device 003: ID 0424:2514 Standard Microsystems Corp. USB 2.0 Hub
Bus 001 Device 002: ID 0424:2514 Standard Microsystems Corp. USB 2.0 Hub
Bus 001 Device 001: ID 1d6b:0002 Linux Foundation 2.0 root hub
```

Il comando `lsusb` mostra i canali USB disponibili e i dispositivi ad essi collegati. Come con `lspci`, l'opzione `-v` mostra un output più dettagliato. Un dispositivo specifico può essere selezionato per l'ispezione fornendo il suo ID all'opzione `-d`:

```
$ lsusb -v -d 1781:0c9f
```

```
Bus 001 Device 029: ID 1781:0c9f Multiple Vendors USBtiny
Device Descriptor:
  bLength                18
  bDescriptorType         1
  bcdUSB                 1.01
  bDeviceClass           255 Vendor Specific Class
  bDeviceSubClass         0
  bDeviceProtocol         0
  bMaxPacketSize0         8
  idVendor               0x1781 Multiple Vendors
  idProduct              0x0c9f USBtiny
  bcdDevice              1.04
  iManufacturer          0
  iProduct               2 USBtiny
  iSerial                0
  bNumConfigurations     1
```

Con l'opzione `-t`, il comando `lsusb` mostra le attuali mappature dei dispositivi USB come una struttura gerarchica ad albero:

```
$ lsusb -t
```

```
/: Bus 01.Port 1: Dev 1, Class=root_hub, Driver=dwc_otg/lp, 480M
```

```

|__ Port 1: Dev 2, If 0, Class=Hub, Driver=hub/4p, 480M
|__ Port 1: Dev 3, If 0, Class=Hub, Driver=hub/3p, 480M
|__ Port 2: Dev 11, If 1, Class=Human Interface Device, Driver=usbhid, 1.5M
|__ Port 2: Dev 11, If 0, Class=Human Interface Device, Driver=usbhid, 1.5M
|__ Port 3: Dev 20, If 0, Class=Wireless, Driver=btusb, 12M
|__ Port 3: Dev 20, If 1, Class=Wireless, Driver=btusb, 12M
|__ Port 3: Dev 20, If 2, Class=Application Specific Interface, Driver=, 12M
|__ Port 1: Dev 7, If 0, Class=Vendor Specific Class, Driver=lan78xx, 480M
|__ Port 2: Dev 28, If 0, Class=Human Interface Device, Driver=usbhid, 1.5M
|__ Port 3: Dev 29, If 0, Class=Vendor Specific Class, Driver=, 1.5M

```

È possibile che non tutti i dispositivi abbiano i moduli corrispondenti associati. La comunicazione con alcuni dispositivi può essere gestita direttamente dall'applicazione, senza l'intermediazione fornita da un modulo. Tuttavia, ci sono informazioni significative nell'output fornito da `lsusb -t`. Quando esiste un modulo corrispondente, il suo nome appare alla fine della riga del dispositivo, come in `Driver=btusb`. Il dispositivo `Class` identifica la categoria generale, come per esempio `Human Interface Device`, `Wireless`, `Vendor Specific Class`, `Mass Storage`. Per verificare quale dispositivo stia utilizzando il modulo `btusb`, presente nell'elenco precedente, entrambi i numeri di `Bus` e `Dev` dovrebbero essere dati all'opzione `-s` del comando `lsusb`:

```
$ lsusb -s 01:20
```

```
Bus 001 Device 020: ID 1131:1001 Integrated System Solution Corp. KY-BT100 Bluetooth Adapter
```

È comune avere un ampio set di moduli kernel caricati, in qualsiasi momento, in un sistema Linux standard. Il modo preferibile per interagire con loro è usare i comandi forniti dal pacchetto `kmod`, che è un insieme di strumenti per gestire le attività comuni con i moduli del kernel Linux quali inserire, rimuovere, elencare, controllare le proprietà, risolvere dipendenze e alias. Il comando `lsmod`, per esempio, mostra tutti i moduli attualmente caricati:

```
$ lsmod
```

Module	Size	Used by
kvm_intel	138528	0
kvm	421021	1 kvm_intel
iTCO_wdt	13480	0
iTCO_vendor_support	13419	1 iTCO_wdt
snd_usb_audio	149112	2
snd_hda_codec_realtek	51465	1
snd_ice1712	75006	3
snd_hda_intel	44075	7
arc4	12608	2
snd_cs8427	13978	1 snd_ice1712

```

snd_i2c                13828  2 snd_ice1712,snd_cs8427
snd_ice17xx_ak4xxx     13128  1 snd_ice1712
snd_ak4xxx_adda        18487  2 snd_ice1712,snd_ice17xx_ak4xxx
microcode              23527  0
snd_usbmidi_lib        24845  1 snd_usb_audio
gspca_pac7302          17481  0
gspca_main             36226  1 gspca_pac7302
videodev               132348 2 gspca_main,gspca_pac7302
rt61pci                32326  0
rt2x00pci              13083  1 rt61pci
media                  20840  1 videodev
rt2x00mmio             13322  1 rt61pci
hid_dr                 12776  0
snd_mpu401_uart        13992  1 snd_ice1712
rt2x00lib              67108  3 rt61pci,rt2x00pci,rt2x00mmio
snd_rawmidi            29394  2 snd_usbmidi_lib,snd_mpu401_uart

```

L'output del comando `lsmod` è diviso in tre colonne:

Module

Nome del modulo.

Size

Quantità di RAM occupata dal modulo, in byte.

Used by

Moduli dipendenti.

Alcuni moduli richiedono altri moduli per funzionare correttamente, come nel caso dei moduli per dispositivi audio:

```

$ lsmod | fgrep -i snd_hda_intel
snd_hda_intel          42658  5
snd_hda_codec          155748 3 snd_hda_codec_hdmi,snd_hda_codec_via,snd_hda_intel
snd_pcm                81999 3 snd_hda_codec_hdmi,snd_hda_codec,snd_hda_intel
snd_page_alloc         13852  2 snd_pcm,snd_hda_intel
snd                    59132  19
snd_hwdep,snd_timer,snd_hda_codec_hdmi,snd_hda_codec_via,snd_pcm,snd_seq,snd_hda_codec,snd_h
da_intel,snd_seq_device

```

La terza colonna, `Used by`, mostra i moduli che richiedono il modulo nella prima colonna per funzionare correttamente. Molti moduli dell'architettura audio di Linux, preceduti da `snd`, sono

interdipendenti. Quando si cercano problemi durante la diagnostica del sistema, può essere utile scaricare i moduli specifici attualmente caricati. Il comando `modprobe` può essere usato sia per caricare sia per scaricare i moduli del kernel: per scaricare un modulo e i relativi moduli, purché non vengano utilizzati da un processo in esecuzione, dovrebbe essere usato il comando `modprobe -r`. Per esempio, per scaricare il modulo `snd-hda-intel` (il modulo per un dispositivo audio Intel HDA) e altri moduli relativi al sistema audio:

```
# modprobe -r snd-hda-intel
```

Oltre a caricare e scaricare i moduli del kernel mentre il sistema è in esecuzione, è possibile modificare i parametri del modulo durante il caricamento del kernel, in maniera non molto diversa dal passare opzioni ai comandi. Ogni modulo accetta parametri specifici, ma la maggior parte delle volte sono consigliati i valori predefiniti e non sono necessari parametri aggiuntivi. Tuttavia, in alcuni casi è necessario utilizzare i parametri per modificare il comportamento di un modulo affinché funzioni come previsto.

Usando il nome del modulo come unico argomento, il comando `modinfo` mostra una descrizione, il file, l'autore, la licenza, l'identificazione, le dipendenze e i parametri disponibili per il modulo dato. I parametri personalizzati per un modulo possono essere resi persistenti includendoli nel file `/etc/modprobe.conf` o in singoli file con l'estensione `.conf` nella directory `/etc/modprobe.d/`. L'opzione `-p` farà in modo che il comando `modinfo` visualizzi tutti i parametri disponibili e ignori le altre informazioni:

```
# modinfo -p nouveau
vram_pushbuf:Create DMA push buffers in VRAM (int)
tv_norm:Default TV norm.
        Supported: PAL, PAL-M, PAL-N, PAL-Nc, NTSC-M, NTSC-J,
        hd480i, hd480p, hd576i, hd576p, hd720p, hd1080i.
        Default: PAL
        NOTE Ignored for cards with external TV encoders. (charp)
nofbaccel:Disable fbcon acceleration (int)
fbcon_bpp:fbcon bits-per-pixel (default: auto) (int)
mst:Enable DisplayPort multi-stream (default: enabled) (int)
tv_disable:Disable TV-out detection (int)
ignorelid:Ignore ACPI lid status (int)
duallink:Allow dual-link TMDS (default: enabled) (int)
hdmimhz:Force a maximum HDMI pixel clock (in MHz) (int)
config:option string to pass to driver core (charp)
debug:debug string to pass to driver core (charp)
noaccel:disable kernel/abi16 acceleration (int)
modeset:enable driver (default: auto, 0 = disabled, 1 = enabled, 2 = headless) (int)
```



```
atomic:Expose atomic ioctl (default: disabled) (int)
runpm:disable (0), force enable (1), optimus only default (-1) (int)
```

L'output di esempio mostra tutti i parametri disponibili per il modulo `nouveau`, un modulo del kernel fornito dal *Nouveau Project* come alternativa ai driver proprietari per le schede GPU NVIDIA. L'opzione `modeset`, per esempio, permette di controllare se la risoluzione e la profondità del display saranno impostate nello spazio del kernel invece che nello spazio utente. L'aggiunta di `options nouveau modeset = 0` al file `/etc/modprobe.d/nouveau.conf` disabiliterà la funzionalità `modeset` del kernel.

Se un modulo sta causando problemi, il file `/etc/modprobe.d/blacklist.conf` può essere usato per bloccare il caricamento del modulo. Per esempio, per impedire il caricamento automatico del modulo `nouveau`, la riga `blacklist nouveau` deve essere aggiunta al file `/etc/modprobe.d/blacklist.conf`. Questa azione è richiesta quando è installato il modulo proprietario `nvidia` e il modulo predefinito `nouveau` deve essere messo da parte.

NOTE

È possibile modificare il file predefinito `/etc/modprobe.d/blacklist.conf`. Tuttavia, il metodo preferito è creare un file di configurazione separato, `/etc/modprobe.d/<module_name>.conf`, che conterrà impostazioni specifiche solo per il modulo del kernel specificato.

I File Informativi e i File dei Dispositivi

I comandi `lspci`, `lsusb` e `lsmod` fungono da front-end per leggere le informazioni hardware memorizzate dal sistema operativo. Questo tipo di informazioni è conservato in file speciali nelle directory `/proc` e `/sys`. Queste directory sono punti di montaggio per i filesystem non presenti in una partizione del dispositivo, ma solo nello spazio RAM utilizzato dal kernel per memorizzare la configurazione di runtime e le informazioni sui processi in esecuzione. Tali filesystem non sono destinati all'archiviazione convenzionale di file, quindi sono chiamati pseudo-filesystem e esistono solo mentre il sistema è in esecuzione. La directory `/proc` contiene file con informazioni relative ai processi in esecuzione e alle risorse hardware. Alcuni dei file importanti in `/proc` per l'ispezione dell'hardware sono:

/proc/cpuinfo

Elenca informazioni dettagliate sulla/e CPU trovate dal sistema operativo.

/proc/interrupts

Un elenco degli interrupt per ogni periferica IO per ogni CPU.

/proc/ioports

Elenca le regioni di memoria attualmente in uso di porte input/output.

/proc/dma

Elenca i canali DMA (Direct Memory Access) in uso.

I file all'interno della directory `/sys` hanno ruoli simili a quelli in `/proc`. Tuttavia, la directory `/sys` ha lo scopo specifico di memorizzare informazioni sul dispositivo e dati del kernel relativi all'hardware, mentre `/proc` contiene anche informazioni su varie strutture di dati del kernel, inclusi i processi in esecuzione e la configurazione.

Un'altra directory direttamente correlata ai dispositivi in un sistema Linux standard è `/dev`. Ogni file all'interno di `/dev` è associato a un dispositivo di sistema, in particolare ai dispositivi di archiviazione. Un disco rigido IDE legacy, per esempio, quando è collegato al primo canale IDE della scheda madre è rappresentato dal file `/dev/hda`. Ogni partizione in questo disco sarà identificata da `/dev/hda1`, `/dev/hda2` e così via.

I dispositivi rimovibili sono gestiti dal sottosistema `udev`, che crea i dispositivi corrispondenti in `/dev`. Il kernel Linux acquisisce l'evento di rilevamento hardware e lo passa al processo `udev`, che quindi identifica il dispositivo e crea dinamicamente i file corrispondenti in `/dev`, usando regole predefinite.

Nelle attuali distribuzioni Linux `udev` è responsabile dell'identificazione e della configurazione dei dispositivi già presenti durante l'accensione della macchina (*coldplug detect*) e dei dispositivi identificati mentre il sistema è in esecuzione (*hotplug detect*). `Udev` si affida a `SysFS`, lo pseudo filesystem per le informazioni relative all'hardware montato in `/sys`.

NOTE

Hotplug è il termine utilizzato per indicare il rilevamento e la configurazione di un dispositivo mentre il sistema è in esecuzione, per esempio quando viene inserito un dispositivo USB. Il kernel Linux supporta le funzionalità hotplug dalla versione 2.6, consentendo alla maggior parte dei bus di sistema (PCI, USB, ecc.) di attivare eventi *hotplug* quando un dispositivo è collegato o disconnesso.

Quando vengono rilevati nuovi dispositivi, `udev` cerca una regola corrispondente nelle regole predefinite memorizzate nella directory `/etc/udev/rules.d/`. Le regole più importanti sono fornite dalla distribuzione, ma è possibile aggiungerne di nuove per casi specifici.

I Dispositivi di Archiviazione

In Linux i dispositivi di archiviazione sono genericamente chiamati *block device*, poiché i dati vengono letti da e verso questi dispositivi in blocchi di dati bufferizzati con dimensioni e posizioni

diverse. Ogni dispositivo a blocchi è identificato da un file nella directory `/dev`, con il nome del file a seconda del tipo di dispositivo (IDE, SATA, SCSI, ecc.) e delle sue partizioni. CD/DVD e dispositivi floppy, per esempio, avranno i loro nomi indicati di conseguenza in `/dev`: un'unità CD/DVD collegata al secondo canale IDE verrà identificata come `/dev/hdc` (`/dev/hda` e `/dev/hdb` sono riservati per i dispositivi master e slave sul primo canale IDE) e una vecchia unità floppy verrà identificata come `/dev/fd0`, `/dev/fd1`, ecc..

A partire dalla versione 2.4 del kernel Linux, la maggior parte dei dispositivi di archiviazione viene ora identificata come se fossero dispositivi SCSI, indipendentemente dal tipo di hardware. I dispositivi di blocco IDE, SSD e USB saranno preceduti da `sd`. Per i dischi IDE, verrà usato il prefisso `sd`, ma verrà scelta la terza lettera a seconda che il drive sia un master o uno slave (nel primo canale IDE, il master sarà `sda` e lo slave sarà `sdb`). Le partizioni sono elencate numericamente. I percorsi `/dev/sda1`, `/dev/sda2`, ecc. sono usati per la prima e la seconda partizione del dispositivo a blocchi identificati per primi e `/dev/sdb1`, `/dev/sdb2`, ecc. usati per identificare la prima e la seconda partizione del dispositivo a blocchi identificato per secondo. L'eccezione a questo modello si verifica con schede di memoria (schede SD) e dispositivi NVMe (SSD collegato al bus PCI Express). Per le schede SD, i percorsi `/dev/mmcblk0p1`, `/dev/mmcblk0p2`, ecc. vengono utilizzati per la prima e la seconda partizione del dispositivo identificate per prime e `/dev/mmcblk1p1`, `/dev/mmcblk1p2`, ecc. usato per identificare la prima e la seconda partizione del dispositivo identificato per secondo. I dispositivi NVMe ricevono il prefisso `nvme`, come in `/dev/nvme0n1p1` e `/dev/nvme0n1p2`.

Esercizi Guidati

1. Supponiamo che un sistema operativo non sia in grado di avviarsi dopo aver aggiunto un secondo disco SATA al sistema. Sapendo che tutte le parti non sono difettose, quale potrebbe essere la possibile causa di questo errore?

2. Supponiamo che tu voglia assicurarti che la scheda video esterna connessa al bus PCI del tuo computer desktop appena acquistato sia davvero quella pubblicizzata dal produttore, ma l'apertura del case del computer annullerebbe la garanzia. Quale comando potrebbe essere utilizzato per elencare i dettagli della scheda video, cos' come rilevati dal sistema operativo?

3. La seguente riga fa parte dell'output generato dal comando `lspci`:

```
03:00.0 RAID bus controller: LSI Logic / Symbios Logic MegaRAID SAS 2208 [Thunderbolt]
(rev 05)
```

Quale comando dovresti eseguire per identificare il modulo del kernel in uso per questo specifico dispositivo?

4. Un amministratore di sistema vuole provare diversi parametri per il modulo kernel bluetooth senza riavviare il sistema. Tuttavia, qualsiasi tentativo di scaricare il modulo con `modprobe -r bluetooth` provoca il seguente errore:

```
modprobe: FATAL: Module bluetooth is in use.
```

Qual è la possibile causa di questo errore?

Esercizi Esplorativi

1. Non è raro trovare macchine legacy negli ambienti di produzione, come quando alcune apparecchiature usano una connessione obsoleta per comunicare con il computer di controllo, rendendo necessario prestare particolare attenzione ad alcune peculiarità di queste macchine più vecchie. Alcuni server x86 con un firmware BIOS datato, per esempio, non si avviano se non viene rilevata una tastiera. Come si può evitare questo particolare problema?

2. I sistemi operativi basati sul kernel Linux sono disponibili anche per un'ampia varietà di architetture di computer diverse da quella x86, come nei computer a scheda singola basati sull'architettura ARM. Un utente attento noterà l'assenza del comando `lspci` su tali macchine, come per esempio Raspberry Pi. Quale differenza con le macchine x86 giustifica tale assenza?

3. Molti router di rete hanno una porta USB che consente la connessione di un dispositivo esterno, come un disco rigido USB. Dato che la maggior parte di questi utilizza un sistema operativo basato su Linux, come verrà nominato un disco rigido USB esterno nella directory `/dev/`, supponendo che nessun altro dispositivo a blocchi convenzionale sia presente nel router?

4. Nel 2018 è stata scoperta la vulnerabilità hardware nota come *Meltdown*. Colpisce quasi tutti i processori di molte architetture. Le versioni recenti del kernel Linux possono informare se l'attuale sistema è vulnerabile. Come si possono ottenere queste informazioni?

Sommario

Questa lezione tratta i concetti generali su come il kernel Linux gestisce le risorse hardware, principalmente nell'architettura x86. La lezione affronta i seguenti argomenti:

- In che modo le impostazioni definite nelle utility di configurazione BIOS o UEFI possono influenzare il modo in cui il sistema operativo interagisce con l'hardware.
- Come utilizzare gli strumenti forniti da un sistema Linux standard per ottenere informazioni sull'hardware.
- Come identificare i dispositivi permanenti e rimovibili di archiviazione nel filesystem.

I comandi e le procedure trattate erano:

- Comandi per ispezionare l'hardware rilevato: `lspci` e `lsusb`.
- Comandi per gestire i moduli del kernel: `lsmod` e `modprobe`.
- File speciali relativi all'hardware, o file presenti nella directory `/dev/` o negli pseudo-filesystem in `/proc/` e `/sys/`.

Risposte agli Esercizi Guidati

1. Supponiamo che un sistema operativo non sia in grado di avviarsi dopo aver aggiunto un secondo disco SATA al sistema. Sapendo che tutte le parti non sono difettose, quale potrebbe essere la possibile causa di questo errore?

L'ordine del dispositivo di avvio deve essere configurato nell'utilità di configurazione del BIOS, altrimenti il BIOS potrebbe non essere in grado di eseguire il bootloader.

2. Supponiamo che tu voglia assicurarti che la scheda video esterna connessa al bus PCI del tuo computer desktop appena acquistato sia davvero quella pubblicizzata dal produttore, ma l'apertura del case del computer annullerà la garanzia. Quale comando potrebbe essere utilizzato per elencare i dettagli della scheda video, così come rilevati dal sistema operativo?

Il comando `lspci` elencherà informazioni dettagliate su tutti i dispositivi attualmente connessi al bus PCI.

3. La seguente riga fa parte dell'output generato dal comando `lspci`:

```
03:00.0 RAID bus controller: LSI Logic / Symbios Logic MegaRAID SAS 2208 [Thunderbolt]
(rev 05)
```

Quale comando dovresti eseguire per identificare il modulo del kernel in uso per questo specifico dispositivo?

Il comando `lspci -s 03:00.0 -v` or `lspci -s 03:00.0 -k`

4. Un amministratore di sistema vuole provare diversi parametri per il modulo kernel bluetooth senza riavviare il sistema. Tuttavia, qualsiasi tentativo di scaricare il modulo con `modprobe -r bluetooth` provoca il seguente errore:

```
modprobe: FATAL: Module bluetooth is in use.
```

Qual è la possibile causa di questo errore?

Il modulo `bluetooth` utilizzato da un processo in esecuzione.

Risposte agli Esercizi Esplorativi

1. Non è raro trovare macchine legacy negli ambienti di produzione, come quando alcune apparecchiature usano una connessione obsoleta per comunicare con il computer di controllo, rendendo necessario prestare particolare attenzione ad alcune peculiarità di queste macchine più vecchie. Alcuni server x86 con un firmware BIOS datato, per esempio, non si avviano se non viene rilevata una tastiera. Come si può evitare questo particolare problema?

L'utilità di configurazione del BIOS ha un'opzione per disattivare il blocco del computer quando non viene trovata una tastiera.

2. I sistemi operativi costruiti attorno al kernel Linux sono disponibili anche per un'ampia varietà di architetture di computer diverse da x86, come nei computer basati sull'architettura ARM. Un utente attento noterà l'assenza del comando `lspci` su tali macchine, come in Raspberry Pi. Quale differenza con le macchine x86 giustifica tale assenza?

A differenza della maggior parte delle macchine x86, un computer basato su ARM come il Raspberry Pi non ha un bus PCI, quindi il comando `lspci` è inutile.

3. Molti router di rete dispongono di una porta USB che consente il collegamento di un dispositivo esterno, come un disco rigido USB. Poiché la maggior parte di questi utilizza un sistema operativo basato su Linux, come verrà denominato un disco rigido USB esterno nella directory `/dev/`, supponendo che nel router non sia presente nessun altro dispositivo a blocchi convenzionale?

I moderni kernel Linux identificano i dischi rigidi USB come dispositivi SATA, quindi il file corrispondente sarà `/dev/sda` poiché nel sistema non esistono altri dispositivi a blocchi convenzionali.

4. Nel 2018 è stata scoperta la vulnerabilità hardware nota come *Meltdown*. Colpisce quasi tutti i processori di molte architetture. Le versioni recenti del kernel Linux possono informare se l'attuale sistema è vulnerabile. Come si possono ottenere queste informazioni?

Il file `/proc/cpuinfo` ha una riga che mostra i bug noti per la CPU corrispondente, come bugs: `cpu_meltdown`.



**Linux
Professional
Institute**

101.2 Avviare il sistema

Reference to LPI objectives

LPIC-1 v5, Exam 101, Objective 101.2

Weight

3

Key knowledge areas

- Fornire comandi comuni al boot loader e opzioni al kernel al momento dell'avvio.
- Dimostrare di conoscere la sequenza di avvio da BIOS/UEFI per completare l'avvio.
- Comprensione di SysVinit e systemd.
- Conoscenza di Upstart.
- Controllare gli eventi di avvio nei file di log.

Partial list of the used files, terms and utilities

- `dmesg`
- `journalctl`
- BIOS
- UEFI
- bootloader
- kernel
- `initramfs`
- `init`
- SysVinit

- systemd



101.2 Lezione 1

Certificazione:	LPIC-1
Versione:	5.0
Argomento:	101 L'Architettura di Sistema
Obiettivo:	101.2 Avviare il Sistema
Lezione:	1 di 1

Introduzione

Per controllare la macchina, il componente principale del sistema operativo — il *kernel* — deve essere caricato da un programma chiamato *bootloader*, che a sua volta viene caricato da un *firmware* preinstallato come il BIOS o l'UEFI. Il bootloader può essere personalizzato per passare parametri al kernel, come per esempio quale partizione contiene il *root filesystem* o in quale modalità deve essere eseguito il sistema operativo. Una volta caricato, il kernel continua il processo di avvio identificando e configurando l'hardware. Infine, il kernel richiama l'utilità responsabile dell'avvio e della gestione dei servizi del sistema.

NOTE

Su alcune distribuzioni Linux, i comandi eseguiti in questa lezione potrebbero richiedere privilegi di root.

BIOS o UEFI

Le procedure eseguite dalle macchine x86 per eseguire il bootloader sono diverse se si utilizza BIOS o UEFI. Il BIOS, abbreviazione di *Basic Input/Output System*, è un programma memorizzato in un chip di memoria non volatile collegato alla scheda madre, eseguito ad ogni accensione del computer. Questo tipo di programma si chiama *firmware* e la sua posizione di archiviazione è separata dagli altri dispositivi di archiviazione che il sistema potrebbe avere. Il BIOS presuppone che i primi 440 byte nel primo dispositivo di archiviazione - seguendo l'ordine definito nell'utilità di configurazione del BIOS - siano il primo stadio del bootloader (chiamato anche *bootstrap*). I primi 512 byte di un dispositivo di archiviazione sono denominati MBR (*Master Boot Record*) dei dispositivi di archiviazione utilizzando lo schema di partizionamento DOS standard e, oltre alla prima fase del bootloader, contiene la tabella delle partizioni. Se l'MBR non contiene i dati corretti, il sistema non sarà in grado di avviarsi, a meno che non venga utilizzato un metodo alternativo.

In generale, i passaggi pre-operativi per l'avvio di un sistema dotato di BIOS sono:

1. Il processo POST (*power-on self-test*) viene eseguito per identificare semplici guasti hardware non appena la macchina viene accesa.
2. Il BIOS attiva i componenti di base per caricare il sistema, come l'output video, la tastiera e i supporti di archiviazione.
3. Il BIOS carica il primo stadio del bootloader dall'MBR (i primi 440 byte del primo dispositivo, come definito nell'utilità di configurazione del BIOS).
4. Il primo stadio del bootloader richiama il secondo stadio, responsabile della presentazione delle opzioni di avvio e del caricamento del kernel.

L'UEFI, abbreviazione di *Unified Extensible Firmware Interface*, differisce dal BIOS in alcuni punti chiave. Come BIOS, l'UEFI è anche un firmware, ma è in grado di identificare le partizioni e leggere molti filesystem in esse contenuti. L'UEFI non si basa sull'MBR, prendendo in considerazione solo le impostazioni memorizzate nella sua memoria non volatile (*NVRAM*) collegata alla scheda madre. Queste definizioni indicano la posizione dei programmi compatibili UEFI, chiamati *EFI applications* che verranno eseguiti automaticamente o richiamati da un menu di avvio. Le applicazioni EFI possono essere bootloader, selettori del sistema operativo, strumenti per la diagnostica e la riparazione del sistema, ecc. Devono trovarsi in una partizione del dispositivo di archiviazione convenzionale e in un file system compatibile. I filesystem compatibili standard sono FAT12, FAT16 e FAT32 per dispositivi a blocchi e ISO-9660 per supporti ottici. Questo approccio consente l'implementazione di strumenti molto più sofisticati di quelli possibili con il BIOS

La partizione contenente le applicazioni EFI è chiamata *EFI System Partition* o solo ESP. Questa partizione non deve essere condivisa con altri filesystem di sistema, come il filesystem di root o i filesystem dei dati utente. La directory EFI nella partizione ESP contiene le applicazioni indicate dalle voci salvate nella NVRAM.

In generale, i passaggi di avvio su un sistema con UEFI sono:

1. Il processo POST (*power-on self-test*) viene eseguito per identificare semplici guasti hardware non appena la macchina viene accesa.
2. L'UEFI attiva i componenti di base per caricare il sistema, come l'output video, la tastiera e i supporti di archiviazione.
3. Il firmware UEFI legge le definizioni archiviate in NVRAM per eseguire l'applicazione EFI predefinita memorizzata nel filesystem della partizione ESP. Di solito, l'applicazione EFI predefinita è un bootloader.
4. Se l'applicazione EFI predefinita è un bootloader, caricherà il kernel per avviare il sistema operativo.

Lo standard UEFI supporta inoltre una funzione chiamata *Secure Boot*, che consente solo l'esecuzione di applicazioni EFI firmate, ovvero applicazioni EFI autorizzate dal produttore dell'hardware. Questa funzione aumenta la protezione da software dannoso, ma può rendere difficile l'installazione di sistemi operativi non supportati dal produttore.

Il Bootloader

Il bootloader più popolare per Linux nell'architettura x86 è GRUB (*Grand Unified Bootloader*). Non appena viene richiamato dal BIOS o dall'UEFI, GRUB visualizza un elenco di sistemi operativi disponibili per l'avvio. A volte l'elenco non appare automaticamente, ma può essere invocato premendo `Shift` mentre GRUB viene chiamato dal BIOS. Nei sistemi UEFI, si dovrebbe usare invece il tasto `Esc`.

Dal menu di GRUB è possibile scegliere quale dei kernel installati deve essere caricato e passarvi nuovi parametri. La maggior parte dei parametri del kernel segue il modello `option=value`. Alcuni dei parametri del kernel più utili sono:

acpi

Abilita/disabilita il supporto ACPI. `acpi=off` disabiliterà il supporto per ACPI.

init

Imposta un iniziatore di sistema alternativo. Per esempio, `init=/bin/bash` imposterà la *shell* Bash come iniziatore. Ciò significa che una sessione di shell inizierà subito dopo il processo di

avvio del kernel.

systemd.unit

Imposta il *systemd* target da attivare. Per esempio, `systemd.unit=graphical.target`. Systemd accetta anche i *runlevel* numerici definiti per SysV. Per attivare il runlevel 1, per esempio, è necessario includere solo il numero 1 o la lettera S (abbreviazione di “single”) come parametro del kernel.

mem

Imposta la quantità di RAM disponibile per il sistema. Questo parametro è utile per le macchine virtuali in modo da limitare la quantità di RAM disponibile per ciascun guest. L'uso di `mem=512M` limiterà a 512 megabyte la quantità di RAM disponibile per un particolare sistema guest.

maxcpus

Limita il numero di processori (o *core* del processore) visibili al sistema nelle macchine multiprocessore simmetriche. È utile anche per macchine virtuali. Un valore di 0 disattiva il supporto per macchine multiprocessore e ha lo stesso effetto del parametro del kernel `nosmp`. Il parametro `maxcpus=2` limiterà il numero di processori disponibili al sistema operativo a due.

quiet

Nasconde la maggior parte dei messaggi di avvio.

vga

Seleziona una modalità video. Il parametro `vga=ask` mostrerà un elenco delle modalità disponibili tra cui scegliere.

root

Imposta la partizione root, distinta da quella preconfigurata nel bootloader. Per esempio, `root=/dev/sda3`.

rootflags

Opzioni di *mount* per il filesystem di root.

ro

Rende il montaggio iniziale del root filesystem di sola lettura.

rw

Consente la scrittura nel root filesystem durante il montaggio iniziale.

Di solito modificare i parametri del kernel non è necessario, ma può essere utile per rilevare e

risolvere problemi relativi al sistema operativo. I parametri del kernel devono essere aggiunti al file `/etc/default/grub` nella riga `GRUB_CMDLINE_LINUX` per renderli persistenti durante i riavvii. Un nuovo file di configurazione per il bootloader deve essere generato ogni volta che cambia `/etc/default/grub`, il che è realizzato dal comando `grub-mkconfig -o /boot/grub/grub.cfg`. Una volta che il sistema operativo è in esecuzione, i parametri del kernel usati per caricare la sessione corrente sono disponibili per la lettura nel file `/proc/cmdline`.

NOTE La configurazione di GRUB verrà discussa ulteriormente in una lezione successiva.

Inizializzazione del Sistema

Oltre che dal kernel, il sistema operativo dipende da altri componenti che forniscono le funzionalità previste. Molti di questi componenti vengono caricati durante il processo di inizializzazione del sistema, variando da semplici script di shell a programmi di servizio più complessi. Gli script vengono spesso utilizzati per eseguire attività di breve durata che verranno avviate e terminate durante il processo di inizializzazione del sistema. I servizi, noti anche come *daemon*, possono essere sempre attivi poiché possono essere responsabili di aspetti fondamentali del sistema operativo.

La diversità dei modi in cui gli script di avvio e i daemon con le più svariate caratteristiche possono essere incorporate in una distribuzione Linux è enorme, un fatto che storicamente ha ostacolato lo sviluppo di un'unica soluzione che soddisfi le aspettative dei manutentori e degli utenti di tutte le distribuzioni Linux. Tuttavia, qualsiasi strumento scelto dai manutentori della distribuzione per eseguire questa funzione sarà almeno in grado di avviare, arrestare e riavviare i servizi di sistema. Queste azioni vengono spesso eseguite dal sistema stesso dopo un aggiornamento del software in automatico, ma l'amministratore di sistema invece dovrà quasi sempre riavviare manualmente il servizio dopo aver apportato modifiche al relativo file di configurazione.

È altresì conveniente per un amministratore di sistema essere in grado di attivare, a seconda delle circostanze, un particolare set di demoni. Dovrebbe essere possibile, per esempio, eseguire solo un set minimo di servizi per eseguire attività di manutenzione di sistema.

NOTE In senso stretto, il sistema operativo è solo il kernel e i suoi componenti che controllano l'hardware e gestiscono tutti i processi. È comune, tuttavia, usare il termine "sistema operativo" più liberamente, per designare un intero gruppo di programmi distinti che compongono l'ambiente software in cui l'utente può eseguire le attività di calcolo di base.

L'inizializzazione del sistema operativo inizia quando il bootloader carica il kernel nella RAM. Quindi, il kernel prenderà in carico la/le CPU e inizierà a rilevare e configurare gli aspetti

fondamentali del sistema operativo, come la configurazione hardware di base e l'indirizzamento della memoria.

Il kernel aprirà quindi l'*initramfs* (*initial RAM filesystem*). Initramfs è un archivio contenente un filesystem usato come filesystem root temporaneo durante il processo di avvio. Lo scopo principale di un file initramfs è quello di fornire i moduli richiesti in modo che il kernel possa accedere al filesystem di root “reale” del sistema operativo.

Non appena il filesystem di root è disponibile, il kernel monterà tutti i filesystem configurati in `/etc/fstab` e quindi eseguirà il primo programma, un'utilità chiamata `init`. Il programma `init` è responsabile dell'esecuzione di tutti gli script di inizializzazione e dei demoni di sistema. Esistono implementazioni distinte di tali iniziatori di sistema oltre all'*init* tradizionale, come *systemd* e *Upstart*. Una volta caricato il programma `init`, initramfs viene rimosso dalla RAM.

SysV standard

Un gestore di servizi basato sullo standard SysVinit controlla quali demoni e risorse saranno disponibili impiegando il concetto di *runlevels*. I runlevel sono numerati da 0 a 6 e sono progettati dai manutentori della distribuzione per soddisfare scopi specifici. Le uniche definizioni di runlevel condivise tra tutte le distribuzioni sono i runlevel 0, 1 e 6.

systemd

systemd è un moderno gestore di sistemi e servizi con un livello di compatibilità per i comandi e i runlevel di SysV. *systemd* ha una struttura concorrente, impiega socket e D-Bus per l'attivazione dei servizi, esecuzione dei demoni su richiesta, monitoraggio dei processi con *cgroup*, supporto alle snapshot, ripristino della sessione di sistema, controllo dei punti di montaggio e controllo dei servizi basato sulle relative dipendenze. Negli ultimi anni la maggior parte delle principali distribuzioni Linux ha gradualmente adottato *systemd* come gestore di sistema predefinito..

Upstart

Come *systemd*, *Upstart* è un sostituto di `init`. L'obiettivo di *Upstart* è accelerare il processo di avvio parallelizzando il processo di caricamento dei servizi di sistema. *Upstart* è stato utilizzato dalle distribuzioni basate su Ubuntu in precedenti versioni alle attuali, ma oggi ha lasciato il posto a *systemd*..

Ispezionare l'inizializzazione

Possono verificarsi errori durante il processo di avvio, ma potrebbero non essere così critici da arrestare completamente il sistema operativo. Nonostante ciò, questi errori possono compromettere il comportamento previsto del sistema. Tutti gli errori generano messaggi che

possono essere utilizzati per future indagini, in quanto contengono informazioni preziose su quando e come si è verificato un errore. Anche quando non vengono generati messaggi di errore, le informazioni raccolte durante il processo di avvio possono essere utili ai fini della regolazione e della configurazione.

Lo spazio di memoria in cui il kernel memorizza i suoi messaggi, inclusi i messaggi di avvio, è chiamato *kernel ring buffer*. I messaggi vengono conservati nel buffer di memoria del kernel anche se non mostrati durante il processo di inizializzazione a causa di immagini e/o animazioni che ne impediscono la visione. Tuttavia il buffer di memoria del kernel perde tutti i dati quando il sistema è spento o eseguendo il comando `dmesg --clear`. Senza opzioni, il comando `dmesg` mostra i messaggi nel buffer di memoria del kernel corrente:

```
$ dmesg
[ 5.262389] EXT4-fs (sda1): mounted filesystem with ordered data mode. Opts: (null)
[ 5.449712] ip_tables: (C) 2000-2006 Netfilter Core Team
[ 5.460286] systemd[1]: systemd 237 running in system mode.
[ 5.480138] systemd[1]: Detected architecture x86-64.
[ 5.481767] systemd[1]: Set hostname to <torre>.
[ 5.636607] systemd[1]: Reached target User and Group Name Lookups.
[ 5.636866] systemd[1]: Created slice System Slice.
[ 5.637000] systemd[1]: Listening on Journal Audit Socket.
[ 5.637085] systemd[1]: Listening on Journal Socket.
[ 5.637827] systemd[1]: Mounting POSIX Message Queue File System...
[ 5.638639] systemd[1]: Started Read required files in advance.
[ 5.641661] systemd[1]: Starting Load Kernel Modules...
[ 5.661672] EXT4-fs (sda1): re-mounted. Opts: errors=remount-ro
[ 5.694322] lp: driver loaded but no devices found
[ 5.702609] ppdev: user-space parallel port driver
[ 5.705384] parport_pc 00:02: reported by Plug and Play ACPI
[ 5.705468] parport0: PC-style at 0x378 (0x778), irq 7, dma 3
[PCSP,TRISTATE,COMPAT,EPP,ECP,DMA]
[ 5.800146] lp0: using parport0 (interrupt-driven).
[ 5.897421] systemd-journald[352]: Received request to flush runtime journal from PID 1
```

L'output di `dmesg` può essere lungo centinaia di righe, quindi la lista precedente contiene solo un piccolo estratto che mostra il kernel che chiama il *systemd service manager*. I valori all'inizio delle righe sono la quantità di secondi dall'inizio del caricamento del kernel.

Nei sistemi basati su `systemd`, il comando `journalctl` mostrerà i messaggi di inizializzazione con le opzioni `-b``, `--boot`, `-k` o `--dmesg`. Il comando `journalctl --list-boots` mostra un elenco di avvii precedenti a quello corrente, un loro numero di identificazione, data e ora di avvio e primo e ultimo messaggio corrispondente:

\$ journalctl --list-boots

```
-4 9e5b3eb4952845208b841ad4dbefa1a6 Thu 2019-10-03 13:39:23 -03--Thu 2019-10-03 13:40:30 -03
-3 9e3d79955535430aa43baa17758f40fa Thu 2019-10-03 13:41:15 -03--Thu 2019-10-03 14:56:19 -03
-2 17672d8851694e6c9bb102df7355452c Thu 2019-10-03 14:56:57 -03--Thu 2019-10-03 19:27:16 -03
-1 55c0d9439bfb4e85a20a62776d0dbb4d Thu 2019-10-03 19:27:53 -03--Fri 2019-10-04 00:28:47 -03
0 08fbbbebd9f964a74b8a02bb27b200622 Fri 2019-10-04 00:31:01 -03--Fri 2019-10-04 10:17:01 -03
```

I log di inizializzazione precedenti vengono mantenuti anche nei sistemi basati su systemd, quindi è ancora possibile controllare i messaggi delle precedenti sessioni del sistema operativo. Se vengono fornite le opzioni `-b 0` o `--boot = 0`, verranno mostrati i messaggi per l'avvio corrente. Le opzioni `-b -1` o `--boot = -1` mostreranno i messaggi della precedente inizializzazione. Le opzioni `-b -2` o `--boot = -2` mostreranno i messaggi dall'inizializzazione prima di quello e così via. Il seguente estratto mostra il kernel che richiama il gestore del servizio systemd nell'ultimo avvio di sistema:

\$ journalctl -b 0

```
oct 04 00:31:01 ubuntu-host kernel: EXT4-fs (sda1): mounted filesystem with ordered data
mode. Opts: (null)
oct 04 00:31:01 ubuntu-host kernel: ip_tables: (C) 2000-2006 Netfilter Core Team
oct 04 00:31:01 ubuntu-host systemd[1]: systemd 237 running in system mode.
oct 04 00:31:01 ubuntu-host systemd[1]: Detected architecture x86-64.
oct 04 00:31:01 ubuntu-host systemd[1]: Set hostname to <torre>.
oct 04 00:31:01 ubuntu-host systemd[1]: Reached target User and Group Name Lookups.
oct 04 00:31:01 ubuntu-host systemd[1]: Created slice System Slice.
oct 04 00:31:01 ubuntu-host systemd[1]: Listening on Journal Audit Socket.
oct 04 00:31:01 ubuntu-host systemd[1]: Listening on Journal Socket.
oct 04 00:31:01 ubuntu-host systemd[1]: Mounting POSIX Message Queue File System...
oct 04 00:31:01 ubuntu-host systemd[1]: Started Read required files in advance.
oct 04 00:31:01 ubuntu-host systemd[1]: Starting Load Kernel Modules...
oct 04 00:31:01 ubuntu-host kernel: EXT4-fs (sda1): re-mounted. Opts:
commit=300,barrier=0,errors=remount-ro
oct 04 00:31:01 ubuntu-host kernel: lp: driver loaded but no devices found
oct 04 00:31:01 ubuntu-host kernel: ppdev: user-space parallel port driver
oct 04 00:31:01 ubuntu-host kernel: parport_pc 00:02: reported by Plug and Play ACPI
oct 04 00:31:01 ubuntu-host kernel: parport0: PC-style at 0x378 (0x778), irq 7, dma 3
[PCSP, TRISTATE, COMPAT, EPP, ECP, DMA]
oct 04 00:31:01 ubuntu-host kernel: lp0: using parport0 (interrupt-driven).
oct 04 00:31:01 ubuntu-host systemd-journald[352]: Journal started
oct 04 00:31:01 ubuntu-host systemd-journald[352]: Runtime journal
(/run/log/journal/abb765408f3741ae9519ab3b96063a15) is 4.9M, max 39.4M, 34.5M free.
oct 04 00:31:01 ubuntu-host systemd-modules-load[335]: Inserted module 'lp'
oct 04 00:31:01 ubuntu-host systemd-modules-load[335]: Inserted module 'ppdev'
```

```
oct 04 00:31:01 ubuntu-host systemd-modules-load[335]: Inserted module 'parport_pc'
oct 04 00:31:01 ubuntu-host systemd[1]: Starting Flush Journal to Persistent Storage...
```

L'inizializzazione e altri messaggi emessi dal sistema operativo sono memorizzati in file all'interno della directory `/var/log/`. Se si verifica un errore critico e il sistema operativo non è in grado di continuare il processo di inizializzazione dopo aver caricato il kernel e `initramfs`, è possibile utilizzare un supporto di avvio alternativo per avviare il sistema e accedere al file system corrispondente. Quindi, i file in `/var/log/` possono essere controllati per scoprire i motivi che causano l'interruzione del processo di avvio. Le opzioni `-D` o `--directory` del comando `journalctl` possono essere usate per leggere i messaggi di registro in directory diverse da `/var/log/journal/``, che è la posizione predefinita per i messaggi di registro di `systemd`. Poiché i messaggi di log di `systemd` non sono memorizzati in formato testo, per leggerli è necessario il comando ``journalctl`.

Esercizi Guidati

1. Su una macchina dotata di un firmware BIOS, dove si trova il programma di bootstrap?

2. Il firmware UEFI supporta funzionalità estese fornite da programmi esterni, chiamati applicazioni EFI. Queste applicazioni, tuttavia, hanno la loro posizione speciale. In quale parte del sistema si troverebbero le applicazioni EFI?

3. I bootloader consentono il passaggio di parametri personalizzati del kernel prima di caricarlo. Supponiamo che il sistema non sia in grado di avviarsi a causa di una posizione errata del filesystem di root. Come verrebbe dato il file system root corretto, situato in `/dev/sda3`, come parametro per il kernel?

4. Il processo di avvio di una macchina Linux termina con il seguente messaggio:

```
ALERT! /dev/sda3 does not exist. Dropping to a shell!
```

Qual è la probabile causa di questo problema?

Esercizi Esplorativi

1. Il bootloader presenterà un elenco di sistemi operativi tra cui scegliere quando sul sistema è installato più di un sistema operativo. Tuttavia, un sistema operativo appena installato può sovrascrivere l'MBR del disco rigido, cancellando il primo stadio del bootloader e rendendo inaccessibile l'altro sistema operativo. Perché ciò non dovrebbe accadere su una macchina dotata di un firmware UEFI?

2. Qual è una conseguenza comune dell'installazione di un kernel personalizzato senza fornire un'immagine initramfs appropriata?

3. Il log di inizializzazione è lungo centinaia di righe, quindi l'output del comando `dmesg` viene spesso reindirizzato a un comando di paginazione — come il comando `less` — per facilitarne la lettura. Con quale opzione `dmesg` impaginerà automaticamente il suo output, eliminando la necessità di usare esplicitamente un altro comando?

4. Un disco rigido contenente l'intero filesystem di una macchina offline è stato rimosso e collegato a una macchina funzionante come unità secondaria. Supponendo che il suo punto di montaggio sia `/mnt/hd`, come verrebbe usato `journalctl` per ispezionare il contenuto dei file journal situati in `/mnt/hd/var/log/journal/`?

Sommario

Questa lezione tratta la sequenza di avvio in un sistema Linux standard. La corretta conoscenza di come funziona il processo di avvio di un sistema Linux aiuta a prevenire errori che possono rendere inaccessibile il sistema. La lezione passa attraverso le seguenti aree tematiche:

- In che modo differiscono i metodi di avvio BIOS e UEFI.
- Tipiche fasi di inizializzazione del sistema.
- Recupero dei messaggi di avvio.

I comandi e le procedure trattate erano:

- Parametri comuni del kernel.
- Comandi per leggere i messaggi di avvio: `dmesg` e `journalctl`.

Risposte agli Esercizi Guidati

1. Su una macchina dotata di un firmware BIOS, dove si trova il programma di bootstrap?

Nell'MBR del primo dispositivo di archiviazione, come definito nell'utilità di configurazione del BIOS.

2. Il firmware UEFI supporta funzionalità estese fornite da programmi esterni, chiamati applicazioni EFI. Queste applicazioni, tuttavia, hanno la loro posizione speciale. In quale parte del sistema si troverebbero le applicazioni EFI?

Le applicazioni EFI sono archiviate nella EFI System Partition (ESP), situata in un qualsiasi blocco di archiviazione disponibile su un filesystem compatibile.(generalmente un filesystem FAT32).

3. I bootloader consentono il passaggio di parametri personalizzati del kernel prima di caricarlo. Supponiamo che il sistema non sia in grado di avviarsi a causa di una posizione errata del filesystem di root. Come verrebbe dato il file system root corretto, situato in `/dev/sda3`, come parametro per il kernel?

Dovrebbe essere usato il parametro `root`, come in `root=/dev/sda3`.

4. Il processo di avvio di una macchina Linux termina con il seguente messaggio:

```
ALERT! /dev/sda3 does not exist. Dropping to a shell!
```

Qual è la probabile causa di questo problema?

Il kernel non è stato in grado di trovare il dispositivo `/dev/sda3`, impostato come filesystem di root.

Risposte agli Esercizi Guidati

1. Il bootloader presenterà un elenco di sistemi operativi tra cui scegliere quando sul sistema è installato più di un sistema operativo. Tuttavia, un sistema operativo appena installato può sovrascrivere l'MBR del disco rigido, cancellando il primo stadio del bootloader e rendendo inaccessibile l'altro sistema operativo. Perché ciò non dovrebbe accadere su una macchina dotata di un firmware UEFI?

I computer UEFI non utilizzano l'MBR del disco rigido per memorizzare il primo stadio del bootloader.

2. Qual è una conseguenza comune dell'installazione di un kernel personalizzato senza fornire un'immagine initramfs appropriata?

Il filesystem di root potrebbe essere inaccessibile se il suo tipo è fornito come modulo esterno del kernel.

3. Il log di inizializzazione è lungo centinaia di righe, quindi l'output del comando `dmesg` viene spesso reindirizzato a un comando di paginazione — come il comando `less` — per facilitarne la lettura. Con quale opzione `dmesg` impaginerà automaticamente il suo output, eliminando la necessità di usare esplicitamente un altro comando?

I comandi `dmesg -H` o `dmesg --human` abiliteranno la paginazione di default.

4. Un disco rigido contenente l'intero filesystem di una macchina offline è stato rimosso e collegato a una macchina funzionante come unità secondaria. Supponendo che il suo punto di montaggio sia `/mnt/hd`, come verrebbe `journalctl` per ispezionare il contenuto dei file journal situati in `/mnt/hd/var/log/journal/`?

Con i comandi `journalctl -D /mnt/hd/var/log/journal` o `journalctl --directory=/mnt/hd/var/log/journal`.



101.3 Modificare runlevel / target di avvio e spegnere o riavviare il sistema

Obiettivi LPI di riferimento

LPIC-1 v5, Exam 101, Objective 101.3

Peso

3

Aree di Conoscenza Chiave

- Impostare il runlevel predefinito o il target di avvio.
- Cambiare tra runlevel / target di avvio inclusa la modalità utente singolo.
- Arresto e riavvio dalla Command Line.
- Avvisare gli utenti prima di cambiare runlevel / target di avvio o altri eventi di sistema importanti.
- Terminare correttamente i processi.
- Conoscenza di acpid.

Di seguito è riportato un elenco parziale dei file, dei termini e dei comandi utilizzati

- `/etc/inittab`
- `shutdown`
- `init`
- `/etc/init.d/`
- `telinit`
- `systemd`
- `systemctl`

- `/etc/systemd/`
- `/usr/lib/systemd/`
- `wall`



101.3 Lezione 1

Certificazione:	LPIC-1
Versione:	5.0
Argomento:	101 L'Architettura di Sistema
Obiettivo:	101.3 Cambiare runlevel / Avviare target e Spegnere o Riavviare il Sistema
Lezione:	1 di 1

Introduzione

Una caratteristica comune tra i sistemi operativi che seguono i principi di progettazione Unix è l'impiego di processi separati per controllare funzioni distinte del sistema. Questi processi, chiamati *demoni* (*daemon* o, più in generale, *servizi*), sono anche responsabili di funzionalità estese alla base del sistema operativo, come servizi di applicazioni di rete (server HTTP, condivisione file, e-mail, ecc.), database, configurazioni su richiesta, ecc. Sebbene Linux utilizzi un *kernel* monolitico, molti suoi aspetti a basso livello sono influenzati dai demoni, come il bilanciamento del carico e la configurazione del firewall.

A seconda dello scopo di funzionamento del sistema avremo attivi alcuni demoni invece di altri. Anche il set di demoni attivi dovrebbe essere modificabile in fase di esecuzione, in modo che i servizi possano essere avviati o arrestati senza dover riavviare l'intero sistema. Per affrontare questo problema, ogni principale distribuzione Linux offre una qualche forma di utilità di gestione dei servizi per gestire il sistema.

I servizi possono essere controllati da script di shell o da un programma e dai suoi file di configurazione di supporto. Il primo metodo è implementato dallo standard *SysVinit*, noto anche

come *System V* o semplicemente *SysV*. Il secondo metodo è implementato da *systemd* e *Upstart*. Storicamente, i gestori di servizi basati su *SysV* erano i più utilizzati dalle distribuzioni Linux. Oggi, i gestori di servizi basati su *systemd* sono largamente implementati nella maggior parte delle distribuzioni Linux. Il service manager è il primo programma lanciato dal kernel durante il processo di avvio, quindi il suo PID (Process Identification Number) sarà sempre 1.

SysVinit

Un gestore di servizi basato sullo standard SysVinit fornirà set predefiniti di stati di sistema, chiamati *runlevel*, e i relativi file di script di servizio da eseguire. I runlevel sono numerati da "0" a "6", in genere assegnati ai seguenti scopi:

Runlevel 0

Spegnimento del sistema.

Runlevel 1, s o single

Modalità utente singolo, senza rete e altre funzionalità non essenziali (modalità manutenzione).

Runlevel 2, 3 o 4

Modalità multiutente. Gli utenti possono accedere tramite console o rete. I runlevel 2 e 4 non sono usati spesso.

Runlevel 5

Modalità multiutente. È equivalente a 3, oltre al login in modalità grafica.

Runlevel 6

Riavvio del sistema.

Il programma responsabile della gestione dei runlevel e dei demoni/risorse associati è `/sbin/init`. Durante l'inizializzazione del sistema, il programma `init` identifica il runlevel richiesto, definito da un parametro del kernel o nel file `/etc/inittab`, e carica gli script associati ivi elencati per il runlevel dato. Ogni runlevel può avere molti file di servizio associati, di solito script nella directory `/etc/init.d/`. Poiché non tutti i runlevel sono equivalenti attraverso diverse distribuzioni Linux, una breve descrizione dello scopo del runlevel può anche essere trovata nelle distribuzioni basate su *SysV*.

La sintassi del file `/etc/inittab` usa questo formato:

```
id:runlevels:action:process
```

"Id" è un nome generico di massimo quattro caratteri utilizzato per identificare la voce. La voce `runlevels` è un elenco di numeri di runlevel per i quali deve essere eseguita un'azione specifica. Il termine `action` definisce come `init` eseguirà il processo indicato dal termine `process`. Le azioni disponibili sono:

boot

Il processo verrà eseguito durante l'inizializzazione del sistema. Il campo `runlevels` viene ignorato.

bootwait

Il processo verrà eseguito durante l'inizializzazione del sistema e `init` attenderà fino al termine per continuare. Il campo `runlevels` viene ignorato.

sysinit

Il processo verrà eseguito dopo l'inizializzazione del sistema, indipendentemente dal runlevel. Il campo `runlevel` viene ignorato.

wait

Il processo verrà eseguito per i runlevel indicati e `init` attenderà fino al termine per continuare.

respawn

Il processo verrà riavviato se viene terminato.

ctrlaltdel

Il processo verrà eseguito quando il processo `init` riceve il segnale `SIGINT`, attivato quando viene premuta la sequenza di tasti `Ctrl` + `Alt` + `Canc`.

Il runlevel predefinito — quello che verrà scelto se nessun altro viene dato come parametro del kernel — è anche definito in `/etc/inittab`, nella voce `id:x:initdefault`. `x` è il numero del runlevel predefinito. Questo numero non dovrebbe mai essere `0` o `6`, dato che causerebbe l'arresto o il riavvio del sistema non appena termina il processo di avvio. Di seguito è mostrato un tipico file `/etc/inittab`:

```
# Default runlevel
id:3:initdefault:

# Configuration script executed during boot
si::sysinit:/etc/init.d/rcS

# Action taken on runlevel S (single user)
```

```
~:S:wait:/sbin/sulogin

# Configuration for each execution level
10:0:wait:/etc/init.d/rc 0
11:1:wait:/etc/init.d/rc 1
12:2:wait:/etc/init.d/rc 2
13:3:wait:/etc/init.d/rc 3
14:4:wait:/etc/init.d/rc 4
15:5:wait:/etc/init.d/rc 5
16:6:wait:/etc/init.d/rc 6

# Action taken upon ctrl+alt+del keystroke
ca::ctrlaltdel:/sbin/shutdown -r now

# Enable consoles for runlevels 2 and 3
1:23:respawn:/sbin/getty tty1 VC linux
2:23:respawn:/sbin/getty tty2 VC linux
3:23:respawn:/sbin/getty tty3 VC linux
4:23:respawn:/sbin/getty tty4 VC linux

# For runlevel 3, also enable serial
# terminals ttyS0 and ttyS1 (modem) consoles
S0:3:respawn:/sbin/getty -L 9600 ttyS0 vt320
S1:3:respawn:/sbin/mgetty -x0 -D ttyS1
```

Il comando `telinit q` dovrebbe essere eseguito ogni volta che il file `/etc/inittab` viene modificato. L'argomento `q` (o `Q`) dice a `init` di ricaricare la sua configurazione. Tale passaggio è importante per evitare l'arresto del sistema a causa di una configurazione errata in `/etc/inittab`.

Gli script usati da `init` per impostare ciascun runlevel sono memorizzati nella directory `/etc/init.d/`. Ogni runlevel ha una directory associata in `/etc/`, chiamata `/etc/rc0.d/`, `/etc/rc1.d/`, `/etc/rc2.d/`, ecc., con all'interno gli script che dovrebbero essere eseguiti all'avvio del runlevel corrispondente. Poiché lo stesso script può essere utilizzato da runlevel diversi, i file in quelle directory sono solo collegamenti simbolici agli script effettivi in `/etc/init.d/`. Inoltre, la prima lettera del nome file del collegamento nella directory del runlevel indica se il servizio deve essere avviato o terminato per il runlevel corrispondente. Il nome file di un collegamento che inizia con la lettera `K` determina che il servizio verrà terminato(kill) quando si accede al runlevel. Iniziando con la lettera `S`, il servizio verrà avviato(start) quando si accede al runlevel. La directory `/etc/rc1.d/`, per esempio, avrà molti collegamenti agli script di rete che iniziano con la lettera `K`, considerando che il runlevel 1 è il runlevel per singolo utente, senza connettività di rete.

Il comando `runlevel` mostra il runlevel corrente per il sistema in uso. Il comando `runlevel` mostra due valori, il primo è il runlevel precedente e il secondo è il runlevel corrente:

```
$ runlevel
N 3
```

La lettera N nell'output mostra che il runlevel non è cambiato dall'ultimo avvio. Nell'esempio, il `runlevel 3` è l'attuale runlevel del sistema.

Lo stesso programma `init` può essere usato per alternare i runlevel in un sistema in esecuzione, senza la necessità di riavviare. Il comando `telinit` può anche essere usato per alternare tra i runlevel. Per esempio, i comandi `telinit 1`, `telinit s` o `telinit S` cambieranno il sistema al runlevel 1.

systemd

Attualmente, `systemd` è il set di strumenti più utilizzato per gestire risorse e servizi di sistema, che vengono definiti come *units* da `systemd`. Un'unità è composta da un nome, un tipo e un file di configurazione corrispondente. Per esempio, l'unità per un processo del server *httpd* (come il web server Apache) sarà `httpd.service` nelle distribuzioni basate su Red Hat e anche il suo file di configurazione sarà chiamato `httpd.service` (nelle distribuzioni basate su Debian questa unità è chiamata `apache2.service`).

Esistono sette tipi distinti di unità `systemd`:

service

Il tipo di unità più comune, per risorse di sistema attive che possono essere avviate, interrotte e ricaricate.

socket

Il tipo di unità *socket* può essere un socket del filesystem o un socket di rete. Tutte le unità socket hanno un'unità di servizio corrispondente, caricata quando il socket riceve una richiesta.

device

Un'unità *device* è associata a un dispositivo hardware identificato dal kernel. Un device verrà preso come unità di sistema solo se esiste una regola `udev` per questo scopo. Un'unità device può essere utilizzata per risolvere le dipendenze di configurazione quando viene rilevato un determinato hardware, dato che le proprietà della regola `udev` possono essere utilizzate come parametri per l'unità device.

mount

Un'unità *mount* è una definizione del punto mount nel filesystem, simile a una voce in `/etc/fstab`.

automount

Un'unità di *automount* è anche una definizione del punto di montaggio nel filesystem, ma montata automaticamente. Ogni unità di automount ha un'unità di montaggio corrispondente, che viene avviata quando si accede al punto di montaggio di montaggio automatico.

target

Un'unità *target* è un raggruppamento di altre unità, gestite come una singola unità.

snapshot

Un'unità *snapshot* è uno stato salvato del gestore di systemd (non disponibile su ogni distribuzione Linux).

Il comando principale per il controllo delle unità systemd è `systemctl`. Il comando `systemctl` viene utilizzato per eseguire tutte le attività relative all'attivazione, disattivazione, esecuzione, interruzione, monitoraggio, ecc. delle unità. Per un'unità fittizia chiamata `unit.service`, per esempio, le azioni `systemctl` più comuni saranno:

systemctl start unit.service

Avvia `unit`.

systemctl stop unit.service

Interrompi `unit`.

systemctl restart unit.service

Riavvia `unit`.

systemctl status unit.service

Mostra lo stato di `unit`, anche se è in esecuzione o meno.

systemctl is-active unit.service

Mostra se `unit` è in esecuzione o meno.

systemctl enable unit.service

Abilita `unit`, ovvero, `unit` verrà avviato durante l'inizializzazione del sistema.

systemctl disable unit.service

`unit` non si avvierà insieme al sistema.

systemctl is-enabled unit.service

Verifica se `unit` si avvia con il sistema. La risposta è memorizzata nella variabile `$?`. Il valore `0` indica che `unit` si avvia con il sistema e il valore `1` indica che `unit` non avvia con il sistema.

Le installazioni più recenti di `systemd` elencheranno effettivamente la configurazione di un'unità relativamente alla fase di boot. Per esempio:

NOTE

```
$ systemctl is-enabled apparmor.service
enabled
```

Se nel sistema non esistono altre unità con lo stesso nome, è possibile eliminare il suffisso dopo il punto. Se, per esempio, c'è una sola unità `httpd` di tipo `service`, allora è sufficiente solo `httpd` come parametro di unità per `systemctl`.

Il comando `systemctl` può anche controllare *system targets*. L'unità `multi-user.target`, per esempio, combina tutte le unità richieste dall'ambiente di sistema multiutente. È simile al runlevel numero 3 in un sistema che utilizza SysV.

Il comando `systemctl isolate` si rende utile per passare tra un target e l'altro. Quindi, per passare manualmente al target `multi-user`:

```
# systemctl isolate multi-user.target
```

Esistono target corrispondenti ai runlevel SysV, a partire da `runlevel0.target` fino a `runlevel6.target`. Comunque, `systemd` non usa il file `/etc/inittab`. Per cambiare la destinazione di sistema predefinita, l'opzione `systemd.unit` può essere aggiunta all'elenco dei parametri del kernel. Per esempio, per usare `multi-user.target` come destinazione standard, il parametro del kernel dovrebbe essere `systemd.unit = multi-user.target`. Tutti i parametri del kernel possono essere resi persistenti modificando la configurazione del bootloader.

Un altro modo per cambiare la destinazione predefinita è modificare il collegamento simbolico `/etc/systemd/system/default.target` in modo che punti alla destinazione desiderata. La ridefinizione del collegamento può essere fatta automaticamente attraverso il comando `systemctl`:

```
# systemctl set-default multi-user.target
```

Allo stesso modo, puoi determinare quale sia la destinazione di avvio predefinita del tuo sistema con il seguente comando:

```
$ systemctl get-default  
graphical.target
```

Similmente ai sistemi che adottano SysV, la destinazione predefinita non dovrebbe mai puntare a `shutdown.target`, poiché corrisponde al runlevel 0 (shutdown).

I file di configurazione associati ad ogni unità si trovano nella directory `/lib/systemd/system/`. Il comando `systemctl list-unit-files` elenca tutte le unità disponibili e mostra se sono abilitate all'avvio del sistema. L'opzione `--type` selezionerà solo le unità per un dato tipo, come in `systemctl list-unit-files --type=service` e `systemctl list-unit-files --type=target`.

Le unità attive o che sono state attive durante la sessione di sistema corrente possono essere elencate con il comando `systemctl list-units`. Come l'opzione `list-unit-files`, il comando `systemctl list-units --type=service` mostrerà solo le unità di tipo `service` mentre il comando `systemctl list-units --type=target` lo farà solo con le unità di tipo "target".

`systemd` è anche responsabile dell'attivazione e della risposta agli eventi correlati all'alimentazione. Il comando `systemctl suspend` metterà il sistema in modalità di sospensione (a basso consumo), mantenendo i dati correnti in memoria. Il comando `systemctl hibernate` copierà tutti i dati della memoria su disco, quindi lo stato corrente del sistema può essere ripristinato anche dopo essere stato spento. Le azioni associate a tali eventi sono definite nel file `/etc/systemd/logind.conf` o in file separati all'interno della directory `/etc/systemd/logind.conf.d/`. Tuttavia, questa funzione di `systemd` può essere usata solo quando non c'è nessun altro power manager in esecuzione nel sistema, come il demone `acpid`. Il demone `acpid` è il principale *power manager* per Linux e consente regolazioni più precise delle azioni a seguito di eventi relativi all'alimentazione, come la chiusura del coperchio del laptop, la batteria scarica o gli stati di carica della batteria.

Upstart

Gli script di inizializzazione utilizzati da Upstart si trovano nella directory `/etc/init/`. I servizi di sistema possono essere elencati con il comando `initctl list`, che mostra anche lo stato corrente dei servizi e, se disponibile, il loro identificativo PID.

```
# initctl list  
avahi-cups-reload stop/waiting  
avahi-daemon start/running, process 1123  
mountall-net stop/waiting  
mountnfs-bootclean.sh start/running
```

```
nmbd start/running, process 3085
passwd stop/waiting
rc stop/waiting
rsyslog start/running, process 1095
tty4 start/running, process 1761
udev start/running, process 1073
upstart-udev-bridge start/running, process 1066
console-setup stop/waiting
irqbalance start/running, process 1842
plymouth-log stop/waiting
smbd start/running, process 1457
tty5 start/running, process 1764
failsafe stop/waiting
```

Ogni azione Upstart ha il suo comando indipendente. Per esempio, il comando `start` può essere utilizzato per avviare un sesto terminale virtuale:

```
# start tty6
```

Lo stato corrente di una risorsa può essere verificato con il comando `status`:

```
# status tty6
tty6 start/running, process 3282
```

E l'interruzione di una risorsa può essere eseguita con il comando `stop`:

```
# stop tty6
```

Upstart non usa il file `/etc/inittab` per definire i runlevel, ma i comandi legacy `runlevel` e `telinit` possono ancora essere usati per verificare e alternare i vari runlevel.

NOTE

Upstart è stato sviluppato per la distribuzione Ubuntu Linux per facilitare l'avvio parallelo dei processi. Ubuntu ha smesso di usare Upstart dal 2015 quando è passato da Upstart a systemd.

Spegnimento e Riavvio

Un comando molto utilizzato per arrestare o riavviare il sistema è sorprendentemente chiamato `shutdown`. Il comando `shutdown` aggiunge funzioni extra al processo di spegnimento: avvisa automaticamente tutti gli utenti che hanno effettuato l'accesso con un messaggio di avviso nelle

loro sessioni di shell e vengono impediti nuovi accessi. Il comando `shutdown` funge da intermediario per le procedure SysV o `systemd`, ovvero esegue l'azione richiesta chiamando l'azione corrispondente nel gestore servizi adottato dal sistema.

Dopo l'esecuzione di `shutdown`, tutti i processi ricevono un segnale `SIGTERM`, seguito dal segnale `SIGKILL`, quindi il sistema si spegne o cambia il suo runlevel. Per impostazione predefinita, quando non vengono utilizzate le opzioni `-h` o `-r`, il sistema si alterna al runlevel 1, ovvero alla modalità utente singolo. Per modificare le opzioni predefinite per `shutdown`, il comando dovrebbe essere eseguito con la seguente sintassi:

```
$ shutdown [option] time [message]
```

È richiesto solo il parametro `time`. Quest'ultimo definisce quando verrà eseguita l'azione richiesta, accettando i seguenti formati:

hh:mm

Questo formato specifica il tempo di esecuzione come ora e minuti.

+m

Questo formato specifica quanti minuti attendere prima dell'esecuzione.

now or +0

Questo formato determina l'esecuzione immediata.

Il parametro `message` è il testo di avviso inviato a tutte le sessioni del terminale degli utenti che hanno effettuato l'accesso.

L'implementazione di SysV consente di limitare gli utenti che saranno in grado di riavviare la macchina premendo `Ctrl` + `Alt` + `Canc`. Questo è possibile posizionando l'opzione `-a` per il comando `shutdown` presente sulla riga riguardante `ctrlaltdel` nel file `/etc/inittab`. In questo modo, solo gli utenti i cui nomi utente si trovano nel file `/etc/shutdown.allow` saranno in grado di riavviare il sistema con la combinazione di tasti `Ctrl` + `Alt` + `Canc`.

Il comando `systemctl` può anche essere usato per spegnere o riavviare la macchina nei sistemi che utilizzano `systemd`. Per riavviare il sistema, si può usare il comando `systemctl reboot`. Per spegnere il sistema, si deve usare invece il comando ``systemctl poweroff``. Entrambi i comandi richiedono per l'esecuzione i privilegi di root, poiché gli utenti ordinari non possono eseguire tali procedure.

NOTE

Alcune distribuzioni Linux collegheranno `poweroff` e `reboot` a `systemctl` come singoli comandi. Per esempio:

```
$ sudo which poweroff
/usr/sbin/poweroff
$ sudo ls -l /usr/sbin/poweroff
lrwxrwxrwx 1 root root 14 Aug 20 07:50 /usr/sbin/poweroff -> /bin/systemctl
```

Non tutte le attività di manutenzione richiedono che il sistema sia spento o riavviato. Tuttavia, quando è necessario passare allo stato del sistema in modalità utente singolo, è importante avvisare gli utenti che hanno effettuato l'accesso in modo che non vengano danneggiati da una brusca interruzione.

Simile a quello che fa il comando `shutdown` quando si spegne o si riavvia il sistema, il comando `wall` è in grado di inviare un messaggio alle sessioni terminale di tutti gli utenti che hanno effettuato l'accesso. Per fare ciò, l'amministratore di sistema deve solo fornire un file o scrivere direttamente il messaggio come parametro del comando `wall`.

Esercizi Guidati

1. Come si può usare il comando `telinit` per riavviare il sistema?

2. Cosa accadrà ai servizi relativi al file `/etc/rc1.d/K90network` quando il sistema entra nel runlevel 1?

3. Usando il comando `systemctl`, come può un utente verificare se l'unità `sshd.service` è in esecuzione?

4. In un sistema basato su `systemd`, quale comando deve essere eseguito per abilitare l'attivazione dell'unità `sshd.service` durante l'avvio del sistema?

Esercizi Esplorativi

1. In un sistema basato su SysV, supponiamo che il runlevel predefinito indicato in `/etc/inittab` sia 3, ma il sistema si avvii sempre nel runlevel 1. Qual è la causa più probabile?
2. Sebbene il file `/sbin/init` possa essere trovato nei sistemi basati su systemd, è solo un collegamento simbolico a un altro file eseguibile. In tali sistemi, qual è il file indicato da `/sbin/init`?
3. Come può essere verificato il target di sistema predefinito in un sistema basato su systemd?
4. Come si può cancellare un riavvio del sistema programmato con il comando `shutdown`?

Sommario

Questa lezione tratta le principali utility utilizzate come gestori di servizi dalle distribuzioni Linux. Le utility SysVinit, systemd e Upstart hanno ciascuna il proprio approccio al controllo dei servizi e degli stati del sistema. La lezione affronta i seguenti argomenti:

- Quali sono i servizi di sistema e il loro ruolo nel sistema operativo.
- Concetti e utilizzo di base dei comandi in SysVinit, systemd e Upstart.
- Come avviare, arrestare e riavviare correttamente i servizi di sistema e il sistema stesso.

I comandi e le procedure trattate erano:

- Comandi e file relativi a SysVinit, come `init`, `/etc/inittab` e `telinit`.
- Il comando principale di systemd: `systemctl`.
- Comandi Upstart: `initctl`, `status`, `start`, `stop`.
- Comandi tradizionali di gestione, come `shutdown`, e `wall`.

Risposte agli Esercizi Guidati

1. Come si può usare il comando `telinit` per riavviare il sistema?

Il comando `telinit 6` passerà il sistema a runlevel 6, ovvero riavvia il sistema.

2. Cosa accadrà ai servizi relativi al file `/etc/rc1.d/K90network` quando il sistema entra nel runlevel 1?

A causa della lettera "K" all'inizio del nome del file, i relativi servizi verranno interrotti.

3. Usando il comando `systemctl`, come può un utente verificare se l'unità `sshd.service` è in esecuzione?

Con il comando `systemctl status sshd.service` o `systemctl is-active sshd.service`.

4. In un sistema basato su `systemd`, quale comando deve essere eseguito per abilitare l'attivazione dell'unità `sshd.service` durante l'avvio del sistema?

Il comando `systemctl enable sshd.service`, eseguito da un'utenza con i privilegi di root.

Risposte agli Esercizi Guidati

1. In un sistema basato su SysV, supponiamo che il runlevel predefinito indicato in `/etc/inittab` sia 3, ma il sistema si avvia sempre nel runlevel 1. Qual è la causa più probabile?

I parametri 1 o S possono essere presenti nell'elenco dei parametri del kernel.

2. Sebbene il file `/sbin/init` possa essere trovato nei sistemi basati su systemd, è solo un collegamento simbolico a un altro file eseguibile. In tali sistemi, qual è il file indicato da `/sbin/init`?

Il comando principale di systemd: `/lib/systemd/systemd`.

3. Come può essere verificato il target di sistema predefinito in un sistema basato su systemd?

Il collegamento simbolico `/etc/systemd/system/default.target` punterà al file unit indicato come target predefinito. Può anche essere usato il comando `systemctl get-default`.

4. Come si può cancellare un riavvio del sistema programmato con il comando `shutdown`?

Dovrebbe essere usato il comando `shutdown -c`.



**Linux
Professional
Institute**

Argomento 102: Installazione di Linux e Gestione dei Pacchetti



102.1 Progettare il layout del disco rigido

Obiettivi LPI di riferimento

LPIC-1 v5, Exam 101, Objective 102.1

Peso

2

Aree di Conoscenza Chiave

- Allocare i filesystem e lo spazio di swap su partizioni o dischi separati.
- Adattare il partizionamento del disco all'uso previsto del sistema.
- Assicurarsi che la partizione /boot sia conforme ai requisiti dell'architettura hardware per l'avvio.
- Conoscenza delle caratteristiche di base di LVM.

Di seguito è riportato un elenco parziale dei file, dei termini e dei comandi utilizzati

- filesystem / (root)
- filesystem /var
- filesystem /home
- filesystem /boot
- EFI System Partition (ESP)
- spazio di swap
- punti di montaggio
- partizioni



102.1 Lezione 1

Certificazione:	LPIC-1
Versione:	5.0
Argomento:	102 L'Installazione di Linux e la Gestione Pacchetti
Obiettivo:	102.1 Progettare il Layout del Disco Rigido
Lezione:	1 di 1

Introduzione

Per completare questa sezione, è necessario comprendere la relazione tra *dischi*, *partizioni*, *filesystem* e *volumi*.

Pensa a un disco (o *dispositivo di archiviazione*, poiché i dispositivi moderni non contengono alcun “disco”) come a un “contenitore fisico” per i tuoi dati.

Prima che un disco possa essere utilizzato da un computer, deve essere *partizionato*. Una partizione è un sottoinsieme logico del disco fisico, come una “recinzione” logica. Il partizionamento è un modo per “compartimentare” le informazioni memorizzate sul disco, separando, per esempio, i dati del sistema operativo dai dati dell’utente.

Ogni disco ha bisogno di almeno una partizione, ma può avere più partizioni e le informazioni su di esse sono archiviate in una *tabella delle partizioni*. Questa tabella include informazioni sul primo e sull’ultimo settore della partizione e sul suo tipo, nonché ulteriori dettagli su ciascuna partizione.

All’interno di ogni partizione c’è un *filesystem*. Il filesystem descrive il modo in cui le informazioni

sono effettivamente memorizzate sul disco. Queste informazioni includono come sono organizzate le directory, qual è la relazione tra loro, dove sono i dati per ciascun file, ecc.

Le partizioni non possono estendersi su più dischi. Ma utilizzando il *Logical Volume Manager* (LVM) è possibile combinare più partizioni, anche su più dischi, per formare un singolo volume logico.

I volumi logici astraggono i limiti dei dispositivi fisici e lasciano il tuo lavoro con “insiemi” di spazio su disco che possono essere combinati o distribuiti in un modo molto più flessibile rispetto alle partizioni tradizionali. LVM è utile in situazioni in cui è necessario aggiungere più spazio a una partizione senza dover migrare i dati su un dispositivo più grande.

In questa sezione imparerai come progettare uno schema di partizionamento del disco per un sistema Linux, allocando filesystem e spazio di swap per separare partizioni o dischi quando necessario.

Come *creare* e *gestire* partizioni e filesystem verranno trattato in altre lezioni. In questa sezione, viene fornita una panoramica generale di LVM , senza però entrare nei dettagli operativi.

Punti di Montaggio

Prima di poter accedere a un filesystem su Linux, questo deve necessariamente essere *montato*. Ciò significa collegare il filesystem a un punto specifico nella struttura di directory del sistema, chiamato *punto di montaggio*.

Una volta montato, il contenuto del filesystem sarà disponibile sotto il punto di montaggio. Per esempio immagina di avere una partizione con i dati personali dei tuoi utenti (le loro home directory), contenenti le directory `/john`, `/jack` e `/carol`. Se montate all'interno di `/home`, il contenuto di tali directory sarà disponibile in `/home/john`, `/home/jack` e `/home/carol`.

Il punto di montaggio deve esistere prima di montare il filesystem. Per esempio non è possibile montare una partizione in `/mnt/userdata` se questa directory non esiste. Tuttavia, se la directory esiste e contiene file, tali file non saranno disponibili finché non si smonta il filesystem. Se si elenca il contenuto della directory, si vedranno i file memorizzati sul filesystem montato e non i contenuti originali della directory.

I filesystem possono essere montati ovunque si voglia. Tuttavia, ci sono alcune buone pratiche da seguire per facilitare l'amministrazione del sistema.

Tradizionalmente `/mnt` era la directory in cui sarebbero stati montati tutti i dispositivi esterni e un numero di *punti di montaggio* preconfigurati per dispositivi comuni, come unità CD-ROM (`/mnt/cdrom`) e floppy disk (`/mnt/floppy`).

Questo è stato sostituito da `/media`, che ora è il punto di montaggio predefinito per qualsiasi supporto rimovibile (per es. dischi esterni, unità flash USB, schede di memoria, dischi ottici, ecc.) collegati al sistema.

Sulla maggior parte delle moderne distribuzioni Linux in ambienti desktop, i dispositivi rimovibili vengono montati automaticamente su `/media/USER/LABEL` quando sono collegati al sistema, dove `USER` è il nome utente e `LABEL` è l'etichetta del dispositivo. Per esempio, flash USB con l'etichetta `FlashDrive` collegata da `john` verrebbe montata sotto `/media/john/FlashDrive/`. Il modo in cui ciò viene gestito può dipendere dall'ambiente desktop utilizzato.

Detto questo, ogni volta che è necessario montare *manualmente* un filesystem, è buona norma montarlo in `/mnt`. I comandi specifici per effettuare il montaggio e lo smontaggio dei filesystem su Linux saranno discussi in un'altra lezione.

Tenere le Cose Separate

Su Linux, ci sono alcune directory che dovresti considerare di tenere su partizioni separate. Ci sono molte ragioni per questo: per esempio, mantenendo i file relativi al *bootloader* (memorizzati su `/boot`) su una *partizione di avvio*, si garantisce che il sistema sarà ancora in grado di avviarsi in caso di un arresto anomalo del *root filesystem*.

Mantenere le directory personali dell'utente (in `/home`) su una partizione separata semplifica la reinstallazione del sistema senza il rischio di toccare accidentalmente i dati dell'utente. Mantenere i dati relativi a un server Web o di database (di solito in `/var`) su una partizione separata (o anche su un disco separato) semplifica l'amministrazione del sistema nel caso in cui sia necessario aggiungere più spazio disco per questi casi d'uso.

Ci possono essere anche motivi di prestazioni per mantenere determinate directory su partizioni separate. Potresti voler mantenere il filesystem di root (`/`) su un'unità SSD veloce e directory più grandi come `/home` e `/var` su dischi rigidi più lenti che offrono però molto più spazio ad una frazione del costo.

La Partizione di Avvio (`/boot`)

La partizione di avvio contiene i file utilizzati dal bootloader per caricare il sistema operativo. Sui sistemi Linux il bootloader è generalmente *GRUB2* o, su sistemi più vecchi, *GRUB Legacy*. La partizione è di solito montata in `/boot` e i suoi file sono memorizzati in `/boot/grub`.

Tecnicamente non è necessaria una partizione di avvio, poiché nella maggior parte dei casi GRUB può montare la partizione di root (`/`) e caricare i file da una directory `/boot` separata.

Tuttavia è possibile che si desideri una partizione di avvio separata per motivi di sicurezza (assicurandosi che il sistema si avvii anche in caso di crash del *root filesystem*) o se si desidera utilizzare un filesystem che il bootloader non può comprendere nella partizione di root o se utilizza un tipo di crittografia e/o di compressione non supportata.

La partizione di avvio è in genere la prima partizione sul disco. Questo perché all'inizio il BIOS dei PC IBM indirizzava i dischi utilizzando *cilindri(cylinder)*, *testine(head)* e *settori(sectors)* (CHS), con un massimo di 1024 cilindri, 256 testine e 63 settori, con una dimensione massima del disco di 528 MB (504 MB in MS-DOS) . Ciò significa che qualsiasi cosa oltre questo spazio non sarebbe accessibile sui sistemi *legacy*, a meno che non fosse utilizzato un diverso schema di indirizzamento del disco (come il *Logical Block Addressing*, LBA).

Quindi, per garantire la massima compatibilità, la partizione di avvio si trova di solito all'inizio del disco e termina prima del cilindro 1024 (528 MB), assicurando così che la macchina sia sempre in grado di caricare il kernel.

Poiché la partizione di avvio memorizza solo i file necessari al bootloader, il disco RAM iniziale (*initrd*) e le immagini del kernel, può rimanere di dimensioni piuttosto ristrette rispetto agli standard odierni. Una buona dimensione è di circa 300 MB.

La Partizione di Sistema EFI (ESP)

L'*EFI System Partition* (ESP) viene utilizzata dalle macchine basate su *Unified Extensible Firmware Interface* (UEFI) per memorizzare boot loader e immagini del kernel per i sistemi operativi installati.

Questa partizione è formattata in un filesystem basato su FAT. Su un disco partizionato con una tabella delle partizioni GUID ha un identificatore univoco globale di C12A7328-F81F-11D2-BA4B-00A0C93EC93B. Se il disco è stato formattato invece con lo schema di partizionamento MBR, l'ID della partizione sarà 0xEF.

Sui computer che eseguono Microsoft Windows questa partizione è in genere la prima sul disco, sebbene non sia strettamente necessario. L'ESP viene creata (o popolata) dal sistema operativo al momento dell'installazione. Su un sistema Linux è montata in `/boot/efi`.

La Partizione `/home`

Ogni utente nel sistema ha una *home directory* home per memorizzare i file e le preferenze personali, e la maggior parte di essi si trova in `/home`. Di solito la home directory è la stessa del nome utente, quindi l'utente John avrà la sua directory in `/home/john`.

Tuttavia ci sono eccezioni. Per esempio la home directory per l'utente root è `/root` e alcuni servizi

di sistema potrebbero avere utenti associati con home directory posizionate altrove.

Non esiste una regola per determinare la dimensione di una partizione per la directory `/home` (la partizione *home*). È necessario tenere conto del numero di utenti nel sistema e di come verrà utilizzato. Per esempio, un utente che esegue solo la navigazione Web ed elaborazione dei testi richiederà meno spazio di uno che lavora con l'editing video.

Dati variabili (`/var`)

Questa directory contiene “dati variabili” o file e directory su cui il sistema deve poter scrivere durante il funzionamento. Ciò include i log di sistema (`/var /log`), i file temporanei (`/var/tmp`) e i dati delle applicazioni memorizzate nella cache (in `/var/cache`).

`/var/www/html` è anche la directory predefinita per i file dei dati del Web Server Apache e `/var/lib/mysql` è la posizione predefinita per i file di database per il server MySQL. Tuttavia, entrambi questi possono essere modificati.

Un buon motivo per mettere `/var` in una partizione separata è la *stabilità*. Molte applicazioni e processi scrivono su `/var` e relative sottodirectory, come `/var/log` o `/var/tmp`. Un processo non corretto può scrivere dati fino a quando non c'è più spazio libero sul filesystem.

Se `/var` è in `/`` ciò può innescare un evento di kernel panic e la corruzione del filesystem, causando una situazione difficilmente recuperabile. Ma se `/var` è tenuto in una partizione separata, il filesystem di root non sarà influenzato.

Come in `/home`, non esiste una regola universale per determinare la dimensione di una partizione per `/var`, poiché varierà al variare del modo in cui viene utilizzato il sistema. Su un sistema domestico, potrebbero essere necessari solo pochi gigabyte. Ma su un database o un server Web potrebbe essere necessario molto più spazio. In tali scenari, può essere saggio mettere `/var` su una partizione su un disco diverso rispetto alla partizione root aggiungendo un ulteriore livello di protezione contro i guasti del disco fisico.

Lo Swap

La partizione di *swap* viene utilizzata per scambiare pagine di memoria dalla RAM al disco secondo necessità. Questa partizione deve essere di un tipo specifico e inizializzata con un'utilità adeguata chiamata `mkswap` prima di poter essere utilizzata.

La partizione di swap non può essere montata come le altre, il che significa che non è possibile accedervi come una normale directory e dare un'occhiata al suo contenuto.

Un sistema può avere più partizioni di swap (anche se questo non è comune) e Linux supporta

anche l'uso di *file* di swap invece di partizioni, che possono essere utili per aumentare rapidamente lo spazio di swap quando necessario.

La dimensione della partizione di swap è un problema controverso. La vecchia regola dei primi tempi di Linux: “due volte la quantità di RAM” potrebbe non essere più applicabile a seconda di come viene utilizzato il sistema e della quantità di RAM fisica installata.

Nella documentazione di Red Hat Enterprise Linux 7, Red Hat consiglia quanto segue:

Quantità di RAM	Dimensione Swap Raccomandata	Dimensione Swap Raccomandata con Ibernazione
< 2 GB di RAM	2x la quantità di RAM	3x la quantità di RAM
2-8 GB di RAM	Uguale alla quantità di RAM	2x la quantità di RAM
8-64 GB di RAM	Almeno 4 GB	1.5x la quantità di RAM
> 64 GB di RAM	Almeno 4 GB	Non raccomandata

Naturalmente la quantità di swap può dipendere dal carico di lavoro. Se la macchina esegue un servizio critico, come un database, un server Web o SAP, è consigliabile controllare la documentazione di questi servizi (o il fornitore del software) per le raccomandazioni specifiche.

NOTE

Per ulteriori informazioni sulla creazione e l'abilitazione di partizioni e file di swap, vedere la sezione 104.1 dell' LPIC-1.

LVM

Abbiamo già discusso di come i dischi sono organizzati in una o più partizioni, con ciascuna partizione contenente un filesystem che descrive come sono memorizzati i file e i metadati associati. Uno degli svantaggi del partizionamento è che l'amministratore di sistema deve decidere in anticipo come verrà distribuito lo spazio su disco disponibile su un dispositivo. Ciò può presentare alcune sfide in seguito, se una partizione richiede più spazio di quanto inizialmente previsto. Naturalmente le partizioni possono essere ridimensionate, ma ciò potrebbe non essere possibile se, per esempio, non c'è ulteriore spazio libero sul disco.

Logical Volume Management (LVM) è una forma di virtualizzazione dello storage che offre agli amministratori di sistema un approccio più flessibile alla gestione dello spazio su disco rispetto al partizionamento tradizionale. L'obiettivo di LVM è facilitare la gestione delle esigenze di archiviazione. L'unità base è il *Physical Volume* (PV), che è un dispositivo a blocchi sul sistema come per esempio una partizione del disco o un array RAID.

I PV sono raggruppati in *Volume Group* (VG) che astraggono i dispositivi sottostanti e sono visti come un singolo dispositivo logico, con la capacità di archiviazione combinata dei vari PV.

Ogni volume in un *Volume Group* è suddiviso in parti di dimensioni fisse chiamati *extents*. Tali parti su un PV sono denominate *Physical Extents* (PE), mentre quelle su un volume logico sono chiamate *Logical Extents* (LE). In genere, ogni estensione logica è mappata su un'estensione fisica, ma ciò può cambiare se vengono utilizzate funzionalità come il *mirroring* del disco.

I *Volume Groups* possono essere suddivisi in *Logical Volumes* (LV), che funzionano in modo simile alle partizioni ma con maggiore flessibilità.

La dimensione di un Logical Volume, come specificato durante la sua creazione, è in effetti definita dalla dimensione delle estensioni (extents) fisiche (4 MB per impostazione predefinita) moltiplicata per il numero di estensioni sul volume. Da questo è facile capire che per far crescere un Logical Volume, per esempio, tutto ciò che l'amministratore di sistema deve fare è aggiungere ulteriori estensioni dal pool disponibile nel Volume Group. Allo stesso modo, le estensioni possono essere rimosse per ridurre l'LV.

Dopo la creazione di un Logical Volume, quest'ultimo viene visualizzato dal sistema operativo come un normale dispositivo a blocchi. Una nuova risorsa verrà creata in `/dev`, chiamata come `/dev/VGNAME/LVNAME`, dove `VGNAME` è il nome del Volume Group e `LVNAME` è il nome del Logical Volume.

Questi dispositivi possono essere formattati con un filesystem a scelta usando utility standard (come per esempio `mkfs.ext4`) e montati manualmente con il comando `mount` o automaticamente aggiungendoli al file `/etc/fstab`.

Esercizi Guidati

1. Sui sistemi Linux, dove sono archiviati i file per il bootloader GRUB?

2. Dove dovrebbe terminare la partizione di avvio per garantire che un PC sia sempre in grado di caricare il kernel?

3. Dove si trova di solito la partizione EFI?

4. Quando si monta manualmente un filesystem, in quale directory dovrebbe essere montato di solito?

Esercizi Esplorativi

1. Qual è l'unità più piccola all'interno di un Volume Group?
2. Come viene definita la dimensione di un Logical Volume?
3. Su un disco formattato con lo schema di partizionamento MBR, qual è l'ID della partizione di sistema EFI?
4. Oltre che con partizioni, come è possibile aumentare rapidamente lo spazio di swap su un sistema Linux?

Sommario

In questa lezione abbiamo appreso i concetti fondamentali del partizionamento, quali directory di solito sono tenute in partizioni separate e il perché di ciò. Inoltre, abbiamo fatto una panoramica su LVM (Logical Volume Management) e sul come può offrire un modo più flessibile di allocare i dati e lo spazio su disco rispetto al partizionamento tradizionale.

Sono stati discussi i seguenti file, termini e utilità:

/

La radice principale del file system Linux.

/var

Il percorso standard per i “dati variabili”, dati che possono sia ridursi e crescere nel tempo.

/home

La directory principale standard per le home degli utenti su un sistema.

/boot

Il percorso standard per i file del bootloader, il kernel Linux e il disco RAM iniziale (*Initrd*).

EFI System Partition (ESP)

Utilizzato dai sistemi che hanno implementato UEFI per l'archiviazione dei file di avvio del sistema.

Spazio swap

Utilizzato per scambiare pagine di memoria del kernel quando la RAM è sovra utilizzata.

Punti di montaggio

Posizioni della directory in cui verrà montato un dispositivo (come un disco rigido).

Partizioni

Spazi di sezionamento logico di un disco rigido

Risposte agli Esercizi Guidati

1. Sui sistemi Linux, dove sono archiviati i file per il bootloader GRUB?

All'interno di `/boot/grub`.

2. Dove dovrebbe terminare la partizione di avvio per garantire che un PC sia sempre in grado di caricare il kernel?

Prima del cilindro 1024.

3. Dove si trova di solito la partizione EFI?

All'interno di `/boot/efi`.

4. Quando si monta manualmente un filesystem, in quale directory dovrebbe essere montato di solito?

All'interno di `/mnt`. Tuttavia, questo non è obbligatorio. È possibile montare una partizione in *qualsiasi* directory desiderata.

Risposte agli Esercizi Guidati

1. Qual è l'unità più piccola all'interno di un Volume Group?

I Volume Group sono suddivisi in estensioni (extents).

2. Come viene definita la dimensione di un Logical Volume?

Attraverso la dimensione delle estensioni fisiche (PE) moltiplicata per il numero di estensioni sul volume.

3. Su un disco formattato con lo schema di partizionamento MBR, qual è l'ID della partizione di sistema EFI?

L'ID è `0xEF`.

4. Oltre che con partizioni, come è possibile aumentare rapidamente lo spazio di swap su un sistema Linux?

Si possono utilizzare i file di swap.



Linux
Professional
Institute

102.2 Installare un boot manager

Obiettivi LPI di riferimento

LPIC-1 v5, Exam 101, Objective 102.2

Peso

2

Aree di Conoscenza Chiave

- Fornire percorsi di avvio alternativi e opzioni di avvio di backup.
- Installare e configurare un boot loader come GRUB Legacy.
- Eseguire modifiche alla configurazione di base di GRUB 2.
- Interagire con il boot loader.

Di seguito è riportato un elenco parziale dei file, dei termini e dei comandi utilizzati

- `menu.lst`, `grub.cfg` e `grub.conf`
- `grub-install`
- `grub-mkconfig`
- MBR



102.2 Lezione 1

Certificazione:	LPIC-1
Versione:	5.0
Argomento:	102 L'Installazione di Linux e la Gestione Pacchetti
Obiettivo:	102.2 Installare un Boot Manager
Lezione:	1 di 1

Introduzione

Quando un computer è avviato, il primo software da eseguire è il boot loader. Questo è un "pezzo" di codice il cui unico scopo è caricare un kernel e consegnargli il controllo. Il kernel caricherà i driver necessari, inizializzerà l'hardware e quindi caricherà il resto del sistema operativo.

GRUB è il *boot loader* utilizzato sulla maggior parte delle distribuzioni Linux. Può caricare il kernel Linux o altri sistemi operativi, come Windows, e può gestire più immagini e parametri del kernel attraverso voci di menu separate. La selezione del kernel all'avvio viene eseguita tramite un'interfaccia basata su tastiera e esiste un'interfaccia da riga di comando per la modifica delle opzioni e dei parametri di avvio.

La maggior parte delle distribuzioni Linux installa e configura GRUB (in realtà, GRUB 2) automaticamente, quindi un normale utente non deve preoccuparsene. Tuttavia, come amministratore di sistema, è fondamentale sapere come controllare il processo di avvio in modo da poter ripristinare il sistema dopo, per esempio, un aggiornamento del kernel non riuscito che impedisce la corretta partenza dello stesso.

In questa lezione imparerai a installare, configurare e interagire con GRUB.

GRUB Legacy e GRUB 2

La versione originale di GRUB (*Grand Unified Bootloader*), ora nota come *GRUB Legacy*, è stata sviluppata nel 1995 come parte del progetto GNU Hurd, e in seguito è stata adottata come boot loader di avvio predefinito di molte distribuzioni Linux, sostituendo precedenti alternative come per esempio LILO.

GRUB 2 è una riscrittura completa di GRUB che mira a essere più pulita, più sicura, più robusta e più potente. Tra i numerosi vantaggi rispetto a GRUB Legacy vi sono un file di configurazione molto più flessibile (con molti più comandi e istruzioni condizionali, simili a un linguaggio di scripting), un design più modulare e una migliore localizzazione/internazionalizzazione.

C'è anche il supporto per temi e menu di avvio grafici, la possibilità di avviare LiveCD ISO direttamente dal disco rigido, un supporto migliore per architetture non x86, supporto universale per UUID (che semplifica l'identificazione di dischi e partizioni) e molto di più.

GRUB Legacy non è più in fase di sviluppo attivo (l'ultima versione è la 0.97 del 2005), e oggi la maggior parte delle principali distribuzioni Linux installa GRUB 2 come boot loader predefinito. Tuttavia, potresti comunque trovare sistemi che utilizzano GRUB Legacy, quindi è importante sapere come usarlo e conoscere le differenze rispetto a GRUB 2.

Dove è il Bootloader?

Storicamente, i dischi rigidi su sistemi compatibili PC IBM sono stati partizionati utilizzando lo schema MBR, creato nel 1982 per IBM PC-DOS (MS-DOS) 2.0.

In questo schema, il primo settore (512 byte) è chiamato *Master Boot Record* e contiene una tabella che descrive le partizioni sul disco (la tabella delle partizioni) e anche il codice bootstrap, chiamato bootloader.

All'accensione del computer, questo codice minimale di bootloader (a causa delle restrizioni di dimensione) viene caricato, eseguito e passa il controllo a un boot loader secondario su disco, solitamente situato in uno spazio di 32 KB tra l'MBR e la prima partizione, che caricherà a turno i sistemi operativi.

Su un disco con partizionamento MBR, il codice di avvio per GRUB è installato sull'MBR. Questo carica e passa il controllo a un'immagine "core" installata tra l'MBR e la prima partizione. Da questo punto, GRUB è in grado di caricare il resto delle risorse necessarie (menu di scelta, file di configurazione e moduli extra) dal disco.

Tuttavia, MBR presenta limitazioni sul numero di partizioni (originariamente un massimo di 4 partizioni primarie, successivamente un massimo di 3 partizioni primarie e 1 partizione estesa suddivisa in un numero di partizioni logiche) e dimensioni massime del disco di 2 TB. Per superare queste limitazioni è stato creato un nuovo schema di partizionamento chiamato GPT (*GUID Partition Table*), parte dello standard UEFI (*Unified Extensible Firmware Interface*).

I dischi con partizionamento GPT possono essere utilizzati con computer con il firmware BIOS tradizionale o con firmware UEFI. Su macchine con BIOS, la seconda parte di GRUB è memorizzata in una partizione speciale di avvio del BIOS.

Sui sistemi con firmware UEFI, GRUB viene caricato dal firmware dai file `grubia32.efi` (per sistemi a 32 bit) o `grubx64.efi` (per sistemi a 64 bit) da una partizione chiamata ESP (*EFI System Partition*).

La Partizione /boot

Su Linux i file necessari per il processo di avvio sono generalmente memorizzati su una partizione avviabile, montati all'interno del file system di root in `/boot`.

Una partizione di avvio specifica non è necessaria sui sistemi attuali, in quanto i boot loader come GRUB possono in genere montare il file system di root e cercare i file necessari all'interno di una directory `/boot`, ma rimane in ogni caso una buona pratica in quanto separa i file necessari per il processo di avvio dal resto del filesystem.

La partizione di avvio è in genere la prima partizione sul disco. Questo perché il BIOS dei PC IBM originali indirizzava i dischi utilizzando *cilindri(cylinder)*, *testine(head)* e *settori(sectors)* (CHS), con un massimo di 1024 cilindri, 256 testine e 63 settori, con una dimensione massima del disco di 528 MB (504 MB in MS-DOS). Ciò significa che qualsiasi cosa oltre questo spazio non sarebbe accessibile sui sistemi legacy, a meno che non fosse utilizzato un diverso schema di indirizzamento del disco (come il *Logical Block Addressing*, LBA)..

Quindi, per la massima compatibilità, la partizione di avvio si trova di solito all'inizio del disco e termina prima del cilindro 1024 (528 MB), assicurando che la macchina sarà sempre in grado di caricare il kernel. Le dimensioni consigliate per questa partizione, su una macchina attuale, sono di 300 MB circa.

Altri motivi per una partizione `/boot` separata sono la crittografia e la compressione poiché alcuni metodi potrebbero non essere ancora supportati da GRUB 2 o se è necessario formattare la partizione root di sistema (`/`) usando un file system non supportato.

Contenuto della Partizione di Boot

Il contenuto della partizione `/boot` può variare a seconda dell'architettura del sistema o del boot loader in uso, ma su un sistema basato su x86 di solito troverai i file qui all'interno. Molti di questi sono chiamati con il suffisso `-VERSION`, dove `-VERSION` è la versione del corrispondente kernel Linux. Quindi, per esempio, un file di configurazione per la versione del kernel Linux `4.15.0-65-generic` sarebbe chiamato `config-4.15.0-65-generic`.

Config file

Questo file, solitamente chiamato `config-VERSION` (vedi esempio sopra), memorizza i parametri di configurazione per il kernel Linux. Questo file viene generato automaticamente quando viene compilato o installato un nuovo kernel e non deve essere modificato direttamente dall'utente.

System map

Questo file è una tabella che abbina i *symbol names* (come variabili o funzioni) alla posizione corrispondente nella memoria. Ciò è utile quando si esegue il debug di un tipo di errore di sistema noto come *kernel panic*, in quanto consente all'utente di sapere quale variabile o funzione veniva chiamata quando si è verificato l'errore. Come per il config file, il nome è di solito `System.map-VERSION` (per esempio `System.map-4.15.0-65-generic`).

Linux kernel

Questo è il kernel del sistema operativo corretto. Il nome è di solito `vmlinux-VERSION` (per esempio `vmlinux-4.15.0-65-generic`). Si può anche trovare il nome `vmlinuz` invece di `vmlinux`, la *z* alla fine significa che il file è stato compresso.

Initial RAM disk

Questo di solito si chiama `initrd.img-VERSION` e contiene un file system root minimale caricato in un disco RAM, contenente utility e moduli del kernel necessari affinché il kernel possa montare il vero filesystem root.

Boot loader related files

Sui sistemi con GRUB installato, questi si trovano di solito in `/boot /grub` e includono il file di configurazione di GRUB (`/boot/grub/grub.cfg` per GRUB 2 o `/boot/grub/menu.lst` nel caso di GRUB Legacy), moduli (in `/boot/grub/i386-pc`), file di traduzione (in `/boot/grub/locale`) e caratteri (in `/boot/grub/fonts`).

GRUB 2

Installare GRUB 2

GRUB 2 può essere installato usando l'utility `grub-install`. Se si è su un sistema non avviabile, sarà necessario eseguire l'avvio utilizzando una Live CD o un disco di ripristino, scoprire qual è la partizione di avvio per il proprio sistema, montarlo e quindi eseguire l'utilità.

NOTE I comandi seguenti presuppongono che si sia effettuato l'accesso come `root`. Altrimenti, prima esegui `sudo su -` per “diventare” `root`. Al termine, digitare `exit` per disconnettersi e tornare un utente senza privilegi.

Il primo disco su un sistema è di solito il *boot device* e potrebbe essere necessario sapere se sul disco è presente una *boot partition*. Questo può essere fatto con l'utility `fdisk`. Per elencare tutte le partizioni sul primo disco del computer, utilizzare:

```
# fdisk -l /dev/sda
Disk /dev/sda: 111,8 GiB, 120034123776 bytes, 234441648 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x97f8fef5

Device      Boot    Start        End    Sectors    Size Id Type
/dev/sda1   *         2048    2000895    1998848    976M 83 Linux
/dev/sda2             2002942  234440703  232437762  110,9G  5 Extended
/dev/sda5             2002944   18008063   16005120    7,6G 82 Linux swap / Solaris
/dev/sda6             18010112  234440703  216430592  103,2G 83 Linux
```

La partizione di avvio è identificata con `*` nella colonna `Boot`. Nell'esempio qui sopra è `/dev/sda1`.

Ora, crea una directory temporanea sotto `/mnt` e monta la partizione all'interno di essa:

```
# mkdir /mnt/tmp
# mount /dev/sda1 /mnt/tmp
```

Quindi esegui `grub-install`, puntandolo al *boot device* (non la sola partizione) e alla directory in cui è montata la partizione di boot. Se il sistema ha una partizione di avvio dedicata, il comando è:

```
# grub-install --boot-directory=/mnt/tmp /dev/sda
```

Se state installando su un sistema che non ha una partizione di avvio, ma solo una directory `/boot` sul filesystem di root, puntatelo direttamente su `grub-install`. Il comando in questo caso è:

```
# grub-install --boot-directory=/boot /dev/sda
```

Configurare GRUB 2

Il file di configurazione predefinito per GRUB 2 è `/boot/grub/grub.cfg`. Questo file viene generato automaticamente e la modifica manuale non è consigliata. Per apportare modifiche alla configurazione di GRUB, è necessario modificare il file `/etc/default/grub` e quindi eseguire l'utilità `update-grub` per generare un file conforme.

NOTE

`update-grub` è generalmente un collegamento a `grub-mkconfig -o /boot/grub/grub.cfg`, quindi producono gli stessi risultati.

Ci sono alcune opzioni nel file `/etc/default/grub` che controllano il comportamento di GRUB 2, come il kernel predefinito per l'avvio, il timeout, i parametri extra della riga di comando, ecc. I più importanti sono:

GRUB_DEFAULT=

La voce di menu predefinita per l'avvio. Può essere un valore numerico (come `0`, `1`, ecc.), Il nome di una voce di menu (come `debian`) o `salvato`, che viene utilizzato insieme a `GRUB_SAVEDEFAULT =`, spiegato di seguito. Tieni presente che le voci di menu iniziano da zero, quindi la prima voce di menu è `0`, la seconda è `1`, ecc.

GRUB_SAVEDEFAULT=

Se questa opzione è impostata su `true` e `GRUB_DEFAULT =` è impostata su `save`, l'opzione di avvio predefinita sarà sempre l'ultima selezionata nel menu di avvio.

GRUB_TIMEOUT=

Il timeout, in secondi, prima che sia avviata la voce di menu predefinita. Se impostato su `0`, il sistema avvierà la voce predefinita senza mostrare un menu. Se impostato su `-1`, il sistema attenderà fino a quando l'utente selezionerà un'opzione, indipendentemente da quanto tempo impiegherà.

GRUB_CMDLINE_LINUX=

Questo elenca le opzioni che verranno aggiunte come parametri all'avvio del kernel Linux..

GRUB_CMDLINE_LINUX_DEFAULT=

Per impostazione predefinita, vengono generate due voci di menu per ciascun kernel Linux, una con le opzioni predefinite e una voce per il ripristino. Con questa opzione è possibile aggiungere parametri aggiuntivi che verranno aggiunti solo alla voce predefinita.

GRUB_ENABLE_CRYPTODISK=

Se impostato su `y`, comandi come `grub-mkconfig`, `update-grub` e `grub-install` cercheranno dischi crittografati e aggiungeranno i comandi necessari per accedervi durante l'avvio. Ciò disabilita l'avvio automatico (`GRUB_TIMEOUT` = con qualsiasi valore diverso da `-1`) perché è necessaria una *passphrase* per decrittare i dischi prima che sia possibile accedervi.

Gestire le Voci del Menu

Quando viene eseguito `update-grub`, GRUB 2 cerca i kernel e i sistemi operativi sulla macchina e genera le corrispondenti voci di menu sul file `/boot/grub/grub.cfg`. Nuove voci possono essere aggiunte manualmente ai file di script nella directory `/etc/grub.d`.

Questi file devono essere eseguibili e vengono elaborati in ordine numerico da `update-grub`. Pertanto `05_debian_theme` viene elaborato prima di `10_linux` e così via. Le voci di menu personalizzate vengono generalmente aggiunte al file `40_custom`.

La sintassi di base per una voce di menu è mostrata di seguito:

```
menuentry "Default OS" {
    set root=(hd0,1)
    linux /vmlinuz root=/dev/sda1 ro quiet splash
    initrd /initrd.img
}
```

La prima riga inizia sempre con `menuentry` e termina con `}`. Il testo tra virgolette verrà visualizzato come etichetta della voce nel menu di avvio di GRUB 2.

Il parametro `set root` definisce il disco e la partizione in cui si trova il file system radice per il sistema operativo. Nota che su GRUB 2 i dischi sono numerati da zero, quindi `hd0` è il primo disco (`sda` in Linux), `hd1` il secondo e così via. Le partizioni, tuttavia, sono numerate a partire dal numero uno. Nell'esempio sopra, il file system di root si trova sul primo disco (`hd0`), sulla prima partizione (`, 1`) o `sda1`.

Invece di specificare direttamente il dispositivo e la partizione, puoi anche fare in modo che GRUB 2 cerchi un file system con un'etichetta specifica o UUID (*Universally Unique Identifier*). Per questo, usa il parametro `search --set=root` seguito dal parametro `--label` e dall'etichetta del file system da cercare, oppure `--fs-uuid` seguito dall'UUID del file system.

Puoi trovare l'UUID di un filesystem con il comando seguente:

```
$ ls -l /dev/disk/by-uuid/
total 0
lrwxrwxrwx 1 root root 10 nov  4 08:40 3e0b34e2-949c-43f2-90b0-25454ac1595d -> ../../sda5
lrwxrwxrwx 1 root root 10 nov  4 08:40 428e35ee-5ad5-4dc8-adca-539aba6c2d84 -> ../../sda6
lrwxrwxrwx 1 root root 10 nov  5 19:10 56C11DCC5D2E1334 -> ../../sdb1
lrwxrwxrwx 1 root root 10 nov  4 08:40 ae71b214-0aec-48e8-80b2-090b6986b625 -> ../../sda1
```

Nell'esempio sopra, l'UUID per `/dev/sda1` è `ae71b214-0aec-48e8-80b2-090b6986b625`. Se si desidera impostarlo come dispositivo root per GRUB 2, il comando sarebbe `search --set=root --fs-uuid ae71b214-0aec-48e8-80b2-090b6986b625`.

Quando si usa il comando `search` è frequente l'uso del parametro `--no-floppy` in modo che GRUB non perda tempo a cercare dischi floppy.

La linea `linux` indica dove si trova il kernel per il sistema operativo (in questo caso, il file `vmlinuz`` nella root del file system). Successivamente, è possibile passare i parametri della riga di comando al kernel.

Nell'esempio sopra abbiamo specificato la partizione root (`root=/dev/sda1`) e abbiamo passato tre parametri del kernel: la partizione root dovrebbe essere montata in sola lettura (`ro`), la maggior parte dei messaggi di log dovrebbe essere disabilitata (`quiet`) e dovrebbe essere visualizzata una schermata di avvio (`splash`).

La riga `initrd` indica dove si trova il disco RAM iniziale. Nell'esempio sopra il file è `initrd.img`, che si trova nella root del file system.

NOTE

La maggior parte delle distribuzioni Linux non colloca effettivamente il kernel e `initrd` nella directory principale del file system radice. Invece, questi sono collegamenti ai file effettivi all'interno della directory o della partizione `/boot`.

L'ultima riga di una voce di menu deve contenere solo il carattere `}`.

Interagire con GRUB 2

Quando si avvia un sistema con GRUB 2, verrà visualizzato un menu di opzioni. Utilizzare i tasti di

direzione per selezionare un'opzione e `Enter` per confermare e avviare la voce selezionata.

TIP

Se vedete solo un conto alla rovescia, ma non un menu, premete `Shift` per visualizzare il menu.

Per modificare un'opzione, selezionare con i tasti freccia e premere `E`. Questo mostrerà una finestra dell'editor con i contenuti di `menuentry` associati a quell'opzione, come definito in `/boot/grub/grub.cfg`.

Dopo aver modificato un'opzione, digitare `Ctrl` + `X` o `F10` per l'avvio, oppure `Esc` per tornare al menu.

Per entrare nella shell di GRUB 2, premere `c` nella schermata del menu (o `Ctrl` + `c`) nella finestra di modifica). Apparirà un prompt dei comandi come il seguente: `grub>`

Digitare `help` per visualizzare un elenco di tutti i comandi disponibili, oppure premere `Esc` per uscire dalla shell e tornare alla schermata del menu.

NOTE

Ricorda che questo menu non apparirà se `GRUB_TIMEOUT` è impostato su `0` in `/etc/default/grub`.

Avviare dalla Shell di GRUB 2

È possibile utilizzare la shell GRUB 2 per avviare il sistema nel caso in cui una configurazione errata in una voce di menu impedisca l'avvio.

La prima cosa da fare è scoprire dove si trova la partizione di avvio. Potete farlo con il comando `ls`, che mostrerà un elenco delle partizioni e dei dischi trovati da GRUB 2.

```
grub> ls
(proc) (hd0) (hd0,msdos1)
```

Nell'esempio sopra, le cose sono facili. C'è solo un disco `(hd0)` con solo una partizione su di esso: `(hd0,msdos1)`.

Nel nostro esempio la prima partizione `hd0` è chiamata `msdos1` perché il disco è stato partizionato usando lo schema di partizionamento MBR. Se fosse stato partizionato usando GPT, il nome sarebbe stato `gpt1`.

Per avviare Linux, abbiamo bisogno di un kernel e di un disco RAM iniziale (`initrd`). Controlliamo il contenuto di `(hd0, msdos1)`:

```
grub> ls (hd0,msdos1)/
```

```
lost+found/ swapfile etc/ media/ bin/ boot/ dev/ home/ lib/ lib64/ mnt/ opt/ proc/ root/
run/ sbin/ srv/ sys/ tmp/ usr/ var/ initrd.img initrd.img.old vmlinuz cdrom/
```

È possibile aggiungere il parametro `-l` a `ls` per ottenere un elenco lungo, simile a quello che si otterrebbe su un terminale Linux. Usare `Tab` per completare automaticamente i nomi di disco, partizione e file.

Notare che c'è un kernel (`vmlinuz`) e un `initrd` (`initrd.img`) direttamente nella directory `root`. Altrimenti, potremmo controllare il contenuto di `/boot` con `list (hd0,msdos1)/boot/`.

Ora, impostare la partizione di avvio:

```
grub> set root=(hd0,msdos1)
```

Caricare il kernel Linux con il comando `linux`, seguito dal percorso del kernel e dall'opzione `root=` per dire al kernel dove si trova il filesystem di root per il sistema operativo.

```
grub> linux /vmlinuz root=/dev/sda1
```

Caricare il disco RAM iniziale con `initrd`, seguito dal percorso completo del file `initrd.img`:

```
grub> initrd /initrd.img
```

Ora, avviare il sistema con `boot`.

Avviare da una Shell di Ripristino

In caso di errore di avvio, GRUB 2 può caricare una shell di ripristino, una versione semplificata della shell di cui abbiamo parlato in precedenza. Lo riconoscerai dal prompt dei comandi, che viene visualizzato come `grub rescue>`.

Il processo per avviare un sistema da questa shell è quasi lo stesso di quello mostrato in precedenza. Tuttavia, per far funzionare le cose dovremo caricare alcuni moduli GRUB 2.

Dopo aver scoperto quale partizione è quella di avvio (con `ls`, come mostrato prima), usare il comando `set prefix=`, seguito dal percorso completo della directory che contiene i file di GRUB 2. Di solito `/boot/grub`. Nel nostro esempio:

```
grub rescue> set prefix=(hd0,msdos1)/boot/grub
```

Ora, caricate i moduli `normal` e `linux` con il comando `insmod`:

```
grub rescue> insmod normal
grub rescue> insmod linux
```

Quindi, impostare la partizione di avvio con `set root=` come indicato in precedenza, caricare il kernel `linux` (con `linux`), il disco RAM iniziale (`initrd`) e provare ad avviare il tutto con `boot`.

GRUB Legacy

Installazione di GRUB Legacy su un sistema in esecuzione

Per installare GRUB Legacy su un disco da un sistema in esecuzione si utilizzerà l'utility `grub-install`. Il comando di base è `grub-install DEVICE` dove `DEVICE` è il disco su cui si desidera installare GRUB Legacy. Un esempio potrebbe essere `/dev/sda`.

```
# grub-install /dev/sda
```

Da notare che se devi specificare il *device* dove verrà installato GRUB Legacy, come `/dev/sda/` e *non la partizione* come per esempio `/dev/sda1`.

Di default GRUB copia i file necessari nella directory `/boot` sul dispositivo specificato. Se si desidera copiarli in un'altra directory, utilizzare il parametro `--boot-directory=`, seguito dal percorso completo in cui devono essere copiati i file.

Installazione di GRUB Legacy da una shell GRUB

Se non si riesce ad avviare il sistema per qualche motivo e si deve reinstallare GRUB Legacy, ciò può essere fatto dalla shell GRUB su un disco di avvio di GRUB Legacy.

Dalla shell GRUB (digitare `c` nel menu di avvio per accedere al prompt `grub>`), il primo passo è impostare il dispositivo di avvio, che contiene la directory `/boot`. Per esempio, se questa directory si trova nella prima partizione del primo disco, il comando sarà:

```
grub> root (hd0,0)
```

Se non si conosce quale dispositivo contiene la directory `/boot`, si può chiedere a GRUB di cercarla con il comando `find``:

```
grub> find /boot/grub/stage1
(hd0,0)
```

Quindi, impostare la partizione di avvio come indicato sopra e usare il comando `setup` per installare GRUB Legacy sull'MBR e copiare i file necessari sul disco:

```
grub> setup (hd0)
```

Al termine, riavviare il sistema e tutto dovrebbe rifunzionare nuovamente.

Configurazione delle Voci e delle Impostazioni del Menu Legacy di GRUB

Le voci e le impostazioni del menu legacy di GRUB sono memorizzate nel file `/boot/grub/menu.lst`. Questo è un semplice file di testo con un elenco di comandi e parametri, che può essere modificato direttamente con l'editor di testo preferito.

Le righe che iniziano con `#` sono considerate commenti e le righe vuote vengono ignorate.

Una voce di menu ha almeno tre comandi. Il primo, `title`, imposta il titolo del sistema operativo nella schermata del menu. Il secondo, `root`, dice a GRUB Legacy da quale dispositivo o partizione avviare.

La terza voce, `kernel`, specifica il percorso completo dell'immagine del kernel che dovrebbe essere caricata quando viene selezionata la voce corrispondente. Si noti che questo percorso è relativo al dispositivo specificato sul parametro `root`.

Segue un semplice esempio:

```
# This line is a comment
title My Linux Distribution
root (hd0,0)
kernel /vmlinuz root=/dev/hda1
```

A differenza di GRUB 2, in GRUB Legacy entrambi i dischi e le partizioni sono numerati da zero. Quindi, il comando `root (hd0,0)` imposterà la partizione di avvio come prima partizione (0) del primo disco (hd0).

È possibile omettere l'istruzione `root` se si specifica il dispositivo di avvio prima del percorso sul comando `kernel`. La sintassi è la stessa, quindi:

```
kernel (hd0,0)/vmlinuz root=dev/hda1
```

è equivalente a:

```
root (hd0,0)
kernel /vmlinuz root=/dev/hda1
```

Entrambi caricheranno il file `vmlinuz` dalla directory `root (/)` della prima partizione del primo disco (`hd0, 0`).

Il parametro `root=/dev/hda1` dopo il comando `kernel` dice al kernel Linux quale partizione dovrebbe essere usata come filesystem di root. Questo è un parametro del kernel Linux, non un comando GRUB Legacy.

NOTE

Per ulteriori informazioni sui parametri del kernel, vedere <https://www.kernel.org/doc/html/v4.14/admin-guide/kernel-parameters.html>.

Potrebbe essere necessario specificare la posizione dell'immagine del disco RAM iniziale per il sistema operativo con il parametro `initrd`. Il percorso completo del file può essere specificato come nel parametro `kernel`, e si può anche specificare un dispositivo o una partizione prima del percorso, per esempio:

```
# This line is a comment
title My Linux Distribution
root (hd0,0)
kernel /vmlinuz root=/dev/hda1
initrd /initrd.img
```

GRUB Legacy ha un design modulare, in cui i moduli (normalmente memorizzati come file `.mod` in `/boot/grub/i386-pc`) possono essere caricati per aggiungere funzionalità extra, come il supporto per hardware insolito, filesystem o nuovi algoritmi di compressione.

I moduli vengono caricati usando il comando `module`, seguito dal percorso completo del corrispondente file `.mod`. Si tenga presente che, come i kernel e le immagini `initrd`, questo percorso è relativo al dispositivo specificato nel comando `root`.

L'esempio seguente caricherà il modulo `915resolution`, necessario per impostare correttamente la risoluzione del *framebuffer* su sistemi con chipset video Intel serie 800 o 900.

```
module /boot/grub/i386-pc/915resolution.mod
```

Chainloading di Altri Sistemi Operativi

GRUB Legacy può essere utilizzato per caricare sistemi operativi non supportati, come Windows, utilizzando un processo chiamato *chainloading*. GRUB Legacy viene caricato per primo e quando viene selezionata l'opzione corrispondente viene caricato il bootloader specifico del sistema desiderato.

Una voce tipica per il chainloading di Windows sarebbe simile a quella mostrata di seguito:

```
# Load Windows
title Windows XP
root (hd0,1)
makeactive
chainload +1
boot
```

Come in precedenza, `root (hd0,1)` specifica il dispositivo e la partizione in cui si trova il boot loader per il sistema operativo che desideriamo caricare. In questo esempio, la *seconda partizione* del primo disco.

makeactive

imposterà un flag indicante che si tratta di una partizione attiva. Funziona solo su partizioni primarie DOS.

chainload +1

dice a GRUB di caricare il primo settore della partizione di avvio. Qui di solito si trovano i bootloader..

boot

eseguirà il bootloader e caricherà il sistema operativo corrispondente.

Esercizi Guidati

1. Qual è il percorso predefinito per il file di configurazione di GRUB 2?
2. Quali sono i passaggi necessari per modificare le impostazioni di GRUB 2?
3. In quale file devono essere aggiunte voci di menu personalizzate di GRUB 2?
4. Dove sono memorizzate le voci di menu per GRUB Legacy?
5. Da un menu GRUB 2 o GRUB Legacy, come puoi accedere a GRUB Shell?

Esercizi Esplorativi

1. Immagina un utente che configura GRUB Legacy per l'avvio dalla seconda partizione del primo disco. Egli scrive la seguente voce di menu personalizzata:

```
title My Linux Distro
root (hd0,2)
kernel /vmlinuz root=/dev/hda1
initrd /initrd.img
```

Tuttavia, il sistema non si avvia. Qual è l'errore?

2. Immagina di avere un disco identificato come `/dev/sda` con più partizioni. Quale comando può essere utilizzato per scoprire qual è la partizione di avvio su un sistema?

3. Quale comando può essere utilizzato per scoprire l'UUID di una partizione?

4. Considera la seguente voce per GRUB 2

```
menuentry "Default OS" {
    set root=(hd0,1)
    linux /vmlinuz root=/dev/sda1 ro quiet splash
    initrd /initrd.img
}
```

Modificalo in modo che il sistema si avvii da un disco con l'UUID `5dda0af3-c995-481a-a6f3-46dcd3b6998d`

5. Come si può impostare GRUB 2 così che attenda 10 secondi prima di avviare la voce del menu predefinita?

6. Da una shell legacy di GRUB, quali sono i comandi per installare GRUB sulla prima partizione del secondo disco?

Sommario

In questa lezione abbiamo imparato:

- Che cos'è un boot loader.
- Le differenze tra GRUB Legacy e GRUB 2.
- Che cos'è una partizione di avvio e quali sono i suoi contenuti.
- Come installare GRUB Legacy e GRUB 2.
- Come configurare GRUB Legacy e GRUB 2.
- Come aggiungere voci di menu personalizzate a GRUB Legacy e GRUB 2.
- Come interagire con la schermata del menu e la console di GRUB Legacy e GRUB 2.
- Come avviare un sistema da una shell GRUB Legacy o GRUB 2 o da una shell di rescue.

In questa lezione sono stati discussi i seguenti comandi:

- `grub-install`
- `update-grub`
- `grub-mkconfig`

Risposte agli Esercizi Guidati

1. Qual è il percorso predefinito per il file di configurazione di GRUB 2?

`/boot/grub/grub.cfg`

2. Quali sono i passaggi necessari per modificare le impostazioni di GRUB 2?

Modificare il file `/etc/default/grub`, quindi aggiornare la configurazione con `update-grub`.

3. In quale file devono essere aggiunte voci di menu personalizzate di GRUB 2?

`/etc/grub.d/40_custom`

4. Dove sono memorizzate le voci di menu per GRUB Legacy?

`/boot/grub/menu.lst`

5. Da un menu GRUB 2 o GRUB Legacy, come puoi accedere a GRUB Shell?

Premere `c` nella schermata del menu.

Risposte agli Esercizi Guidati

1. Immagina un utente che configura GRUB Legacy per l'avvio dalla seconda partizione del primo disco. Egli scrive la seguente voce di menu personalizzata:

```
title My Linux Distro
root (hd0,2)
kernel /vmlinuz root=/dev/hda1
initrd /initrd.img
```

Tuttavia, il sistema non si avvia. Qual è l'errore?

La partizione di avvio è errata. Ricorda che, a differenza di GRUB 2, GRUB Legacy conta le partizioni da *zero*. Quindi, il comando corretto per la seconda partizione del primo disco dovrebbe essere `root (hd0,1)`.

2. Immagina di avere un disco identificato come `/dev/sda` con più partizioni. Quale comando può essere utilizzato per scoprire qual è la partizione di avvio su un sistema?

Utilizzare `fdisk -l /dev/sda`. La partizione di avvio verrà contrassegnata nell'elenco con un asterisco (*).

3. Quale comando può essere utilizzato per scoprire l'UUID di una partizione?

Usare `ls -la /dev/disk/by-uuid/` e cercare l'UUID che punta alla partizione.

4. Considera la seguente voce per GRUB 2

```
menuentry "Default OS" {
    set root=(hd0,1)
    linux /vmlinuz root=/dev/sda1 ro quiet splash
    initrd /initrd.img
}
```

Modificalo in modo che il sistema si avvii da un disco con l'UUID `5dda0af3-c995-481a-a6f3-46dcd3b6998d`

Cambiare l'istruzione `set root`. Invece di specificare un disco e una partizione, far cercare a Grub la partizione con l'UUID desiderato.

```
menuentry "Default OS" {
```

```
search --set=root --fs-uuid 5dda0af3-c995-481a-a6f3-46dcd3b6998d
linux /vmlinuz root=/dev/sda1 ro quiet splash
initrd /initrd.img
}
```

5. Come si può impostare GRUB 2 così che attenda 10 secondi prima di avviare la voce del menu predefinita?

Aggiungere il parametro `GRUB_TIMEOUT=10` a `/etc/default/grub`.

6. Da una shell legacy di GRUB, quali sono i comandi per installare GRUB sulla prima partizione del secondo disco?

```
grub> root (hd1,0)
grub> setup (hd1)
```



102.3 Gestire le librerie condivise

Obiettivi LPI di riferimento

LPIC-1 v5, Exam 101, Objective 102.3

Peso

1

Aree di Conoscenza Chiave

- Identificare le librerie condivise.
- Identificare le posizioni tipiche delle librerie di sistema.
- Caricare le librerie condivise.

Di seguito è riportato un elenco parziale dei file, dei termini e dei comandi utilizzati

- `ldd`
- `ldconfig`
- `/etc/ld.so.conf`
- `LD_LIBRARY_PATH`



Linux
Professional
Institute

102.3 Lezione 1

Certificazione:	LPIC-1
Versione:	5.0
Argomento:	102 L'Installazione di Linux e la Gestione Pacchetti
Obiettivo:	102.3 Gestire le Librerie Condivise
Lezione:	1 di 1

Introduzione

In questa lezione discuteremo di *librerie condivise*, anche conosciute come *oggetti condivisi*: sequenze di codice compilato e riutilizzabile come funzioni o classi, che vengono utilizzati di frequente da vari programmi.

Per iniziare, spiegheremo cosa sono le librerie condivise, come identificarle e dove si trovano. Successivamente, esamineremo come configurare le loro posizioni di archiviazione; infine, mostreremo come cercare le librerie condivise da cui dipende un determinato programma.

Le Librerie Condivise

Similmente alle loro controparti fisiche, le librerie di software sono raccolte di codice che possono essere utilizzate da molti programmi diversi; così come, nel mondo reale, le biblioteche mantengono libri e altre risorse che possono essere utilizzate da molte persone diverse.

Per creare un file eseguibile dal codice sorgente di un programma, è necessario eseguire due passaggi importanti. Innanzitutto, il *compilatore* che trasforma il codice sorgente in codice

macchina generando i cosiddetti *file oggetto*. In secondo luogo, il *linker* che combina i vari file oggetto e li *collega* alle librerie per generare il file eseguibile finale. Questo collegamento può essere eseguito in maniera *statica* o *dinamica*. A seconda del metodo scelto, parleremo di librerie statiche o, in caso di collegamento dinamico, di librerie condivise. Spieghiamone le differenze.

Librerie Statiche

Una libreria statica viene unita al programma al momento del collegamento. Una copia del codice della libreria è incorporata nel programma e ne diventa parte. Pertanto, il programma non ha dipendenze dalla libreria in fase di esecuzione perché il programma contiene già il codice delle librerie. Non avere dipendenze può essere visto come un vantaggio poiché non ci si deve preoccupare che le librerie utilizzate siano sempre disponibili. L'aspetto negativo è che i programmi collegati staticamente sono più pesanti.

Librerie condivise (o dinamiche)

Nel caso di librerie condivise, il linker si occupa semplicemente che il programma faccia riferimento correttamente alle librerie. Il linker, tuttavia, non copia alcun codice di libreria nel file di programma. In fase di esecuzione, tuttavia, la libreria condivisa deve essere disponibile per soddisfare le dipendenze del programma. Questo è un approccio economico alla gestione delle risorse di sistema in quanto aiuta a ridurre le dimensioni dei file di programma e solo una copia della libreria viene caricata in memoria, anche quando viene utilizzata da più programmi.

Convenzioni di Denominazione dei File Oggetto Condivisi

Il nome di una libreria condivisa, noto anche come *soname*, segue uno schema composto da tre elementi:

- Nome della libreria (normalmente preceduto da `lib`)
- `so` (che sta per “oggetto condiviso”)
- Numero di versione della libreria

Ecco un esempio: `libpthread.so.0`

Al contrario, i nomi delle librerie statiche terminano in `.a`, ad es. `libpthread.a`.

NOTE

Poiché i file contenenti librerie condivise devono essere disponibili quando viene eseguito il programma, la maggior parte dei sistemi Linux contiene librerie condivise. Al contrario le librerie statiche sono invece richieste in un file dedicato solo quando un programma è collegato a essa, quindi, data la loro scarsa diffusione, potrebbero non essere presenti sul sistema dell'utente finale.

`glibc` (libreria GNU C) è un buon esempio di libreria condivisa. Su un sistema Debian GNU/Linux 9.9, il suo file è chiamato `libc.so.6`. Tali nomi di file piuttosto generici sono normalmente collegamenti simbolici che puntano al file effettivo contenente una libreria, il cui nome contiene il numero esatto di versione. Nel caso di `glibc`, questo collegamento simbolico è simile al seguente:

```
$ ls -l /lib/x86_64-linux-gnu/libc.so.6
lrwxrwxrwx 1 root root 12 feb  6 22:17 /lib/x86_64-linux-gnu/libc.so.6 -> libc-2.24.so
```

Questo modo di riferirsi ai file di libreria condivisa con nomi di file più generali è una pratica piuttosto comune.

Altri esempi di librerie condivise includono `libreadline` (che consente agli utenti di modificare le righe di comando durante la digitazione e include il supporto per entrambe le modalità di modifica di *Emacs* e *vi*), `libcrypt` (che contiene funzioni relative a crittografia, hash e codifica), o `libcurl` (che è una libreria di trasferimento file multiprotocollo).

Le posizioni comuni per le librerie condivise in un sistema Linux sono:

- `/lib`
- `/lib32`
- `/lib64`
- `/usr/lib`
- `/usr/local/lib`

NOTE

Il concetto di librerie condivise non è esclusivo di Linux. In Windows, per esempio, sono chiamate DLL, che significa *dynamic linked libraries*.

Configurazione dei Percorsi delle Librerie Condivise

I riferimenti alle librerie collegate dinamicamente vengono risolti dal linker dinamico (`ld.so` o `ld-linux.so`) quando il programma viene eseguito. Il linker dinamico cerca le librerie in una serie di directory. Queste directory sono specificate dal *library path*. Il percorso della libreria è configurato nella directory `/etc`, vale a dire nel file `/etc/ld.so.conf` e a oggi più comunemente nei file che risiedono nella directory `/etc/ld.so.conf.d`. Normalmente, il file include solo una singola riga `include` per i file `*.conf` che verranno a trovarsi nella directory precedentemente indicata:

```
$ cat /etc/ld.so.conf
```

```
include /etc/ld.so.conf.d/*.conf
```

La directory `/etc/ld.so.conf.d` contiene diversi file `*.conf`:

```
$ ls /etc/ld.so.conf.d/  
libc.conf  x86_64-linux-gnu.conf
```

Questi file `*.conf` hanno al loro interno i percorsi assoluti delle directory che contengono le librerie condivise:

```
$ cat /etc/ld.so.conf.d/x86_64-linux-gnu.conf  
# Multiarch support  
/lib/x86_64-linux-gnu  
/usr/lib/x86_64-linux-gnu
```

Il comando `ldconfig` si occupa della lettura di questi file di configurazione, creando i collegamenti simbolici, precedentemente descritti, che aiutano a localizzare le singole librerie effettuando infine l'aggiornamento del file di cache `/etc/ld.so.cache`. Pertanto, `ldconfig` deve essere eseguito ogni volta che vengono aggiunti o aggiornati i file di configurazione.

Opzioni utili per `ldconfig` sono:

-v, --verbose

Visualizza le attività in dettaglio svolte dal comando durante la sua esecuzione:

```
$ sudo ldconfig -v  
/usr/local/lib:  
/lib/x86_64-linux-gnu:  
  libnss_myhostname.so.2 -> libnss_myhostname.so.2  
  libfuse.so.2 -> libfuse.so.2.9.7  
  libidn.so.11 -> libidn.so.11.6.16  
  libnss_mdns4.so.2 -> libnss_mdns4.so.2  
  libparted.so.2 -> libparted.so.2.0.1  
  (...)
```

Quindi come possiamo vedere per esempio, `libfuse.so.2` è collegato all'oggetto file condiviso `libfuse.so.2.9.7`.

-p, --print-cache

Stampa gli elenchi di directory e librerie archiviati nella cache corrente:

```
$ sudo ldconfig -p
1094 libs found in the cache `/etc/ld.so.cache'
  libzvbi.so.0 (libc6,x86-64) => /usr/lib/x86_64-linux-gnu/libzvbi.so.0
  libzvbi-chains.so.0 (libc6,x86-64) => /usr/lib/x86_64-linux-gnu/libzvbi-chains.so.0
  libzmq.so.5 (libc6,x86-64) => /usr/lib/x86_64-linux-gnu/libzmq.so.5
  libzeitgeist-2.0.so.0 (libc6,x86-64) => /usr/lib/x86_64-linux-gnu/libzeitgeist-
2.0.so.0
  (...)
```

Nota come la cache utilizza il soname completo dei collegamenti:

```
$ sudo ldconfig -p |grep libfuse
  libfuse.so.2 (libc6,x86-64) => /lib/x86_64-linux-gnu/libfuse.so.2
```

Se visualizziamo in formato long `/lib/x86_64-linux-gnu/libfuse.so.2`, troveremo il riferimento all'effettivo file oggetto condiviso `libfuse.so.2.9.7` che è memorizzato nella stessa directory:

```
$ ls -l /lib/x86_64-linux-gnu/libfuse.so.2
lrwxrwxrwx 1 root root 16 Aug 21 2018 /lib/x86_64-linux-gnu/libfuse.so.2 ->
libfuse.so.2.9.7
```

NOTE

Dato che è richiesto l'accesso in scrittura al file `/etc/ld.so.cache` (di proprietà di root), è necessario essere root o utilizzare `sudo` per invocare `ldconfig`. Per maggiori informazioni sulle opzioni di `ldconfig`, consultare la pagina di manuale relativa.

Oltre ai file di configurazione sopra descritti, la variabile d'ambiente `LD_LIBRARY_PATH` può essere usata per aggiungere temporaneamente nuovi percorsi per librerie condivise. È costituito da una serie di directory separate da due punti (:) in cui vengono cercate le librerie. Per aggiungere, per esempio, `/usr/local/mylib` al percorso delle librerie nella sessione di shell corrente, è possibile digitare:

```
$ LD_LIBRARY_PATH=/usr/local/mylib
```

Ora possiamo verificarne il valore:

```
$ echo $LD_LIBRARY_PATH
```

```
/usr/local/mylib
```

Per aggiungere `/usr/local/mylib` al percorso delle librerie condivise nella sessione di shell corrente e farlo esportare in tutti i processi figlio generati da quella shell, si dovrebbe digitare:

```
$ export LD_LIBRARY_PATH=/usr/local/mylib
```

Per rimuovere la variabile d'ambiente `LD_LIBRARY_PATH`, basta digitare:

```
$ unset LD_LIBRARY_PATH
```

Per rendere permanenti le modifiche, è possibile scrivere la riga

```
export LD_LIBRARY_PATH=/usr/local/mylib
```

in uno degli script di inizializzazione di Bash come `/etc/bash.bashrc` o `~/.bashrc`.

NOTE

`LD_LIBRARY_PATH` è per le librerie condivise ciò che `PATH` è per gli eseguibili. Per ulteriori informazioni sulle variabili di ambiente e sulla configurazione della shell, consultare le rispettive lezioni.

Ricerca delle Dipendenze di un Eseguibile Specifico

Per cercare le librerie condivise richieste da un programma specifico, usare il comando `ldd` seguito dal percorso assoluto del programma. L'output mostra il percorso del file della libreria condivisa e l'indirizzo di memoria esadecimale in cui è caricato:

```
$ ldd /usr/bin/git
linux-vdso.so.1 => (0x00007ffcfbb31000)
libpcr.so.3 => /lib/x86_64-linux-gnu/libpcr.so.3 (0x00007f18241eb000)
libz.so.1 => /lib/x86_64-linux-gnu/libz.so.1 (0x00007f1823fd1000)
libresolv.so.2 => /lib/x86_64-linux-gnu/libresolv.so.2 (0x00007f1823db6000)
libpthread.so.0 => /lib/x86_64-linux-gnu/libpthread.so.0 (0x00007f1823b99000)
librt.so.1 => /lib/x86_64-linux-gnu/librt.so.1 (0x00007f1823991000)
libc.so.6 => /lib/x86_64-linux-gnu/libc.so.6 (0x00007f18235c7000)
/lib64/ld-linux-x86-64.so.2 (0x00007f182445b000)
```

Allo stesso modo, usiamo `ldd` per cercare le dipendenze di un oggetto condiviso:

```
$ ldd /lib/x86_64-linux-gnu/libc.so.6
/lib64/ld-linux-x86-64.so.2 (0x00007fbfed578000)
linux-vdso.so.1 (0x00007fffb7bf5000)
```

Con l'opzione `-u` (o `--unused`) `ldd` visualizza le dipendenze dirette inutilizzate (se presenti):

```
$ ldd -u /usr/bin/git
Unused direct dependencies:
/lib/x86_64-linux-gnu/libz.so.1
/lib/x86_64-linux-gnu/libpthread.so.0
/lib/x86_64-linux-gnu/librt.so.1
```

Il motivo delle dipendenze non utilizzate è correlato alle opzioni utilizzate dal linker durante la creazione del file binario. Sebbene il programma non abbia bisogno di una libreria inutilizzata, è stato comunque collegato ed etichettato come "NECESSARIO" nelle informazioni sul file oggetto. È possibile verificarne la presenza usando comandi `readelf` o `objdump`, che verranno mostrati negli esercizi esplorativi.

Esercizi Guidati

1. Dividere i seguenti nomi di libreria condivisa nelle loro parti:

Nome completo del file	Nome della libreria	Suffisso <i>so</i>	Numero della versione
linux-vdso.so.1			
libprocps.so.6			
libdl.so.2			
libc.so.6			
libsystemd.so.0			
ld-linux-x86-64.so.2			

2. È stato sviluppato un software e si vuole aggiungere una nuova directory di libreria condivisa al sistema (`/opt/lib/mylib`). Scrivere il suo percorso assoluto in un file chiamato `mylib.conf`.

- In quale directory si deve inserire questo file?

- Quale comando si deve eseguire per rendere le modifiche pienamente operative?

3. Quale comando si utilizza per elencare le librerie condivise richieste dal comando `kill`?

Esercizi Esplorativi

1. `objdump` è un'utilità della riga di comando che visualizza le informazioni dai file oggetto. Controlla se è installato nel tuo sistema con `which objdump`. In caso contrario, installalo.

- Utilizza `objdump` con l'opzione `-p` (o `--private-headers`) e `grep` per visualizzare le dipendenze di `glibc`:

- Utilizza `objdump` con l'opzione `-p` (o `--private-headers`) e `grep` per stampare il soname di glibc:`

- Usa `objdump` con l'opzione `-p` (o `--private-headers`) e `grep` per stampare le dipendenze di Bash:

Sommario

In questa lezione abbiamo imparato:

- Che cos'è una libreria condivisa (o dinamica).
- Le differenze tra librerie condivise e statiche.
- I nomi delle librerie condivise (*sonames*).
- Le posizioni predefinite per le librerie condivise in un sistema Linux come `/lib` o `/usr/lib`.
- Lo scopo del linker dinamico `ld.so` (o `ld-linux.so`).
- Come configurare i percorsi delle librerie condivise tramite i file `/etc/ld.so.conf` o quelli nella directory `ld.so.conf.d`.
- Come configurare i percorsi delle librerie condivise mediante la variabile d'ambiente `LD_LIBRARY_PATH`.
- Come cercare le dipendenze di libreria per eseguibili e oggetti condivisi.

Comandi utilizzati in questa lezione:

ls

Elenca i contenuti della directory.

cat

Concatena i file e visualizza sullo standard output.

sudo

Consente a un utente autorizzato di eseguire un comando con privilegi amministrativi.

ldconfig

Configura i collegamenti di runtime del linker dinamico.

echo

Visualizza il valore di una variabile d'ambiente.

export

Esporta il valore di una variabile d'ambiente in shell figlie.

unset

Rimuove variabile d'ambiente.

ldd

Visualizza le dipendenze di un programma.

readelf

Visualizza informazioni sui file ELF (ELF sta per *executable and linkable format*).

objdump

Visualizza le informazioni dei file oggetto.

Risposte agli Esercizi Guidati

1. Dividere i seguenti nomi di libreria condivisa nelle loro parti:

Nome completo del file	Nome della libreria	Suffisso <i>so</i>	Numero della versione
linux-vdso.so.1	linux-vdso	so	1
libprocps.so.6	libprocps	so	6
libdl.so.2	libdl	so	2
libc.so.6	libc	so	6
libsystemd.so.0	libsystemd	so	0
ld-linux-x86-64.so.2	ld-linux-x86-64	so	2

2. È stato sviluppato un software e si vuole aggiungere una nuova directory di libreria condivisa al sistema (/opt/lib/mylib). Scrivere il suo percorso assoluto in un file chiamato mylib.conf.

- In quale directory si deve inserire questo file?

```
/etc/ld.so.conf.d
```

- Quale comando si deve eseguire per rendere le modifiche pienamente operative?

```
ldconfig
```

3. Quale comando si utilizza per elencare le librerie condivise richieste dal comando `kill`?

```
ldd /bin/kill
```

Risposte agli Esercizi Esplorativi

1. `objdump` è un'utilità della riga di comando che visualizza le informazioni dai file oggetto. Controlla se è installato nel tuo sistema con `which objdump`. In caso contrario, installalo.

- Utilizza `objdump` con l'opzione `-p` (o `--private-headers`) e `grep` per stampare il soname di `glibc`:

```
objdump -p /lib/x86_64-linux-gnu/libc.so.6 | grep NEEDED
```

- Utilizza `objdump` con l'opzione `-p` (o `--private-headers`) e `grep` per stampare il soname di `glibc`:

```
objdump -p /lib/x86_64-linux-gnu/libc.so.6 | grep SONAME
```

- Usa `objdump` con l'opzione `-p` (o `--private-headers`) e `grep` per stampare le dipendenze di Bash:

```
objdump -p /bin/bash | grep NEEDED
```



102.4 Utilizzare la gestione dei pacchetti Debian

Obiettivi LPI di riferimento

LPIC-1 v5, Exam 101, Objective 102.4

Peso

3

Aree di Conoscenza Chiave

- Installare, aggiornare e disinstallare i pacchetti binari Debian.
- Trovare i pacchetti contenenti file o librerie specifici che possono o non possono essere installati.
- Ottenere informazioni sul pacchetto come versione, contenuto, dipendenze, integrità del pacchetto e stato dell'installazione (indipendentemente dal fatto che il pacchetto sia installato o meno).
- Conoscenza di apt.

Di seguito è riportato un elenco parziale dei file, dei termini e dei comandi utilizzati

- `/etc/apt/sources.list`
- `dpkg`
- `dpkg-reconfigure`
- `apt-get`
- `apt-cache`



102.4 Lezione 1

Certificazione:	LPIC-1
Versione:	5.0
Argomento:	102 L'Installazione di Linux e la Gestione Pacchetti
Obiettivo:	102.4 Utilizzare la Gestione Pacchetti Debian
Lezione:	1 di 1

Introduzione

Molto tempo fa, quando Linux era ancora agli inizi, il modo più comune per distribuire il software era un file compresso (di solito un archivio `.tar.gz`) con codice sorgente, che si sarebbe decompresso e compilato per conto proprio.

Tuttavia, con l'aumentare della quantità e della complessità del software, divenne evidente la necessità di un modo per distribuire software precompilato. Dopotutto, non tutti avevano le risorse, sia in termini di tempo sia di potenza di elaborazione, per compilare grandi progetti come il kernel Linux o un X Server.

Presto sono cresciuti gli sforzi per standardizzare la distribuzione di questi “pacchetti” software e sono nati i primi gestori di pacchetti. Questi strumenti hanno reso molto più semplice l'installazione, la configurazione o la rimozione di software da un sistema.

Uno di questi era il formato di pacchetto *Debian* (`.deb`) e il suo strumento di gestione (`dpkg`). Oggi sono ampiamente utilizzati non solo sulla stessa Debian, ma anche sulle sue derivate, come Ubuntu e molte altre.

Un altro strumento di gestione dei pacchetti che è popolare sui sistemi basati su Debian è *Advanced Package Tool* (`apt`), che può semplificare molti aspetti dell'installazione, della manutenzione e della rimozione dei pacchetti, rendendolo ancora più semplice.

In questa lezione impareremo come usare sia `dpkg` sia `apt` per ottenere, installare, mantenere e rimuovere software su un sistema Linux basato su Debian.

Lo Strumento di Gestione Pacchetti in Debian (dpkg)

Lo strumento *Debian Package* (`dpkg`) è il comando essenziale per installare, configurare, mantenere e rimuovere i pacchetti software su sistemi basati su Debian. L'operazione più semplice è installare un pacchetto `.deb`, che può essere fatto con:

```
# dpkg -i PACKAGENAME
```

Dove `PACKAGENAME` è il nome del file `.deb` che si desidera installare.

Gli aggiornamenti dei pacchetti vengono gestiti allo stesso modo. Prima di installare un pacchetto, `dpkg` verificherà se nel sistema esiste già una versione precedente. In tal caso, il pacchetto verrà aggiornato alla nuova versione. In caso contrario, verrà installata una nuova copia.

Gestire le Dipendenze

Molto spesso un pacchetto può dipendere da altri per funzionare a dovere. Per esempio, un editor di immagini potrebbe aver bisogno di librerie per aprire file JPEG o un'altra utility potrebbe aver bisogno di un *widget toolkit* come Qt o GTK per la sua interfaccia utente.

`dpkg` controllerà se queste dipendenze sono installate sul sistema e non installerà il pacchetto in mancanza di esse. In questo caso, `dpkg` elencherà quali pacchetti mancano. Tuttavia, non può risolvere le dipendenze da solo: spetta all'utente trovare i pacchetti `.deb` con le dipendenze corrispondenti e installarli.

Nell'esempio seguente, l'utente tenta di installare il pacchetto dell'editor video OpenShot, ma mancano alcune dipendenze:

```
# dpkg -i openshot-qt_2.4.3+dfsg1-1_all.deb
(Reading database ... 269630 files and directories currently installed.)
Preparing to unpack openshot-qt_2.4.3+dfsg1-1_all.deb ...
Unpacking openshot-qt (2.4.3+dfsg1-1) over (2.4.3+dfsg1-1) ...
dpkg: dependency problems prevent configuration of openshot-qt:
openshot-qt depends on fonts-cantarell; however:
```

```
Package fonts-cantarell is not installed.
openshot-qt depends on python3-openshot; however:
Package python3-openshot is not installed.
openshot-qt depends on python3-pyqt5; however:
Package python3-pyqt5 is not installed.
openshot-qt depends on python3-pyqt5.qtsvg; however:
Package python3-pyqt5.qtsvg is not installed.
openshot-qt depends on python3-pyqt5.qtwebkit; however:
Package python3-pyqt5.qtwebkit is not installed.
openshot-qt depends on python3-zmq; however:
Package python3-zmq is not installed.
```

```
dpkg: error processing package openshot-qt (--install):
dependency problems - leaving unconfigured
Processing triggers for mime-support (3.60ubuntu1) ...
Processing triggers for gnome-menus (3.32.0-1ubuntu1) ...
Processing triggers for desktop-file-utils (0.23-4ubuntu1) ...
Processing triggers for hicolor-icon-theme (0.17-2) ...
Processing triggers for man-db (2.8.5-2) ...
Errors were encountered while processing:
openshot-qt
```

Come mostrato sopra, OpenShot dipende dai pacchetti `fonts-cantarell`, `python3-openshot`, `python3-pyqt5`, `python3-pyqt5.qtsvg`, `python3-pyqt5.qtwebkit` e `python3-zmq`. Tutti questi devono essere installati prima che l'installazione di OpenShot possa avvenire con successo.

Rimozione dei Pacchetti

Per rimuovere un pacchetto, passare il parametro `-r` a `dpkg`, seguito dal nome del pacchetto. Per esempio, il seguente comando rimuoverà il pacchetto `unrar` dal sistema:

```
# dpkg -r unrar
(Reading database ... 269630 files and directories currently installed.)
Removing unrar (1:5.6.6-2) ...
Processing triggers for man-db (2.8.5-2) ...
```

L'operazione di rimozione esegue anche un controllo di dipendenza e un pacchetto non può essere rimosso a meno che non venga rimosso anche ogni altro pacchetto che dipende da esso. Se si prova a farlo, si riceverà un messaggio di errore come quello di seguito:

```
# dpkg -r p7zip
```

```
dpkg: dependency problems prevent removal of p7zip:
  winetricks depends on p7zip; however:
    Package p7zip is to be removed.
  p7zip-full depends on p7zip (= 16.02+dfsg-6).
```

```
dpkg: error processing package p7zip (--remove):
  dependency problems - not removing
Errors were encountered while processing:
  p7zip
```

Si possono passare più nomi di pacchetti a `dpkg -r`, al fine di poterli rimuovere tutti in una volta.

Quando un pacchetto viene rimosso, i file di configurazione corrispondenti vengono lasciati sul sistema. Se si vuole rimuovere *qualsiasi cosa* associata al pacchetto, usare l'opzione `-P` (purge) invece di `-r`.

NOTE

È possibile forzare `dpkg` ad installare o rimuovere un pacchetto, anche se le dipendenze non sono soddisfatte, aggiungendo il parametro `--force` come in `dpkg -i --force PACKAGENAME`. Tuttavia, molto probabilmente, ciò lascerà il pacchetto installato, o anche il tuo sistema, in uno stato di inconsistenza. *Non* utilizzare `--force` a meno di non essere assolutamente sicuri di cosa si stia facendo.

Ottenere Informazioni sui Pacchetti

Per ottenere informazioni su un pacchetto `.deb`, come la sua versione, architettura, *maintainer*, dipendenze e altro, usare il comando `dpkg` con l'opzione `-I`, seguito dal nome del file del pacchetto che si vuole controllare:

```
# dpkg -I google-chrome-stable_current_amd64.deb
new Debian package, version 2.0.
size 59477810 bytes: control archive=10394 bytes.
   1222 bytes,   13 lines   control
  16906 bytes,  457 lines *  postinst          #!/bin/sh
  12983 bytes,  344 lines *  postrm           #!/bin/sh
   1385 bytes,   42 lines *  prerm            #!/bin/sh
Package: google-chrome-stable
Version: 76.0.3809.100-1
Architecture: amd64
Maintainer: Chrome Linux Team <chromium-dev@chromium.org>
Installed-Size: 205436
Pre-Depends: dpkg (>= 1.14.0)
Depends: ca-certificates, fonts-liberation, libappindicator3-1, libasound2 (>= 1.0.16),
```



```
libatk-bridge2.0-0 (>= 2.5.3), libatk1.0-0 (>= 2.2.0), libatspi2.0-0 (>= 2.9.90), libc6 (>=
2.16), libcairo2 (>= 1.6.0), libcups2 (>= 1.4.0), libdbus-1-3 (>= 1.5.12), libexpat1 (>=
2.0.1), libgcc1 (>= 1:3.0), libgdk-pixbuf2.0-0 (>= 2.22.0), libglib2.0-0 (>= 2.31.8),
libgtk-3-0 (>= 3.9.10), libnspr4 (>= 2:4.9-2~), libnss3 (>= 2:3.22), libpango-1.0-0 (>=
1.14.0), libpangocairo-1.0-0 (>= 1.14.0), libuuid1 (>= 2.16), libx11-6 (>= 2:1.4.99.1),
libx11-xcb1, libxcb1 (>= 1.6), libxcomposite1 (>= 1:0.3-1), libxcursor1 (>= 1.1.2),
libxdamage1 (>= 1:1.1), libxext6, libxfixed3, libxi6 (>= 2:1.2.99.4), libxrandr2 (>=
2:1.2.99.3), libxrender1, libxss1, libxtst6, lsb-release, wget, xdg-utils (>= 1.0.2)
Recommends: libu2f-udev
Provides: www-browser
Section: web
Priority: optional
Description: The web browser from Google
Google Chrome is a browser that combines a minimal design with sophisticated technology to
make the web faster, safer, and easier.
```

Elencare i Pacchetti Installati e il loro Contenuto

Per ottenere un elenco di tutti i pacchetti installati sul sistema, usare l'opzione `--get-selections`, come in `dpkg --get-selections`. Per ottenere un elenco di tutti i file installati da un pacchetto specifico si può utilizzare il parametro `-L PACKAGENAME` come di seguito:

```
# dpkg -L unrar
/.
/usr
/usr/bin
/usr/bin/unrar-nonfree
/usr/share
/usr/share/doc
/usr/share/doc/unrar
/usr/share/doc/unrar/changelog.Debian.gz
/usr/share/doc/unrar/copyright
/usr/share/man
/usr/share/man/man1
/usr/share/man/man1/unrar-nonfree.1.gz
```

Scoprire quale Pacchetto Detiene un File Specifico

A volte si ha la necessità di sapere a quale pacchetto appartiene un certo file presente sul sistema. Ciò può essere ottenuto usando l'utilità `dpkg-query`, seguita dall'opzione `-S` e dal percorso del file in questione:

```
# dpkg-query -S /usr/bin/unrar-nonfree
unrar: /usr/bin/unrar-nonfree
```

Riconfigurare i Pacchetti Installati

Quando viene installato un pacchetto esiste un passaggio di configurazione chiamato *post-install* in cui viene eseguito uno script per impostare tutto il necessario per l'esecuzione del software, per esempio autorizzazioni, posizionamento dei file di configurazione, ecc. Questo può anche porre alcune domande all'utente per impostare le preferenze su come verrà eseguito il software.

A volte, a causa di un file di configurazione corrotto o non valido, si potrebbe voler ripristinare le impostazioni di un pacchetto al suo stato "iniziale". Oppure si potrebbe voler cambiare le risposte date alle domande iniziali di configurazione. Per fare ciò, eseguire l'utilità `dpkg-reconfigure`, seguita dal nome del pacchetto.

Questo programma eseguirà il backup dei vecchi file di configurazione, decomprimerà quelli nuovi nelle directory corrette ed eseguirà lo script *post-install* fornito dal pacchetto, come se il pacchetto fosse stato installato per la prima volta. Prova a riconfigurare il pacchetto `tzdata` con il seguente esempio:

```
# dpkg-reconfigure tzdata
```

Advanced Package Tool (apt)

Advanced Package Tool (APT) è un sistema di gestione dei pacchetti, che include un set di strumenti, che semplifica notevolmente l'installazione, l'aggiornamento, la rimozione e la gestione dei pacchetti. APT offre funzionalità come capacità di ricerca avanzata e risoluzione automatica delle dipendenze.

APT non è un "sostituto" di `dpkg`. Lo si può pensare come un suo "front-end", che ne semplifica l'utilizzo e colmandone le lacune, come la non automatica risoluzione delle dipendenze.

APT funziona di concerto con repository di software che contengono i pacchetti pronti per l'installazione. Tali repository possono essere un server locale o uno remoto o (meno comune) anche un disco CD-ROM.

Le distribuzioni Linux, come Debian e Ubuntu, mantengono i propri repository e altri repository possono essere gestiti da sviluppatori o gruppi di utenti per fornire software ulteriore.

Esistono molte utility che interagiscono con APT, le principali sono:

apt-get

utilizzata per scaricare, installare, aggiornare o rimuovere i pacchetti dal sistema.

apt-cache

utilizzata per eseguire operazioni, come ricerche, nell'indice dei pacchetti.

apt-file

utilizzata per la ricerca di file all'interno dei pacchetti.

C'è anche un'utilità "più amichevole" chiamata semplicemente `apt`, che combina le opzioni più utilizzate di `apt-get` e `apt-cache` in un'unica utility. Molti dei comandi per `apt` sono gli stessi di `apt-get`, quindi sono in molti casi intercambiabili. Tuttavia, poiché `apt` potrebbe non essere installato sul sistema, si consiglia di imparare ad usare `apt-get` e `apt-cache`.

NOTE

`apt` e `apt-get` potrebbero richiedere una connessione di rete, poiché i pacchetti e gli indici relativi potrebbero dover essere scaricati da un server remoto.

Aggiornare l'Indice dei Pacchetti

Prima di installare o aggiornare il software con APT, si consiglia di aggiornare l'indice dei pacchetti al fine di recuperare informazioni sui pacchetti nuovi e aggiornati. Questo viene fatto con il comando `apt-get`, seguito dal parametro `update`:

```
# apt-get update
Ign:1 http://dl.google.com/linux/chrome/deb stable InRelease
Hit:2 https://repo.skype.com/deb stable InRelease
Hit:3 http://us.archive.ubuntu.com/ubuntu disco InRelease
Hit:4 http://repository.spotify.com stable InRelease
Hit:5 http://dl.google.com/linux/chrome/deb stable Release
Hit:6 http://apt.pop-os.org/proprietary disco InRelease
Hit:7 http://ppa.launchpad.net/system76/pop/ubuntu disco InRelease
Hit:8 http://us.archive.ubuntu.com/ubuntu disco-security InRelease
Hit:9 http://us.archive.ubuntu.com/ubuntu disco-updates InRelease
Hit:10 http://us.archive.ubuntu.com/ubuntu disco-backports InRelease
Reading package lists... Done
```

TIP

Invece di `apt-get update`, si può utilizzare `apt update`.

Installare e Rimuovere Pacchetti

Con l'indice dei pacchetti aggiornato, è ora possibile installare un pacchetto. Questo viene fatto

con `apt-get install`, seguito dal nome del pacchetto che si desidera installare:

```
# apt-get install xournal
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  xournal
0 upgraded, 1 newly installed, 0 to remove and 75 not upgraded.
Need to get 285 kB of archives.
After this operation, 1041 kB of additional disk space will be used.
```

Allo stesso modo, per rimuovere un pacchetto usare `apt-get remove`, seguito dal nome del pacchetto:

```
# apt-get remove xournal
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages will be REMOVED:
  xournal
0 upgraded, 0 newly installed, 1 to remove and 75 not upgraded.
After this operation, 1041 kB disk space will be freed.
Do you want to continue? [Y/n]
```

Tenere presente che durante l'installazione o la rimozione di pacchetti, APT eseguirà la risoluzione automatica delle dipendenze. Ciò significa che *verranno installati anche* eventuali pacchetti aggiuntivi richiesti dal pacchetto che si sta installando e che verranno rimossi anche i pacchetti che dipendono dal pacchetto che si sta rimuovendo. APT mostrerà sempre cosa si stà per installare o rimuovere prima che ciò avvenga effettivamente:

```
# apt-get remove p7zip
Reading package lists... Done
Building dependency tree
The following packages will be REMOVED:
  android-libbacktrace android-libunwind android-libutils
  android-libziparchive android-sdk-platform-tools fastboot p7zip p7zip-full
0 upgraded, 0 newly installed, 8 to remove and 75 not upgraded.
After this operation, 6545 kB disk space will be freed.
Do you want to continue? [Y/n]
```

Quando un pacchetto viene rimosso i file di configurazione corrispondenti vengono lasciati sul sistema. Se si vuole rimuovere *qualsiasi cosa* associata al pacchetto, usare il parametro `purge` invece di `remove` o il parametro `remove` con l'opzione `--purge`:

```
# apt-get purge p7zip
```

oppure

```
# apt-get remove --purge p7zip
```

TIP | Puoi anche usare `apt install` e `apt remove`.

Riparare Dipendenze Errate

È possibile avere “dipendenze errate” su un sistema. Ciò significa che uno o più pacchetti installati dipendono da altri pacchetti che non sono stati installati o che non sono più presenti. Ciò può accadere a causa di un errore APT o di un pacchetto installato manualmente.

Per risolvere questo problema, usare il comando `apt-get install -f`. Questo tenterà di “riparare” i pacchetti malfunzionanti installando le dipendenze mancanti, assicurando quindi che tutti i pacchetti siano nuovamente coerenti.

TIP | È possibile usare anche `apt install -f`.

Aggiornare i Pacchetti

APT può essere utilizzato per aggiornare automaticamente tutti i pacchetti installati alle ultime versioni disponibili dai repository. Questo viene fatto con il comando `apt-get upgrade`. Prima di eseguirlo, aggiornare innanzitutto l'indice del pacchetto con `apt-get update`:

```
# apt-get update
Hit:1 http://us.archive.ubuntu.com/ubuntu disco InRelease
Hit:2 http://us.archive.ubuntu.com/ubuntu disco-security InRelease
Hit:3 http://us.archive.ubuntu.com/ubuntu disco-updates InRelease
Hit:4 http://us.archive.ubuntu.com/ubuntu disco-backports InRelease
Reading package lists... Done

# apt-get upgrade
Reading package lists... Done
Building dependency tree
```

```
Reading state information... Done
Calculating upgrade... Done
The following packages have been kept back:
  gnome-control-center
The following packages will be upgraded:
  cups cups-bsd cups-client cups-common cups-core-drivers cups-daemon
  cups-ipp-utils cups-ppdc cups-server-common firefox-locale-ar (...)

74 upgraded, 0 newly installed, 0 to remove and 1 not upgraded.
Need to get 243 MB of archives.
After this operation, 30.7 kB of additional disk space will be used.
Do you want to continue? [Y/n]
```

Il riepilogo alla fine dell'output mostra quanti pacchetti verranno aggiornati, quanti verranno installati, rimossi o conservati, le dimensioni totali del download e la quantità di spazio su disco aggiuntiva necessaria per completare l'operazione. Per completare l'aggiornamento, basta rispondere **Y** e attendere che `apt-get` finisca l'attività.

Per aggiornare un singolo pacchetto, basta eseguire `apt-get upgrade` seguito dal nome del pacchetto. Come in `dpkg`, `apt-get` controlla prima se è installata una versione precedente di un pacchetto. In tal caso, il pacchetto verrà aggiornato alla versione più recente disponibile nel repository. In caso contrario, verrà installata una nuova copia.

TIP | È possibile anche usare `apt upgrade` e `apt update`.

La Cache Locale

Quando si installa o si aggiorna un pacchetto, il corrispondente file `.deb` viene scaricato in una directory locale di cache. Di default, questa directory è `/var/cache/apt/archives`. I file parzialmente scaricati vengono invece copiati in `/var/cache/apt/archives/partial/`.

Durante l'installazione e l'aggiornamento dei pacchetti, la directory della cache può diventare piuttosto grande. Per recuperare spazio, è possibile svuotarla usando il comando `apt-get clean`. Questo rimuoverà il contenuto delle directory `/var/cache/apt/archives` e `/var/cache/apt/archives/partial/`.

TIP | È possibile utilizzare anche `apt clean`.

Ricerca Pacchetti

L'utility `apt-cache` può essere usata per eseguire operazioni sull'indice dei pacchetti, come cercare un pacchetto specifico o elencare quali pacchetti contengono un file specifico.

Per condurre una ricerca, usare `apt-cache search` seguito da una maschera di ricerca. L'output sarà un elenco di ogni pacchetto che soddisfa il pattern, nel nome, nella descrizione o nei file forniti con il pacchetto.

```
# apt-cache search p7zip
liblzma-dev - XZ-format compression library - development files
liblzma5 - XZ-format compression library
forensics-extra - Forensics Environment - extra console components (metapackage)
p7zip - 7zr file archiver with high compression ratio
p7zip-full - 7z and 7za file archivers with high compression ratio
p7zip-rar - non-free rar module for p7zip
```

Nell'esempio sopra, la voce `liblzma5 - XZ-format compression library` non sembra corrispondere alla maschera di ricerca. Tuttavia, se mostriamo le informazioni complete, inclusa la descrizione, usando il parametro `show`, troveremo il perché di questa visualizzazione:

```
# apt-cache show liblzma5
Package: liblzma5
Architecture: amd64
Version: 5.2.4-1
Multi-Arch: same
Priority: required
Section: libs
Source: xz-utils
Origin: Ubuntu
Maintainer: Ubuntu Developers <ubuntu-devel-discuss@lists.ubuntu.com>
Original-Maintainer: Jonathan Nieder <jrnieder@gmail.com>
Bugs: https://bugs.launchpad.net/ubuntu/+filebug
Installed-Size: 259
Depends: libc6 (>= 2.17)
Breaks: liblzma2 (<< 5.1.1alpha+20110809-3~)
Filename: pool/main/x/xz-utils/liblzma5_5.2.4-1_amd64.deb
Size: 92352
MD5sum: 223533a347dc76a8cc9445cfc6146ec3
SHA1: 8ed14092fb1caecfebc556fda0745e1e74ba5a67
SHA256: 01020b5a0515dbc9a7c00b464a65450f788b0258c3fbb733ecad0438f5124800
Homepage: https://tukaani.org/xz/
Description-en: XZ-format compression library
XZ is the successor to the Lempel-Ziv/Markov-chain Algorithm
compression format, which provides memory-hungry but powerful
compression (often better than bzip2) and fast, easy decompression.
.
```

The native format of liblzma is XZ; it also supports raw (headerless) streams and the older LZMA format used by lzma. (For 7-Zip's related format, use the **p7zip** package instead.)

È possibile utilizzare *espressioni regolari* con il pattern di ricerca, consentendo ricerche molto complesse (e precise). Tuttavia, questo argomento non rientra nell'ambito di questa lezione.

TIP È possibile anche usare `apt search` invece di `apt-cache search` e `apt show` invece di `apt-cache show`.

La Lista delle Fonti

APT utilizza un elenco di fonti per sapere da dove ottenere i pacchetti. Questo elenco è memorizzato nel file `sources.list`, situato nella directory `/etc/apt`. Questo può essere modificato utilizzando un normale editor di testo come `vi`, `pico` o `nano`, o attraverso un'utilità grafica come `aptitude` o `synaptic`.

Una linea tipica all'interno di `sources.list` è simile a questa:

```
deb http://us.archive.ubuntu.com/ubuntu/ disco main restricted universe multiverse
```

La sintassi è: Tipo archivio, URL, Distribuzione e uno o più Componenti, dove:

Tipo archivio

Un repository può contenere pacchetti con software pronto per l'esecuzione (pacchetti binari, indicati come `deb`) o con il codice sorgente (pacchetti sorgente, indicati come `deb-src`). L'esempio sopra fornisce pacchetti binari.

URL

La URL del repository.

Distribuzione

Il nome (o nome in codice) della distribuzione per cui vengono forniti i pacchetti. Un repository può ospitare pacchetti per più distribuzioni. Nell'esempio sopra, `disco` è il nome in codice di Ubuntu 19.04 *Disco Dingo*.

Componenti

Ogni componente rappresenta un insieme di pacchetti. Questi componenti possono essere diversi su ogni distribuzione Linux. Per esempio, su Ubuntu e derivati, sono:

main

contiene pacchetti ufficialmente supportati di tipo open source.

restricted

contiene software ufficialmente supportato a codice sorgente chiuso, per esempio i driver di dispositivo per alcune schede grafiche.

universe

contiene software open source gestito dalla comunità.

multiverse

contiene software non supportato, a codice sorgente chiuso o protetto da brevetto. Su Debian, i componenti principali sono:

main

è costituito da pacchetti conformi alle *Debian Free Software Guidelines* (DFSG), che non si basano su software al di fuori di quest'area per operare. I pacchetti inclusi qui sono considerati parte della distribuzione Debian.

contrib

contiene pacchetti conformi a DFSG, ma che dipendono da altri pacchetti che non siano in **main**.

non-free

contiene pacchetti non conformi al DFSG.

security

contiene aggiornamenti di sicurezza.

backports

contiene versioni più recenti di pacchetti che si trovano in **main**. Il ciclo di sviluppo delle versioni stabili di Debian è piuttosto lungo (circa due anni), e questo consente agli utenti di ottenere pacchetti più aggiornati senza dover modificare il repository core **main**.

NOTE

Per saperne di più sulle *Debian Free Software Guidelines*: https://www.debian.org/social_contract#guidelines

Per aggiungere un nuovo repository da cui ottenere i pacchetti, si può semplicemente aggiungere la riga corrispondente (di solito fornita dal manutentore del repository) alla fine di `sources.list`, salvare il file e ricaricare l'indice dei pacchetti con `apt-get update`. A seguito di ciò, i pacchetti del nuovo repository saranno disponibili per l'installazione usando `apt-get`

`install.`

Tenere presente che le righe che iniziano con il carattere `#` sono considerate commenti e vengono ignorate.

La Directory `/etc/apt/sources.list.d`

All'interno della directory `/etc/apt/sources.list.d` è possibile aggiungere file di repository aggiuntivi che devono essere usati da APT, senza la necessità di modificare il file principale `/etc/apt/sources.list`. Questi sono semplici file di testo, con la stessa sintassi descritta sopra e l'estensione del file `.list`.

Di seguito il contenuto di un file chiamato `/etc/apt/sources.list.d/buster-backports.list`:

```
deb http://deb.debian.org/debian buster-backports main contrib non-free
deb-src http://deb.debian.org/debian buster-backports main contrib non-free
```

Elencare i Contenuti di un Pacchetto e Ricerare dei File

Un'utilità chiamata `apt-file` può essere usata per eseguire più operazioni nell'indice del pacchetto, come elencare il contenuto di un pacchetto o trovare un pacchetto che contiene un file specifico. Questa utility potrebbe non essere installata per impostazione predefinita nel sistema. In questo caso, di solito è possibile installarlo usando `apt-get`:

```
# apt-get install apt-file
```

Dopo l'installazione, sarà necessario aggiornare la cache dei pacchetti:

```
# apt-file update
```

Questo di solito richiede solo pochi secondi. Dopodiché, si è pronti per usare `apt-file`.

Per elencare il contenuto di un pacchetto, usare il parametro `list` seguito dal nome del pacchetto:

```
# apt-file list unrar
unrar: /usr/bin/unrar-nonfree
unrar: /usr/share/doc/unrar/changelog.Debian.gz
unrar: /usr/share/doc/unrar/copyright
```

```
unrar: /usr/share/man/man1/unrar-nonfree.1.gz
```

TIP | È possibile usare `apt list` invece di `apt-file list`.

Si può cercare un file in tutti i pacchetti usando il parametro `search`, seguito dal nome del file. Per esempio, se si desidera sapere quale pacchetto contiene un file chiamato `libSDL2.so`, è possibile utilizzare:

```
# apt-file search libSDL2.so
libSDL2-dev: /usr/lib/x86_64-linux-gnu/libSDL2.so
```

La risposta è il pacchetto `libSDL2-dev`, che fornisce il file `/usr/lib/x86_64-linux-gnu/libSDL2.so`.

La differenza tra `apt-file search` e `dpkg-query` è che `apt-file search` cercherà anche i pacchetti disinstallati, mentre `dpkg-query` può mostrare solo i file che appartengono a un pacchetto già installato.

Esercizi Guidati

1. Qual è il comando per installare un pacchetto chiamato `package.deb` usando `dpkg`?

2. Utilizzando `dpkg-query`, trovare quale pacchetto contiene un file chiamato `7zr.1.gz`.

3. Si può rimuovere un pacchetto chiamato `unzip` dal sistema usando `dpkg -r unzip` se il pacchetto `file-roller` dipende da esso? In caso contrario, quale sarebbe il modo corretto di farlo?

4. Usando `apt-file`, come si può scoprire quale pacchetto contiene il file `unrar`?

5. Usando `apt-cache`, qual è il comando completo per mostrare informazioni sul pacchetto `gimp`?

Esercizi Esplorativi

1. Si consideri un repository con pacchetti sorgente Debian per la distribuzione Ubuntu xenial, ospitato su `http://us.archive.ubuntu.com/ubuntu/` e con pacchetti per il componente universe. Quale sarebbe la riga corrispondente da aggiungere a `/etc/apt/sources.list`?
2. Durante la compilazione di un programma, viene visualizzato un messaggio di errore in cui si afferma che il file di intestazione `zzip-io.h` non è presente nel sistema. Come si può scoprire quale pacchetto fornisce questo file?
3. Come si può ignorare un avviso di dipendenza e rimuovere un pacchetto usando `dpkg`, anche se ci sono pacchetti che dipendono da esso nel sistema?
4. Come si può ottenere maggiori informazioni su un pacchetto chiamato `midori` usando `apt`?
5. Prima di installare o aggiornare i pacchetti con `apt`, quale comando dovrebbe essere usato per assicurarsi che l'indice dei pacchetti sia aggiornato?

Sommario

In questa lezione abbiamo imparato:

- Come usare `dpkg` per installare e rimuovere i pacchetti.
- Come elencare i pacchetti installati e il contenuto dei pacchetti.
- Come riconfigurare un pacchetto installato.
- Che cos'è `apt` e come installare, aggiornare e rimuovere i pacchetti utilizzandolo.
- Come usare `apt-cache` per cercare pacchetti.
- Come è strutturato il file `/etc/apt/sources.list`.
- Come usare `apt-file` per mostrare il contenuto di un pacchetto o come trovare quale pacchetto contiene un file specifico.

Sono stati discussi i seguenti file, termini e utilità:

`dpkg -i`

Installa un singolo pacchetto, o un elenco di pacchetti separati da spazi.

`dpkg -r`

Rimuove un pacchetto o un elenco di pacchetti separato da spazi.

`dpkg -I`

Ispeziona un pacchetto, fornendo dettagli sul software che ha installato e su eventuali dipendenze necessarie.

`dpkg --get-selections`

Elenca tutti i pacchetti che `dpkg` ha installato sul sistema.

`dpkg -L`

Stampa un elenco di tutti i file installati da un determinato pacchetto.

`dpkg-query`

Con un nome file specificato, questo comando stampa il pacchetto che ha installato il file.

`dpkg-reconfigure`

Questo comando esegue nuovamente uno script *post-install*.

apt-get update

Questo comando aggiorna l'indice dei pacchetti in modo che corrisponda a ciò che è disponibile all'interno dei repository configurati nella directory `/etc/apt/`.

apt-get install

Questo comando scarica un pacchetto da un repository remoto e lo installa insieme alle sue dipendenze, può anche essere usato per installare un pacchetto Debian che è già stato scaricato.

apt-get remove

Questo comando disinstalla i pacchetti specificati dal sistema.

apt-cache show

Proprio come il comando `dpkg -I`, questo comando può essere usato per mostrare i dettagli su un pacchetto specifico.

apt-cache search

Questo comando cerca nel database cache di APT un pacchetto particolare.

apt-file update

Questo comando aggiorna la cache dei pacchetti in modo che il comando `apt-file` possa interrogarne il contenuto.

apt-file search

Questo comando può cercare il nome di un pacchetto che ha installato un determinato file, proprio come il comando `dpkg-query`.

apt-file list

Questo comando è usato per elencare il contenuto di un pacchetto, proprio come il comando `dpkg -L`.

Risposte agli Esercizi Guidati

1. Qual è il comando per installare un pacchetto chiamato `package.deb` usando `dpkg`?

Passare l'opzione `-i` a `dpkg`:

```
# dpkg -i package.deb
```

2. Utilizzando `dpkg-query`, trovare quale pacchetto contiene un file chiamato `7zr.1.gz`.

Passare l'opzione `-S` a `dpkg-query`:

```
# dpkg-query -S 7zr.1.gz
```

3. Si può rimuovere un pacchetto chiamato `unzip` dal sistema usando `dpkg -r unzip` se il pacchetto `file-roller` dipende da esso? In caso contrario, quale sarebbe il modo corretto di farlo?

No. `dpkg` non risolverà le dipendenze e non permetterà di rimuovere un pacchetto se un altro pacchetto installato dipende da esso. In questo esempio, si dovrebbe prima rimuovere `file-roller` (supponendo che nulla dipenda da esso) e quindi rimuovere `unzip`, oppure rimuoverli entrambi contemporaneamente con:

```
# dpkg -r unzip file-roller
```

4. Come si può sapere quale pacchetto contiene il file `/usr/bin/unrar` usando l'utilità `apt-file`?

Usare il parametro `search` seguito dal percorso (o nome file):

```
# apt-file search /usr/bin/unrar
```

5. Usando `apt-cache`, qual è il comando completo per mostrare informazioni sul pacchetto `gimp`?

Usare il parametro `show` seguito dal nome del pacchetto:

```
# apt-cache show gimp
```


Risposte agli Esercizi Esplorativi

1. Si consideri un repository con pacchetti sorgente Debian per la distribuzione Ubuntu `xenial`, ospitato su `http://us.archive.ubuntu.com/ubuntu/` e con pacchetti per il componente `universe`. Quale sarebbe la riga corrispondente da aggiungere a `/etc/apt/sources.list`?

I pacchetti sorgente sono del tipo `deb-src`, quindi la linea dovrebbe essere:

```
deb-src http://us.archive.ubuntu.com/ubuntu/ xenial universe
```

Questa riga potrebbe anche essere aggiunta all'interno di un file `.list` in `/etc/apt/sources.list.d/`. Il nome dipende da te ma dovrebbe essere descrittivo, qualcosa come `xenial_sources.list`.

2. Durante la compilazione di un programma, viene visualizzato un messaggio di errore in cui si afferma che il file di intestazione `zzip-io.h` non è presente nel sistema. Come si può scoprire quale pacchetto fornisce questo file?

Usare `apt-file search` per trovare quale pacchetto contiene un file non presente nel sistema:

```
# apt-file search zzip-io.h
```

3. Come si può ignorare un avviso di dipendenza e rimuovere un pacchetto usando `dpkg`, anche se ci sono pacchetti che dipendono da esso nel sistema?

È possibile utilizzare il parametro `--force`, ma ciò non dovrebbe mai essere fatto a meno che non si sappia esattamente cosa si sta facendo, poiché esiste il rischio che il sistema venga lasciato in uno stato incoerente o "rotto".

4. Come si possono ottenere maggiori informazioni su un pacchetto chiamato `midori` usando `apt-cache`?

Usare `apt-cache show` seguito dal nome del pacchetto:

```
# apt-cache show midori
```

5. Prima di installare o aggiornare i pacchetti con `apt`, quale comando dovrebbe essere usato per assicurarsi che l'indice dei pacchetti sia aggiornato?

Si dovrebbe usare `apt-get update`. Questo scaricherà gli indici dei pacchetti più recenti dai repository descritti nel file `/etc/apt/sources.list` e nei file presenti nella directory `/etc/apt/sources.list.d/`.



102.5 Utilizzare la gestione dei pacchetti RPM e YUM

Obiettivi LPI di riferimento

LPIC-1 v5, Exam 101, Objective 102.5

Peso

3

Aree di Conoscenza Chiave

- Installare, reinstallare, aggiornare e rimuovere i pacchetti usando RPM, YUM e Zypper.
- Ottenere informazioni sui pacchetti RPM come versione, stato, dipendenze, integrità e firme.
- Determinare quali file fornisce un pacchetto; trovare da quale pacchetto proviene un file specifico.
- Conoscenza di dnf.

Di seguito è riportato un elenco parziale dei file, dei termini e dei comandi utilizzati

- rpm
- rpm2cpio
- /etc/yum.conf
- /etc/yum.repos.d/
- yum
- zypper



102.5 Lezione 1

Certificazione:	LPIC-1
Versione:	5.0
Argomento:	102 L'Installazione di Linux e la Gestione Pacchetti
Obiettivo:	102.5 Utilizzare la Gestione Pacchetti RPM e YUM
Lezione:	1 di 1

Introduzione

Molto tempo fa, quando Linux era ancora agli inizi, il modo più comune per distribuire il software era un file compresso (di solito un archivio `.tar.gz`) con codice sorgente, che si sarebbe decompresso e compilato per conto proprio.

Tuttavia, con l'aumentare della quantità e della complessità del software, divenne evidente la necessità di un modo per distribuire software precompilato. Dopotutto, non tutti avevano le risorse, sia in termini di tempo sia di potenza di elaborazione, per compilare grandi progetti come il kernel Linux o un X Server.

Presto sono cresciuti gli sforzi per standardizzare la distribuzione di questi “pacchetti” software e sono nati i primi gestori di pacchetti. Questi strumenti hanno reso molto più semplice l'installazione, la configurazione o la rimozione di software da un sistema.

Uno di questi era l' *RPM Package Manager* e il suo strumento corrispondente (`rpm`), sviluppato da Red Hat. Oggi entrambi sono ampiamente utilizzati non solo su Red Hat Enterprise Linux (RHEL)

stesso, ma anche su molte delle sue derivate come Fedora, CentOS e Oracle Linux, altre distribuzioni come openSUSE e persino altri sistemi operativi, come IBM AIX.

Altri strumenti di gestione dei pacchetti popolari nelle distribuzioni compatibili con Red Hat sono `yum` (YellowDog Updater Modified), `dnf` (Dandified YUM) e `zypper`, che possono semplificare molti degli aspetti dell'installazione, manutenzione e rimozione dei pacchetti, rendendo la gestione dei pacchetti molto più semplice.

In questa lezione impareremo a utilizzare `rpm`, `yum`, `dnf` e `zypper` per ottenere, installare, gestire e rimuovere software su un sistema Linux.

NOTE

Nonostante utilizzi lo stesso formato di pacchetto, esistono differenze interne tra le distribuzioni, quindi un pacchetto creato appositamente per openSUSE potrebbe non funzionare su un sistema RHEL e viceversa. Quando si cercano pacchetti, va verificata sempre la compatibilità e se possibile occorre trovarne uno realizzato specificatamente per la distribuzione in uso.

Il Gestore di Pacchetti RPM (rpm)

RPM Package Manager (`rpm`) è lo strumento essenziale per la gestione di pacchetti software su sistemi basati su Red Hat (o derivati).

Installare, Aggiornare e Rimuovere Pacchetti

L'operazione più semplice è installare un pacchetto, che può essere fatto attraverso:

```
# rpm -i PACKAGENAME
```

Dove `PACKAGENAME` è il nome del pacchetto `.rpm` che si desidera installare.

Se esiste una versione precedente di un pacchetto sul sistema, è possibile eseguire l'aggiornamento a una versione più recente usando il parametro `-U`:

```
# rpm -U PACKAGENAME
```

Se non è installata una versione precedente di `PACKAGENAME`, verrà installata una nuova copia. Per evitare ciò e quindi *solo* aggiornare un pacchetto già *installato*, usare l'opzione `-F`.

In entrambe le operazioni è possibile aggiungere il parametro `-v` per ottenere un output dettagliato (durante l'installazione vengono visualizzate ulteriori informazioni) e `-h` per ottenere

una barra di avanzamento durante l'installazione fatta da tanti singoli caratteri di hash (#). Più parametri possono essere combinati in uno, quindi `rpm -i -v -h` è lo stesso di `rpm -ivh`.

Per rimuovere un pacchetto installato, passare il parametro `-e` (come in “erase”) a `rpm`, seguito dal nome del pacchetto che si desidera rimuovere:

```
# rpm -e wget
```

Se un pacchetto installato dipende dal pacchetto rimosso, verrà visualizzato un messaggio di errore:

```
# rpm -e unzip
error: Failed dependencies:
    /usr/bin/unzip is needed by (installed) file-roller-3.28.1-2.el7.x86_64
```

Per completare l'operazione, per prima cosa rimuovere i pacchetti che dipendono da quello che si vuole rimuovere (nell'esempio sopra, `file-roller`). È possibile passare più nomi di pacchetti a `rpm -e` per rimuovere più pacchetti contemporaneamente.

Gestire le Dipendenze

Molto spesso un pacchetto può dipendere da altri per funzionare a dovere. Per esempio, un editor di immagini potrebbe aver bisogno di librerie per aprire file JPEG o un'altra utility potrebbe aver bisogno di un widget toolkit come Qt o GTK per la sua interfaccia utente.

`rpm` controllerà se queste dipendenze sono installate sul sistema e non installerà il pacchetto in mancanza di esse. In questo caso, `rpm` elencherà quali pacchetti mancano. Tuttavia, *non può risolvere* le dipendenze da solo.

Nell'esempio seguente, l'utente tenta di installare il pacchetto dell'editor di immagini GIMP, ma mancano alcune dipendenze:

```
# rpm -i gimp-2.8.22-1.el7.x86_64.rpm
error: Failed dependencies:
    babl(x86-64) >= 0.1.10 is needed by gimp-2:2.8.22-1.el7.x86_64
    gegl(x86-64) >= 0.2.0 is needed by gimp-2:2.8.22-1.el7.x86_64
    gimp-libs(x86-64) = 2:2.8.22-1.el7 is needed by gimp-2:2.8.22-1.el7.x86_64
    libbabl-0.1.so.0()(64bit) is needed by gimp-2:2.8.22-1.el7.x86_64
    libgegl-0.2.so.0()(64bit) is needed by gimp-2:2.8.22-1.el7.x86_64
    libgimp-2.0.so.0()(64bit) is needed by gimp-2:2.8.22-1.el7.x86_64
    libgimpbase-2.0.so.0()(64bit) is needed by gimp-2:2.8.22-1.el7.x86_64
```

```
libgimpcolor-2.0.so.0()(64bit) is needed by gimp-2:2.8.22-1.el7.x86_64
libgimpconfig-2.0.so.0()(64bit) is needed by gimp-2:2.8.22-1.el7.x86_64
libgimpmath-2.0.so.0()(64bit) is needed by gimp-2:2.8.22-1.el7.x86_64
libgimpmodule-2.0.so.0()(64bit) is needed by gimp-2:2.8.22-1.el7.x86_64
libgimpthumb-2.0.so.0()(64bit) is needed by gimp-2:2.8.22-1.el7.x86_64
libgimpui-2.0.so.0()(64bit) is needed by gimp-2:2.8.22-1.el7.x86_64
libgimpwidgets-2.0.so.0()(64bit) is needed by gimp-2:2.8.22-1.el7.x86_64
libmng.so.1()(64bit) is needed by gimp-2:2.8.22-1.el7.x86_64
libwmf-0.2.so.7()(64bit) is needed by gimp-2:2.8.22-1.el7.x86_64
libwmflite-0.2.so.7()(64bit) is needed by gimp-2:2.8.22-1.el7.x86_64
```

Spetta all'utente trovare i pacchetti `.rpm` con le dipendenze corrispondenti e installarli. Gestori di pacchetti come `yum`, `zypper` e `dnf` hanno strumenti che possono dire quale pacchetto fornisce un file specifico. Questi saranno discussi più avanti in questa lezione.

Elencare i Pacchetti Installati

Per ottenere un elenco di tutti i pacchetti installati sul sistema, utilizzare `rpm -qa` (si pensi a “query all”).

```
# rpm -qa
selinux-policy-3.13.1-229.el7.noarch
pciutils-libs-3.5.1-3.el7.x86_64
redhat-menus-12.0.2-8.el7.noarch
grubby-8.28-25.el7.x86_64
hunspell-en-0.20121024-6.el7.noarch
dejavu-fonts-common-2.33-6.el7.noarch
xorg-x11-drv-dummy-0.3.7-1.el7.1.x86_64
libevdev-1.5.6-1.el7.x86_64
[...]
```

Ottenere Informazioni sui Pacchetti

Per ottenere informazioni su un pacchetto *installato*, come il suo numero di versione, l'architettura, la data di installazione, il packager, il sommario, ecc., utilizzare `rpm` con l'opzione `-qi` (si pensi a “query info”), seguito dal nome del pacchetto. Per esempio:

```
# rpm -qi unzip
Name       : unzip
Version    : 6.0
Release    : 19.el7
```

```

Architecture: x86_64
Install Date: Sun 25 Aug 2019 05:14:39 PM EDT
Group       : Applications/Archiving
Size        : 373986
License     : BSD
Signature   : RSA/SHA256, Wed 25 Apr 2018 07:50:02 AM EDT, Key ID 24c6a8a7f4a80eb5
Source RPM  : unzip-6.0-19.el7.src.rpm
Build Date  : Wed 11 Apr 2018 01:24:53 AM EDT
Build Host  : x86-01.bsys.centos.org
Relocations : (not relocatable)
Packager    : CentOS BuildSystem <http://bugs.centos.org>
Vendor      : CentOS
URL         : http://www.info-zip.org/UnZip.html
Summary     : A utility for unpacking zip files
Description :
The unzip utility is used to list, test, or extract files from a zip
archive. Zip archives are commonly found on MS-DOS systems. The zip
utility, included in the zip package, creates zip archives. Zip and
unzip are both compatible with archives created by PKWARE(R)'s PKZIP
for MS-DOS, but the programs' options and default behaviors do differ
in some respects.

Install the unzip package if you need to list, test or extract files from
a zip archive.

```

Per ottenere un elenco di quali file si trovano all'interno di un pacchetto *installato* usare l'opzione `-ql` (si pensi a “query list”) seguito dal nome del pacchetto:

```

# rpm -ql unzip
/usr/bin/funzip
/usr/bin/unzip
/usr/bin/unzipsfx
/usr/bin/zipgrep
/usr/bin/zipinfo
/usr/share/doc/unzip-6.0
/usr/share/doc/unzip-6.0/BUGS
/usr/share/doc/unzip-6.0/LICENSE
/usr/share/doc/unzip-6.0/README
/usr/share/man/man1/funzip.1.gz
/usr/share/man/man1/unzip.1.gz
/usr/share/man/man1/unzipsfx.1.gz
/usr/share/man/man1/zipgrep.1.gz
/usr/share/man/man1/zipinfo.1.gz

```


Se si desidera ottenere informazioni o un elenco di file da un pacchetto che non è stato ancora installato, aggiungere semplicemente l'opzione `-p` ai comandi sopra, seguito dal nome del file RPM (FILENAME). Quindi `rpm -qi PACKAGENAME` diventa `rpm -qip FILENAME`, e `rpm -ql PACKAGENAME` diventa `rpm -qlp FILENAME`, come mostrato sotto.

```
# rpm -qip atom.x86_64.rpm
Name       : atom
Version    : 1.40.0
Release    : 0.1
Architecture: x86_64
Install Date: (not installed)
Group      : Unspecified
Size       : 570783704
License    : MIT
Signature  : (none)
Source RPM : atom-1.40.0-0.1.src.rpm
Build Date : sex 09 ago 2019 12:36:31 -03
Build Host : b01bbeaf3a88
Relocations : /usr
URL        : https://atom.io/
Summary    : A hackable text editor for the 21st Century.
Description :
A hackable text editor for the 21st Century.
```

```
# rpm -qlp atom.x86_64.rpm
/usr/bin/apm
/usr/bin/atom
/usr/share/applications/atom.desktop
/usr/share/atom
/usr/share/atom/LICENSE
/usr/share/atom/LICENSES.chromium.html
/usr/share/atom/atom
/usr/share/atom/atom.png
/usr/share/atom/blink_image_resources_200_percent.pak
/usr/share/atom/content_resources_200_percent.pak
/usr/share/atom/content_shell.pak

(la lista prosegue)
```

Scoprire quale Pacchetto Detiene un File Specifico

Per elencare il contenuto di un pacchetto, usare l'opzione `-qf` (si pensi a “query file”) seguito dal percorso completo del pacchetto:

```
# rpm -qf /usr/bin/unzip
unzip-6.0-19.el7.x86_64
```

Nell'esempio sopra, il file `/usr/bin/unzip` appartiene al pacchetto `unzip-6.0-19.el7.x86_64`.

YellowDog Updater Modified (YUM)

`yum` è stato originariamente sviluppato come *Yellow Dog Updater* (YUP), uno strumento per la gestione dei pacchetti sulla distribuzione Linux di Yellow Dog. Nel tempo, si è evoluto per gestire i pacchetti su altri sistemi basati su RPM, come Fedora, CentOS, Red Hat Enterprise Linux e Oracle Linux.

Funzionalmente è simile all'utilità `apt` sui sistemi basati su Debian, essendo in grado di cercare, installare, aggiornare e rimuovere pacchetti e gestire automaticamente le dipendenze. `yum` può essere usato per installare un singolo pacchetto, o per aggiornare contemporaneamente un intero sistema.

Ricerca Pacchetti

Per installare un pacchetto, è necessario conoscerne il nome. Per questo eseguire una ricerca con `yum search PATTERN`, dove `PATTERN` è il nome del pacchetto che si sta cercando. Il risultato è un elenco di pacchetti il cui nome o riepilogo contiene il pattern di ricerca specificato. Per esempio, se si ha bisogno di un programma di utilità per gestire i file compressi 7Zip (con l'estensione `.7z`) si può usare:

```
# yum search 7zip
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
 * base: mirror.ufscar.br
 * epel: mirror.globo.com
 * extras: mirror.ufscar.br
 * updates: mirror.ufscar.br
===== N/S matchyutr54ed: 7zip =====
p7zip-plugins.x86_64 : Additional plugins for p7zip
p7zip.x86_64 : Very high compression ratio file archiver
p7zip-doc.noarch : Manual documentation and contrib directory
```

```
p7zip-gui.x86_64 : 7zG - 7-Zip GUI version
```

Name and summary matches only, use "search all" for everything.

Installare, Aggiornare e Rimuovere Pacchetti

Per installare un pacchetto usando `yum`, usare il comando `yum install PACKAGENAME`, dove `PACKAGENAME` è il nome del pacchetto. `yum` prenderà il pacchetto e le dipendenze corrispondenti da un repository online e li installerà nel sistema.

```
# yum install p7zip
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
* base: mirror.ufscar.br
* epel: mirror.globo.com
* extras: mirror.ufscar.br
* updates: mirror.ufscar.br
Resolving Dependencies
--> Running transaction check
---> Package p7zip.x86_64 0:16.02-10.el7 will be installed
--> Finished Dependency Resolution
```

Dependencies Resolved

```
=====
Package           Arch           Version           Repository        Size
=====
Installing:
p7zip             x86_64         16.02-10.el7      epel              604 k
```

Transaction Summary

```
=====
Install 1 Package

Total download size: 604 k
Installed size: 1.7 M
Is this ok [y/d/N]:
```

Per aggiornare un pacchetto installato, utilizzare `yum update PACKAGENAME`, dove `PACKAGENAME` è il nome del pacchetto che si desidera aggiornare. Per esempio:

```
# yum update wget
```

```

Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
 * base: mirror.ufscar.br
 * epel: mirror.globo.com
 * extras: mirror.ufscar.br
 * updates: mirror.ufscar.br
Resolving Dependencies
--> Running transaction check
---> Package wget.x86_64 0:1.14-18.el7 will be updated
---> Package wget.x86_64 0:1.14-18.el7_6.1 will be an update
--> Finished Dependency Resolution

Dependencies Resolved

=====
Package      Arch          Version              Repository           Size
=====
Updating:
 wget        x86_64        1.14-18.el7_6.1      updates              547 k

Transaction Summary
=====
Upgrade 1 Package

Total download size: 547 k
Is this ok [y/d/N]:

```

Se si omette il nome del pacchetto, è possibile aggiornare tutti i pacchetti sul sistema per i quali è disponibile un aggiornamento.

Per verificare se è disponibile un aggiornamento per un pacchetto specifico, utilizzare `yum check-update PACKAGENAME`. Come prima, se si omette il nome del pacchetto, `yum` controllerà gli aggiornamenti per ogni pacchetto installato sul sistema.

Per rimuovere un pacchetto installato, usa `yum remove PACKAGENAME`, dove `PACKAGENAME` è il nome del pacchetto che si desidera rimuovere.

Trovare quale Pacchetto Fornisce un File Specifico

In un esempio precedente abbiamo mostrato un tentativo di installare l'editor di immagini `gimp`, fallito a causa di dipendenze non soddisfatte. Tuttavia, `rpm` mostra quali file mancano, ma non elenca il nome dei pacchetti che li forniscono..

Per esempio, una delle dipendenze mancanti era `libgimpui-2.0.so.0`. Per vedere quale pacchetto lo fornisce, è possibile usare `yum whatprovides`, seguito dal nome del file che si sta cercando:

```
# yum whatprovides libgimpui-2.0.so.0
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
* base: mirror.ufscar.br
* epel: mirror.globo.com
* extras: mirror.ufscar.br
* updates: mirror.ufscar.br
2:gimp-libs-2.8.22-1.el7.i686 : GIMP libraries
Repo          : base
Matched from:
Provides      : libgimpui-2.0.so.0
```

La risposta è `gimp-libs-2.8.22-1.el7.i686`. È quindi possibile installare il pacchetto con il comando `yum install gimp-libs`.

Questo funziona anche con i file già presenti nel sistema. Per esempio, se si desidera sapere da dove proviene il file `/etc/hosts`, è possibile utilizzare:

```
# yum whatprovides /etc/hosts
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
* base: mirror.ufscar.br
* epel: mirror.globo.com
* extras: mirror.ufscar.br
* updates: mirror.ufscar.br
setup-2.8.71-10.el7.noarch : A set of system configuration and setup files
Repo          : base
Matched from:
Filename      : /etc/hosts
```

La risposta è `setup-2.8.71-10.el7.noarch`.

Ottenere Informazioni su un Pacchetto

Per ottenere informazioni su un pacchetto, come la sua versione, architettura, descrizione, dimensione e altro, utilizzare `yum info PACKAGENAME` dove `PACKAGENAME` è il nome del pacchetto per il quale si desidereremo informazioni:

```
# yum info firefox
Last metadata expiration check: 0:24:16 ago on Sat 21 Sep 2019 02:39:43 PM -03.
Installed Packages
Name           : firefox
Version        : 69.0.1
Release        : 3.fc30
Architecture   : x86_64
Size           : 268 M
Source         : firefox-69.0.1-3.fc30.src.rpm
Repository     : @System
From repo      : updates
Summary        : Mozilla Firefox Web browser
URL            : https://www.mozilla.org/firefox/
License        : MPLv1.1 or GPLv2+ or LGPLv2+
Description    : Mozilla Firefox is an open-source web browser, designed
                  : for standards compliance, performance and portability.
```

Gestire i Repository Software

Per yum i “repos” sono elencati nella directory `/etc/yum.repos.d/`. Ogni repository è rappresentato da un file `.repo`, come per esempio `CentOS-Base.repo`.

Ulteriori repository possono essere aggiunti dall’utente creando file `.repo` nella directory sopra menzionata, o alla fine di `/etc/yum.conf`. Tuttavia, il modo raccomandato per aggiungere o gestire i repository è con lo strumento `yum-config-manager`.

Per aggiungere un repository, usare l’opzione `--add-repo`, seguita dalla URL del file `.repo`.

```
# yum-config-manager --add-repo https://rpms.remirepo.net/enterprise/remi.repo
Loaded plugins: fastestmirror, langpacks
adding repo from: https://rpms.remirepo.net/enterprise/remi.repo
grabbing file https://rpms.remirepo.net/enterprise/remi.repo to /etc/yum.repos.d/remi.repo
repo saved to /etc/yum.repos.d/remi.repo
```

Per ottenere un elenco di tutti i repository disponibili usare `yum repolist all`. Si otterrà un output simile a questo:

```
# yum repolist all
Loaded plugins: fastestmirror, langpacks
Loading mirror speeds from cached hostfile
* base: mirror.ufscar.br
```

```
* epel: mirror.globo.com
* extras: mirror.ufscar.br
* updates: mirror.ufscar.br
```

repo id	repo name	status
updates/7/x86_64	CentOS-7 - Updates	enabled: 2,500
updates-source/7	CentOS-7 - Updates Sources	disabled

I repository **disabilitati** verranno ignorati durante l'installazione o l'aggiornamento del software. Per abilitare o disabilitare un repository, usare l'utilità `yum-config-manager`, seguita dal repository ID.

Nell'output sopra, il repository ID è mostrato sulla prima colonna (`repo id`) di ogni riga. Utilizzare solo la parte prima del primo `/`, quindi l'ID per il repository `CentOS-7 - Updates` è solo `updates`, e non `updates/7/x86_64`.

```
# yum-config-manager --disable updates
```

Il comando sopra disabilerà il repository `updates`. Per riattivarlo utilizzare:

```
# yum-config-manager --enable updates
```

NOTE

Yum memorizza i pacchetti scaricati e i metadati associati in una directory cache (di solito `/var/cache/yum`). Man mano che il sistema viene aggiornato e vengono installati nuovi pacchetti, questa cache può diventare piuttosto grande. Per pulire la cache e recuperare spazio su disco si può utilizzare il comando `yum clean`, seguito da cosa pulire. I parametri più utili sono `packages` (`yum clean packages`) per eliminare i pacchetti scaricati e i metadati (`yum clean metadata`) per eliminare i metadati associati. Vedere la pagina di manuale di `yum` (digitare `man yum`) per maggiori informazioni.

DNF

`dnf` è lo strumento di gestione dei pacchetti usato su Fedora, ed è un fork di `yum`. Pertanto, molti comandi e parametri sono simili. Questa sezione fornisce solo una rapida panoramica di `dnf`.

Ricerca pacchetti

`dnf search PATTERN`, dove `PATTERN` è ciò che stai cercando. Per esempio, `dnf search unzip` mostrerà tutti i pacchetti che contengono la parola `unzip` nel nome o nella descrizione.

Ottenere informazioni su un pacchetto

```
dnf info PACKAGENAME
```

Installare pacchetti

`dnf install PACKAGENAME`, dove `PACKAGENAME` è il nome del pacchetto che si desidera installare. È possibile trovare il nome eseguendo una ricerca.

Rimuovere pacchetti

```
dnf remove PACKAGENAME
```

Aggiornare pacchetti

`dnf upgrade PACKAGENAME` per aggiornare un solo pacchetto. Omettere il nome del pacchetto per aggiornare tutti i pacchetti nel sistema.

Scoprire quale pacchetto fornisce un file specifico

```
dnf provides FILENAME
```

Ottenere un elenco di tutti i pacchetti installati nel sistema

```
dnf list --installed
```

Elencare il contenuto di un pacchetto

```
dnf repoquery -l PACKAGENAME
```

NOTE

`dnf` ha un sistema di aiuto integrato, che mostra più informazioni (come parametri extra) per ogni comando. Per usarlo, digitare `dnf help` seguito dal comando, come per esempio `dnf help install`.

Gestire i Repository Software

Proprio come con `yum` e `zypper`, `dnf` funziona con i repository di software (repository). Ogni distribuzione ha un elenco di repository predefinito e gli amministratori possono aggiungere o rimuovere repository secondo le loro necessità.

Per ottenere un elenco di tutti i repository disponibili, usare `dnf repolist`. Per elencare solo i repository abilitati, aggiungere l'opzione `--enabled` e per elencare solo i repository disabilitati, aggiungere l'opzione `--disabled`.

```
# dnf repolist
```

```
Last metadata expiration check: 0:20:09 ago on Sat 21 Sep 2019 02:39:43 PM -03.
```

repo id	repo name	status
*fedora	Fedora 30 - x86_64	56,582

*fedora-modular	Fedora Modular 30 - x86_64	135
*updates	Fedora 30 - x86_64 - Updates	12,774
*updates-modular	Fedora Modular 30 - x86_64 - Updates	145

Per aggiungere un repository, usare `dnf config-manager --add-repo URL`, dove URL è l'URL completa del repository. Per abilitare un repository, usare `dnf config-manager --set-enabled REPO_ID`.

Allo stesso modo, per disabilitare un repository usare `dnf config-manager --set-disabled REPO_ID`. In entrambi i casi, REPO_ID è l'ID univoco per il repository, che è possibile ottenere usando `dnf repolist`. I repository aggiunti sono abilitati per impostazione predefinita.

I repository sono memorizzati nei file `.repo` nella directory `/etc/yum.repos.d/`, con esattamente la stessa sintassi usata per `yum`.

Zypper

`zypper` è lo strumento di gestione dei pacchetti utilizzato su SUSE Linux e OpenSUSE. Per quanto riguarda le caratteristiche, è simile a `apt` e `yum`, essendo in grado di installare, aggiornare e rimuovere i pacchetti da un sistema, con risoluzione automatica delle dipendenze.

Aggiornare l'Indice dei Pacchetti

Come altri strumenti di gestione dei pacchetti, `zypper` funziona con repository contenenti pacchetti e metadati. Questi metadati devono essere aggiornati periodicamente, in modo che l'utilità sia a conoscenza degli ultimi pacchetti disponibili. Per eseguire un aggiornamento, digitare semplicemente:

```
# zypper refresh
Repository 'Non-OSS Repository' is up to date.
Repository 'Main Repository' is up to date.
Repository 'Main Update Repository' is up to date.
Repository 'Update Repository (Non-Oss)' is up to date.
All repositories have been refreshed.
```

`zypper` ha una funzione di aggiornamento automatico che può essere abilitata in base al repository, il che significa che alcuni repository possono essere aggiornati automaticamente prima di una query o dell'installazione di un pacchetto, mentre altri potrebbero dover essere aggiornati manualmente. Ciò verrà spiegato a breve.

Ricerca Pacchetti

Per cercare un pacchetto, utilizzare il parametro `search` (o `se`), seguito dal nome del pacchetto:

```
# zypper se gnumeric
Loading repository data...
Reading installed packages...

S | Name | Summary | Type
--+-----+-----+-----
  | gnumeric | Spreadsheet Application | package
  | gnumeric-devel | Spreadsheet Application | package
  | gnumeric-doc | Documentation files for Gnumeric | package
  | gnumeric-lang | Translations for package gnumeric | package
```

Il parametro di ricerca può anche essere utilizzato per ottenere un elenco di tutti i pacchetti installati nel sistema. Per fare ciò, usare l'opzione `-i` senza un nome di pacchetto, come in `zypper se -i`.

Per vedere se è installato un pacchetto specifico, aggiungere il nome del pacchetto al comando sopra. Per esempio, il comando seguente cercherà tra i pacchetti installati tutti quelli con “firefox” nel nome:

```
# zypper se -i firefox
Loading repository data...
Reading installed packages...

S | Name | Summary | Type
--+-----+-----+-----
i | MozillaFirefox | Mozilla Firefox Web B-> | package
i | MozillaFirefox-branding-openSUSE | openSUSE branding of -> | package
i | MozillaFirefox-translations-common | Common translations f-> | package
```

Per cercare solo tra i pacchetti *non installati*, aggiungere l'opzione `-u` all'operatore `se`.

Installare, Aggiornare e Rimuovere Pacchetti

Per installare un pacchetto software, utilizzare il parametro `install` (o `in`), seguito dal nome del pacchetto.:

```
# zypper in unrar
```

```
zypper in unrar
Loading repository data...
Reading installed packages...
Resolving package dependencies...

The following NEW package is going to be installed:
  unrar

1 new package to install.
Overall download size: 141.2 KiB. Already cached: 0 B. After the operation, additional 301.6
KiB will be used.
Continue? [y/n/v/...? shows all options] (y): y
Retrieving package unrar-5.7.5-lp151.1.1.x86_64
                                (1/1), 141.2 KiB (301.6 KiB unpacked)
Retrieving: unrar-5.7.5-lp151.1.1.x86_64.rpm .....[done]
Checking for file conflicts: .....[done]
(1/1) Installing: unrar-5.7.5-lp151.1.1.x86_64 .....[done]
```

zypper può anche essere usato per installare un pacchetto RPM su disco, mentre cerca di soddisfare le sue dipendenze usando pacchetti dai repository. Per fare ciò, basta fornire il percorso completo al pacchetto, come per esempio: `zypper in /home/john/newpackage.rpm`.

Per aggiornare i pacchetti installati sul sistema, utilizzare `zypper update`. Come nel processo di installazione, questo mostrerà un elenco di pacchetti da installare/aggiornare prima di chiedere se si desidera procedere.

Se si desidera elencare solo gli aggiornamenti disponibili, senza installare nulla, è possibile utilizzare: `zypper list-updates`.

Per rimuovere un pacchetto, utilizzare il parametro `remove` (o `rm`), seguito dal nome del pacchetto:

```
# zypper rm unrar
Loading repository data...
Reading installed packages...
Resolving package dependencies...

The following package is going to be REMOVED:
  unrar

1 package to remove.
After the operation, 301.6 KiB will be freed.
Continue? [y/n/v/...? shows all options] (y): y
```

```
(1/1) Removing unrar-5.7.5-lp151.1.1.x86_64 .....[done]
```

Si tenga presente che la rimozione di un pacchetto rimuove anche tutti gli altri pacchetti che dipendono da esso. Per esempio:

```
# zypper rm libgimp-2_0-0
Loading repository data...
Warning: No repositories defined. Operating only with the installed resolvables. Nothing can
be installed.
Reading installed packages...
Resolving package dependencies...

The following 6 packages are going to be REMOVED:
  gimp gimp-help gimp-lang gimp-plugins-python libgimp-2_0-0
  libgimpui-2_0-0

6 packages to remove.
After the operation, 98.0 MiB will be freed.
Continue? [y/n/v/...? shows all options] (y):
```

Trovare quale Pacchetto Contiene un File Specifico

Per vedere quali pacchetti contengono un file specifico, usare il parametro di ricerca seguito dall'opzione `--provides` e dal nome del file (o dal percorso completo aa esso). Per esempio, se si vuole sapere quali pacchetti contengono il file `libgimpmodule-2.0.so.0` in `/usr/lib64/` usare:

```
# zypper se --provides /usr/lib64/libgimpmodule-2.0.so.0
Loading repository data...
Reading installed packages...

S | Name                | Summary                                | Type
--+-+-----+-----+-----+-----+-----+-----+-----+-----+-----+
i | libgimp-2_0-0        | The GNU Image Manipulation Program -  | package
  |                      | Libra->                                |
```

Ottenere Informazioni sui Pacchetti

Per vedere i metadati associati a un pacchetto, usare il parametro `info` seguito dal nome del pacchetto. Questo fornirà il repository di origine, il nome del pacchetto, la versione, l'architettura, il fornitore, le dimensioni installate, se è installato o meno, lo stato (se è aggiornato), il pacchetto di origine e una descrizione.

```
# zypper info gimp
Loading repository data...
Reading installed packages...

Information for package gimp:
-----
Repository      : Main Repository
Name            : gimp
Version        : 2.8.22-lp151.4.6
Arch           : x86_64
Vendor         : openSUSE
Installed Size  : 29.1 MiB
Installed      : Yes (automatically)
Status         : up-to-date
Source package  : gimp-2.8.22-lp151.4.6.src
Summary        : The GNU Image Manipulation Program
Description     :
    The GIMP is an image composition and editing program, which can be
    used for creating logos and other graphics for Web pages. The GIMP
    offers many tools and filters, and provides a large image
    manipulation toolbox, including channel operations and layers,
    effects, subpixel imaging and antialiasing, and conversions, together
    with multilevel undo. The GIMP offers a scripting facility, but many
    of the included scripts rely on fonts that we cannot distribute.
```

Gestire i Repository Software

zypper può anche essere usato per gestire i repository di software. Per vedere un elenco di tutti i repository attualmente registrati nel sistema, usare `zypper repos`:

```
# zypper repos
Repository priorities are without effect. All enabled repositories share the same priority.

# | Alias | Name | Enabled | GPG Check |
Refresh
---+-----+-----+-----+-----+
-----
1 | openSUSE-Leap-15.1-1 | openSUSE-Leap-15.1-1 | No | ---- |
----
2 | repo-debug | Debug Repository | No | ---- |
----
3 | repo-debug-non-oss | Debug Repository (Non-OSS) | No | ---- |
```

```

----
 4 | repo-debug-update          | Update Repository (Debug)          | No      | ----      |
----
 5 | repo-debug-update-non-oss    | Update Repository (Debug, Non-OSS) | No      | ----      |
----
 6 | repo-non-oss                | Non-OSS Repository                 | Yes     | (x ) Yes  |
Yes
 7 | repo-oss                    | Main Repository                    | Yes     | (x ) Yes  |
Yes
 8 | repo-source                 | Source Repository                  | No      | ----      |
----
 9 | repo-source-non-oss          | Source Repository (Non-OSS)        | No      | ----      |
----
10 | repo-update                 | Main Update Repository              | Yes     | (x ) Yes  |
Yes
11 | repo-update-non-oss          | Update Repository (Non-Oss)        | Yes     | (x ) Yes  |
Yes

```

Si può notare che nella colonna `Enabled` alcuni archivi sono abilitati, mentre altri no. Ciò può essere modificato con il parametro `editrepo`, seguito dall'opzione `-e` (enable) o `-d` (disable) e dall'alias del repository (la seconda colonna nell'output).

```

# zypper modifyrepo -d repo-non-oss
Repository 'repo-non-oss' has been successfully disabled.

# zypper modifyrepo -e repo-non-oss
Repository 'repo-non-oss' has been successfully enabled.

```

In precedenza si è detto che `zypper` ha una funzionalità di *auto refresh* che può essere abilitata in base al repository. Quando abilitato, questo flag farà eseguire a `zypper` un'operazione di aggiornamento (la stessa di `zypper refresh`) prima di lavorare con il repository specificato. Questo può essere controllato con le opzioni `-f` e `-F` del parametro `editrepo`:

```

# zypper modifyrepo -F repo-non-oss
Autorefresh has been disabled for repository 'repo-non-oss'.

# zypper modifyrepo -f repo-non-oss
Autorefresh has been enabled for repository 'repo-non-oss'.

```

Aggiungere e Rimuovere Repositories

Per aggiungere un nuovo repository di software in `zypper`, utilizzare il parametro `addrepo` seguito dall'URL del repository e dal nome del repository, come di seguito:

```
# zypper addrepo http://packman.inode.at/suse/openSUSE_Leap_15.1/ packman
Adding repository 'packman' .....[done]
Repository 'packman' successfully added

URI          : http://packman.inode.at/suse/openSUSE_Leap_15.1/
Enabled      : Yes
GPG Check    : Yes
Autorefresh  : No
Priority     : 99 (default priority)

Repository priorities are without effect. All enabled repositories share the same priority.
```

Durante l'aggiunta di un repository, è possibile abilitare gli aggiornamenti automatici con il l'opzione `-f`. I repository aggiunti sono abilitati di default, ma è possibile aggiungere e disabilitare un repository contemporaneamente usando l'opzione `-d`.

Per rimuovere un repository, usare il parametro `removerepo`, seguito dal nome del repository (Alias). Per rimuovere il repository aggiunto nell'esempio sopra, il comando sarà:

```
# zypper removerepo packman
Removing repository 'packman' .....[done]
Repository 'packman' has been removed.
```

Esercizi Guidati

1. Usando `rpm` su un sistema Red Hat Enterprise Linux, come installereste il pacchetto `file-roller-3.28.1-2.el7.x86_64.rpm` tanto da fargli mostrare una barra di avanzamento durante l'installazione?
2. Usando `rpm`, scopri quale pacchetto contiene il file `/etc/redhat-release`.
3. Come useresti `yum` per verificare la presenza di aggiornamenti per tutti i pacchetti nel sistema?
4. Usando `zypper`, come disabilitaresti un repository chiamato `repo-extras`?
5. Se hai un file `.repo` che descrive un nuovo repository, dove dovrebbe essere messo questo file in modo che venga riconosciuto da DNF?

Esercizi Esplorativi

1. Come useresti zypper per scoprire quale pacchetto detiene il file `/usr/sbin/swapon`?

2. Come puoi ottenere un elenco di tutti i pacchetti installati nel sistema usando dnf?

3. Usando dnf, qual è il comando per aggiungere un repository situato in <https://www.example.url/home:reponame.repo> al sistema?

4. Come puoi usare zypper per verificare se il pacchetto unzip è installato?

5. Usando yum, scopri quale pacchetto fornisce il file `/bin/wget`.

Sommario

In questa lezione abbiamo imparato:

- Come usare `rpm` per installare, aggiornare e rimuovere i pacchetti.
- Come usare `yum`, `zypper` e `dnf`.
- Come ottenere informazioni su un pacchetto.
- Come ottenere un elenco dei contenuti di un pacchetto.
- Come scoprire da quale pacchetto proviene un file.
- Come elencare, aggiungere, rimuovere, abilitare o disabilitare i repository di software.

Sono stati discussi i seguenti file, termini e utilità:

- `rpm`
- `yum`
- `dnf`
- `zypper`

Risposte agli Esercizi Guidati

1. Usando `rpm` su un sistema Red Hat Enterprise Linux, come installereste il pacchetto `file-roller-3.28.1-2.el7.x86_64.rpm` tanto da fargli mostrare una barra di avanzamento durante l'installazione?

Utilizzare l'opzione `-i` per installare un pacchetto e l'opzione `-h` per abilitare gli "hash mark" per mostrare l'avanzamento dell'installazione. Quindi, la risposta è: `rpm -ih file-roller-3.28.1-2.el7.x86_64.rpm`.

2. Usando `rpm`, scopri quale pacchetto contiene il file `/etc/redhat-release`.

Si stanno richiedendo informazioni su un file, quindi usare l'opzione `-qf`: `rpm -qf /etc/redhat-release`.

3. Come useresti `yum` per verificare la presenza di aggiornamenti per tutti i pacchetti nel sistema?

Utilizzare l'operazione `check-update` senza un nome pacchetto: `yum check-update`.

4. Usando `zypper`, come disabilitaresti un repository chiamato `repo-extras`?

Usare il parametro `editrepo` per cambiare i parametri di un repo, e l'opzione `-d` per disabilitarlo: `zypper editrepo -d repo-extra`.

5. Se hai un file `.repo` che descrive un nuovo repository, dove dovrebbe essere messo questo file in modo che venga riconosciuto da DNF?

I file `.repo` per DNF dovrebbero essere messi nella stessa locazione usata da YUM, quindi all'interno di `/etc/yum.repos.d/`.

Risposte agli Esercizi Esplorativi

1. Come useresti `zypper` per scoprire quale pacchetto detiene il file `/usr/sbin/swapon`?

Usare il parametro `se` (search) e l'opzione `--provides: zypper se --provides /usr/sbin/swapon`.

2. Come puoi ottenere un elenco di tutti i pacchetti installati nel sistema usando `dnf`?

Utilizzare il parametro `list`, seguito dall'opzione `--installed: dnf list --installed`.

3. Usando `dnf`, qual è il comando per aggiungere un repository situato in <https://www.example.url/home:reponame.repo> al sistema?

Apportare modifiche ai repository è un “configuration change”, quindi usare `config-manager` e l'opzione `--add_repo: dnf config-manager --add_repo https://www.example.url/home:reponame.repo`.

4. Come puoi usare `zypper` per verificare se il pacchetto `unzip` è installato?

IS deve effettuare una ricerca (`se`) sui pacchetti installati (`-i`): `zypper se -i unzip`.

5. Usando `yum`, scopri quale pacchetto fornisce il file `/bin/wget`.

Per scoprire cosa fornisce un file, usare `whatprovides` e il nome del file: `yum whatprovides /bin/wget`.



102.6 Linux come sistema virtualizzato

Obiettivi LPI di riferimento

LPIC-1, Exam 101, Objective 102.6

Peso

1

Aree di Conoscenza Chiave

- Comprendere i concetti generali di macchine virtuali e container.
- Comprendere gli elementi comuni delle macchine virtuali in una cloud IaaS, come istanze di elaborazione, archiviazione a blocchi e rete.
- Comprendere le proprietà uniche di un sistema Linux che devono essere modificate quando un sistema viene clonato o utilizzato come template.
- Comprendere come vengono utilizzate le immagini di sistema per distribuire macchine virtuali, istanze cloud e container.
- Comprendere le estensioni Linux che integrano Linux con un prodotto di virtualizzazione.
- Conoscenza di cloud-init.

Di seguito è riportato un elenco parziale dei file, dei termini e dei comandi utilizzati

- Virtual machine
- Container Linux
- Container applicativo
- Guest driver
- Chiavi SSH di sistema
- D-Bus machine id



102.6 Lezione 1

Certificazione:	LPIC-1
Versione:	5.0
Argomento:	102 L'Installazione di Linux e la Gestione Pacchetti
Obiettivo:	102.6 Linux come guest virtualizzato
Lezione:	1 di 1

Introduzione

Uno dei grandi punti di forza di Linux è la sua versatilità. Un aspetto di questa versatilità è la capacità di utilizzare Linux come mezzo per ospitare altri sistemi operativi, o singole applicazioni, in un ambiente completamente isolato e sicuro. Questa lezione si concentrerà sui concetti di virtualizzazione e tecnologie container, insieme ad alcuni dettagli tecnici che dovrebbero essere presi in considerazione quando si distribuisce una macchina virtuale su una piattaforma cloud.

Concetti di Virtualizzazione

La virtualizzazione è una tecnologia che consente a una piattaforma software, denominata *hypervisor*, di eseguire processi che contengono un sistema informatico completamente emulato. L'hypervisor è responsabile della gestione delle risorse dell'hardware fisico che possono essere utilizzate dalle singole macchine virtuali. Queste macchine virtuali sono chiamate *guest* dell'hypervisor. Una macchina virtuale ha molti aspetti di un computer fisico emulato nel software, come il BIOS di un sistema e i controller del disco rigido. Una macchina virtuale utilizzerà spesso immagini del disco rigido archiviate come singoli file e avrà accesso alla RAM e alla CPU della macchina host tramite l'hypervisor. Quest'ultimo separa l'accesso alle risorse

hardware del sistema host tra i vari guest, consentendo l'esecuzione di più sistemi operativi su un singolo sistema host.

Gli hypervisor comunemente usati per Linux sono:

Xen

Xen è un hypervisor di tipo 1 open source, il che significa che non si basa su un sistema operativo sottostante per funzionare. Un hypervisor di questo tipo è noto come hypervisor *bare-metal* poiché il computer può avviarsi direttamente nell'hypervisor.

KVM

Kernel Virtual Machine è un modulo del kernel Linux per la virtualizzazione. KVM è un hypervisor sia di tipo 1 che di tipo 2 perché, sebbene abbia bisogno di un sistema operativo Linux generico per funzionare, è in grado di funzionare come hypervisor integrandosi perfettamente con un'installazione Linux in esecuzione. Le macchine virtuali erogate tramite KVM usano il demone `libvirt` e le utility software associate per essere create e gestite.

VirtualBox

Una popolare applicazione desktop che semplifica la creazione e la gestione di macchine virtuali. Oracle VM VirtualBox è un'ambiente multiplatforma e funziona su Linux, macOS e Microsoft Windows. Poiché VirtualBox richiede l'esecuzione di un sistema operativo sottostante, lo possiamo considerare come un hypervisor di tipo 2.

Alcuni hypervisor consentono il trasferimento dinamico di una macchina virtuale. Il processo di spostamento di una macchina virtuale da un'installazione hypervisor a un'altra si chiama *migration* e le tecniche coinvolte possono differire tra un hypervisor e l'altro. Alcune migrazioni possono essere eseguite solo quando il sistema guest è stato completamente spento, mentre altre possono essere eseguite mentre il guest è in esecuzione (*live migration*). Tali tecniche possono essere di aiuto durante le operazioni di manutenzione degli hypervisor, o per la resilienza di sistema quando un hypervisor smette di essere operativo e la VM Guest deve essere spostata su un hypervisor funzionante.

Tipi di Macchine Virtuali

Esistono tre tipi principali di macchine virtuali, le *completamente virtualizzate*, le *paravirtualizzate* e le *ibride*.

Completamente virtualizzate

Tutte le istruzioni che un sistema operativo guest esegue devono essere in grado di farlo all'interno del sistema operativo virtualizzato. La ragione di ciò è che nessun driver software aggiuntivo è installato all'interno del guest per tradurre le istruzioni in hardware simulato o

reale. Un guest completamente virtualizzato è quello in cui il guest (o HardwareVM) non è consapevole di essere un'istanza di macchina virtuale in esecuzione. Affinché questo tipo di virtualizzazione avvenga su hardware x86, le estensioni della CPU Intel VT-x o AMD-V devono essere abilitate sul sistema su cui è installato l'hypervisor. Questo può essere fatto da un menu di configurazione del firmware BIOS o UEFI.

Paravirtualizzate

Un guest paravirtualizzato (o PVM) è quello in cui il sistema operativo guest è consapevole che si tratta di un'istanza di macchina virtuale in esecuzione. Questi tipi di guest faranno uso di un kernel modificato e di driver speciali (noti come *guest drivers*) che aiuteranno il sistema operativo guest a utilizzare le risorse software e hardware dell'hypervisor. Le prestazioni di un guest paravirtualizzato sono spesso migliori di quelle del guest completamente virtualizzato a causa del vantaggio offerto da questi driver software.

Ibride

La paravirtualizzazione e la virtualizzazione completa possono essere combinate per consentire ai sistemi operativi non modificati di ricevere prestazioni di I/O quasi native utilizzando driver paravirtualizzati su sistemi operativi completamente virtualizzati (HVM). I driver paravirtualizzati contengono driver di archiviazione e rete con prestazioni avanzate di I/O.

Le piattaforme di virtualizzazione spesso forniscono driver specifici per i sistemi operativi guest. KVM utilizza i driver del progetto *Virtio*, mentre Oracle VM VirtualBox utilizza *Guest Extensions* disponibili da un file immagine CD-ROM ISO scaricabile.

Esempio di una Virtual Machine gestita tramite libvirt

Vedremo un esempio di macchina virtuale gestita da `libvirt` e che utilizza l'hypervisor KVM. Una macchina virtuale è spesso costituita da un gruppo di file, principalmente un file XML che definisce la macchina virtuale (come la sua configurazione hardware, connettività di rete, funzionalità di visualizzazione e altro) e un file immagine del disco rigido che contiene l'installazione del sistema operativo e il relativo software.

Innanzitutto, iniziamo a esaminare un file di configurazione XML di esempio per una macchina virtuale e il suo ambiente di rete:

```
$ ls /etc/libvirt/qemu
total 24
drwxr-xr-x 3 root root 4096 Oct 29 17:48 networks
-rw----- 1 root root 5667 Jun 29 17:17 rhel8.0.xml
```


NOTE

La parte `qemu` del percorso della directory si riferisce al software su cui si basano le macchine virtuali basate su KVM. Il progetto QEMU fornisce all'hypervisor il software per emulare i dispositivi hardware che la macchina virtuale utilizzerà, come controller del disco, accesso alla CPU dell'host, emulazione della scheda di rete e altro ancora.

Da notare inoltre che esiste una directory denominata `networks`. Questa directory contiene i file (sempre di tipo XML) che definiscono le configurazioni di rete che possono essere utilizzate dalle macchine virtuali. Questo hypervisor utilizza solo una rete e quindi esiste un solo file di definizione che contiene una configurazione per un segmento di rete virtuale che verrà utilizzato da questi sistemi.

```
$ ls -l /etc/libvirt/qemu/networks/
total 8
drwxr-xr-x 2 root root 4096 Jun 29 17:15 autostart
-rw----- 1 root root 576 Jun 28 16:39 default.xml
$ sudo cat /etc/libvirt/qemu/networks/default.xml
<!--
WARNING: THIS IS AN AUTO-GENERATED FILE. CHANGES TO IT ARE LIKELY TO BE
OVERWRITTEN AND LOST. Changes to this xml configuration should be made using:
    virsh net-edit default
or other application using the libvirt API.
-->

<network>
  <name>default</name>
  <uuid>55ab064f-62f8-49d3-8d25-8ef36a524344</uuid>
  <forward mode='nat' />
  <bridge name='virbr0' stp='on' delay='0' />
  <mac address='52:54:00:b8:e0:15' />
  <ip address='192.168.122.1' netmask='255.255.255.0'>
    <dhcp>
      <range start='192.168.122.2' end='192.168.122.254' />
    </dhcp>
  </ip>
</network>
```

Questa definizione include una rete privata di classe C e un dispositivo hardware emulato per fungere da router per questa rete. Esiste anche un intervallo di indirizzi IP che l'hypervisor, svolgendo il ruolo di server DHCP, può assegnare alle macchine virtuali che utilizzano questa rete. Questa configurazione di rete utilizza anche la *network address translation* (NAT) per inoltrare i pacchetti ad altre reti, come la LAN dell'hypervisor.

Rivolgiamo ora la nostra attenzione a un file di definizione di una VM Red Hat Enterprise Linux 8. (le sezioni speciali sono in grassetto):

```
$ sudo cat /etc/libvirt/qemu/rhel8.0.xml
<!--
WARNING: THIS IS AN AUTO-GENERATED FILE. CHANGES TO IT ARE LIKELY TO BE
OVERWRITTEN AND LOST. Changes to this xml configuration should be made using:
    virsh edit rhel8.0
or other application using the libvirt API.
-->

<domain type='kvm'>
  <name>rhel8.0</name>
  <uuid>fadd8c5d-c5e1-410e-b425-30da7598d0f6</uuid>
  <metadata>
    <libosinfo:libosinfo xmlns:libosinfo="http://libosinfo.org/xmlns/libvirt/domain/1.0">
      <libosinfo:os id="http://redhat.com/rhel/8.0"/>
    </libosinfo:libosinfo>
  </metadata>
  <memory unit='KiB'>4194304</memory>
  <currentMemory unit='KiB'>4194304</currentMemory>
  <vcpu placement='static'>2</vcpu>
  <os>
    <type arch='x86_64' machine='pc-q35-3.1'>hvm</type>
    <boot dev='hd' />
  </os>
  <features>
    <acpi/>
    <apic/>
    <vmport state='off' />
  </features>
  <cpu mode='host-model' check='partial'>
    <model fallback='allow' />
  </cpu>
  <clock offset='utc'>
    <timer name='rtc' tickpolicy='catchup' />
    <timer name='pit' tickpolicy='delay' />
    <timer name='hpet' present='no' />
  </clock>
  <on_poweroff>destroy</on_poweroff>
  <on_reboot>restart</on_reboot>
  <on_crash>destroy</on_crash>
  <pm>
    <suspend-to-mem enabled='no' />
  </pm>
</domain>
```

```

    <suspend-to-disk enabled='no' />
</pm>
<devices>
  <emulator>/usr/bin/qemu-system-x86_64</emulator>
  <disk type='file' device='disk'>
    <driver name='qemu' type='qcow2' />
    <source file='/var/lib/libvirt/images/rhel8' />
    <target dev='vda' bus='virtio' />
    <address type='pci' domain='0x0000' bus='0x04' slot='0x00' function='0x0' />
  </disk>
  <controller type='usb' index='0' model='qemu-xhci' ports='15'>
    <address type='pci' domain='0x0000' bus='0x02' slot='0x00' function='0x0' />
  </controller>
  <controller type='sata' index='0'>
    <address type='pci' domain='0x0000' bus='0x00' slot='0x1f' function='0x2' />
  </controller>
  <controller type='pci' index='0' model='pcie-root' />
  <controller type='pci' index='1' model='pcie-root-port'>
    <model name='pcie-root-port' />
    <target chassis='1' port='0x10' />
    <address type='pci' domain='0x0000' bus='0x00' slot='0x02' function='0x0'
multifunction='on' />
  </controller>
  <controller type='pci' index='2' model='pcie-root-port'>
    <model name='pcie-root-port' />
    <target chassis='2' port='0x11' />
    <address type='pci' domain='0x0000' bus='0x00' slot='0x02' function='0x1' />
  </controller>
  <controller type='pci' index='3' model='pcie-root-port'>
    <model name='pcie-root-port' />
    <target chassis='3' port='0x12' />
    <address type='pci' domain='0x0000' bus='0x00' slot='0x02' function='0x2' />
  </controller>
  <controller type='pci' index='4' model='pcie-root-port'>
    <model name='pcie-root-port' />
    <target chassis='4' port='0x13' />
    <address type='pci' domain='0x0000' bus='0x00' slot='0x02' function='0x3' />
  </controller>
  <controller type='pci' index='5' model='pcie-root-port'>
    <model name='pcie-root-port' />
    <target chassis='5' port='0x14' />
    <address type='pci' domain='0x0000' bus='0x00' slot='0x02' function='0x4' />
  </controller>
  <controller type='pci' index='6' model='pcie-root-port'>

```

```

    <model name='pcie-root-port' />
    <target chassis='6' port='0x15' />
    <address type='pci' domain='0x0000' bus='0x00' slot='0x02' function='0x5' />
</controller>
<controller type='pci' index='7' model='pcie-root-port'>
    <model name='pcie-root-port' />
    <target chassis='7' port='0x16' />
    <address type='pci' domain='0x0000' bus='0x00' slot='0x02' function='0x6' />
</controller>
<controller type='virtio-serial' index='0'>
    <address type='pci' domain='0x0000' bus='0x03' slot='0x00' function='0x0' />
</controller>
<interface type='network'>
    <mac address='52:54:00:50:a7:18' />
    <source network='default' />
    <model type='virtio' />
    <address type='pci' domain='0x0000' bus='0x01' slot='0x00' function='0x0' />
</interface>
<serial type='pty'>
    <target type='isa-serial' port='0'>
        <model name='isa-serial' />
    </target>
</serial>
<console type='pty'>
    <target type='serial' port='0' />
</console>
<channel type='unix'>
    <target type='virtio' name='org.qemu.guest_agent.0' />
    <address type='virtio-serial' controller='0' bus='0' port='1' />
</channel>
<channel type='spicevmc'>
    <target type='virtio' name='com.redhat.spice.0' />
    <address type='virtio-serial' controller='0' bus='0' port='2' />
</channel>
<input type='tablet' bus='usb'>
    <address type='usb' bus='0' port='1' />
</input>
<input type='mouse' bus='ps2' />
<input type='keyboard' bus='ps2' />
<graphics type='spice' autoport='yes'>
    <listen type='address' />
    <image compression='off' />
</graphics>
<sound model='ich9'>

```

```

    <address type='pci' domain='0x0000' bus='0x00' slot='0x1b' function='0x0' />
</sound>
<video>
    <model type='virtio' heads='1' primary='yes' />
    <address type='pci' domain='0x0000' bus='0x00' slot='0x01' function='0x0' />
</video>
<redirdev bus='usb' type='spicevmc'>
    <address type='usb' bus='0' port='2' />
</redirdev>
<redirdev bus='usb' type='spicevmc'>
    <address type='usb' bus='0' port='3' />
</redirdev>
<memballoon model='virtio'>
    <address type='pci' domain='0x0000' bus='0x05' slot='0x00' function='0x0' />
</memballoon>
<rng model='virtio'>
    <backend model='random'>/dev/urandom</backend>
    <address type='pci' domain='0x0000' bus='0x06' slot='0x00' function='0x0' />
</rng>
</devices>
</domain>

```

Questo file definisce una serie di impostazioni hardware utilizzate da questo guest, come la quantità di RAM che gli sarà assegnata, il numero di *core* della CPU dell'hypervisor a cui il guest avrà accesso, il file di immagine del disco rigido che è associato a questo guest (sotto la dicitura `disk`), le sue capacità di visualizzazione (tramite il protocollo SPICE) e l'accesso a dispositivi USB, nonché gli input di tastiera e mouse emulati.

Esempio di Archiviazione su Disco della Macchina Virtuale

L'immagine del disco rigido di questa macchina virtuale risiede in `/var/lib/libvirt/images/rhel8`. Ecco di seguito l'immagine del disco su questo hypervisor:

```

$ sudo ls -lh /var/lib/libvirt/images/rhel8
-rw----- 1 root root 5.5G Oct 25 15:57 /var/lib/libvirt/images/rhel8

```

La dimensione corrente di questa immagine disco occupa solo 5,5 GB di spazio sull'hypervisor. Tuttavia, il sistema operativo all'interno di questo guest vede un disco di dimensioni 23,3 GB, come evidenziato dall'output del comando seguente dall'interno della macchina virtuale in esecuzione:

```
$ lsblk
NAME          MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
vda           252:0    0 23.3G  0 disk
├─vda1        252:1    0   1G  0 part /boot
└─vda2        252:2    0 22.3G  0 part
   ├─rhel-root 253:0    0  20G  0 lvm  /
   └─rhel-swap 253:1    0  2.3G  0 lvm  [SWAP]
```

Ciò è dovuto al tipo di *provisioning* del disco utilizzato per questo guest. Esistono più tipi di immagini disco che una macchina virtuale può utilizzare. I due tipi principali sono:

COW

Copy-on-write (noto anche come *thin-provisioning* o *sparse images*) è un metodo in cui viene creato un file su disco con un limite di dimensione superiore predefinito. La dimensione dell'immagine del disco aumenta solo quando i nuovi dati vengono scritti sul disco. Proprio come nell'esempio precedente, il sistema operativo guest vede il limite predefinito del disco di 23,3 GB, ma ha scritto solo 5,5 GB di dati nel file del disco reale. Il formato dell'immagine del disco usato per la macchina virtuale di esempio è `qcow2` che è un formato di file COW QEMU.

RAW

Un tipo di disco *raw* o *full* è un file che ha tutto il suo spazio pre-allocato. Per esempio, un file di immagine del disco *raw* da 10 GB occupa 10 GB di spazio sul disco effettivo sull'hypervisor. Vi è un vantaggio in termini di prestazioni in questo tipo di disco in quanto esiste già tutto lo spazio su disco necessario, quindi l'hypervisor sottostante può semplicemente scrivere i dati sul disco senza il calo di prestazioni dovuto al monitoraggio dell'immagine del disco e l'estensione della dimensione del file man mano che vengono scritti nuovi dati.

Esistono altre piattaforme di gestione della virtualizzazione come *Red Hat Enterprise Virtualization* e *oVirt* che possono utilizzare dischi fisici per fungere da storage di backend per il sistema operativo di una macchina virtuale. Questi sistemi possono utilizzare dispositivi SAN (Storage Area Network) o NAS (Network Attached Storage) per scrivere i loro dati e l'hypervisor tiene traccia di quali posizioni di archiviazione appartengono a quali macchine virtuali. Questi sistemi di archiviazione possono utilizzare tecnologie come la gestione del volume logico (LVM) per aumentare o ridurre le dimensioni dell'archiviazione su disco di una macchina virtuale in base alle esigenze e per facilitare la creazione e la gestione di snapshot.

Lavorare con i Template di Macchine Virtuali

Poiché le macchine virtuali sono in genere solo file in esecuzione su un hypervisor, è facile creare modelli (*templates*) che possono essere personalizzati per particolari scenari di distribuzione.

Spesso una macchina virtuale avrà un'installazione di base del sistema operativo e alcune impostazioni di configurazione dell'autenticazione preconfigurate configurate per facilitare le esecuzioni future del sistema. Ciò riduce la quantità di tempo necessaria per installare un nuovo sistema riducendo la quantità di lavoro che viene spesso ripetuto, come l'installazione dei pacchetti di base e le impostazioni locali.

Questo modello di macchina virtuale potrebbe successivamente essere copiato per un nuovo sistema guest. In questo caso, il nuovo guest avrà un nuovo nome, un nuovo indirizzo MAC per la sua interfaccia di rete e altre modifiche possono essere fatte a seconda dell'uso previsto.

Il D-Bus Machine ID

Molte installazioni Linux utilizzano un numero di identificazione di macchina generatosi al momento dell'installazione, chiamato *D-Bus machine ID*. Tuttavia, se una macchina virtuale viene *clonata* per essere utilizzata come modello per altre installazioni di macchine virtuali, è necessario creare un nuovo D-Bus machine ID per garantire che le risorse di sistema dall'hypervisor vengano indirizzate al sistema guest appropriato..

Il seguente comando può essere utilizzato per convalidare l'esistenza di un D-Bus machine ID per il sistema in esecuzione:

```
$ dbus-uuidgen --ensure
```

Se non vengono visualizzati messaggi di errore, esiste un ID per il sistema. Per visualizzare il D-Bus machine ID corrente, eseguire quanto segue:

```
$ dbus-uuidgen --get  
17f2e0698e844e31b12ccd3f9aa4d94a
```

La stringa di testo visualizzata è il numero ID corrente. Nessun sistema Linux in esecuzione su un hypervisor dovrebbe avere lo stesso D-Bus machine ID.

Il D-Bus machine ID si trova in `/var/lib/dbus/machine-id` ed è simbolicamente collegato a `/etc/machine-id`. La modifica di questo ID su un sistema in esecuzione è sconsigliata poiché si potrebbe verificare dell'instabilità di sistema e arresti anomali. Se due macchine virtuali hanno lo stesso D-Bus machine ID, seguire la procedura seguente per generarne uno nuovo:

```
$ sudo rm -f /etc/machine-id  
$ sudo dbus-uuidgen --ensure=/etc/machine-id
```

Nel caso in cui `/var/lib/dbus/machine-id` non sia un collegamento simbolico a `/etc/machine-id`, sarà necessario rimuovere il file `/var/lib/dbus/machine-id`.

Distribuire Macchine Virtuali in Cloud

Sono disponibili numerosi provider IaaS (*infrastructure as a service*) che eseguono sistemi hypervisor e possono distribuire immagini guest virtuali per un'organizzazione. Praticamente tutti questi provider dispongono di strumenti che consentono a un amministratore di creare, distribuire e configurare macchine virtuali personalizzate basate su una varietà di distribuzioni Linux. Molte di queste aziende dispongono inoltre di sistemi che consentono l'implementazione e le migrazioni di macchine virtuali costruite all'interno delle infrastrutture dei clienti.

Quando si valuta il rilascio di un sistema Linux in un ambiente IaaS, ci sono alcuni elementi chiave che un amministratore dovrebbe conoscere:

Istanze di calcolo

Molti fornitori di servizi cloud addebiteranno i tassi di utilizzo in base a "istanze di elaborazione" o alla quantità di tempo della CPU che verrà utilizzata dall'infrastruttura in Cloud. Un'attenta pianificazione di quanto tempo di elaborazione richiederanno effettivamente le applicazioni aiuterà a mantenere gestibili i costi di una soluzione Cloud.

Le istanze di calcolo spesso faranno riferimento al numero di macchine virtuali fornite in un ambiente Cloud. Anche in questo caso, il numero maggiore di istanze di sistemi in esecuzione contemporaneamente determinerà anche il tempo complessivo delle CPU che verrà addebitato a un'organizzazione.

Archiviazione

I fornitori Cloud hanno anche vari livelli di storage a blocchi da utilizzare per un'organizzazione. Alcune offerte sono semplicemente intese come archiviazione di rete basata su soluzioni *Web-based* per i file e altre offerte riguardano l'archiviazione esterna per una macchina virtuale da utilizzare per l'hosting dei file.

Il costo per tali offerte varierà in base alla quantità di memoria utilizzata e alla velocità della memoria all'interno dei data center del provider. L'accesso più veloce all'archiviazione in genere avrà un costo maggiore e, al contrario, i dati "a riposo" (come nell'archiviazione) sono spesso molto economici.

Rete

Uno dei componenti principali del lavorare con un fornitore di soluzioni Cloud è la configurazione relativa alla rete virtuale. Molti provider IaaS disporranno di una qualche forma di utility basata su Web che può essere utilizzata per la progettazione e

l'implementazione di differenti rotte di rete, sottoreti e configurazioni firewall. Alcuni forniranno persino soluzioni DNS in modo tale che nomi FQDN (*fully qualified domain names*) possano essere assegnati ai sistemi esposti su Internet. Esistono anche soluzioni “ibride” in grado di connettere un'infrastruttura di rete locale (*on-premise*) a un'infrastruttura basata su Cloud tramite una VPN (*virtual private network*), collegando così le due infrastrutture.

Accesso Sicuro ai Guest nel Cloud

Il metodo più diffuso in uso per accedere a un guest virtuale remoto su una piattaforma cloud è attraverso l'uso del software OpenSSH. Un sistema Linux che risiede nel cloud avrà il server OpenSSH in esecuzione, mentre l'amministratore usà un client OpenSSH con chiavi precondivise per l'accesso remoto.

Un amministratore eseguirà il seguente comando:

```
$ ssh-keygen
```

e a seguire le istruzioni per creare una coppia di chiavi SSH pubbliche e private. La chiave privata rimane sul sistema locale dell'amministratore (memorizzata in `~/.ssh/`) e la chiave pubblica viene copiata sul sistema cloud remoto, esattamente lo stesso metodo che si dovrebbe utilizzare quando si lavora con macchine in rete su una LAN aziendale.

L'amministratore eseguirà quindi il seguente comando:

```
$ ssh-copy-id -i <public_key> user@cloud_server
```

Questo copierà la chiave SSH pubblica, dalla coppia di chiavi appena generata, sul server cloud remoto. La chiave pubblica verrà registrata nel file `~/.ssh/authorized_keys` del server cloud e imposterà le autorizzazioni appropriate sul file.

NOTE

Se c'è un solo file di chiave pubblica nella directory `~/.ssh/`, allora l'opzione `-i` può essere omessa, poiché il comando `ssh-copy-id` verrà impostato come predefinito sul file della chiave pubblica nella directory (in genere il file che termina con l'estensione `.pub`).

Alcuni provider cloud genereranno automaticamente una coppia di chiavi quando viene effettuato il provisioning di un nuovo sistema Linux. L'amministratore dovrà quindi scaricare la chiave privata per il nuovo sistema dal provider cloud e memorizzarla sul proprio sistema locale. Si noti che le autorizzazioni per le chiavi SSH devono essere `0600` per una chiave privata e `0644` per una chiave pubblica.

Preconfigurazione dei Sistemi Cloud

Uno strumento utile che semplifica l'implementazione di una macchina virtuale in Cloud è l'utilità `cloud-init`. Questo comando, insieme ai file di configurazione associati e all'immagine della macchina virtuale predefinita, è un metodo indipendente dal fornitore per distribuire un guest Linux su un gran numero di provider IaaS. Utilizzando semplici file di testo YAML (*YAML Ain't Markup Language*) un amministratore può preconfigurare le impostazioni di rete, le selezioni dei pacchetti software, la configurazione della chiave SSH, la creazione di account utente, le impostazioni locali, insieme a una miriade di altre opzioni per creare rapidamente nuove sistemi.

Durante l'avvio iniziale di un nuovo sistema, `cloud-init` leggerà le impostazioni dai file di configurazione YAML e le applicherà. Questo processo deve essere applicato solo alla configurazione iniziale di un sistema e semplifica l'implementazione di flotte di nuovi sistemi sulla piattaforma di un provider cloud.

La sintassi del file YAML usata con `cloud-init` si chiama *cloud-config*. Ecco un esempio di file `cloud-config`:

```
#cloud-config
timezone: Africa/Dar_es_Salaam
hostname: test-system

# Update the system when it first boots up
apt_update: true
apt_upgrade: true

# Install the Nginx web server
packages:
  - nginx
```

Da notare che nella riga superiore non c'è spazio tra il simbolo hash (#) e il termine `cloud-config`.

NOTE

`cloud-init` non è solo per macchine virtuali. La suite di strumenti `cloud-init` può anche essere utilizzata per preconfigurare containers (come quelli LXD di Linux) prima della loro distribuzione.

I Container

La tecnologia dei container è simile in alcuni aspetti a una macchina virtuale, in cui si ottiene un ambiente isolato per distribuire facilmente un'applicazione. Mentre con una macchina virtuale

viene emulato un intero computer, un container utilizza il software sufficiente per eseguire un'applicazione. In questo modo, le risorse generali necessarie alla sua esecuzione sono molto basse.

I container consentono una maggiore flessibilità rispetto a quella di una macchina virtuale. Un container di applicazioni può essere migrato da un host a un altro, proprio come una macchina virtuale può essere migrata da un hypervisor a un altro. Tuttavia, a volte una macchina virtuale dovrà essere spenta prima che possa essere migrata, mentre con un container, l'applicazione è sempre in esecuzione mentre viene migrata. I container semplificano inoltre la distribuzione di nuove versioni di applicazioni in tandem con una versione esistente. Man mano che gli utenti chiudono le sessioni con container in esecuzione, questi container possono essere rimossi automaticamente dal sistema dal software di orchestrazione dei container e sostituiti con la nuova versione, riducendo così i tempi di inattività.

NOTE

Esistono numerose tecnologie container disponibili per Linux, come *Docker*, *Kubernetes*, *LXD* / *LXC*, *systemd-nspawn*, *OpenShift* e altre. L'esatta implementazione di un pacchetto software container va oltre lo scopo dell'esame LPIC-1.

I container fanno uso del meccanismo *control groups* (meglio noto come *cgroups*) all'interno del kernel Linux. Il cgroup è un modo per partizionare risorse di sistema come memoria, tempo del processore e larghezza di banda del disco e della rete per una singola applicazione. Un amministratore può utilizzare direttamente i cgroup per impostare i limiti delle risorse di sistema su un'applicazione o un gruppo di applicazioni che potrebbero esistere all'interno di un singolo cgroup. In sostanza, questo è ciò che fa il software container per l'amministratore, oltre a fornire strumenti che facilitano la gestione e la distribuzione di cgroups.

NOTE

Attualmente, la conoscenza dei cgroups non è necessaria per superare l'esame LPIC-1. Il concetto di cgroup è menzionato qui in modo che il Candidato abbia almeno una conoscenza di base di come un'applicazione è separata per un miglior utilizzo delle risorse di sistema.

Esercizi Guidati

1. Quali estensioni della CPU sono necessarie su una piattaforma hardware basata su x86 che eseguirà guest completamente virtualizzati?

2. Un'installazione server *mission-critical* che richiederà le prestazioni più veloci probabilmente utilizzerà quale tipo di virtualizzazione?

3. Due macchine virtuali clonate dallo stesso template e che utilizzano D-Bus funzionano in modo irregolare. Entrambi hanno nomi host e impostazioni di configurazione di rete separati. Quale comando verrebbe utilizzato per determinare se ciascuna delle macchine virtuali ha D-Bus Machine ID diversi?

Esercizi Esplorativi

1. Esegui il comando seguente per vedere se il tuo sistema ha già le estensioni della CPU abilitate per eseguire una macchina virtuale (i risultati possono variare a seconda della CPU):

```
grep --color -E "vmx|svm" /proc/cpuinfo
```

A seconda dell'output, è possibile che sia evidenziato `vmx` (per CPU con abilitate le estensioni Intel VT-x) o `svm` (per CPU con abilitate le estensioni AMD SVM). Se non si ottengono risultati, consultare le istruzioni del firmware BIOS o UEFI su come abilitare la virtualizzazione per il proprio processore.

2. Se il tuo processore supporta la virtualizzazione, cerca la documentazione della tua distribuzione per eseguire un hypervisor KVM.

- Installa i pacchetti necessari per eseguire un hypervisor KVM.

- Se si utilizza un ambiente desktop grafico, si consiglia di installare anche l'applicazione `virt-manager` che è un front-end grafico che può essere usato su un'installazione KVM. Ciò aiuterà nell'installazione e nella gestione di macchine virtuali.

- Scarica un'immagine ISO della distribuzione Linux di tua scelta e, seguendo la documentazione della tua distribuzione, crea una nuova macchina virtuale usando questa ISO.

Sommario

In questa lezione sono state illustrate le basi concettuali riguardo le macchine virtuali e container e come queste tecnologie possono essere utilizzate con Linux.

Abbiamo brevemente descritto i seguenti comandi:

dbus-uuidgen

Usato per verificare e visualizzare il DBus ID di un sistema.

ssh-keygen

Usato per generare una coppia di chiavi SSH pubbliche e private da usare quando si accede a sistemi remoti basati su cloud.

ssh-copy-id

Usato per copiare la chiave SSH pubblica di un sistema su un sistema remoto per facilitare l'autenticazione remota.

cloud-init

Usato per aiutare nella configurazione e distribuzione di macchine virtuali e container in un ambiente cloud..

Risposte agli Esercizi Guidati

1. Quali estensioni della CPU sono necessarie su una piattaforma hardware basata su x86 che eseguirà guest completamente virtualizzati?

VT-x per CPU Intel o AMD-V per CPU AMD.

2. Un'installazione server mission-critical che richiederà le prestazioni più veloci probabilmente utilizzerà quale tipo di virtualizzazione?

Un sistema operativo che utilizza la paravirtualizzazione come Xen, in quanto, il sistema operativo guest può fare un uso migliore delle risorse hardware disponibili anche attraverso l'uso di driver software progettati per funzionare con l'hypervisor.

3. Due macchine virtuali clonate dallo stesso template e che utilizzano D-Bus funzionano in modo irregolare. Entrambi hanno nomi host e impostazioni di configurazione di rete separati. Quale comando verrebbe utilizzato per determinare se ciascuna delle macchine virtuali ha D-Bus Machine ID diversi?

```
dbus-uuidgen --get
```

Risposte agli Esercizi Esplorativi

1. Esegui il comando seguente per vedere se il tuo sistema ha già le estensioni della CPU abilitate per eseguire una macchina virtuale (i risultati possono variare a seconda della CPU): `grep --color -E "vmx|svm" /proc/cpuinfo`. A seconda dell'output, è possibile che sia evidenziato `vmx` (per CPU con abilitate le estensioni Intel VT-x) o `svm` (per CPU con abilitate le estensioni AMD SVM). Se non si ottengono risultati, consultare le istruzioni del firmware BIOS o UEFI su come abilitare la virtualizzazione per il proprio processore.

I risultati varieranno a seconda della CPU che hai. Ecco un esempio di output da un computer con una CPU Intel con estensioni di virtualizzazione abilitate nel firmware UEFI:

```
$ grep --color -E "vmx|svm" /proc/cpuinfo
flags          : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36
clflush dts acpi mmx fxsr sse sse2 ss ht tm pbe syscall nx pdpe1gb rdtscp lm constant_tsc
art arch_perfmon pebs bts rep_good nopl xtopology nonstop_tsc cpuid aperfmperf pni
pclmulqdq dtes64 monitor ds_cpl vmx smx est tm2 ssse3 sdbg fma cx16 xtpr pdcm pcid sse4_1
sse4_2 x2apic movbe popcnt tsc_deadline_timer aes xsave avx f16c rdrand lahf_lm abm
3dnowprefetch cpuid_fault epb invpcid_single pti ssbd ibrs ibpb stibp tpr_shadow vnmi
flexpriority ept vpid ept_ad fsgsbase tsc_adjust bmi1 hle avx2 smep bmi2 erms invpcid rtm
mpx rdseed adx smap clflushopt intel_pt xsaveopt xsavec xgetbv1 xsaves dtherm ida arat
pln pts hwp hwp_notify hwp_act_window hwp_epp md_clear flush_l1d
```

2. Se il tuo processore supporta la virtualizzazione, cerca la documentazione della tua distribuzione per eseguire un hypervisor KVM.
 - Installa i pacchetti necessari per eseguire un hypervisor KVM.

Questo varierà a seconda della tua distribuzione, ma qui ci sono alcuni punti di partenza:

Ubuntu — <https://help.ubuntu.com/lts/serverguide/libvirt.html>

Fedora — <https://docs.fedoraproject.org/en-US/quick-docs/getting-started-with-virtualization/>

Arch Linux — <https://wiki.archlinux.org/index.php/KVM>

- Se si utilizza un ambiente desktop grafico, si consiglia di installare anche l'applicazione `virt-manager` che è un front-end grafico che può essere usato su un'installazione KVM. Ciò aiuterà nelle installazioni e nella gestione di macchine virtuali.

Ancora una volta, questo varierà in base alla distribuzione. Un esempio con Ubuntu è il seguente:


```
$ sudo apt install virt-manager
```

- Scarica un'immagine ISO della distribuzione Linux di tua scelta e, seguendo la documentazione della tua distribuzione, crea una nuova macchina virtuale usando questa ISO.

Questo compito è facilmente gestibile dal pacchetto `virt-manager`. Comunque una macchina virtuale può anche essere creata dalla CLI usando il comando `virt-install`. Prova entrambi i metodi per comprendere come vengono create le macchine virtuali..



**Linux
Professional
Institute**

Argomento 103: GNU e Unix Commands



Linux
Professional
Institute

103.1 Lavorare con la Command Line

Obiettivi LPI di riferimento

LPIC-1 version 5.0, Exam 101, Objective 103.1

Peso

4

Aree di Conoscenza Chiave

- Utilizzare comandi di shell singoli e sequenze di comandi per eseguire attività di base nella Command Line.
- Usare e modificare l'ambiente della shell inclusa la definizione, il riferimento e l'esportazione delle variabili d'ambiente.
- Usare e modificare la cronologia dei comandi.
- Richiamare comandi all'interno e all'esterno del percorso definito.

Di seguito è riportato un elenco parziale dei file, dei termini e dei comandi utilizzati

- `bash`
- `echo`
- `env`
- `export`
- `pwd`
- `set`
- `unset`
- `type`
- `which`

- `man`
- `uname`
- `history`
- `.bash_history`
- Quoting



Linux
Professional
Institute

103.1 Lezione 1

Certificazione:	LPIC-1
Versione:	5.0
Argomento:	103 Comandi GNU e Unix
Obiettivo:	103.1 Lavorare con la Riga di Comando
Lezione:	1 di 2

Introduzione

I nuovi arrivati nel mondo dell'amministrazione Linux e della *shell Bash* spesso si sentono un po' persi senza i confort rassicuranti di un'interfaccia GUI. Sono abituati ad avere accesso con il tasto destro del mouse ai segnali visivi e alle informazioni contestuali rese disponibili dalle utility grafiche di gestione file. Quindi è importante imparare e padroneggiare rapidamente l'insieme relativamente piccolo di strumenti da riga di comando attraverso il quale è possibile attingere istantaneamente a tutti i dati disponibili.

Ottenere Informazioni di Sistema

Mentre fissi il rettangolo lampeggiante di un prompt della riga di comando, la tua prima domanda sarà probabilmente "Dove sono?" o, più precisamente, "`Dove mi trovo nel filesystem Linux in questo momento e se, diciamo, ho creato un nuovo file, dove risiederebbe?" Quello che stai cercando qui è la tua directory di lavoro attuale, e il comando ``pwd`` ti dirà ciò che vuoi sapere:

```
$ pwd
```

```
/home/frank
```

Supponendo che Frank sia attualmente connesso al sistema e che ora si trovi nella sua home directory: `/home/frank/`. Se Frank crea un file vuoto usando il comando `touch` senza specificare un'altra posizione nel filesystem, il file verrà creato in `/home/frank/`. Attraverso il comando `ls` ci verrà mostrato questo nuovo file:

```
$ touch newfile
$ ls
newfile
```

Oltre alla tua posizione nel filesystem, spesso vorrai informazioni sul sistema Linux che stai eseguendo. Ciò potrebbe includere il numero esatto di rilascio della distribuzione o la versione del kernel Linux attualmente caricato. Lo strumento `uname` è ciò che cerchi. E, in particolare, `uname` usando l'opzione `-a` ("all").

```
$ uname -a
Linux base 4.18.0-18-generic #19~18.04.1-Ubuntu SMP Fri Apr 5 10:22:13 UTC 2019 x86_64
x86_64 x86_64 GNU/Linux
```

`uname` mostra che la macchina di Frank ha installato la versione del kernel Linux 4.18.0 ed esegue Ubuntu 18.04 su una CPU a 64 bit (`x86_64`).

Ottenere Informazioni sui Comandi

Ti imbatteai spesso in documentazione che parla di comandi Linux con i quali non hai ancora familiarità. La stessa riga di comando offre tutti i tipi di informazioni utili su che cosa facciano i comandi e su come usarli efficacemente. Forse le informazioni più utili si trovano nei numerosi file del sistema `man`.

Di norma gli sviluppatori Linux scrivono file `man` e li distribuiscono insieme alle utility che creano. I file `man` sono documenti altamente strutturati il cui contenuto è intuitivamente diviso per titoli di sezione standard. Digitando `man` seguito dal nome di un comando si otterranno informazioni che includono il nome del comando, una breve sinossi di utilizzo, una descrizione più dettagliata e alcuni importanti precedenti storici e di licenza. Ecco un esempio:

```
$ man uname
NAME(1)                User Commands                UNAME(1)
NAME
```

uname - print system information

SYNOPSIS

uname [OPTION]...

DESCRIPTION

Print certain system information. With no OPTION, same as -s.

-a, --all

print all information, in the following order, except omit -p and -i if unknown:

-s, --kernel-name

print the kernel name

-n, --nodename

print the network node hostname

-r, --kernel-release

print the kernel release

-v, --kernel-version

print the kernel version

-m, --machine

print the machine hardware name

-p, --processor

print the processor type (non-portable)

-i, --hardware-platform

print the hardware platform (non-portable)

-o, --operating-system

print the operating system

--help display this help and exit

--version

output version information and exit

AUTHOR

Written by David MacKenzie.

REPORTING BUGS

GNU coreutils online help: <<http://www.gnu.org/software/coreutils/>>

Report uname translation bugs to

<<http://translationproject.org/team/>>

COPYRIGHT

Copyright©2017 Free Software Foundation, Inc. License GPLv3+: GNU GPL version 3 or later <<http://gnu.org/licenses/gpl.html>>.

This is free software: you are free to change and redistribute it.

There is NO WARRANTY, to the extent permitted by law.

SEE ALSO

arch(1), uname(2)

Full documentation at: <<http://www.gnu.org/software/coreutils/uname>>

or available locally via: info '(coreutils) uname invocation'

GNU coreutils 8.28

January 2018

UNAME(1)

`man` funziona solo quando gli viene fornito il nome esatto di comando. Se, tuttavia, non si è sicuri del nome del comando che si sta cercando, si può usare il comando `apropos` per cercare tra i nomi e le descrizioni delle pagine di `man`. Supponendo, per esempio, che non ci si ricordi che è `uname` che indicherà l'attuale versione del kernel Linux, si può passare la parola `kernel` ad `apropos`. Probabilmente si otterranno molte righe di output, ma tra queste ci dovrebbero essere:

```
$ apropos kernel
systemd-udevd-kernel.socket (8) - Device event managing daemon
uname (2) - get name and information about current kernel
urandom (4) - kernel random number source devices
```

Se non hai bisogno della documentazione completa di un comando, puoi ottenere rapidamente le sue informazioni di base usando `type`. Questo esempio usa `type` per interrogare quattro comandi separati contemporaneamente. I risultati ci mostrano che `cp` ("copy") è un programma che risiede in `/bin/cp` e che `kill` (cambia lo stato di un processo in esecuzione) è uno *shell builtin* - nel senso che è in realtà integrato nella shell Bash stessa:

```
$ type uname cp kill which
uname is hashed (/bin/uname)
cp is /bin/cp
kill is a shell builtin
which is /usr/bin/which
```

Si noti che, oltre a essere un normale comando binario come `cp`, `uname` è anche "hashed". Questo perché Frank ha recentemente usato `uname` e, per aumentare l'efficienza del sistema, è stato aggiunto a una tabella hash al fine di renderlo più velocemente accessibile alla successiva esecuzione. Se eseguisse `type uname` dopo un avvio del sistema, Frank troverebbe che `type` descrive ancora una volta `uname` come un normale binario.

NOTE Un modo più rapido per ripulire la tabella hash è eseguire il comando `hash -d`.

A volte, specialmente quando si lavora con script automatici, è necessaria una fonte di informazioni più semplice su un comando. Il comando `which`, che il nostro precedente comando `type` ha tracciato per noi, non restituirà altro che la posizione assoluta di un comando. Questo esempio individua i comandi `uname` e `which`.

```
$ which uname which
/bin/uname
/usr/bin/which
```


NOTE

Se volete visualizzare informazioni sui comandi “builtin”, potete usare il comando `help` seguito dal comando.

Utilizzare l'History dei Comandi

Spesso ricercherai attentamente l'utilizzo corretto di un comando e lo eseguirai insieme a una complicata serie di opzioni e argomenti. Ma cosa succede poche settimane dopo quando è necessario eseguire lo stesso comando con le stesse opzioni e gli stessi argomenti ma non è possibile ricordare i dettagli? Invece di dover ricominciare tutto da capo, sarai spesso in grado di recuperare il comando originale usando `history`.

Digitando `history` verranno restituiti i comandi che hai eseguito precedentemente, con il più recente per ultimo. Puoi facilmente cercare tra questi comandi reindirizzando una stringa specifica al comando `grep`. Questo esempio cercherà qualsiasi comando che includa il testo `bash_history`:

```
$ history | grep bash_history
1605 sudo find /home -name ".bash_history" | xargs grep sudo
```

Viene restituito un singolo comando insieme al suo numero progressivo, 1605.

Parlando di `bash_history`, questo è in realtà il nome di un file nascosto che dovresti trovare nella home directory del tuo utente. Dato che è un file nascosto (indicato come tale dal punto che precede il suo nome file), sarà visibile solo elencando il contenuto della directory usando `ls` con l'argomento `-a`:

```
$ ls /home/frank
newfile
$ ls -a /home/frank
.  ..  .bash_history  .bash_logout  .bashrc  .profile  .ssh  newfile
```

Che cosa c'è nel file `.bash_history`? Dai un'occhiata tu stesso: vedrai centinaia e centinaia dei tuoi comandi più recenti. Potresti, tuttavia, essere sorpreso di scoprire che mancano alcuni dei tuoi comandi più recenti. Questo perché, mentre vengono immediatamente aggiunti al database dinamico `history`, le ultime aggiunte alla cronologia dei comandi non vengono scritte nel file `.bash_history` fino a quando non si esce dalla sessione.

Puoi sfruttare i contenuti di "history" per rendere la tua esperienza da riga di comando molto più veloce ed efficiente usando i tasti freccia su e giù della tastiera. Premendo il tasto “Su” più volte, la riga di comando verrà popolata con comandi recenti. Quando arrivi a quello che desideri eseguire

una seconda volta, puoi eseguirlo premendo Invio. Ciò semplifica il richiamo e, se desiderato, la modifica dei comandi più volte durante una sessione di shell.

Esercizi Guidati

1. Usa il sistema `man` per determinare quale opzione consente ad `apropos` di eseguirsi, per poi immediatamente richiudersi, solo indicando un breve riepilogo del suo funzionamento.

2. Usa il sistema `man` per determinare quale licenza è utilizzata dal comando `grep`.

Esercizi Esplorativi

1. Identifica l'architettura hardware e la versione del kernel Linux utilizzata sul tuo computer in un formato di output di facile lettura.

2. Visualizza le ultime venti righe del database dinamico `history` e del file `.bash_history` per confrontarli.

3. Usa lo strumento `apropos` per identificare la pagina `man` del comando di cui avrai bisogno per visualizzare la dimensione in byte anziché in megabyte o gigabyte di un dispositivo a blocchi.

Sommario

In questa lezione abbiamo imparato:

- Come ottenere informazioni riguardo la posizione sul filesystem e sullo stack del software del sistema operativo.
- Come trovare aiuto per l'utilizzo di un comando.
- Come identificare la posizione sul filesystem e il tipo di comandi binari.
- Come trovare e riutilizzare i comandi eseguiti in precedenza.

In questa lezione sono stati discussi i seguenti comandi:

Pwd

Visualizza la posizione di lavoro corrente sul filesystem.

Uname

Visualizza l'architettura hardware del sistema, la versione del kernel Linux, la distribuzione e la versione di distribuzione.

Man

Accede ai file della guida che documentano l'utilizzo dei comandi.

Type

Visualizza la posizione sul filesystem e il tipo per uno o più comandi.

Which

Visualizza la posizione sul filesystem per un comando.

History

Visualizza i comandi eseguiti in precedenza.

Risposte agli Esercizi Guidati

1. Usa il sistema `man` per determinare quale opzione consente ad `apropos` di eseguirsi, per poi immediatamente richiudersi, solo indicando un breve riepilogo del suo funzionamento.

Esegui `man apropos` e scorri verso il basso attraverso la sezione “Options” fino ad arrivare alla parte riguardante l’opzione `--usage`.

2. Usa il sistema `man` per determinare quale licenza è utilizzata dal comando `grep`.

Esegui `man grep` e scorri verso il basso fino alla sezione “Copyright”. Nota che il programma utilizza una licenza d’uso della Free Software Foundation.

Risposte agli Esercizi Esplorativi

1. Identifica l'architettura hardware e la versione del kernel Linux in uso sul tuo computer in un formato di output di facile lettura.

Esegui `man uname`, leggi la sezione “Description” e identifica gli argomenti del comando che mostreranno solo i risultati esatti che desideri. Nota come `-v` ti fornirà la versione del kernel e `-i` fornirà la piattaforma hardware.

```
$ man uname
$ uname -v
$ uname -i
```

2. Stampa le ultime venti righe del database dinamico `history` e del file `.bash_history` per confrontarli.

```
$ history 20
$ tail -n 20 .bash_history
```

3. Usa il comando `apropos` per identificare la pagina `man` dove troverai il comando di cui avrai bisogno per visualizzare la dimensione di un dispositivo a blocchi collegato al sistema in byte invece che in megabyte o gigabyte.

Un modo è quello di eseguire `apropos` seguito dalla stringa `block`, leggere i risultati, notare che `lsblk` elenca i dispositivi a blocchi (sarebbe quindi lo strumento più probabile per le nostre esigenze), eseguire `man lsblk`, scorrere fino alla sezione “Description” e scoprire che `-b` mostrerà la dimensione del dispositivo in byte. Infine, eseguire `lsblk -b` per vedere cosa viene visualizzato.

```
$ apropos block
$ man lsblk
$ lsblk -b
```



103.1 Lezione 2

Certificazione:	LPIC-1
Versione:	5.0
Argomento:	103 Comandi GNU e Unix
Obiettivo:	103.1 Lavorare con la Riga di Comando
Lezione:	2 di 2

Introduzione

Un ambiente di sistema operativo include gli strumenti di base, come la shell della riga di comando e talvolta una GUI, necessari per eseguire ciò di cui si ha bisogno. Il tuo ambiente inoltre ti fornirà tutta una serie di scorciatoie e valori predefiniti. In questa sezione impareremo come elencare, invocare e gestire quei valori.

Trovare le variabili di ambiente

Per iniziare, come possiamo identificare i valori attuali per ciascuna delle nostre variabili d'ambiente? Un modo è attraverso il comando `env`:

```
$ env
DBUS_SESSION_BUS_ADDRESS=unix:path=/run/user/1000/bus
XDG_RUNTIME_DIR=/run/user/1000
XAUTHORITY=/run/user/1000/gdm/Xauthority
XDG_CONFIG_DIRS=/etc/xdg/xdg-ubuntu:/etc/xdg
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin
```



```
GJS_DEBUG_TOPICS=JS ERROR;JS LOG
[...]
```

Otterrai un gran numero di informazioni in output, molto più di quanto è incluso nell'estratto sopra. Ma per ora nota la voce `PATH`, che contiene le directory in cui la tua shell (e altri programmi) cercheranno altri programmi senza necessariamente doverli richiamare indicandone il percorso completo. Con questo set, puoi eseguire un programma binario che risiede in `/usr/local/bin` dalla tua home directory, come se il file fosse locale.

Cambiamo argomento per un attimo. Il comando `echo` stamperà sullo schermo qualunque cosa tu gli dica. Che tu ci creda o no, ci saranno delle volte in cui usare `echo`, per ripetere letteralmente qualcosa, sarà molto utile.

```
$ echo "Hi. How are you?"
Hi. How are you?
```

Ma c'è qualcos'altro che puoi fare con `echo`. Quando gli dai il nome di una variabile d'ambiente - e gli dici che questa è una variabile antepoendo al suo nome un carattere `$` — ciò facendo, invece di stampare semplicemente il nome della variabile, la shell restituirà il suo valore. Non sei sicuro che la tua directory preferita sia attualmente nella variabile `PATH`? Puoi verificare rapidamente attraverso il comando `echo`:

```
$ echo $PATH
/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin
```

Creare Nuove Variabili d'Ambiente

Puoi aggiungere le tue variabili personalizzate al tuo ambiente. Il modo più semplice è usare il carattere `=`. La stringa a sinistra sarà il nome della tua nuova variabile e la stringa a destra sarà il suo valore. Ora puoi fornire il nome della variabile a `echo` per verificare se ha funzionato:

```
$ myvar=hello
$ echo $myvar
hello
```

NOTE

Da notare che non c'è spazio su entrambi i lati del segno uguale durante l'assegnazione variabile.

Ma ha funzionato davvero? Digita "bash" nel terminale per aprire una nuova shell. Questa nuova shell assomiglia esattamente a quella in cui ti trovavi, ma in realtà è una *figlia* (child) di quella originale (che chiamiamo *genitore*). Ora, all'interno di questa nuova shell figlia, cerca di far sì che `echo` realizzi la sua magia come prima. Niente. Cosa sta succedendo?

```
$ bash
$ echo $myvar

$
```

Una variabile che crei come abbiamo appena fatto sarà disponibile solo localmente - entro la sessione di shell stessa. Se avvii una nuova shell - o chiudi la sessione usando `exit` - la variabile non ti seguirà. Digitando `exit` qui tornerai alla shell genitore originale che, in questo momento, è dove vogliamo essere. Puoi eseguire `echo $myvar` ancora una volta se vuoi solo confermare che la variabile è ancora valida. Ora digita `export myvar` per passare la variabile a tutte le shell figlie che potresti successivamente creare. Provalo: digita `bash` per una nuova shell e poi `echo` seguito dalla variabile:

```
$ exit
$ export myvar
$ bash
$ echo $myvar
hello
```

Tutto ciò può sembrare un po' sciocco quando stiamo creando Shell senza uno scopo reale. Ma capire come vengono propagate le variabili della shell attraverso il tuo sistema diventerà molto importante una volta che si inizierà a scrivere script.

Cancellare una Variabile d'Ambiente

Vuoi sapere come eliminare tutte quelle variabili effimere che hai creato? Un modo è semplicemente quello di chiudere la shell genitore o riavviare il computer. Ma ci sono modi più semplici. Per esempio, "unset". Digitare `unset` seguito dalla variabile (senza `$`) per eliminarla. `echo` ora lo dimostrerà.

```
$ unset myvar
$ echo $myvar

$
```

Se c'è un comando `unset`, allora puoi scommettere che ci sia anche un comando `set`. L'esecuzione di `set` da sola mostrerà molte informazioni in output, ma in realtà non è poi così diverso da quello che ti ha restituito `env`. Guarda la prima riga di output che otterrai quando filtri per la stringa `PATH`:

```
$ set | grep PATH
PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin
[...]
```

Qual è la differenza tra `set` ed `env`? Per i nostri scopi, la cosa principale è che l'esecuzione di `set` produrrà in output variabili e funzioni. Illustriamolo. Creeremo una nuova variabile chiamata `mynewvar` e poi confermeremo che è lì:

```
$ mynewvar=goodbye
$ echo $mynewvar
goodbye
```

Ora, eseguendo `env` insieme a `grep` per filtrare la stringa `mynewvar` non otterremo nessun risultato nell'output. Ma eseguire `set` allo stesso modo ci mostrerà la nostra variabile locale precedentemente creata.

```
$ env | grep mynewvar

$ set | grep mynewvar
mynewvar=goodbye
```

Usare il Virgolettato per Escludere i Caratteri Speciali

Ora è un buon momento per farti affrontare il problema dei caratteri speciali. I caratteri alfanumerici (a-z e 0-9) vengono normalmente letti letteralmente da Bash. Se provi a creare un nuovo file chiamato `myfile`, dovresti semplicemente digitare `touch myfile` e Bash saprà cosa farne. Ma se vuoi includere un carattere speciale nel tuo nome file, dovrai fare un po' più di lavoro.

Per illustrare questo, eseguiremo `touch` seguito da: `my big file`. Il problema è che ci sono due spazi tra le parole che Bash interpreterà. Mentre, tecnicamente, non chiameresti uno spazio un "carattere", per Bash invece è come se lo fosse letteralmente. Se elenchi i contenuti della tua directory corrente, invece di un file chiamato `my big file`, vedrai tre file denominati,

rispettivamente, `my`, `big` e `file`. Questo perché Bash ha ritenuto che tu volessi creare più file i cui nomi stavi passando in un elenco:

```
$ touch my big file
$ ls
my big file
```

Gli spazi verranno interpretati allo stesso modo se si eliminano (`rm`) i tre file in un solo comando:

```
$ rm my big file
```

Ora proviamo nel modo giusto. Digita `touch` e le tre parti del tuo nome file, ma questa volta racchiudi il nome tra virgolette. Questa volta ha funzionato. L'elenco dei contenuti della directory ti mostrerà un singolo file con il nome corretto.

```
$ touch "my big file"
$ ls
'my big file'
```

Esistono altri modi per ottenere lo stesso effetto. Le virgolette singole, per esempio, funzionano così come le virgolette doppie. (Nota che le virgolette singole conserveranno il valore letterale di tutti i caratteri, mentre le virgolette doppie conserveranno tutti i caratteri *ad eccezione* di `$`, ```, `\` e, in alcuni casi, `!`.)

```
$ rm 'my big file'
```

Anteporre un carattere di back slash (`\`) a un carattere speciale farà sì che Bash lo interpreti letteralmente.

```
$ touch my\ big\ file
```

Esercizi Guidati

1. Usa il comando `export` per aggiungere una nuova directory al tuo path (questa modifica non sopravviverà al riavvio).

2. Usa il comando `unset` per cancellare la variabile `PATH`. Prova a eseguire un comando usando `sudo` (per esempio `sudo cat /etc/shadow`) . Cosa è successo? Perché? (L'uscita dalla shell ti riporterà al tuo stato originale.)

Esercizi Esplorativi

1. Ricerca in Internet l'elenco completo dei caratteri speciali.
2. Prova a eseguire comandi usando stringhe composte da caratteri speciali e usando vari metodi per evitarli. Ci sono differenze tra il modo in cui questi metodi si comportano?

Sommario

In questa lezione abbiamo imparato:

- Come identificare le variabili d'ambiente del tuo sistema.
- Come creare le proprie variabili di ambiente ed esportarle in altre shell figlie.
- Come rimuovere le variabili d'ambiente e come usare sia i comandi `env` che `set`.
- Come gestire i caratteri speciali in modo che Bash li legga letteralmente.

In questa lezione sono stati discussi i seguenti comandi:

echo

Stampa in output stringhe e variabili.

env

Visualizza e modifica le variabili di ambiente.

export

Passa una variabile alle shell figlie.

unset

Rimuove i valori e gli attributi delle variabili e delle funzioni della shell.

Risposte agli Esercizi Guidati

1. Usa il comando `export` per aggiungere una nuova directory al tuo path (questa modifica non sopravviverà al riavvio).

Puoi aggiungere temporaneamente una nuova directory (chiamata per esempio `myfiles` che risiede nella tua home directory) al tuo percorso usando `export PATH="/home/youname/myfiles:$PATH"`. Crea un semplice script nella directory `myfiles` /, rendilo eseguibile e prova ad eseguirlo da una directory diversa. Questi comandi presumono che tu sia nella tua home directory che contiene una directory chiamata `myfiles`.

```
$ touch myfiles/myscript.sh
$ echo '#!/bin/bash' >> myfiles/myscript.sh
$ echo 'echo Hello' >> myfiles/myscript.sh
$ chmod +x myfiles/myscript.sh
$ myscript.sh
Hello
```

2. Usa il comando `unset` per cancellare la variabile `PATH`. Prova a eseguire un comando usando `sudo` (per esempio `sudo cat /etc/shadow`). Che cosa è successo? Perché? (L'uscita dalla shell ti riporterà al tuo stato originale.)

Digitando `unset PATH` si cancelleranno le impostazioni della variabile path corrente. Tentare di invocare un binario senza il suo percorso assoluto fallirà. Per questa ragione, tentare di eseguire un comando usando `sudo` (che è esso stesso un programma binario situato in `/usr/bin/sudo`) fallirà - a meno che non si specifichi la posizione assoluta, come in: `/usr/bin/sudo /bin/cat /etc/shadow`. Puoi resettare la tua `PATH` usando `export` o semplicemente uscendo dalla shell.

Risposte agli Esercizi Esplorativi

1. Ricerca in Internet l'elenco completo dei caratteri speciali.

Ecco la lista: `& ; | * ? " ' [ ] ( ) $ < > { } # / \ ! ~.`

2. Prova a eseguire comandi usando stringhe composte da caratteri speciali e usando vari metodi per evitarli. Ci sono differenze tra il modo in cui questi metodi si comportano?

Utilizzando il carattere `"` si manterranno i valori speciali del simbolo del dollaro (`$`), dell'apice traversa (`()`) e del back slash. L'utilizzo del carattere ```, annullerà invece *tutti* i significati speciali dei caratteri e consentirà una loro interpretazione letterale.

```
$ echo "$mynewvar"  
goodbye  
$ echo '$mynewvar'  
$mynewvar
```



103.2 Elaborare flussi di testo utilizzando i filtri

Obiettivi LPI di riferimento

LPIC-1 v5, Exam 101, Objective 103.2

Peso

2

Aree di Conoscenza Chiave

- Inviare file di testo e flussi di output tramite filtri di utilità di testo per modificare l'output utilizzando i comandi UNIX standard presenti nel pacchetto GNU textutils.

Di seguito è riportato un elenco parziale dei file, dei termini e dei comandi utilizzati

- `bzcat`
- `cat`
- `cut`
- `head`
- `less`
- `md5sum`
- `nl`
- `od`
- `paste`
- `sed`
- `sha256sum`
- `sha512sum`
- `sort`

- `split`
- `tail`
- `tr`
- `uniq`
- `wc`
- `xzcat`
- `zcat`



103.2 Lezione 1

Certificazione:	LPIC-1
Versione:	5.0
Argomento:	103 Comandi GNU e Unix
Obiettivo:	103.2 Elaborare flussi di testo utilizzando i filtri
Lezione:	1 di 1

Introduzione

Trattare il testo è una parte importante del lavoro di ogni amministratore di sistema. Doug McIlroy, un membro del team di sviluppo originale di Unix, ha sintetizzato la filosofia Unix e ha detto (tra le altre cose importanti): "[è necessario] Scrivere programmi per gestire i flussi di testo, perché questa è un'interfaccia universale". Linux si ispira al funzionamento del sistema Unix e adotta fortemente la sua filosofia, quindi un amministratore deve aspettarsi molti strumenti di manipolazione del testo all'interno di una distribuzione Linux.

Una Rapida Rassegna su Reindirizzamenti e Pipe

Ulteriori riferimenti alla filosofia Unix

- Scrivi programmi che facciano una cosa e la facciano bene.
- Scrivi programmi che lavorino insieme.

Un modo importante per far lavorare i programmi insieme è attraverso il *piping* e i *reindirizzamenti* (redirects). Praticamente tutti i programmi di manipolazione del testo partono da un input standard (*stdin*), lo inviano a un output standard (*stdout*) e inviano eventuali errori a

un output degli errori standard (*stderr*). Se non diversamente specificato, l'input standard è quello che viene digitato sulla tastiera (il programma lo legge dopo aver premuto il tasto Invio). Allo stesso modo, l'output standard e gli errori vengono visualizzati nella schermata del terminale. Vediamo come funziona il tutto.

Nel tuo terminale, digita `cat` e quindi premi il tasto Invio. Quindi digitare del testo casuale.

```
$ cat
This is a test
This is a test
Hey!
Hey!
It is repeating everything I type!
It is repeating everything I type!
(I will hit ctrl+c so I will stop this nonsense)
(I will hit ctrl+c so I will stop this nonsense)
^C
```

Per maggiori informazioni sul comando `cat` (il termine deriva da “concatenare”) fare riferimento alle pagine di man.

NOTE

Se stai lavorando su un'installazione davvero semplice di un server Linux, alcuni comandi come `info` e `less` potrebbero essere non disponibili. In tal caso, installare questi strumenti utilizzando la procedura corretta nel sistema in uso come descritto nelle lezioni corrispondenti.

Come dimostrato sopra, se non specifichi da dove `cat` deve leggere, questo lo farà dall'input standard (qualunque cosa tu digiti) e produrrà tutto ciò che legge sulla tua finestra del terminale (il suo output standard).

Ora prova quanto segue:

```
$ cat > mytextfile
This is a test
I hope cat is storing this to mytextfile as I redirected the output
I will hit ctrl+c now and check this
^C

$ cat mytextfile
This is a test
I hope cat is storing this to mytextfile as I redirected the output
```

```
I will hit ctrl+c now and check this
```

Il carattere `>` (Maggiore di) dice a `cat` di indirizzare il suo output sul file `mytextfile`, non sullo standard output. Ora prova questo:

```
$ cat mytextfile > mynewtextfile
$ cat mynewtextfile
This is a test
I hope cat is storing this to mytextfile as I redirected the output
I will hit ctrl+c now and check this
```

Questo ha l'effetto di copiare `mytextfile` in `mynewtextfile`. Puoi effettivamente verificare che questi due file abbiano lo stesso contenuto eseguendo il comando `diff`:

```
$ diff mynewtextfile mytextfile
```

Poiché non esiste alcun output, i file sono uguali. Ora prova un reindirizzamento accodato (*append redirection*) (`>>`):

```
$ echo 'This is my new line' >> mynewtextfile
$ diff mynewtextfile mytextfile
4d3
< This is my new line
```

Finora abbiamo usato i reindirizzamenti per creare e manipolare i file. Possiamo anche usare le pipe (rappresentate dal simbolo `|`) per reindirizzare l'output di un programma ad un altro programma. Cerchiamo di trovare le righe in cui si trova la parola "this":

```
$ cat mytextfile | grep this
I hope cat is storing this to mytextfile as I redirected the output
I will hit ctrl+c now and check this

$ cat mytextfile | grep -i this
This is a test
I hope cat is storing this to mytextfile as I redirected the output
I will hit ctrl+c now and check this
```

Ora abbiamo reindirizzato l'output di `cat` a un altro comando: `grep`. Nota che quando ignoriamo maiuscole e minuscole (usando l'opzione `-i`) otteniamo di conseguenza una riga in più.

Elaborare Flussi di Testo

Lettura di un File Compresso

Creeremo un file chiamato `ftu.txt` contenente un elenco dei seguenti comandi:

```
bzcat
cat
cut
head
less
md5sum
nl
od
paste
sed
sha256sum
sha512sum
sort
split
tail
tr
uniq
wc
xzcat
zcat
```

Ora useremo il comando `grep` per stampare tutte le linee contenenti la stringa `cat`:

```
$ cat ftu.txt | grep cat
bzcat
cat
xzcat
zcat
```

Un altro modo per ottenere queste informazioni è semplicemente usare il comando `grep` per filtrare direttamente il testo, senza la necessità di usare un'altra applicazione per inviare il flusso di testo all'*stdout*.

```
$ grep cat ftu.txt
bzcat
```

```
cat
xzcat
zcat
```

NOTE | Ricorda che ci sono molti modi per eseguire la stessa attività usando Linux.

Ci sono altri comandi che gestiscono i file compressi (`bzcat` per i file compressi `bzip`, `xzcat` per i file compressi `xz` e `zcat` per i file compressi `gzip`) e ognuno viene usato per visualizzare il contenuto di un file compresso basato sull'algoritmo di compressione utilizzato.

Verifica che il file appena creato `ftu.txt` sia l'unico nella directory, quindi crea una versione compressa del file `gzip`:

```
$ ls ftu*
ftu.txt

$ gzip ftu.txt
$ ls ftu*
ftu.txt.gz
```

Quindi, usa il comando `zcat` per visualizzare il contenuto del file compresso con `gzip`:

```
$ zcat ftu.txt.gz
bzcat
cat
cut
head
less
md5sum
nl
od
paste
sed
sha256sum
sha512sum
sort
split
tail
tr
uniq
wc
xzcat
```



```
zcat
```

Nota che `gzip` comprime `ftu.txt` in `ftu.txt.gz` e rimuoverà il file originale. Di default, non verrà visualizzato alcun output dal comando `gzip`. Comunque, se vuoi che `gzip` ti dica cosa sta facendo, usa l'opzione `-v` per l'output "verboso".

Visualizzare un File in un Paginatore

Sai che `cat` concatena un file nell'output standard (una volta che viene fornito un file dopo il comando). Il file `/var/log/syslog` è dove il tuo sistema Linux memorizza tutto quello che sta accadendo nel sistema. Usare il comando `sudo` per elevare i privilegi in modo da poter leggere il file `/var/log/syslog`:

```
$ sudo cat /var/log/syslog
```

- i. vedrai i messaggi scorrere molto velocemente nella finestra del tuo terminale. È possibile reindirizzare l'output al programma `less` in modo che i risultati vengano impaginati. Usando `less` puoi usare i tasti freccia per navigare nell'output e anche usare i comandi come utilizzassi `vi` per navigare e cercare in tutto il testo.

Tuttavia, invece di reindirizzare il comando `cat` in un programma di impaginazione, è più pragmatico usare direttamente il programma di impaginazione:

```
$ sudo less /var/log/syslog
... (output omitted for clarity)
```

Ottenere una Parte di un File di Testo

Se è necessario rivedere solo l'inizio o la fine di un file, sono disponibili altri metodi. Il comando `head` è usato per leggere di default le prime dieci righe di un file, e il comando `tail` è usato per leggere di default le ultime dieci righe di un file. Ora prova:

```
$ sudo head /var/log/syslog
Nov 12 08:04:30 hypatia rsyslogd: [origin software="rsyslogd" swVersion="8.1910.0" x-
pid="811" x-info="https://www.rsyslog.com"] rsyslogd was HUPed
Nov 12 08:04:30 hypatia systemd[1]: logrotate.service: Succeeded.
Nov 12 08:04:30 hypatia systemd[1]: Started Rotate log files.
Nov 12 08:04:30 hypatia vdr: [928] video directory scanner thread started (pid=882, tid=928,
prio=low)
Nov 12 08:04:30 hypatia vdr: [882] registered source parameters for 'A - ATSC'
```

```

Nov 12 08:04:30 hypatia vdr: [882] registered source parameters for 'C - DVB-C'
Nov 12 08:04:30 hypatia vdr: [882] registered source parameters for 'S - DVB-S'
Nov 12 08:04:30 hypatia vdr: [882] registered source parameters for 'T - DVB-T'
Nov 12 08:04:30 hypatia vdr[882]: vdr: no primary device found - using first device!
Nov 12 08:04:30 hypatia vdr: [929] epg data reader thread started (pid=882, tid=929,
prio=high)
$ sudo tail /var/log/syslog
Nov 13 10:24:45 hypatia kernel: [ 8001.679238] mce: CPU7: Core temperature/speed normal
Nov 13 10:24:46 hypatia dbus-daemon[2023]: [session uid=1000 pid=2023] Activating via
systemd: service name='org.freedesktop.Tracker1.Miner.Extract' unit='tracker-
extract.service' requested by ':1.73' (uid=1000 pid=2425 comm="/usr/lib/tracker/tracker-
miner-fs ")
Nov 13 10:24:46 hypatia systemd[2004]: Starting Tracker metadata extractor...
Nov 13 10:24:47 hypatia dbus-daemon[2023]: [session uid=1000 pid=2023] Successfully
activated service 'org.freedesktop.Tracker1.Miner.Extract'
Nov 13 10:24:47 hypatia systemd[2004]: Started Tracker metadata extractor.
Nov 13 10:24:54 hypatia kernel: [ 8010.462227] mce: CPU0: Core temperature above threshold,
cpu clock throttled (total events = 502907)
Nov 13 10:24:54 hypatia kernel: [ 8010.462228] mce: CPU4: Core temperature above threshold,
cpu clock throttled (total events = 502911)
Nov 13 10:24:54 hypatia kernel: [ 8010.469221] mce: CPU0: Core temperature/speed normal
Nov 13 10:24:54 hypatia kernel: [ 8010.469222] mce: CPU4: Core temperature/speed normal
Nov 13 10:25:03 hypatia systemd[2004]: tracker-extract.service: Succeeded.

```

Per indicare il numero di righe visualizzate, possiamo reindirizzare l'output del comando `head` al comando `nl`, che mostrerà il numero di righe di testo trasmesse al comando:

```

$ sudo head /var/log/syslog | nl
1 Nov 12 08:04:30 hypatia rsyslogd: [origin software="rsyslogd" swVersion="8.1910.0" x-
pid="811" x-info="https://www.rsyslog.com"] rsyslogd was HUPed
2 Nov 12 08:04:30 hypatia systemd[1]: logrotate.service: Succeeded.
3 Nov 12 08:04:30 hypatia systemd[1]: Started Rotate log files.
4 Nov 12 08:04:30 hypatia vdr: [928] video directory scanner thread started (pid=882,
tid=928, prio=low)
5 Nov 12 08:04:30 hypatia vdr: [882] registered source parameters for 'A - ATSC'
6 Nov 12 08:04:30 hypatia vdr: [882] registered source parameters for 'C - DVB-C'
7 Nov 12 08:04:30 hypatia vdr: [882] registered source parameters for 'S - DVB-S'
8 Nov 12 08:04:30 hypatia vdr: [882] registered source parameters for 'T - DVB-T'
9 Nov 12 08:04:30 hypatia vdr[882]: vdr: no primary device found - using first device!
10 Nov 12 08:04:30 hypatia vdr: [929] epg data reader thread started (pid=882, tid=929,
prio=high)

```

Possiamo fare lo stesso eseguendo il piping dell'output del comando `tail` al comando `wc`, che di

default conta il numero di parole all'interno di un documento, e usando l'opzione `-l` per stampare il numero di righe di testo che il comando ha letto:

```
$ sudo tail /var/log/syslog | wc -l
10
```

Se un amministratore deve visualizzare più (o meno) parti dall'inizio o dalla fine di un file, l'opzione `-n` può essere usata per definire l'output dei comandi:

```
$ sudo tail -n 5 /var/log/syslog
Nov 13 10:37:24 hypatia systemd[2004]: tracker-extract.service: Succeeded.
Nov 13 10:37:42 hypatia dbus-daemon[2023]: [session uid=1000 pid=2023] Activating via
systemd: service name='org.freedesktop.Tracker1.Miner.Extract' unit='tracker-
extract.service' requested by ':1.73' (uid=1000 pid=2425 comm="/usr/lib/tracker/tracker-
miner-fs ")
Nov 13 10:37:42 hypatia systemd[2004]: Starting Tracker metadata extractor...
Nov 13 10:37:43 hypatia dbus-daemon[2023]: [session uid=1000 pid=2023] Successfully
activated service 'org.freedesktop.Tracker1.Miner.Extract'
Nov 13 10:37:43 hypatia systemd[2004]: Started Tracker metadata extractor.
$ sudo head -n 12 /var/log/syslog
Nov 12 08:04:30 hypatia rsyslogd: [origin software="rsyslogd" swVersion="8.1910.0" x-
pid="811" x-info="https://www.rsyslog.com"] rsyslogd was HUPed
Nov 12 08:04:30 hypatia systemd[1]: logrotate.service: Succeeded.
Nov 12 08:04:30 hypatia systemd[1]: Started Rotate log files.
Nov 12 08:04:30 hypatia vdr: [928] video directory scanner thread started (pid=882, tid=928,
prio=low)
Nov 12 08:04:30 hypatia vdr: [882] registered source parameters for 'A - ATSC'
Nov 12 08:04:30 hypatia vdr: [882] registered source parameters for 'C - DVB-C'
Nov 12 08:04:30 hypatia vdr: [882] registered source parameters for 'S - DVB-S'
Nov 12 08:04:30 hypatia vdr: [882] registered source parameters for 'T - DVB-T'
Nov 12 08:04:30 hypatia vdr[882]: vdr: no primary device found - using first device!
Nov 12 08:04:30 hypatia vdr: [929] epg data reader thread started (pid=882, tid=929,
prio=high)
Nov 12 08:04:30 hypatia vdr: [882] no DVB device found
Nov 12 08:04:30 hypatia vdr: [882] initializing plugin: vnsiserver (1.8.0): VDR-Network-
Streaming-Interface (VNSI) Server
```

Le Basi di sed, l'Editor di Flussi di Testo

Diamo un'occhiata agli altri file, termini e utilità che non hanno "cat" nei loro nomi. Possiamo farlo passando l'opzione `-v` a `grep`, che indicherà al comando di produrre solo le righe che non

contengono `cat`:

```
$ zcat ftu.txt.gz | grep -v cat
cut
head
less
md5sum
nl
od
paste
sed
sha256sum
sha512sum
sort
split
tail
tr
uniq
wc
```

Gran parte di ciò che possiamo fare con `grep` possiamo anche farla con `sed` - l'editor di flussi per filtrare e trasformare il testo (come indicato nella pagina di manuale di `sed`). Per prima cosa ripristineremo il nostro file `ftu.txt` decomprimendo il nostro archivio `gzip`:

```
$ gunzip ftu.txt.gz
$ ls ftu*
ftu.txt
```

Ora, possiamo usare `sed` per elencare solo le righe che contengono la stringa `cat`:

```
$ sed -n /cat/p < ftu.txt
bzcac
cat
xzcat
zcat
```

Abbiamo usato il segno minore `<` per indirizzare il contenuto del file `ftu.txt` nel nostro comando `sed`. La stringa racchiusa tra *slash* (ovvero `/cat/`) è il termine che stiamo cercando. L'opzione `-n` indica a `sed` di non produrre output (tranne quello successivamente indicato dal comando `p`). Prova a eseguire questo stesso comando senza l'opzione `-n` per vedere cosa succede. Quindi prova

questo:

```
$ sed /cat/d < ftu.txt
cut
head
less
md5sum
nl
od
paste
sed
sha256sum
sha512sum
sort
split
tail
tr
uniq
wc
```

Se non usiamo l'opzione `-n`, `sed` visualizzerà il contenuto proveniente dal file ad eccezione di ciò che `d` indica a `sed` di cancellare dal suo output.

Un uso comune di `sed` è trovare e sostituire il testo all'interno di un file. Supponiamo di voler cambiare ogni ricorrenza di `cat` in `dog`. Puoi usare `sed` per fare ciò fornendo l'opzione `s` per scambiare ogni occorrenza del primo termine, `cat`, per il secondo termine, `dog`:

```
$ sed s/cat/dog/ < ftu.txt
bzdog
dog
cut
head
less
md5sum
nl
od
paste
sed
sha256sum
sha512sum
sort
split
```

```
tail
tr
uniq
wc
xzdog
zdog
```

Invece di usare un operatore di reindirizzamento (<) per passare il file `ftu.txt` nel nostro comando `sed`, possiamo semplicemente far funzionare direttamente il comando `sed` sul file. Lo proveremo ora, creando contemporaneamente un backup del file originale:

```
$ sed -i.backup s/cat/dog/ ftu.txt
$ ls ftu*
ftu.txt  ftu.txt.backup
```

L'opzione `-i` eseguirà un'operazione `sed` sul tuo file *originale*. Se non usi il `.backup` dopo il parametro `-i`, avresti semplicemente riscritto il tuo file *originale*. Qualunque cosa tu usi come testo dopo il parametro `-i` sarà il nome in cui il file originale verrà salvato prima delle modifiche che hai richiesto a `sed`.

Garantire l'Integrità dei Dati

Abbiamo dimostrato quanto sia facile manipolare i file in Linux. Ci sono momenti in cui potresti voler distribuire un file a qualcun altro e vuoi essere sicuro che il destinatario riceva una copia autentica del file originale. Un uso molto comune di questa tecnica viene praticato quando i server di distribuzione Linux ospitano immagini CD o DVD scaricabili del loro software insieme a file che contengono i valori di *checksum* calcolati di tali immagini disco. Ecco per esempio un elenco da un mirror di download della distribuzione Debian:

[PARENTDIR]	Parent Directory	-
[SUM]	MD5SUMS	2019-09-08 17:46 274
[CRT]	MD5SUMS.sign	2019-09-08 17:52 833
[SUM]	SHA1SUMS	2019-09-08 17:46 306
[CRT]	SHA1SUMS.sign	2019-09-08 17:52 833
[SUM]	SHA256SUMS	2019-09-08 17:46 402
[CRT]	SHA256SUMS.sign	2019-09-08 17:52 833
[SUM]	SHA512SUMS	2019-09-08 17:46 658
[CRT]	SHA512SUMS.sign	2019-09-08 17:52 833
[ISO]	debian-10.1.0-amd64-netinst.iso	2019-09-08 04:37 335M
[ISO]	debian-10.1.0-amd64-xfce-CD-1.iso	2019-09-08 04:38 641M

```
[ISO]      debian-edu-10.1.0-amd64-netinst.iso  2019-09-08 04:38 405M
[ISO]      debian-mac-10.1.0-amd64-netinst.iso  2019-09-08 04:38 334M
```

Nell'elenco sopra, i file di immagine dell'installatore Debian sono accompagnati da file di testo che contengono checksum dei file generati da vari algoritmi (MD5, SHA1, SHA256 e SHA512).

NOTE

Un checksum è un valore derivato da un calcolo matematico, basato su una funzione hash crittografica, rispetto a un file. Esistono diversi tipi di funzioni *hash* crittografiche che variano di intensità. L'esame prevede che tu abbia familiarità con l'uso di `md5sum`, `sha256sum` e `sha512sum`.

Una volta scaricato un file (per esempio, l'immagine `debian-10.1.0-amd64-netinst.iso`), confronterai il checksum del file scaricato con un valore di checksum che ti è stato fornito.

Ecco un esempio per illustrare quanto fin qui detto. Calcoleremo il valore SHA256 del file `ftu.txt` usando il comando `sha256sum`:

```
$ sha256sum ftu.txt
345452304fc26999a715652543c352e5fc7ee0c1b9deac6f57542ec91daf261c  ftu.txt
```

La lunga stringa di caratteri che precede il nome del file è il valore di checksum SHA256 di questo file di testo. Creiamo un file che contenga quel valore, in modo da poterlo utilizzare per verificare l'integrità del nostro file originale di testo. Possiamo farlo con lo stesso comando `sha256sum` e reindirizzare l'output su un file:

```
$ sha256sum ftu.txt > sha256.txt
```

Ora, per verificare il file `ftu.txt`, utilizziamo semplicemente lo stesso comando e forniamo il nome file che contiene il nostro valore di checksum insieme all'opzione `-c`:

```
$ sha256sum -c sha256.txt
ftu.txt: OK
```

Il valore contenuto nel file corrisponde al checksum SHA256 calcolato per il nostro file `ftu.txt`, proprio come ci aspetteremmo. Tuttavia, se il file originale fosse modificato (come alcuni byte persi durante il download di un file, o se qualcuno lo avesse deliberatamente manomesso), il controllo del valore fallirebbe. In tali casi sappiamo che il nostro file è danneggiato o corrotto e non possiamo fidarci dell'integrità del suo contenuto. Per dimostrare il punto, aggiungeremo del

testo alla fine del file:

```
$ echo "new entry" >> ftu.txt
```

Ora faremo un tentativo di verificare l'integrità del file:

```
$ sha256sum -c sha256.txt
ftu.txt: FAILED
sha256sum: WARNING: 1 computed checksum did NOT match
```

Vediamo che il checksum non corrisponde a quanto previsto per il file. Pertanto, non potevamo fidarci dell'integrità di questo file. Potremmo tentare di scaricare una nuova copia di un file, segnalare l'errore del checksum al mittente del file o segnalarlo a un team di sicurezza del data center in base all'importanza del file.

Guardare più in Profondità nei File

Il comando `octal dump` (`od`) è spesso usato per il debug di applicazioni e vari file. Di per sé, il comando `od` elencherà semplicemente i contenuti di un file in formato ottale. Possiamo usare il nostro file `ftu.txt` in precedenza per esercitarci con questo comando:

```
$ od ftu.txt
0000000 075142 060543 005164 060543 005164 072543 005164 062550
0000020 062141 066012 071545 005163 062155 071465 066565 067012
0000040 005154 062157 070012 071541 062564 071412 062145 071412
0000060 060550 032462 071466 066565 071412 060550 030465 071462
0000100 066565 071412 071157 005164 070163 064554 005164 060564
0000120 066151 072012 005162 067165 070551 073412 005143 075170
0000140 060543 005164 061572 072141 000012
0000151
```

La prima colonna di output è il *byte offset* per ogni riga di output. Poiché `od` stampa le informazioni in formato ottale di default, ogni riga inizia con un byte offset di otto bit, seguito da otto colonne, ognuna contenente il valore ottale dei dati all'interno di quella colonna.

TIP | Ricordiamoci che un *byte* è composto da 8 bit.

Se dovessi visualizzare il contenuto di un file in formato esadecimale, usa l'opzione `-x`:

```
$ od -x ftu.txt
```



```

0000000 7a62 6163 0a74 6163 0a74 7563 0a74 6568
0000020 6461 6c0a 7365 0a73 646d 7335 6d75 6e0a
0000040 0a6c 646f 700a 7361 6574 730a 6465 730a
0000060 6168 3532 7336 6d75 730a 6168 3135 7332
0000100 6d75 730a 726f 0a74 7073 696c 0a74 6174
0000120 6c69 740a 0a72 6e75 7169 770a 0a63 7a78
0000140 6163 0a74 637a 7461 000a
0000151

```

Ora ciascuna delle otto colonne dopo il byte offset è rappresentata dal loro equivalente esadecimale.

Un utile uso del comando `od` è per il debug degli script. Per esempio, il comando `od` può mostrarci caratteri normalmente non presenti in un file, come per esempio quelli di *newline*. Possiamo farlo con l'opzione `-c`, in modo che, invece di visualizzare la notazione numerica per ogni byte, queste voci di colonna verranno mostrate come equivalenti di carattere:

```

$ od -c ftu.txt
0000000 b z c a t \n c a t \n c u t \n h e
0000020 a d \n l e s s \n m d 5 s u m \n n
0000040 l \n o d \n p a s t e \n s e d \n s
0000060 h a 2 5 6 s u m \n s h a 5 1 2 s
0000100 u m \n s o r t \n s p l i t \n t a
0000120 i l \n t r \n u n i q \n w c \n x z
0000140 c a t \n z c a t \n
0000151

```

Tutte le voci *newline* all'interno del file sono rappresentate dai caratteri nascosti `\n`. Se si desidera solo visualizzare tutti i caratteri all'interno di un file e non è necessario visualizzare le informazioni di byte offset, questa colonna può essere rimossa dall'output in questo modo:

```

$ od -An -c ftu.txt
b z c a t \n c a t \n c u t \n h e
a d \n l e s s \n m d 5 s u m \n n
l \n o d \n p a s t e \n s e d \n s
h a 2 5 6 s u m \n s h a 5 1 2 s
u m \n s o r t \n s p l i t \n t a
i l \n t r \n u n i q \n w c \n x z
c a t \n z c a t \n

```

Esercizi Guidati

1. Qualcuno ha appena donato un laptop alla tua scuola e ora desideri installare Linux su di esso. Non esiste un manuale e sei stato costretto ad avviarlo senza interfaccia grafica da una chiavetta USB. Ottieni un terminale shell e sai che, per ogni processore, ci saranno una serie di riferimenti nel file `/proc/cpuinfo`:

```
processor      : 0
vendor_id     : GenuineIntel
cpu family    : 6
model         : 158
```

(linee saltate)

```
processor      : 1
vendor_id     : GenuineIntel
cpu family    : 6
model         : 158
```

(altre linee saltate)

- Usando i comandi `grep` e `wc` mostra quanti processori hai.

- Fai la stessa cosa con `sed` invece che con `grep`.

2. Esplora il tuo file locale `/etc/passwd` con i comandi `grep`, `sed`, `head` e `tail` per le attività seguenti:

- Quali utenti hanno accesso a una shell Bash?

- Il tuo sistema ha vari utenti che esistono per gestire programmi specifici o per scopi amministrativi. Non hanno accesso a una shell. Quanti ne esistono nel tuo sistema?

- Quanti utenti esistono nel tuo sistema (ricorda: usa solo il file `/etc/passwd`)?

- Elenca solo la prima riga, l'ultima riga e la decima riga del tuo file `/etc/passwd`.

3. Considera questo esempio di file `/etc/passwd`. Copiare le righe seguenti in un file locale chiamato `mypasswd` per questo esercizio.

```
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
nvidia-persistenced:x:121:128:NVIDIA Persistence Daemon,,,:/nonexistent:/sbin/nologin
libvirt-qemu:x:64055:130:Libvirt Qemu,,,:/var/lib/libvirt:/usr/sbin/nologin
libvirt-dnsmasq:x:122:133:Libvirt Dnsmasq,,,:/var/lib/libvirt/dnsmasq:/usr/sbin/nologin
carol:x:1000:2000:Carol Smith,Finance,,Main Office:/home/carol:/bin/bash
dave:x:1001:1000:Dave Edwards,Finance,,Main Office:/home/dave:/bin/ksh
emma:x:1002:1000:Emma Jones,Finance,,Main Office:/home/emma:/bin/bash
frank:x:1003:1000:Frank Cassidy,Finance,,Main Office:/home/frank:/bin/bash
grace:x:1004:1000:Grace Kearns,Engineering,,Main Office:/home/grace:/bin/ksh
henry:x:1005:1000:Henry Adams,Sales,,Main Office:/home/henry:/bin/bash
john:x:1006:1000:John Chapel,Sales,,Main Office:/home/john:/bin/bash
```

- Elenca tutti gli utenti nel gruppo `1000` (usa `sed` per selezionare solo il campo appropriato) dal tuo file `mypasswd`.
- Elenca solo i nomi completi di tutti gli utenti di questo gruppo (usa `sed` e `cut`).

Esercizi Esplorativi

1. Ancora una volta usando il file `mypasswd` degli esercizi precedenti, creare un comando Bash che selezioni un individuo del *Main Office* per fargli vincere un concorso a premi. Utilizzare il comando `sed` per visualizzare solo le righe relative al main Office, quindi una sequenza di comandi `cut` per recuperare il nome di ciascun utente da queste righe. Successivamente si procederà a ordinare casualmente questi nomi e a visualizzare solo il nome in testa all'elenco.

2. Quante persone lavorano in Finance, Engineering e Sales? (Prova a utilizzare il comando `uniq`.)

3. Ora vuoi preparare un file CSV (valori separati da virgola) in modo da poter importare facilmente, dal file `mypasswd` nell'esempio precedente, il file `names.csv` in LibreOffice. Il contenuto del file avrà il seguente formato:

```
First Name,Last Name,Position
Carol,Smith,Finance
...
John,Chapel,Sales
```

TIP

Utilizza i comandi `sed`, `cut` e `paste` per ottenere i risultati desiderati. Nota che la virgola (,) sarà il delimitatore per questo file.

4. Supponiamo che il foglio di calcolo `names.csv` creato nell'esercizio precedente sia un file importante e vogliamo assicurarci che nessuno lo manometterà dal momento in cui lo inviamo a qualcuno fino a quando il nostro destinatario lo riceverà. Come possiamo assicurare l'integrità di questo file usando `md5sum`?

5. Ti sei ripromesso di leggere cento righe di un testo classico al giorno e hai deciso di iniziare con *Mariner and Mystic* di Herman Melville. Elabora un comando usando `split` che separerà questo libro in sezioni di 100 righe ciascuna. Per ottenere il libro in formato testo normale, cercalo su <https://www.gutenberg.org>.

6. Usando `ls -l` nella directory `/etc`, che tipo di elenco ottieni? Usando il comando `cut` sull'output del comando `ls` come visualizzeresti solo i nomi dei file? E come il nome del file e del proprietario dei file? Insieme ai comandi `ls -l` e `cut`, usa il comando `tr` per *sostituire* più occorrenze di uno spazio con un singolo spazio per aiutare a formattare l'output generato da

un comando `cut`.

7. Questo esercizio presuppone che tu sia su una macchina reale (non una macchina virtuale). Devi anche avere una chiavetta USB con te. Esamina le pagine del manuale per il comando `tail` e scopri come *seguire* un file mentre il testo viene aggiunto ad esso. Durante il monitoraggio dell'output di un comando `tail` sul file `/var/log/syslog`, inserire una chiavetta USB. Scrivi il comando completo che utilizzeresti per ottenere *Product*, *Manufacturer* e *total amount of memory* della tua chiavetta USB.

Sommario

La gestione dei flussi di testo è di grande importanza in qualsiasi sistema Linux. I flussi di testo possono essere elaborati utilizzando gli script per automatizzare le attività quotidiane o trovare informazioni di debug rilevanti nei file di log. Ecco un breve riassunto dei comandi trattati in questa lezione:

cat

Usato per combinare o leggere file di testo semplice..

bzcat

Permette l'elaborazione o la lettura di file compressi usando il metodo `bzip2`.

xzcat

Permette l'elaborazione o la lettura di file compressi usando il metodo `xz`.

zcat

Permette l'elaborazione o la lettura di file compressi usando il metodo `gzip`.

less

Questo comando impagina il contenuto di un file e consente la navigazione e la funzionalità di ricerca.

head

Questo comando visualizza le prime 10 righe di un file per impostazione predefinita. Con l'uso dell'opzione `-n` è possibile visualizzare più o meno righe.

tail

Questo comando visualizza le ultime 10 righe di un file per impostazione predefinita. Con l'uso dell'opzione `-n` è possibile visualizzare più o meno righe. L'opzione `-f` è usata per seguire l'output di un file di testo in cui vengono scritti nuovi dati.

wc

Abbreviazione di “word count”, ma a seconda dei parametri che si utilizzano conterà caratteri, parole e righe.

sort

Utilizzato per organizzare l'output di un elenco in ordine alfabetico, in ordine alfabetico inverso o in ordine casuale.

uniq

Utilizzato per elencare (e contare) stringhe uguali.

od

Il comando "octal dump" viene utilizzato per visualizzare un file binario in notazione ottale, decimale o esadecimale.

nl

Il comando "number line" visualizza il numero di linee in un file, oltre che a ricreare un file con ogni riga preceduta dal suo numero di riga.

sed

L'editor di flusso può essere utilizzato per trovare occorrenze corrispondenti delle stringhe utilizzando Regular Expressions e per modificare i file utilizzando modelli predefiniti.

tr

Il comando "translate" può sostituire i caratteri e rimuove e comprime i caratteri ricorrenti.

cut

Questo comando può estrapolare parti di file di testo come fossero campi, se identificati da un delimitatore di caratteri.

paste

Unisce i file in colonne in base all'uso dei separatori di campo.

split

Questo comando può dividere file più grandi in file più piccoli a seconda dei criteri impostati dalle opzioni del comando.

md5sum

Utilizzato per il calcolo del valore *hash* MD5 di un file. Utilizzato anche per verificare un file rispetto a un valore hash esistente per garantirne l'integrità.

sha256sum

Utilizzato per il calcolo del valore hash SHA256 di un file. Utilizzato anche per verificare un file rispetto a un valore hash esistente per garantirne l'integrità.

sha512sum

Utilizzato per il calcolo del valore hash SHA512 di un file. Utilizzato anche per verificare un file rispetto a un valore hash esistente per garantirne l'integrità.

Risposte agli Esercizi Guidati

1. Qualcuno ha appena donato un laptop alla tua scuola e ora desideri installare Linux su di esso. Non esiste un manuale e sei stato costretto ad avviarlo senza interfaccia grafica da una chiavetta USB. Ottieni un terminale shell e sai che, per ogni processore, ci saranno una serie di riferimenti nel file `/proc/cpuinfo`:

```
processor      : 0
vendor_id     : GenuineIntel
cpu family    : 6
model         : 158
```

(linee saltate)

```
processor      : 1
vendor_id     : GenuineIntel
cpu family    : 6
model         : 158
```

(altre linee saltate)

- Usando i comandi `grep` e `wc` mostra quanti processori hai.

Ecco due varianti:

```
$ cat /proc/cpuinfo | grep processor | wc -l
$ grep processor /proc/cpuinfo | wc -l
```

Ora che sai che ci sono diversi modi in cui puoi fare la stessa cosa, ma quando dovresti usare l'uno o l'altro? Dipende davvero da diversi fattori, i due più importanti sono le prestazioni e la leggibilità. La maggior parte delle volte utilizzerai i comandi della shell all'interno di script per automatizzare le tue attività e più grandi e complessi saranno gli script, più dovrai preoccuparti di mantenerli veloci.

- Fai la stessa cosa con `sed` invece di `grep`

Ora, invece di `grep`, proveremo con `sed`:

```
$ sed -n /processor/p /proc/cpuinfo | wc -l
```


Qui abbiamo usato `sed` con l'opzione `-n` in modo che `sed` non stamperà nulla tranne ciò che corrisponde all'espressione `processor`, come indicato dal comando `p`. Come abbiamo fatto con le soluzioni `grep`, `wc -l` conterà il numero di linee, quindi il numero di processori che abbiamo.

Studia il prossimo esempio:

```
$ sed -n /processor/p /proc/cpuinfo | sed -n '$='
```

Questa sequenza di comandi fornisce risultati identici all'esempio precedente in cui l'output di `sed` è stato reindirizzato al comando `wc`. La differenza qui è che invece di usare `wc -l` per contare il numero di righe, `sed` viene nuovamente invocato per fornire funzionalità equivalenti. Ancora una volta, stiamo sopprimendo l'output di `sed` con l'opzione `-n`, ad eccezione dell'espressione che stiamo esplicitamente chiamando, che è `'$='`. Questa espressione dice a `sed` di trovare l'ultima riga (\$) e quindi di visualizzarne il numero relativo (=).

2. Esplora il tuo file locale `/etc/passwd` con i comandi `grep`, `sed`, `head` e `tail` per le attività seguenti:

- Quali utenti hanno accesso a una shell Bash?

```
$ grep ":/bin/bash$" /etc/passwd
```

Miglioreremo questa risposta visualizzando solo il nome dell'utente che utilizza la shell Bash.

```
$ grep ":/bin/bash$" /etc/passwd | cut -d: -f1
```

Il nome utente è il primo campo (parametro `-f1` del comando `cut`) e il file `/etc/passwd` usa `:` come separatore (parametro `-d:` del comando `cut`) così basta reindirizzare l'output del comando `grep` al comando `cut` appropriato.

- Il tuo sistema ha vari utenti per gestire programmi specifici o per scopi amministrativi. Non hanno accesso a una shell. Quanti ne esistono nel tuo sistema?

Il modo più semplice per trovarli è stampare le righe per gli account che non usano la shell Bash:

```
$ grep -v ":/bin/bash$" /etc/passwd | wc -l
```

- Quanti utenti e gruppi esistono nel tuo sistema (ricorda: usa solo il file `/etc/passwd`)

Il primo campo di una determinata riga nel file `/etc/passwd` è il nome utente, il secondo è in genere una `x` che indica che la password dell'utente non è memorizzata qui (è crittografata nel file `/etc/shadow`). Il terzo è l'id utente (UID) e il quarto è l'id gruppo (GID). Quindi questo dovrebbe darci il numero di utenti:

```
$ cut -d: -f3 /etc/passwd | wc -l
```

Bene, il più delle volte questo dovrebbe bastare. Tuttavia, ci sono situazioni in cui imposterai diversi super utenti o altri tipi speciali di utenti che condividono lo stesso UID (ID utente). Quindi, per essere al sicuro, inoltreremo il risultato del nostro comando `cut` al comando `sort` e poi conteremo il numero di righe.

```
$ cut -d: -f3 /etc/passwd | sort -u | wc -l
```

Ora, il numero di gruppi:

```
$ cut -d: -f4 /etc/passwd | sort -u | wc -l
```

- Elenca solo la prima riga, l'ultima riga e la decima riga del tuo file `/etc/passwd`

Questo si realizzerà attraverso:

```
$ sed -n -e '1'p -e '10'p -e '$'p /etc/passwd
```

Ricorda che il parametro `-n` dice a `sed` di non stampare altro che ciò che è specificato dal parametro `p`. Il simbolo del dollaro (\$) usato nell'esempio indica l'ultima riga del file.

3. Considera questo esempio di file `/etc/passwd`. Copia le righe seguenti in un file locale chiamato `mypasswd` per questo esercizio.

```
root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
```

```
sync:x:4:65534:sync:/bin:/bin/sync
nvidia-persistenced:x:121:128:NVIDIA Persistence Daemon,,,:/nonexistent:/sbin/nologin
libvirt-qemu:x:64055:130:Libvirt Qemu,,,:/var/lib/libvirt:/usr/sbin/nologin
libvirt-dnsmasq:x:122:133:Libvirt Dnsmasq,,,:/var/lib/libvirt/dnsmasq:/usr/sbin/nologin
carol:x:1000:2000:Carol Smith,Finance,,Main Office:/home/carol:/bin/bash
dave:x:1001:1000:Dave Edwards,Finance,,Main Office:/home/dave:/bin/ksh
emma:x:1002:1000:Emma Jones,Finance,,Main Office:/home/emma:/bin/bash
frank:x:1003:1000:Frank Cassidy,Finance,,Main Office:/home/frank:/bin/bash
grace:x:1004:1000:Grace Kearns,Engineering,,Main Office:/home/grace:/bin/ksh
henry:x:1005:1000:Henry Adams,Sales,,Main Office:/home/henry:/bin/bash
john:x:1006:1000:John Chapel,Sales,,Main Office:/home/john:/bin/bash
```

- Elenca tutti gli utenti nel gruppo 1000 (usa `sed` per selezionare solo il campo appropriato) dal tuo file `mypasswd`.

Il GID è il quarto campo nel file `/etc/passwd`. Potresti essere tentato di provare questo:

```
$ sed -n /1000/p mypasswd
```

In questo caso otterrai anche questa riga:

```
carol:x:1000:2000:Carol Smith,Finance,,Main Office:/home/carol:/bin/bash
```

Sai che questo non è corretto poiché Carol Smith è un membro del GID 2000 e la corrispondenza è avvenuta a causa dell'UID. Tuttavia, potresti aver notato che dopo il GID il campo successivo inizia con un carattere maiuscolo. Possiamo usare un'espressione regolare per risolvere questo problema.

```
$ sed -n /:1000:[A-Z]/p mypasswd
```

L'espressione `[A-Z]` corrisponderà a ogni singolo carattere maiuscolo. Imparerai di più al riguardo nella lezione dedicata.

- Elenca solo i nomi completi di tutti gli utenti per questo gruppo (usa `sed` e `cut`):

Usa la stessa tecnica che hai usato per risolvere la prima parte di questo esercizio e inoltralo a un comando `cut`.

```
$ sed -n /:1000:[A-Z]/p mypasswd | cut -d: -f5
Dave Edwards,Finance,,Main Office
```

```
Emma Jones,Finance,,,Main Office
Frank Cassidy,Finance,,,Main Office
Grace Kearns,Engineering,,,Main Office
Henry Adams,Sales,,,Main Office
John Chapel,Sales,,,Main Office
```

Non ci siamo ancora! Nota come i campi all'interno dei tuoi risultati possono essere separati da `,`. Quindi reindirizzeremo l'output ad un altro comando `cut`, usando `,` come delimitatore.

```
$ sed -n /:1000:[A-Z]/p mypasswd | cut -d: -f5 | cut -d, -f1
Dave Edwards
Emma Jones
Frank Cassidy
Grace Kearns
Henry Adams
John Chapel
```

Risposte agli Esercizi Esplorativi

1. Ancora una volta usando il file `mypasswd` degli esercizi precedenti, crea un comando Bash che selezionerà un individuo dal Main Office per vincere una lotteria. Usa il comando `sed` per stampare solo le righe per il Main Office, e poi una sequenza di comandi `cut` per recuperare il primo nome di ogni utente da queste righe. Successivamente dovrai ordinare in modo casuale questi nomi e stampare solo il nome superiore dall'elenco.

Prima esplora come l'opzione `-R` manipoli l'output del comando `sort`. Ripeti questo comando un paio di volte sulla tua macchina (nota che dovrai racchiudere 'Main Office' tra virgolette singole, quindi `sed` lo gestirà come una singola stringa):

```
$ sed -n '/Main Office'/p mypasswd | cut -d: -f5 | cut -d, -f1 | sort -R
```

Ecco la soluzione al tuo problema:

```
$ sed -n '/Main Office'/p mypasswd | cut -d: -f5 | cut -d, -f1 | sort -R | head -1
```

2. Quante persone lavorano in Finance, Engineering e Sales? (Prova a utilizzare il comando `uniq`.)

Continua a sviluppare ciò che hai imparato dagli esercizi precedenti. Prova quanto segue:

```
$ sed -n '/Main Office'/p mypasswd
$ sed -n '/Main Office'/p mypasswd | cut -d, -f2
```

Nota ora che non ci interessa il delimitatore `:`. Vogliamo solo il secondo campo quando dividiamo le righe con i caratteri `",`.

```
$ sed -n '/Main Office'/p mypasswd | cut -d, -f2 | uniq -c
  4 Finance
  1 Engineering
  2 Sales
```

Il comando `uniq` produrrà solo le righe uniche (non le righe ripetute) e il parametro `-c` dice a `uniq` di contare le occorrenze delle righe uguali. C'è un avvertimento qui: `uniq` prenderà in considerazione solo le linee adiacenti. Quando questo non sarà il caso, dovrai usare il comando `sort`.

3. Ora vuoi preparare un file CSV (valori separati da virgola) in modo da poter importare

facilmente, dal file `mypasswd` nell'esempio precedente, il file `names.csv` in LibreOffice. Il contenuto del file avrà il seguente formato:

```
First Name,Last Name,Position
Carol,Smith,Finance
...
John,Chapel,Sales
```

Suggerimento: usa i comandi `sed`, `cut` e `paste` per ottenere i risultati desiderati. Nota che la virgola (,) sarà il delimitatore per questo file.

Inizia con i comandi `sed` e `cut`, partendo da ciò che abbiamo imparato dagli esercizi precedenti:

```
$ sed -n '/Main Office'/p mypasswd | cut -d: -f5 | cut -d" " -f1 > firstname
```

Ora abbiamo il file `firstname` con i nomi dei nostri dipendenti.

```
$ sed -n '/Main Office'/p mypasswd | cut -d: -f5 | cut -d" " -f2 | cut -d, -f1 > lastname
```

Ora abbiamo il file `lastname` contenente i cognomi di ogni dipendente.

Successivamente determiniamo in quale reparto lavora ogni dipendente:

```
$ sed -n '/Main Office'/p mypasswd | cut -d: -f5 | cut -d, -f2 > department
```

Prima di lavorare alla soluzione finale, prova i seguenti comandi per vedere che tipo di output generano:

```
$ cat firstname lastname department
$ paste firstname lastname department
```

E ora per la soluzione finale:

```
$ paste firstname lastname department | tr '\t' ,
$ paste firstname lastname department | tr '\t' , > names.csv
```

Qui usiamo il comando `tr` per “translate” `\t`, il separatore di tabulazione, con un `,`. `tr` è molto

utile quando dobbiamo sostituire un carattere con un altro. Assicurati di rivedere le pagine man sia per `tr` che per `paste`. Per esempio, possiamo usare l'opzione `-d` per il delimitatore, per rendere meno complesso il comando precedente:

```
$ paste -d, firstname lastname department
```

Abbiamo usato il comando `paste` già una volta per familiarizzare con esso. Tuttavia avremmo potuto eseguire facilmente tutte le attività in una singola catena di comandi:

```
$ sed -n '/Main Office'/p mypasswd | cut -d: -f5 | cut -d, -f1,2 | tr ' ' , > names.csv
```

- Supponiamo che il foglio di calcolo `names.csv` creato nell'esercizio precedente sia un file importante e vogliamo assicurarci che nessuno lo manometterà dal momento in cui lo inviamo a qualcuno fino a quando il nostro destinatario lo riceverà. Come possiamo assicurare l'integrità di questo file usando `md5sum`?

Se guardi nelle pagine man di `md5sum`, `sha256sum` e `sha512sum` vedrai che iniziano tutte con il seguente testo:

“compute and check XXX message digest”

Dove “XXX” è l'algoritmo che verrà utilizzato per creare questo messaggio *digest*.

Useremo `md5sum` come esempio e successivamente potrai provare con gli altri comandi.

```
$ md5sum names.csv
61f0251fcab61d9575b1d0cbf0195e25 names.csv
```

Ora, per esempio, puoi rendere disponibile il file tramite un servizio ftp sicuro e inviare il *message digest* generato utilizzando un altro mezzo di comunicazione sicuro. Se il file è stato in qualche modo alterato, il *message digest* sarà completamente diverso. Solo per dimostrarlo, modifica `names.csv` e cambia Jones in James come dimostrato qui:

```
$ sed -i.backup s/Jones/James/ names.csv
$ md5sum names.csv
f44a0d68cb480466099021bf6d6d2e65 names.csv
```

Ogni volta che rendi i file disponibili per il download, è sempre buona norma distribuire anche un corrispondente *message digest* in modo che le persone che scaricano il tuo file possano

produrre un nuovo *message digest* e confrontarlo con l'originale. Se navighi su <https://kernel.org> troverai la pagina <https://mirrors.edge.kernel.org/pub/linux/kernel/v5.x/sha256sums.asc> dove puoi ottenere lo sha256sum per tutti i file disponibili per il download.

5. Ti sei ripromesso di leggere cento righe di un testo classico al giorno e hai deciso di iniziare con *Mariner and Mystic* di Herman Melville. Elabora un comando usando `split` che separerà questo libro in sezioni di 100 righe ciascuna. Per ottenere il libro in formato testo normale, cercalo su <https://www.gutenberg.org>.

Per prima cosa otterremo l'intero libro dal sito di Project Gutenberg, dove è possibile ottenere questo e altri libri di pubblico dominio.

```
$ wget https://www.gutenberg.org/files/50461/50461-0.txt
```

Potrebbe essere necessario installare `wget` se non è già installato nel tuo sistema. In alternativa, puoi anche usare `curl`. Usa `less` per verificare il libro:

```
$ less 50461-0.txt
```

Ora divideremo il libro in blocchi di 100 righe ciascuno:

```
$ split -l 100 -d 50461-0.txt melville
```

`50461-0.txt` è il file che divideremo. `melville` sarà il prefisso del nome per i file divisi. `-l 100` specifica il numero di righe e l'opzione `-d` dice a `split` di numerare i file attraverso un suffisso. Puoi usare `nl` su uno qualsiasi dei file divisi (probabilmente non sull'ultimo) e confermare che ognuno di essi abbia 100 righe.

6. Usando `ls -l` nella directory `/etc`, che tipo di elenco ottieni? Usando il comando `cut` sull'output del comando `ls` come visualizzeresti solo i nomi dei file? E come il nome del file e del proprietario dei file? Insieme ai comandi `ls -l` e `cut`, usa il comando `tr` per *sostituire* più occorrenze di uno spazio con un singolo spazio per aiutare a formattare l'output generato da un comando `cut`.

Il comando `ls` darà solo i nomi dei file. Possiamo, tuttavia, elaborare l'output di `ls -l` (lunga lista) per estrarre informazioni più specifiche.

```
$ ls -l /etc | tr -s ' ' ,
drwxr-xr-x,3,root,root,4096,out,24,16:58,acpi
```



```
-rw-r--r-- ,1,root,root,3028,dez,17,2018,adduser.conf
-rw-r--r-- ,1,root,root,10,out,2,17:38,adjtime
drwxr-xr-x,2,root,root,12288,out,31,09:40,alternatives
-rw-r--r-- ,1,root,root,401,mai,29,2017,anacrontab
-rw-r--r-- ,1,root,root,433,out,1,2017,apg.conf
drwxr-xr-x,6,root,root,4096,dez,17,2018,apm
drwxr-xr-x,3,root,root,4096,out,24,16:58,apparmor
drwxr-xr-x,9,root,root,4096,nov,6,20:20,apparmor.d
```

L'opzione `-s` indica a `tr` di ridurre gli spazi ripetuti in un singolo spazio. Il comando `tr` funziona per qualsiasi tipo di carattere ripetitivo specificato. Quindi sostituiamo gli spazi con una virgola `,`. In realtà non abbiamo bisogno di sostituire gli spazi nel nostro esempio, quindi ometteremo `,`.

```
$ ls -l /etc | tr -s ' '
drwxr-xr-x 3 root root 4096 out 24 16:58 acpi
-rw-r--r-- 1 root root 3028 dez 17 2018 adduser.conf
-rw-r--r-- 1 root root 10 out 2 17:38 adjtime
drwxr-xr-x 2 root root 12288 out 31 09:40 alternatives
-rw-r--r-- 1 root root 401 mai 29 2017 anacrontab
-rw-r--r-- 1 root root 433 out 1 2017 apg.conf
drwxr-xr-x 6 root root 4096 dez 17 2018 apm
drwxr-xr-x 3 root root 4096 out 24 16:58 apparmor
```

Se voglio solo i nomi dei file, tutto ciò di cui abbiamo bisogno è il nono campo:

```
$ ls -l /etc | tr -s ' ' | cut -d" " -f9
```

Per il nome file e il proprietario di un file avremo bisogno del nono e del terzo campo:

```
$ ls -l /etc | tr -s ' ' | cut -d" " -f9,3
```

E se avessimo solo bisogno dei nomi delle cartelle e del loro proprietario?

```
$ ls -l /etc | grep ^d | tr -s ' ' | cut -d" " -f9,3
```

- Questo esercizio presuppone che tu sia su una macchina reale (non una macchina virtuale). Devi anche avere una chiavetta USB con te. Esamina le pagine del manuale per il comando `tail` e scopri come *seguire* un file mentre il testo viene aggiunto ad esso. Durante il

monitoraggio dell'output di un comando `tail` sul file `/var/log/syslog`, inserire una chiavetta USB. Scrivi il comando completo che utilizzeresti per ottenere *Product*, *Manufacturer* e *total amount of memory* della tua chiavetta USB.

```
$ tail -f /var/log/syslog | grep -i 'product\\:\\|blocks\\|manufacturer'
Nov  8 06:01:35 brod-avell kernel: [124954.369361] usb 1-4.3: Product: Cruzer Blade
Nov  8 06:01:35 brod-avell kernel: [124954.369364] usb 1-4.3: Manufacturer: SanDisk
Nov  8 06:01:37 brod-avell kernel: [124955.419267] sd 2:0:0:0: [sdc] 61056064 512-byte
logical blocks: (31.3 GB/29.1 GiB)
```

Ovviamente questo è un esempio e i risultati possono variare a seconda del produttore della memory stick USB. Notate ora che usiamo il parametro `-i` con il comando `grep` poiché non siamo sicuri che le stringhe, che stiamo cercando, siano in maiuscolo o minuscolo. Abbiamo anche usato `|` come OR logico, quindi cerchiamo le linee che contengono `product` o `blocks` o `manufacturer`.



Linux
Professional
Institute

103.3 Eseguire la gestione di base dei file

Obiettivi LPI di riferimento

LPIC-1 v5, Exam 101, Objective 103.3

Peso

4

Aree di Conoscenza Chiave

- Copiare, spostare e rimuovere file e directory individualmente.
- Copiare più file e directory in modo ricorsivo.
- Rimuovere file e directory in modo ricorsivo.
- Utilizzare caratteri jolly, semplici e avanzati, nei comandi.
- Utilizzare il comando `find` per individuare e agire sui file in base al tipo, alle dimensioni o al tempo.
- Utilizzo di `tar`, `cpio` e `dd`.

Di seguito è riportato un elenco parziale dei file, dei termini e dei comandi utilizzati

- `cp`
- `find`
- `mkdir`
- `mv`
- `ls`
- `rm`
- `rmdir`
- `touch`

- tar
- cpio
- dd
- file
- gzip
- gunzip
- bzip2
- bunzip2
- file globbing



103.3 Lezione 1

Certificazione:	LPIC-1
Versione:	5.0
Argomento:	103 Comandi GNU e Unix
Obiettivo:	103.3 Eseguire la gestione base sui file
Lezione:	1 di 2

Introduzione

Tutto in Linux è un file, quindi sapere come manipolarli è molto importante. In questa lezione, tratteremo le operazioni di base sui file.

In generale, come utente Linux, sarai chiamato a navigare nel file system, copiare file da una posizione a un'altra ed eliminare file. Tratteremo anche i comandi associati alla gestione dei file.

Un file è un'entità che memorizza dati e programmi. Consiste di contenuto e metadati (dimensione del file, proprietario, data di creazione, autorizzazioni). I file sono organizzati in directory. Una directory è un file che memorizza altri file.

I diversi tipi di file includono:

File regolari

che memorizzano dati e programmi.

Directory

che contengono altri file.

File speciali

che vengono utilizzati per l'input e l'output.

Ovviamente esistono altri tipi di file ma esulano dallo scopo di questa lezione. Successivamente discuteremo come identificare questi diversi tipi di file.

Manipolazione dei File

Utilizzo di "ls" per Elencare i File

Il comando `ls` è uno dei più importanti strumenti da riga di comando che dovresti imparare per navigare nel file system.

Nella sua forma base, `ls` elencherà *solo* i nomi di file e directory:

```
$ ls
Desktop Downloads  emp_salary file1  Music  Public Videos
Documents  emp_name  examples.desktop  file2  Pictures  Templates
```

Quando viene utilizzato con `-l`, o anche indicato come "elenco lungo", mostra i permessi di file o directory, proprietario, dimensione, data di ultima modifica, ora e nome:

```
$ ls -l
total 60
drwxr-xr-x 2 frank frank 4096 Apr 1 2018 Desktop
drwxr-xr-x 2 frank frank 4096 Apr 1 2018 Documents
drwxr-xr-x 2 frank frank 4096 Apr 1 2018 Downloads
-rw-r--r-- 1 frank frank 21 Sep 7 12:59 emp_name
-rw-r--r-- 1 frank frank 20 Sep 7 13:03 emp_salary
-rw-r--r-- 1 frank frank 8980 Apr 1 2018 examples.desktop
-rw-r--r-- 1 frank frank 10 Sep 1 2018 file1
-rw-r--r-- 1 frank frank 10 Sep 1 2018 file2
drwxr-xr-x 2 frank frank 4096 Apr 1 2018 Music
drwxr-xr-x 2 frank frank 4096 Apr 1 2018 Pictures
drwxr-xr-x 2 frank frank 4096 Apr 1 2018 Public
drwxr-xr-x 2 frank frank 4096 Apr 1 2018 Templates
drwxr-xr-x 2 frank frank 4096 Apr 1 2018 Videos
```

Il primo carattere nell'output indica il tipo di file:

-
per un file regolare.

d
per una directory.

c
per un file speciale.

Per mostrare le dimensioni dei file in un formato leggibile dall'uomo, aggiungi l'opzione `-lh`:

```
$ ls -lh
total 60K
drwxr-xr-x 2 frank frank 4.0K Apr 1 2018 Desktop
drwxr-xr-x 2 frank frank 4.0K Apr 1 2018 Documents
drwxr-xr-x 2 frank frank 4.0K Apr 1 2018 Downloads
-rw-r--r-- 1 frank frank 21 Sep 7 12:59 emp_name
-rw-r--r-- 1 frank frank 20 Sep 7 13:03 emp_salary
-rw-r--r-- 1 frank frank 8.8K Apr 1 2018 examples.desktop
-rw-r--r-- 1 frank frank 10 Sep 1 2018 file1
-rw-r--r-- 1 frank frank 10 Sep 1 2018 file2
drwxr-xr-x 2 frank frank 4.0K Apr 1 2018 Music
drwxr-xr-x 2 frank frank 4.0K Apr 1 2018 Pictures
drwxr-xr-x 2 frank frank 4.0K Apr 1 2018 Public
drwxr-xr-x 2 frank frank 4.0K Apr 1 2018 Templates
drwxr-xr-x 2 frank frank 4.0K Apr 1 2018 Videos
```

Per elencare tutti i file, inclusi i file nascosti (quelli che iniziano con `.`) usa l'opzione `-a`:

```
$ ls -a
.          .dbus  file1  .profile
..         Desktop file2   Public
.bash_history .dmrc  .gconf .sudo_as_admin_successful
```

I file di configurazione come `.bash_history`, che sono nascosti per impostazione predefinita, sono ora visibili.

In generale, la sintassi del comando `ls` è data da:

```
ls OPTIONS FILE
```

Dove `OPTIONS` è una qualsiasi delle opzioni mostrate in precedenza (per visualizzare tutte le opzioni possibili eseguire `man ls`), e `FILE` è il nome del file o dei dettagli della directory che desideri elencare.

NOTE Quando `FILE` non è specificato, la directory corrente è implicita.

Creazione, Copia, Spostamento ed Eliminazione di file

Creare file attraverso `touch`

Il comando `touch` è il modo più semplice per creare nuovi file vuoti. Puoi anche usarlo per cambiare il *timestamp* (cioè l'ora di modifica) di file e directory esistenti. La sintassi per usare `touch` è:

```
touch OPTIONS FILE_NAME(S)
```

Senza alcuna opzione, `touch` crea nuovi file per qualsiasi nome fornito come argomento, a condizione che i file con tali nomi non esistano già. `touch` può creare un numero qualsiasi di file contemporaneamente:

```
$ touch file1 file2 file3
```

Questo creerà tre nuovi file vuoti denominati `file1`, `file2` e `file3`.

Diverse opzioni di `touch` sono progettate specificamente per consentire all'utente di modificare il timestamp dei file. Per esempio, l'opzione `-a` cambia solo l'ora di accesso, mentre l'opzione `-m` cambia solo l'ora di modifica. L'uso di entrambe le opzioni modifica i tempi di accesso e anche di modifica all'ora corrente:

```
$ touch -am file3
```

Copiare i file con `cp`

Come utente Linux copierai spesso file da una posizione a un'altra. Che si tratti di spostare un file musicale da una directory a un'altra o di un file di sistema, usa `cp` per tutte le attività di copia:

```
$ cp file1 dir2
```

Questo comando può essere letteralmente interpretato come copia `file1` nella directory `dir2`. Il

risultato è la presenza di `file1` all'interno di `dir2`. Affinché questo comando venga eseguito con successo, `file1` dovrebbe essere presente nella directory corrente dell'utente. In caso contrario, il sistema segnala un errore con il messaggio `No such file or directory`.

```
$ cp dir1/file1 dir2
```

In questo caso osserva che il percorso di `file1` è più esplicito. Il percorso di origine può essere espresso come percorso *relativo* o *percorso assoluto*. I percorsi relativi vengono forniti in riferimento a una directory specifica, mentre i percorsi assoluti non vengono forniti con un riferimento. Di seguito chiariremo ulteriormente questa nozione.

Per il momento osserva che questo comando copia `file1` nella directory `dir2`. Il percorso di `file1` è fornito con maggiori dettagli poiché l'utente attualmente non si trova in `dir1`.

```
$ cp /home/frank/Documents/file2 /home/frank/Documents/Backup
```

In questo terzo caso, `file2` che si trova in `/home/frank/Documents` viene copiato nella directory `/home/frank/Documents/Backup`. Il percorso di origine fornito qui è *assoluto*. Nei due esempi precedenti, i percorsi di origine sono *relativi*. Quando un percorso inizia con il carattere `/` è un percorso assoluto, altrimenti è un percorso relativo.

La sintassi generale di `cp` è:

```
cp OPTIONS SOURCE DESTINATION
```

`SOURCE` è il file da copiare e `DESTINATION` la directory in cui verrà copiato il file. `SOURCE` e `DESTINATION` possono essere specificati come percorsi assoluti o relativi.

Spostare File con `mv`

Proprio come `cp` per la copia, Linux fornisce un comando per spostare e rinominare i file. Si chiama `mv`.

L'operazione di spostamento è analoga all'operazione di taglia e incolla che generalmente si esegue tramite un'interfaccia utente grafica (GUI).

Se desideri spostare un file in una nuova posizione, usa `mv` nel modo seguente:

```
mv FILENAME DESTINATION_DIRECTORY
```

Ecco un esempio:

```
$ mv myfile.txt /home/frank/Documents
```

Il risultato è che `myfile.txt` viene spostato nella destinazione `/home/frank/Documents`.

Per rinominare un file, `mv` viene utilizzato nel modo seguente:

```
$ mv old_file_name new_file_name
```

Questo cambia il nome del file da `old_file_name` a `new_file_name`.

Per impostazione predefinita, `mv` non cercherà la tua conferma se desideri sovrascrivere (rinominare) un file esistente. Tuttavia, puoi consentire al sistema di richiedere conferma, utilizzando l'opzione `-i`:

```
$ mv -i old_file_name new_file_name
mv: overwrite 'new_file_name'?
```

Questo comando chiederebbe l'autorizzazione dell'utente prima di sovrascrivere `old_file_name` in `new_file_name`.

Al contrario, utilizzando `-f`:

```
$ mv -f old_file_name new_file_name
```

sovrascriverebbe forzatamente il file, senza chiedere alcun permesso.

Rimouvere File con `rm`

`rm` viene utilizzato per eliminare i file. Pensala come una forma abbreviata della parola “rimuovi”. Notare che l'azione di rimozione di un file è solitamente irreversibile, quindi questo comando deve essere utilizzato con cautela.

```
$ rm file1
```

Il comando dovrebbe cancellare `file1`.

```
$ rm -i file1
rm: remove regular file 'file1'?
```

Questo comando richiederebbe all'utente la conferma prima di eliminare `file1`. Ricorda, abbiamo visto sopra l'opzione `-i` quando si usa `mv`.

```
$ rm -f file1
```

Questo comando elimina forzatamente `file1` senza richiedere la tua conferma.

È possibile eliminare più file contemporaneamente:

```
$ rm file1 file2 file3
```

In questo esempio, `file1`, `file2` e `file3` vengono eliminati simultaneamente.

La sintassi per `rm` è generalmente data da:

```
rm OPTIONS FILE
```

Creare e Rimuovere Directory

Creare Directory con `mkdir`

La creazione di directory è fondamentale per organizzare file e cartelle. I file possono essere raggruppati insieme in modo logico mantenendoli all'interno di una directory. Per creare una directory, usa `mkdir`:

```
mkdir OPTIONS DIRECTORY_NAME
```

dove `DIRECTORY_NAME` è il nome della directory da creare. È possibile creare un numero qualsiasi di directory contemporaneamente:

```
$ mkdir dir1
```

creerà la directory `dir1` nella directory corrente dell'utente.

```
$ mkdir dir1 dir2 dir3
```

Il comando precedente creerà tre directory `dir1`, `dir2` e `dir3` nello stesso tempo.

Per creare una directory insieme alle sue sottodirectory usa l'opzione `-p` ("parents"):

```
$ mkdir -p parents/children
```

Questo comando creerà la struttura delle directory `parents/children`, cioè creerà le directory `parents` e `children`. `children` si troverà all'interno di "parents"

Rimozione di Directory con `rmdir`

`rmdir` cancella una directory *se è vuota*. La sua sintassi è data da:

```
rmdir OPTIONS DIRECTORY
```

dove "DIRECTORY" potrebbe essere un singolo argomento o un elenco di argomenti.

```
$ rmdir dir1
```

Questo comando eliminerà `dir1`.

```
$ rmdir dir1 dir2
```

Questo comando eliminerà simultaneamente `dir1` e `dir2`.

Puoi rimuovere una directory con la sua sottodirectory:

```
$ rmdir -p parents/children
```

Ciò rimuoverà la struttura della directory `parents/children`. Da notare che se una qualsiasi delle directory non è vuota, non verrà eliminata.

Manipolazione Ricorsiva di File e Directory

Per manipolare una directory e il suo contenuto, è necessario utilizzare la *ricorsività*. Ricorsione

significa, eseguire un'azione e ripetere quell'azione lungo tutta la struttura della directory. In Linux, le opzioni `-r` o `-R` o `--recursive` sono generalmente associate alla ricorsione.

Il seguente scenario ti aiuterà a comprendere meglio la ricorsione

Vuoi elencare tutto il contenuto di una directory `students`, che contiene due sottodirectory `level 1` e `level 2` e il file chiamato `frank`. Applicando la ricorsione, il comando `ls` elenca il contenuto di `students`, cioè `level 1`, `level 2` e `frank`, ma non finisce qui. Entrerà nelle sottodirectory `level 1` e `level 2` e ne elencherà il contenuto e così via lungo l'albero delle directory.

Elenco Ricorsivo con `ls -R`

`ls -R` è usato per elencare il contenuto di una directory insieme alle sue sottodirectory e file.

```
$ ls -R mydirectory
mydirectory/:
file1      newdirectory

mydirectory/newdirectory:
```

Nell'elenco sopra, è elencato `mydirectory` compreso tutto il suo contenuto. Puoi osservare che `mydirectory` contiene la sottodirectory `newdirectory` e il file `file1`. `newdirectory` è vuota, ecco perché non viene mostrato alcun contenuto.

In generale, per elencare il contenuto di una directory comprese le sue sottodirectory, utilizzare:

```
ls -R DIRECTORY_NAME
```

L'aggiunta di uno slash a "`DIRECTORY_NAME`" non ha alcun effetto:

```
$ ls -R animal
```

è simile a:

```
$ ls -R animal/
```

Copia Ricursiva con `cp -r`

`cp -r` (o `-R` o `--recursive`) consente di copiare una directory insieme a tutte le sue

sottodirectory e file.

```
$ tree mydir
mydir
|_file1
|_newdir
    |_file2
    |_insidenew
    |_lastdir

3 directories, 2 files
$ mkdir newcopy
$ cp mydir newcopy
cp: omitting directory 'mydir'
$ cp -r mydir newcopy
* tree newcopy
newcopy
|_mydir
    |_file1
    |_newdir
        |_file2
        |_insidenew
        |_lastdir

4 directories, 2 files
```

Nell'elenco sopra, osserviamo che provando a copiare `mydir` in `newcopy`, usando `cp` senza `-r`, il sistema visualizza il messaggio `cp: omitting directory 'mydir'`. Tuttavia, aggiungendo l'opzione `-r`, tutti i contenuti di `mydir`, incluso se stesso, vengono copiati in `newcopy`.

Per copiare directory e sottodirectory utilizzare:

```
cp -r SOURCE DESTINATION
```

Cancellazione Ricorsiva con `rm -r`

`rm -r` rimuoverà una directory e tutto il suo contenuto (sottodirectory e file).

WARNING

Fai molta attenzione con la combinazione di opzioni `-r` o `-rf` quando usata con il comando `rm`. Un comando di rimozione ricorsivo su un'importante

directory di sistema potrebbe rendere il sistema inutilizzabile. Utilizzare il comando di rimozione ricorsivo solo quando si è assolutamente certi che il contenuto di una directory può essere rimosso in sicurezza da un computer.

Nel tentativo di eliminare una directory senza usare `-r`, il sistema segnala un errore:

```
$ rm newcopy/
rm: cannot remove 'newcopy/': Is a directory
$ rm -r newcopy/
```

Devi aggiungere `-r` come nel secondo comando affinché l'eliminazione abbia effetto.

NOTE

Forse ti starai chiedendo perché in questo caso non usiamo `rmdir`. C'è una sottile differenza tra i due comandi. `rmdir` riuscirà a cancellare solo se la directory data è vuota mentre `rm -r` può essere usato indipendentemente dal fatto che questa directory sia vuota o meno.

Aggiungi l'opzione `-i` per chiedere conferma prima che il file venga eliminato:

```
$ rm -ri mydir/
rm: remove directory 'mydir/'?
```

Il sistema chiede prima di provare a cancellare `mydir`.

File Globbing e Wildcards

Il file *globbing* è una funzionalità fornita dalla shell Unix/Linux per rappresentare più nomi di file utilizzando caratteri speciali chiamati *wildcards*. I caratteri jolly sono essenzialmente simboli che possono essere utilizzati per sostituire uno o più caratteri. Consentono, per esempio, di mostrare tutti i file che iniziano con la lettera `A` o tutti i file che terminano con le lettere `.conf`.

I caratteri jolly sono molto utili in quanto possono essere utilizzati con comandi come `cp`, `ls` o `rm`.

Di seguito sono riportati alcuni esempi di file globbing:

`rm *`

Elimina tutti i file nella directory di lavoro corrente.

`ls l?st`

Elenca tutti i file con nomi che iniziano con `l` seguito da un singolo carattere e terminano con

st.

rmdir [a-z] *

Rimuove tutte le directory il cui nome inizia con una lettera minuscola.

Tipi di Wildcard

Ci sono tre caratteri che possono essere usati come caratteri jolly in Linux:

*** (asterisco)**

che rappresenta zero, una o più occorrenze di qualsiasi carattere.

? (punto interrogativo)

che rappresenta una singola occorrenza di qualsiasi carattere.

[] (caratteri tra parentesi quadre)

che rappresenta qualsiasi occorrenza del carattere(i) racchiuso tra parentesi quadre. È possibile utilizzare diversi tipi di caratteri quali numeri, lettere, altri caratteri speciali. Per esempio, l'espressione `[0-9]` corrisponde a tutte le cifre.

L'asterisco

Un asterisco ("`*`") corrisponde a zero, una o più occorrenze di qualsiasi carattere.

Per esempio:

```
$ find /home -name *.png
```

Questo troverà tutti i file che finiscono con `.png` come `photo.png`, `cat.png`, `frank.png`. Il comando `find` verrà esplorato ulteriormente in una lezione successiva.

Similarmente:

```
$ ls lpic-*.txt
```

elenca tutti i file di testo che iniziano con i caratteri `lpic-` seguiti da un numero qualsiasi di caratteri e terminano con `.txt`, come `lpic-1.txt` e `lpic-2.txt`.

Il carattere jolly asterisco può essere utilizzato per manipolare (copiare, eliminare o spostare) tutti i contenuti di una directory:


```
$ cp -r animal/* forest
```

In questo esempio, tutto il contenuto di `animal` viene copiato in `forest`.

In generale per copiare tutto il contenuto di una directory utilizziamo:

```
cp -r SOURCE_PATH/* DEST_PATH
```

dove `SOURCE_PATH` può essere omesso se ci troviamo già nella directory richiesta.

L'asterisco, proprio come qualsiasi altro carattere jolly, potrebbe essere utilizzato più volte nello stesso comando e in qualsiasi posizione:

```
$ rm *ate*
```

Nomi di file con prefisso zero, una o più occorrenze di qualsiasi carattere, seguite dalle lettere `ate` e terminanti con zero, una o più occorrenze di qualsiasi carattere verranno rimosse

Il Punto Interrogativo

Il punto interrogativo (?) corrisponde a una *sola* occorrenza di un carattere.

Considera l'elenco:

```
$ ls
last.txt    lest.txt    list.txt    third.txt   past.txt
```

Per restituire solo i file che iniziano con `l` seguito da un singolo carattere e dai caratteri `st.txt`, utilizziamo il carattere jolly punto interrogativo (?):

```
$ ls l?st.txt
last.txt    lest.txt    list.txt
```

Vengono visualizzati solo i file `last.txt`, `lest.txt` e `list.txt` poiché corrispondono ai criteri specificati.

Similarmente,

```
$ ls ??st.txt
last.txt    lest.txt    list.txt    past.txt
```

file di output che sono preceduti da due caratteri qualsiasi seguiti dal testo `st.txt`.

Caratteri tra Parentesi Quadre

I caratteri jolly tra parentesi corrispondono a qualsiasi occorrenza del carattere(i) racchiuso tra parentesi quadre:

```
$ ls l[ae]st.txt
last.txt    lest.txt
```

Questo comando elenca tutti i file che iniziano con `l` seguito da *qualsiasi* dei caratteri nel set `ae` e finiscono con `st.txt`.

Le parentesi quadre potrebbero anche comprendere intervalli:

```
$ ls l[a-z]st.txt
last.txt    lest.txt    list.txt
```

Questo restituisce tutti i file con nomi che iniziano con `l` seguito da qualsiasi lettera minuscola nell'intervallo da `a` a `z` e finiscono con `st.txt`.

È anche possibile applicare più intervalli tra parentesi quadre:

```
$ ls
student-1A.txt student-2A.txt student-3.txt
$ ls student-[0-9][A-Z].txt
student-1A.txt student-2A.txt
```

L'esecuzione del comando mostra un elenco scolastico di studenti registrati. Per elencare solo gli studenti i cui numeri di registrazione soddisfano i seguenti criteri:

- inizia con `student-`
- seguito da un numero e un carattere maiuscolo
- e termina con `.txt`

Combinazione di Wildcard

Le wildcard (caratteri jolly) possono essere combinate nei seguenti modi:

```
$ ls
last.txt  lest.txt  list.txt  third.txt  past.txt
$ ls [plf]?st*
last.txt  lest.txt  list.txt  past.txt
```

Il primo componente jolly ([plf]) corrisponde a uno qualsiasi dei caratteri p, l o f. Il secondo componente jolly (?) corrisponde a qualsiasi carattere singolo. Il terzo componente jolly (*) corrisponde a zero, una o più occorrenze di qualsiasi carattere.

```
$ ls
file1.txt file.txt file23.txt fom23.txt
$ ls f*[0-9].txt
file1.txt file23.txt fom23.txt
```

Il comando precedente mostra tutti i file che iniziano con la lettera f, seguiti da qualsiasi insieme di lettere, almeno un'occorrenza di una cifra e termina con .txt. Nota che file.txt non viene visualizzato poiché non corrisponde a questo criterio.

Esercizi Guidati

1. Considera l'elenco seguente:

```
$ ls -lh
total 60K
drwxr-xr-x 2 frank frank 4.0K Apr 1 2018 Desktop
drwxr-xr-x 2 frank frank 4.0K Apr 1 2018 Documents
drwxr-xr-x 2 frank frank 4.0K Apr 1 2018 Downloads
-rw-r--r-- 1 frank frank 21 Sep 7 12:59 emp_name
-rw-r--r-- 1 frank frank 20 Sep 7 13:03 emp_salary
-rw-r--r-- 1 frank frank 8.8K Apr 1 2018 examples.desktop
-rw-r--r-- 1 frank frank 10 Sep 1 2018 file1
-rw-r--r-- 1 frank frank 10 Sep 1 2018 file2
drwxr-xr-x 2 frank frank 4.0K Apr 1 2018 Music
drwxr-xr-x 2 frank frank 4.0K Apr 1 2018 Pictures
drwxr-xr-x 2 frank frank 4.0K Apr 1 2018 Public
drwxr-xr-x 2 frank frank 4.0K Apr 1 2018 Templates
drwxr-xr-x 2 frank frank 4.0K Apr 1 2018 Videos
```

- Cosa rappresenta il carattere d nell'output?
- Perché le dimensioni sono fornite in un formato leggibile dall'uomo?
- Quale sarebbe la differenza nell'output se ls fosse usato senza argomenti?

2. Considera il comando seguente:

```
$ cp /home/frank/emp_name /home/frank/backup
```

- Cosa succederebbe al file emp_name se questo comando fosse eseguito con successo?
- Se emp_name fosse una directory quale opzione dovrebbe essere aggiunta a cp per eseguire il comando?

- Se `cp` venisse cambiato in `mv`, quali risultati ti aspetteresti?

3. Considera l'elenco:

```
$ ls  
file1.txt file2.txt file3.txt file4.txt
```

Quale carattere jolly aiuterebbe a cancellare tutto il contenuto di questa directory?

4. Sulla base dell'elenco precedente, quali file verrebbero visualizzati dal seguente comando?

```
$ ls file*.txt
```

5. Completa il comando aggiungendo le cifre e i caratteri appropriati nelle parentesi quadre per elencare tutto il contenuto precedentemente indicato:

```
$ ls file[ ].txt
```

Esercizi Esplorativi

1. Nella tua directory home, crea i file chiamati `dog` e `cat`.
2. Sempre nella tua directory home, crea la directory chiamata `animal`. Sposta `dog` e `cat` all'interno di `animal`.
3. Vai alla cartella `Documents` che si trova nella tua directory home e all'interno, crea la directory `backup`.
4. Copia `animal` e il suo contenuto all'interno di `backup`.
5. Rinomina `animal` all'interno di `backup` in `animal.bkup`
6. La directory `/home/lpi/databases` contiene molti file tra i quali: `db-1.tar.gz`, `db-2.tar.gz` e `db-3.tar.gz`. Quale singolo comando puoi utilizzare per elencare solo i file sopra menzionati?

7. Considera il seguente elenco:

```
$ ls
cne1222223.pdf cne12349.txt cne1234.pdf
```

Attraverso l'uso di un singolo carattere di globbing, quale comando rimuoverebbe solo i file pdf?

Sommario

In questa lezione abbiamo imparato a visualizzare cosa c'è all'interno di una directory con il comando `ls`, a copiare file e cartelle (`cp`) e a spostarli (`mv`). Abbiamo anche esaminato come creare una nuova directory con il comando `mkdir`. Sono stati discussi anche i comandi per rimuovere file (`rm`) e cartelle (`rmdir`).

In questa lezione hai anche imparato a conoscere il file globbing e i caratteri jolly. Il globbing dei file viene utilizzato per rappresentare più nomi di file utilizzando caratteri speciali chiamati caratteri jolly. I caratteri jolly di base e il loro significato:

? (punto interrogativo)

rappresenta una singola occorrenza di un carattere.

[] (parentesi quadre)

rappresenta qualsiasi occorrenza del carattere(i) racchiuso tra parentesi quadre.

*** (asterisco)**

rappresenta zero, una o più occorrenze di qualsiasi carattere.

È possibile combinare tutti questi caratteri jolly nella stessa istruzione.

Risposte agli Esercizi Guidati

1. Considera l'elenco seguente:

```
$ ls -lh
total 60K
drwxr-xr-x 2 frank frank 4.0K Apr 1 2018 Desktop
drwxr-xr-x 2 frank frank 4.0K Apr 1 2018 Documents
drwxr-xr-x 2 frank frank 4.0K Apr 1 2018 Downloads
-rw-r--r-- 1 frank frank 21 Sep 7 12:59 emp_name
-rw-r--r-- 1 frank frank 20 Sep 7 13:03 emp_salary
-rw-r--r-- 1 frank frank 8.8K Apr 1 2018 examples.desktop
-rw-r--r-- 1 frank frank 10 Sep 1 2018 file1
-rw-r--r-- 1 frank frank 10 Sep 1 2018 file2
drwxr-xr-x 2 frank frank 4.0K Apr 1 2018 Music
drwxr-xr-x 2 frank frank 4.0K Apr 1 2018 Pictures
drwxr-xr-x 2 frank frank 4.0K Apr 1 2018 Public
drwxr-xr-x 2 frank frank 4.0K Apr 1 2018 Templates
drwxr-xr-x 2 frank frank 4.0K Apr 1 2018 Videos
```

- Cosa rappresenta il carattere `d` nell'output?

`d` è il carattere che identifica una directory.

- Perché le dimensioni sono fornite in un formato leggibile dall'uomo?

Per l'utilizzo dell'opzione `-h`.

- Quale sarebbe la differenza nell'output se `ls` fosse usato senza argomenti?

Sarebbero visualizzati solo i nomi delle directory e dei file.

2. Considera il comando seguente:

```
$ cp /home/frank/emp_name /home/frank/backup
```

- Cosa succederebbe al file `emp_name` se questo comando fosse eseguito con successo?

`emp_name` verrebbe copiato all'interno di `backup`.

- Se `emp_name` fosse una directory quale opzione dovrebbe essere aggiunta a `cp` per eseguire il comando?

-r

- Se `cp` venisse cambiato in `mv`, quali risultati ti aspetteresti?

`emp_name` verrebbe copiato all'interno di `backup`. Non sarebbe più presente nella home directory dell'utente `frank`.

3. Considera l'elenco:

```
$ ls  
file1.txt file2.txt file3.txt file4.txt
```

Quale carattere jolly aiuterebbe a cancellare tutto il contenuto di questa directory?

L'asterisco `*`.

4. Sulla base dell'elenco precedente, quali file verrebbero visualizzati dal seguente comando?

```
$ ls file*.txt
```

Tutti, poiché il carattere asterisco rappresenta un numero qualsiasi di caratteri.

5. Completa il comando aggiungendo le cifre e i caratteri appropriati nelle parentesi quadre per elencare tutto il contenuto precedentemente indicato:

```
$ ls file[ ].txt
```

`file[0-9].txt`

Risposte agli Esercizi Esplorativi

1. Nella tua directory home, crea i file chiamati `dog` e `cat`.

```
$ touch dog cat
```

2. Sempre nella tua directory home, crea la directory chiamata `animal`. Sposta `dog` e `cat` all'interno di `animal`.

```
$ mkdir animal  
$ mv dog cat -t animal/
```

3. Vai alla cartella `Documents` che si trova nella tua directory home e all'interno, crea la directory `backup`.

```
$ cd ~/Documents  
$ mkdir backup
```

4. Copia `animal` e il suo contenuto all'interno di `backup`.

```
$ cp -r animal ~/Documents/backup
```

5. Rinomina `animal` all'interno di `backup` in `animal.bkup`

```
$ mv animal/ animal.bkup
```

6. La directory `/home/lpi/databases` contiene molti file tra i quali: `db-1.tar.gz`, `db-2.tar.gz` e `db-3.tar.gz`. Quale singolo comando puoi utilizzare per elencare solo i file sopra menzionati?

```
$ ls db-[1-3].tar.gz
```

7. Considera il seguente elenco:

```
$ ls  
cne1222223.pdf cne12349.txt cne1234.pdf
```

Attraverso l'uso di un singolo carattere di globbing, quale comando rimuoverebbe solo i file pdf?

```
$ rm *.pdf
```



103.3 Lezione 2

Certificazione:	LPIC-1
Versione:	5.0
Argomento:	103 Comandi GNU e Unix
Obiettivo:	103.3 Eseguire la gestione base sui file
Lezione:	2 di 2

Introduzione

Come Ricercare File

Man mano che si utilizza la macchina, i file crescono progressivamente in numero e dimensione. A volte diventa difficile individuare un particolare file. Fortunatamente Linux fornisce `find` per cercare e individuare rapidamente i file. `find` utilizza la seguente sintassi:

```
find STARTING_PATH OPTIONS EXPRESSION
```

STARTING_PATH

definisce la directory in cui inizia la ricerca.

OPTIONS

controlla il comportamento e aggiunge criteri specifici per ottimizzare il processo di ricerca.

EXPRESSION

definisce la query di ricerca.

```
$ find . -name "myfile.txt"
./myfile.txt
```

Il percorso iniziale in questo caso è la directory corrente. L'opzione `-name` specifica che la ricerca è basata sul nome del file. `myfile.txt` è il nome del file da cercare. Quando si utilizza il globbing dei file, assicurarsi di includere l'espressione tra virgolette:

```
$ find /home/frank -name "*.png"
/home/frank/Pictures/logo.png
/home/frank/screenshot.png
```

Questo comando trova tutti i file che terminano con `.png` a partire dalla directory `/home/frank/`. L'uso dell'asterisco (*) è stato trattato in una lezione precedente

Utilizzo dei Criteri per Velocizzare la Ricerca

Usa `find` per individuare i file in base a *type*, *size* o *time*. Specificando una o più opzioni, i risultati desiderati si ottengono in minor tempo.

Le opzioni per la ricerca di file in base al tipo includono:

-type f

ricerca di file.

-type d

ricerca di directory.

-type l

ricerca di link simbolici.

```
$ find . -type d -name "example"
```

Questo comando trova tutte le directory, nella directory corrente e sottostanti, che hanno il nome `example`.

Altri criteri che potrebbero essere usati con `find` includono:

-name

esegue una ricerca basata sul nome dato.

-iname

effettua la ricerca in base al nome non considerando maiuscole e minuscole (myFile diventa uguale a MYFILE).

-not

restituisce quei risultati che *non* corrispondono allo scenario di test.

-maxdepth N

ricerca nella directory corrente così come nelle sottodirectory a N livelli sottostanti.

Individuazione dei File in Base all'Ora di Modifica

find permette anche di filtrare una gerarchia di directory in base a quando il file è stato modificato:

```
$ sudo find / -name "*.conf" -mtime 7
/etc/logrotate.conf
```

Questo comando cercherà tutti i file nell'intero file system (il percorso iniziale è la directory principale, cioè /) che terminano con i caratteri .conf e sono stati modificati negli ultimi sette giorni. Questo comando richiede privilegi elevati per accedere a tutte le directory a partire dalla base della struttura delle directory del sistema, da qui l'uso di sudo. L'argomento passato a mtime rappresenta il *numero di giorni* dall'ultima modifica del file.

Individuazione dei File per Dimensione

find può anche individuare i file per *dimensione*. Per esempio, cercando file più grandi di 2G in /var:

```
$ sudo find /var -size +2G
/var/lib/libvirt/images/debian10.qcow2
/var/lib/libvirt/images/rhel8.qcow2
```

L'opzione -size mostra i file di dimensioni corrispondenti all'argomento passato. Alcuni argomenti di esempio includono:

-size 100c

file che sono esattamente di 100 byte.

-size +100k

file più grandi di 100 kilobyte.

-size -20M

file di dimensioni inferiori a 20 megabyte.

-size + 2G

file più grandi di 2 gigabyte.

NOTE Per trovare file vuoti possiamo usare: `find . -size 0c` o `find . -empty`.

Agire sul Set di Risultati

Una volta terminata la ricerca, è possibile eseguire un'azione sull'insieme risultante usando `-exec`:

```
$ find . -name "*.conf" -exec chmod 644 '{}' \;
```

Questo comando filtra ogni oggetto nella directory corrente (.) e sottostanti per i nomi di file che terminano con `.conf`; successivamente esegue `chmod 644` per modificare i permessi degli stessi file.

Per ora, non preoccuparti del significato di `'{}' \;` perchè verrà discusso più avanti.

Usare `grep` per filtrare i file in base al contenuto

`grep` viene utilizzato per cercare una o più occorrenze di una parola chiave.

Considera una situazione in cui dobbiamo trovare file in base al contenuto:

```
$ find . -type f -exec grep "lpi" '{}' \; -print
./bash_history
Alpine/M
helping/M
```

Questo comando cercherà ogni oggetto nella gerarchia di directory corrente (.) che è un file (`-type f`) e quindi esegue il comando `grep "lpi"` per ogni file che soddisfa le condizioni. I file che soddisfano queste condizioni vengono visualizzati sullo schermo (`-print`). Le parentesi graffe ({}) sono un segnaposto per i risultati della corrispondenza di `find`. I caratteri {} sono racchiusi tra virgolette singole (') per evitare di passare file greppati con nomi contenenti caratteri

speciali. Il comando `-exec` termina con un punto e virgola (;), che dovrebbe essere preceduto dal carattere di escape (\) per evitare l'interpretazione da parte della shell.

L'aggiunta dell'opzione `-delete` alla fine di un'espressione elimina tutti i file che corrispondono: questa opzione dovrebbe essere utilizzata quando si è certi che i risultati corrispondono solo ai file che si desidera eliminare.

Nell'esempio seguente, `find` individua tutti i file nella gerarchia a partire dalla directory corrente quindi elimina tutti i file che terminano con i caratteri `.bak`:

```
$ find . -name "*.bak" -delete
```

Archiviazione di File

Il Comando `tar` (Archiviazione e Compressione)

Il comando `tar`, abbreviazione di “tape archive(r)”, è usato per creare archivi tar convertendo un gruppo di file in un archivio. Gli archivi vengono creati in modo da spostare o eseguire il backup facilmente di un gruppo di file. Pensa a `tar` come a uno strumento che crea un collante su cui i file possono essere allegati, raggruppati e spostati facilmente.

`tar` ha anche la capacità di estrarre archivi tar, visualizzare un elenco dei file inclusi nell'archivio e aggiungere file ulteriori a un archivio esistente.

La sintassi del comando `tar` è la seguente:

```
tar [OPERATION_AND_OPTIONS] [ARCHIVE_NAME] [FILE_NAME(S)]
```

OPERATION

È consentito e richiesto una sola operazione alla volta. Le operazioni più utilizzate sono:

--create (-c)

Crea un nuovo archivio tar.

--extract (-x)

Estrae l'intero archivio o uno o più file da un archivio.

--list (-t)

Visualizza un elenco dei file inclusi nell'archivio.

OPTIONS

Le opzioni utilizzate più di frequente sono:

--verbose (-v)

Mostra i file elaborati dal comando `tar`.

--file=nome-archivio (-f nome-archivio)

Specifica il nome del file di archivio.

ARCHIVE_NAME

Il nome dell'archivio.

FILE_NAME(S)

Un elenco separato da spazi di nomi di file da estrarre. In caso contrario, viene estratto l'intero archivio.

Creazione di un Archivio

Supponiamo di avere una directory chiamata `stuff` nella directory corrente e vogliamo salvarla in un file chiamato `archive.tar`. Dovremmo eseguire il seguente comando:

```
$ tar -cvf archive.tar stuff
stuff/
stuff/service.conf
```

Ecco cosa significano effettivamente queste opzioni:

-c

Crea un archivio.

-v

Mostra l'avanzamento nel terminale durante la creazione dell'archivio, noto anche come modalità "verbosa". La `-v` è sempre opzionale in questi comandi, ma utile.

-f

Permette di specificare il nome del file dell'archivio.

In generale per archiviare una singola directory o un singolo file su Linux, utilizziamo:

```
tar -cvf NAME-OF-ARCHIVE.tar /PATH/TO/DIRECTORY-OR-FILE
```

NOTE

`tar` funziona in modo ricorsivo. Eseguirà l'azione richiesta su ogni directory successiva all'interno della directory specificata.

Per archiviare più directory contemporaneamente, elenchiamo tutte le directory delimitandole da uno spazio nella sezione `/PATH/TO/DIRECTORY-OR-FILE`:

```
$ tar -cvf archive.tar stuff1 stuff2
```

Questo produrrebbe un archivio di `stuff1` e `stuff2` in `archive.tar`

Estrarre un Archivio

Possiamo estrarre un archivio usando `tar`:

```
$ tar -xvf archive.tar
stuff/
stuff/service.conf
```

Questo estrarrà il contenuto di `archive.tar` nella directory corrente.

Questo comando è lo stesso del comando di creazione dell'archivio usato sopra, ad eccezione dell'opzione `-x` che sostituisce l'opzione `-c`.

Per estrarre il contenuto dell'archivio in una directory specifica usiamo `-C`:

```
$ tar -xvf archive.tar -C /tmp
```

Questo estrarrà il contenuto di `archive.tar` nella directory `/tmp`.

```
$ ls /tmp
stuff
```

Comprimere con `tar`

Il comando GNU `tar` incluso con le distribuzioni Linux può creare un archivio `.tar` e comprimerlo con la compressione `gzip` o `bzip2` in un unico comando:

```
$ tar -czvf name-of-archive.tar.gz stuff
```

Questo comando creerebbe un file compresso usando l'algoritmo `gzip` (`-z`).

Sebbene la compressione `gzip` sia usata più frequentemente per creare file `.tar.gz` o `.tgz`, `tar` supporta anche la compressione `bzip2`. Questo permette la creazione di file compressi `bzip2`, spesso chiamati file `.tar.bz2`, `.tar.bz` o `tbz`.

Per farlo, sostituiamo `-z` per `gzip` con `-j` per `bzip2`:

```
$ tar -cjvf name-of-archive.tar.bz stuff
```

Per decomprimere il file, sostituiamo `-c` con `-x`, dove `x` sta per “extract”:

```
$ tar -xvzf archive.tar.gz
```

`gzip` è più veloce, ma generalmente comprime un po' meno, quindi si ottiene un file leggermente più grande. `bzip2` è più lento, ma comprime un po' di più. In generale, però, `gzip` e `bzip2` sono praticamente la stessa cosa ed entrambi funzioneranno in modo simile.

In alternativa possiamo applicare la compressione `gzip` o `bzip2` usando il comando `gzip` per le compressioni `gzip` e il comando `bzip2` per le compressioni `bzip`. Per esempio, per applicare la compressione `gzip`, usa:

```
gzip FILE-TO-COMPRESS
```

`gzip`

crea il file compresso con lo stesso nome ma con un finale `.gz`.

`gzip`

rimuove i file originali dopo aver creato il file compresso.

Il comando `bzip2` funziona in modo simile.

Per decomprimere i file usiamo `gunzip` o `bunzip2` a seconda dell'algoritmo usato per comprimere un file.

Il Comando `cpio`

`cpio` sta per “copy in, copy out”. Viene utilizzato per elaborare file di archivio come file `\* .cpio` o `\* .tar`.

`cpio` esegue le seguenti operazioni:

- Copiare file in un archivio.
- Estrarre file da un archivio.

Prende l'elenco dei file dallo standard input (principalmente l'output da `ls`).

Per creare un archivio `cpio`, usiamo:

```
$ ls | cpio -o > archive.cpio
```

L'opzione `-o` dice a `cpio` di creare un output. In questo caso, il file di output creato è `archive.cpio`. Il comando `ls` elenca i contenuti della directory corrente che devono essere archiviati.

Per estrarre l'archivio utilizziamo:

```
$ cpio -id < archive.cpio
```

L'opzione `-i` viene utilizzata per eseguire l'estrazione. L'opzione `-d` creerà cartelle di destinazione se necessario. Il carattere `<` rappresenta lo standard input. Il file di input da estrarre è `archive.cpio`.

Il Comando `dd`

`dd` copia i dati da una posizione all'altra. La sintassi della riga di comando di `dd` è diversa da molti altri programmi Unix, utilizza la sintassi `option=value` per le sue opzioni della riga di comando piuttosto che i formati standard GNU `-option value` o `--option=value`:

```
$ dd if=oldfile of=newfile
```

Questo comando copia il contenuto di `oldfile` in `newfile`, dove `if=` è il file di input e `of=` si riferisce al file di output.

NOTE

Il comando `dd` tipicamente non mostrerà nulla sullo schermo fino a quando il comando non sarà terminato. Fornendo l'opzione `status=progress`, la console mostrerà la quantità di lavoro svolto dal comando. Per esempio: `dd status=progress if=oldfile of=newfile`.

`dd` è anche usato per cambiare i dati in maiuscolo/minuscolo o scrivere direttamente su dispositivi

a blocchi come `/dev/sdb`:

```
$ dd if=oldfile of=newfile conv=ucase
```

Questo copierà tutto il contenuto di `oldfile` in `newfile` e renderà tutto il testo in maiuscolo.

Il comando seguente eseguirà il backup dell'intero disco rigido situato in `/dev/sda` in un file chiamato `backup.dd`:

```
$ dd if=/dev/sda of=backup.dd bs=4096
```

Esercizi Guidati

1. Considera il seguente elenco:

```
$ find /home/frank/Documents/ -type d
/home/frank/Documents/
/home/frank/Documents/animal
/home/frank/Documents/animal/domestic
/home/frank/Documents/animal/wild
```

- Che tipo di file produrrebbe questo comando?

- In quale directory inizia la ricerca?

2. Un utente desidera comprimere la sua cartella di backup. Lui usa il seguente comando:

```
$ tar cvf /home/frank/backup.tar.gz /home/frank/dir1
```

Quale opzione manca per comprimere il backup usando l'algoritmo `gzip`?

Esercizi Esplorativi

1. In qualità di amministratore di sistema, è necessario eseguire controlli regolari per rimuovere file voluminosi. Questi file voluminosi si trovano in `/var` e terminano con un'estensione `.backup`.

- Scrivi il comando, usando `find`, per individuare questi file:

- Un'analisi delle dimensioni di questi file rivela che vanno da `100M` a `1000M`. Completa il comando precedente con queste nuove informazioni, in modo da poter individuare quei file di backup che vanno da `100M` a `1000M`:

- Infine, completa questo comando, con l'azione di eliminazione in modo che questi file vengano rimossi:

2. Nella directory `/var`, esistono quattro file di backup:

```
db-jan-2018.backup  
db-feb-2018.backup  
db-march-2018.backup  
db-apr-2018.backup
```

- Utilizzando `tar`, specificare il comando che creerebbe un file di archivio con il nome `db-first-quarter-2018.backup.tar`:

- Usando `tar`, specifica il comando che creerebbe l'archivio e comprimilo usando `gzip`. Tieni presente che il nome del file risultante dovrebbe terminare con `.gz`:

Sommario

In questa lezione abbiamo imparato:

- Come trovare file con `find`.
- Come aggiungere criteri di ricerca in base all'ora, al tipo di file o alla dimensione fornendo un argomento a `find`.
- Come agire su un set restituito.
- Come archiviare, comprimere e decomprimere file usando `tar`.
- Elaborazione di archivi con `cpio`.
- Copia di file con `dd`.

Risposte agli Esercizi Guidati

1. Considera il seguente elenco::

```
$ find /home/frank/Documents/ -type d
/home/frank/Documents/
/home/frank/Documents/animal
/home/frank/Documents/animal/domestic
/home/frank/Documents/animal/wild
```

- Che tipo di file produce questo comando?

Directories.

- In quale directory inizia la ricerca?

/home/frank/Documents

2. Un utente desidera comprimere la sua cartella di backup. Lui usa il seguente comando:

```
$ tar cvf /home/frank/backup.tar.gz /home/frank/dir1
```

Quale opzione manca per comprimere il backup usando l'algoritmo gzip?

L'opzione -z.

Risposte agli Esercizi Esplorativi

1. In qualità di amministratore di sistema, è necessario eseguire controlli regolari per rimuovere file voluminosi. Questi file voluminosi si trovano in `/var` e terminano con un'estensione `.backup`.

- Scrivi il comando, usando `find`, per individuare questi file:

```
$ find /var -name *.backup
```

- Un'analisi delle dimensioni di questi file rivela che vanno da `100M` a `1G`. Completa il comando precedente con queste nuove informazioni, in modo da poter individuare quei file di backup che vanno da `100M` a `1G`:

```
$ find /var -name *.backup -size +100M -size -1G
```

- Infine, completa questo comando, con l'azione di eliminazione in modo che questi file vengano rimossi:

```
$ find /var -name *.backup -size +100M -size -1G -delete
```

2. Nella directory `/var`, esistono quattro file di backup:

```
db-jan-2018.backup  
db-feb-2018.backup  
db-march-2018.backup  
db-apr-2018.backup
```

- Utilizzando `tar`, specificare il comando che creerebbe un file di archivio con il nome `db-first-quarter-2018.backup.tar`:

```
$ tar -cvf db-first-quarter-2018.backup.tar db-jan-2018.backup db-feb-2018.backup db-march-2018.backup db-apr-2018.backup
```

- Usando `tar`, specifica il comando che creerebbe l'archivio e comprimilo usando `gzip`. Tieni presente che il nome del file risultante dovrebbe terminare con `.gz`:

```
$ tar -zcvf db-first-quarter-2018.backup.tar.gz db-jan-2018.backup db-feb-2018.backup
```

```
db-march-2018.backup db-apr-2018.backup
```



103.4 Utilizzare flussi, pipe e reindirizzamenti

Obiettivi LPI di riferimento

LPIC-1 v5, Exam 101, Objective 103.4

Peso

4

Aree di Conoscenza Chiave

- Reindirizzamento di standard input, standard output e standard error.
- Collegare l'output di un comando all'input di un altro comando.
- Usare l'output di un comando come argomento per un altro comando.
- Inviare l'output sia allo stdout che a un file.

Di seguito è riportato un elenco parziale dei file, dei termini e dei comandi utilizzati

- tee
- xargs



Linux
Professional
Institute

103.4 Lezione 1

Certificazione:	LPIC-1 (101)
Versione:	5.0
Argomento:	103 Comandi GNU e Unix
Obiettivo:	103.4 Utilizzare stream, pipe e reindirizzamenti
Lezione:	1 di 2

Introduzione

Tutti i programmi per computer seguono lo stesso principio generale: i dati ricevuti da qualche fonte vengono trasformati per generare un risultato intelligibile. Nel contesto della shell di Linux, l'origine dati può essere un file locale, un file remoto, un dispositivo (come una tastiera), e altro ancora. L'output del programma viene solitamente visualizzato su uno schermo, ma è anche comune memorizzare i dati di output in un filesystem locale, inviarlo a un dispositivo remoto, riprodurlo tramite altoparlanti audio, ecc.

I sistemi operativi ispirati a Unix, come Linux, offrono una grande varietà di metodi di input/output. In particolare il metodo dei *file descriptors* consente di associare dinamicamente numeri interi ai canali dati, in modo che un processo possa riferirli come propri flussi di dati di input/output.

I processi Linux standard hanno tre canali di comunicazione aperti per impostazione predefinita: il canale *standard input* (il più delle volte chiamato semplicemente *stdin*), il canale *standard output* (*stdout*) e il canale *standard error* (*stderr*). I descrittori di file numerici assegnati a questi canali sono `0` per *stdin*, `1` per *stdout* e `2` per *stderr*. I canali di comunicazione sono anche accessibili tramite i dispositivi speciali `/dev/stdin`, `/dev/stdout` e `/dev/stderr`.

Questi tre canali di comunicazione standard consentono ai programmatori di scrivere codice che legge e scrive dati senza preoccuparsi del tipo di supporto da cui provengono o a cui vanno. Per esempio, se un programma ha bisogno di un insieme di dati come input, può semplicemente chiedere i dati allo standard input e qualsiasi cosa venga utilizzata come standard input fornirà quei dati. Allo stesso modo, il metodo più semplice che un programma possa utilizzare per visualizzare il proprio output è scriverlo nell'output standard. In una sessione di shell standard, la tastiera è definita come *stdin* e lo schermo del monitor è definito come *stdout* e *stderr*.

La shell Bash ha la capacità di riassegnare i canali di comunicazione durante il caricamento di un programma. Consente, per esempio, di sovrascrivere lo schermo come output standard e di utilizzare un file nel filesystem locale come *stdout*.

Reindirizzamenti

La riassegnazione del descrittore di file di un canale nell'ambiente della shell è chiamata *redirect*. Un reindirizzamento è definito da un carattere speciale all'interno della riga di comando. Per esempio, per reindirizzare l'output standard di un processo su un file, il simbolo *maggiore di* `>` è posizionato alla fine del comando e seguito dal percorso del file che riceverà l'output reindirizzato:

```
$ cat /proc/cpuinfo >/tmp/cpu.txt
```

Per impostazione predefinita, viene reindirizzato solo il contenuto che arriva a *stdout*. Ciò accade perché il valore numerico del descrittore di file deve essere specificato appena prima del simbolo *maggiore di* e, se non specificato, Bash reindirizza l'output standard. Pertanto, usare `>` equivale a usare `1>` (il valore del descrittore di file di *stdout* è 1).

Per catturare il contenuto di *stderr*, si dovrebbe invece usare il reindirizzamento `2>`. La maggior parte dei programmi da riga di comando invia informazioni di debug e messaggi di errore al canale di errore standard. È possibile, per esempio, catturare il messaggio di errore innescato da un tentativo di leggere un file inesistente:

```
$ cat /proc/cpu_info 2>/tmp/error.txt
$ cat /tmp/error.txt
cat: /proc/cpu_info: No such file or directory
```

Sia *stdout* che *stderr* vengono reindirizzati alla stessa destinazione con `&>` o `>&`. È importante non inserire spazi accanto alla *e commerciale*, altrimenti Bash lo prenderà come istruzione per eseguire il processo in background e non per eseguire il reindirizzamento.

La destinazione deve essere un percorso a un file scrivibile, come `/tmp/cpu.txt`, o un descrittore di file scrivibile. La destinazione di un descrittore di file è rappresentata da una *e commerciale* seguita dal valore numerico del descrittore di file. Per esempio, `1>&2` reindirizza *stdout* a *stderr*. Per fare l'opposto, da *stderr* a *stdout*, dovrebbero essere usati `2>&1`.

Sebbene non sia molto utile, dato che esiste un modo più breve per eseguire la stessa operazione, è possibile reindirizzare *stderr* a *stdout* e quindi reindirizzarlo a un file. Per esempio, un redirect per scrivere sia *stderr* che *stdout* in un file chiamato `log.txt` può essere scritto come `>log.txt 2>&1`. Tuttavia, il motivo principale per reindirizzare *stderr* a *stdout* è consentire l'analisi dei messaggi di debug e di errore. È possibile reindirizzare lo standard output di un programma allo standard input di un altro programma, ma *non* è possibile reindirizzare direttamente lo standard error allo standard input di un altro programma. Pertanto, i messaggi del programma inviati a *stderr* devono prima essere reindirizzati a *stdout* per essere letti dallo *stdin* di un altro programma.

Per scartare semplicemente l'output di un comando, il suo contenuto può essere reindirizzato al file speciale `/dev/null`. Per esempio, `>log.txt 2>/dev/null` salva il contenuto di *stdout* nel file `log.txt` e scarta lo *stderr*. Il file `/dev/null` è scrivibile da qualsiasi utente ma nessun dato può essere recuperato da esso, poiché non è memorizzato da nessuna parte.

Viene visualizzato un messaggio di errore se la destinazione specificata non è scrivibile (se il percorso punta a una directory o a un file di sola lettura) e non viene apportata alcuna modifica alla destinazione. Tuttavia, un reindirizzamento dell'output sovrascrive una destinazione scrivibile esistente senza alcuna conferma. I file vengono sovrascritti dai reindirizzamenti di output a meno che non sia abilitata l'opzione Bash `noclobber`, che può essere eseguita per la sessione corrente con il comando `set -o noclobber` o `set -C`:

```
$ set -o noclobber
$ cat /proc/cpu_info 2>/tmp/error.txt
-bash: /tmp/error.txt: cannot overwrite existing file
```

Per annullare l'impostazione dell'opzione `noclobber` per la sessione corrente, eseguire `set +o noclobber` o `set +C`. Per rendere persistente l'opzione `noclobber`, deve essere inclusa nel profilo Bash dell'utente o nel profilo a livello di sistema.

Anche con l'opzione `noclobber` abilitata è possibile aggiungere dati reindirizzati a contenuto esistente. Ciò si ottiene con un reindirizzamento scritto con due simboli *maggiore di* `>>`:

```
$ cat /proc/cpu_info 2>>/tmp/error.txt
$ cat /tmp/error.txt
cat: /proc/cpu_info: No such file or directory
```

```
cat: /proc/cpu_info: No such file or directory
```

Nell'esempio precedente, il nuovo messaggio di errore è stato aggiunto a quello esistente nel file `/tmp/error.txt`. Se il file non esiste ancora, verrà creato con i nuovi dati.

Anche l'origine dati dello standard input di un processo può essere riassegnata. Il simbolo *minore di* `<` viene utilizzato per reindirizzare il contenuto di un file allo *stdin* di un processo. In questo caso, i dati scorrono da destra a sinistra: si assume che il descrittore riassegnato sia 0 a sinistra del simbolo *minore di* e l'origine dati (un percorso di un file) deve essere a destra del simbolo *minore di*. Il comando `uniq`, come la maggior parte delle utilità della riga di comando per l'elaborazione del testo, accetta i dati inviati allo *stdin* per impostazione predefinita:

```
$ uniq -c </tmp/error.txt
2 cat: /proc/cpu_info: No such file or directory
```

L'opzione `-c` fa visualizzare a `uniq` quante volte una riga ripetuta appare nel testo. Poiché il valore numerico del descrittore di file reindirizzato è stato soppresso, il comando di esempio è equivalente a `uniq -c 0</tmp/error.txt`. Usare un descrittore di file diverso da 0 in un reindirizzamento dell'input ha senso solo in contesti specifici, perché è possibile che un programma richieda dati ai descrittori di file 3, 4 e così via. Infatti, i programmi possono usare qualsiasi numero intero superiore a 2 come nuovi descrittori di file per l'input/output dei dati. Per esempio il seguente codice C legge i dati dal descrittore di file 3 e li replica semplicemente nel descrittore di file 4:

NOTE

Il programma deve gestire correttamente tali descrittori di file, altrimenti potrebbe tentare un'operazione di lettura o scrittura non valida con conseguente crash applicativo.

```
#include <stdio.h>

int main(int argc, char **argv){
    FILE *fd_3, *fd_4;
    // Open file descriptor 3
    fd_3 = fdopen(3, "r");
    // Open file descriptor 4
    fd_4 = fdopen(4, "w");
    // Read from file descriptor 3
    char buf[32];
    while ( fgets(buf, 32, fd_3) != NULL ){
        // Write to file descriptor 4
        fprintf(fd_4, "%s", buf);
    }
}
```



```

}
// Close both file descriptors
fclose(fd_3);
fclose(fd_4);
}

```

Per testarlo, salva il codice di esempio come `fd.c` e compilalo con `gcc -o fd fd.c`. Questo programma richiede che i descrittori di file 3 e 4 siano disponibili in modo da poterli leggere e scrivere. Per esempio, il file precedentemente creato `/tmp/error.txt` può essere utilizzato come sorgente per il descrittore di file 3 e il descrittore di file 4 può essere reindirizzato a `stdout`:

```

$ ./fd 3</tmp/error.txt 4>&1
cat: /proc/cpu_info: No such file or directory
cat: /proc/cpu_info: No such file or directory

```

Dal punto di vista di un programmatore, l'uso di descrittori di file evita di dover gestire l'analisi delle opzioni e i percorsi del file system. Lo stesso descrittore di file può anche essere utilizzato come input e output. In questo caso, il descrittore di file è definito nella riga di comando con simboli sia *minore di* che *maggiore di*, come in `3<>/tmp/error.txt`.

Here Document e Here String

Un altro modo per reindirizzare l'input coinvolge i metodi *Here document* e *Here string*. Il reindirizzamento del *Here document* consente di digitare del testo su più righe che verrà utilizzato come contenuto reindirizzato. Due simboli *meno di* `<<` indicano un reindirizzamento del *Here document*:

```

$ wc -c <<EOF
> How many characters
> in this Here document?
> EOF
43

```

A destra dei due simboli meno di `<<` è il termine di fine `EOF`. La modalità di inserimento terminerà non appena verrà inserita una riga contenente solo il termine di fine. Qualsiasi altro termine può essere utilizzato come termine di fine, ma è importante non inserire caratteri vuoti tra il simbolo *minore di* e il termine di fine. Nell'esempio sopra, le due righe di testo sono state inviate allo *stdin* del comando `wc -c`, che mostra il conteggio dei caratteri. Come per i reindirizzamenti di input per i file, viene assunto lo *stdin* (descrittore di file `0`) se il descrittore di file reindirizzato viene soppresso.

Il metodo Here string è molto simile al metodo Here document, ma solo per una riga:

```
$ wc -c <<<"How many characters in this Here string?"  
41
```

In questo esempio, la stringa a destra dei tre segni *minore di* viene inviata allo *stdin* di `wc -c`, che conta il numero di caratteri. Le stringhe contenenti spazi devono essere racchiuse tra virgolette, altrimenti solo la prima parola verrà utilizzata come Here string e le restanti verranno passate come argomenti al comando.

Esercizi Guidati

1. Oltre ai file di testo, il comando `cat` può funzionare anche con dati binari, come l'invio del contenuto di un dispositivo a blocchi a un file. Usando il reindirizzamento, come può `cat` inviare il contenuto del dispositivo `/dev/sdc` al file `sdc.img` nella directory corrente?
2. Qual è il nome del canale standard reindirizzato dal comando `date 1> now.txt`?
3. Dopo aver provato a sovrascrivere un file usando il reindirizzamento, un utente riceve un errore che informa che l'opzione `noclobber` è abilitata. Come si può disattivare l'opzione `noclobber` per la sessione corrente?
4. Quale sarà il risultato del comando `cat <<./dev/stdout`?

Esercizi Esplorativi

1. Il comando `cat /proc/cpu_info` mostra un messaggio di errore perché `/proc/cpu_info` è inesistente. Il comando `cat /proc/cpu_info 2>1` dove reindirizza il messaggio di errore?
2. Sarà ancora possibile scartare il contenuto inviato a `/dev/null` se l'opzione `noclobber` è abilitata per la sessione corrente della shell?
3. Senza usare `echo`, come potrebbe il contenuto della variabile `$USER` essere reindirizzato allo *stdin* del comando `sha1sum`?
4. Il kernel Linux mantiene collegamenti simbolici in `/proc/PID/fd/` a ogni file aperto da un processo, dove *PID* è il numero di identificazione del processo corrispondente. Come potrebbe l'amministratore di sistema utilizzare quella directory per verificare la posizione dei file di log aperti da `nginx`, supponendo che il suo PID sia `1234`?
5. È possibile eseguire calcoli aritmetici utilizzando solo i comandi incorporati della shell, ma i calcoli in virgola mobile richiedono programmi specifici, come `bc` (*basic calculator*). Con `bc` è anche possibile specificare il numero di cifre decimali, con il parametro `scale`. Tuttavia, `bc` accetta operazioni solo tramite il suo standard input, solitamente inserito in modalità interattiva. Usando una *Here string*, come può l'operazione in virgola mobile `scale = 6; 1 / 3` essere inviato allo standard input di `bc`?

Sommario

Questa lezione copre i metodi per eseguire un programma reindirizzando i suoi canali di comunicazione standard. I processi Linux utilizzano questi canali standard come descrittori di file generici per leggere e scrivere dati, rendendo possibile modificarli arbitrariamente in file o dispositivi. La lezione tratta i seguenti argomenti:

- Cosa sono i descrittori di file e il ruolo che svolgono in Linux.
- I canali di comunicazione standard di ogni processo: *stdin*, *stdout* e *stderr*.
- Come eseguire correttamente un comando utilizzando il reindirizzamento dei dati, sia per l'input che per l'output.
- Come utilizzare *Here Documents* e *Here Strings* nei reindirizzamenti di input.

I comandi e le procedure trattate erano:

- Operatori di Reindirizzamento: `>`, `<`, `>>`, `<<`, `<<<`.
- Comandi `cat`, `set`, `uniq` e `wc`.

Risposte agli Esercizi Guidati

1. Oltre ai file di testo, il comando `cat` può funzionare anche con dati binari, come l'invio del contenuto di un dispositivo a blocchi a un file. Usando il reindirizzamento, come può `cat` inviare il contenuto del dispositivo `/dev/sdc` al file `sdcard.img` nella directory corrente?

```
$ cat /dev/sdc > sdcard.img
```

2. Qual è il nome del canale standard reindirizzato dal comando `date 1> now.txt`?

Standard output o *stdout*

3. Dopo aver provato a sovrascrivere un file usando il reindirizzamento, un utente riceve un errore che informa che l'opzione `noclobber` è abilitata. Come si può disattivare l'opzione `noclobber` per la sessione corrente?

```
set +C o set +o noclobber
```

4. Quale sarà il risultato del comando `cat <<./dev/stdout`?

Bash entrerà in modalità di immissione *Heredoc*, quindi uscirà quando un punto appare in una riga da solo. Il testo digitato verrà reindirizzato a *stdout* (stampato sullo schermo).

Risposte agli Esercizi Esplorativi

1. Il comando `cat /proc/cpu_info` mostra un messaggio di errore perché `/proc/cpu_info` è inesistente. Il comando `cat /proc/cpu_info 2>1` dove reindirizza il messaggio di errore?

A un file chiamato `1` nella directory corrente.

2. Sarà ancora possibile scartare il contenuto inviato a `/dev/null` se l'opzione `noclobber` è abilitata per la sessione corrente della shell?

Sì. `/dev/null` è un file speciale non influenzato da `noclobber`.

3. Senza usare `echo`, come potrebbe il contenuto della variabile `$USER` essere reindirizzato allo *stdin* del comando `sha1sum`?

```
$ sha1sum <<<$USER
```

4. Il kernel Linux mantiene collegamenti simbolici in `/proc/PID/fd/` a ogni file aperto da un processo, dove *PID* è il numero di identificazione del processo corrispondente. Come potrebbe l'amministratore di sistema utilizzare quella directory per verificare la posizione dei file di log aperti da `nginx`, supponendo che il suo PID sia 1234?

Eseguendo il comando `ls -l /proc/1234/fd`, che mostrerà le destinazioni di ogni collegamento simbolico nella directory.

5. È possibile eseguire calcoli aritmetici utilizzando solo i comandi incorporati della shell, ma i calcoli in virgola mobile richiedono programmi specifici, come `bc` (*basic calculator*). Con `bc` è anche possibile specificare il numero di cifre decimali, con il parametro `scale`. Tuttavia, `bc` accetta operazioni solo tramite il suo standard input, solitamente inserito in modalità interattiva. Usando una *Here string*, come può l'operazione in virgola mobile `scale = 6; 1 / 3` essere inviato allo standard input di `bc`?

```
$ bc <<<"scale=6; 1/3"
```



103.4 Lezione 2

Certificazione:	LPIC-1 (101)
Versione:	5.0
Argomento:	103 I Comandi GNU e Unix
Obiettivo:	103.4 Utilizzare stream, pipe e reindirizzamenti
Lezione:	2 di 2

Introduzione

Un principio della filosofia Unix afferma che ogni programma dovrebbe avere uno scopo specifico e non dovrebbe cercare di incorporare funzionalità al di fuori del suo ambito. Ma mantenere le cose semplici non significa risultati meno elaborati, perché diversi programmi possono essere concatenati insieme per produrre un output combinato. Il carattere della barra verticale `|`, noto anche come simbolo *pipe*, può essere utilizzato per creare una pipeline che collega l'output di un programma direttamente all'ingresso di un altro programma, mentre la *command substitution* consente di memorizzare l'output di un programma in una variabile oppure usarlo direttamente come argomento per un altro comando.

Le Pipe

A differenza dei reindirizzamenti, con le pipe i dati fluiscono da sinistra a destra nella riga di comando e l'obiettivo è un altro processo, non un percorso del file system, un descrittore di file o un *Here document*. Il carattere pipe `|` dice alla shell di avviare tutti i comandi distintamente e di collegare l'output del comando precedente all'input del comando successivo, da sinistra a destra. Per esempio, invece di usare i redirect, il contenuto del file `/proc/cpuinfo` inviato allo standard output da `cat` può essere reindirizzato allo *stdin* di `wc` con il seguente comando:


```
$ cat /proc/cpuinfo | wc
208      1184      6096
```

In assenza di un percorso a un file, `wc` conta il numero di righe, parole e caratteri che riceve sul suo *stdin*, come nel caso dell'esempio. Molte pipe possono essere presenti in un comando composto. Nell'esempio seguente vengono utilizzate due pipe:

```
$ cat /proc/cpuinfo | grep 'model name' | uniq
model name      : Intel(R) Xeon(R) CPU           X5355  @ 2.66GHz
```

Il contenuto del file `/proc/cpuinfo` prodotto da `cat /proc/cpuinfo` è stato reindirizzato al comando `grep 'model name'`, che quindi seleziona solo le righe contenenti il termine `model name`. Il sistema che esegue l'esempio ha molte CPU, quindi ci sono righe ripetute con `model name`. L'ultima pipe collega `grep 'model name'` a `uniq`, che si occuperà di eliminare qualsiasi riga uguale alla precedente.

Le pipe possono essere combinate con i reindirizzamenti nella stessa riga di comando. L'esempio precedente può essere riscritto in una forma più semplice:

```
$ grep 'model name' </proc/cpuinfo | uniq
model name      : Intel(R) Xeon(R) CPU           X5355  @ 2.66GHz
```

Il reindirizzamento dell'input per `grep` non è strettamente necessario in quanto `grep` accetta un percorso di file come argomento, ma l'esempio dimostra come costruire tali comandi combinati.

Le pipe e i reindirizzamenti sono esclusivi, ovvero una sorgente può essere mappata su una sola destinazione. Tuttavia, è possibile reindirizzare un output su un file e vederlo ancora sullo schermo con il programma `tee`. Per farlo, il primo programma invia il suo output allo *stdin* di `tee` e a quest'ultimo viene fornito un nome di file dove memorizzare i dati:

```
$ grep 'model name' </proc/cpuinfo | uniq | tee cpu_model.txt
model name      : Intel(R) Xeon(R) CPU           X5355  @ 2.66GHz
$ cat cpu_model.txt
model name      : Intel(R) Xeon(R) CPU           X5355  @ 2.66GHz
```

L'output dell'ultimo programma della catena, generato da `uniq`, viene visualizzato e memorizzato nel file `cpu_model.txt`. Per non sovrascrivere il contenuto del file fornito ma per aggiungere dati a esso, l'opzione `-a` deve essere fornita a `tee`.

Solo l'output standard di un processo viene catturato da una pipe. Supponiamo che sia necessario eseguire un lungo processo di compilazione sullo schermo e allo stesso tempo salvare sia lo standard output che lo standard error in un file per un'ispezione successiva. Supponendo che la tua directory corrente non abbia un *Makefile*, il seguente comando restituirà un errore:

```
$ make | tee log.txt
make: *** No targets specified and no makefile found. Stop.
```

Sebbene mostrato sullo schermo, il messaggio di errore generato da `make` non è stato catturato da `tee` e il file `log.txt` è stato creato vuoto. È necessario eseguire un reindirizzamento prima che una pipe possa acquisire lo *stderr*:

```
$ make 2>&1 | tee log.txt
make: *** No targets specified and no makefile found. Stop.
$ cat log.txt
make: *** No targets specified and no makefile found. Stop.
```

In questo esempio lo *stderr* di `make` è stato reindirizzato allo *stdout*, quindi `tee` è stato in grado di catturarlo con una pipe, visualizzarlo sullo schermo e salvarlo nel file `log.txt`. In casi come questo, può essere utile salvare i messaggi di errore per un controllo successivo.

Sostituzione dei Comandi

Un altro metodo per catturare l'output di un comando è attraverso la *command substitution*. Inserendo un comando all'interno delle virgolette ```, Bash lo sostituisce con il suo output standard. L'esempio seguente mostra come utilizzare lo *stdout* di un programma come argomento per un altro programma:

```
$ mkdir `date +%Y-%m-%d`
$ ls
2019-09-05
```

L'output del programma `date`, la data corrente formattata come *year-month-day*, è stato usato come argomento per creare una directory con `mkdir`. Un risultato identico si ottiene usando `$( )` invece delle virgolette

```
$ rmdir 2019-09-05
$ mkdir $(date +%Y-%m-%d)
$ ls
```

2019-09-05

Lo stesso metodo può essere utilizzato per memorizzare l'output di un comando come variabile:

```
$ OS=`uname -o`  
$ echo $OS  
GNU/Linux
```

Il comando `uname -o` restituisce il nome generico del sistema operativo corrente, che è stato memorizzato nella variabile di sessione `OS`. Assegnare l'output di un comando a una variabile è molto utile negli script, poiché consente di memorizzare e valutare i dati in molti modi distinti.

A seconda dell'output generato dal comando sostituito, la sostituzione del comando integrato potrebbe non essere appropriata. Un metodo più sofisticato per usare l'output di un programma come argomento di un altro programma impiega un intermedio chiamato `xargs`. Il programma `xargs` usa il contenuto che riceve tramite *stdin* per eseguire un dato comando con il contenuto come argomento. Il seguente esempio mostra `xargs` che esegue il programma `identify` con gli argomenti forniti dal programma `find`:

```
$ find /usr/share/icons -name 'debian*' | xargs identify -format "%f: %wx%h\n"  
debian-swirl.svg: 48x48  
debian-swirl.png: 22x22  
debian-swirl.png: 32x32  
debian-swirl.png: 256x256  
debian-swirl.png: 48x48  
debian-swirl.png: 16x16  
debian-swirl.png: 24x24  
debian-swirl.svg: 48x48
```

Il programma `identify` fa parte di *ImageMagick*, un insieme di strumenti a riga di comando per ispezionare, convertire e modificare la maggior parte dei tipi di file immagine. Nell'esempio, `xargs` ha preso tutti i percorsi elencati da `find` e li ha inseriti come argomenti di `identify`, che poi mostra le informazioni per ogni file formattato come richiesto dall'opzione `-format`. I file trovati da `find` nell'esempio sono immagini contenenti il logo della distribuzione in un filesystem Debian. `-format` è un parametro per `identify`, non per `xargs`.

L'opzione `-n 1` richiede che `xargs` esegua il comando dato con un solo argomento alla volta. Nel caso dell'esempio, invece di passare tutti i percorsi trovati da `find` come un elenco di argomenti a `identify`, usando `xargs -n 1` si eseguirà il comando `identify` per ogni percorso separatamente. Usando `-n 2` si eseguirà `identify` con due percorsi come argomenti, `-n 3` con

tre percorsi come argomenti e così via. Allo stesso modo, quando `xargs` elabora contenuti multilinea — come nel caso dell'input fornito da `find` — l'opzione `-L` può essere usata per limitare il numero di righe che saranno usate come argomenti per l'esecuzione del comando.

NOTE

Usare `xargs` con l'opzione `-n 1` o `-L 1` per elaborare l'output generato da `find` potrebbe non essere necessario. Il comando `find` ha l'opzione `-exec` per eseguire un dato comando per ogni elemento del risultato di ricerca.

Se i percorsi hanno caratteri di spazio, è importante eseguire `find` con l'opzione `-print0`. Questa opzione indica a `find` di usare un carattere nullo tra ogni voce in modo che l'elenco possa essere correttamente analizzato da `xargs` (l'output è stato soppresso):

```
$ find . -name '*avi' -print0 -o -name '*mp4' -print0 -o -name '*mkv' -print0 | xargs -0 du
| sort -n
```

L'opzione `-0` dice a `xargs` che il carattere nullo dovrebbe essere usato come separatore. In questo modo i percorsi dei file forniti da `find` vengono analizzati correttamente anche se contengono caratteri vuoti o altri caratteri speciali. L'esempio precedente mostra come utilizzare il comando `du` per scoprire l'utilizzo del disco di ogni file trovato e quindi ordinare i risultati per dimensione. L'output è stato soppresso per concisione. Nota che per ogni criterio di ricerca è necessario inserire l'opzione `-print0` per `find`.

Per impostazione predefinita, `xargs` inserisce gli argomenti del comando eseguito per ultimo. Per cambiare quel comportamento, dovrebbe essere usata l'opzione `-I`:

```
$ find . -mindepth 2 -name '*avi' -print0 -o -name '*mp4' -print0 -o -name '*mkv' -print0 |
xargs -0 -I PATH mv PATH ./
```

Nell'ultimo esempio, ogni file trovato da `find` viene spostato nella directory corrente. Poiché il percorso di origine deve essere dato a `mv` prima del percorso di destinazione, viene fornito un termine di sostituzione all'opzione `-I` di `xargs` che viene poi opportunamente posizionato accanto a `mv`. Utilizzando il carattere *null* come separatore, non è necessario racchiudere il termine di sostituzione tra virgolette.

Esercizi Guidati

1. È conveniente salvare la data di esecuzione delle azioni eseguite dagli script automatici. Il comando `date +%Y-%m-%d` mostra la data corrente nel formato *year-month-day*. Come può l'output di un tale comando essere memorizzato in una variabile di shell chiamata `TODAY` usando la sostituzione del comando?

2. Usando il comando `echo`, come può il contenuto della variabile `TODAY` essere inviato allo standard input del comando `sed` `s /-/. / g`?

3. In che modo l'output del comando `date +%Y-%m-%d` può essere usato come *Here string* da fornire a `sed` `s /-/. / g`?

4. Il comando `convert image.jpeg -resize 25% small/image.jpeg` crea una versione più piccola di `image.jpeg` e colloca l'immagine risultante in un file con lo stesso nome all'interno della sottodirectory `small`. Usando `xargs`, come è possibile eseguire lo stesso comando per ogni immagine elencata nel file `filelist.txt`?

Esercizi Esplorativi

1. Una semplice routine di backup crea periodicamente un'immagine della partizione `/dev/sda1` con `dd < /dev/sda1> sda1.img`. Per eseguire futuri controlli di integrità dei dati, la routine genera anche un hash SHA1 del file con `sha1sum < sda1.img > sda1.sha1`. Aggiungendo pipe e il comando `tee`, come sarebbero combinati questi due comandi in uno?

2. Il comando `tar` viene utilizzato per archiviare molti file in un unico file, preservando la struttura delle directory. L'opzione `-T` permette di specificare un file contenente i percorsi da archiviare. Per esempio, `find /etc -type f | tar -cJ -f /srv/backup/etc.tar.xz -T -` crea un file tar compresso `etc.tar.xz` dalla lista fornita dal comando `find` (l'opzione `-T -` indica lo standard input come elenco dei percorsi). Al fine di evitare possibili errori di analisi dovuti a percorsi contenenti spazi, quali opzioni di comando dovrebbero essere presenti per `find` e `tar`?

3. Invece di aprire una nuova sessione di shell remota, il comando `ssh` può semplicemente eseguire un comando indicato come argomento: `ssh user@storage "remote command"`. Dato che `ssh` permette anche di reindirizzare l'output standard di un programma locale allo standard input del programma remoto, come potrebbe il comando `cat` reindirizzare un file locale chiamato `etc.tar.gz` a `/srv/backup/etc.tar.gz` in `user@storage` tramite `ssh`?

Sommario

Questa lezione copre le tradizionali tecniche di comunicazione tra processi impiegate da Linux. La *command pipelining* crea un canale di comunicazione unidirezionale tra due processi e la *command substitution* consente di memorizzare l'output di un processo in una variabile di shell. La lezione segue i seguenti passaggi:

- Come la *pipe* può essere utilizzata per trasmettere l'output di un processo all'input di un altro processo.
- Lo scopo dei comandi *tee* e *xargs*.
- Come catturare l'output di un processo con la *command substitution*, memorizzandolo in una variabile o usandolo direttamente come parametro per un altro comando.

I comandi e le procedure trattate erano:

- Comand *pipelining* con `|`.
- Sostituzione dei comandi con backtick e `$()`.
- Comandi *tee*, *xargs* e *find*.

Risposte agli Esercizi Guidati

1. È conveniente salvare la data di esecuzione delle azioni eseguite dagli script automatici. Il comando `date +%Y-%m-%d` mostra la data corrente nel formato *year-month-day*. Come può l'output di un tale comando essere memorizzato in una variabile di shell chiamata `TODAY` usando la sostituzione del comando?

```
$ TODAY=`date +%Y-%m-%d`
```

0

```
$ TODAY=$(date +%Y-%m-%d)
```

2. Usando il comando `echo`, come può il contenuto della variabile `TODAY` essere inviato allo standard input del comando `sed s /-/. / g`?

```
$ echo $TODAY | sed s /-/. / g
```

3. In che modo l'output del comando `date +%Y-%m-%d` può essere usato come *Here string* da fornire a `sed s /-/. / g`?

```
$ sed s /-/. / g <<< `date +%Y-%m-%d`
```

0

```
$ sed s /-/. / g <<< $(date +%Y-%m-%d)
```

4. Il comando `convert image.jpeg -resize 25% small/image.jpeg` crea una versione più piccola di `image.jpeg` e colloca l'immagine risultante in un file con lo stesso nome all'interno della sottodirectory `small`. Usando `xargs`, come è possibile eseguire lo stesso comando per ogni immagine elencata nel file `filelist.txt`?

```
$ xargs -I IMG convert IMG -resize 25% small/IMG < filelist.txt
```

0


```
$ cat filelist.txt | xargs -I IMG convert IMG -resize 25% small/IMG
```

Risposte agli Esercizi Esplorativi

1. Una semplice routine di backup crea periodicamente un'immagine della partizione `/dev/sda1` con `dd < /dev/sda1> sda1.img`. Per eseguire futuri controlli di integrità dei dati, la routine genera anche un hash SHA1 del file con `sha1sum < sda1.img > sda1.sha1`. Aggiungendo pipe e il comando `tee`, come sarebbero combinati questi due comandi in uno?

```
# dd < /dev/sda1 | tee sda1.img | sha1sum > sda1.sha1
```

2. Il comando `tar` viene utilizzato per archiviare molti file in un unico file, preservando la struttura delle directory. L'opzione `-T` permette di specificare un file contenente i percorsi da archiviare. Per esempio, `find /etc -type f | tar -cJ -f /srv/backup/etc.tar.xz -T -` crea un file tar compresso `etc.tar.xz` dalla lista fornita dal comando `find` (l'opzione `-T -` indica lo standard input come elenco dei percorsi). Al fine di evitare possibili errori di analisi dovuti a percorsi contenenti spazi, quali opzioni di comando dovrebbero essere presenti per `find` e `tar`?

Opzioni `-print0` e `--null`:

```
$ find /etc -type f -print0 | tar -cJ -f /srv/backup/etc.tar.xz --null -T -
```

3. Invece di aprire una nuova sessione di shell remota, il comando `ssh` può semplicemente eseguire un comando indicato come argomento: `ssh user@storage "remote command"`. Dato che `ssh` permette anche di reindirizzare l'output standard di un programma locale allo standard input del programma remoto, come potrebbe il comando `cat` reindirizzare un file locale chiamato `etc.tar.gz` a `/srv/backup/etc.tar.gz` in `user@storage` tramite `ssh`?

```
$ cat etc.tar.gz | ssh user@storage "cat > /srv/backup/etc.tar.gz"
```

OR

```
$ ssh user@storage "cat > /srv/backup/etc.tar.gz" < etc.tar.gz
```



103.5 Creare, controllare e terminare i processi

Obiettivi LPI di riferimento

LPIC-1 v5, Exam 101, Objective 103.5

Peso

4

Aree di Conoscenza Chiave

- Eseguire lavori in primo piano e in background.
- Consentire a un programma di continuare l'esecuzione dopo il logout.
- Controllare i processi attivi.
- Selezionare e ordinare i processi in visualizzazione.
- Inviare segnali ai processi.

Di seguito è riportato un elenco parziale dei file, dei termini e dei comandi utilizzati

- `&`
- `bg`
- `fg`
- `jobs`
- `kill`
- `nohup`
- `ps`
- `top`
- `free`

- `uptime`
- `pgrep`
- `pkill`
- `killall`
- `watch`
- `screen`
- `tmux`



Linux
Professional
Institute

103.5 Lezione 1

Certificazione:	LPIC-1
Versione:	5.0
Argomento:	103 Comandi GNU e Unix
Obiettivo:	103.5 Creare, controllare e terminare i processi
Lezione:	1 di 2

Introduzione

Ogni volta che invochiamo un comando, vengono avviati uno o più processi. Un amministratore di sistema ben addestrato non solo deve creare processi, ma anche essere in grado di tenerne traccia e inviare loro diversi tipi di segnali se e quando richiesto. In questa lezione esamineremo il controllo del lavoro e il monitoraggio dei processi.

Controllare i Job

I *Job* sono processi che sono stati avviati in modo interattivo tramite un terminale, inviati in background e non hanno ancora terminato l'esecuzione. Puoi scoprire i *job* attivi (e il loro stato) nel tuo sistema Linux eseguendo `jobs`:

```
$ jobs
```

Il comando `jobs` sopra non ha prodotto alcun output, il che significa che non ci sono *job* attivi al momento. Creiamo il nostro primo *job* eseguendo un comando che impiega un po' di tempo per terminare l'esecuzione (il comando `sleep` con un parametro di `60`) e —durante

l'esecuzione — premi `Ctrl` + `Z`:

```
$ sleep 60
^Z
[1]+  Stopped                  sleep 60
```

L'esecuzione del comando è stata interrotta (o — meglio — sospesa) e il prompt dei comandi è nuovamente disponibile. Puoi cercare lavori una seconda volta e troverai ora quello *sospeso* :

```
$ jobs
[1]+  Stopped                  sleep 60
```

Cerchiamo di spiegare l'output:

[1]

Questo numero è l'ID del lavoro e può essere usato — preceduto da un simbolo di percentuale (%) — per cambiare lo stato del lavoro dalle utilità `fg`, `bg` e `kill` (come ti verrà mostrato in seguito).

+

Il segno più indica il lavoro predefinito corrente (ovvero, l'ultimo sospeso o inviato in *background*). Il lavoro precedente è contrassegnato con un segno meno (-). Eventuali altri lavori precedenti non vengono contrassegnati.

Stopped

Descrizione dello stato del lavoro.

sleep 60

Il comando o il lavoro stesso.

Con l'opzione `-l`, i lavori visualizzeranno inoltre l'ID del processo (PID) subito prima dello stato:

```
$ jobs -l
[1]+  1114 Stopped              sleep 60
```

Le restanti possibili opzioni di lavoro sono:

-n

Elenca solo i processi che hanno cambiato stato dall'ultima notifica. I possibili stati includono:

Running, Stopped, Terminated o Done.

-p

Elenca gli ID dei processi.

-r

Elenca solo i lavori in esecuzione.

-s

Elenca solo i lavori fermati (o sospesi).

NOTE Ricorda, un lavoro ha sia un *job ID* che un *process ID* (PID).

Specifiche del Job

Il comando `jobs` così come altre utilità come `fg`, `bg` e `kill` (che vedrai nella prossima sezione) necessitano di una specifica del lavoro (o `jobspec`) per agire su un particolare lavoro. Come abbiamo appena visto, questo può essere — e normalmente lo è — l'ID del lavoro preceduto da `%`. Tuttavia, sono possibili anche altre specifiche di lavoro. Vediamole insieme:

%n

Job il cui numero di identificazione è `n`:

```
$ jobs %1
[1]+  Stopped                  sleep 60
```

%str

Job la cui riga di comando inizia con `str`:

```
$ jobs %s1
[1]+  Stopped                  sleep 60
```

%?str

Job la cui riga di comando contiene `str`:

```
$ jobs %?le
[1]+  Stopped                  sleep 60
```

%+ or %%

Job corrente (l'ultimo avviato in background o sospeso in primo piano):

```
$ jobs %+
[1]+  Stopped                  sleep 60
```

%-

Job precedente (quello che era %+ prima di quello predefinito, quello attuale):

```
$ jobs %-
[1]+  Stopped                  sleep 60
```

Nel nostro caso, poiché c'è un solo *job*, appare come sia attuale sia precedente.

Stato del Job: Sospensione, Foreground e Background

Una volta che un lavoro è in background o è stato sospeso, possiamo eseguire una delle tre operazioni seguenti:

1. Portalo in primo piano con **fg**:

```
$ fg %1
sleep 60
```

fg sposta il job specificato in primo piano e lo rende il lavoro corrente. Ora possiamo aspettare che finisca, fermarlo di nuovo con **Ctrl + Z** o terminarlo con **Ctrl + C**

2. Portalo in background con **bg**:

```
$ bg %1
[1]+ sleep 60 &
```

Una volta in background, il lavoro può essere riportato in primo piano con **fg** o terminato (vedi sotto). Nota la *e commerciale* (&) che significa che il lavoro è stato inviato in background. È un dato di fatto, puoi anche usare la *e commerciale* per avviare un processo direttamente in background:

```
$ sleep 100 &
```



```
[2] 970
```

Insieme all’ID lavoro del nuovo lavoro ([2]), ora otteniamo anche il suo ID processo (970). Ora entrambi i lavori sono in esecuzione in background:

```
$ jobs
[1]-  Running                sleep 60 &
[2]+  Running                sleep 100 &
```

Un po' più tardi il primo lavoro termina l’esecuzione:

```
$ jobs
[1]-  Done                   sleep 60
[2]+  Running                sleep 100 &
```

3. Terminalo tramite un segnale `SIGTERM` con il comando `kill`:

```
$ kill %2
```

Per assicurarti che il lavoro sia stato terminato, esegui di nuovo `jobs`:

```
$ jobs
[2]+  Terminated           sleep 100
```

NOTE

Se non viene specificato alcun lavoro, `fg` e `bg` agiranno su quello corrente predefinito. Tuttavia, `kill` richiede sempre una specifica del lavoro.

Distaccare i Job: `nohup`

I lavori che abbiamo visto nelle sezioni precedenti erano tutti allegati alla sessione dell’utente che li ha invocati. Ciò significa che se la sessione viene terminata, lo saranno anche i *job*. Tuttavia, è possibile scollegare i lavori dalle sessioni e farli eseguire anche dopo la chiusura della sessione. Ciò si ottiene con il comando `nohup` (“no hangup”). La sintassi è la seguente:

```
nohup COMMAND &
```

Ricorda, `&` invia il processo in background e libera il terminale su cui stai lavorando

Scollegiamo il lavoro in background `ping localhost` dalla sessione corrente:

```
$ nohup ping localhost &
[1] 1251
$ nohup: ignoring input and appending output to 'nohup.out'
^C
```

L'output ci mostra l'ID del lavoro ([1]) e il PID (1251), seguito da un messaggio che ci informa del file `nohup.out`. Questo è il file predefinito in cui verranno salvati *stdout* e *stderr*. Ora possiamo premere `ctrl` + `c` per liberare il prompt dei comandi, chiudere la sessione, avviarne un'altra come root e usare `tail -f` per verificare se il comando è in esecuzione e l'output viene scritto nel file predefinito:

```
$ exit
logout
$ tail -f /home/carol/nohup.out
64 bytes from localhost (::1): icmp_seq=3 ttl=64 time=0.070 ms
64 bytes from localhost (::1): icmp_seq=4 ttl=64 time=0.068 ms
64 bytes from localhost (::1): icmp_seq=5 ttl=64 time=0.070 ms
^C
```

TIP

Invece di usare il valore predefinito `nohup.out` potresti aver specificato un file di output di tua scelta con `nohup ping localhost > /path/to/your/file &`.

Se vogliamo terminare il processo, dovremmo specificare il suo PID

```
# kill 1251
```

Controllare i Processi

Un processo o un'attività è un'istanza di un programma in esecuzione. Pertanto, crei nuovi processi ogni volta che digiti comandi nel terminale.

Il comando `watch` esegue periodicamente un programma (ogni due secondi per impostazione predefinita) e ci permette di *osservare* il cambiamento dell'output del programma nel tempo. Per esempio, possiamo monitorare come cambia la media del carico man mano che vengono eseguiti più processi digitando `watch uptime`:

```
Every 2.0s: uptime                debian: Tue Aug 20 23:31:27 2019
```

```
23:31:27 up 21 min,  1 user,  load average: 0.00, 0.00, 0.00
```

Il comando viene eseguito finché non viene interrotto, quindi dovremmo fermarlo con `Ctrl + C`. Otteniamo due righe come output: la prima corrisponde a `watch` e ci dice quanto spesso verrà eseguito il comando (Ogni 2.0s: uptime), quale comando/programma guardare (uptime) e nome host e data (debian: Tue Aug 20 23:31:27 2019). La seconda riga di output è il tempo di attività e include il tempo (23:31:27), quanto tempo è stato attivo il sistema (up 21 min), il numero di utenti attivi (1 user) e il carico di sistema medio o numero di processi in esecuzione o in attesa negli ultimi 1, 5 e 15 minuti (load average: 0.00, 0.00, 0.00)

Allo stesso modo, puoi controllare l'uso della memoria quando vengono creati nuovi processi con `watch free`:

```
Every 2.0s: free          debian: Tue Aug 20 23:43:37 2019

23:43:37 up 24 min,  1 user,  load average: 0.00, 0.00, 0.00

      total        used        free      shared  buff/cache   available
Mem:    16274868    493984    14729396        35064     1051488     15462040
Swap:    16777212         0     16777212
```

Per cambiare l'intervallo di aggiornamento di `watch` usa le opzioni `-n` o `--interval` più il numero di secondi come in:

```
$ watch -n 5 free
```

Ora il comando `free` verrà eseguito ogni 5 secondi.

Per maggiori informazioni sulle opzioni di `uptime`, `free` e `watch`, fare riferimento alle rispettive pagine di manuale.

NOTE

Le informazioni fornite da `uptime` e `free` sono anche integrate negli strumenti più completi `top` e `ps` (vedi sotto).

Inviare Segnali ai Processi: kill

Ogni singolo processo ha un identificatore di processo univoco o PID. Un modo per scoprire il PID di un processo è usare il comando `pgrep` seguito dal nome del processo:

```
$ pgrep sleep
```

1201

NOTE

Un identificatore di processo può anche essere scoperto tramite il comando `pidof` (per esempio `pidof sleep`).

Simile a `pgrep`, il comando `pkill` termina un processo in base al suo nome:

```
$ pkill sleep
[1]+  Terminated          sleep 60
```

Per terminare più istanze dello stesso processo, è possibile utilizzare il comando `killall`:

```
$ sleep 60 &
[1] 1246
$ sleep 70 &
[2] 1247
$ killall sleep
[1]-  Terminated          sleep 60
[2]+  Terminated          sleep 70
```

Sia `pkill` che `killall` funzionano in modo molto simile a `kill` in quanto inviano un segnale di terminazione ai processi specificati. Se non viene fornito alcun segnale, viene inviato il valore predefinito di "SIGTERM". Tuttavia, `kill` accetta solo un lavoro o un ID di processo come argomento.

I segnali possono essere specificati mediante:

- Nome:

```
$ kill -SIGHUP 1247
```

- Numero:

```
$ kill -1 1247
```

- Opzione:

```
$ kill -s SIGHUP 1247
```

Per fare in modo che `kill` funzioni in modo simile a `pkill` o `killall` (ed evitare a noi i vari comandi per trovare prima i PID) possiamo usare la sostituzione dei comandi:

```
$ kill -1 $(pgrep sleep)
```

Come dovresti già sapere, una sintassi alternativa è `kill -1 `pgrep sleep``.

TIP

Per un elenco esaustivo di tutti i segnali accettati da `kill` e dei loro codici, digita `kill -l` nel terminale. Usa `-KILL` (-9 o `-s KILL`) per terminare i processi quando qualsiasi altro segnale fallisce.

top e ps

Quando si tratta di monitoraggio dei processi, due strumenti insostituibili sono `top` e `ps`. Mentre il primo produce output dinamicamente, il secondo lo fa staticamente. In ogni caso, entrambe sono ottime utility per avere una visione completa di tutti i processi del sistema.

Interagire con top

Per invocare `top`, digita semplicemente `top`:

```
$ top
```

```
top - 11:10:29 up 2:21, 1 user, load average: 0,11, 0,20, 0,14
Tasks: 73 total, 1 running, 72 sleeping, 0 stopped, 0 zombie
%Cpu(s): 0,0 us, 0,3 sy, 0,0 ni, 99,7 id, 0,0 wa, 0,0 hi, 0,0 si, 0,0 st
KiB Mem : 1020332 total, 909492 free, 38796 used, 72044 buff/cache
KiB Swap: 1046524 total, 1046524 free, 0 used. 873264 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
436	carol	20	0	42696	3624	3060	R	0,7	0,4	0:00.30	top
4	root	20	0	0	0	0	S	0,3	0,0	0:00.12	kworker/0:0
399	root	20	0	95204	6748	5780	S	0,3	0,7	0:00.22	sshd
1	root	20	0	56872	6596	5208	S	0,0	0,6	0:01.29	systemd
2	root	20	0	0	0	0	S	0,0	0,0	0:00.00	kthreadd
3	root	20	0	0	0	0	S	0,0	0,0	0:00.02	ksoftirqd/0
5	root	0	-20	0	0	0	S	0,0	0,0	0:00.00	kworker/0:0H
6	root	20	0	0	0	0	S	0,0	0,0	0:00.00	kworker/u2:0
7	root	20	0	0	0	0	S	0,0	0,0	0:00.08	rcu_sched
8	root	20	0	0	0	0	S	0,0	0,0	0:00.00	rcu_bh
9	root	rt	0	0	0	0	S	0,0	0,0	0:00.00	migration/0
10	root	0	-20	0	0	0	S	0,0	0,0	0:00.00	lru-add-drain

(...)

`top` consente all'utente alcune interazioni. Per impostazione predefinita, l'output viene ordinato in base alla percentuale di tempo della CPU utilizzata da ciascun processo in ordine decrescente. Questo comportamento può essere modificato premendo i seguenti tasti da dentro `top`:

M

Ordina per utilizzo della *memoria*.

N

Ordina per *numero* ID processo.

T

Ordina per *tempo* di esecuzione.

P

Ordina per *percentuale* di utilizzo della CPU.

TIP

Per passare dall'ordine discendente a quello ascendente, basta premere `R`.

Altre tasti interessanti per interagire con `top` sono:

? o h

Help.

k

Termina un processo. `top` chiederà qual'è il PID da terminare e quale segnale inviare (per impostazione predefinita `SIGTERM` o 15).

r

Cambia la priorità di un processo (*renice*). `top` ti chiederà il valore *nice*. I valori possibili sono compresi tra -20 e 19, ma solo il superutente (*root*) può impostarlo su un valore negativo o inferiore a quello corrente.

u

Elenca i processi di un particolare utente (per impostazione predefinita vengono mostrati i processi di tutti gli utenti).

c

Mostra i percorsi assoluti dei programmi e distingue tra processi userspace e processi kernelspace (tra parentesi quadre).

V

Vista foresta/gerarchia dei processi.

t e m

Cambia l'aspetto delle letture della CPU e della memoria rispettivamente in un ciclo a quattro fasi: le prime due pressioni mostrano le barre di avanzamento, la terza nasconde la barra e la quarta la riporta indietro.

W

Salva le impostazioni di configurazione in `~/ .toprc`.

TIP

Una versione più elaborata e più amichevole di `top` è `htop`. Un'altra alternativa, forse più esaustiva, è `atop`. Se non sono già installati nel tuo sistema, usa il tuo gestore di pacchetti per installarli e provarli.

Una spiegazione dell'output di top

L'output di `top` è diviso in due aree: la *summary area* e la *task area*.

La Summary Area in top

La *summary area* è composta dalle cinque righe superiori e fornisce le seguenti informazioni:

- `top - 11:10:29 up 2:21, 1 user, load average: 0,11, 0,20, 0,14`
 - current time (in formato 24-ore): `11:20:29`
 - uptime (da quanto tempo il sistema è avviato e funzionante): `up 2:21`
 - numero di utenti attivi e carico medio della CPU per gli ultimi 1, 5 e 15 minuti, rispettivamente: `load average: 0,11, 0,20, 0,14`
- `Tasks: 73 total, 1 running, 72 sleeping, 0 stopped, 0 zombie` (informazione riguardo i processi)
 - numero totale di processi in modalità attiva: `73 total`
 - running (quelli in esecuzione): `1 running`
 - sleeping (quelli in attesa di riprendere l'esecuzione): `72 sleeping`
 - stopped (da un segnale di controllo sul job): `0 stopped`
 - zombie (quelli che hanno completato l'esecuzione ma stanno ancora aspettando che il loro processo padre li rimuova dalla tabella dei processi): `0 zombie`
- `%Cpu(s): 0,0 us, 0,3 sy, 0,0 ni, 99,7 id, 0,0 wa, 0,0 hi, 0,0 si, 0,0 st`

(percentuale del tempo di CPU impiegato)

- processi utente: 0,0 us
 - processi system/kernel: 0,4 sy
 - processi impostati tramite un valore *nice* — più alto è il valore, minore è la priorità: 0,0 ni
 - tempo di inattività della CPU: 99,7 id
 - processi in attesa di operazioni di I/O: 0,0 wa
 - processi che servono gli *interrupt hardware* — periferiche che inviano segnali al processore che richiedono attenzione: 0,0 hi
 - processi che servono gli *interrupt software*: 0,0 si
 - processi che servono le attività di altre macchine virtuali in un ambiente virtuale, quindi rubano tempo processore: 0,0 st
- KiB Mem : 1020332 total, 909492 free, 38796 used, 72044 buff/cache (informazioni sulla memoria in kilobyte)
 - la quantità totale di memoria: 1020332 total
 - memoria inutilizzata: 909492 free
 - memoria in uso: 38796 used
 - la memoria che viene bufferizzata e memorizzata nella cache per evitare un accesso eccessivo al disco: 72044 buff/cache

Nota come total è la somma degli altri tre valori — free, used and buff/cache — (circa 1 GB nel nostro caso)
 - KiB Swap: 1046524 total, 1046524 free, 0 used. 873264 avail Mem (informazioni sullo swap in kilobyte)
 - la quantità totale di spazio di swap: 1046524 total
 - spazio di swap non utilizzato: 1046524 free
 - spazio di swap in uso: 0 used
 - la quantità di memoria di swap che può essere allocata ai processi senza causare ulteriori scambi: 873264 avail Mem

La Task Area in top: Campi e Colonne

Sotto la *summary area* si trova la *task area*, che include una serie di *campi* e *colonne* che riportano informazioni sui processi in esecuzione:

PID

Identificativo di processo.

USER

Utente che ha eseguito il comando che ha generato il processo.

PR

Priorità del processo al kernel.

NI

Valore di nice del processo. I valori più bassi hanno una priorità più alta.

VIRT

Quantità totale di memoria utilizzata dal processo (incluso lo Swap).

RES

Memoria RAM utilizzata dal processo.

SHR

Memoria del processo condivisa con altri processi.

S

Stato del processo. I valori includono: S (interruptible sleep — in attesa del termine di un evento), R (*runnable* — in esecuzione o in coda da eseguire) o Z (zombie - processi figli terminati le cui strutture dati non sono ancora state rimosse dalla tabella dei processi).

%CPU

Percentuale di CPU utilizzata dal processo.

%MEM

Percentuale di RAM utilizzata dal processo, ovvero il valore RES espresso in percentuale.

TIME+

Tempo totale di attività del processo.

COMMAND

Nome del comando/programma che ha generato il processo.

Visualizzazione statica dei processi: ps

Come detto sopra, **ps** mostra un'istantanea dei processi. Per vedere tutti i processi con un

terminale (tty), digita `ps a`:

```
$ ps a
PID TTY      STAT   TIME COMMAND
386 tty1      Ss+    0:00 /sbin/agetty --noclear tty1 linux
424 tty7      Ssl+   0:00 /usr/lib/xorg/Xorg :0 -seat seat0 (...)
655 pts/0     Ss      0:00 -bash
1186 pts/0    R+      0:00 ps a
(...)
```

Una Spiegazione della Sintassi e dell'Output delle Opzioni di `ps`

Per quanto riguarda le opzioni, `ps` può accettare tre diversi stili: BSD, UNIX e GNU. Vediamo come funziona ciascuno di questi stili quando si cercano informazioni su un particolare ID processo:

BSD

Le opzioni non seguono alcun trattino iniziale:

```
$ ps p 811
PID TTY      STAT   TIME COMMAND
811 pts/0     S       0:00 -su
```

UNIX

Le opzioni seguono un trattino iniziale:

```
$ ps -p 811
PID TTY      TIME CMD
811 pts/0    00:00:00 bash
```

GNU

Le opzioni seguono due trattini iniziali:

```
$ ps --pid 811
PID TTY      TIME CMD
811 pts/0    00:00:00 bash
```

In tutti e tre i casi, `ps` riporta le informazioni sul processo il cui PID è 811 — in questo caso, `bash`.

Allo stesso modo, puoi usare `ps` per cercare i processi avviati da un particolare utente:

- `ps U carol` (BSD)
- `ps -u carol` (UNIX)
- `ps --user carol` (GNU)

Controlliamo i processi avviati da `carol`:

```
$ ps U carol
  PID TTY          STAT       TIME COMMAND
   811 pts/0        S          0:00  -su
   898 pts/0        R+         0:00  ps U carol
```

Ha avviato due processi: `bash` (`-su`) e `ps` (`ps U carol`). La colonna `STAT` ci dice lo stato del processo (vedi sotto).

Possiamo ottenere il meglio da `ps` combinando alcune delle sue opzioni. Un comando molto utile (che produce un output simile a quello di `top`) è `ps aux` (stile BSD). In questo caso, vengono mostrati i processi di tutte le shell (non solo quella corrente). Il significato degli interruttori è il seguente:

a

Mostra i processi collegati a una `tty` o un terminale.

u

Visualizza in un formato orientato all'utente.

x

Mostra i processi che non sono collegati a una `tty` o un terminale.

```
$ ps aux
USER      PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.0  0.1 204504 6780 ?        Ss   14:04   0:00 /sbin/init
root         2  0.0  0.0      0     0 ?        S    14:04   0:00 [kthreadd]
root         3  0.0  0.0      0     0 ?        S    14:04   0:00 [ksoftirqd/0]
root         5  0.0  0.0      0     0 ?        S<   14:04   0:00 [kworker/0:0H]
root         7  0.0  0.0      0     0 ?        S    14:04   0:00 [rcu_sched]
root         8  0.0  0.0      0     0 ?        S    14:04   0:00 [rcu_bh]
root         9  0.0  0.0      0     0 ?        S    14:04   0:00 [migration/0]
(...)
```

Spieghiamo il significato delle colonne:

USER

Titolare del processo.

PID

Identificativo di processo.

%CPU

Percentuale di CPU utilizzata.

%MEM

Percentuale di memoria fisica utilizzata.

VSZ

Memoria virtuale di processo in KiB.

RSS

Memoria fisica non di tipo swap utilizzata dal processo in KiB.

TT

Terminale (tty) che controlla il processo.

STAT

Codice che rappresenta lo stato del processo. Oltre a S, R e Z (che abbiamo visto descrivendo l'output di `top`), altri possibili valori includono: D (uninterruptible sleep — normalmente in attesa di I/O), T(stopped — normalmente da un segnale di controllo). Alcuni modificatori extra includono: < (high-priority — non adatto ad altri processi), N (low-priority — non impattante con altri processi), o +(nel gruppo di processi in primo piano).

STARTED

Ora in cui è iniziato il processo.

TIME

Tempo CPU accumulato.

COMMAND

Comando che ha avviato il processo.

Esercizi Guidati

1. `oneko` è un simpatico programma che mostra un gatto che insegue il cursore del mouse. Se non è già installato nel sistema desktop, installalo utilizzando il gestore di pacchetti della tua distribuzione. Lo useremo per studiare il controllo dei job.

- Avvia il programma. Come si fa a farlo?

- Muovi il cursore del mouse per vedere come il gatto lo insegue. Ora sospendi il processo. Come si fa a farlo? Qual è l'output?

- Controlla quanti job hai attualmente. Cosa scrivi? Qual è l'output?

- Ora invialo in background specificando il suo job ID. Qual è l'output? Come puoi sapere che il lavoro è in esecuzione in background?

- Infine, termina il job specificando il suo ID. Cosa scrivi?

2. Scopri i PID di tutti i processi generati dal server web *Apache HTTPD* (`apache2`) con due diversi comandi:

3. Termina tutti i processi `apache2` senza usare i loro PID e con due diversi comandi:

4. Supponi di dover terminare tutte le istanze di `apache2` e non hai tempo per scoprire quali sono i loro PID. Come lo realizzeresti usando `kill` con il segnale predefinito `SIGTERM` in una riga:

5. Avvia `top` e interagisci con esso eseguendo quanto segue:

- Mostra una vista della foresta dei processi:

-
- Mostra i percorsi completi dei processi che si differenziano tra spazio utente e spazio kernel:
-

6. Digita il comando `ps` per visualizzare tutti i processi avviati dall'utente *Apache HTTPD web server* (`wwwdata`):

- Utilizzando la sintassi BSD:
-

- Utilizzando la sintassi UNIX:
-

- Utilizzando la sintassi GNU:
-

Esercizi Esplorativi

1. Il segnale `SIGHUP` può essere usato come un modo per riavviare alcuni demoni. Con il server web *Apache HTTPD* — per esempio — l'invio di `SIGHUP` al processo genitore (quello avviato da `init`) termina i suoi figli. Il genitore, tuttavia, rilegge i suoi file di configurazione, riapre i file di log e genera un nuovo set di figli. Eseguire le seguenti attività:

- Avviare il web server:

- Assicurati di conoscere il PID del processo genitore:

- Riavvia il server web Apache HTTPD inviando il segnale `SIGHUP` al processo genitore:

- Controlla che il genitore non sia stato terminato e che siano stati generati nuovi figli:

2. Sebbene inizialmente statico, l'output di `ps` può essere *reso* dinamico combinando `ps` e `watch`. Monitoreremo il server web *Apache HTTPD* per nuove connessioni. Prima di eseguire le attività descritte di seguito si consiglia di leggere la descrizione della direttiva `MaxConnectionsPerChild` in https://httpd.apache.org/docs/current/mod/mpm_common.html [Apache MPM Common Directives].

- Aggiungi la direttiva `MaxConnectionsPerChild` con un valore di 1 nel file di configurazione di `apache2` - in *Debian* e derivati che si trova in `/etc/apache2/apache2.conf`; nella famiglia *CentOS*, in `/etc/httpd/conf/httpd.conf`. Non dimenticare di riavviare `apache2` affinché le modifiche abbiano effetto.

- Digita un comando che usi `watch`, `ps` e `grep` per le connessioni `apache2`.

- Ora apri un browser web o usa un browser a riga di comando come `lynx` per stabilire una connessione al server web tramite il suo indirizzo IP. Cosa osservi nell'output di `watch`?

3. Come hai imparato, per impostazione predefinita `top` ordina le attività in base alla percentuale di utilizzo della CPU in ordine decrescente (i valori maggiori in alto). Questo comportamento

può essere modificato con i tasti interattivi M (memory usage), N (process unique identifier), T (running time) e P (percentage of CPU time). Tuttavia, puoi anche ordinare l'elenco delle attività a tuo piacimento avviando `top` con l'opzione `-o` (per maggiori informazioni, controlla la pagina `man` di `top`). Ora, esegui le seguenti attività:

- Avvia `top` in modo che le attività siano ordinate in base all'utilizzo della memoria:

- Verifica di aver digitato il comando corretto evidenziando la colonna della memoria:

4. `ps` ha anche un'opzione `-o` per specificare le colonne che vuoi mostrare. Esamina questa opzione ed esegui le seguenti attività:

- Avvia `ps` in modo che vengano mostrate solo le informazioni su *user*, *percentage of memory used*, *percentage of CPU time used* e *full command*:

- Ora, avvia `ps` in modo che le uniche informazioni visualizzate siano quelle dell'utente e il nome dei programmi che stanno utilizzando:

Sommario

In questa lezione hai imparato a conoscere *jobs* e *job control*. Fatti e concetti importanti da tenere a mente sono:

- I lavori (*jobs*) sono processi che vengono inviati in background.
- Oltre a un `_ ID processo_`, ai lavori viene assegnato anche un `_ ID lavoro_` quando vengono creati.
- Per controllare i lavori, è richiesta una specifica di lavoro (`jobspec`).
- I lavori possono essere portati in primo piano, inviati in secondo piano, sospesi e terminati (o *uccisi*).
- Un lavoro può essere scollegato dal terminale e dalla sessione in cui è stato creato.

Allo stesso modo, abbiamo anche discusso il concetto di *processi* e *monitoraggio del processo*. I concetti più rilevanti sono:

- I processi stanno eseguendo programmi.
- I processi possono essere monitorati.
- Diverse utilità ci consentono di scoprire l'*ID* dei processi e di inviare loro segnali.
- I segnali possono essere specificati per nome (es. `-SIGTERM`), numero (es. `-15`) o attraverso un opzione (es. `-S SIGTERM`).
- `top` e `ps` sono molto potenti quando si tratta di monitorare i processi. L'output del primo è dinamico e si aggiorna costantemente; `ps` rivela invece l'output staticamente.

Comandi utilizzati in questa lezione:

jobs

Visualizza i lavori attivi e il loro stato.

sleep

Ritardo per un periodo di tempo specifico.

fg

Porta il lavoro in primo piano.

bg

Sposta il lavoro in background.

kill

Termina il lavoro.

nohup

Scollega il lavoro dalla sessione/terminale.

exit

Esce dalla shell corrente.

tail

Visualizza le righe più recenti in un file.

watch

Esegue ripetutamente un comando.(ogni 2 secondi di default).

uptime

Visualizza per quanto tempo il sistema è in esecuzione, il numero di utenti correnti e il carico medio del sistema.

free

Visualizza l'utilizzo della memoria.

pgrep

Cerca l'ID del processo in base al nome.

pidof

Cerca l'ID del processo in base al nome.

pkill

Invia segnale al processo per nome.

killall

Elimina i processi per nome.

top

Visualizza i processi Linux.

ps

Visualizza un'istantanea dei processi in corso.

Risposte agli Esercizi Guidati

1. `oneko` è un simpatico programma che mostra un gatto che insegue il cursore del mouse. Se non è già installato nel sistema desktop, installalo utilizzando il gestore di pacchetti della tua distribuzione. Lo useremo per studiare il controllo dei job.

- Avvia il programma. Come si fa a farlo?

Digitando `oneko` all'interno di un terminale.

- Muovi il cursore del mouse per vedere come il gatto lo insegue. Ora sospendi il processo. Come si fa a farlo? Qual è l'output?

Premendo la combinazione di tasti `Ctrl + Z`:

```
[1]+  Stopped                  oneko
```

- Controlla quanti job hai attualmente. Cosa scrivi? Qual è l'output?

```
$ jobs
[1]+  Stopped                  oneko
```

- Ora invialo in background specificando il suo job ID. Qual è l'output? Come puoi sapere che il lavoro è in esecuzione in background?

```
$ bg %1
[1]+ oneko &
```

Il gatto si muove nuovamente.

- Infine, termina il job specificando il suo ID. Cosa scrivi?

```
$ kill %1
```

2. Scopri i PID di tutti i processi generati dal server web *Apache HTTPD* (`apache2`) con due diversi comandi:

```
$ pgrep apache2
```

0

```
$ pidof apache2
```

3. Termina tutti i processi `apache2` senza usare i loro PID e con due diversi comandi:

```
$ pkill apache2
```

0

```
$ killall apache2
```

4. Supponi di dover terminare tutte le istanze di `apache2` e non hai tempo per scoprire quali sono i loro PID. Come lo realizzeresti usando `kill` con il segnale predefinito `SIGTERM` in una riga:

```
$ kill $(pgrep apache2)
$ kill `pgrep apache2`
```

oppure

```
$ kill $(pidof apache2)
$ kill `pidof apache2`
```

NOTE

Poiché `SIGTERM` (15) è il segnale predefinito, non è necessario passare alcuna opzione a `kill`.

5. Avvia `top` e interagisci con esso eseguendo quanto segue:

- Mostra una vista della foresta dei processi:

Premi `V`.

- Mostra i percorsi completi dei processi che si differenziano tra spazio utente e spazio kernel:

Premi `c`.

6. Digita il comando `ps` per visualizzare tutti i processi avviati dall'utente *Apache HTTPD web server* (`wwwdata`):

- Utilizzando la sintassi BSD:

```
$ ps U www-data
```

- Utilizzando la sintassi UNIX:

```
$ ps -u www-data
```

- Utilizzando la sintassi GNU:

```
$ ps --user www-data
```

Risposte agli Esercizi Esplorativi

1. Il segnale `SIGHUP` può essere usato come un modo per riavviare alcuni demoni. Con il server web *Apache HTTPD* — per esempio — l'invio di `SIGHUP` al processo genitore (quello avviato da `init`) termina i suoi figli. Il genitore, tuttavia, rilegge i suoi file di configurazione, riapre i file di log e genera un nuovo set di figli. Eseguire le seguenti attività:

- Avviare il web server:

```
$ sudo systemctl start apache2
```

- Assicurati di conoscere il PID del processo genitore:

```
$ ps aux | grep apache2
```

Il processo genitore è quello avviato dall'utente `root`. Nel nostro caso quello con PID 1653.

- Riavvia il server web Apache HTTPD inviando il segnale `SIGHUP` al processo genitore:

```
$ kill -SIGHUP 1653
```

- Controlla che il genitore non sia stato terminato e che siano stati generati nuovi figli:

```
$ ps aux | grep apache2
```

Ora dovresti vedere il processo genitore `apache2` insieme a due nuovi processi figli.

2. Sebbene inizialmente statico, l'output di `ps` può essere *reso* dinamico combinando `ps` e `watch`. Monitoreremo il server web *Apache HTTPD* per nuove connessioni. Prima di eseguire le attività descritte di seguito si consiglia di leggere la descrizione della direttiva `MaxConnectionsPerChild` in https://httpd.apache.org/docs/current/mod/mpm_common.html [Apache MPM Common Directives].

- Aggiungi la direttiva `MaxConnectionsPerChild` con un valore di `1` nel file di configurazione di `apache2` - in *Debian* e derivati che si trova in `/etc/apache2/apache2.conf`; nella famiglia *CentOS*, in `/etc/httpd/conf/httpd.conf`. Non dimenticare di riavviare `apache2` affinché le modifiche abbiano effetto.

La riga da includere nel file di configurazione è `MaxConnectionsPerChild 1`. Un modo per riavviare il server web è tramite: `sudo systemctl restart apache2`.

- Digita un comando che usi `watch`, `ps` e `grep` per le connessioni `apache2`.

```
$ watch 'ps aux | grep apache2'
```

O

```
$ watch "ps aux | grep apache2"
```

- Ora apri un browser web o usa un browser a riga di comando come `lynx` per stabilire una connessione al server web tramite il suo indirizzo IP. Cosa osservi nell'output di `watch`?

Uno dei processi figlio di proprietà di `www-data` scompare.

3. Come hai imparato, per impostazione predefinita `top` ordina le attività in base alla percentuale di utilizzo della CPU in ordine decrescente (i valori maggiori in alto). Questo comportamento può essere modificato con i tasti interattivi M (memory usage), N (process unique identifier), T (running time) e P (percentage of CPU time). Tuttavia, puoi anche ordinare l'elenco delle attività a tuo piacimento avviando `top` con l'opzione `-o` (per maggiori informazioni, controlla la pagina `man` di `top`). Ora, esegui le seguenti attività:

- Avvia `top` in modo che le attività siano ordinate in base all'utilizzo della memoria:

```
$ top -o %MEM
```

- Verifica di aver digitato il comando corretto evidenziando la colonna della memoria:

Premi `x`.

4. `ps` ha anche un'opzione `o` per specificare le colonne che vuoi mostrare. Esamina questa opzione ed esegui le seguenti attività:

- Avvia `ps` in modo che vengano mostrate solo le informazioni su *user*, *percentage of memory used*, *percentage of CPU time used* e *full command*:

```
$ ps o user,%mem,%cpu,cmd
```

- Ora, avvia `ps` in modo che le uniche informazioni visualizzate siano quelle dell'utente e il nome dei programmi che stanno utilizzando:

```
$ ps o user,comm
```




103.5 Lezione 2

Certificazione:	LPIC-1
Versione:	5.0
Argomento:	103 Comandi GNU e Unix
Obiettivo:	103.5 Creare, controllare e terminare i processi
Lezione:	2 di 2

Introduzione

Gli strumenti e le utilità visti nella lezione precedente sono molto utili per il monitoraggio dei processi in generale. Tuttavia, un amministratore di sistema potrebbe dover fare un ulteriore passo avanti. In questa lezione discuteremo il concetto di *multiplexer* di terminale e impareremo a conoscere *GNU Screen* e *tmux* poiché — nonostante i moderni e ottimi emulatori di terminale di oggi — i multiplexer conservano ancora alcune funzionalità interessanti e potenti per un amministratore di sistema produttivo.

Caratteristiche dei Terminali Multiplexer

In elettronica, un multiplexer (o *mux*) è un dispositivo che consente di collegare più ingressi a una singola uscita. Pertanto, un multiplexer di terminale ci dà la possibilità di passare tra diversi ingressi come richiesto. Sebbene non siano esattamente la stessa cosa, `screen` e `tmux` condividono una serie di caratteristiche comuni:

- Qualsiasi invocazione riuscita risulterà in almeno una sessione che, a sua volta, includerà almeno una finestra. Le finestre contengono programmi.

- Le finestre possono essere suddivise in regioni o riquadri, il che può aumentare la produttività quando si lavora con vari programmi contemporaneamente.
- Facilità di controllo: per eseguire la maggior parte dei comandi, usano una combinazione di tasti - il cosiddetto *command prefix* o *command key* — seguito da un altro carattere.
- Le sessioni possono essere scollegate dal terminale corrente (ovvero, i programmi vengono inviati in background e continuano a essere eseguiti). Ciò garantisce l'esecuzione completa dei programmi, indipendentemente dal fatto che chiudiamo accidentalmente un terminale, si verifichi un blocco occasionale del terminale o persino la perdita di connessione remota.
- Connessioni via socket.
- Modalità *copia*.
- Sono altamente personalizzabili.

GNU Screen

All'inizio di Unix (anni '70 -'80) i computer erano fondamentalmente costituiti da terminali collegati a un computer centrale. Questo era tutto, niente finestre o schede. E questo è stato il motivo alla base della creazione di GNU Screen nel 1987: emulare più *schermi VT100* indipendenti su un singolo terminale fisico.

Finestre

GNU Screen viene richiamato semplicemente digitando `screen` nel terminale. Prima vedrai un messaggio di benvenuto:

```
GNU Screen version 4.05.00 (GNU) 10-Dec-16

Copyright (c) 2010 Juergen Weigert, Sadrul Habib Chowdhury
Copyright (c) 2008, 2009 Juergen Weigert, Michael Schroeder, Micah Cowan, Sadrul Habib Chowdhury
Copyright (c) 1993-2002, 2003, 2005, 2006, 2007 Juergen Weigert, Michael Schroeder
Copyright (c) 1987 Oliver Laumann
(...)
```

Premi Spazio o Invio per chiudere il messaggio e ti verrà mostrato un prompt dei comandi:

```
$
```

Può sembrare che non sia successo nulla, ma il fatto è che `screen` ha già creato e gestisce la sua

prima sessione. Il prefisso del comando dello schermo è `Ctrl` + `a`. Per vedere tutte le finestre nella parte inferiore del display del terminale, digita `Ctrl` + `a` - `w`:

```
0*$ bash
```

Eccola, la nostra unica finestra fino ad ora! Nota che il conteggio inizia da 0. Per creare un'altra finestra, digita `Ctrl` + `a` - `c`. Vedrai un nuovo messaggio. Iniziamo con `ps` in questa nuova finestra:

```
$ ps
PID TTY          TIME CMD
 974 pts/2        00:00:00 bash
 981 pts/2        00:00:00 ps
```

e digita `Ctrl` + `a` - `w` di nuovo:

```
0-$ bash 1*$ bash
```

Abbiamo ora le nostre due finestre (nota l'asterisco che indica quella che viene mostrata al momento). Tuttavia, poiché sono stati avviati con Bash, entrambi hanno lo stesso nome. Dato che abbiamo invocato `ps` nella finestra corrente, rinominiamola con lo stesso nome. Per questo, devi digitare `Ctrl` + `a` - `A` e scrivere il nome della nuova finestra (`ps`) quando richiesto:

```
Set window's title to: ps
```

Ora, creiamo un'altra finestra ma diamo un nome dall'inizio: `yetanotherwindow`. Questo viene fatto invocando `screen` con l'opzione `-t`:

```
$ screen -t yetanotherwindow
```

Puoi spostarti tra le finestre in diversi modi:

- Utilizzando `Ctrl` + `a` - `n` (vai alla finestra *successiva*) e `Ctrl` + `a` - `p` (vai alla finestra *precedente*).
- Utilizzando `Ctrl` + `a` - `_ number_` (vai alla finestra numero *number*).
- Usando `Ctrl` + `a` - `"` per vedere un elenco di tutte le finestre. Puoi muoverti su e giù con i tasti freccia e selezionare quella che vuoi con Invio:

```
Num Name
```

```
Flags
```

```

0 bash                                $
1 ps                                  $
2 yetanotherwindow

```

Mentre si lavora con le finestre è importante ricordare quanto segue:

- Le finestre eseguono i propri programmi in modo completamente indipendente le une dalle altre.
- I programmi continueranno a funzionare anche se la loro finestra non è visibile (anche quando la sessione dello schermo è staccata come vedremo a breve).

Per rimuovere una finestra, è sufficiente terminare il programma in esecuzione in essa (una volta rimossa l'ultima finestra, `screen` terminerà automaticamente). In alternativa, usa `Ctrl` + `a` - `k` mentre sei nella finestra che vuoi rimuovere; ti verrà chiesta conferma:

```
Really kill this window [y/n]
```

```
Window 0 (bash) killed.
```

Regioni

`screen` può dividere lo schermo di un terminale in più regioni in cui ospitare le finestre. Queste divisioni possono essere orizzontali (`Ctrl` + `a` - `S`) o verticali (`Ctrl` + `a` - `|`).

L'unica cosa che la nuova regione mostrerà è solo `--` in basso, il che significa che è vuota:

```
1 ps                                --
```

Per passare alla nuova regione, digita `Ctrl` + `a` - `Tab`. Ora puoi aggiungere una finestra con uno qualsiasi dei metodi che abbiamo già visto, per esempio: `Ctrl` + `a` - `2`. Ora il `--` dovrebbe essere diventato `2 yetanotherwindow`

```

$ ps                                $
  PID TTY          TIME CMD
 1020 pts/2    00:00:00 bash
 1033 pts/2    00:00:00 ps
$ screen -t yetanotherwindow

```

1 ps

2 yetanotherwindow

Aspetti importanti da tenere a mente quando si lavora con le regioni sono:

- Ci si sposta tra le regioni digitando `Ctrl + a - Tab`.
- Puoi terminare tutte le regioni tranne quella corrente con `Ctrl + a - Q`.
- Puoi terminare la regione corrente con `Ctrl + a - X`.
- La chiusura di una regione non termina la sua finestra associata.

Sessioni

Finora abbiamo giocato con poche finestre e regioni, ma tutte appartenenti alla stessa e unica sessione. È ora di iniziare a giocare con le sessioni. Per vedere un elenco di tutte le sessioni, digita `screen -list` o `screen -ls`:

```
$ screen -list
There is a screen on:
      1037.pts-0.debian      (08/24/19 13:53:35)      (Attached)
1 Socket in /run/screen/S-carol.
```

Al momento, questa è la nostra unica sessione:

PID

1037

Name

`pts-0.debian` (indica il terminale — nel nostro caso uno *pseudo terminale* — e l'hostname).

Status

Attached

Creiamo una nuova sessione dandole un nome più descrittivo:

```
$ screen -S "second session"
```

Il terminale verrà ripulito e verrà visualizzato un nuovo prompt. Puoi controllare ancora una volta le sessioni:

```
$ screen -ls
There are screens on:
    1090.second session      (08/24/19 14:38:35)    (Attached)
    1037.pts-0.debian        (08/24/19 13:53:36)    (Attached)
2 Sockets in /run/screen/S-carol.
```

Per terminare una sessione, esci da tutte le sue finestre o digita semplicemente il comando `screen -S SESSION-PID -X quit` (puoi anche fornire il nome della sessione). Liberiamoci della nostra prima sessione:

```
$ screen -S 1037 -X quit
```

Verrai rimandato al prompt del tuo terminale al di fuori di `screen`. Ma ricorda, la nostra seconda sessione è ancora operativa:

```
$ screen -ls
There is a screen on:
    1090.second session (08/24/19 14:38:35) (Detached)
1 Socket in /run/screen/S-carol.
```

Tuttavia, poiché abbiamo terminato la sua sessione padre, viene assegnata una nuova etichetta: `Detached`.

Distacco della Sessione

Per una serie di motivi potresti voler scollegare una sessione dello schermo dal suo terminale:

- Per lasciare che il tuo computer al lavoro prosegua nelle sue attività e connettersi in seguito da remoto.
- Per condividere una sessione con altri utenti.

Scollega una sessione con la combinazione di tasti `Ctrl + a -d`. Verrai reindirizzato al tuo terminale:

```
[detached from 1090.second session]
$
```

Per collegarti di nuovo alla sessione, usa il comando `screen -r SESSION-PID`. In alternativa, puoi usare `SESSION-NAME` come abbiamo visto sopra. Se c'è solo una sessione distaccata, nessuna delle due è obbligatoria:

```
$ screen -r
```

Questo comando è sufficiente per ricollegarsi alla nostra seconda sessione:

```
$ screen -ls
There is a screen on:
      1090.second session      (08/24/19 14:38:35)      (Attached)
1 Socket in /run/screen/S-carol.
```

Opzioni importanti per il ricollegamento della sessione:

-d -r

Ricollegare una sessione e — se necessario — scollegarla prima.

-d -R

Come **-d -r** ma **screen** creerà prima la sessione se non esiste.

-d -RR

Uguale a **-d -R**. Tuttavia, utilizza la prima sessione se più di una è disponibile.

-D -r

Ricollega una sessione. Se necessario, scollegandola e disconnettendola prima da remoto.

-D -R

Se una sessione è in esecuzione, ricollega (scollegandola e disconnettendola da remoto se necessario). Se non era in esecuzione, ne crea una nuova e avvisa l'utente.

-D -RR

Come per **-D -R**

-d -m

Avvia **screen** in *detached mode*. Questo crea una nuova sessione ma non si collega ad essa. Questo è utile per gli script di avvio del sistema.

-D -m

Uguale a **-d -m**, ma non esegue il *fork* di un nuovo processo. Il comando termina se la sessione termina.

Leggi le pagine di manuale di **screen** per scoprire altre opzioni.

Copia e Incolla: Scrollback Mode

GNU Screen implementa una funzionalità di copia o *scrollback mode*. Una volta attivata, puoi spostare il cursore nella finestra corrente e muoverti nel testo con i tasti freccia. Puoi contrassegnare il testo e copiarlo tra le finestre. I passaggi da seguire sono:

1. Accedi alla copia/scrollback mode: `Ctrl + a-[`.
2. Spostati all'inizio della parte di testo che desideri copiare utilizzando i tasti freccia.
3. Segna l'inizio della parte di testo che desideri copiare: Premi la barra spaziatrice.
4. Spostati alla fine della parte di testo che desideri copiare utilizzando i tasti freccia.
5. Contrassegna la fine del pezzo di testo che desideri copiare: Premi la barra spaziatrice.
6. Vai alla finestra di tua scelta e incolla quanto precedentemente copiato: `Ctrl + a-]`.

Personalizzazione di screen

Il file di configurazione a livello di sistema per screen è `/etc/screenrc`. In alternativa, può essere usato a livello utente `~/.screenrc`. Il file include quattro sezioni di configurazione principali:

SCREEN SETTINGS

Puoi definire le impostazioni generali specificando la *direttiva* seguita da uno spazio e il *valore* come in: `defscrollback 1024`.

SCREEN KEYBINDINGS

Questa sezione è piuttosto interessante in quanto consente di ridefinire le associazioni di tasti che forse interferiscono con l'uso quotidiano del terminale. Usa la parola chiave `bind` seguita da uno spazio, il carattere da usare dopo il prefisso del comando, un altro spazio e il comando come in: `bind l kill` (questa impostazione cambierà il modo predefinito di terminare una finestra in `Ctrl + a-l`).

Per visualizzare tutti i collegamenti dello schermo, digita `Ctrl + a-?` oppure consulta la pagina di manuale.

TIP

Ovviamente puoi anche modificare il prefisso del comando stesso. Per esempio, per passare da `Ctrl + a` a `kbd: [Ctrl+b]`, aggiungi semplicemente questa riga: `escape ^ Bb`.

TERMINAL SETTINGS

Questa sezione include le impostazioni relative alle dimensioni della finestra del terminale e ai

buffer. Per abilitare la *non-blocking mode* per gestire meglio le connessioni ssh instabili, per esempio, viene utilizzata la seguente configurazione: `defnonblock 5`.

STARTUP SCREENS

Puoi includere comandi per avere vari programmi in esecuzione all'avvio di `screen`; per esempio: `screen -t top top` (lo schermo aprirà una finestra chiamata `top` con `top` all'interno).

tmux

`tmux` è stato rilasciato nel 2007. Sebbene molto simile a `screen`, include alcune differenze:

- Modello client-server: il server fornisce una serie di sessioni, ciascuna delle quali può avere un numero di finestre a essa collegate che possono, a loro volta, essere condivise da diversi client.
- Selezione interattiva delle sessioni, finestre e client tramite menu.
- La stessa finestra può essere collegata a più sessioni.
- Disponibilità dei layout dei tasti *vim* e *Emacs*.
- Supporto terminale a UTF-8 e 256 colori.

Finestre

`tmux` può essere invocato semplicemente digitando `tmux` al prompt dei comandi. Ti verrà mostrato un prompt della shell e una barra di stato nella parte inferiore della finestra:

```
[0] 0: bash*
```

```
"debian" 18:53 27-Aug-19
```

Oltre all' `hostname`, l'ora e la data, la barra di stato fornisce le seguenti informazioni:

Nome della sessione

```
[0]
```

Numero della finestra

```
0:
```

Window name

`bash *`. Di default questo è il nome del programma in esecuzione all'interno della finestra e, a differenza di `screen`, `tmux` lo aggiornerà automaticamente per riflettere il programma in esecuzione. Notare l'asterisco che indica che questa è la finestra visibile corrente.

È possibile assegnare nomi di sessioni e finestre quando si invoca `tmux`:

```
$ tmux new -s "LPI" -n "Window zero"
```

La barra di stato cambierà di conseguenza:

```
[LPI] 0:Window zero*                                "debian" 19:01 27-Aug-19
```

Il prefisso del comando di `tmux` è `Ctrl + b`. Per creare una nuova finestra, digita semplicemente `Ctrl + b -c`; verrai reindirizzato a un nuovo prompt e la barra di stato rifletterà la nuova finestra:

```
[LPI] 0:Window zero- 1:bash*                          "debian" 19:02 27-Aug-19
```

Poiché Bash è la shell sottostante, alla nuova finestra viene assegnato quel nome per impostazione predefinita. Avvia `top` e guarda come il nome cambia in `top`:

```
[LPI] 0:Window zero- 1:top*                            "debian" 19:03 27-Aug-19
```

In ogni caso, puoi rinominare una finestra con `Ctrl + b -r`. Quando richiesto, fornire il nuovo nome e premere Invio:

```
(rename-window) Window one
```

È possibile visualizzare tutte le finestre, per selezionarne una, con `Ctrl + b -w` (utilizzare i tasti freccia per spostarsi su e giù e `invio` per sceglierne una):

```
(0) 0: Window zero- "debian"
(1) 1: Window one*  "debian"
```

In modo simile a `screen`, possiamo muoverci da una finestra all'altra con:

`Ctrl + b -n`

vai alla finestra *successiva*.

Ctrl + **b-p**vai alla finestra *precedente*.**Ctrl** + **b-number**vai alla finestra numero *number*..Per sbarazzarti di una finestra, usa **Ctrl** + **b-&**. Ti verrà chiesta conferma:

kill-window Window one? (y/n)

Altri comandi di finestra interessanti includono:

Ctrl + **b-f**

trova la finestra per nome.

Ctrl + **b-.**

cambia il numero di indice della finestra.

Per leggere l'intera lista dei comandi, consultare la pagina di manuale.

Riquadri

La funzione di suddivisione della finestra di `screen` è presente anche in `tmux`. Tuttavia, le divisioni risultanti non sono chiamate *regioni* (*regions*) ma *riquadri* (*panes*). La differenza più importante tra regioni e riquadri è che questi ultimi sono pseudo-terminali completi collegati a una finestra. Ciò significa che terminare un riquadro terminerà anche il suo pseudo-terminale e tutti i programmi associati in esecuzione all'interno.

Per dividere una finestra orizzontalmente, usiamo **Ctrl** + **b-"**:

```
Tasks:  93 total,   1 running,  92 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.0 us,   0.0 sy,   0.0 ni,100.0 id,   0.0 wa,   0.0 hi,   0.0 si,   0.0 st
KiB Mem : 4050960 total, 3730920 free,  114880 used,  205160 buff/cache
KiB Swap: 4192252 total, 4192252 free,    0 used. 3716004 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
1340	carol	20	0	44876	3400	2800	R	0.3	0.1	0:00.24	top
1	root	20	0	139088	6988	5264	S	0.0	0.2	0:00.50	systemd
2	root	20	0	0	0	0	S	0.0	0.0	0:00.00	kthreadd
3	root	20	0	0	0	0	S	0.0	0.0	0:00.04	ksoftirqd/0
4	root	20	0	0	0	0	S	0.0	0.0	0:01.62	kworker/0:0

```

 5 root      0 -20      0      0      0 S  0.0  0.0   0:00.00 kworker/0:0H
 7 root      20   0      0      0      0 S  0.0  0.0   0:00.06 rcu_sched
 8 root      20   0      0      0      0 S  0.0  0.0   0:00.00 rcu_bh
 9 root      rt   0      0      0      0 S  0.0  0.0   0:00.00 migration/0
10 root      0 -20      0      0      0 S  0.0  0.0   0:00.00 lru-add-drain
11 root      rt   0      0      0      0 S  0.0  0.0   0:00.01 watchdog/0
12 root      20   0      0      0      0 S  0.0  0.0   0:00.00 cpuhp/0
$

```

```
$
```

```
[LPI] 0:Window zero- 1:Window one*
Aug-19
```

```
"debian" 19:05 27-
```

Per dividerlo verticalmente, usa `Ctrl + b-%`:

```

 1 root      20   0 139088  6988  5264 S  0.0  0.2   0:00.50 systemd | $
 2 root      20   0      0      0      0 S  0.0  0.0   0:00.00 kthreadd |
 3 root      20   0      0      0      0 S  0.0  0.0   0:00.04 ksoftirqd/0 |
 4 root      20   0      0      0      0 S  0.0  0.0   0:01.62 kworker/0:0 |
 5 root      0 -20      0      0      0 S  0.0  0.0   0:00.00 kworker/0:0H |
 7 root      20   0      0      0      0 S  0.0  0.0   0:00.06 rcu_sched |
 8 root      20   0      0      0      0 S  0.0  0.0   0:00.00 rcu_bh |
 9 root      rt   0      0      0      0 S  0.0  0.0   0:00.00 migration/0 |
10 root      0 -20      0      0      0 S  0.0  0.0   0:00.00 lru-add-drai |
n
11 root      rt   0      0      0      0 S  0.0  0.0   0:00.01 watchdog/0 |

```

```

12 root      20   0   0   0   0 S  0.0  0.0  0:00.00 cpuhp/0
$
_____
$

[LPI] 0:Window zero- 1:Window one*
Aug-19                                     "debian" 19:05 27-
```

Per eliminare il riquadro corrente (insieme al suo pseudo-terminale in esecuzione al suo interno, insieme a tutti i programmi associati), usa `Ctrl + b - x`. Ti verrà richiesta una conferma nella barra di stato:

```
kill-pane 1? (y/n)
```

Comandi importanti del riquadro:

`Ctrl + b - ↑, ↓, ←, →`

spostarsi tra riquadri.

`Ctrl + b - ;`

spostarsi nell'ultimo riquadro attivo.

`Ctrl + b - Ctrl + arrow key`

ridimensionare il riquadro di una riga.

`Ctrl + b - Alt + arrow key`

ridimensionare il riquadro di cinque righe.

`Ctrl + b - {`

Scambia i riquadri (dal corrente al precedente).

`Ctrl + b - }`

Scambia i riquadri (dal corrente al successivo).

`Ctrl + b - z`

zoom in/out del riquadro.

`Ctrl + b - t`

`tmux` mostra un orologio all'interno del riquadro (terminalo premendo `q`).

`Ctrl + b - !`

trasforma il riquadro in una finestra.

Per leggere l'intera lista dei comandi, consultare la pagina di manuale.

Sessioni

Per elencare le sessioni in `tmux` puoi usare `Ctrl + b - s`:

```
(0) + LPI: 2 windows (attached)
```

In alternativa, puoi usare il comando `tmux ls`:

```
$ tmux ls
LPI: 2 windows (created Tue Aug 27 19:01:49 2019) [158x39] (attached)
```

C'è solo una sessione (LPI) che include due finestre. Creiamo una nuova sessione dall'interno della nostra sessione corrente. Questo può essere ottenuto usando `Ctrl + b`, digita `:new` al prompt, quindi premi *Invio*. Verrai indirizzato alla nuova sessione come si può osservare nella barra di stato:

```
[2] 0:bash* "debian" 19:15 27-Aug-19
```

Per impostazione predefinita, `tmux` ha chiamato la sessione 2. Per rinominarlo, usa `Ctrl + b - $`.

Quando richiesto, fornire il nuovo nome e premere *Invio*:

```
(rename-session) Second Session
```

Puoi cambiare sessione con `Ctrl + b - s` (usa i tasti freccia e `enter`):

```
(0) + LPI: 2 windows
(1) + Second Session: 1 windows (attached)
```

Per terminare una sessione, puoi usare il comando `tmux kill-session -t SESSION-NAME`. Se digiti il comando dall'interno della sessione corrente, verrai portato fuori da `tmux` e tornerai alla sessione iniziale del terminale:

```
$ tmux kill-session -t "Second Session"
[exited]
$
```

Sganciamento dalla Sessione

Terminando `Second Session` siamo stati buttati fuori da `tmux`. Tuttavia, abbiamo ancora una sessione attiva. Chiedi a `tmux` un elenco di sessioni e lo troverai sicuramente lì:

```
$ tmux ls
LPI: 2 windows (created Tue Aug 27 19:01:49 2019) [158x39]
```

Tuttavia questa sessione è scollegata dal suo terminale. Possiamo collegarlo con `tmux attach -t SESSION-NAME` (`attach` può essere sostituito da `at` o — semplicemente — `a`). Quando è presente una sola sessione, la specifica del nome è facoltativa:

```
$ tmux a
```

Ora sei tornato nella tua sessione; per sganciarti da essa, premi `Ctrl + b - d`:

```
[detached (from session LPI)]
$
```

TIP

La stessa sessione può essere collegata a più di un terminale. Se vuoi allegare una

sessione assicurandoti che sia prima scollegata da qualsiasi altro terminale, usa l'opzione `-d: tmux attach -d -t SESSION-NAME`.

Comandi importanti per il collegamento/scollegamento della sessione:

`Ctrl + b -D`

seleziona quale client scollegare.

`Ctrl + b -r`

aggiorna il terminale del client.

Per leggere l'intera lista dei comandi, consultare la pagina di manuale.

Copia e Incolla: Scrollback Mode

Anche `tmux` dispone della modalità di copia fondamentalmente allo stesso modo di `screen` (ricorda di usare il prefisso del comando di `tmux` e non quello di `screen`!). L'unica differenza dal punto di vista dei comandi è che usi `Ctrl + Space` per contrassegnare l'inizio della selezione e `Alt + w` per copiare il testo selezionato.

Personalizzazione di tmux

I file di configurazione per `tmux` si trovano tipicamente in `/etc/tmux.conf` e `~/.tmux.conf`. Quando viene avviato, `tmux` fa riferimento a questi file se esistono. C'è anche la possibilità di avviare `tmux` con l'opzione `-f` per fornire un file di configurazione alternativo. Puoi trovare un esempio di file di configurazione `tmux` situato in `/usr/share/doc/tmux/example_tmux.conf`. Il livello di personalizzazione che puoi ottenere è davvero alto. Alcune delle cose che puoi fare includono:

- Cambia la chiave del prefisso

```
# Change the prefix key to C-a
set -g prefix C-a
unbind C-b
bind C-a send-prefix
```

- Imposta associazioni di tasti aggiuntive per finestre superiori a 9

```
# Some extra key bindings to select higher numbered windows
bind F1 selectw -t:10
bind F2 selectw -t:11
```



```
bind F3 selectw -t:12
```

Per un elenco completo di tutte le associazioni, digita `Ctrl + b - ?` (Premi `q` per uscire) o consulta la pagina di manuale.

Esercizi Guidati

1. Indica se le seguenti affermazioni/caratteristiche corrispondono a GNU Screen, tmux o a entrambi:

Affermazione/Caratteristica	GNU Screen	tmux
Il prefisso del comando predefinito è <code>Ctrl</code> + <code>a</code>		
Modello Client-Server		
I riquadri sono pseudo-terminali		
La terminazione di una regione non termina le finestre associate		
Le sessioni includono le finestre		
Le sessioni possono essere scollegate		

2. Installa GNU Screen sul tuo computer (nome del pacchetto: `screen`) e completa le seguenti attività:

- Avvia il programma. Che comando usi?

- Avvia `top`:

- Utilizzando il prefisso chiave di `screen`, apri una nuova finestra; quindi, apri `/etc/screenrc` usando `vi`:

- Elenca le finestre nella parte inferiore dello schermo:

- Cambia il nome della finestra corrente in `vi`:

- Cambia il nome della finestra rimanente in `top`. Per fare ciò, prima visualizza un elenco di tutte le finestre in modo da poterti spostare su e giù e selezionare quella giusta:

- Verifica che i nomi siano cambiati visualizzando nuovamente i nomi delle finestre nella parte inferiore dello schermo:

--

- Ora, scollega la sessione e chiedi a `screen` di crearne una nuova chiamata `ssh`:

- Scollegati anche da `ssh` e fai visualizzare a `screen` l'elenco delle sessioni:

- Ora, collega alla prima sessione usando il suo PID:

--

- Dovresti essere di nuovo alla finestra che mostra `top`. Dividi la finestra orizzontalmente e passa alla nuova regione vuota:

- Fai in modo che `screen` elenchi tutte le finestre e seleziona `vi` da visualizzare nella nuova regione vuota:

--

- Ora, dividi la regione corrente verticalmente, spostati nella regione vuota appena creata e associala a una nuova finestra:

- Termina tutte le regioni tranne quella attuale (ricorda, anche se termini le regioni, le finestre sono ancora attive). Quindi, esci da tutte le finestre della sessione corrente fino a quando la sessione stessa non viene terminata:

- Infine, fai in modo che `screen` elenchi le sue sessioni ancora una volta, termina la restante sessione `ssh` tramite PID e controlla che non ci siano effettivamente sessioni rimaste:

3. Installa `tmux` sul tuo computer (nome del pacchetto: `tmux`) e completa le seguenti attività:

- Avvia il programma. Che comando usi?

- Avvia `top` (nota come — in un paio di secondi — il nome della finestra cambia in `top` nella barra di stato):

- Usando i comandi di `tmux`, apri una nuova finestra; quindi, crea `~/.tmux.conf` usando `nano`:

- Dividi la finestra verticalmente e riduci alcune volte le dimensioni del riquadro appena creato:

- Ora cambia il nome della finestra corrente in `text editing`; quindi, fai visualizzare a `tmux` un elenco con tutte le sue sessioni:

- Passa alla finestra che esegue `top` e torna a quella corrente utilizzando la stessa combinazione di tasti:

- Scollegati dalla sessione corrente e creane una nuova il cui nome è `ssh` e il nome della sua finestra è `ssh window`:

- Scollegati anche dalla sessione `ssh` e fai visualizzare nuovamente a `tmux` l'elenco delle sessioni:

NOTE

Da questo punto in poi l'esercizio richiede che tu usi una macchina *remota* per le connessioni `ssh` al tuo host locale (una macchina virtuale è perfettamente valida e può rivelarsi davvero pratica). Assicurati di avere `openssh-server` installato e in esecuzione sulla tua macchina locale e che

almeno `openssh-client` sia installato sulla macchina remota.

- Ora, avvia una macchina remota e connettiti tramite `ssh` al tuo host locale. Una volta stabilita la connessione, controlla le sessioni `tmux`:

- Sull'host remoto, agganciati alla sessione denominata `ssh`:

- Tornati sulla macchina locale, connettiti alla sessione `ssh` assicurandoti che la connessione all'host remoto sia terminata prima:

- Visualizza tutte le sessioni per la selezione e vai alla tua prima sessione (`[0]`). Una volta lì, termina la sessione `ssh`:

- Infine, scollegati dalla sessione corrente e terminala attraverso il suo nome:

Esercizi Esplorativi

1. Sia `screen` che `tmux` possono entrare in modalità riga di comando tramite *command prefix* + `:` (abbiamo già visto un breve esempio con `tmux`). Esegui alcune ricerche e le seguenti attività in modalità riga di comando:

- Fai in modo che `screen` entri in modalità copia:

- Fai in modo che `tmux` rinomini la finestra corrente:

- Fai in modo che `screen` chiuda tutte le finestre e termini la sessione:

- Fai in modo che `tmux` divida un riquadro in due:

- Fai in modo che `tmux` termini la finestra corrente:

2. Quando entri in modalità copia in `screen` non solo puoi usare i tasti freccia e `PgUP` o `PgDown` per navigare nella finestra corrente e nel buffer di scorrimento. C'è anche la possibilità di utilizzare un editor a schermo intero simile a `vi`. Utilizzando questo editor, eseguire le seguenti attività:

- Echo `supercalifragilisticexpialidocious` nel tuo terminale `screen`:

- Ora, copia i cinque caratteri consecutivi (da sinistra a destra) nella riga sopra il cursore:

- Infine, incolla la selezione (`stice`) al prompt dei comandi:

3. Supponi di voler condividere una sessione `tmux` (`our_session`) con un altro utente. Hai creato il socket (`/tmp/our_socket`) con i permessi giusti in modo che, sia tu sia l'altro utente, possiate leggere e scrivere. Quali altre due condizioni dovrebbero essere soddisfatte affinché il secondo utente possa allegare con successo la sessione tramite `tmux -S /tmp/our_socket a -t`

our_session?

Sommario

In questa lezione hai imparato a conoscere in generale i *terminali multiplexer* e GNU Screen e tmux in particolare. I concetti importanti da ricordare includono:

- Prefisso del comando: `screen` usa `Ctrl` + `a` + *character*; `tmux`, `Ctrl` + `b` + *character*.
- Struttura di sessioni, finestre e divisioni di finestre (regioni o riquadri).
- Modalità copia.
- Distacco di sessione: una delle caratteristiche più potenti dei multiplexer.

Comandi utilizzati in questa lezione:

screen

Avvia una sessione `screen`.

tmux

Avvia una sessione `tmux`.

Risposte agli Esercizi Guidati

1. Indica se le seguenti affermazioni/caratteristiche corrispondono a GNU Screen, tmux o a entrambi:

Affermazione/Caratteristica	GNU Screen	tmux
Il prefisso del comando predefinito è <code>Ctrl + a</code>	x	
Modello Client-Server		x
I riquadri sono pseudo-terminali		x
La terminazione di una regione non termina le finestre associate	x	
Le sessioni includono le finestre	x	x
Le sessioni possono essere scollegate	x	x

2. Installa GNU Screen sul tuo computer (nome del pacchetto: `screen`) e completa le seguenti attività:

- Avvia il programma. Che comando usi?

`screen`

- Avvia `top`:

`top`

- Utilizzando il prefisso chiave di `screen`, apri una nuova finestra; quindi, apri `/etc/screenrc` usando `vi`:

`Ctrl + a - c`

`sudo vi /etc/screenrc`

- Elenca le finestre nella parte inferiore dello schermo:

`Ctrl + a - w`

- Cambia il nome della finestra corrente in `vi`:

`Ctrl` + `a-A`. Then we have to type `vi` and press `enter`.

- Cambia il nome della finestra rimanente in `top`. Per fare ciò, prima visualizza un elenco di tutte le finestre in modo da poterti spostare su e giù e selezionare quella giusta:

Per prima cosa digitiamo `Ctrl` + `a-"`. Quindi usiamo i tasti freccia per contrassegnare quello che indica `0` `bash` e premiamo `enter`. Infine, digitiamo `Ctrl` + `a-A`, digita `top` e premi `enter`.

- Verifica che i nomi siano cambiati visualizzando nuovamente i nomi delle finestre nella parte inferiore dello schermo:

`Ctrl` + `a-w`

- Ora, scollega la sessione e chiedi a `screen` di crearne una nuova chiamata `ssh`:

`Ctrl` + `a-d` `screen -S "ssh"` e premi `enter`.

- Scollegati anche da `ssh` e fai visualizzare a `screen` l'elenco delle sessioni:

`Ctrl` + `a-d` `screen -list` o `screen -ls`.

- Ora, collega alla prima sessione usando il suo PID:

`screen -r PID-OF-SESSION`

- Dovresti essere di nuovo alla finestra che mostra `top`. Dividi la finestra orizzontalmente e passa alla nuova regione vuota:

`Ctrl` + `a-S`

`Ctrl` + `a-Tab`

- Fai in modo che `screen` elenchi tutte le finestre e seleziona `vi` da visualizzare nella nuova regione vuota:

Usiamo `Ctrl` + `a-"` per visualizzare tutte le finestre per la selezione, contrassegnare `vi` e premere `enter`.

- Ora, dividi la regione corrente verticalmente, spostati nella regione vuota appena creata e associala a una nuova finestra:

`Ctrl` + `a-|`

`Ctrl + a - Tab``Ctrl + a - c`

- Termina tutte le regioni tranne quella attuale (ricorda, anche se termini le regioni, le finestre sono ancora attive). Quindi, esci da tutte le finestre della sessione corrente fino a quando la sessione stessa non viene terminata:

`Ctrl + a - q`. Exit (per uscire da Bash). `Shift + :`, quindi digitiamo `quit` e premiamo `enter` (per uscire da vi). Dopodiché, digitiamo `exit` (per uscire dalla shell Bash sottostante) `q` (per terminare `top`); quindi digitiamo `exit` (per uscire dalla shell Bash sottostante).

- Infine, fai in modo che `screen` elenchi le sue sessioni ancora una volta, termina la restante sessione `ssh` tramite PID e controlla che non ci siano effettivamente sessioni rimaste:

`screen -list o screen -ls``screen -S PID-OF-SESSION -X quit``screen -list o screen -ls`

3. Installa `tmux` sul tuo computer (nome del pacchetto: `tmux`) e completa le seguenti attività:

- Avvia il programma. Che comando usi?

`tmux`

- Avvia `top` (nota come — in un paio di secondi — il nome della finestra cambia in `top` nella barra di stato):

`top`

- Usando i comandi di `tmux`, apri una nuova finestra; quindi, crea `~/ .tmux.conf` usando `nano`:

`Ctrl + b - c nano ~/ .tmux.conf`

- Dividi la finestra verticalmente e riduci alcune volte le dimensioni del riquadro appena creato:

`Ctrl + b - "``Ctrl + b - Ctrl + i`

- Ora cambia il nome della finestra corrente in `text editing`; quindi, fai visualizzare a `tmux`

un elenco con tutte le sue sessioni:

`Ctrl` + `b`-, Poi forniamo il nuovo nome e premiamo `enter`, `Ctrl` + `b`-s o `tmux ls`.

- Passa alla finestra che esegue `top` e torna a quella corrente utilizzando la stessa combinazione di tasti:

`Ctrl` + `b`-n o `Ctrl` + `b`-p

- Scollega dalla sessione corrente e creane una nuova il cui nome è `ssh` e il nome della sua finestra è `ssh window`:

`Ctrl` + `b`-d `tmux new -s "ssh" -n "ssh window"`

- Scollegati anche dalla sessione `ssh` e fai visualizzare nuovamente a `tmux` l'elenco delle sessioni:

`Ctrl` + `b`-d `tmux ls`

NOTE

Da questo punto in poi l'esercizio richiede che tu usi una macchina *remota* per le connessioni `ssh` al tuo host locale (una macchina virtuale è perfettamente valida e può rivelarsi davvero pratica). Assicurati di avere `openssh-server` installato e in esecuzione sulla tua macchina locale e che almeno `openssh-client` sia installato sulla macchina remota.

- Ora, avvia una macchina remota e connettiti tramite `ssh` al tuo host locale. Una volta stabilita la connessione, controlla le sessioni `tmux`:

Sull'host remoto: `ssh local-username@local-ipaddress`. Una volta connesso al sistema locale: `tmux ls`.

- Sull'host remoto, agganciati alla sessione denominata `ssh`:

`tmux a -t ssh` (a può essere sostituito da `at` o `attach`).

- Tornati sulla macchina locale, connettiti alla sessione `ssh` assicurandoti che la connessione all'host remoto sia terminata prima:

`tmux a -d -t ssh` (a può essere sostituito da `at` o `attach`).

- Visualizza tutte le sessioni per la selezione e vai alla tua prima sessione (`[0]`). Una volta lì, termina la sessione `ssh`:

Digitiamo `Ctrl` + `b`-s, utilizza le frecce per selezionare la sessione `0` e premi `enter` `tmux kill-`

```
session -t ssh.
```

- Infine, scollegati dalla sessione corrente e terminala attraverso il suo nome:

```
ctrl + b -d tmux kill-session -t 0.
```

Risposte agli Esercizi Esplorativi

1. Sia `screen` che `tmux` possono entrare in modalità riga di comando tramite *command prefix* + `:` (abbiamo già visto un breve esempio con `tmux`). Esegui alcune ricerche e le seguenti attività in modalità riga di comando:

- Fai in modo che `screen` entri in modalità copia:

`Ctrl` + `a` - `:` — quindi, digita `copy`.

- Fai in modo che `tmux` rinomini la finestra corrente:

`Ctrl` + `b` - `:` — quindi, digita `rename-window`.

- Fai in modo che `screen` chiuda tutte le finestre e termini la sessione:

`Ctrl` + `a` - `:` — quindi, digita `quit`.

- Fai in modo che `tmux` divida un riquadro in due:

`Ctrl` + `b` - `:` — quindi, digita `split-window`.

- Fai in modo che `tmux` termini la finestra corrente:

`Ctrl` + `b` - `:` — quindi, digita `kill-window`.

2. Quando entri in modalità copia in `screen` non solo puoi usare i tasti freccia e `PgUP` o `PgDown` per navigare nella finestra corrente e nel buffer di scorrimento. C'è anche la possibilità di utilizzare un editor a schermo intero simile a `vi`. Utilizzando questo editor, esegui le seguenti attività:

- Echo `supercalifragilisticexpialidocious` nel tuo terminale `screen`:

```
echo supercalifragilisticexpialidocious
```

- Ora, copia i cinque caratteri consecutivi (da sinistra a destra) nella riga sopra il cursore:

Entra in modalità copia: `Ctrl` + `a` - `[` o `Ctrl` + `a` - `:` e poi digita `copy`. Quindi spostati sulla riga sopra usando `k` e premi `enter` per contrassegnare l'inizio della selezione. Infine, avanza di quattro caratteri usando `l` e premi di nuovo `enter` per contrassegnare la fine della selezione.

- Infine, incolla la selezione (`stice`) al prompt dei comandi:

`Ctrl` + `a` - `J`

3. Supponi di voler condividere una sessione `tmux` (`our_session`) con un altro utente. Hai creato

il socket (/tmp/our_socket) con i permessi giusti in modo che, sia tu sia l'altro utente, possiate leggere e scrivere. Quali altre due condizioni dovrebbero essere soddisfatte affinché il secondo utente possa allegare con successo la sessione tramite `tmux -S /tmp/our_socket a -t our_session`?

Entrambi gli utenti devono avere un gruppo in comune, per esempio `multiplexer`. Quindi, dobbiamo cambiare anche il socket in quel gruppo: `chgrp multiplexer /tmp/our_socket`.



103.6 Modificare le priorità di esecuzione del processo

Obiettivi LPI di riferimento

LPIC-1 v5, Exam 101, Objective 103.6

Peso

2

Aree di Conoscenza Chiave

- Conoscere la priorità predefinita di un processo che viene creato.
- Eseguire un programma con priorità maggiore o minore di quella predefinita.
- Modificare la priorità di un processo in esecuzione.

Di seguito è riportato un elenco parziale dei file, dei termini e dei comandi utilizzati

- `nice`
- `ps`
- `renice`
- `top`



103.6 Lezione 1

Certificazione:	LPIC-1
Versione:	5.0
Argomento:	103 Comandi GNU e Unix
Obiettivo:	103.6 Modificare le priorità di esecuzione di un processo
Lezione:	1 di 1

Introduzione

I sistemi operativi in grado di eseguire più di un processo contemporaneamente sono chiamati sistemi *multi-tasking* o multi-elaborazione. Mentre la vera simultaneità si verifica solo quando è disponibile più di un'unità di elaborazione, anche i sistemi a processore singolo possono simulare la simultaneità passando da un processo all'altro molto rapidamente. Questa tecnica viene impiegata anche in sistemi con molte CPU equivalenti, o sistemi *symmetric multi-processor (SMP)*, dato che il numero di potenziali processi simultanei supera notevolmente il numero di unità processore disponibili.

Infatti, solo un processo alla volta può utilizzare la CPU. Tuttavia, la maggior parte delle attività del processo sono *chiamate di sistema (system calls)*, ovvero il processo in esecuzione trasferisce il controllo della CPU ad un altro processo in modo che anch'esso esegua l'operazione richiesta. Le chiamate di sistema sono responsabili di tutte le comunicazioni tra dispositivi, come allocazioni di memoria, lettura e scrittura sui file system, stampa di testo sullo schermo, interazione dell'utente, trasferimenti di rete e così via. Il trasferimento del controllo della CPU durante una chiamata di sistema consente al sistema operativo di decidere se per riportare il controllo della CPU al processo precedente o per passarlo a un altro processo. Poiché le moderne CPU possono eseguire

le istruzioni molto più velocemente di quanto la maggior parte dell'hardware esterno possa comunicare, un nuovo processo di controllo può svolgere molto lavoro sulla CPU mentre le risposte hardware richieste in precedenza non sono ancora disponibili. Per garantire il massimo sfruttamento della CPU, i sistemi operativi multielaborazione mantengono una coda dinamica di processi attivi in attesa di uno slot di tempo della CPU.

Sebbene consentano di migliorare significativamente l'utilizzo del tempo della CPU, fare affidamento esclusivamente sulle chiamate di sistema per passare da un processo all'altro non è sufficiente per ottenere prestazioni multi-tasking soddisfacenti. Un processo che non effettua chiamate di sistema potrebbe utilizzare la CPU a tempo indeterminato. Questo è il motivo per cui i sistemi operativi moderni sono anche *preemptive*, ovvero un processo in esecuzione può essere rimesso in coda in modo che un processo più importante possa usare la CPU, anche se il processo in esecuzione non ha effettuato una chiamata di sistema.

Il Linux Scheduler

Linux, come sistema operativo multi-elaborazione preventivo (*preemptive*), implementa uno *scheduler* che organizza la coda dei processi. Più precisamente, lo scheduler decide anche quale *thread* accodato verrà eseguito - un processo può ramificare molti thread indipendenti - ma processo e thread sono termini intercambiabili in questo contesto. Ogni processo ha due predicati che intervengono sulla sua schedulazione: la *scheduling policy* e la *scheduling priority*.

Esistono due tipi principali di criteri di pianificazione (*scheduling*): *politiche in tempo reale* e *politiche normali*. I processi nell'ambito di una politica in tempo reale sono programmati direttamente dai loro valori di priorità. Se un processo più importante diventa pronto per essere eseguito, un processo in esecuzione meno importante viene interrotto e il processo con priorità più alta assume il controllo della CPU. Un processo con priorità più bassa otterrà il controllo della CPU solo se i processi con priorità più alta sono inattivi o in attesa di risposta hardware.

Qualsiasi processo in tempo reale ha una priorità maggiore rispetto a un processo normale. Come sistema operativo generico, Linux esegue solo pochi processi in tempo reale. La maggior parte dei processi, inclusi i programmi di sistema e utente, vengono eseguiti secondo i normali criteri di pianificazione. I processi normali di solito hanno lo stesso valore di priorità, ma le politiche normali possono definire regole di priorità di esecuzione utilizzando un altro predicato di processo: il *nice value*. Per evitare confusione con le priorità dinamiche derivate da valori accettabili, le priorità di pianificazione sono solitamente chiamate priorità di pianificazione *statiche*.

Lo scheduler di Linux può essere configurato in molti modi diversi ed esistono modi ancora più complessi per stabilire le priorità, ma questi concetti generali si applicano sempre. Durante l'ispezione e l'ottimizzazione della pianificazione dei processi, è importante tenere presente che

saranno interessati solo i processi con i normali criteri di pianificazione.

Leggere le Priorità

Linux riserva priorità statiche che vanno da 0 a 99 per i processi in tempo reale e priorità statiche che vanno da 100 a 139 per i processi normali, il che significa che ci sono 39 diversi livelli di priorità per i processi normali. Valori più bassi significano priorità più alta. La priorità statica di un processo attivo può essere trovata nel file `sched`, situato nella sua rispettiva directory all'interno del filesystem `/proc`

```
$ grep ^prio /proc/1/sched
prio                :          120
```

Come mostrato nell'esempio, la riga che inizia con `prio` fornisce il valore di priorità del processo (il processo PID 1 è il processo `init` o `systemd`, il primo processo che il kernel avvia durante l'inizializzazione del sistema). La priorità standard per i processi normali è 120, quindi può essere ridotta a 100 o aumentata a 139. Le priorità di tutti i processi in esecuzione possono essere verificate con il comando `ps -Al` o `ps -el`:

```
$ ps -el
F S  UID  PID  PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
4 S   0    1    0  0  80   0 -  9292 -        ?         00:00:00 systemd
4 S   0   19    1  0  80   0 -  8817 -        ?         00:00:00 systemd-journal
4 S  104   61    1  0  80   0 - 64097 -        ?         00:00:00 rsyslogd
4 S   0   63    1  0  80   0 -  7244 -        ?         00:00:00 cron
1 S   0  126    1  0  80   0 -  4031 -        ?         00:00:00 dhclient
4 S   0  154    1  0  80   0 -  3937 -      pts/0    00:00:00 agetty
4 S   0  155    1  0  80   0 -  3937 -      pts/1    00:00:00 agetty
4 S   0  156    1  0  80   0 -  3937 -      pts/2    00:00:00 agetty
4 S   0  157    1  0  80   0 -  3937 -      pts/3    00:00:00 agetty
4 S   0  158    1  0  80   0 -  3937 -    console  00:00:00 agetty
4 S   0  160    1  0  80   0 - 16377 -        ?         00:00:00 sshd
4 S   0  280    0  0  80   0 -  5301 -        ?         00:00:00 bash
0 R   0  392  280  0  80   0 -  7221 -        ?         00:00:00 ps
```

La colonna `PRI` indica la priorità statica assegnata dal kernel. Notare, tuttavia, che il valore di priorità visualizzato da `ps` è diverso da quello ottenuto nell'esempio precedente. Per ragioni storiche, le priorità visualizzate da `ps` vanno da -40 a 99 per impostazione predefinita, quindi la priorità effettiva si ottiene aggiungendovi 40 (in particolare, $80 + 40 = 120$).

È anche possibile monitorare continuamente i processi attualmente gestiti dal kernel Linux con il programma `top`. Come con `ps`, anche `top` mostra il valore di priorità in modo diverso. Per rendere più facile l'identificazione dei processi in tempo reale, `top` sottrae il valore di priorità di 100, rendendo così tutte le priorità in tempo reale *negative*, con un numero negativo o *rt* che le identifica. Pertanto, le priorità normali visualizzate da `top` vanno da 0 a 39.

NOTE

Per ottenere maggiori dettagli dal comando `ps`, è possibile utilizzare opzioni aggiuntive. Confronta l'output di questo comando con quello del nostro esempio precedente:

```
$ ps -e -o user,uid,comm,ttty,pid,ppid,pri,pmem,pcpu --sort=-pcpu | head
```

La Niceness del processo

Ogni processo normale inizia con un valore di *nice*ness predefinito di 0 (priorità 120). Il nome *nice* deriva dall'idea che i processi "più gentili" consentono ad altri processi di essere eseguiti prima di loro in una particolare coda di esecuzione. I numeri di *nice*ness vanno da -20 (meno gentilezza, priorità alta) a 19 (più gentilezza, priorità bassa). Linux consente anche di assegnare diversi valori di *nice*ness ai thread all'interno dello stesso processo. La colonna "NI" nell'output `ps` indica il numero *nice*.

Solo l'utente `root` può diminuire la *nice*ness di un processo sotto lo zero. È possibile avviare un processo con una priorità non standard con il comando `nice`. Per impostazione predefinita, `nice` cambia la *nice*ness in 10, ma può essere specificato con l'opzione `-n`:

```
$ nice -n 15 tar czf home_backup.tar.gz /home
```

In questo esempio, il comando `tar` viene eseguito con una *nice*ness di 15. Il comando `renice` può essere usato per cambiare la priorità di un processo in esecuzione. L'opzione `-p` indica il numero PID del processo di destinazione. Per esempio:

```
# renice -10 -p 2164
2164 (process ID) old priority 0, new priority -10
```

Le opzioni `-g` e `-u` sono usate per modificare rispettivamente tutti i processi di un gruppo o di un utente specifico. Con `renice +5 -g users`, la *nice*ness dei processi posseduti dagli utenti del gruppo `users` sarà aumentata a 5.

Oltre a `renice`, la priorità dei processi può essere modificata con altri programmi, come il

programma `top`. Nella schermata principale in alto, la niceness di un processo può essere modificata premendo `r` e quindi il numero PID del processo:

```
top - 11:55:21 up 23:38,  1 user,  load average: 0,10, 0,04, 0,05
Tasks:  20 total,   1 running, 19 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0,5 us,   0,3 sy,   0,0 ni, 99,0 id,   0,0 wa,   0,2 hi,   0,0 si,   0,0 st
KiB Mem : 4035808 total,  774700 free, 1612600 used, 1648508 buff/cache
KiB Swap: 7999828 total,  7738780 free,   261048 used. 2006688 avail Mem

PID to renice [default pid = 1]
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
1	root	20	0	74232	7904	6416	S	0,000	0,196	0:00.12	systemd
15	root	20	0	67436	6144	5568	S	0,000	0,152	0:00.03	systemd-journal
21	root	20	0	61552	5628	5000	S	0,000	0,139	0:00.01	systemd-logind
22	message+	20	0	43540	4072	3620	S	0,000	0,101	0:00.03	dbus-daemon
23	root	20	0	45652	6204	4992	S	0,000	0,154	0:00.06	wickedd-dhcp4
24	root	20	0	45648	6276	5068	S	0,000	0,156	0:00.06	wickedd-auto4
25	root	20	0	45648	6272	5060	S	0,000	0,155	0:00.06	wickedd-dhcp6

Il messaggio `PID to renice [default pid = 1]` appare con il primo processo elencato selezionato per impostazione predefinita. Per modificare la priorità di un altro processo, digita il suo PID e premi Invio. Apparirà quindi il messaggio `Renice PID 1 to value` (con il numero PID richiesto) e sarà possibile assegnare un nuovo valore di niceness.

Esercizi Guidati

1. In un sistema multitasking preventivo, cosa succede quando un processo con priorità inferiore occupa il processore e un processo con priorità più alta viene messo in coda per essere eseguito?

2. Osserva la seguente schermata di `top`:

```
top - 08:43:14 up 23 days, 12:29, 5 users, load average: 0,13, 0,18, 0,21
Tasks: 240 total, 2 running, 238 sleeping, 0 stopped, 0 zombie
%Cpu(s): 1,4 us, 0,4 sy, 0,0 ni, 98,1 id, 0,0 wa, 0,0 hi, 0,0 si, 0,0 st
MiB Mem : 7726,4 total, 590,9 free, 1600,8 used, 5534,7 buff/cache
MiB Swap: 30517,0 total, 30462,5 free, 54,5 used. 5769,4 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
1	root	20	0	171420	10668	7612	S	0,0	0,1	9:59.15	systemd
2	root	20	0	0	0	0	S	0,0	0,0	0:02.76	kthreadd
3	root	0	-20	0	0	0	I	0,0	0,0	0:00.00	rcu_gp
4	root	0	-20	0	0	0	I	0,0	0,0	0:00.00	rcu_par_gp
8	root	0	-20	0	0	0	I	0,0	0,0	0:00.00	mm_percpu_wq
9	root	20	0	0	0	0	S	0,0	0,0	0:49.06	ksoftirqd/0
10	root	20	0	0	0	0	I	0,0	0,0	18:24.20	rcu_sched
11	root	20	0	0	0	0	I	0,0	0,0	0:00.00	rcu_bh
12	root	rt	0	0	0	0	S	0,0	0,0	0:08.17	migration/0
14	root	20	0	0	0	0	S	0,0	0,0	0:00.00	cpuhp/0
15	root	20	0	0	0	0	S	0,0	0,0	0:00.00	cpuhp/1
16	root	rt	0	0	0	0	S	0,0	0,0	0:11.79	migration/1
17	root	20	0	0	0	0	S	0,0	0,0	0:26.01	ksoftirqd/1

Quali PID hanno priorità in tempo reale?

3. Osserva il seguente elenco `ps -el`:

F	S	UID	PID	PPID	C	PRI	NI	ADDR	SZ	WCHAN	TTY	TIME	CMD
4	S	0	1	0	0	80	0	-	42855	-	?	00:09:59	systemd
1	S	0	2	0	0	80	0	-	0	-	?	00:00:02	kthreadd
1	I	0	3	2	0	60	-20	-	0	-	?	00:00:00	rcu_gp
1	S	0	9	2	0	80	0	-	0	-	?	00:00:49	ksoftirqd/0
1	I	0	10	2	0	80	0	-	0	-	?	00:18:26	rcu_sched

1	I	0	11	2	0	80	0	-	0	-	?	00:00:00	rcu_bh
1	S	0	12	2	0	-40	-	-	0	-	?	00:00:08	migration/0
1	S	0	14	2	0	80	0	-	0	-	?	00:00:00	cpuhp/0
5	S	0	15	2	0	80	0	-	0	-	?	00:00:00	cpuhp/1

Quale PID ha la priorità più alta?

4. Dopo aver provato a rinominare un processo con `renice`, si verifica il seguente errore:

```
$ renice -10 21704
renice: failed to set priority for 21704 (process ID): Permission denied
```

Qual è la probabile causa dell'errore?

Esercizi Esplorativi

1. La modifica delle priorità del processo è generalmente richiesta quando un processo occupa troppo tempo della CPU. Usando `ps` con le opzioni standard per visualizzare tutti i processi di sistema in formato lungo, quale flag `--sort` ordinerà i processi in base all'utilizzo della CPU, aumentando l'ordine?

2. Il comando `schedtool` può impostare tutti i parametri di pianificazione della CPU di cui Linux è capace o visualizzare informazioni per determinati processi. Come può essere utilizzato per visualizzare i parametri di schedulazione del processo 1750? Inoltre, come può essere usato `schedtool` per cambiare il processo 1750 in tempo reale con priorità -90 (come mostrato da `top`)?

Sommario

Questa lezione illustra come Linux condivide il tempo della CPU tra i suoi processi gestiti. Per garantire le migliori prestazioni, i processi più critici devono superare i processi meno critici. La lezione segue i seguenti passaggi:

- Concetti di base sui sistemi multi-processing.
- Che cos'è uno scheduler di processo e come Linux lo implementa.
- Quali sono le priorità di Linux, i valori di niceness e il loro scopo.
- Come leggere e interpretare le priorità del processo in Linux.
- Come modificare la priorità di un processo, prima e durante la sua esecuzione

Risposte agli Esercizi Guidati

1. In un sistema multitasking preventivo, cosa succede quando un processo con priorità inferiore occupa il processore e un processo con priorità più alta viene messo in coda per essere eseguito?

Il processo con priorità più bassa si interrompe e al suo posto viene eseguito il processo con priorità più alta.

2. Osserva la seguente schermata di `top`:

```
top - 08:43:14 up 23 days, 12:29, 5 users, load average: 0,13, 0,18, 0,21
Tasks: 240 total, 2 running, 238 sleeping, 0 stopped, 0 zombie
%Cpu(s): 1,4 us, 0,4 sy, 0,0 ni, 98,1 id, 0,0 wa, 0,0 hi, 0,0 si, 0,0 st
MiB Mem : 7726,4 total, 590,9 free, 1600,8 used, 5534,7 buff/cache
MiB Swap: 30517,0 total, 30462,5 free, 54,5 used. 5769,4 avail Mem

  PID USER      PR  NI   VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
    1 root        20   0  171420  10668   7612 S   0,0   0,1   9:59.15 systemd
    2 root        20   0        0        0        0 S   0,0   0,0   0:02.76 kthreadd
    3 root         0 -20        0        0        0 I   0,0   0,0   0:00.00 rcu_gp
    4 root         0 -20        0        0        0 I   0,0   0,0   0:00.00 rcu_par_gp
    8 root         0 -20        0        0        0 I   0,0   0,0   0:00.00 mm_percpu_wq
    9 root        20   0        0        0        0 S   0,0   0,0   0:49.06 ksoftirqd/0
   10 root        20   0        0        0        0 I   0,0   0,0  18:24.20 rcu_sched
   11 root        20   0        0        0        0 I   0,0   0,0   0:00.00 rcu_bh
   12 root        rt    0        0        0        0 S   0,0   0,0   0:08.17 migration/0
   14 root        20   0        0        0        0 S   0,0   0,0   0:00.00 cpuhp/0
   15 root        20   0        0        0        0 S   0,0   0,0   0:00.00 cpuhp/1
   16 root        rt    0        0        0        0 S   0,0   0,0   0:11.79 migration/1
   17 root        20   0        0        0        0 S   0,0   0,0   0:26.01 ksoftirqd/1
```

Quali PID hanno priorità in tempo reale?

PIDs 12 e 16.

3. Osserva il seguente elenco `ps -el::`:

```
F S  UID    PID  PPID  C PRI  NI ADDR SZ WCHAN  TTY          TIME CMD
4 S   0      1     0  0  80   0 - 42855 -      ?           00:09:59 systemd
1 S   0      2     0  0  80   0 -      0 -      ?           00:00:02 kthreadd
1 I   0      3     2  0  60 -20 -      0 -      ?           00:00:00 rcu_gp
```

```

1 S      0      9      2 0 80 0 - 0 - ?      00:00:49 ksoftirqd/0
1 I      0     10      2 0 80 0 - 0 - ?      00:18:26 rcu_sched
1 I      0     11      2 0 80 0 - 0 - ?      00:00:00 rcu_bh
1 S      0     12      2 0 -40 - - 0 - ?      00:00:08 migration/0
1 S      0     14      2 0 80 0 - 0 - ?      00:00:00 cpuhp/0
5 S      0     15      2 0 80 0 - 0 - ?      00:00:00 cpuhp/1

```

Quale PID ha la priorità più alta?

PID 12.

4. Dopo aver provato a rinominare un processo con `renice`, si verifica il seguente errore:

```

$ renice -10 21704
renice: failed to set priority for 21704 (process ID): Permission denied

```

Qual è la probabile causa dell'errore?

Solo l'utente root può diminuire i valori di niceness sotto lo zero.

Risposte agli Esercizi Esplorativi

1. La modifica delle priorità del processo è generalmente richiesta quando un processo occupa troppo tempo della CPU. Usando `ps` con le opzioni standard per visualizzare tutti i processi di sistema in formato lungo, quale flag `--sort` ordinerà i processi in base all'utilizzo della CPU, aumentando l'ordine?

```
$ ps -el --sort=pcpu
```

2. Il comando `schedtool` può impostare tutti i parametri di pianificazione della CPU di cui Linux è capace o visualizzare informazioni per determinati processi. Come può essere utilizzato per visualizzare i parametri di schedulazione del processo 1750? Inoltre, come può essere usato `schedtool` per cambiare il processo 1750 in tempo reale con priorità -90 (come mostrato da `top`)?

```
$ schedtool 1750
```

```
$ schedtool -R -p 89 1750
```



103.7 Cercare file di testo utilizzando espressioni regolari

Obiettivi LPI di riferimento

LPIC-1 v5, Exam 101, Objective 103.7

Peso

3

Aree di Conoscenza Chiave

- Creare semplici espressioni regolari contenenti diversi elementi notazionali.
- Comprendere le differenze tra le espressioni regolari di base ed estese.
- Comprendere i concetti di caratteri speciali, classi di caratteri, quantificatori e ancore.
- Utilizzare strumenti di espressioni regolari per eseguire ricerche in un file system o nel contenuto di file.
- Utilizzare espressioni regolari per eliminare, modificare e sostituire il testo.

Di seguito è riportato un elenco parziale dei file, dei termini e dei comandi utilizzati

- `grep`
- `egrep`
- `fgrep`
- `sed`
- `regex(7)`



103.7 Lezione 1

Certificazione:	LPIC-1
Versione:	5.0
Argomento:	103 Comandi GNU e Unix
Obiettivo:	103.7 Cercare file di testo utilizzando espressioni regolari
Lezione:	1 di 2

Introduzione

Gli algoritmi di ricerca delle stringhe sono ampiamente utilizzati in diversi compiti di elaborazione dati, tanto che i sistemi operativi *Unix-like* hanno la loro onnipresente implementazione: *Espressioni regolari (Regular expressions)*, spesso abbreviate in *REs*. Le espressioni regolari sono costituite da sequenze di caratteri che costituiscono un modello generico utilizzato per individuare e talvolta modificare una sequenza corrispondente in una stringa di caratteri più grande. Le espressioni regolari espandono notevolmente la capacità di:

- Scrivere regole di analisi per le richieste nei server HTTP, in particolare *nginx*.
- Scrivere script che convertono set di dati basati su testo in un altro formato.
- Ricercare le occorrenze di interesse in voci di diario o documenti.
- Filtrare i documenti di markup, mantenendo il contenuto semantico.

L'espressione regolare più semplice contiene almeno un *atomo*. Un atomo, così chiamato perché è l'elemento base di un'espressione regolare, è solo un carattere che può avere o meno un significato speciale. La maggior parte dei caratteri ordinari non sono ambigui, mantengono il loro

significato letterale, mentre altri hanno un significato speciale:

. (punto)

Il carattere corrisponde a un qualsiasi carattere.

^ (segno d'omissione)

Il carattere corrisponde all'inizio della linea.

\$ (simbolo del dollaro)

Il carattere corrisponde alla fine della linea.

Per esempio, l'espressione regolare `bc`, composta dagli atomi letterali `b` e `c`, può essere trovata nella stringa `abcd`, ma non può essere trovata nella stringa `a1cd`. D'altra parte, l'espressione regolare `.c` può essere trovata in entrambe le stringhe `abcd` e `a1cd`, poiché il punto `.` corrisponde a qualsiasi carattere.

Gli atomi del segno di omissione e del dollaro vengono utilizzati quando interessano solo le corrispondenze all'inizio o alla fine della stringa. Per questo motivo sono anche chiamati *anchors*. `cd` può essere trovato in `abcd`, ma `^cd` no. Allo stesso modo, `ab` può essere trovato in `abcd`, ma `ab$` non può. Il simbolo di omissione `^` è un carattere letterale tranne quando all'inizio e `$` è un carattere letterale tranne quando alla fine dell'espressione regolare.

Espressioni tra Parentesi Quadre

Esiste un altro tipo di atomo denominato *bracket expression*. Sebbene non sia un singolo carattere, le parentesi `[]` (incluso il loro contenuto) sono considerate un singolo atomo. Un'espressione tra parentesi quadre di solito è solo un elenco di caratteri letterali racchiusi da `[]`, facendo sì che l'atomo corrisponda a ogni singolo carattere dell'elenco. Per esempio, l'espressione `[1b]` può essere trovata in entrambe le stringhe `abcd` e `a1cd`. Per specificare i caratteri a cui l'atomo non deve corrispondere, l'elenco deve iniziare con `^`, come in `[^ 1b]`. È anche possibile specificare intervalli di caratteri nelle espressioni di parentesi. Per esempio, `[0-9]` corrisponde alle cifre da 0 a 9 e `[a-z]` corrisponde a qualsiasi lettera minuscola. Gli intervalli devono essere utilizzati con cautela, poiché potrebbero non essere coerenti tra le diverse impostazioni locali.

Gli elenchi di espressioni tra parentesi quadre accettano anche classi invece di singoli caratteri e intervalli. Le classi sono:

[:alnum:]

Rappresenta un carattere alfanumerico.

[:alpha:]

Rappresenta un carattere alfabetico.

[:ascii:]

Rappresenta un carattere che si adatta al set di caratteri ASCII.

[:blank:]

Rappresenta un carattere vuoto, ovvero uno spazio o una tabulazione.

[:cntrl:]

Rappresenta un carattere di controllo.

[:digit:]

Rappresenta una cifra (da 0 a 9).

[:graph:]

Rappresenta qualsiasi carattere stampabile tranne lo spazio.

[:lower:]

Rappresenta un carattere minuscolo.

[:print:]

Rappresenta qualsiasi carattere stampabile compreso lo spazio.

[:punct:]

Rappresenta qualsiasi carattere stampabile che non sia uno spazio o un carattere alfanumerico.

[:space:]

Rappresenta i caratteri spazi vuoti: spazio, avanzamento modulo (`\f`), nuova riga (`\n`), ritorno a capo (`\r`), tabulazione orizzontale (`\t`) e tabulazione verticale (`\v`).

[:upper:]

Rappresenta un carattere maiuscolo.

[:xdigit:]

Rappresenta cifre esadecimali (da 0 a F)

Le classi di caratteri possono essere combinate con singoli caratteri e intervalli, ma non possono essere utilizzate come punto finale di un intervallo. Inoltre, le classi di caratteri possono essere utilizzate solo nelle espressioni di parentesi, non come un atomo indipendente al di fuori delle

parentesi.

Quantificatori

La portata di un atomo, sia di un singolo carattere che di una parentesi, può essere regolata utilizzando un *quantificatore*. I quantificatori atomici definiscono le sequenze atomiche, ovvero le corrispondenze si verificano quando nella stringa viene trovata una ripetizione contigua per l'atomo. La sottostringa corrispondente alla corrispondenza è chiamata *parte*. Tuttavia, i quantificatori e altre caratteristiche delle espressioni regolari vengono trattati in modo diverso a seconda dello standard utilizzato.

Come definito dallo standard POSIX, ci sono due forme di espressioni regolari: espressioni regolari “basic” ed espressioni regolari “extended”. La maggior parte dei programmi relativi al testo in qualsiasi distribuzione Linux convenzionale supporta entrambe le forme, quindi è importante conoscere le loro differenze al fine di evitare problemi di compatibilità e scegliere l'implementazione più adatta per l'attività prevista.

Il quantificatore `*` ha la stessa funzione sia nelle RE di base che in quelle estese (l'atomo ricorre zero o più volte) ed è un carattere letterale se appare all'inizio dell'espressione regolare o se è preceduto da un back slash `\`. Il quantificatore del segno più ``` selezionerà i "pezzi" contenenti uno o più corrispondenze di atomi in sequenza. Con il quantificatore del punto interrogativo ``?`, Si verificherà una corrispondenza se l'atomo corrispondente appare una volta o se non compare affatto. Se preceduto da un \_back slash\_ ``\``, il loro significato speciale non viene considerato. Le espressioni regolari di base supportano anche i quantificatori ``?` e ``+``, Ma devono essere preceduti da un back slash. A differenza delle espressioni regolari estese, `+`` e `?`` sono di per sé caratteri letterali nelle espressioni regolari di base.

Limiti

Un *bound* è un quantificatore di atomi che, come suggerisce il nome, consente a un utente di specificare limiti di quantità precisi per un atomo. Nelle espressioni regolari estese un limite può apparire in tre forme:

`{i}`

L'atomo deve apparire esattamente `i` volte (`i` è un numero intero). Per esempio, `[[:blank:]]{2}` corrisponde esattamente a due caratteri vuoti.

`{i,}`

L'atomo deve apparire almeno `i` volte (`i` è un numero intero). Per esempio, `[[:blank:]]{2,}`

corrisponde a qualsiasi sequenza di due o più caratteri vuoti.

{i,j}

L'atomo deve apparire almeno *i* volte e al massimo *j* volte (*i* e *j* sono numeri interi, *j* è maggiore di *i*). Per esempio, `xyz{2,4}` corrisponde alla stringa `xy` seguita da due a quattro del carattere `z`.

In ogni caso, se una sottostringa corrisponde a un'espressione regolare e anche una sottostringa più lunga che inizia nello stesso punto corrisponde, verrà considerata la sottostringa più lunga.

Anche le espressioni regolari di base supportano i limiti, ma i delimitatori devono essere preceduti da `\`: `\{` e `\}`. Di per sé, `{` e `}` vengono interpretati come caratteri letterali. Una `\{` seguita da un carattere diverso da una cifra è un carattere letterale, non l'inizio di un limite.

Rami e Riferimenti all'Indietro

Le espressioni regolari di base differiscono dalle espressioni regolari estese anche per un altro aspetto importante: un'espressione regolare estesa può essere suddivisa in *rami* (*branches*), ciascuna un'espressione regolare indipendente. I rami sono separati da `|` e l'espressione regolare combinata corrisponderà a tutto ciò che corrisponde a uno qualsiasi dei rami. Per esempio, `he|him` corrisponderà se nella stringa in esame si trova la sottostringa `he` o `him`. Le espressioni regolari di base interpretano `|` come un carattere letterale. Tuttavia, la maggior parte dei programmi che supportano le espressioni regolari di base consentiranno i rami con `\|`.

Un'espressione regolare estesa racchiusa tra `()` può essere utilizzata in un *riferimento all'indietro*. Per esempio: `([[:digit:]])\1` corrisponderà a qualsiasi espressione regolare che si ripete almeno una volta, perché `\1` nell'espressione è il riferimento all'indietro al pezzo corrispondente alla prima parentesi sottoespressione. Se nell'espressione regolare esiste più di una sottoespressione tra parentesi, è possibile fare riferimento a `\2`, `\3` e così via.

Per le RE di base, le sottoespressioni devono essere racchiuse tra `\(` e `\)`, con `(` e `)` da sole saranno considerate come caratteri ordinari. L'indicatore di riferimento all'indietro viene utilizzato come nelle espressioni regolari estese

Ricerca con Espressioni Regolari

Il vantaggio immediato offerto dalle espressioni regolari è quello di migliorare le ricerche sui filesystem e nei documenti di testo. L'opzione `-regex` del comando `find` permette di testare ogni percorso in una gerarchia di directory rispetto a un'espressione regolare. Per esempio:

```
$ find $HOME -regex '.*\/\..*' -size +100M
```

Cerca file più grandi di 100 megabyte (100 unità di 1048576 byte), ma solo nei percorsi all'interno della directory *home* dell'utente che contengono una corrispondenza con `.*\/\..*`, cioè un `/` circondato da qualsiasi altro numero di caratteri. In altre parole, verranno elencati solo i file nascosti o i file all'interno di directory nascoste, indipendentemente dalla posizione di `/`, nel percorso corrispondente. Per le espressioni regolari senza distinzione tra maiuscole e minuscole, dovrebbe essere utilizzata l'opzione `-iregex`:

```
$ find /usr/share/fonts -regextype posix-extended -iregex
'.*(dejavu|liberation).*sans.*(italic|oblique).*'
/usr/share/fonts/dejavu/DejaVuSansCondensed-BoldOblique.ttf
/usr/share/fonts/dejavu/DejaVuSansCondensed-Oblique.ttf
/usr/share/fonts/dejavu/DejaVuSans-BoldOblique.ttf
/usr/share/fonts/dejavu/DejaVuSans-Oblique.ttf
/usr/share/fonts/dejavu/DejaVuSansMono-BoldOblique.ttf
/usr/share/fonts/dejavu/DejaVuSansMono-Oblique.ttf
/usr/share/fonts/liberation/LiberationSans-BoldItalic.ttf
/usr/share/fonts/liberation/LiberationSans-Italic.ttf
```

In questo esempio, l'espressione regolare contiene rami (scritti in stile *extended*) per elencare solo file di font specifici nella gerarchia di directory `/usr/share/fonts`. Le espressioni regolari estese non sono supportate di default, ma `find` permette di abilitarle con `-regextype posix-extended` o `-regextype egrep`. Lo standard RE predefinito per `find` è *findutils-default*, che è virtualmente un clone di espressioni regolari di base.

Spesso è necessario passare l'output di un programma al comando `less` quando non si adatta allo schermo. Il comando `less` divide il suo input in pagine, una schermata alla volta, consentendo all'utente di navigare facilmente nel testo su e giù. Inoltre, `less` consente anche a un utente di eseguire ricerche basate su espressioni regolari. Questa caratteristica è particolarmente importante perché `less` è il paginatore predefinito usato per molte attività quotidiane, come l'ispezione delle voci del diario o la consultazione delle pagine di manuale. Quando si legge una pagina di manuale, per esempio, premendo il tasto `/` si aprirà un prompt di ricerca. Questo è uno scenario tipico in cui le espressioni regolari sono utili, poiché le opzioni di comando sono elencate subito dopo un margine di pagina nel layout generale della pagina del manuale. Tuttavia, la stessa opzione potrebbe apparire molte volte nel testo, rendendo irrealizzabili le ricerche letterali. Indipendentemente da ciò, digitando `^[[[:blank:]]-o -o` o più semplicemente: `^-o—` nel prompt di ricerca si salterà immediatamente all'opzione della sezione `-o` (se esiste) dopo aver premuto Invio, consentendo così di consultare più rapidamente la descrizione di un'opzione.

Esercizi Guidati

Quale espressione regolare estesa corrisponderebbe a qualsiasi indirizzo email, come ``info@example.org``?

+

1. Quale espressione regolare estesa corrisponderebbe solo a qualsiasi indirizzo IPv4 nel formato standard a quattro punti, come `192.168.15.1`?

2. Come può essere usato il comando `grep` per elencare il contenuto del file `/etc/services`, scartando tutti i commenti (righe che iniziano con `#`)?

3. Il file `domains.txt` contiene un elenco di nomi di dominio, uno per riga. Come sarebbe usato il comando `egrep` per elencare solo i domini `.org` o `.com`?

Esercizi Esplorativi

1. Dalla directory corrente, in che modo il comando `find` userebbe un'espressione regolare estesa per cercare tutti i file che non contengono un suffisso di file standard (i nomi di file non terminano con `.txt` o `.c`, per esempio)?

2. Il comando `less` è il paginatore predefinito per visualizzare file di testo lunghi nell'ambiente shell. Digitando `/` è possibile inserire un'espressione regolare nel prompt di ricerca per passare alla prima corrispondenza trovata. Per rimanere nella posizione del documento corrente ed evidenziare solo le corrispondenze, quale combinazione di tasti deve essere inserita al prompt di ricerca?

3. In `less`, come sarebbe possibile filtrare l'output in modo che vengano visualizzate solo le righe che corrispondono a un'espressione regolare?

Sommario

Questa lezione tratta del supporto generale di Linux alle espressioni regolari, uno standard ampiamente utilizzato le cui capacità di corrispondenza dei modelli sono supportate dalla maggior parte dei programmi relativi al testo. La lezione segue i seguenti passaggi:

- Che cos'è un'espressione regolare.
- I componenti principali di un'espressione regolare.
- Le differenze tra espressioni regolari basiche ed estese.
- Come eseguire semplici ricerche di testo e file utilizzando espressioni regolari.

Risposte agli Esercizi Guidati

1. Quale espressione regolare estesa corrisponderebbe a qualsiasi indirizzo email, come `info@example.org`?

```
egrep "\S+@\S+\.\S+"
```

2. Quale espressione regolare estesa corrisponderebbe solo a qualsiasi indirizzo IPv4 nel formato standard a quattro punti, come `192.168.15.1`?

```
egrep "[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}"
```

3. Come può essere usato il comando `grep` per elencare il contenuto del file `/etc/services`, scartando tutti i commenti (righe che iniziano con `#`)?

```
grep -v ^# /etc/services
```

4. Il file `domains.txt` contiene un elenco di nomi di dominio, uno per riga. Come sarebbe usato il comando `egrep` per elencare solo i domini `.org` o `.com`?

```
egrep "\.org$|\.com$" domains.txt
```

Risposte agli Esercizi Esplorativi

1. Dalla directory corrente, in che modo il comando `find` userebbe un'espressione regolare estesa per cercare tutti i file che non contengono un suffisso di file standard (i nomi di file non terminano con `.txt` o `.c`, per esempio)?

```
find . -type f -regextype egrep -not -regex '.*\.[[:alnum:]]{1,}$'
```

2. Il comando `less` è il paginatore predefinito per visualizzare file di testo lunghi nell'ambiente shell. Digitando `/` è possibile inserire un'espressione regolare nel prompt di ricerca per passare alla prima corrispondenza trovata. Per rimanere nella posizione del documento corrente ed evidenziare solo le corrispondenze, quale combinazione di tasti deve essere inserita al prompt di ricerca?

Premendo `Ctrl + K` prima di inserire l'espressione di ricerca.

3. In `less`, come sarebbe possibile filtrare l'output in modo che vengano visualizzate solo le righe che corrispondono a un'espressione regolare?

Premendo `&` e inserendo l'espressione di ricerca.



Linux
Professional
Institute

103.7 Lezione 2

Certificazione:	LPIC-1
Versione:	5.0
Argomento:	103 Comandi GNU e Unix
Obiettivo:	103.7 Cercare file di testo utilizzando espressioni regolari
Lezione:	2 di 2

Introduzione

Lo streaming dei dati attraverso una catena di comandi in *pipe* consente l'applicazione di filtri composti basati su espressioni regolari. Le espressioni regolari sono una tecnica importante utilizzata non solo nell'amministrazione del sistema, ma anche nel *data mining* e nelle aree correlate. Due comandi sono particolarmente adatti per manipolare file e dati di testo usando espressioni regolari: `grep` e `sed`. `grep` è un cercatore di pattern e `sed` è un editor di flussi. Sono utili da soli, ma è quando lavorano insieme ad altri processi che diventano insostituibili.

Il "Cercatore" di Pattern: `grep`

Uno degli usi più comuni di `grep` è quello di facilitare l'ispezione di file lunghi, utilizzando l'espressione regolare come filtro applicato a ciascuna riga. Può essere utilizzato per mostrare solo le righe che iniziano con un certo termine. `grep` può essere usato per cercare in un file di configurazione i moduli del kernel, elencando solo le righe delle opzioni:

```
$ grep '^options' /etc/modprobe.d/alsa-base.conf
```

```
options snd-pcsp index=-2
options snd-usb-audio index=-2
options bt87x index=-2
options cx88_alsa index=-2
options snd-atiixp-modem index=-2
options snd-intel8x0m index=-2
options snd-via82xx-modem index=-2
```

Il carattere pipe `|` può essere impiegato per reindirizzare l'output di un comando direttamente all'input di `grep`. L'esempio seguente utilizza un'espressione con parentesi quadre per selezionare le righe dall'output di `fdisk -l`, che iniziano con `Disk /dev/sda` o `Disk /dev/sdb`:

```
# fdisk -l | grep '^Disk /dev/sd[ab] '
Disk /dev/sda: 320.1 GB, 320072933376 bytes, 625142448 sectors
Disk /dev/sdb: 7998 MB, 7998537728 bytes, 15622144 sectors
```

La semplice selezione di linee con corrispondenze potrebbe non essere appropriata per un compito particolare, richiedendo aggiustamenti al comportamento di `grep` attraverso le sue opzioni. Per esempio, l'opzione `-c` o `--count` dice a `grep` di mostrare quante righe hanno corrispondenze:

```
# fdisk -l | grep '^Disk /dev/sd[ab] ' -c
2
```

L'opzione può essere inserita prima o dopo l'espressione regolare. Altre importanti opzioni di `grep` sono:

-c o --count

Invece di visualizzare i risultati della ricerca, visualizza solo il conteggio totale del numero di volte in cui si verifica una corrispondenza in un determinato file.

-i o --ignore-case

Imposta la ricerca senza distinzione tra maiuscole e minuscole.

-f FILE o --file=FILE

Indica un file contenente l'espressione regolare da utilizzare.

-n o --line-number

Mostra il numero della riga.

-v o --invert-match

Seleziona ogni riga, tranne quelle che contengono corrispondenze.

-H o --with-filename

Stampa anche il nome del file contenente la riga.

-z o --null-data

Invece di fare in modo che `grep` tratti i flussi di dati di input e output come linee separate (usando *newline* per impostazione predefinita), prendi l'input o l'output come una sequenza di righe. Quando si combina l'output del comando `find` usando la sua opzione `-print0` con il comando `grep`, l'opzione `-z o --null-data` dovrebbe essere usata per elaborare lo *stream* nello stesso modo.

Sebbene sia attivata per impostazione predefinita quando vengono forniti più percorsi di file come input, l'opzione `-H` non è attivata per singoli file. Ciò può essere critico in situazioni speciali, come quando `grep` viene chiamato direttamente da `find`, per esempio:

```
$ find /usr/share/doc -type f -exec grep -i '3d modeling' "{}" \; | cut -c -100
artistic aspects of 3D modeling. Thus this might be the application you are
This major approach of 3D modeling has not been supported
oce is a C++ 3D modeling library. It can be used to develop CAD/CAM softwares, for instance
[FreeCad
```

In questo esempio, `find` elenca tutti i file in `/usr/share/doc`, quindi li passa ciascuno a `grep`, che a sua volta esegue una ricerca senza distinzione tra maiuscole e minuscole per `3d modeling` all'interno del file. Il flusso da tagliare (`cut`) serve solo per limitare la lunghezza di output a 100 colonne. Notare, tuttavia, che non c'è modo di sapere da quale file provengano le righe. Questo problema viene risolto aggiungendo `-H` a `grep`:

```
$ find /usr/share/doc -type f -exec grep -i -H '3d modeling' "{}" \; | cut -c -100
/usr/share/doc/openscad/README.md:artistic aspects of 3D modeling. Thus this might be the
applicatio
/usr/share/doc/openscg/doc/publications.html:This major approach of 3D modeling has not been
support
```

Ora è possibile identificare i file in cui è stata trovata ogni corrispondenza. Per rendere l'elenco ancora più informativo, è possibile aggiungere righe iniziali e finali alle righe con corrispondenze:

```
$ find /usr/share/doc -type f -exec grep -i -H -1 '3d modeling' "{}" \; | cut -c -100
/usr/share/doc/openscad/README.md-application Blender), OpenSCAD focuses on the CAD aspects
```

```
rather t
/usr/share/doc/openscad/README.md:artistic aspects of 3D modeling. Thus this might be the
applicatio
/usr/share/doc/openscad/README.md-looking for when you are planning to create 3D models of
machine p
/usr/share/doc/opencsg/doc/publications.html-3D graphics library for Constructive Solid
Geometry (CS
/usr/share/doc/opencsg/doc/publications.html:This major approach of 3D modeling has not been
support
/usr/share/doc/opencsg/doc/publications.html-by real-time computer graphics until recently.
```

L'opzione `-1` dice a `grep` di includere una riga prima e una riga dopo la riga trovata con una corrispondenza. Queste righe aggiuntive sono chiamate *linee di contesto* e sono identificate nell'output da un segno meno dopo il nome del file. Lo stesso risultato può essere ottenuto con `-C 1` o `--context=1` e possono essere indicate altre quantità di righe di contesto.

Ci sono due programmi complementari a `grep`: `egrep` e `fgrep`. Il programma `egrep` è equivalente al comando `grep -E`, che incorpora funzionalità extra oltre alle espressioni regolari di base. Per esempio, con `egrep` è possibile utilizzare funzionalità di espressioni regolari estese, come la ramificazione:

```
$ find /usr/share/doc -type f -exec egrep -i -H -1 '3d (modeling|printing)' "{}" \; | cut -c
-100
/usr/share/doc/openscad/README.md-application Blender), OpenSCAD focuses on the CAD aspects
rather t
/usr/share/doc/openscad/README.md:artistic aspects of 3D modeling. Thus this might be the
applicatio
/usr/share/doc/openscad/README.md-looking for when you are planning to create 3D models of
machine p
/usr/share/doc/openscad/RELEASE_NOTES.md-* Support for using 3D-Mouse / Joystick / Gamepad
input dev
/usr/share/doc/openscad/RELEASE_NOTES.md:* 3D Printing support: Purchase from a print
service partne
/usr/share/doc/openscad/RELEASE_NOTES.md-* New export file formats: SVG, 3MF, AMF
/usr/share/doc/opencsg/doc/publications.html-3D graphics library for Constructive Solid
Geometry (CS
/usr/share/doc/opencsg/doc/publications.html:This major approach of 3D modeling has not been
support
/usr/share/doc/opencsg/doc/publications.html-by real-time computer graphics until recently.
```

In questo esempio, `3d modeling` o `3d printing` corrisponderanno all'espressione, senza distinzione tra maiuscole e minuscole. Per visualizzare solo le parti di un flusso di testo che

corrispondono all'espressione usata da `egrep`, utilizzare l'opzione `-o`.

Il programma `fgrep` è equivalente a `grep -F`, cioè non analizza le espressioni regolari: è utile nelle ricerche semplici in cui l'obiettivo è trovare una corrispondenza con un'espressione letterale. Pertanto, caratteri speciali come il simbolo del dollaro e il punto verranno presi alla lettera e non dal loro significato in un'espressione regolare.

L'Editor di Stream: sed

Lo scopo del programma `sed` è quello di modificare i dati basati su testo in modo non interattivo. Questo significa che tutte le modifiche vengono effettuate tramite istruzioni predefinite, non digitando direttamente su un testo visualizzato a schermo. In termini "moderni", `sed` può essere inteso come un *template parser*: dato un testo come input, colloca il contenuto personalizzato in posizioni predefinite o quando trova una corrispondenza per un'espressione regolare.

`Sed`, come suggerisce il nome, è adatto per il testo trasmesso tramite pipeline. La sua sintassi di base è `sed -f SCRIPT` quando le istruzioni di modifica sono memorizzate nel file `SCRIPT` o `sed -e COMMANDS` per eseguire `COMMANDS` direttamente dalla riga di comando. Se non sono presenti né `-f` né `-e`, `sed` utilizza il primo parametro non di opzione come file di script. È anche possibile usare un file come input semplicemente dando il suo percorso come argomento a `sed`.

Le istruzioni `sed` sono composte da un singolo carattere, possibilmente preceduto da un indirizzo o seguite da una o più opzioni, e vengono applicate ad una riga alla volta. Gli indirizzi possono essere un numero di riga singola, un'espressione regolare o un intervallo di righe. Per esempio, la prima riga di un flusso di testo può essere eliminata con `1d`, dove `1` specifica la riga in cui verrà applicato il comando di eliminazione `d`. Per chiarire l'uso di `sed`, prendi l'output del comando `factor `seq 12``, che restituisce i fattori primi per i numeri da 1 a 12:

```
$ factor `seq 12`
1:
2: 2
3: 3
4: 2 2
5: 5
6: 2 3
7: 7
8: 2 2 2
9: 3 3
10: 2 5
11: 11
12: 2 2 3
```

L'eliminazione della prima riga con `sed` si ottiene con `1d`:

```
$ factor `seq 12` | sed 1d
2: 2
3: 3
4: 2 2
5: 5
6: 2 3
7: 7
8: 2 2 2
9: 3 3
10: 2 5
11: 11
12: 2 2 3
```

È possibile specificare un intervallo di righe con una virgola di separazione:

```
$ factor `seq 12` | sed 1,7d
8: 2 2 2
9: 3 3
10: 2 5
11: 11
12: 2 2 3
```

È possibile utilizzare più di un'istruzione nella stessa esecuzione, separate da punto e virgola. In questo caso, però, è importante racchiuderli tra virgolette in modo che il punto e virgola non venga interpretato dalla shell:

```
$ factor `seq 12` | sed "1,7d;11d"
8: 2 2 2
9: 3 3
10: 2 5
12: 2 2 3
```

In questo esempio, sono state eseguite due istruzioni di cancellazione, prima sulle righe da 1 a 7 e poi sulla riga 11. Un indirizzo può anche essere un'espressione regolare, quindi solo le righe con una corrispondenza saranno influenzate dall'istruzione:

```
$ factor `seq 12` | sed "1d;/:.*2.*/d"
3: 3
```

```
5: 5
7: 7
9: 3 3
11: 11
```

L'espressione regolare `:. *2.*` corrisponde a qualsiasi occorrenza del numero 2 in qualsiasi punto dopo i due punti, causando la cancellazione delle righe corrispondenti ai numeri con 2 come fattore. Con `sed`, qualsiasi cosa inserita tra gli slash (/) è considerata un'espressione regolare e per impostazione predefinita sono supportate tutte le RE di base. Per esempio, `sed -e "/^#/d" /etc/services` mostra il contenuto del file `/etc/services` senza le righe che iniziano con `#` (righe di commento).

L'istruzione di cancellazione `d` è solo una delle tante istruzioni di modifica fornite da `sed`. Invece di eliminare una riga, `sed` può sostituirla con un dato testo:

```
$ factor `seq 12` | sed "1d;/:.*2.*/c REMOVED"
REMOVED
3: 3
REMOVED
5: 5
REMOVED
7: 7
REMOVED
9: 3 3
REMOVED
11: 11
REMOVED
```

L'istruzione `c REMOVED` sostituisce semplicemente una riga con il testo `REMOVED`. Nel caso dell'esempio, ogni riga con una sottostringa che corrisponde all'espressione regolare `:. *2.*` è influenzata dall'istruzione `c REMOVED`. L'istruzione `a TEXT` copia il testo indicato da `TEXT` in una nuova riga dopo la riga con una corrispondenza. L'istruzione `r FILE` fa lo stesso, ma copia il contenuto del file indicato da `FILE`. L'istruzione `w` fa l'opposto di `r`, cioè la riga verrà aggiunta al file indicato.

Di gran lunga l'istruzione `sed` più usata è `s/FIND/REPLACE/`, che viene usata per sostituire una corrispondenza all'espressione regolare `FIND` con il testo indicato da `REPLACE`. Per esempio, l'istruzione `s/hda/sda/` sostituisce una sottostringa che corrisponde al letterale RE `hda` con `sda`. Solo la prima corrispondenza trovata nella riga verrà sostituita, a meno che il *flag* `g` non sia posto dopo l'istruzione, come in `s/hda/sda/g`.

Un caso più realistico aiuterà a illustrare le caratteristiche di `sed`. Supponiamo che una clinica medica desideri inviare messaggi di testo ai propri clienti, ricordando loro gli appuntamenti programmati per il giorno successivo. Uno scenario di implementazione tipico si basa su un servizio di messaggistica istantanea professionale, che fornisce un'API per accedere al sistema responsabile della consegna dei messaggi. Questi messaggi di solito provengono dallo stesso sistema che esegue l'applicazione controllando gli appuntamenti del cliente, messaggi attivati a un momento specifico della giornata o da qualche specifico evento. In questa ipotetica situazione l'applicazione potrebbe generare un file chiamato `appointments.csv` contenente i dati tabulati con tutti gli appuntamenti per il giorno successivo, poi usato da `sed` per visualizzare i messaggi di testo da un file modello chiamato `template.txt`. I file CSV sono un modo standard per esportare i dati dalle query del database, quindi gli appuntamenti di esempio potrebbero essere forniti come segue:

```
$ cat appointments.csv
"NAME", "TIME", "PHONE"
"Carol", "11am", "55557777"
"Dave", "2pm", "33334444"
```

La prima riga contiene le etichette per ogni colonna, che verranno utilizzate per abbinare i tag all'interno del file modello di esempio:

```
$ cat template.txt
Hey <NAME>, don't forget your appointment tomorrow at <TIME>.
```

I segni minore di `<` e maggiore di `>` sono stati posti attorno alle etichette solo per aiutare a identificarli come tag. Il seguente script Bash analizza tutti gli appuntamenti accodati usando `template.txt` come modello di messaggio:

```
#!/bin/bash

TEMPLATE=`cat template.txt`
TAGS=(`sed -ne '1s/^"//;1s/"", "/\n/g;1s/"$//p' appointments.csv`)
mapfile -t -s 1 ROWS < appointments.csv
for (( r = 0; r < ${#ROWS[*]}; r++ ))
do
    MSG=$TEMPLATE
    VALS=(`sed -e 's/^"//;s/"", "/\n/g;s/"$//' <<<${ROWS[$r]}`)
    for (( c = 0; c < ${#TAGS[*]}; c++ ))
    do
        MSG=`sed -e "s/<${TAGS[$c]}>/${VALS[$c]}/g" <<<"$MSG"`
```



```
done
echo curl --data message=\"$MSG\" --data phone=\"${VALS[2]}\" https://mysmsprovider/api
done
```

Uno script in produzione gestirà anche l'autenticazione, il controllo degli errori e la registrazione, ma l'esempio ha funzionalità di base con cui iniziare. Le prime istruzioni eseguite da `sed` sono applicate solo alla prima riga: l'indirizzo 1 in `1s/^"//;1s/","/\n/g;1s/"$//p` — per rimuovere le virgolette iniziali e finali — `1s/^"//` e `1s/"$//` — e per sostituire i separatori di campo con un carattere di nuova riga: `1s /"," /\n/g`. Solo la prima riga è necessaria per caricare i nomi delle colonne, quindi le righe non corrispondenti verranno soppresse dall'opzione `-n`, richiedendo che il flag `p` sia posto dopo l'ultimo comando `sed` per stampare la riga corrispondente. I tag vengono quindi memorizzati nella variabile `TAGS` come un array Bash. Un'altra variabile dell'array Bash viene creata dal comando `mapfile` per memorizzare le righe contenenti gli appuntamenti nella variabile dell'array `ROWS`.

Un ciclo `for` viene utilizzato per elaborare ogni riga di appuntamento trovata in `ROWS`. Quindi, virgolette e separatori nell'appuntamento — l'appuntamento è nella variabile `${ROWS[$r]}` usata come *here string* — vengono sostituiti da `sed`, in modo simile ai comandi usati per caricare i tag. I valori separati per l'appuntamento vengono quindi memorizzati nella variabile di array `VALS`, dove gli indici di array 0, 1 e 2 corrispondono ai valori di `NAME`, `TIME` e `PHONE`.

Infine, un ciclo annidato `for` percorre l'array `TAGS` e sostituisce ogni tag trovato nel modello con il valore corrispondente in `VALS`. La variabile `MSG` contiene una copia del modello renderizzato, aggiornato dal comando di sostituzione `s/<${TAGS[$ c]}>/${VALS[$c]}/g` ad ogni ciclo che passa attraverso `TAGS`.

Ciò si traduce in un messaggio visualizzato come: "Ehi Carol, non dimenticare il tuo appuntamento domani alle 11:00." Il messaggio visualizzato può quindi essere inviato come parametro tramite una richiesta HTTP con `curl`, come messaggio di posta elettronica o un altro metodo simile

Combinare grep e sed

I comandi `grep` e `sed` possono essere usati insieme quando sono richieste procedure di *text mining* più complesse. In qualità di amministratore di sistema, potresti voler controllare tutti i tentativi di accesso a un server. Per esempio, il file `/var/log/wtmp` registra tutti i login e logout, mentre il file `/var/log/btmp` registra i tentativi di login falliti. Sono scritti in un formato binario, che può essere letto rispettivamente dai comandi `last` e `lastb`.

L'output di `lastb` mostra non solo il nome utente utilizzato nel tentativo di accesso errato, ma anche il suo indirizzo IP:

```
# lastb -d -a -n 10 --time-format notime
user      ssh:notty      (00:00)      81.161.63.251
nrostagn  ssh:notty      (00:00)      vmd60532.contaboserver.net
pi        ssh:notty      (00:00)      132.red-88-20-39.staticip.rima-tde.net
pi        ssh:notty      (00:00)      132.red-88-20-39.staticip.rima-tde.net
pi        ssh:notty      (00:00)      46.6.11.56
pi        ssh:notty      (00:00)      46.6.11.56
nps       ssh:notty      (00:00)      vmd60532.contaboserver.net
narmadan  ssh:notty      (00:00)      vmd60532.contaboserver.net
nominati  ssh:notty      (00:00)      vmd60532.contaboserver.net
nominati  ssh:notty      (00:00)      vmd60532.contaboserver.net
```

L'opzione `-d` traduce l'indirizzo IP nel nome host corrispondente. Il nome host può fornire indizi sull'ISP o sul servizio di hosting utilizzato per eseguire questi tentativi di accesso errati. L'opzione `-a` inserisce il nome host nell'ultima colonna, il che facilita il filtraggio ancora da applicare. L'opzione `--time-format notime` sopprime l'ora in cui si è verificato il tentativo di accesso. Il comando `lastb` può richiedere del tempo per terminare se ci sono stati troppi tentativi di accesso errati, quindi l'output è stato limitato a dieci voci con l'opzione `-n 10`.

Non tutti gli IP remoti hanno un nome host associato, quindi il DNS inverso non si applica a loro e possono essere ignorati. Sebbene sia possibile scrivere un'espressione regolare che corrisponda al formato previsto per un nome host alla fine della riga, è probabilmente più semplice scrivere un'espressione regolare da abbinare a una lettera dell'alfabeto o a una singola cifra alla fine del linea. L'esempio seguente mostra come il comando `grep` prende l'elenco nel suo standard input e rimuove le righe senza nomi host:

```
# lastb -d -a --time-format notime | grep -v '[0-9]$' | head -n 10
nvidia    ssh:notty      (00:00)      vmd60532.contaboserver.net
n_tonson  ssh:notty      (00:00)      vmd60532.contaboserver.net
nrostagn  ssh:notty      (00:00)      vmd60532.contaboserver.net
pi        ssh:notty      (00:00)      132.red-88-20-39.staticip.rima-tde.net
pi        ssh:notty      (00:00)      132.red-88-20-39.staticip.rima-tde.net
nps       ssh:notty      (00:00)      vmd60532.contaboserver.net
narmadan  ssh:notty      (00:00)      vmd60532.contaboserver.net
nominati  ssh:notty      (00:00)      vmd60532.contaboserver.net
nominati  ssh:notty      (00:00)      vmd60532.contaboserver.net
nominati  ssh:notty      (00:00)      vmd60532.contaboserver.net
```

L'opzione `-v` del comando `grep` mostra solo le righe che non corrispondono all'espressione regolare data. Un'espressione regolare che corrisponde a qualsiasi riga che termina con un numero (per esempio `[0-9]$`) catturerà solo le voci senza un nome host. Pertanto, `grep -v '[0-`

9]\$' mostrerà solo le righe che finiscono con un nome host.

L'output può essere ulteriormente filtrato, mantenendo solo il nome di dominio e rimuovendo le altre parti da ogni riga. Il comando `sed` può farlo con un comando di sostituzione per modificare l'intera riga con un riferimento al nome di dominio in essa contenuto:

```
# lastb -d -a --time-format notime | grep -v '[0-9]$' | sed -e 's/.* \(.*)$/\1/' | head -n 10
vmd60532.contaboserver.net
vmd60532.contaboserver.net
vmd60532.contaboserver.net
132.red-88-20-39.staticip.rima-tde.net
132.red-88-20-39.staticip.rima-tde.net
vmd60532.contaboserver.net
vmd60532.contaboserver.net
vmd60532.contaboserver.net
vmd60532.contaboserver.net
vmd60532.contaboserver.net
```

La parentesi con *escape* in `. * \(. * \)` indica a `sed` di ricordare quella parte della riga, cioè la parte tra l'ultimo carattere di spazio e la fine della riga. Nell'esempio, questa parte viene referenziata con `\ 1` e viene utilizzata la sostituzione dell'intera riga.

È chiaro che la maggior parte degli host remoti tenta di accedere più di una volta, quindi lo stesso nome di dominio si ripete. Per sopprimere le voci ripetute, prima devono essere ordinate (con il comando `sort`) quindi passate al comando `uniq`:

```
# lastb -d -a --time-format notime | grep -v '[0-9]$' | sed -e 's/.* \(.*)$/\1/' | sort | uniq | head -n 10
116-25-254-113-on-nets.com
132.red-88-20-39.staticip.rima-tde.net
145-40-33-205.power-speed.at
tor.laquadrature.net
tor.momx.site
ua-83-226-233-154.bbcust.telenor.se
vmd38161.contaboserver.net
vmd60532.contaboserver.net
vmi488063.contaboserver.net
vmi515749.contaboserver.net
```

Questo mostra come sia possibile combinare diversi comandi per produrre il risultato desiderato.

L'elenco dei nomi host può quindi essere utilizzato per scrivere regole di blocco del firewall o per adottare altre misure per rafforzare la sicurezza del server.

Esercizi Guidati

1. Il comando `last` mostra un elenco degli ultimi utenti che hanno effettuato un accesso sul sistema, inclusi i loro IP di origine. Come verrebbe usato il comando `egrep` per filtrare l'output di `last`, mostrando solo le occorrenze di un indirizzo IPv4, scartando qualsiasi informazione aggiuntiva nella riga corrispondente?

2. Quale opzione dovrebbe essere data a `grep` per filtrare correttamente l'output generato dal comando `find` eseguito con l'opzione `-print0`?

3. Il comando `uptime -s` mostra l'ultima data in cui il sistema è stato acceso, come in `2019-08-05 20:13:22`. Quale sarà il risultato del comando `uptime -s | sed -e 's/(.*) (.*)/\1/'`?

4. Quale opzione dovrebbe essere data a `grep` in modo che conti le linee corrispondenti invece di visualizzarle?

Esercizi Esplorativi

1. La struttura di base di un file HTML inizia con gli elementi `html`, `head` e `body`, per esempio:

```
<html>
<head>
  <title>News Site</title>
</head>
<body>
  <h1>Headline</h1>
  <p>Information of interest.</p>
</body>
</html>
```

Descrivi come gli indirizzi potrebbero essere usati in `sed` per visualizzare solo l'elemento `body` e il suo contenuto.

2. Quale espressione di `sed` rimuoverà tutti i tag da un documento HTML, mantenendo solo il testo visualizzato?

3. I file con estensione `.ovpn` sono molto popolari per configurare i client VPN poiché contengono non solo le impostazioni, ma anche il contenuto delle chiavi e dei certificati per il client. Queste chiavi e certificati sono originariamente in file separati, quindi devono essere copiati nel file `.ovpn`. Dato il seguente estratto di un modello `.ovpn`:

```
client
dev tun
remote 192.168.1.155 1194
<ca>
ca.crt
</ca>
<cert>
client.crt
</cert>
<key>
client.key
</key>
<tls-auth>
ta.key
```

```
</tls-auth>
```

Supponendo che i file `ca.crt`, `client.crt`, `client.key` e `ta.key` si trovino nella directory corrente, come verrebbe modificata la configurazione del modello di `sed` per sostituire ogni nome di file con il suo contenuto?

Sommario

Questa lezione copre i due comandi Linux più importanti relativi alle espressioni regolari: `grep` e `sed`. Script e comandi composti si basano su `grep` e `sed` per eseguire un'ampia gamma di operazioni di filtraggio e analisi del testo. La lezione segue i seguenti passaggi

- Come usare `grep` e le sue varianti come `egrep` e `fgrep`.
- Come usare `sed` e le sue istruzioni interne per manipolare il testo.
- Esempi di applicazioni di espressioni regolari che utilizzano `grep` e `sed`.

Risposte agli Esercizi Guidati

1. Il comando `last` mostra un elenco degli ultimi utenti che hanno effettuato un accesso sul sistema, inclusi i loro IP di origine. Come verrebbe usato il comando `egrep` per filtrare l'output di `last`, mostrando solo le occorrenze di un indirizzo IPv4, scartando qualsiasi informazione aggiuntiva nella riga corrispondente?

```
last -i | egrep -o '[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}\.[0-9]{1,3}'
```

2. Quale opzione dovrebbe essere data a `grep` per filtrare correttamente l'output generato dal comando `find` eseguito con l'opzione `-print0`?

L'opzione `-z` o `--null-data`, come in `find . -print0 | grep -z expression`.

3. Il comando `uptime -s` mostra l'ultima data in cui il sistema è stato acceso, come in `2019-08-05 20:13:22`. Quale sarà il risultato del comando `uptime -s | sed -e 's/(.*) (.*)/\1/'`?

Si verificherà un errore. Per impostazione predefinita, le parentesi dovrebbero essere precedute da caratteri di escape per utilizzare i riferimenti a ritroso in `sed`.

4. Quale opzione dovrebbe essere data a `grep` in modo che conti le linee corrispondenti invece di visualizzarle?

L'opzione `-c`.

Risposte agli Esercizi Esplorativi

1. La struttura di base di un file HTML inizia con gli elementi `html`, `head` e `body`, per esempio:

```
<html>
<head>
  <title>News Site</title>
</head>
<body>
  <h1>Headline</h1>
  <p>Information of interest.</p>
</body>
</html>
```

Descrivi come gli indirizzi potrebbero essere usati in `sed` per visualizzare solo l'elemento `body` e il suo contenuto.

Per mostrare solo `body`, gli indirizzi dovrebbero essere `/<body>/, /<\body>/`, come in `sed -n -e '/<body>/, /<\body>/p'`. L'opzione `-n` è data a `sed` quindi non stampa le righe per impostazione predefinita, da qui il comando `p` alla fine dell'espressione `sed` per stampare le righe corrispondenti.

2. Quale espressione di `sed` rimuoverà tutti i tag da un documento HTML, mantenendo solo il testo visualizzato?

L'espressione `sed s/<[^>]*>//g` sostituirà qualsiasi contenuto racchiuso in `<>` con una stringa vuota.

3. I file con estensione `.ovpn` sono molto popolari per configurare i client VPN poiché contengono non solo le impostazioni, ma anche il contenuto delle chiavi e dei certificati per il client. Queste chiavi e certificati sono originariamente in file separati, quindi devono essere copiati nel file `.ovpn`. Dato il seguente estratto di un modello `.ovpn`:

```
client
dev tun
remote 192.168.1.155 1194
<ca>
ca.crt
</ca>
<cert>
client.crt
```

```
</cert>
<key>
client.key
</key>
<tls-auth>
ta.key
</tls-auth>
```

Supponendo che i file `ca.crt`, `client.crt`, `client.key` e `ta.key` si trovino nella directory corrente, come verrebbe modificata la configurazione del modello di `sed` per sostituire ogni nome di file con il suo contenuto?

Il comando

```
sed -r -e 's/([^.]*)\.(crt|key)$/cat \1.\2/e' < client.template > client.ovpn
```

sostituisce qualsiasi riga che termini in `.crt` o `.key` con il contenuto di un file il cui nome è uguale alla riga. L'opzione `-r` dice a `sed` di usare espressioni regolari estese, mentre `e` alla fine dell'espressione dice a `sed` di sostituire le corrispondenze con l'output del comando `cat \1.\2`. I riferimenti a ritroso `\1` e `\2` corrispondono al nome del file e all'estensione trovati nella corrispondenza.



103.8 Modifica base di un file

Obiettivi LPI di riferimento

LPIC-1 v5, Exam 101, Objective 103.8

Peso

3

Aree di Conoscenza Chiave

- Muoversi in un documento utilizzando vi.
- Comprendere e utilizzare le modalità di vi.
- Inserire, modificare, eliminare, copiare e trovare testo in vi.
- Conoscenza degli editor Emacs, nano e vim.
- Configurare l'editor predefinito.

Di seguito è riportato un elenco parziale dei file, dei termini e dei comandi utilizzati

- vi
- /, ?
- h, j, k, l
- i, o, a
- d, p, y, dd, yy
- ZZ, :w!, :q!
- EDITOR



Linux
Professional
Institute

103.8 Lezione 1

Certificazione:	LPIC-1
Versione:	5.0
Argomento:	103 Comandi GNU e Unix
Obiettivo:	103.8 Modifica di file di base
Lezione:	1 di 1

Introduzione

Nella maggior parte delle distribuzioni Linux, `vi`, abbreviazione di “visual”, è preinstallato ed è l'editor standard nell'ambiente shell. `vi` è un editor di testo interattivo, mostra il contenuto del file sullo schermo mentre viene modificato. In quanto tale, consente all'utente di spostarsi e di apportare modifiche in qualsiasi punto del documento. Tuttavia, a differenza degli editor visuali dal desktop grafico, l'editor `vi` è un'applicazione shell con scorciatoie da tastiera per ogni attività di modifica.

Un'alternativa a `vi`, chiamata `vim` (*vi improved*), è talvolta usata come un più moderno sostituto di `vi`. Tra gli altri miglioramenti, `vim` offre il supporto per l'evidenziazione della sintassi, l'annullamento / ripristino multilivello e la modifica di più documenti. Sebbene più ingegnoso, `vim` è completamente retrocompatibile con `vi`, il che li rende pressoché indistinguibili per la maggior parte delle attività.

Il modo standard per avviare `vi` è di dargli un percorso a un file come parametro. Per passare direttamente a una riga specifica, il suo numero dovrebbe essere indicato con un segno più, come in `vi +9 /etc/fstab` per aprire `/etc/fstab` / e posizionare il cursore sulla nona riga. Senza un numero, il segno più da solo, posiziona il cursore sull'ultima riga.

L'interfaccia di `vi` è molto semplice: tutto lo spazio disponibile nella finestra del terminale è occupato per presentare all'utente un file, normalmente informato come argomento di comando. Gli unici indizi visivi sono una riga a piè di pagina che mostra la posizione corrente del cursore e una tilde `~` che indica dove finisce il file. Ci sono diverse modalità di esecuzione per `vi` in cui il comportamento del programma cambia. Le più comuni sono: *insert mode* e *normal mode*.

Insert Mode

La *insert mode* è semplice: il testo appare sullo schermo mentre viene digitato sulla tastiera. È il tipo di interazione che la maggior parte degli utenti si aspetta da un editor di testo, ma non è il modo in cui `vi` presenta un documento per la prima volta. Per entrare in modalità di inserimento, l'utente deve eseguire un comando di inserimento in modalità normale. Il tasto `Esc` termina la modalità di inserimento e torna alla *normal mode*, la modalità predefinita `vi`.

NOTE

Se sei interessato a saperne di più sulle altre modalità di esecuzione, apri `vi` e digita:

```
:help vim-modes-intro
```

Normal Mode

Normal mode — conosciuta anche come *command mode* — è il modo in cui `vi` si avvia di default. In questa modalità i tasti della tastiera sono associati ai comandi per la navigazione e le attività di manipolazione del testo. La maggior parte dei comandi in questa modalità sono tasti univoci. Alcuni dei tasti e le loro funzioni in modalità normale sono:

0, \$

Vai all'inizio e alla fine della riga.

1G, G

Vai all'inizio e alla fine del documento.

(,)

Vai all'inizio e alla fine della frase.

{, }

Vai all'inizio e alla fine del paragrafo.

w, W

Salta di parola in parola inclusa la punteggiatura.

h, j, k, l

A sinistra, in basso, in alto, a destra.

e or E

Vai alla fine della parola corrente.

/, ?

Cerca avanti e indietro.

i, I

Accedi alla modalità di inserimento prima della posizione corrente del cursore e all'inizio della riga corrente.

a, A

Accedi alla modalità di inserimento dopo la posizione corrente del cursore e alla fine della riga corrente.

o, O

Aggiungi una nuova riga e accedere alla modalità di inserimento nella riga successiva o nella riga precedente.

s, S

Cancella il carattere sotto il cursore o l'intera riga e accedi alla modalità di inserimento.

c

Cambia i caratteri sotto il cursore.

r

Sostituisci il carattere sotto il cursore.

x

Elimina i caratteri selezionati o il carattere sotto il cursore.

v, V

Inizia una nuova selezione con il carattere corrente o l'intera riga.

y, yy

Copia (*yank*) i caratteri o l'intera riga.

p, P

Incolla (*paste*) il contenuto copiato, dopo o prima della posizione corrente.

u

Annulla l'ultima azione.

Ctrl-R

Ripeti l'ultima azione.

ZZ

Salva e chiudi.

ZQ

Chiudi senza salvare.

Se preceduto da un numero, il comando verrà eseguito lo stesso numero di volte. Per esempio, premere `3yy` per copiare la riga corrente più le due seguenti, premere `d5w` per eliminare la parola corrente e le seguenti 4 parole e così via.

La maggior parte delle attività di modifica risulta da combinazioni di più comandi. Per esempio, la sequenza di tasti `vey` viene utilizzata per copiare una selezione a partire dalla posizione corrente fino alla fine della parola corrente. La ripetizione dei comandi può anche essere usata in combinazioni, quindi `v3ey` copia una selezione a partire dalla posizione corrente fino alla fine della terza parola da lì.

`vi` può organizzare il testo copiato in registri, consentendo di mantenere contenuti distinti allo stesso tempo. Un registro è specificato da un carattere preceduto da `"` e una volta creato viene mantenuto fino alla fine della sessione corrente. La sequenza di tasti `"ly` crea un registro contenente la selezione corrente, che sarà accessibile tramite il tasto `l`. Quindi, il registro `|` può essere incollato con `"lp`.

C'è anche un modo per impostare segni personalizzati in posizioni arbitrarie lungo il testo, rendendo più facile passare rapidamente da uno all'altro. I segni vengono creati premendo il tasto `m` e quindi un tasto per indirizzare la posizione corrente. Fatto ciò, il cursore tornerà alla posizione contrassegnata quando vengono premuti `'` seguito dal tasto scelto.

Qualsiasi sequenza di tasti può essere registrata come macro per l'esecuzione futura. È possibile registrare una macro, per esempio, per racchiudere un testo selezionato tra virgolette doppie.

Innanzitutto, viene selezionata una stringa di testo e viene premuto il tasto `q`, seguito da un tasto di registro a cui associare la macro, come `d`. La riga `recording @d` apparirà nella riga a piè di pagina, indicando che la registrazione è attiva. Si presume che del testo sia già selezionato, quindi il primo comando è `x` per rimuovere (e copiare automaticamente) il testo selezionato. Il tasto `i` viene premuto per inserire due virgolette doppie nella posizione corrente, quindi `Esc` ritorna alla modalità normale. L'ultimo comando è `P`, per reinserire la selezione eliminata appena prima dell'ultimo doppio apice. Premendo di nuovo `q` la registrazione viene terminata. Ora, una macro composta dalla sequenza di tasti `x`, `i`, `"`, `Esc` e `P` verrà eseguita ogni volta che i tasti `@d` vengono premuti in modalità normale, dove `d` è la chiave di registro associata alla macro.

Tuttavia, la macro sarà disponibile solo durante la sessione corrente. Per rendere persistenti le macro è necessario memorizzarle nel file di configurazione. Poiché la maggior parte delle distribuzioni moderne usa `vim` come editor compatibile con `vi`, il file di configurazione dell'utente è `~/.vimrc`. All'interno di `~/.vimrc`, la riga `let @d = 'xi""^[P'` imposterà il registro `d` alla sequenza di tasti tra virgolette singole. Lo stesso registro precedentemente assegnato a una macro può essere utilizzato per incollare la sua sequenza di tasti.

Comandi dei Due Punti

La modalità normale supporta anche un altro set di comandi `vi`: i *colon commands*. I comandi dei due punti, come suggerisce il nome, vengono eseguiti dopo aver premuto il tasto due punti `:` in modalità normale. I comandi dei due punti consentono all'utente di eseguire ricerche, salvare, uscire, eseguire comandi della shell, modificare le impostazioni di `vi`, ecc. Per tornare alla modalità normale, deve essere eseguito il comando `:visual` o il tasto `ENTER` premuto senza alcun comando. Alcuni dei comandi dei due punti più comuni sono indicati qui (l'iniziale non fa parte del comando):

`:s/REGEX/TEXT/g`

Sostituisce tutte le occorrenze dell'espressione regolare `REGEX` con `TEXT` nella riga corrente. Accetta la stessa sintassi del comando `sed`, inclusi gli indirizzi.

`:!`

Esegui un seguente comando di shell.

`:quit o :q`

Esce dal programma.

`:quit! o :q!`

Esce dal programma senza salvare.

:wq

Esce dal programma dopo aver salvato.

:exit or :x or :e

Salva e esce, se necessario.

:visual

Torna alla *navigation mode*.

Il programma standard **vi** è in grado di eseguire la maggior parte delle attività di modifica del testo, ma qualsiasi altro editor non grafico può essere utilizzato per modificare i file di testo nell'ambiente shell.

TIP

Gli utenti inesperti potrebbero avere difficoltà a memorizzare i tasti di comando di **vi** tutti in una volta. Le distribuzioni che adottano **vim** hanno anche il comando **vimtutor**, che usa **vim** stesso per aprire una guida passo passo alle principali attività. Il file è una copia modificabile che può essere utilizzata per esercitarsi con i comandi e progressivamente abituarsi a loro.

Editor Alternativi

Gli utenti che non hanno familiarità con **vi** potrebbero avere difficoltà ad adattarlo, poiché il suo funzionamento non è intuitivo. Un'alternativa più semplice è GNU **nano**, un agile editor di testo che offre tutte le funzionalità di base per la modifica del testo come annulla/ripristina, colorazione della sintassi, ricerca e sostituzione interattiva, rientro automatico, numeri di riga, completamento delle parole, blocco dei file, file di backup e supporto all'internazionalizzazione. A differenza di **vi**, tutti i tasti premuti vengono semplicemente inseriti nel documento in fase di modifica. I comandi in **nano** sono dati usando il tasto **Ctrl** o il Meta tasto (a seconda del sistema, Meta sarà **Alt** o **⌘**).

Ctrl-6 o Meta-A

Inizia una nuova selezione. È anche possibile creare una selezione premendo Shift e spostando il cursore.

Meta-6

Copia la selezione corrente.

Ctrl-K

Taglia la selezione corrente.

Ctrl-U

Incolla il contenuto precedentemente copiato.

Meta-U

Undo.

Meta-E

Redo.

Ctrl-

Sostituisci il testo nella selezione.

Ctrl-T

Avvia una sessione di controllo ortografico per il documento o la selezione corrente.

Emacs è un altro editor di testo molto popolare per l'ambiente shell. Mentre il testo viene inserito semplicemente digitandolo, come in `nano`, la navigazione nel documento è assistita dai comandi della tastiera, come in `vi`. Emacs include molte funzionalità che lo rendono più di un semplice editor di testo. È anche un IDE (*integrated development environment*) in grado di compilare, eseguire e testare programmi. Emacs può essere configurato come client di posta elettronica, news o RSS, rendendolo un'autentica suite di produttività.

La shell stessa eseguirà un editor di testo predefinito, di solito `vi`, ogni volta che è necessario. Questo è il caso, per esempio, quando viene eseguito `crontab -e` per modificare `cronjobs`. Bash usa le variabili di sessione `VISUAL` o `EDITOR` per trovare l'editor di testo predefinito per l'ambiente shell. Per esempio, il comando `export EDITOR = nano` definisce `nano` come l'editor di testo predefinito nella sessione di shell corrente. Per rendere questa modifica persistente tra le sessioni, il comando dovrebbe essere incluso in `~/.bash_profile`.

Esercizi Guidati

1. `vi` è usato soprattutto come editor per file di configurazione e codice sorgente, dove l'indentazione aiuta a identificare sezioni di testo. Una selezione può essere rientrata a sinistra premendo `<` e a destra premendo `>`. Quali tasti devono essere premuti in modalità normale per far rientrare la selezione corrente di tre passaggi a sinistra?

2. È possibile selezionare un'intera riga premendo `V` nella normal mode di `vi`. Tuttavia, è incluso anche il carattere di fine riga. Quali tasti dovrebbero essere premuti in normal mode per selezionare dal carattere iniziale fino al carattere di nuova riga, ma escluso?

3. Come dovrebbe essere eseguito `vi` nella riga di comando per aprire `~/.bash_profile` e saltare direttamente all'ultima riga?

4. Quali tasti dovrebbero essere premuti nella normal mode di `vi` per eliminare i caratteri dalla posizione corrente del cursore fino al successivo punto?

Esercizi Esplorativi

1. `vim` permette di selezionare blocchi di testo con larghezza arbitraria, non solo sezioni con intere righe. Premendo `Ctrl` + `V` in normal mode, viene effettuata una selezione spostando il cursore su, giù, sinistra e destra. Utilizzando questo metodo, come verrebbe cancellato un blocco che inizia dal primo carattere nella riga corrente, contenente le successive otto colonne e cinque righe di testo?
2. Una sessione `vi` è stata interrotta da un'interruzione di corrente imprevista. Quando si riapre il file, `vi` chiede all'utente se desidera recuperare lo swap file (una copia automatica fatta da `vi`). Cosa deve fare l'utente per eliminare il file di scambio?
3. In una sessione `vim`, una riga era stata precedentemente copiata nel registro `l`. Quale combinazione di tasti registrerebbe una macro nel registro `a` per incollare la riga nel registro `l` immediatamente prima della riga corrente?

Sommario

Questa lezione copre l'editor di testo standard per l'ambiente shell Linux: l'editor `vi`. Sebbene intimidisca l'utente non familiare con la riga di comando, `vi` ha caratteristiche che lo rendono una buona scelta per l'editing di testi tecnici e non. La lezione segue i seguenti passaggi:

- Utilizzo di base di `vi` e funzioni utili.
- Cos'è `vim` — il `vi` migliorato — e altri editor alternativi.
- Come definire l'editor di testo predefinito nell'ambiente shell.

I comandi e le procedure trattate erano:

- Editor `vi` e la sua versione migliorata `vim`.
- Modifica del testo di base in `vi`.
- Editor alternativi `emacs` e `nano`.

Risposte agli Esercizi Guidati

1. `vi` è usato soprattutto come editor per file di configurazione e codice sorgente, dove l'indentazione aiuta a identificare sezioni di testo. Una selezione può essere rientrata a sinistra premendo `<` e a destra premendo `>`. Quali tasti devono essere premuti in modalità normale per far rientrare la selezione corrente di tre passaggi a sinistra?

La sequenza `3<`, indica tre rientri a sinistra.

2. È possibile selezionare un'intera riga premendo `V` nella normal mode di `vi`. Tuttavia, è incluso anche il carattere di fine riga. Quali tasti dovrebbero essere premuti in normal mode per selezionare dal carattere iniziale fino al carattere di nuova riga, ma escluso?

La sequenza `0v$h`, che significa `0` (vai all'inizio di una riga), `v` (inizio selezione carattere), `$` (vai alla fine della riga) e `h` (torna indietro di una posizione).

3. Come dovrebbe essere eseguito `vi` nella riga di comando per aprire `~/.bash_profile` e saltare direttamente all'ultima riga?

Il comando `vi + ~/.bash_profile` aprirà il file e posizionerà il cursore sulla sua ultima riga.

4. Quali tasti dovrebbero essere premuti nella normal mode di `vi` per eliminare i caratteri dalla posizione corrente del cursore fino al successivo punto?

La sequenza `dt.`, che significa `d` (inizio cancellazione), `t` (salta al carattere successivo) e `.` (carattere punto).

Risposte agli Esercizi Esplorativi

1. `vim` permette di selezionare blocchi di testo con larghezza arbitraria, non solo sezioni con intere righe. Premendo `Ctrl` + `V` in normal mode, viene effettuata una selezione spostando il cursore su, giù, sinistra e destra. Utilizzando questo metodo, come verrebbe cancellato un blocco che inizia dal primo carattere nella riga corrente, contenente le successive otto colonne e cinque righe di testo?

La combinazione `O`, `Ctrl-V` e `815jd` selezionerà ed eliminerà il blocco corrispondente.

2. Una sessione `vi` è stata interrotta da un'interruzione di corrente imprevista. Quando si riapre il file, `vi` chiede all'utente se desidera recuperare lo swap file (una copia automatica fatta da `vi`). Cosa deve fare l'utente per eliminare il file di scambio?

Premere `d` quando richiesto da `vi`.

3. In una sessione `vim`, una riga era stata precedentemente copiata nel registro `l`. Quale combinazione di tasti registrerebbe una macro nel registro `a` per incollare la riga nel registro `l` immediatamente prima della riga corrente?

La combinazione `qa"lPq`, che significa `q` (avvia la registrazione della macro), `a` (assegna registro `a` alla macro), `"l` (seleziona il testo nel registro `l`), `P` (incolla prima della riga corrente) e `q` (fine della registrazione macro).



**Linux
Professional
Institute**

Argomento 104: Dispositivi, il File System Linux, il Filesystem Hierarchy Standard



104.1 Creare partizioni e filesystem

Obiettivi LPI di riferimento

LPIC-1 v5, Exam 101, Objective 104.1

Peso

2

Aree di Conoscenza Chiave

- Gestire le tabelle delle partizioni MBR e GPT
- Usare vari comandi mkfs per creare vari filesystem come:
 - ext2/ext3/ext4
 - XFS
 - VFAT
 - exFAT
- Conoscenza delle caratteristiche di base di Btrfs, inclusi filesystem multi-dispositivo, compressione e sottovolumi.

Di seguito è riportato un elenco parziale dei file, dei termini e dei comandi utilizzati

- fdisk
- gdisk
- parted
- mkfs
- mkswap



Linux
Professional
Institute

104.1 Lezione 1

Certificazione:	LPIC-1
Versione:	5.0
Argomento:	104 Dispositivi, Il Filesystem Linux, Filesystem Hierarchy Standard
Obiettivo:	104.1 Creare partizioni e filesystem
Lezione:	1 di 1

Introduzione

Su qualsiasi sistema operativo, un disco deve essere partizionato prima di poter essere utilizzato. Una partizione è un sottoinsieme logico del disco fisico; le informazioni sulle partizioni sono archiviate in una tabella delle partizioni. Questa tabella include informazioni sul primo e sull'ultimo settore della partizione e sul suo tipo e ulteriori dettagli su ciascuna partizione.

Di solito ogni partizione è vista da un sistema operativo come un “disco” separato, anche se risiedono tutte nello stesso supporto fisico. Su sistemi Windows vengono assegnate lettere come C: (storicamente il disco principale), D: e così via. Su Linux ogni partizione è assegnata a una directory sotto `/dev`, come `/dev/sda1` o `/dev/sda2`.

In questa lezione imparerai come creare, eliminare, ripristinare e ridimensionare le partizioni usando le tre utilità più comuni (`fdisk`, `gdisk` e `parted`), come creare un filesystem su di esse e come creare e impostare una *partizione di swap* o un *file di swap* da utilizzare come memoria virtuale.

NOTE

Per ragioni storiche, in questa lezione faremo riferimento ai supporti di

archiviazione come “dischi”, anche se i sistemi di archiviazione moderni, come gli SSD e i *Flash Storage*, non contengono affatto “dischi”.

Comprendere MBR e GPT

Esistono due modi principali per memorizzare le informazioni delle partizioni sui dischi rigidi. Il primo è MBR (*Master Boot Record*) e il secondo è GPT (*GUID Partition Table*)

MBR

Questo è un residuo degli albori di MS-DOS (più specificamente, PC-DOS 2.0 del 1983) e per decenni è stato lo schema di partizionamento standard sui PC. La tabella delle partizioni è memorizzata sul primo settore di un disco, chiamato *Boot Sector*, insieme a un boot loader, che sui sistemi Linux è solitamente il bootloader *GRUB*. Ma MBR ha una serie di limitazioni che ne ostacolano l'uso sui sistemi moderni, come l'incapacità di indirizzare dischi di dimensioni superiori a 2 TB e il limite di sole 4 partizioni primarie per disco.

GUID

Un sistema di partizionamento che risolve molti dei limiti di MBR. Non esiste un limite pratico alla dimensione del disco e il numero massimo di partizioni è limitato solo dal sistema operativo stesso. Si trova più comunemente su macchine più moderne che utilizzano UEFI invece del vecchio BIOS del PC.

Durante le attività di amministrazione del sistema è altamente probabile che troverai entrambi gli schemi in uso, quindi è importante sapere come utilizzare gli strumenti associati a ciascuno per creare, eliminare o modificare le partizioni.

Gestire Partizioni MBR con FDISK

L'utilità standard per la gestione delle partizioni MBR su Linux è `fdisk`. Questa è un'utilità interattiva basata su menu. Per usarla digita `fdisk` e il nome del dispositivo corrispondente al disco che desideri modificare. Per esempio, il comando

```
# fdisk /dev/sda
```

modifica la tabella delle partizioni del primo dispositivo connesso a SATA (`sda`) sul sistema. Tieni presente che devi specificare il dispositivo corrispondente al disco fisico, non una delle sue partizioni (come `/dev/sda1`).

NOTE

Tutte le operazioni relative al disco in questa lezione devono essere eseguite come utente `root` (l'amministratore di sistema), o con privilegi di root usando `sudo`.

Quando invocato, `fdisk` mostrerà un saluto, quindi un avviso e attenderà i tuoi comandi.

```
# fdisk /dev/sda
Welcome to fdisk (util-linux 2.33.1).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.

Command (m for help):
```

L'avvertimento è importante. È possibile creare, modificare o eliminare partizioni a piacere, ma *nulla* verrà scritto su disco a meno che non si utilizzi il comando `write` (`w`). Quindi puoi “esercitarti” senza il rischio di perdere dati, a patto di stare lontano dal tasto `w`. Per uscire da `fdisk` senza salvare le modifiche, utilizzare il comando `q`.

NOTE

Detto questo, non dovresti mai esercitarti su un disco importante, poiché ci sono sempre dei rischi. Utilizzare invece un disco esterno di riserva o un'unità flash USB.

Stampare la Tabella delle Partizioni Corrente

Il comando `p` viene utilizzato per stampare la tabella delle partizioni corrente. L'output è qualcosa del genere:

```
Command (m for help): p
Disk /dev/sda: 111.8 GiB, 120034123776 bytes, 234441648 sectors
Disk model: CT120BX500SSD1
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x97f8fef5

Device      Boot      Start          End      Sectors   Size Id Type
/dev/sda1                4096 226048942 226044847 107.8G 83 Linux
/dev/sda2           226048944 234437550   8388607    4G 82 Linux swap / Solaris
```

Ecco il significato di ogni colonna:

Device

Il dispositivo assegnato alla partizione.

Boot

Mostra se la partizione è “avviabile” o meno.

Start

Il settore in cui inizia la partizione.

End

Il settore in cui finisce la partizione.

Sectors

Il numero totale di settori nella partizione. Moltiplicalo per la dimensione del settore per ottenere la dimensione della partizione in byte.

Size

La dimensione della partizione nel formato “human readable”. Nell’esempio sopra, i valori sono in gigabyte.

Id

Il valore numerico che rappresenta il tipo di partizione.

Type

La descrizione per il tipo di partizione.

Partizioni Primarie ed Estese a Confronto

Su un disco MBR puoi avere 2 tipi principali di partizioni, *primarie* ed *estese*. Come abbiamo detto prima, puoi avere solo 4 partizioni primarie sul disco, e se vuoi rendere il disco “avviabile”, la prima partizione deve essere primaria.

Un modo per aggirare questa limitazione è creare una partizione estesa che funge da contenitore per le partizioni *logiche*. Si potrebbe avere, per esempio, una partizione primaria, una partizione estesa che occupa il resto dello spazio su disco e cinque partizioni logiche al suo interno.

Per un sistema operativo come Linux le partizioni primarie ed estese sono trattate esattamente allo stesso modo, quindi non ci sono “vantaggi” nell’usare una rispetto all’altra.

Creare una Partizione

Per creare una partizione usa il comando `n`. Per impostazione predefinita le partizioni verranno create all’inizio dello spazio non allocato sul disco. Ti verrà chiesto il tipo di partizione (primaria o estesa), primo settore e ultimo settore.

Per il primo settore, di solito puoi accettare il valore predefinito suggerito da `fdisk`, a meno che tu non abbia bisogno di una partizione per iniziare in un settore specifico. Invece di specificare l'ultimo settore, è possibile specificare una dimensione seguita dalle lettere K, M, G, T o P (Kilo, Mega, Giga, Tera o Peta). Quindi, se vuoi creare una partizione da 1 GB, puoi specificare `+1G` come Ultimo settore e `fdisk` ridimensionerà la partizione di conseguenza. Vedi questo esempio per la creazione di una partizione primaria:

```
Command (m for help): n
Partition type
   p   primary (0 primary, 0 extended, 4 free)
   e   extended (container for logical partitions)
Select (default p): p
Partition number (1-4, default 1): 1
First sector (2048-3903577, default 2048): 2048
Last sector, +/-sectors or +/-size{K,M,G,T,P} (2048-3903577, default 3903577): +1G
```

Verifica dello Spazio non Allocato

Se non sai quanto spazio libero c'è sul disco, puoi usare il comando `F` per mostrare lo spazio non allocato, in questo modo:

```
Command (m for help): F
Unpartitioned space /dev/sdd: 881 MiB, 923841536 bytes, 1804378 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes

   Start      End Sectors  Size
2099200 3903577 1804378  881M
```

Cancellare Partizioni

Per eliminare una partizione usa il comando `d`. `fdisk` ti chiederà il numero della partizione che desideri eliminare, *a meno che non ci sia una sola partizione sul disco*. In questo caso, questa partizione verrà *selezionata ed eliminata immediatamente*.

Tieni presente che se elimini una partizione estesa, verranno eliminate anche tutte le partizioni logiche al suo interno.

Attenzione allo Spazio!

Tieni presente che quando crei una nuova partizione con `fdisk`, la dimensione massima sarà

limitata alla quantità massima di spazio *contiguo* non allocato sul disco. Supponiamo, per esempio, di avere la seguente tabella delle partizioni:

Device	Boot	Start	End	Sectors	Size	Id	Type
/dev/sdd1		2048	1050623	1048576	512M	83	Linux
/dev/sdd2		1050624	2099199	1048576	512M	83	Linux
/dev/sdd3		2099200	3147775	1048576	512M	83	Linux

Quindi elimini la partizione 2 e controlla lo spazio libero:

```
Command (m for help): F
Unpartitioned space /dev/sdd: 881 MiB, 923841536 bytes, 1804378 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes

   Start      End Sectors  Size
1050624 2099199 1048576  512M
3147776 3903577  755802  369M
```

Sommando la dimensione dello spazio non allocato, in teoria abbiamo 881 MB disponibili. Ma guarda cosa succede quando proviamo a creare una partizione da 700 MB:

```
Command (m for help): n
Partition type
   p   primary (2 primary, 0 extended, 2 free)
   e   extended (container for logical partitions)
Select (default p): p
Partition number (2,4, default 2): 2
First sector (1050624-3903577, default 1050624):
Last sector, +/-sectors or +/-size{K,M,G,T,P} (1050624-2099199, default 2099199): +700M
Value out of range.
```

Ciò accade perché il più grande spazio contiguo non allocato sul disco è il blocco di 512 MB che apparteneva alla partizione 2. La tua nuova partizione non può “oltrepassare” la partizione 3 per utilizzare parte dello spazio non allocato dopo di essa.

Cambiare il Tipo di Partizione

Occasionalmente, potrebbe essere necessario modificare il tipo di partizione, soprattutto quando si tratta di dischi che verranno utilizzati su altri sistemi operativi e piattaforme. Questo viene fatto

con il comando `t`, seguito dal numero della partizione che desideri modificare.

Il tipo di partizione deve essere specificato dal suo codice esadecimale corrispondente e puoi vedere un elenco di tutti i codici validi usando il comando `l`.

Non confondere il tipo di partizione con il filesystem utilizzato su di essa. Sebbene all'inizio ci fosse una relazione tra loro, oggi non puoi presumere che sia così. Una partizione Linux, per esempio, può contenere qualsiasi file system nativo di Linux, come *ext4* o *ReiserFS*.

TIP

Le partizioni Linux sono di tipo 83 (Linux). Le partizioni di swap sono di tipo 82 (Linux Swap).

Gestire Partizioni GUID con GDISK

L'utilità `gdisk` è l'equivalente di `fdisk` quando si ha a che fare con dischi partizionati GPT. In effetti, l'interfaccia è modellata su `fdisk`, con un prompt interattivo e gli stessi (o molto simili) comandi.

Stampare la Tabella delle Partizioni Corrente

Il comando `p` viene utilizzato per stampare la tabella delle partizioni corrente. L'output è qualcosa del genere:

```
Command (? for help): p
Disk /dev/sdb: 3903578 sectors, 1.9 GiB
Model: DataTraveler 2.0
Sector size (logical/physical): 512/512 bytes
Disk identifier (GUID): AB41B5AA-A217-4D1E-8200-E062C54285BE
Partition table holds up to 128 entries
Main partition table begins at sector 2 and ends at sector 33
First usable sector is 34, last usable sector is 3903544
Partitions will be aligned on 2048-sector boundaries
Total free space is 1282071 sectors (626.0 MiB)
```

Number	Start (sector)	End (sector)	Size	Code	Name
1	2048	2099199	1024.0 MiB	8300	Linux filesystem
2	2623488	3147775	256.0 MiB	8300	Linux filesystem

Fin dall'inizio, notiamo alcune cose diverse:

- Ogni disco ha un identificatore disco (GUID) univoco. Si tratta di un numero esadecimale a 128 bit, assegnato in modo casuale quando viene creata la tabella delle partizioni. Poiché ci sono 3.4

$\times 10^{38}$ valori possibili per questo numero, le possibilità che 2 dischi casuali abbiano lo stesso GUID sono piuttosto scarse. Il GUID può essere usato per identificare quali filesystem montare all'avvio (e dove), eliminando la necessità di usare il percorso del dispositivo per farlo (come `/dev/sdb`).

- Vedi la frase `Partition table holds up to 128 entries`? Esatto, puoi avere fino a 128 partizioni su un disco GPT. Per questo motivo, non sono necessarie le partizioni *primarie* e *estese*.
- Lo spazio libero è elencato nell'ultima riga, quindi non è necessario un equivalente del comando `F` da `fdisk`.

Creare una Partizione

Il comando per creare una partizione è `n`, proprio come in `fdisk`. La differenza principale è che oltre al numero di partizione e al primo e all'ultimo settore (o dimensione), è anche possibile specificare il tipo di partizione durante la creazione. Le partizioni GPT supportano molti più tipi rispetto a MBR. Puoi controllare un elenco di tutti i tipi supportati usando il comando `l`.

Cancellare una Partizione

Per eliminare una partizione, digitare `d` e il numero della partizione. A differenza di `fdisk`, la prima partizione non verrà selezionata automaticamente se è l'unica sul disco.

Sui dischi GPT le partizioni possono essere facilmente riordinate, o riorganizzate, per evitare lacune nella sequenza di numerazione. Per fare ciò usa semplicemente il comando `s`. Per esempio, immagina un disco con la seguente tabella delle partizioni

Number	Start (sector)	End (sector)	Size	Code	Name
1	2048	2099199	1024.0 MiB	8300	Linux filesystem
2	2099200	2361343	128.0 MiB	8300	Linux filesystem
3	2361344	2623487	128.0 MiB	8300	Linux filesystem

Se elimini la seconda partizione, la tabella diventerà:

Number	Start (sector)	End (sector)	Size	Code	Name
1	2048	2099199	1024.0 MiB	8300	Linux filesystem
3	2361344	2623487	128.0 MiB	8300	Linux filesystem

Se usi il comando `s`, diventerebbe:

Number	Start (sector)	End (sector)	Size	Code	Name
1	2048	2099199	1024.0 MiB	8300	Linux filesystem
2	2361344	2623487	128.0 MiB	8300	Linux filesystem

Notare che la terza partizione è diventata la seconda.

Spazio? Quale Spazio?

A differenza dei dischi MBR, quando si crea una partizione su dischi GPT la dimensione non è limitata dalla quantità massima di spazio *contiguo* non allocato. È possibile utilizzare fino all'ultimo bit di un settore libero, indipendentemente da dove si trova sul disco.

Opzioni di Ripristino

I dischi GPT archiviano copie di backup dell'intestazione GPT e della tabella delle partizioni, semplificando il ripristino dei dischi nel caso in cui questi dati siano stati danneggiati. `gdisk` fornisce funzionalità per aiutare nelle attività di ripristino, accessibili con il comando `r`.

È possibile ricostruire un'intestazione GPT principale o una tabella delle partizioni danneggiata rispettivamente con `b` e `c`, oppure utilizzare l'intestazione e la tabella principali per ricostruire un backup con `d` e `e`. Puoi anche convertire un MBR in un GPT con `f` e fare l'opposto con `g`, tra le altre operazioni. Digita `?` nel menu di ripristino per ottenere un elenco di tutti i comandi di ripristino disponibili e le descrizioni di ciò che fanno.

Creare un File System

Il partizionamento del disco è solo il primo passo per poterlo utilizzare. Dopodiché, dovrai formattare la partizione con un filesystem prima di utilizzarlo per memorizzare i dati.

Un filesystem controlla il modo in cui i dati vengono archiviati e acceduti sul disco. Linux supporta molti filesystem: alcuni nativi, come la famiglia *ext* (Extended Filesystem); altri provengono da sistemi operativi differenti come FAT da MS-DOS, NTFS da Windows NT, HFS e HFS+ da Mac OS, ecc.

Lo strumento standard usato per creare un filesystem su Linux è `mkfs`, disponibile in molte “varianti” a seconda del filesystem con cui deve lavorare.

Creare un Filesystem ext2/ext3/ext4

L' *Extended Filesystem* (*ext*) è stato il primo filesystem per Linux e nel corso degli anni è stato sostituito da nuove versioni chiamate *ext2*, *ext3* ed *ext4*, che attualmente è il filesystem

predefinito per molte distribuzioni Linux.

Le utilità `mkfs.ext2`, `mkfs.ext3` e `mkfs.ext4` sono usate per creare filesystem `ext2`, `ext3` ed `ext4`. In effetti, tutte queste “utilità” esistono solo come collegamenti simbolici a un'altra utilità chiamata `mke2fs`. `mke2fs` altera i suoi valori predefiniti in base al nome con cui è chiamato. In quanto tali, hanno tutti lo stesso comportamento e parametri della riga di comando.

La forma più semplice di utilizzo è:

```
# mkfs.ext2 TARGET
```

Dove `TARGET` è il nome della partizione in cui deve essere creato il filesystem. Per esempio, per creare un filesystem `ext3` su `/dev/sdb1` il comando sarebbe:

```
# mkfs.ext3 /dev/sdb1
```

Invece di usare il comando corrispondente al filesystem che desideri creare, puoi passare il parametro `-t` a `mke2fs` seguito dal nome del filesystem. Per esempio, i seguenti comandi sono equivalenti e creeranno un filesystem `ext4` su `/dev/sdb1`

```
# mkfs.ext4 /dev/sdb1
# mke2fs -t ext4 /dev/sdb1
```

Parametri da Linea di Comando

`mke2fs` supporta un'ampia gamma di parametri e opzioni dalla riga di comando. Ecco alcuni dei più significativi. Tutti si applicano anche a `mkfs.ext2`, `mkfs.ext3` e `mkfs.ext4`:

-b SIZE

Imposta la dimensione dei blocchi di dati nel dispositivo su `SIZE`, che può essere 1024, 2048 o 4096 byte per blocco.

-c

Controlla il dispositivo di destinazione per i blocchi danneggiati prima di creare il filesystem. È possibile eseguire un controllo completo, ma molto più lento, passando questo parametro due volte, come in `mkfs.ext4 -c -c TARGET`.

-d DIRECTORY

Copia il contenuto della directory specificata nella radice del nuovo filesystem. Utile se hai

bisogno di “pre-popolare” il disco con un insieme predefinito di file.

-F

Pericolo, Will Robinson! Questa opzione *forza (force)* `mke2fs` a creare un filesystem, anche se le altre opzioni passate a esso o l’obiettivo sono pericolose o non hanno alcun senso. Se specificato due volte (come in `-F -F`) può anche essere usato per creare un filesystem su un dispositivo che è montato o in uso: una cosa molto, *molto* brutta da farsi!

-L VOLUME_LABEL

Imposta l’etichetta del volume su quella specificata in `VOLUME_LABEL`. Questa etichetta deve contenere al massimo 16 caratteri.

-n

Questa è un’opzione veramente utile che simula la creazione del filesystem e mostra cosa verrebbe fatto se eseguito senza l’opzione `n`. Considerala come una modalità *prova*. È utile controllare le cose prima di eseguire effettivamente la *commit* delle modifiche sul disco.

-q

Modalità silenziosa. `mke2fs` verrà eseguito normalmente, ma non produrrà alcun output al terminale. Utile quando si esegue `mke2fs` da uno script.

-U ID

Ciò imposterà l’UUID (*Universally Unique Identifier*) di una partizione sul valore specificato come ID. Gli UUID sono numeri a 128 bit in notazione esadecimale che servono a identificare in modo univoco una partizione per il sistema operativo. Questo numero è specificato come una stringa di 32 cifre nel formato 8-4-4-4-12, che significa 8 cifre, trattino, 4 cifre, trattino, 4 cifre, trattino, 4 cifre, trattino, 12 cifre, come `D249E380 -7719-45A1-813C-35186883987E`. Invece di un ID puoi anche specificare parametri come `clear` per cancellare l’UUID del filesystem, `random`, per usare un UUID generato casualmente, o `time` per creare un UUID basato sul tempo.

-v

Modalità dettagliata, stampa molte più informazioni rispetto al normale durante il funzionamento. Utile per scopi di debug.

Creare un Filesystem XFS

XFS è un filesystem ad alte prestazioni originariamente sviluppato da Silicon Graphics nel 1993 per il suo sistema operativo IRIX. A causa delle sue caratteristiche di prestazioni e affidabilità, è comunemente usato per server e altri ambienti che richiedono una larghezza di banda del

filesystem elevata (o garantita).

Gli strumenti per la gestione dei filesystem XFS fanno parte del pacchetto `xfsprogs`. Potrebbe essere necessario installare questo pacchetto manualmente, poiché non è sempre incluso di default. In Red Hat Enterprise Linux 7, XFS è usato come filesystem predefinito.

I filesystem XFS sono divisi in almeno 2 parti, una *sezione di log* dove viene mantenuto un registro di tutte le operazioni del filesystem (comunemente chiamato *Journal*) e la *data section*. La sezione di log può trovarsi all'interno della sezione dati (il comportamento predefinito), o anche su un disco separato del tutto, per migliori prestazioni e affidabilità.

Il comando più semplice per creare un filesystem XFS è `mkfs.xfs TARGET`, dove `TARGET` è la partizione in cui vuoi che venga creato il filesystem. Per esempio: `mkfs.xfs /dev/sda1`.

Come in `mke2fs`, `mkfs.xfs` supporta un certo numero di opzioni dalla riga di comando. Ecco alcuni dei più comuni.

-b size=VALUE

Imposta la dimensione del blocco sul filesystem, in byte, a quella specificata in `VALUE`. Il valore predefinito è 4096 byte (4 KiB), il minimo è 512 e il massimo è 65536 (64 KiB).

-m crc=VALUE

I parametri che iniziano con `-m` sono opzioni di metadati. Questo abilita (se `VALUE` è 1) o disabilita (se `VALUE` è 0) l'uso dei controlli CRC32c per verificare l'integrità di tutti i metadati sul disco. Ciò consente un migliore rilevamento degli errori e ripristino da arresti anomali relativi a problemi hardware, quindi è abilitato per impostazione predefinita. L'impatto sulle prestazioni di questo controllo dovrebbe essere minimo, quindi normalmente non c'è motivo per disabilitarlo.

-m uuid=VALUE

Imposta l'UUID della partizione su quello specificato come `VALUE`. Ricorda che gli UUID sono numeri di 32 caratteri (128 bit) in base esadecimale, specificati in gruppi di 8, 4, 4, 4 e 12 cifre separati da trattini, come `1E83E3A3-3AE9-4AAC-BF7E-29DFFECD36C0`.

-f

Forza la creazione di un filesystem sul dispositivo di destinazione anche se viene rilevato un altro filesystem su di esso.

-l logdev=DEVICE

Questo metterà la sezione di registro del filesystem sul dispositivo specificato, invece che all'interno della sezione dati.

-l size=VALUE

Questo imposterà la dimensione della sezione del registro a quella specificata in VALUE. La dimensione può essere specificata in byte e possono essere usati suffissi come m o g. `-l size=10m`, per esempio, limiterà la sezione del registro a 10 Megabyte.

-q

Modalità silenziosa. In questa modalità, `mkfs.xfs` non visualizzerà a schermo i parametri del file system che si sta creando.

-L LABEL

Imposta l'etichetta del filesystem, che può essere lunga al massimo 12 caratteri.

-N

Simile al parametro `-n` di `mke2fs`, farà in modo che `mkfs.xfs` visualizzi tutti i parametri per la creazione del file system, senza crearlo effettivamente.

Creare un Filesystem FAT o VFAT

Il filesystem FAT ha avuto origine da MS-DOS, e negli anni ha ricevuto molte revisioni culminate nel formato FAT32 rilasciato nel 1996 con Windows 95 OSR2.

VFAT è un'estensione del formato FAT16 con supporto per nomi di file lunghi (fino a 255 caratteri). Entrambi i filesystem sono gestiti dalla stessa utility, `mkfs.fat`. `mkfs.vfat` è un alias.

Il filesystem FAT ha importanti svantaggi che ne limitano l'uso su dischi di grandi dimensioni. FAT16, per esempio, supporta volumi di massimo 4 GB e una dimensione massima del file di 2 GB. FAT32 aumenta la dimensione del volume fino a 2 PB e la dimensione massima del file a 4 GB. Per questo motivo, i filesystem FAT sono oggi più comunemente usati su piccole unità flash o schede di memoria (fino a 2 GB di dimensione), o dispositivi legacy e sistemi operativi che non supportano filesystem più avanzati.

Il comando più semplice per la creazione di un filesystem FAT è `mkfs.fat TARGET`, dove TARGET è la partizione in cui vuoi che venga creato il filesystem. Per esempio: `mkfs.fat /dev/sdc1`.

Come altre utilità, `mkfs.fat` supporta una serie di opzioni della riga di comando. Di seguito sono riportati i più importanti. Un elenco completo e una descrizione di ogni opzione può essere trovato nel manuale dell'utilità, con il comando `man mkfs.fat`.

-c

Controlla il dispositivo di destinazione per i blocchi danneggiati prima di creare il filesystem.

-C FILENAME BLOCK_COUNT

Creerà il file specificato in `FILENAME` e quindi creerà un filesystem FAT al suo interno, creando effettivamente una “immagine disco” vuota, che può essere successivamente scritta su un dispositivo usando un'utilità come `dd` o montata come dispositivo `loop`. Quando si utilizza questa opzione, il numero di blocchi nel filesystem (`BLOCK_COUNT`) deve essere specificato dopo il nome del dispositivo.

-F SIZE

Seleziona la dimensione della FAT (*File Allocation Table*), tra 12, 16 o 32, ovvero tra FAT12, FAT16 o FAT32. Se non specificato, `mkfs.fat` selezionerà l'opzione appropriata in base alla dimensione del filesystem.

-n NAME

Imposta l'etichetta del volume, o il nome, per il filesystem. Può essere lunga fino a 11 caratteri e l'impostazione predefinita è nessun nome.

-v

Modalità dettagliata (Verbose mode). Visualizza molte più informazioni del solito, utili per il debug

NOTE

`mkfs.fat` non può creare un filesystem “avviabile”. Secondo la pagina del manuale “non è così facile come potresti pensare” e non sarà implementato.

Creare un Filesystem exFAT

exFAT è un filesystem creato da Microsoft nel 2006 che affronta uno dei limiti più importanti di FAT32: file e dimensioni del disco. In exFAT, la dimensione massima del file è di 16 exabyte (da 4 GB su FAT32) e la dimensione massima del disco è di 128 petabyte.

Poiché è ben supportato da tutti e tre i principali sistemi operativi (Windows, Linux e Mac OS), è una buona scelta dove è necessaria l'interoperabilità, come su unità flash di grande capacità, schede di memoria e dischi esterni. In effetti, è il filesystem predefinito, come definito dalla *SD Association*, per le schede di memoria SDXC più grandi di 32 GB.

L'utilità predefinita per la creazione di filesystem exFAT è `mkfs.exfat`, che è un collegamento a `mkexfatfs`. Il comando più semplice è `mkfs.exfat TARGET`, dove `TARGET` è la partizione in cui vuoi creare il filesystem. Per esempio: `mkfs.exfat /dev/sdb2`.

Contrariamente alle altre utilità discusse in questa lezione, `mkfs.exfat` ha pochissime opzioni della riga di comando. Sono:

-i VOL_ID

Imposta l'ID del volume sul valore specificato in VOL_ID. Questo è un numero esadecimale a 32 bit. Se non definito, viene impostato un ID basato sull'ora corrente.

-n NAME

Imposta l'etichetta o il nome del volume. Può contenere fino a 15 caratteri e l'impostazione predefinita è nessun nome.

-p SECTOR

Specifica il primo settore della prima partizione sul disco. Questo è un valore facoltativo e il valore predefinito è zero.

-s SECTORS

Definisce il numero di settori fisici per cluster di allocazione. Deve essere una potenza di due, come 1, 2, 4, 8 e così via.

Conoscere il Filesystem Btrfs

Btrfs (ufficialmente *B-Tree Filesystem*, pronunciato come “Butter FS”, “Better FS” o anche “Butterfuss”, a tua scelta) è un filesystem in sviluppo dal 2007 realizzato specificamente per Linux da Oracle Corporation e altre società, tra cui Fujitsu, Red Hat, Intel e SUSE.

Ci sono molte caratteristiche che rendono Btrfs attraente sui sistemi moderni in cui sono comuni enormi quantità di spazio di archiviazione. Tra questi ci sono il supporto di più dispositivi (inclusi striping, mirroring e striping + mirroring, come in una configurazione RAID), compressione trasparente, ottimizzazioni SSD, backup incrementali, snapshot, deframmentazione online, controlli offline, supporto per volumi secondari (con quote), deduplicazione e molto altro.

Essendo un filesystem *copy-on-write* è molto resistente ai crash. Inoltre, Btrfs è semplice da usare e ben supportato da molte distribuzioni Linux. E alcuni di loro, come SUSE, lo usano come filesystem predefinito.

NOTE

In un filesystem tradizionale, quando si desidera sovrascrivere una parte di un file, i nuovi dati vengono inseriti direttamente sui vecchi dati che sta sostituendo. Su un filesystem *copy-on-write* i nuovi dati vengono scritti nello spazio libero su disco, quindi i metadati originali del file vengono aggiornati per fare riferimento ai nuovi dati e solo allora i vecchi dati vengono liberati, poiché non sono più necessari. Ciò riduce le possibilità di perdita di dati in caso di arresto anomalo, poiché i vecchi dati vengono eliminati solo dopo che il filesystem è assolutamente sicuro che non sono più necessari e i nuovi dati sono al posto giusto.

Creare un Filesystem Btrfs

L'utilità `mkfs.btrfs` viene utilizzata per creare un filesystem Btrfs. L'uso del comando senza alcuna opzione crea un file system Btrfs su un determinato dispositivo, in questo modo:

```
# mkfs.btrfs /dev/sdb1
```

TIP

Se non hai l'utilità `mkfs.btrfs` sul tuo sistema, cerca `btrfs-progs` nel gestore dei pacchetti della tua distribuzione.

Puoi usare `-L` per impostare un'etichetta (o un nome) per il tuo filesystem. Le etichette Btrfs possono contenere fino a 256 caratteri, ad eccezione di nuove righe:

```
# mkfs.btrfs /dev/sdb1 -L "New Disk"
```

TIP

Racchiudi l'etichetta tra virgolette (come sopra) se contiene spazi.

Tieni in considerazione questa cosa peculiare di Btrfs: puoi passare dispositivi *multipli* al comando `mkfs.btrfs`. Il passaggio di più di un dispositivo estenderà il filesystem su tutti i dispositivi che è simile a una configurazione RAID o LVM. Per specificare come verranno distribuiti i metadati nell'array di dischi, utilizzare il parametro `-m`. I parametri validi sono `raid0`, `raid1`, `raid5`, `raid6`, `raid10`, `single` e `dup`.

Per esempio, per creare un filesystem che si estende su `/dev/sdb1` e `/dev/sdc1`, concatenando le due partizioni in una più grande, usa:

```
# mkfs.btrfs -d single -m single /dev/sdb /dev/sdc
```

WARNING

I filesystem che si estendono su più partizioni, come sopra, potrebbero sembrare vantaggiosi all'inizio, ma non sono una buona idea dal punto di vista della sicurezza dei dati, poiché un guasto su un singolo disco dell'array significa una perdita certa di dati. Il rischio aumenta con l'aumentare del numero di dischi utilizzati, poiché si hanno anche più possibili punti in cui l'errore possa verificarsi.

Gestire i Subvolume

I *subvolume* sono come filesystem all'interno di altri filesystem. Pensa a loro come a una directory che può essere montata come (e trattata come) un filesystem separato. I subvolume semplificano l'organizzazione e l'amministrazione del sistema, poiché ognuno di essi può avere quote separate

o regole di snapshot.

NOTE

I sottovolumi non sono partizioni. Una partizione alloca uno spazio fisso su un'unità. Questo può portare a problemi vari, come per esempio una partizione che esaurisce lo spazio quando un'altra invece ne ha molto a disposizione. Non così con i sottovolumi, poiché “condividono” lo spazio libero dal loro filesystem di root e crescono secondo necessità.

Supponiamo di avere un filesystem Btrfs montato su `/mnt/disk` e di voler creare un sottovolume al suo interno per memorizzare i vostri backup. Chiamiamolo BKP:

```
# btrfs subvolume create /mnt/disk/BKP
```

Successivamente elenchiamo i contenuti del filesystem `/mnt/disk`. Vedrai che abbiamo una nuova directory, che prende il nome dal nostro sottovolume

```
$ ls -lh /mnt/disk/
total 0
drwxr-xr-x 1 root  root    0 jul 13 17:35 BKP
drwxrwxr-x 1 carol carol 988 jul 13 17:30 Images
```

NOTE

Sì, è anche possibile accedere ai sottovolumi come ad una qualsiasi altra directory.

Possiamo verificare che il sottovolume sia attivo, con il comando:

```
# btrfs subvolume show /mnt/disk/BKP/
Name:          BKP
UUID:          e90a1afe-69fa-da4f-9764-3384f66fa32e
Parent UUID:    -
Received UUID:  -
Creation time:  2019-07-13 17:35:40 -0300
Subvolume ID:   260
Generation:     23
Gen at creation: 22
Parent ID:      5
Top level ID:   5
Flags:          -
Snapshot(s):
```

Puoi montare il sottovolume su `/mnt/BKP` passando il parametro `-t btrfs -o subvol=NAME` al comando `mount`:

```
# mount -t btrfs -o subvol=BKP /dev/sdb1 /mnt/bkp
```

NOTE Il parametro `-t` specifica il tipo di filesystem da montare.

Lavorare con le Snapshot

Le *snapshot* sono come volumi secondari, ma precompilate con i contenuti del volume su cui è stata scattata un'istantanea.

In fase di creazione, una snapshot e il volume originale hanno esattamente lo stesso contenuto. Ma da quel momento in poi, divergeranno. Le modifiche apportate al volume originale (come i file aggiunti, rinominati o eliminati) non si rifletteranno sull'istantanea e viceversa.

Tieni presente che una snapshot *non* duplica i file e inizialmente occupa quasi nessuno spazio su disco. Duplica semplicemente l'albero del filesystem, mentre punta ai dati originali.

Il comando per creare una snapshot è lo stesso usato per creare un sottovolume, basta aggiungere il parametro `snapshot` dopo `btrfs subvolume`. Il comando seguente creerà un'istantanea del filesystem Btrfs montato in `/mnt/disk` in `/mnt/disk/snap`:

```
# btrfs subvolume snapshot /mnt/disk /mnt/disk/snap
```

Ora, immagina di avere i seguenti contenuti in `/mnt/disk`:

```
$ ls -lh
total 2,8M
-rw-rw-r-- 1 carol carol 109K jul 10 16:22 Galaxy_Note_10.png
-rw-rw-r-- 1 carol carol 484K jul  5 15:01 geminoid2.jpg
-rw-rw-r-- 1 carol carol 429K jul  5 14:52 geminoid.jpg
-rw-rw-r-- 1 carol carol 467K jul  2 11:48 LG-G8S-ThinQ-Mirror-White.jpg
-rw-rw-r-- 1 carol carol 654K jul  2 11:39 LG-G8S-ThinQ-Range.jpg
-rw-rw-r-- 1 carol carol  94K jul  2 15:43 Memoji_Comparativo.jpg
-rw-rw-r-- 1 carol carol 112K jul 10 16:20 Note10Plus.jpg
drwx----- 1 carol carol  366 jul 13 17:56 snap
-rw-rw-r-- 1 carol carol 118K jul 11 16:36 Twitter_Down_20190711.jpg
-rw-rw-r-- 1 carol carol 324K jul  2 15:22 Xiaomi_Memoji.png
```

Notare la directory `snap`, contenente l'istantanea. Ora rimuoviamo alcuni file e controlliamo il contenuto della directory:

```
$ rm LG-G8S-ThinQ-*
$ ls -lh
total 1,7M
-rw-rw-r-- 1 carol carol 109K jul 10 16:22 Galaxy_Note_10.png
-rw-rw-r-- 1 carol carol 484K jul  5 15:01 geminoid2.jpg
-rw-rw-r-- 1 carol carol 429K jul  5 14:52 geminoid.jpg
-rw-rw-r-- 1 carol carol  94K jul  2 15:43 Memoji_Comparativo.jpg
-rw-rw-r-- 1 carol carol 112K jul 10 16:20 Note10Plus.jpg
drwx----- 1 carol carol  366 jul 13 17:56 snap
-rw-rw-r-- 1 carol carol 118K jul 11 16:36 Twitter_Down_20190711.jpg
-rw-rw-r-- 1 carol carol 324K jul  2 15:22 Xiaomi_Memoji.png
```

Tuttavia, se controlli all'interno della directory `snap`, i file eliminati sono ancora lì e possono essere ripristinati se necessario.

```
$ ls -lh snap/
total 2,8M
-rw-rw-r-- 1 carol carol 109K jul 10 16:22 Galaxy_Note_10.png
-rw-rw-r-- 1 carol carol 484K jul  5 15:01 geminoid2.jpg
-rw-rw-r-- 1 carol carol 429K jul  5 14:52 geminoid.jpg
-rw-rw-r-- 1 carol carol 467K jul  2 11:48 LG-G8S-ThinQ-Mirror-White.jpg
-rw-rw-r-- 1 carol carol 654K jul  2 11:39 LG-G8S-ThinQ-Range.jpg
-rw-rw-r-- 1 carol carol  94K jul  2 15:43 Memoji_Comparativo.jpg
-rw-rw-r-- 1 carol carol 112K jul 10 16:20 Note10Plus.jpg
-rw-rw-r-- 1 carol carol 118K jul 11 16:36 Twitter_Down_20190711.jpg
-rw-rw-r-- 1 carol carol 324K jul  2 15:22 Xiaomi_Memoji.png
```

È anche possibile creare istantanee di sola lettura. Funzionano esattamente come le istantanee scrivibili, con la differenza che i contenuti dell'istantanea non possono essere modificati, sono “congelati” nel tempo. Aggiungi semplicemente il parametro `-r` durante la creazione dell'istantanea:

```
# btrfs subvolume snapshot -r /mnt/disk /mnt/disk/snap
```

Accenni sulla Compressione

Btrfs supporta la compressione trasparente dei file, con tre diversi algoritmi a disposizione dell'utente. Questo viene fatto automaticamente per file, fintanto che il filesystem è montato con l'opzione `-o compress`. Gli algoritmi sono abbastanza intelligenti da rilevare file incompressibili e non tenteranno di comprimerli, risparmiando risorse di sistema. Quindi su una singola directory

potresti avere file compressi e non compressi insieme. L'algoritmo di compressione predefinito è ZLIB, ma sono disponibili LZO (più veloce, rapporto di compressione peggiore) o ZSTD (più veloce di ZLIB, compressione comparabile), con più livelli di compressione (vedere la parte corrispondente sulle opzioni di montaggio).

Gestire le Partizioni con GNU Parted

GNU Parted è un potente editor di partizioni (da cui il nome) che può essere utilizzato per creare, eliminare, spostare, ridimensionare, salvare e copiare partizioni. Può funzionare con dischi GPT e MBR e coprire quasi tutte le esigenze di gestione del disco.

Ci sono molti front-end grafici che rendono molto più facile lavorare con `parted`, come *GParted* per ambienti desktop basati su GNOME e *KDE Partition Manager* per desktop KDE. Tuttavia, dovresti imparare come usare `parted` dalla riga di comando, poiché in un sistema server non puoi contare sulla disponibilità di un ambiente desktop grafico.

WARNING

A differenza di `fdisk` e `gdisk`, `parted` apporta modifiche al disco *immediatamente* dopo che il comando è stato dato, senza attendere che un altro comando scriva le modifiche sul disco. Quando ti eserciti, è consigliabile farlo su un disco vuoto o di riserva o su un'unità flash, così che non ci sia rischio di perdita di dati in caso di errore.

Il modo più semplice per iniziare a usare `parted` è digitare `parted DEVICE`, dove `DEVICE` è il dispositivo che vuoi gestire (`parted /dev/sdb`). Il programma avvia un'interfaccia a riga di comando interattiva come `fdisk` e `gdisk` con un prompt (`parted`) per inserire i comandi.

```
# parted /dev/sdb
GNU Parted 3.2
Using /dev/sdb
Welcome to GNU Parted! Type 'help' to view a list of commands.

(parted)
```

WARNING

Stai attento! Se non specifichi un dispositivo, `parted` selezionerà automaticamente il disco principale (di solito `/dev/sda`) con cui lavorare.

Selezionare i Dischi

Per passare a un disco diverso da quello specificato sulla riga di comando puoi utilizzare il comando `select`, seguito dal nome del dispositivo:

```
(parted) select /dev/sdb
Using /dev/sdb
```

Ottenere Informazioni

Il comando `print` può essere usato per ottenere maggiori informazioni su una specifica partizione o anche su tutti i dispositivi a blocchi (dischi) collegati al tuo sistema.

Per ottenere informazioni sulla partizione attualmente selezionata, digita semplicemente `print`:

```
(parted) print
Model: ATA CT120BX500SSD1 (scsi)
Disk /dev/sda: 120GB
Sector size (logical/physical): 512B/512B
Partition Table: msdos
Disk Flags:

Number  Start   End     Size    Type    File system  Flags
  1      2097kB 116GB   116GB   primary ext4
  2      116GB  120GB   4295MB  primary linux-swap(v1)
```

Puoi ottenere un elenco di tutti i dispositivi a blocchi collegati al tuo sistema usando `print devices`:

```
(parted) print devices
/dev/sdb (1999MB)
/dev/sda (120GB)
/dev/sdc (320GB)
/dev/mapper/cryptswap (4294MB)
```

Per ottenere informazioni su tutti i dispositivi collegati contemporaneamente puoi usare `print all`. Se desideri sapere quanto spazio libero c'è in ognuno di essi, puoi usare `print free`:

```
(parted) print free
Model: ATA CT120BX500SSD1 (scsi)
Disk /dev/sda: 120GB
Sector size (logical/physical): 512B/512B
Partition Table: msdos
Disk Flags:
```

Number	Start	End	Size	Type	File system	Flags
	32.3kB	2097kB	2065kB		Free Space	
1	2097kB	116GB	116GB	primary	ext4	
	116GB	116GB	512B		Free Space	
2	116GB	120GB	4295MB	primary	linux-swap(v1)	
	120GB	120GB	2098kB		Free Space	

Creare una Tabella delle Partizioni su un Disco Vuoto

Per creare una tabella delle partizioni su un disco vuoto, utilizzare il comando `mklabel`, seguito dal tipo di tabella delle partizioni che si desidera utilizzare.

Esistono molti tipi di tabelle delle partizioni supportate, ma i tipi principali di cui dovresti essere a conoscenza sono `msdos` che viene utilizzato qui per fare riferimento a una tabella delle partizioni MBR e `gpt` per fare riferimento a una tabella delle partizioni GPT. Per creare una tabella delle partizioni MBR, digita:

```
(parted) mklabel msdos
```

E per creare una tabella delle partizioni GPT, il comando è:

```
(parted) mklabel gpt
```

Creare una Partizione

Per creare una partizione viene utilizzato il comando `mkpart`, utilizzando la sintassi `mkpart PARTTYPE FSTYPE START END`, dove:

PARTTYPE

È il tipo di partizione, che può essere `primary`, `logical` o `extended` nel caso in cui venga utilizzata una tabella delle partizioni MBR.

FSTYPE

Specifica quale file system verrà utilizzato su questa partizione. Nota che `parted` *non* creerà il filesystem. Imposta solo un flag sulla partizione che dice al sistema operativo che tipo di dati aspettarsi da esso.

START

Specifica il punto esatto sul dispositivo in cui inizia la partizione. È possibile utilizzare unità

diverse per specificare questo punto. `2s` può essere utilizzato per fare riferimento al secondo settore del disco, mentre `1m` si riferisce all'inizio del primo megabyte del disco. Altre unità comuni sono `B` (byte) e `%` (percentuale del disco).

END

Specifica la fine della partizione. Nota che questa *non* è la dimensione della partizione, ma *il punto del disco dove finisce*. Per esempio, se specifichi `100m`, la partizione terminerà 100 MB dopo l'inizio del disco. È possibile utilizzare le stesse unità del parametro `START`.

Così, il comando:

```
(parted) mkpart primary ext4 1m 100m
```

Crea una partizione primaria di tipo `ext4`, iniziando dal primo megabyte del disco e terminando dopo il centesimo megabyte.

Eliminare una Partizione

Per rimuovere una partizione, usa il comando `rm` seguito dal numero della partizione, che puoi visualizzare usando il comando `print`. Quindi, `rm 2` rimuove la seconda partizione sul disco attualmente selezionato.

Recuperare Partizioni

`parted` può recuperare una partizione cancellata. Considera di avere la seguente struttura delle partizioni:

Number	Start	End	Size	File system	Name	Flags
1	1049kB	99.6MB	98.6MB	ext4	primary	
2	99.6MB	200MB	100MB	ext4	primary	
3	200MB	300MB	99.6MB	ext4	primary	

Per sbaglio, hai rimosso la partizione 2 usando `rm 2`. Per ripristinarlo, puoi usare il comando `rescue`, con la sintassi `rescue START END`, dove `START` è la posizione approssimativa in cui iniziava la partizione e `END` la posizione approssimativa in cui finiva.

`parted` eseguirà la scansione del disco alla ricerca di partizioni e si offrirà di ripristinare quelle trovate. Nell'esempio sopra, la partizione 2 iniziava con 99,6 MB e si concludeva con 200 MB. Quindi puoi usare il seguente comando per ripristinare la partizione:

```
(parted) rescue 90m 210m
Information: A ext4 primary partition was found at 99.6MB -> 200MB.
Do you want to add it to the partition table?

Yes/No/Cancel? y
```

Ciò ripristinerà la partizione e il suo contenuto. Notare che `rescue` può recuperare solo partizioni su cui è installato un filesystem. Le partizioni vuote non vengono rilevate.

Ridimensionamento Partizioni ext2/3/4

`parted` può essere utilizzato per ridimensionare le partizioni al fine di renderle più grandi o più piccole. Tuttavia, ci sono alcuni avvertimenti:

- Durante il ridimensionamento, la partizione deve essere inutilizzata e smontata.
- Hai bisogno di spazio libero sufficiente *dopo* la partizione per ingrandirla fino alla dimensione desiderata.

Il comando è `resizepart`, seguito dal numero di partizione e dove dovrebbe finire. Per esempio, se hai la seguente tabella delle partizioni:

Number	Start	End	Size	File system	Name	Flags
1	1049kB	99.6MB	98.6MB	ext4	primary	
2	99.6MB	200MB	100MB	ext4		
3	200MB	300MB	99.6MB	ext4	primary	

Cercare di aumentare la partizione 1 usando `resizepart` attiverrebbe un messaggio di errore, perché con la nuova dimensione la partizione 1 si sovrapporrebbe alla partizione 2. Tuttavia la partizione 3 può essere ridimensionata in quanto c'è spazio libero dopo di essa, che può essere verificato con il comando `print free`:

```
(parted) print free
Model: Kingston DataTraveler 2.0 (scsi)
Disk /dev/sdb: 1999MB
Sector size (logical/physical): 512B/512B
Partition Table: gpt
Disk Flags:

Number  Start   End     Size    File system  Name      Flags
    17.4kB 1049kB 1031kB  Free Space
```

1	1049kB	99.6MB	98.6MB	ext4	primary
2	99.6MB	200MB	100MB	ext4	
3	200MB	300MB	99.6MB	ext4	primary
	300MB	1999MB	1699MB	Free Space	

Quindi puoi usare il seguente comando per ridimensionare la partizione 3 a 350 MB:

```
(parted) resizepart 3 350m
```

```
(parted) print
```

```
Model: Kingston DataTraveler 2.0 (scsi)
```

```
Disk /dev/sdb: 1999MB
```

```
Sector size (logical/physical): 512B/512B
```

```
Partition Table: gpt
```

```
Disk Flags:
```

Number	Start	End	Size	File system	Name	Flags
1	1049kB	99.6MB	98.6MB	ext4	primary	
2	99.6MB	200MB	100MB	ext4		
3	200MB	350MB	150MB	ext4	primary	

Ricorda che il nuovo punto finale viene specificato contando dall'inizio del disco. Quindi, poiché la partizione 3 è terminata a 300 MB, ora deve terminare a 350 MB.

Ma il ridimensionamento della partizione è solo una parte dell'attività. È inoltre necessario ridimensionare il filesystem che risiede al suo interno. Per i filesystem ext2/3/4 questo viene fatto con il comando `resize2fs`. Nel caso dell'esempio sopra, la partizione 3 mostra ancora la dimensione “vecchia” quando montata:

```
$ df -h /dev/sdb3
```

Filesystem	Size	Used	Avail	Use%	Mounted on
/dev/sdb3	88M	1.6M	80M	2%	/media/carol/part3

Per regolare la dimensione, è possibile utilizzare il comando `resize2fs DEVICE SIZE`, dove `DEVICE` corrisponde alla partizione che si desidera ridimensionare e `SIZE` è la nuova dimensione. Se ometti il parametro `size`, utilizzerà tutto lo spazio disponibile della partizione. Prima di ridimensionare, si consiglia di smontare la partizione.

Nell'esempio seguente:

```
$ sudo resize2fs /dev/sdb3
```

```
resize2fs 1.44.6 (5-Mar-2019)
Resizing the filesystem on /dev/sdb3 to 146212 (1k) blocks.
The filesystem on /dev/sdb3 is now 146212 (1k) blocks long.
```

```
$ df -h /dev/sdb3
```

```
Filesystem      Size  Used Avail Use% Mounted on
/dev/sdb3       135M  1.6M  123M   2% /media/carol/part3
```

Per *restringere* una partizione, il processo deve essere eseguito nell'ordine inverso. *Prima* ridimensionate il filesystem alla nuova dimensione più piccola, poi ridimensionate la partizione stessa usando `parted`.

WARNING

Prestare attenzione quando si restringono le partizioni. Se sbagli l'ordine delle cose, perderai i dati!

Nel nostro esempio:

```
# resize2fs /dev/sdb3 88m
resize2fs 1.44.6 (5-Mar-2019)
Resizing the filesystem on /dev/sdb3 to 90112 (1k) blocks.
The filesystem on /dev/sdb3 is now 90112 (1k) blocks long.

# parted /dev/sdb3
(parted) resizepart 3 300m
Warning: Shrinking a partition can cause data loss, are you sure
you want to continue?

Yes/No? y

(parted) print
Model: Kingston DataTraveler 2.0 (scsi)
Disk /dev/sdb: 1999MB
Sector size (logical/physical): 512B/512B
Partition Table: gpt
Disk Flags:

Number  Start   End     Size    File system  Name      Flags
  1      1049kB  99.6MB  98.6MB  ext4         primary
  2      99.6MB  200MB   100MB   ext4
  3      200MB   300MB   99.7MB  ext4         primary
```

TIP

Invece di specificare una nuova dimensione, puoi usare il parametro `-M` di

`resize2fs` per regolare la dimensione del filesystem in modo che sia abbastanza grande per i file su di esso.

Creare Partizioni di Swap

Su Linux, il sistema può scambiare le pagine di memoria dalla RAM al disco secondo necessità, memorizzandole in uno spazio separato solitamente implementato come partizione separata su un disco, chiamata *partizione di swap* o semplicemente *swap*. Questa partizione deve essere di un tipo specifico e configurata con un'utilità appropriata (`mkswap`) prima di poter essere utilizzata.

Per creare la partizione di swap usando `fdisk` o `gdisk`, procedi come se stessi creando una partizione normale, come spiegato prima. L'unica differenza è che dovrai cambiare il tipo di partizione in *Linux swap*.

- Su `fdisk` usa il comando `t`. Seleziona la partizione che desideri utilizzare e cambia il suo tipo in `82`. Scrivi le modifiche su disco ed esci con `w`.
- Su `gdisk` il comando per cambiare il tipo di partizione è sempre `t`, ma il codice è `8200`. Scrivi le modifiche su disco ed esci con `w`.

Se stai usando `parted`, la partizione dovrebbe essere identificata come partizione di swap durante la creazione, usa semplicemente `linux-swap` come tipo di filesystem. Per esempio, il comando per creare una partizione di swap da 500 MB, a partire da 300 MB su un disco è:

```
(parted) mkpart primary linux-swap 301m 800m
```

Una volta che la partizione è stata creata e correttamente identificata, usa semplicemente `mkswap` seguito dal dispositivo che rappresenta la partizione che vuoi usare, come per esempio:

```
# mkswap /dev/sda2
```

Per abilitare lo swap su questa partizione, usa `swapon` seguito dal nome del dispositivo:

```
# swapon /dev/sda2
```

Allo stesso modo, `swapoff`, seguito dal nome del dispositivo, disabiliterà lo swap su quel dispositivo.

Linux supporta anche l'uso di *file* di swap invece delle partizioni. Basta creare un file vuoto della dimensione che si desidera utilizzando `dd` e quindi utilizzare `mkswap` e `swapon` con questo file

come destinazione.

I seguenti comandi creeranno un file da 1 GB chiamato `myswap` nella directory corrente, riempito di “zeri”, quindi lo configureranno e lo abiliteranno come file di swap.

Crea il file di swap:

```
$ dd if=/dev/zero of=myswap bs=1M count=1024
1024+0 records in
1024+0 records out
1073741824 bytes (1.1 GB, 1.0 GiB) copied, 7.49254 s, 143 MB/s
```

`if=` è il file di input, l’origine dei dati che verranno scritti nel file. In questo caso è il dispositivo `/dev/zero`, che fornisce tanti caratteri NULL quanti sono richiesti. `of =` è il file di output, il file che verrà creato. `bs=` è la dimensione dei blocchi di dati, qui specificata in Megabyte, e `count=` è la quantità di blocchi da scrivere nell’output. 1024 blocchi da 1 MB ciascuno equivalgono a 1 GB.

```
# mkswap myswap
Setting up swapspace version 1, size = 1024 MiB (1073737728 bytes)
no label, UUID=49c53bc4-c4b1-4a8b-a613-8f42cb275b2b

# swapon myswap
```

Utilizzando i comandi precedenti, questo file di swap verrà utilizzato solo durante la sessione di sistema corrente. Se la macchina viene riavviata, il file sarà ancora disponibile, ma non verrà caricato automaticamente. Puoi automatizzarlo aggiungendo il nuovo file di swap a `/etc/fstab`, di cui parleremo in una lezione successiva.

TIP

Sia `mkswap` che `swapon` obietteranno se il tuo file di swap ha permessi non sicuri. Il security mode di autorizzazione file consigliato è `0600`. Il proprietario e il gruppo dovrebbero essere `root`.

Esercizi Guidati

1. Quale schema di partizionamento dovrebbe essere utilizzato per partizionare un disco rigido da 3 TB in tre partizioni da 1 GB? Perché?
2. Su `gdisk`, come possiamo sapere quanto spazio è disponibile sul disco?
3. Quale è il comando per creare un filesystem `ext3`, controllando prima la presenza di blocchi danneggiati, con l'etichetta `MyDisk` e un UUID casuale, sul dispositivo `/dev/sdc1`?
4. Utilizzando `parted`, qual è il comando per creare una partizione `ext4` da 300 MB, a partire da 500 MB sul disco?
5. Immagina di avere 2 partizioni, una su `/dev/sda1` e l'altra su `/dev/sda2`, entrambe di 20 GB. Come puoi usarli su un singolo filesystem `Btrfs`, in modo tale che il contenuto di una partizione venga automaticamente duplicato sull'altra, come su una configurazione `RAID1`? Quanto sarà grande il filesystem?

Esercizi Esplorativi

1. Considera un disco da 2 GB con una tabella delle partizioni MBR e il seguente layout:

```
Disk /dev/sdb: 1.9 GiB, 1998631936 bytes, 3903578 sectors
Disk model: DataTraveler 2.0
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x31a83a48
```

Device	Boot	Start	End	Sectors	Size	Id	Type
/dev/sdb1		2048	1050623	1048576	512M	83	Linux
/dev/sdb3		2099200	3147775	1048576	512M	83	Linux

Puoi creare una partizione da 600 MB su di essa? Perché?

2. Su un disco in `/dev/sdc`, abbiamo una prima partizione di 1 GB, contenente circa 256 MB di file. Usando `parted`, come puoi ridurlo in modo che abbia spazio appena sufficiente per i file?

3. Immagina di avere un disco in `/dev/sdb` e di voler creare una partizione di swap da 1 GB all'inizio. Quindi, usando `parted`, crei la partizione con `mkpart primary linux-swaps 0 1024M`. Quindi, abiliti lo swap su questa partizione con `swapon /dev/sdb1`, ma ottieni il seguente messaggio di errore:

```
swapon: /dev/sdb1: read swap header failed
```

Che cosa non ha funzionato?

4. Nel corso di questa lezione, stavi provando alcuni comandi in `parted` ma, per errore, hai cancellato la terza partizione sul tuo disco rigido. Sai che era dopo una partizione EFI da 250 MB e una partizione di swap da 4 GB e aveva una dimensione di 10 GB. Quale comando puoi usare per ripristinarlo?

5. Immagina di avere una partizione inutilizzata da 4 GB su `/dev/sda3`. Usando `fdisk`, quale sarebbe la sequenza di operazioni per trasformarlo in una partizione di swap attiva?

Sommario

In questa lezione abbiamo imparato:

- Come creare una tabella delle partizioni MBR su un disco con `fdisk` e come usarlo per creare ed eliminare partizioni.
- Come creare una tabella delle partizioni MBR su un disco con `gdisk` e come usarlo per creare ed eliminare partizioni.
- Come creare partizioni `ext2`, `ext3`, `ext4`, `XFS`, `VFAT` ed `exFAT`.
- Come utilizzare `parted` per creare, eliminare e ripristinare partizioni su entrambi i dischi MBR e GPT.
- Come utilizzare il ridimensionamento delle partizioni `ext2`, `ext3`, `ext4` e `Brts`.
- Come creare, impostare e attivare partizioni e file di swap.

In questa lezione sono stati discussi i seguenti comandi:

- `fdisk`
- `gdisk`
- `mkfs.ext2`, `mkfs.ext3`, `mkfs.ext4`, `mkfs.xfs`, `mkfs.vfat` and `mkfs.exfat`
- `parted`
- `btrfs`
- `mkswap`
- `swapon` e `swapoff`

Risposte agli Esercizi Guidati

1. Quale schema di partizionamento dovrebbe essere utilizzato per partizionare un disco rigido da 3 TB in tre partizioni da 1 GB? Perché?

GPT, poiché MBR supporta al massimo dischi rigidi da 2 TB.

2. Su `gdisk`, come possiamo sapere quanto spazio è disponibile sul disco?

Usa `p` (print). Lo spazio libero totale verrà visualizzato come ultima riga di informazioni prima della tabella delle partizioni stessa.

3. Quale è il comando per creare un filesystem `ext3`, controllando prima la presenza di blocchi danneggiati, con l'etichetta `MyDisk` e un UUID casuale, sul dispositivo `/dev/sdc1`?

Il comando sarebbe `mkfs.ext3 -c -L MyDisk -U random /dev/sdc1`. In alternativa, si può usare anche `mke2fs -t ext3` al posto di `mkfs.ext3`

4. Utilizzando `parted`, qual è il comando per creare una partizione `ext4` da 300 MB, a partire da 500 MB sul disco?

Usa `mkpart primary ext4 500m 800m`. Ricorda che dovrai creare il filesystem usando `mkfs.ext4`, poiché `parted` non lo fa.

5. Immagina di avere 2 partizioni, una su `/dev/sda1` e l'altra su `/dev/sda2`, entrambe di 20 GB. Come puoi usarli su un singolo filesystem `Btrfs`, in modo tale che il contenuto di una partizione venga automaticamente duplicato sull'altra, come su una configurazione `RAID1`? Quanto sarà grande il filesystem?

Usa `mkfs.btrfs /dev/sda1 /dev/sdb1 -m raid1`. Il filesystem risultante avrà una dimensione di 20 GB, poiché una partizione funge semplicemente da duplicato dell'altra.

Risposte agli Esercizi Esplorativi

1. Considera un disco da 2 GB con una tabella delle partizioni MBR e il seguente layout:

```
Disk /dev/sdb: 1.9 GiB, 1998631936 bytes, 3903578 sectors
Disk model: DataTraveler 2.0
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x31a83a48
```

Device	Boot	Start	End	Sectors	Size	Id	Type
/dev/sdb1		2048	1050623	1048576	512M	83	Linux
/dev/sdb3		2099200	3147775	1048576	512M	83	Linux

Puoi creare una partizione da 600 MB su di essa? Perché?

Non puoi, perché non c'è abbastanza spazio contiguo. Il primo indizio che qualcosa “non va” è l'elenco dei dispositivi: hai `/dev/sdb1` e `/dev/sdb3`, ma non `/dev/sdb2`. Quindi, manca qualcosa.

Devi guardare dove finisce una partizione e inizia l'altra. La partizione uno termina al settore `1050623` e la partizione 2 inizia al `2099200`. Questo è un vuoto di `1048577` settori. A `512` byte per settore, corrisponde a `536.871.424` byte. Se lo dividi per `1024` ottieni `524,288` kilobyte. Dividi di nuovo per `1024` e ottieni ... `512` MB. Questa è la dimensione del “vuoto”.

Se il disco è da `2 GB`, allora abbiamo al massimo altri `512 MB` dopo la partizione 3. Anche se abbiamo in totale circa `1 GB` non allocato, il blocco contiguo più grande è `512 MB`. Quindi, non c'è spazio per una partizione da `600 MB`.

2. Su un disco in `/dev/sdc`, abbiamo una prima partizione di `1 GB`, contenente circa `256 MB` di file. Usando `parted`, come puoi ridurlo in modo che abbia spazio appena sufficiente per i file?

Questa è un'operazione in più parti. Per prima cosa devi ridurre il filesystem usando `resize2fs`. Invece di specificare direttamente la nuova dimensione, puoi usare il parametro `-M` in modo che sia semplicemente “abbastanza grande”. Quindi: `resize2fs -M / dev / sdc1`.

Quindi, ridimensiona la partizione stessa con `parted` usando `resizepart`. Poiché è la prima partizione, possiamo supporre che inizi da zero e termini a `241 MB`. Quindi il comando è `resizepart 1 241M`.

3. Immagina di avere un disco in `/dev/sdb` e di voler creare una partizione di swap da 1 GB all'inizio. Quindi, usando `parted`, crei la partizione con `mkpart primary linux-swap 0 1024M`. Quindi, abiliti lo swap su questa partizione con `swapon /dev/sdb1`, ma ottieni il seguente messaggio di errore:

```
swapon: /dev/sdb1: read swap header failed
```

Che cosa non ha funzionato?

Hai creato una partizione del tipo corretto (`linux-swap`), ma ricorda che `mkpart` *non crea un filesystem*. Hai dimenticato di impostare la partizione come spazio di swap con `mkswap` prima di usarla.

4. Nel corso di questa lezione, stavi provando alcuni comandi in `parted` ma, per errore, hai cancellato la terza partizione sul tuo disco rigido. Sai che era dopo una partizione EFI da 250 MB e una partizione di swap da 4 GB e aveva una dimensione di 10 GB. Quale comando puoi usare per ripristinarlo?

Niente panico, hai tutte le informazioni necessarie per ripristinare la partizione, usa semplicemente `rescue` e fai i conti. Avevi 250 MB + 4.096 MB ($4 * 1024$) prima, quindi il punto di inizio dovrebbe essere di circa 4346 MB. Più 10,240 MB ($10 * 1024$) di dimensione, dovrebbe terminare a 14,586 MB. Quindi, `rescue 4346m 14586m` dovrebbe andare. Potrebbe essere necessario dare un po' di margine di flessibilità per il salvataggio, iniziando un po' prima e terminando un po' dopo, a seconda della geometria del disco.

5. Immagina di avere una partizione inutilizzata da 4 GB su `/dev/sda3`. Usando `fdisk`, quale sarebbe la sequenza di operazioni per trasformarlo in una partizione di swap attiva?

Per prima cosa, cambia il tipo di partizione in “Linux Swap” (82), scrivi le modifiche sul disco ed esci. Quindi, usa `mkswap` per impostare la partizione come area di swap. Quindi, usa `swapon` per attivarla.



104.2 Mantenere l'integrità dei filesystem

Obiettivi LPI di riferimento

LPIC-1 version 5.0, Exam 101, Objective 104.2

Peso

2

Aree di Conoscenza Chiave

- Verificare l'integrità dei filesystem.
- Controllare lo spazio libero e gli inode.
- Riparare semplici problemi di filesystem.

Di seguito è riportato un elenco parziale dei file, dei termini e dei comandi utilizzati

- `du`
- `df`
- `fsck`
- `e2fsck`
- `mke2fs`
- `tune2fs`
- `xfs_repair`
- `xfs_fsr`
- `xfs_db`



104.2 Lezione 1

Certificazione:	LPIC-1
Versione:	5.0
Argomento:	104 Dispositivi, Il Filesystem Linux, Filesystem Hierarchy Standard
Obiettivo:	104.2 Mantenere l'integrità dei filesystem
Lezione:	1 di 1

Introduzione

I moderni filesystem Linux hanno un "diario di bordo". Ciò significa che ogni operazione viene registrata in un log interno (il *journal*) prima di essere eseguita. Se l'operazione viene interrotta a causa di un errore di sistema (come un *kernel panic*, un'interruzione di corrente, ecc.), questa può essere ricostruita controllando il journal, evitando il danneggiamento del filesystem e la perdita di dati.

Ciò riduce notevolmente la necessità di controlli manuali del file system, anche se potrebbero comunque ancora essere necessari. Conoscere gli strumenti specifici utilizzati (e i parametri corrispondenti) può rappresentare la differenza tra una cena a casa con la tua famiglia o una notte intera nella sala server a lavoro.

In questa lezione discuteremo gli strumenti disponibili per monitorare l'utilizzo del filesystem, ottimizzarne il funzionamento e come controllare e riparare i danni.

Controllare l'Utilizzo dei Dischi

Ci sono due comandi che possono essere usati per controllare quanto spazio viene usato e quanto ne resta su un determinato filesystem. Il primo è `du`, che sta per “disk usage” (utilizzo del disco).

`du` è di natura ricorsiva. Nella sua forma più elementare, il comando mostrerà semplicemente quanti blocchi da 1 Kilobyte vengono utilizzati dalla directory corrente e da tutte le sue sottodirectory:

```
$ du
4816 .
```

Questo non è molto utile, quindi possiamo richiedere un output più “human readable” aggiungendo il parametro `-h`:

```
$ du -h
4.8M .
```

Per impostazione predefinita, `du` mostra solo il conteggio dell'utilizzo delle directory (considerando tutti i file e le sottodirectory al suo interno). Per mostrare un conteggio individuale per tutti i file nella directory, usa il parametro `-a`:

```
$ du -ah
432K ./geminoid.jpg
508K ./Linear_B_Hero.jpg
468K ./LG-G8S-ThinQ-Mirror-White.jpg
656K ./LG-G8S-ThinQ-Range.jpg
60K ./Stranger3_Titolo.png
108K ./Baidu_Banho.jpg
324K ./Xiaomi_Mimoji.png
284K ./Mi_CC_9e.jpg
96K ./Mimoji_Comparativo.jpg
32K ./Xiaomi FCC.jpg
484K ./geminoid2.jpg
108K ./Mimoji_Abre.jpg
88K ./Mi8_Hero.jpg
832K ./Tablet_Linear_B.jpg
332K ./Mimoji_Comparativo.png
4.8M .
```


Il comportamento predefinito è mostrare l'utilizzo di ogni sottodirectory, quindi l'utilizzo totale della directory corrente, *inclusa* la sottodirectory:

```
$ du -h
4.8M    ./Temp
6.0M    .
```

Nell'esempio sopra, possiamo vedere che la sottodirectory `Temp` occupa 4.8 MB e la directory corrente, *inclusa* `Temp`, occupa 6.0 MB. Ma quanto spazio occupano i *file* nella directory corrente, escluse le sottodirectory? Per questo abbiamo il parametro `-S`:

```
$ du -Sh
4.8M    ./Temp
1.3M    .
```

TIP

Tieni presente che i parametri della riga di comando fanno distinzione tra maiuscole e minuscole: `-s` è diverso da `-S`.

Se vuoi mantenere questa distinzione tra lo spazio usato dai file nella directory corrente e lo spazio usato dalle sottodirectory, ma vuoi *anche* un totale generale alla fine, puoi aggiungere il parametro `-c`:

```
$ du -Shc
4.8M    ./Temp
1.3M    .
6.0M    total
```

Puoi controllare quanto “profondo” l'output di `du` dovrebbe andare con il parametro `-d N`, dove `N` indica i livelli. Per esempio, se usi il parametro `-d 1`, mostrerà la directory corrente e le sue sottodirectory, ma non le sottodirectory di queste ultime.

Vedi la differenza di seguito. Senza `-d`:

```
$ du -h
216K    ./somedir/anotherdir
224K    ./somedir
232K    .
```

E limitando la profondità a un livello con `-d 1`:

```
$ du -h -d1
224K    ./somedir
232K    .
```

Si noti che anche se `anotherdir` non viene mostrato, le sue dimensioni vengono comunque prese in considerazione.

Potresti voler escludere alcuni tipi di file dal conteggio, cosa che puoi fare con `--exclude="PATTERN"`, dove `PATTERN` è il modello rispetto al quale desideri rapportarti. Considera questa directory:

```
$ du -ah
124K    ./ASM68K.EXE
2.0M    ./Contra.bin
36K     ./fixheadr.exe
4.0K    ./README.txt
2.1M    ./Contra_NEW.bin
4.0K    ./Built.bat
8.0K    ./Contra_Main.asm
4.2M    .
```

Ora, useremo `--exclude` per filtrare ogni file con l'estensione `.bin`:

```
$ du -ah --exclude="*.bin"
124K    ./ASM68K.EXE
36K     ./fixheadr.exe
4.0K    ./README.txt
4.0K    ./Built.bat
8.0K    ./Contra_Main.asm
180K    .
```

Notare che il totale non riflette più la dimensione dei file esclusi.

Controllo dello Spazio Libero

`du` funziona a livello di file. C'è un altro comando che può mostrare l'utilizzo del disco e quanto spazio è disponibile a livello di filesystem. Questo comando è `df`.

Il comando `df` fornirà un elenco di tutti i filesystem disponibili (già montati) sul tuo sistema, inclusa la loro dimensione totale, quanto spazio è stato utilizzato, quanto spazio è disponibile, la

percentuale di utilizzo e dove è montato:

```
$ df
Filesystem      1K-blocks      Used Available Use% Mounted on
udev            2943068          0    2943068   0% /dev
tmpfs           595892        2496    593396   1% /run
/dev/sda1       110722904 25600600 79454800 25% /
tmpfs           2979440     951208    2028232 32% /dev/shm
tmpfs            5120           0        5120   0% /run/lock
tmpfs           2979440          0    2979440   0% /sys/fs/cgroup
tmpfs           595888         24    595864   1% /run/user/119
tmpfs           595888        116    595772   1% /run/user/1000
/dev/sdb1        89111        1550     80824   2% /media/carol/part1
/dev/sdb3        83187        1550     75330   3% /media/carol/part3
/dev/sdb2        90827        1921     82045   3% /media/carol/part2
/dev/sdc1       312570036 233740356 78829680 75% /media/carol/Samsung Externo
```

Tuttavia, mostrare la dimensione in blocchi da 1 KB non è molto facile da usare. Come in `du`, puoi aggiungere l'opzione `-h` per ottenere un output più “human readable”:

```
$ df -h
Filesystem      Size  Used Avail Use% Mounted on
udev            2.9G     0  2.9G   0% /dev
tmpfs           582M  2.5M  580M   1% /run
/dev/sda1       106G   25G   76G  25% /
tmpfs           2.9G  930M  2.0G  32% /dev/shm
tmpfs           5.0M     0  5.0M   0% /run/lock
tmpfs           2.9G     0  2.9G   0% /sys/fs/cgroup
tmpfs           582M   24K  582M   1% /run/user/119
tmpfs           582M  116K  582M   1% /run/user/1000
/dev/sdb1        88M   1.6M   79M   2% /media/carol/part1
/dev/sdb3        82M   1.6M   74M   3% /media/carol/part3
/dev/sdb2        89M   1.9M   81M   3% /media/carol/part2
/dev/sdc1       299G  223G   76G  75% /media/carol/Samsung Externo
```

Puoi anche usare il parametro `-i` per mostrare gli inode usati/disponibili invece dei blocchi:

```
$ df -i
Filesystem      Inodes  IUsed  IFree IUse% Mounted on
udev            737142    547  736595    1% /dev
tmpfs           745218    908  744310    1% /run
/dev/sda6       6766592 307153 6459439    5% /
```

```
tmpfs      745218    215  745003    1% /dev/shm
tmpfs      745218      4  745214    1% /run/lock
tmpfs      745218    18  745200    1% /sys/fs/cgroup
/dev/sda1   62464    355  62109    1% /boot
tmpfs      745218    43  745175    1% /run/user/1000
```

Un parametro utile è `-T`, che stamperà anche il tipo di ogni filesystem:

```
$ df -hT
Filesystem      Type      Size  Used Avail Use% Mounted on
udev            devtmpfs  2.9G     0  2.9G   0% /dev
tmpfs           tmpfs     582M   2.5M  580M   1% /run
/dev/sda1       ext4      106G   25G   76G  25% /
tmpfs           tmpfs     2.9G   930M  2.0G  32% /dev/shm
tmpfs           tmpfs     5.0M     0  5.0M   0% /run/lock
tmpfs           tmpfs     2.9G     0  2.9G   0% /sys/fs/cgroup
tmpfs           tmpfs     582M   24K  582M   1% /run/user/119
tmpfs           tmpfs     582M  116K  582M   1% /run/user/1000
/dev/sdb1       ext4       88M   1.6M   79M   2% /media/carol/part1
/dev/sdb3       ext4       82M   1.6M   74M   3% /media/carol/part3
/dev/sdb2       ext4       89M   1.9M   81M   3% /media/carol/part2
/dev/sdc1       fuseblk   299G  223G   76G  75% /media/carol/Samsung Externo
```

Conoscendo il tipo di filesystem è possibile filtrare l'output. Puoi mostrare solo i filesystem di un dato tipo con `-t TYPE`, o escludere i filesystem di un dato tipo con `-x TYPE`, come negli esempi seguenti.

Escludere i filesystem `tmpfs`:

```
$ df -hx tmpfs
Filesystem      Size  Used Avail Use% Mounted on
udev            2.9G     0  2.9G   0% /dev
/dev/sda1       106G   25G   76G  25% /
/dev/sdb1       88M   1.6M   79M   2% /media/carol/part1
/dev/sdb3       82M   1.6M   74M   3% /media/carol/part3
/dev/sdb2       89M   1.9M   81M   3% /media/carol/part2
/dev/sdc1       299G  223G   76G  75% /media/carol/Samsung Externo
```

Mostrare solo filesystem `ext4`:

```
$ df -ht ext4
```

Filesystem	Size	Used	Avail	Use%	Mounted on
/dev/sda1	106G	25G	76G	25%	/
/dev/sdb1	88M	1.6M	79M	2%	/media/carol/part1
/dev/sdb3	82M	1.6M	74M	3%	/media/carol/part3
/dev/sdb2	89M	1.9M	81M	3%	/media/carol/part2

Puoi anche personalizzare l'output di `df`, selezionando cosa visualizzare e in quale ordine, usando il parametro `--output` = seguito da un elenco di campi separati da virgole che desideri visualizzare. Alcuni dei campi disponibili sono:

source

Il dispositivo corrispondente al filesystem.

fstype

Il tipo di filesystem.

size

La dimensione totale di un filesystem.

used

Quanto spazio viene utilizzato.

avail

Quanto spazio è disponibile.

pcent

La percentuale utilizzata.

target

Dove è montato il filesystem (*mount point*).

Se desideri un output che mostri destinazione, origine, tipo e utilizzo, puoi utilizzare:

```
$ df -h --output=target,source,fstype,pcent
Mounted on      Filesystem      Type      Use%
/dev            udev            devtmpfs   0%
/run            tmpfs           tmpfs      1%
/               /dev/sda1       ext4       25%
/dev/shm        tmpfs           tmpfs      32%
/run/lock       tmpfs           tmpfs      0%
/sys/fs/cgroup  tmpfs           tmpfs      0%
```

/run/user/119	tmpfs	tmpfs	1%
/run/user/1000	tmpfs	tmpfs	1%
/media/carol/part1	/dev/sdb1	ext4	2%
/media/carol/part3	/dev/sdb3	ext4	3%
/media/carol/part2	/dev/sdb2	ext4	3%
/media/carol/Samsung Externo	/dev/sdc1	fuseblk	75%

`df` può anche essere usato per controllare le informazioni sull'inode, passando i seguenti campi a `--output=`:

itotal

Il numero totale di inode nel filesystem.

iused

Il numero di *inode* utilizzati nel filesystem.

ivail

Il numero di inode disponibili nel filesystem.

ipcent

La percentuale di inode utilizzati nel filesystem.

Per esempio:

```
$ df --output=source,fstype,itotal,iused,ipcent
Filesystem      Type      Inodes   IUsed IUse%
udev            devtmpfs  735764    593   1%
tmpfs           tmpfs     744858   1048   1%
/dev/sda1       ext4      7069696 318651   5%
tmpfs           tmpfs     744858    222   1%
tmpfs           tmpfs     744858     3    1%
tmpfs           tmpfs     744858    18   1%
tmpfs           tmpfs     744858    22   1%
tmpfs           tmpfs     744858    40   1%
```

Manutenzione dei filesystem ext2, ext3 e ext4

Per controllare un filesystem dagli errori (e, si spera, correggerli), Linux fornisce l'utilità `fsck` (pensa a “filesystem check” e non dimenticherai mai il nome). Nella sua forma più semplice, puoi invocarlo con `fsck` seguito dalla posizione del filesystem che vuoi controllare:

```
# fsck /dev/sdb1
fsck from util-linux 2.33.1
e2fsck 1.44.6 (5-Mar-2019)
DT_2GB: clean, 20/121920 files, 369880/487680 blocks
```

WARNING

NON eseguire MAI `fsck` (o utilità correlate) su un filesystem montato. In tal caso, i dati potrebbero andare persi.

`fsck` stesso non controllerà il filesystem, chiamerà semplicemente per farlo l'utilità appropriata per il tipo di filesystem. Nell'esempio sopra, poiché non è stato specificato un tipo di filesystem, `fsck` ha assunto un filesystem `ext2/3/4` per impostazione predefinita e chiamato `e2fsck`.

Per specificare un filesystem, usa l'opzione `-t`, seguita dal nome del filesystem, come in `fsck -t vfat /dev/sdc`. In alternativa, puoi chiamare direttamente un'utilità specifica del filesystem, come `fsck.msdos` per i filesystem FAT.

TIP

Digita `fsck` seguito da `Tab` due volte per visualizzare un elenco di tutti i comandi correlati sul tuo sistema.

`fsck` può accettare alcuni argomenti dalla riga di comando. Questi sono alcuni dei più comuni:

-A

Controlla tutti i filesystem elencati in `/etc/fstab`.

-C

Visualizza una barra di avanzamento durante il controllo di un filesystem. Attualmente funziona solo su filesystem `ext2/3/4`.

-N

Visualizza ciò che sarebbe stato fatto e uscirà, senza effettivamente controllare il filesystem.

-R

Se usato insieme a `-A`, salta il controllo del filesystem di root.

-V

Modalità dettagliata, visualizza più informazioni del normale durante il funzionamento. Questo è utile per il debug.

L'utilità specifica per i filesystem `ext2`, `ext3` ed `ext4` è `e2fsck`, chiamata anche `fsck.ext2`, `fsck.ext3` e `fsck.ext4` (questi tre sono semplicemente collegamenti a `e2fsck`). Per impostazione predefinita, viene eseguito in modalità interattiva: quando viene rilevato un errore

del file system, si ferma e chiede all'utente cosa fare. L'utente deve digitare `y` per risolvere il problema, `n` per lasciarlo non risolto o `a` per correggere il problema corrente e tutti i successivi.

Ovviamente sedersi davanti a un terminale in attesa che `e2fsck` ti chieda cosa fare non è un uso produttivo del tuo tempo, specialmente se hai a che fare con un grande filesystem. Quindi, ci sono opzioni che causano l'esecuzione di `e2fsck` in modalità non interattiva:

-p

Tenta di correggere automaticamente eventuali errori trovati. Se viene rilevato un errore che richiede l'intervento dell'amministratore di sistema, `e2fsck` fornisce una descrizione del problema e uscirà.

-y

Questo risponde `y` (yes) a tutte le domande.

-n

L'opposto di `-y`. Oltre a rispondere `n` (no) a tutte le domande, fa sì che il filesystem venga montato in sola lettura, quindi non può essere modificato.

-f

Forza `e2fsck` a controllare un filesystem anche se è contrassegnato come “pulito” (*clean*), cioè è stato correttamente smontato.

Ottimizzazione di un Filesystem ext

I filesystem `ext2`, `ext3` ed `ext4` hanno un numero di parametri che possono essere regolati, o ottimizzati, dall'amministratore per soddisfare meglio le esigenze del sistema. L'utility usata per visualizzare o modificare questi parametri si chiama `tune2fs`.

Per vedere i parametri correnti per un dato filesystem, usa l'opzione `-l` seguita dal dispositivo che rappresenta la partizione. L'esempio seguente mostra l'output di questo comando sulla prima partizione del primo disco (`/dev/sda1`) di una macchina:

```
# tune2fs -l /dev/sda1
tune2fs 1.44.6 (5-Mar-2019)
Filesystem volume name:   <none>
Last mounted on:         /
Filesystem UUID:          6e2c12e3-472d-4bac-a257-c49ac07f3761
Filesystem magic number:  0xEF53
Filesystem revision #:    1 (dynamic)
Filesystem features:      has_journal ext_attr resize_inode dir_index filetype
```



```

needs_recovery extent 64bit flex_bg sparse_super large_file huge_file dir_nlink extra_isize
metadata_csum
Filesystem flags:          signed_directory_hash
Default mount options:    user_xattr acl
Filesystem state:         clean
Errors behavior:          Continue
Filesystem OS type:       Linux
Inode count:              7069696
Block count:              28255605
Reserved block count:     1412780
Free blocks:              23007462
Free inodes:              6801648
First block:              0
Block size:               4096
Fragment size:            4096
Group descriptor size:    64
Reserved GDT blocks:      1024
Blocks per group:         32768
Fragments per group:     32768
Inodes per group:         8192
Inode blocks per group:   512
Flex block group size:    16
Filesystem created:       Mon Jun 17 13:49:59 2019
Last mount time:          Fri Jun 28 21:14:38 2019
Last write time:          Mon Jun 17 13:53:39 2019
Mount count:              8
Maximum mount count:      -1
Last checked:             Mon Jun 17 13:49:59 2019
Check interval:           0 (<none>)
Lifetime writes:          20 GB
Reserved blocks uid:      0 (user root)
Reserved blocks gid:      0 (group root)
First inode:              11
Inode size:               256
Required extra isize:     32
Desired extra isize:      32
Journal inode:            8
First orphan inode:       5117383
Default directory hash:   half_md4
Directory Hash Seed:      fa95a22a-a119-4667-a73e-78f77af6172f
Journal backup:           inode blocks
Checksum type:            crc32c
Checksum:                 0xe084fe23

```

I filesystem ext mantengono un contatore dei montaggi (*Mount counts*). Questo conteggio aumenta di 1 ogni volta che il filesystem viene montato, e quando viene raggiunto un valore di soglia (il *Maximum mount count*) il sistema viene automaticamente controllato con `e2fsck` all'avvio successivo.

Il numero massimo di montaggi può essere impostato con il parametro `-c N`, dove `N` è il numero di volte in cui il filesystem può essere montato senza essere controllato. Il parametro `-C N` imposta il numero di volte che il sistema è stato montato sul valore di `N`. Nota che i parametri della riga di comando fanno distinzione tra maiuscole e minuscole, quindi `-c` è diverso da `-C`.

È anche possibile definire un intervallo di tempo tra le verifiche, con il parametro `-i`, seguito da un numero e dalle lettere `d` per giorni, `m` per mesi e `y` per anni. Per esempio, `-i 10d` controllerebbe il filesystem al successivo riavvio ogni 10 giorni. Usa zero come valore per disabilitare questa funzione.

`-L` può essere usato per impostare un'etichetta per il filesystem. Questa etichetta può contenere fino a 16 caratteri. Il parametro `-U` imposta l'UUID per il filesystem, che è un numero esadecimale a 128 bit. Nell'esempio sopra, l'UUID è `6e2c12e3-472d-4bac-a257-c49ac07f3761`. Sia l'etichetta che l'UUID possono essere usati al posto del nome del dispositivo (come `/dev/sda1`) per montare il filesystem.

L'opzione `-e BEHAVIOUR` definisce il comportamento del kernel quando viene rilevato un errore del filesystem. Ci sono tre possibili comportamenti:

continue

Continuerà l'esecuzione normalmente.

remount-ro

Rimonterà il filesystem in sola lettura.

panic

Provocherà un kernel panic.

Il comportamento predefinito è `continue`. `remount-ro` potrebbe essere utile nelle applicazioni sensibili ai dati, poiché interromperà immediatamente le scritture sul disco, evitando ulteriori potenziali errori.

I filesystem ext3 sono fondamentalmente filesystem ext2 con un journal. Usando `tune2fs` puoi aggiungere un journal a un filesystem ext2, convertendolo così in ext3. La procedura è semplice, basta passare il parametro `-j` a `tune2fs`, seguito dal dispositivo contenente il filesystem:

```
# tune2fs -j /dev/sda1
```

Successivamente, quando si monta il filesystem convertito, non dimenticare di impostare il tipo su `ext3` in modo che il journal possa essere utilizzato.

Quando si ha a che fare con filesystem journal, il parametro `-J` consente di utilizzare parametri aggiuntivi per impostare alcune opzioni del journal, come `-J size=` per impostare la dimensione del journal (in megabyte), `-J location=` per specificare dove il journal dovrebbe essere memorizzato (o un blocco specifico, o una posizione specifica sul disco con suffissi come `M` o `G`) e persino posizionare il journal su un dispositivo esterno con `-J device=`.

È possibile specificare più parametri contemporaneamente separandoli con una virgola. Per esempio: `-J size=10, location=100M,device=/dev/sdb1` creerà un Journal da 10 MB nella posizione 100 MB sul dispositivo `/dev/sdb1`.

WARNING

`tune2fs` ha un'opzione "forza brutta" (*brute force*), `-f`, che lo costringerà a completare un'operazione anche se vengono rilevati errori. Inutile dire che questo dovrebbe essere usato solo con estrema cautela.

Mantenere un filesystem XFS

Per i filesystem XFS, l'equivalente di `fsck` è `xfs_repair`. Se sospetti che qualcosa non vada nel filesystem, la prima cosa da fare è controllarlo per rilevare eventuali danni.

Questo può essere fatto passando il parametro `-n` a `xfs_repair`, seguito dal dispositivo contenente il filesystem. Il parametro `-n` significa "nessuna modifica": il filesystem verrà controllato, verranno segnalati errori ma non verrà effettuata alcuna riparazione:

```
# xfs_repair -n /dev/sdb1
Phase 1 - find and verify superblock...
Phase 2 - using internal log
    - zero log...
    - scan filesystem freespace and inode maps...
    - found root inode chunk
Phase 3 - for each AG...
    - scan (but do not clear) agi unlinked lists...
    - process known inodes and perform inode discovery...
    - agno = 0
    - agno = 1
    - agno = 2
    - agno = 3
```

```

- process newly discovered inodes...
Phase 4 - check for duplicate blocks...
- setting up duplicate extent list...
- check for inodes claiming duplicate blocks...
- agno = 1
- agno = 3
- agno = 0
- agno = 2
No modify flag set, skipping phase 5
Phase 6 - check inode connectivity...
- traversing filesystem ...
- traversal finished ...
- moving disconnected inodes to lost+found ...
Phase 7 - verify link counts...
No modify flag set, skipping filesystem flush and exiting.

```

Se vengono trovati errori, puoi procedere con le riparazioni senza il parametro `-n`, in questo modo: `xfs_repair /dev/sdb1`.

`xfs_repair` accetta un certo numero di opzioni della riga di comando. Tra loro:

-l LOGDEV e -r RTDEV

Questi sono necessari se il filesystem ha un registro esterno e sezioni in tempo reale. In questo caso, sostituire LOGDEV e RTDEV con i dispositivi corrispondenti.

-m N

Viene usato per limitare l'uso della memoria di `xfs_repair` a N megabyte, cosa che può essere utile nelle impostazioni di un server. Secondo la pagina man, per impostazione predefinita `xfs_repair` ridimensionerà il suo utilizzo della memoria secondo necessità, fino al 75% della RAM fisica del sistema.

-d

La modalità "*dangerous*" abiliterà la riparazione dei filesystem che sono montati in sola lettura.

-v

Potresti averlo indovinato: modalità dettagliata. Ogni volta che viene usato questo parametro, la "verbosità" viene aumentata (per esempio, `-vv` visualizzerà più informazioni rispetto a `-v`).

Notare che `xfs_repair` non è in grado di riparare i filesystem con un registro "sporco" (*dirty*). Puoi "azzerare" un registro corrotto con il parametro `-L`, ma tieni presente che questa è una *ultima risorsa* in quanto potrebbe provocare il danneggiamento del filesystem e la perdita di dati.

Per eseguire il debug di un filesystem XFS, è possibile utilizzare l'utility `xfs_db`, come in `xfs_db /dev/sdb1`. Viene utilizzato principalmente per ispezionare vari elementi e parametri del filesystem.

Questa utility ha un prompt interattivo, come `parted`, con molti comandi interni. È disponibile anche un sistema di aiuto: digita `help` per vedere un elenco di tutti i comandi, e `help` seguito dal nome del comando per vedere maggiori informazioni sul comando.

Un altro utile comando è `xfs_fsr`, che può essere usato per riorganizzare (“deframmentare”) un filesystem XFS. Quando viene eseguito senza alcun argomento aggiuntivo, verrà eseguito per due ore e proverà a deframmentare tutti i filesystem XFS montati e scrivibili elencati nel file `/etc/mtab/`. Potrebbe essere necessario installare questa utilità utilizzando il gestore di pacchetti per la distribuzione Linux, poiché potrebbe non far parte di un'installazione predefinita. Per maggiori informazioni consultare la pagina man corrispondente.

Esercizi Guidati

1. Usando `du`, come possiamo controllare quanto spazio viene utilizzato solo dai file nella directory corrente?

2. Usando `df`, elenca le informazioni per ogni filesystem `ext4`, con gli output che includono i seguenti campi, in ordine: *device*, *mount point*, *total number of inodes*, *number of available inodes*, *percentage of free space*.

3. Qual è il comando per eseguire `e2fsck` su `/dev/sdc1` in modalità non interattiva, cercando di correggere automaticamente la maggior parte degli errori?

4. Supponiamo che `/dev/sdb1` sia un filesystem `ext2`. Come puoi convertirlo in `ext3` e allo stesso tempo resettare il conteggio dei montaggi e cambiare la sua etichetta in `UserData`?

5. Come puoi controllare gli errori su un filesystem `XFS`, *senza* riparare i danni riscontrati?

Esercizi Esplorativi

1. Considera di avere un filesystem ext4 su `/dev/sda1` con i seguenti parametri, ottenuti tramite `tune2fs`:

```
Mount count:          8
Maximum mount count:  -1
```

Cosa succederà al prossimo avvio se viene eseguito il comando `tune2fs -c 9 /dev/sda1`?

2. Considera il seguente output di `du -h`:

```
$ du -h
216K    ./somedir/anotherdir
224K    ./somedir
232K    .
```

Quanto spazio è occupato solo dai file nella directory corrente? Come potremmo riscrivere il comando per mostrare più chiaramente queste informazioni?

3. Cosa succederebbe al filesystem ext2 `/dev/sdb1` se venisse eseguito il comando seguente?

```
# tune2fs -j /dev/sdb1 -J device=/dev/sdc1 -i 30d
```

4. Come possiamo controllare gli errori su un filesystem XFS su `/dev/sda1` che ha una sezione di log su `/dev/sdc1`, senza effettivamente fare alcuna riparazione?

5. Qual è la differenza tra i parametri `-T` e `-t` in `df`?

Sommario

In questa lezione abbiamo imparato:

- Come controllare lo spazio utilizzato e quello libero su un filesystem.
- Come adattare l'output di `df` alle proprie esigenze.
- Come controllare l'integrità e riparare un filesystem con `fsck` e `e2fsck`.
- Come mettere a punto un filesystem ext con `tune2fs`.
- Come controllare e riparare i filesystem XFS con `xfs_repair`.

In questa lezione sono stati discussi i seguenti comandi:

du

Visualizza la quantità di spazio su disco in uso su un filesystem.

df

Visualizza la quantità di spazio su disco disponibile (libero) su un filesystem.

fsck

L'utilità di riparazione generica del controllo del file system.

e2fsck

L'utilità di riparazione del controllo del file system specifica per i file system estesi (ext2/3/4).

tune2fs

Modifica i parametri del filesystem su un filesystem esteso (ext2/3/4).

xfs_repair

L'equivalente di `fsck` per i filesystem XFS.

xfs_db

Questa utility è usata per visualizzare vari parametri di un filesystem XFS.

Risposte agli Esercizi Guidati

1. Usando `du`, come possiamo controllare quanto spazio viene utilizzato solo dai file nella directory corrente?

Per prima cosa, usa il parametro `-S` per separare l'output della directory corrente dalle sue sottodirectory. Quindi, usa `-d 0` per limitare la profondità di output a zero, che significa “nessuna sottodirectory”. Non dimenticare `-h` per ottenere un output in un formato “leggibile dall'uomo”:

```
$ du -S -h -d 0
```

0

```
$ du -Shd 0
```

2. Usando `df`, elenca le informazioni per ogni filesystem `ext4`, con gli output che includono i seguenti campi, in ordine: *device*, *mount point*, *total number of inodes*, *number of available inodes*, *percentage of free space*.

Puoi filtrare i filesystem con l'opzione `-t` seguita dal nome del filesystem. Per ottenere l'output desiderato, utilizzare `--output=source,target,itotal,iavail,pcent`. Quindi la risposta è:

```
$ df -t ext4 --output=source,target,itotal,iavail,pcent
```

3. Qual è il comando per eseguire `e2fsck` su `/dev/sdc1` in modalità non interattiva, cercando di correggere automaticamente la maggior parte degli errori?

Il parametro per provare automaticamente a correggere la maggior parte degli errori è `-p`. Quindi la risposta è:

```
# e2fsck -p /dev/sdc1
```

4. Supponiamo che `/dev/sdb1` sia un filesystem `ext2`. Come puoi convertirlo in `ext3` e allo stesso tempo resettare il conteggio dei montaggi e cambiare la sua etichetta in `UserData`?

Ricorda che per convertire un filesystem `ext2` in `ext3` è solo necessario aggiungere un journal, che può essere fatto con il parametro `-j`. Per azzerare il conteggio delle montature, usa `-c 0`.

Per cambiare l'etichetta usa `-L UserData`. La risposta corretta è quindi:

```
# tune2fs -j -C 0 -L UserData /dev/sdb1
```

5. Come puoi controllare gli errori su un filesystem XFS, *senza* riparare i danni riscontrati?

Usa il parametro `-n`, come in `xfs -n`, seguito dal dispositivo corrispondente.

Risposte agli Esercizi Esplorativi

1. Considera di avere un filesystem ext4 su `/dev/sda1` con i seguenti parametri, ottenuti tramite `tune2fs`:

```
Mount count:          8
Maximum mount count:  -1
```

Cosa succederà al prossimo avvio se viene eseguito il comando `tune2fs -c 9 /dev/sda1`?

Il comando imposterà il numero massimo di montaggi per il filesystem a 9. Dato che il numero di montaggi è attualmente 8, il prossimo avvio del sistema provocherà un controllo del filesystem.

2. Considera il seguente output di `du -h`:

```
$ du -h
216K  ./somedir/anotherdir
224K  ./somedir
232K  .
```

Quanto spazio è occupato solo dai file nella directory corrente? Come potremmo riscrivere il comando per mostrare più chiaramente queste informazioni?

Del totale di 232K utilizzati, 224K sono usati dalla sottodirectory `somedir` e dalle sue sottodirectory. Quindi, escludendo quelli, abbiamo 8K occupati dai file nella directory corrente. Questa informazione può essere mostrata più chiaramente usando il parametro `-S`, che separerà le directory nel conteggio.

3. Cosa succederebbe al filesystem ext2 `/dev/sdb1` se venisse eseguito il comando seguente?

```
# tune2fs -j /dev/sdb1 -J device=/dev/sdc1 -i 30d
```

Un journal verrà aggiunto a `/dev/sdb1`, convertendolo in ext3. Il journal sarà memorizzato sul dispositivo `/dev/sdc1` e il filesystem verrà controllato ogni 30 giorni.

4. Come possiamo controllare gli errori su un filesystem XFS su `/dev/sda1` che ha una sezione di log su `/dev/sdc1`, senza effettivamente fare alcuna riparazione?

Usa `xfs_repair`, seguito da `-l /dev/sdc1` per indicare il dispositivo contenente la sezione di

log e `-n` per evitare di apportare modifiche.

```
# xfs_repair -l /dev/sdc1 -n
```

5. Qual è la differenza tra i parametri `-T` e `-t` in `df`?

L'opzione `-T` includerà il tipo di ogni filesystem nell'output di `df`. `-t` invece mostrerà solo i filesystem del tipo dato in output, escludendo tutti gli altri.



104.3 Verificare il montaggio e lo smontaggio dei filesystem

Obiettivi LPI di riferimento

LPIC-1 version 5.0, Exam 101, Objective 104.3

Peso

3

Aree di Conoscenza Chiave

- Montare e smontare manualmente i filesystem.
- Configurare il montaggio del filesystem all'avvio.
- Configurare filesystem rimovibili montabili dall'utente.
- Uso di etichette e UUID per identificare e montare i filesystem.
- Conoscenza delle mount unit di systemd.

Di seguito è riportato un elenco parziale dei file, dei termini e dei comandi utilizzati

- `/etc/fstab`
- `/media/`
- `mount`
- `umount`
- `blkid`
- `lsblk`



104.3 Lezione 1

Certificazione:	LPIC-1
Versione:	5.0
Argomento:	104 Dispositivi, Il Filesystem Linux, Filesystem Hierarchy Standard
Obiettivo:	104.3 Controlla il montaggio e lo smontaggio dei filesystem
Lezione:	1 di 1

Introduzione

Fino ad ora, hai imparato come partizionare i dischi e come creare e mantenere i filesystem su di essi. Tuttavia, prima di poter accedere a un filesystem su Linux, è necessario che sia *montato*.

Ciò significa collegare il filesystem a un punto specifico dell'albero delle directory del sistema, chiamato punto di montaggio (*mount point*). I filesystem possono essere montati manualmente o automaticamente e ci sono molti modi per farlo: ne impareremo alcuni in questa lezione.

Montare e Smontare i Filesystem

Il comando per montare manualmente un filesystem si chiama `mount` e la sua sintassi è:

```
mount -t TYPE DEVICE MOUNTPOINT
```

Dove:

TYPE

Il tipo di filesystem da montare (es. `ext4`, `btrfs`, `exfat`, ecc.).

DEVICE

Il nome della partizione contenente il filesystem (per esempio `/dev/sdb1`)

MOUNTPOINT

Dove verrà montato il filesystem. La directory montata non deve essere vuota, sebbene debba esistere. Qualsiasi file in essa, tuttavia, sarà inaccessibile per nome mentre il filesystem è montato.

Per esempio, per montare un'unità flash USB contenente un filesystem exFAT situato su `/dev/sdb1` in una directory chiamata `flash` sotto la tua home directory, potresti usare:

```
# mount -t exfat /dev/sdb1 ~/flash/
```

TIP

Molti sistemi Linux usano la shell Bash, e su questi la tilde `~` sul percorso del punto di montaggio è una scorciatoia per la directory home dell'utente corrente. Se l'utente corrente si chiama `john`, per esempio, sarà sostituito da `/home/john`.

Dopo il montaggio, il contenuto del filesystem sarà accessibile nella directory `~/flash`:

```
$ ls -lh ~/flash/
total 469M
-rwxrwxrwx 1 root root 454M jul 19 09:49 lineage-16.0-20190711-MOD-quark.zip
-rwxrwxrwx 1 root root 16M jul 19 09:44 twrp-3.2.3-mod_4-quark.img
```

Elencare i Filesystem Montati

Se digiti solo `mount`, otterrai un elenco di tutti i filesystem attualmente montati sul tuo sistema. Questo elenco può essere abbastanza esteso, perché oltre ai dischi collegati al sistema, contiene anche un certo numero di filesystem runtime in memoria che servono a vari scopi. Per filtrare l'output, puoi usare il parametro `-t` per elencare solo i filesystem del tipo corrispondente, come di seguito:

```
# mount -t ext4
/dev/sda1 on / type ext4 (rw,noatime,errors=remount-ro)
```

Puoi specificare più filesystem contemporaneamente separandoli con una virgola:

```
# mount -t ext4,fuseblk
/dev/sda1 on / type ext4 (rw,noatime,errors=remount-ro)
/dev/sdb1 on /home/carol/flash type fuseblk
(rw,nosuid,nodev,relatime,user_id=0,group_id=0,default_permissions,allow_other,blksize=4096)
[DT_8GB]
```

L'output negli esempi precedenti può essere descritto nel formato:

```
SOURCE on TARGET type TYPE OPTIONS
```

Dove **SOURCE** è la partizione che contiene il filesystem, **TARGET** è la directory in cui è montato, **TYPE** è il tipo di filesystem e **OPTIONS** sono le opzioni passate al comando `mount` al momento del montaggio.

Parametri Aggiuntivi della Riga di Comando

Ci sono molti parametri della riga di comando che possono essere usati con `mount`. Alcuni dei più utilizzati sono:

-a

Monta tutti i filesystem elencati nel file `/etc/fstab` (ne parleremo nella prossima sezione).

-o o --options

`-o` or `--options`:: Passa un elenco di *opzioni di montaggio* separate da virgole al comando `mount`, tale che può cambiare il modo in cui il filesystem verrà montato. Queste opzioni saranno discusse insieme a `/etc/fstab`.

-r o -ro

Monta il filesystem in sola lettura (read-only).

-w or -rw

Monta il filesystem in modalità scrivibile (writable).

Per *smontare* un filesystem, usa il comando `umount`, seguito dal nome del dispositivo o dal punto di montaggio. Considerando l'esempio sopra, i comandi seguenti sono intercambiabili:

```
# umount /dev/sdb1
# umount ~/flash
```


Alcuni dei parametri della riga di comando per `umount` sono:

-a

Smonta tutti i filesystem elencati in `/etc/fstab`.

-f

Forza lo smontaggio di un filesystem. Può essere utile se hai montato un filesystem remoto che è diventato irraggiungibile.

-r

Se il filesystem non può essere smontato, prova a renderlo di sola lettura.

Trattare con i File Aperti

Quando si smonta un filesystem, si può incontrare un messaggio di errore che indica che la destinazione è occupata (*target is busy*). Ciò accadrà se risultano aperti dei file su di esso. Tuttavia, potrebbe non essere immediatamente ovvio dove si trova un file aperto o cosa accede al filesystem.

In questi casi puoi usare il comando `lsof`, seguito dal nome del dispositivo che contiene il filesystem, per vedere un elenco dei processi che vi accedono e quali file sono aperti. Per esempio:

```
# umount /dev/sdb1
umount: /media/carol/External_Drive: target is busy.

# lsof /dev/sdb1
COMMAND  PID   USER   FD   TYPE DEVICE SIZE/OFF NODE NAME
evince   3135 carol   16r   REG   8,17 21881768 5195 /media/carol/External_Drive/Documents/E-
Books/MagPi40.pdf
```

`COMMAND` è il nome dell'eseguibile che ha aperto il file e `PID` è il numero del processo. `NAME` è il nome del file che è aperto. Nell'esempio sopra, il file `MagPi40.pdf` è aperto dal programma `evince` (un visualizzatore di PDF). Se chiudiamo il programma, saremo in grado di smontare il filesystem.

Prima che appaia l'output di `lsof`, gli utenti di GNOME potrebbero visualizzare un messaggio di avviso nella finestra del terminale.

NOTE

```
lsof: WARNING: can't stat() fuse.gvfsd-fuse file system /run/user/1000/gvfs
Output information may be incomplete.
```

`lsuf` cerca di elaborare tutti i filesystem montati. Questo messaggio di avviso viene generato perché `lsuf` ha rilevato uno GNOME Virtual file system (GVFS). Questo è un caso speciale di un filesystem nello spazio utente (FUSE). Funge da ponte tra GNOME, le sue API e il kernel. Nessuno, nemmeno `root`, può accedere a uno di questi file system, a parte il proprietario che lo ha montato (in questo caso, GNOME). Puoi ignorare questo avviso.

Dove Montare?

Puoi montare un filesystem ovunque tu voglia. Tuttavia, ci sono alcune buone pratiche che dovrebbero essere seguite per rendere più facile l'amministrazione del sistema.

Tradizionalmente, `/mnt` era la directory in cui sarebbero stati montati tutti i dispositivi esterni e un certo numero di “punti di ancoraggio” preconfigurati per dispositivi comuni, come unità CD-ROM (`/mnt/cdrom`) e floppy disk (`/mnt/floppy`).

Questo è stato sostituito da `/media`, che ora è il punto di montaggio predefinito per qualsiasi supporto rimovibile collegato dall'utente al sistema.(per esempio dischi esterni, unità flash USB, lettori di schede di memoria, ecc.).

Sulla maggior parte delle moderne distribuzioni Linux in ambienti desktop, i dispositivi rimovibili vengono montati automaticamente su `/media/USER/LABEL` quando sono collegati al sistema, dove `USER` è il nome utente e `LABEL` è l'etichetta del dispositivo . Per esempio, flash USB con l'etichetta `FlashDrive` collegata da `john` verrebbe montata sotto `/media/john/FlashDrive/`. Il modo in cui ciò viene gestito può dipendere dall'ambiente desktop utilizzato.

Detto questo, ogni volta che è necessario montare *manualmente* un filesystem, è buona norma farlo sotto `/mnt`.

Montare un Filesystem all'Avvio

Il file `/etc/fstab` contiene le descrizioni sui filesystem che devono essere montati. Questo è un file di testo, in cui ogni riga descrive un filesystem da montare, con sei campi per riga nel seguente ordine:

```
FILESYSTEM MOUNTPOINT TYPE OPTIONS DUMP PASS
```

Dove:

FILESYSTEM

Il dispositivo contenente il filesystem da montare. Al posto del dispositivo, puoi specificare l'UUID o l'etichetta della partizione, cosa di cui parleremo più avanti.

MOUNTPoint

Dove il filesystem verrà montato.

TYPE

Il tipo di filesystem.

OPTIONS

Opzioni di montaggio che verranno passate a `mount`.

DUMP

Indica se un qualsiasi filesystem di tipo `ext2`, `ext3` o `ext4` debba essere considerato per il backup dal comando `dump`. Di solito è zero, il che significa che dovrebbero essere ignorati.

PASS

Quando è diverso da zero, definisce l'ordine in cui i filesystem verranno controllati all'avvio.

Per esempio, la prima partizione sul primo disco di una macchina potrebbe essere descritta come:

```
/dev/sda1 / ext4 noatime,errors
```

Le opzioni di montaggio su `OPTIONS` sono un elenco di parametri separati da virgole, che possono essere generici o specifici del filesystem. Tra quelli generici abbiamo:

atime e noatime

Per impostazione predefinita, ogni volta che un file viene letto, le informazioni sul tempo di accesso vengono aggiornate. Disabilitarlo (con `noatime`) può velocizzare l'I/O del disco. Non confonderlo con l'ora della modifica, che viene aggiornata ogni volta che viene scritto un file.

auto e noauto

Se il filesystem può (o non può) essere montato automaticamente con `mount -a`.

defaults

Questo passa le opzioni `rw`, `suid`, `dev`, `exec`, `auto`, `nouser` e `async` a `mount`.

dev e nodev

Indica se devono essere interpretati i dispositivi a caratteri o a blocchi nel filesystem montato.

exec e noexec

Consente o nega il permesso di eseguire file binari sul filesystem.

user e nouser

Consente (o meno) a un utente normale di montare il filesystem.

group

Consente a un utente di montare il filesystem se l'utente appartiene allo stesso gruppo che possiede il dispositivo che lo contiene.

owner

Consente a un utente di montare un filesystem se è il proprietario del dispositivo che lo contiene.

suid e nosuid

Consente o meno ai bit SETUID e SETGID di avere effetto.

ro e rw

Monta un filesystem in sola lettura o in modalità scrivibile.

remount

Tenterà di rimontare un filesystem già montato. Non è usato su `/etc/fstab`, ma come parametro per `mount -o`. Per esempio, per rimontare la partizione già montata `/dev/sdb1` in sola lettura, potresti usare il comando `mount -o remount, ro /dev/sdb1`. Quando si rimonta, non è necessario specificare il tipo di filesystem, solo il nome del dispositivo o il punto di montaggio.

sync e async

Indica se eseguire tutte le operazioni di I/O sul filesystem in modo sincrono o asincrono. `async` è normalmente l'impostazione predefinita. La pagina di manuale per `mount` avverte che l'uso di `sync` su un supporto con un numero limitato di cicli di scrittura (come unità flash o schede di memoria) può ridurre la durata del dispositivo.

Utilizzare UUID ed Etichette

Specificare il nome del dispositivo contenente il filesystem da montare può introdurre alcuni problemi. A volte lo stesso nome del dispositivo può essere assegnato a un altro dispositivo a seconda di quando o dove è stato collegato al sistema. Per esempio, un'unità flash USB su `/dev/sdb1` può essere assegnata a `/dev/sdc1` se collegata a un'altra porta o dopo un'altra unità flash.

Un modo per evitarlo è specificare l'etichetta o l'UUID (*Universally Unique Identifier*) del volume. Entrambi sono specificati quando il filesystem viene creato e non cambieranno, a meno che il filesystem non venga distrutto o assegnato manualmente una nuova etichetta o UUID.

Il comando `lsblk` può essere usato per avere informazioni su un filesystem e scoprire l'etichetta e/o l'UUID ad esso associati. Per fare ciò, usa il parametro `-f`, seguito dal nome del dispositivo:

```
$ lsblk -f /dev/sda1
NAME FSTYPE LABEL UUID                                FSAVAIL FSUSE% MOUNTPOINT
sda1 ext4          6e2c12e3-472d-4bac-a257-c49ac07f3761    64,9G    33% /
```

Ecco il significato di ogni colonna:

NAME

Nome del dispositivo contenente il file system.

FSTYPE

Tipo di filesystem.

LABEL

Etichetta del filesystem.

UUID

Universally Unique Identifier (UUID) assegnato al filesystem.

FSAVAIL

Quanto spazio è disponibile nel filesystem.

FSUSE%

Percentuale di utilizzo del filesystem.

MOUNTPOINT

Dove è montato il filesystem.

In `/etc/fstab` un dispositivo può essere specificato dal suo UUID con l'opzione `UUID=`, seguita dall'UUID, o con `LABEL=`, seguita dall'etichetta. Quindi, invece di:

```
/dev/sda1 / ext4 noatime,errors
```

Useresti:

```
UUID=6e2c12e3-472d-4bac-a257-c49ac07f3761 / ext4 noatime,errors
```

Oppure, se hai un disco con l'etichetta `homedisk`:

```
LABEL=homedisk /home ext4 defaults
```

La stessa sintassi può essere utilizzata con il comando `mount`. Invece del nome del dispositivo, passa l'UUID o l'etichetta. Per esempio, per montare un disco NTFS esterno con l'UUID `56C11DCC5D2E1334` su `/mnt/external`, il comando sarebbe:

```
$ mount -t ntfs UUID=56C11DCC5D2E1334 /mnt/external
```

Montare Dischi con Systemd

Systemd è il processo *init*, il primo processo da eseguire su molte distribuzioni Linux. È responsabile della generazione di altri processi, dell'avvio dei servizi e del bootstrap del sistema. Tra molte altre attività, *systemd* può essere utilizzato anche per gestire il montaggio (e il montaggio automatico) dei file system.

Per utilizzare questa funzione di *systemd*, è necessario creare un file di configurazione chiamato *mount unit*. Ogni volume da montare ha la propria *mount unit* che deve essere posizionata in `/etc/systemd/system/`.

Le unità di montaggio (mount unit) sono semplici file di testo con l'estensione `.mount`. Il formato di base è mostrato di seguito:

```
[Unit]
Description=

[Mount]
What=
Where=
Type=
Options=

[Install]
WantedBy=
```

Description=

Breve descrizione dell'unità di montaggio, qualcosa come "Monta il disco di backup".

What=

Che cosa dovrebbe essere montato. Il volume deve essere specificato come `/dev/disk/by-uuid/VOL_UUID` dove `VOL_UUID` è l'UUID del volume.

Where=

Dovrebbe essere il percorso completo in cui deve essere montato il volume.

Type=

Il tipo di filesystem.

Options=

Le opzioni di montaggio che potresti voler passare, sono le stesse usate con il comando `mount` o in `/etc/fstab`.

WantedBy=

Utilizzato per la gestione delle dipendenze. In questo caso, useremo `multi-user.target`, il che significa che ogni volta che il sistema si avvia in un ambiente multiutente (un avvio normale) l'unità verrà montata.

Il nostro precedente esempio del disco esterno potrebbe essere scritto come di seguito:

```
[Unit]
Description=External data disk

[Mount]
What=/dev/disk/by-uuid/56C11DCC5D2E1334
Where=/mnt/external
Type=ntfs
Options=defaults

[Install]
WantedBy=multi-user.target
```

Ma non abbiamo ancora finito. Per funzionare correttamente, l'unità di montaggio *deve* avere lo stesso nome del punto di montaggio. In questo caso, il punto di montaggio è `/mnt/external`, quindi il file dovrebbe essere chiamato `mnt-external.mount`.

Successivamente, è necessario riavviare il demone `systemd` con il comando `systemctl` e avviare

l'unità:

```
# systemctl daemon-reload
# systemctl start mnt-external.mount
```

Ora il contenuto del disco esterno dovrebbe essere disponibile su `/mnt/external`. Puoi controllare lo stato del montaggio con il comando `systemctl status mnt-external.mount`, come di seguito:

```
# systemctl status mnt-external.mount
• mnt-external.mount - External data disk
  Loaded: loaded (/etc/systemd/system/mnt-external.mount; disabled; vendor pres
  Active: active (mounted) since Mon 2019-08-19 22:27:02 -03; 14s ago
    Where: /mnt/external
    What: /dev/sdb1
   Tasks: 0 (limit: 4915)
  Memory: 128.0K
   CGroup: /system.slice/mnt-external.mount

ago 19 22:27:02 pop-os systemd[1]: Mounting External data disk...
ago 19 22:27:02 pop-os systemd[1]: Mounted External data disk.
```

Il comando `systemctl start mnt-external.mount` abiliterà l'unità solo per la sessione corrente. Se vuoi abilitarlo a ogni avvio, sostituisci `start` con `enable`:

```
# systemctl enable mnt-external.mount
```

Montaggio automatico di una Mount Unit

Le unità di montaggio possono essere montate automaticamente ogni volta che si accede al punto di montaggio. Per fare questo, hai bisogno di un file `.automount`, accanto al file `.mount` che descrive l'unità. Il formato di base è:

```
[Unit]
Description=

[Automount]
Where=

[Install]
```



```
WantedBy=multi-user.target
```

Come prima, `Description=` è una breve descrizione del file e `Where=` è il mountpoint. Per esempio, un file `.automount` per il nostro esempio precedente sarebbe:

```
[Unit]
Description=Automount for the external data disk

[Automount]
Where=/mnt/external

[Install]
WantedBy=multi-user.target
```

Salva il file con lo stesso nome del punto di montaggio (in questo caso, `mnt-external.automount`), ricarica `systemd` e avvia l'unità:

```
# systemctl daemon-reload
# systemctl start mnt-external.automount
```

Ora ogni volta che si accede alla directory `/mnt/external`, il disco viene montato. Come prima, per abilitare l'automount a ogni avvio useresti:

```
# systemctl enable mnt-external.automount
```

Esercizi Guidati

1. Usando `mount`, come puoi montare un filesystem `ext4` su `/dev/sdc1` all'interno di `/mnt/external` in sola lettura, usando le opzioni `noatime` e `async`?

2. Quando si smonta un filesystem su `/dev/sdd2`, viene visualizzato il messaggio di errore `target is busy`. Come puoi scoprire quali file sul filesystem sono aperti e quali processi li hanno aperti?

3. Considera la seguente voce in `/etc/fstab`: `/dev/sdb1 /data ext4 noatime,noauto,async`. Questo filesystem verrà montato se viene emesso il comando `mount -a`? Perché?

4. Come puoi trovare l'UUID di un filesystem sotto `/dev/sdb1`?

5. Come puoi usare `mount` per rimontare in sola lettura un filesystem exFAT con UUID `6e2c12e3-472d-4bac-a257-c49ac07f3761`, montato su `/mnt/data`?

6. Come puoi ottenere un elenco di tutti i filesystem `ext3` e `ntfs` attualmente montati su un sistema?

Esercizi Esplorativi

1. Considera la seguente voce in `/etc/fstab`: `/dev/sdc1 /backup ext4 noatime,nouser,async`. Un utente può montare questo filesystem con il comando `mount /backup`? Perché?

2. Considera un filesystem remoto montato su `/mnt/server`, che è diventato irraggiungibile a causa di una perdita di connettività di rete. Come potresti forzare lo smontaggio, o il montaggio in sola lettura se ciò non fosse possibile?

3. Scrivi una voce in `/etc/fstab` che monti un volume `btrfs` con l'etichetta `Backup` su `/mnt/backup`, con le opzioni predefinite e senza consentire l'esecuzione di binari da esso.

4. Considera la seguente unità di montaggio `systemd`:

```
[Unit]
Description=External data disk

[Mount]
What=/dev/disk/by-uuid/56C11DCC5D2E1334
Where=/mnt/external
Type=ntfs
Options=defaults

[Install]
WantedBy=multi-user.target
```

Quale sarebbe una voce in `/etc/fstab` equivalente per questo filesystem?

5. Quale dovrebbe essere il nome del file per l'unità nell'esercizio precedente, in modo che possa essere utilizzato da `systemd`? Dove va posizionato?

Sommario

In questa lezione, hai imparato come montare e smontare i filesystem, manualmente o automaticamente. Alcuni dei comandi e dei concetti spiegati sono stati:

- `mount` (monta un dispositivo in una posizione)
- `umount` (smonta un dispositivo)
- `lsdf` (elenca i processi che accedono a un filesystem)
- Le directory `/mnt` e `/media`
- `/etc/fstab`
- `lsblk` (elenca il tipo e l'UUID di un filesystem)
- Come montare un filesystem usando il suo UUID o etichetta.
- Come montare un filesystem usando le unità di montaggio di `systemd`.
- Come montare automaticamente un filesystem utilizzando le unità di montaggio `systemd`.

Risposte agli Esercizi Guidati

1. Usando `mount`, come puoi montare un filesystem `ext4` su `/dev/sdc1` all'interno di `/mnt/external` in sola lettura, usando le opzioni `noatime` e `async`?

```
# mount -t ext4 -o noatime,async,ro /dev/sdc1 /mnt/external
```

2. Quando si smonta un filesystem su `/dev/sdd2`, viene visualizzato il messaggio di errore `target is busy`. Come puoi scoprire quali file sul filesystem sono aperti e quali processi li hanno aperti?

Utilizza `lsof` seguito dal nome del device:

```
$ lsof /dev/sdd2
```

3. Considera la seguente voce in `/etc/fstab`: `/dev/sdb1 /data ext4 noatime,noauto,async`. Questo filesystem verrà montato se viene emesso il comando `mount -a`? Perché?

Non verrà montato. La chiave è il parametro `noauto`, il che significa che questa voce deve essere ignorata da `mount -a`.

4. Come puoi trovare l'UUID di un filesystem sotto `/dev/sdb1`?

Utilizza `lsblk -f`, seguito dal nome del device:

```
$ lsblk -f /dev/sdb1
```

5. Come puoi usare `mount` per rimontare in sola lettura un filesystem exFAT con UUID `6e2c12e3-472d-4bac-a257-c49ac07f3761`, montato su `/mnt/data`?

Dato che il filesystem è montato, non devi preoccuparti del tipo di filesystem o dell'ID, usa semplicemente l'opzione `remount` con il parametro `ro` (sola lettura) e il punto di montaggio:

```
# mount -o remount,ro /mnt/data
```

6. Come puoi ottenere un elenco di tutti i filesystem `ext3` e `ntfs` attualmente montati su un sistema?

Utilizza `mount -t`, seguito da una lista separata da virgole dei filesystem:

```
# mount -t ext3,ntfs
```

Risposte agli Esercizi Esplorativi

1. Considera la seguente voce in `/etc/fstab`: `/dev/sdc1 /backup ext4 noatime,nouser,async`. Un utente può montare questo filesystem con il comando `mount /backup`? Perché?

No, il parametro `nouser` non consentirà agli utenti ordinari di montare questo filesystem.

2. Considera un filesystem remoto montato su `/mnt/server`, che è diventato irraggiungibile a causa di una perdita di connettività di rete. Come puoi forzare lo smontaggio o il montaggio in sola lettura se ciò non è possibile?

Passa le opzioni `-f` e `-r` per smontare. Il comando sarebbe `umount -f -r /mnt/server`. Ricorda che puoi raggruppare i parametri, quindi funzionerebbe anche `umount -fr /mnt/server`

3. Scrivi una voce in `/etc/fstab` che monti un volume `btrfs` con l'etichetta `Backup` su `/mnt/backup`, con le opzioni predefinite e senza consentire l'esecuzione di binari da esso.

La riga dovrebbe essere `LABEL=Backup /mnt/backup btrfs defaults,noexec`

4. Considera la seguente unità di montaggio `systemd`:

```
[Unit]
Description=External data disk

[Mount]
What=/dev/disk/by-uuid/56C11DCC5D2E1334
Where=/mnt/external
Type=ntfs
Options=defaults

[Install]
WantedBy=multi-user.target
```

Quale sarebbe una voce in `/etc/fstab` equivalente per questo filesystem?

La voce sarebbe: `UUID=56C11DCC5D2E1334 /mnt/external ntfs defaults`

5. Quale dovrebbe essere il nome del file per l'unità nell'esercizio precedente, in modo che possa essere utilizzato da `systemd`? Dove va posizionato?

Il nome del file deve essere lo stesso del punto di montaggio, quindi `mnt-external.mount`, posto in `/etc/systemd/system`.



104.5 Gestire le autorizzazioni e la proprietà dei file

Obiettivi LPI di riferimento

LPIC-1 version 5.0, Exam 101, Objective 104.5

Peso

3

Aree di Conoscenza Chiave

- Gestire i permessi di accesso su file regolari e file speciali e directory.
- Usare modalità di accesso come suid, sgid e sticky bit per mantenere la sicurezza.
- Saper cambiare la maschera di creazione del file.
- Utilizzare il campo gruppo per concedere l'accesso ai file ai membri di un gruppo.

Di seguito è riportato un elenco parziale dei file, dei termini e dei comandi utilizzati

- `chmod`
- `umask`
- `chown`
- `chgrp`



104.5 Lezione 1

Certificazione:	LPIC-1
Versione:	5.0
Argomento:	104 Dispositivi, Il Filesystem Linux, Filesystem Hierarchy Standard
Obiettivo:	104.5 Gestire le autorizzazioni e le proprietà dei file
Lezione:	1 di 1

Introduzione

Essendo un sistema multiutente Linux ha bisogno di un modo per tenere traccia del possesso di ogni file e se un utente è autorizzato o meno a eseguire azioni su un file. Questo per garantire la privacy degli utenti che potrebbero voler mantenere riservato il contenuto dei propri file, nonché per garantire la collaborazione rendendo alcuni file accessibili a più utenti.

Tutto ciò viene gestito attraverso un sistema di autorizzazioni a tre livelli. Ogni file su disco è di proprietà di un utente e di un gruppo di utenti e dispone di tre serie di autorizzazioni: una per il suo proprietario, una per il gruppo che possiede il file e una per tutti gli altri. In questa lezione imparerai come interrogare i permessi di un file, il significato di questi permessi e come manipolarli.

Richiedere Informazioni Riguardo File e Directory

Il comando `ls` viene utilizzato per ottenere un elenco dei contenuti di qualsiasi directory. In questa forma base, tutto ciò che ottieni sono i nomi dei file:

```
$ ls
Another_Directory  picture.jpg  text.txt
```

Ma sono disponibili molte più informazioni per ogni file, inclusi il tipo, le dimensioni, la proprietà e altro ancora. Per vedere queste informazioni devi chiedere a `ls` un elenco in *forma lunga*, usando il parametro `-l`:

```
$ ls -l
total 536
drwxrwxr-x 2 carol carol 4096 Dec 10 15:57 Another_Directory
-rw----- 1 carol carol 539663 Dec 10 10:43 picture.jpg
-rw-rw-r-- 1 carol carol 1881 Dec 10 15:57 text.txt
```

Ogni colonna dell'output sopra ha un significato. Diamo uno sguardo alle colonne rilevanti per questa lezione.

- La *prima* colonna nell'elenco mostra il tipo di file e le autorizzazioni. Per esempio su `drwxrwxr-x`:
 - Il primo carattere, `d`, indica il tipo di file.
 - I successivi tre caratteri, `rw`, indicano i permessi per il proprietario del file, denominato anche *user* o `u`.
 - I successivi tre caratteri, `rw`, indicano i permessi del gruppo che possiede il file, denominato anche *group* o `g`.
 - Gli ultimi tre caratteri, `r-x`, indicano i permessi per chiunque altro, noto anche come *others* o `o`.

TIP

È anche comune indicare il set di permessi *others* come permessi *mondiali*, come in “Tutti gli altri nel mondo hanno questi permessi”.

- La *terza* e la *quarta* colonna mostrano le informazioni sulla proprietà: rispettivamente l'utente e il gruppo che possiede il file.
- La *settima* e ultima colonna mostra il nome del file.

La *seconda* colonna indica il numero di *hard link* che puntano a quel file. La *quinta* colonna mostra la dimensione del file. La *sesta* colonna mostra la data e l'ora dell'ultima modifica del file. Ma queste colonne non sono rilevanti per l'argomento corrente.

E Riguardo le Directory?

Se provi a ricercare informazioni su una directory usando `ls -l`, ti verrà mostrato invece un elenco dei suoi contenuti:

```
$ ls -l Another_Directory/
total 0
-rw-r--r-- 1 carol carol 0 Dec 10 17:59 another_file.txt
```

Per evitare ciò e richiedere informazioni sulla directory stessa, aggiungi il parametro `-d` a `ls`:

```
$ ls -l -d Another_Directory/
drwxrwxr-x 2 carol carol 4096 Dec 10 17:59 Another_Directory/
```

Visualizzare i File Nascosti

L'elenco delle directory che abbiamo recuperato utilizzando in precedenza `ls -l` è incompleto:

```
$ ls -l
total 544
drwxrwxr-x 2 carol carol 4096 Dec 10 17:59 Another_Directory
-rw----- 1 carol carol 539663 Dec 10 10:43 picture.jpg
-rw-rw-r-- 1 carol carol 1881 Dec 10 15:57 text.txt
```

Ci sono altri tre file in quella directory, ma sono nascosti. Su Linux, i file il cui nome inizia con un punto (.) vengono automaticamente nascosti. Per vederli dobbiamo aggiungere il parametro `-a` a `ls`

```
$ ls -l -a
total 544
drwxrwxr-x 3 carol carol 4096 Dec 10 16:01 .
drwxrwxr-x 4 carol carol 4096 Dec 10 15:56 ..
drwxrwxr-x 2 carol carol 4096 Dec 10 17:59 Another_Directory
-rw----- 1 carol carol 539663 Dec 10 10:43 picture.jpg
-rw-rw-r-- 1 carol carol 1881 Dec 10 15:57 text.txt
-rw-r--r-- 1 carol carol 0 Dec 10 16:01 .thisIsHidden
```

Il file `.thisIsHidden` è semplicemente nascosto perché il suo nome inizia con `..`.

Le directory `.` e `..` tuttavia sono speciali. `.` è un puntatore alla directory corrente. E `..` è un puntatore alla directory padre, quella che contiene quella corrente. In Linux, *ogni* directory contiene almeno queste due directory.

TIP

Puoi combinare più parametri per `ls` (come in molti altri comandi Linux). `ls -l -a` può, per esempio, essere scritto come `ls -la`.

Comprendere i Tipi di File

Abbiamo già detto che la prima lettera in ogni output di `ls -l` descrive il tipo di file. I tre tipi di file più comuni sono:

- (normal file)

Un file può contenere dati di qualsiasi tipo e aiutarci a gestirli. I file possono essere modificati, spostati, copiati ed eliminati.

d (directory)

Una directory contiene altri file o directory e aiuta a organizzare il file system. Tecnicamente, le directory sono un tipo speciale di file.

l (symbolic link)

Questo “file” è un puntatore a un altro file o directory memorizzato altrove nel filesystem.

Oltre a questi, ci sono altri tre tipi di file che dovresti conoscere, ma che non rientrano nell’ambito di questa lezione:

b (block device)

Questo file sta per un dispositivo virtuale o fisico, solitamente dischi o altri tipi di dispositivi di memorizzazione, come il primo disco rigido che potrebbe essere rappresentato da `/dev/sda`.

c (character device)

Questo file sta per un dispositivo virtuale o fisico. I terminali (come il terminale principale su `/dev/ttyS0`) e le porte seriali sono esempi comuni di dispositivi a caratteri.

s (socket)

I socket servono come “tramite” per il passaggio di informazioni tra due programmi.

WARNING

Non modificare nessuna delle autorizzazioni su dispositivi a blocchi, dispositivi a caratteri o socket, a meno che tu non sappia cosa stai facendo. Questo potrebbe impedire al tuo sistema di funzionare!

Comprendere i Permessi

Nell'output di `ls -l` i permessi del file sono mostrati subito dopo il tipo di file, come tre gruppi di tre caratteri ciascuno, nell'ordine `r`, `w` e `x`. Ecco di seguito il loro significato. Tieni presente che un trattino `-` rappresenta la mancanza di un'autorizzazione.

Permessi sui File

r

Sta per *lettura (read)* e ha un valore ottale di 4 (non preoccuparti, discuteremo a breve degli ottali). Ciò indica il permesso di aprire un file e leggerne il contenuto.

w

Sta per *scrittura (write)* e ha un valore ottale di 2. Ciò indica il permesso di modificare un file.

x

Sta per *esecuzione (execute)* e ha un valore ottale di 1. Ciò indica che il file può essere eseguito come eseguibile o script.

Quindi, per esempio, un file con i permessi `rw-` può essere letto e modificato, ma non può essere eseguito.

Permessi sulle Directory

r

Sta per *lettura (read)* e ha un valore ottale di 4. Ciò indica il permesso di leggere il contenuto della directory, come i nomi dei file. Ma *non* implica il permesso di leggere i file stessi.

w

Sta per *scrittura (write)* e ha un valore ottale di 2. Ciò indica il permesso di creare o eliminare file in una directory o di modificarne nomi, permessi e proprietari.

Se un utente ha il permesso `w` su una directory, l'utente può modificare i permessi di qualsiasi file nella directory (i *contenuti* della directory), anche se l'utente non ha i permessi sul file o se il file è di proprietà di un altro utente.

Tieni presente che avere i permessi di scrittura su una directory o un file non significa avere il permesso di rimuovere o rinominare la directory o il file stesso.

x

Sta per *esecuzione (execute)* e ha un valore ottale di 1. Ciò indica il permesso di entrare in una

directory, ma non di elencare i suoi file (per questo è necessaria la `r`).

L'ultima parte sulle directory può sembrare un po' confusa. Immaginiamo, per esempio, di avere una directory chiamata `Another_Directory`, con i seguenti permessi:

```
$ ls -ld Another_Directory/
d--x--x--x 2 carol carol 4,0K Dec 20 18:46 Another_Directory
```

Immagina anche che all'interno di questa directory tu abbia uno script di shell chiamato `hello.sh`:

```
-rwxr-xr-x 1 carol carol 33 Dec 20 18:46 hello.sh
```

Se sei l'utente `carol` e cerchi di elencare il contenuto di `Another_Directory`, riceverai un messaggio di errore, poiché il tuo utente non ha i permessi di lettura per quella directory:

```
$ ls -l Another_Directory/
ls: cannot open directory 'Another_Directory/': Permission denied
```

Tuttavia, l'utente `carol` *dispone* dei permessi di esecuzione, il che significa che può entrare nella directory. Pertanto, l'utente `carol` può accedere ai file all'interno della directory, purché abbia i permessi corretti *per il rispettivo file*. Supponiamo che l'utente abbia i permessi completi (`rwx`) per lo script `hello.sh`. Quindi *può* eseguire lo script, anche se *non può* leggere il contenuto della directory che lo contiene se conosce il nome completo del file:

```
$ sh Another_Directory/hello.sh
Hello LPI World!
```

Come abbiamo detto prima, i permessi sono specificati in sequenza: prima per il proprietario del file, poi per il gruppo proprietario e poi per tutti gli altri utenti. Ogni volta che qualcuno tenta di eseguire un'azione sul file, le autorizzazioni vengono verificate allo stesso modo.

Per prima cosa il sistema controlla se l'utente corrente abbia il possesso del file e, se questo è vero, applica solo il primo insieme di permessi. In caso contrario, controlla se l'utente corrente appartiene al gruppo proprietario del file. In tal caso, applica solo la seconda serie di autorizzazioni. In ogni altro caso, il sistema applicherà la terza serie di autorizzazioni.

Ciò significa che se l'utente corrente è il proprietario del file, solo le autorizzazioni del proprietario sono effettive, anche se il gruppo o altre autorizzazioni sono più permissive delle

autorizzazioni del proprietario.

Modificare i Permessi sui File

Il comando `chmod` è usato per modificare i permessi di un file, e richiede almeno due parametri: il primo descrive quali permessi cambiare; il secondo indica il file o la directory dove verrà effettuata la modifica. Tieni presente che solo il proprietario del file o l'amministratore di sistema (root) può modificare le autorizzazioni su un file.

I permessi per cambiare possono essere descritti in due modalità differenti, o “modi” (*modes*).

Il primo, chiamato *modo simbolico* (*symbolic mode*), offre un controllo *granulare*, consentendo di aggiungere o revocare un singolo permesso senza modificare gli altri sul set generale. L'altra modalità, chiamata *modo ottale* (*octal mode*), è più facile da ricordare e più veloce da usare se desideri impostare tutti i valori dei permessi contemporaneamente.

Entrambe le modalità porteranno allo stesso risultato finale. Quindi, per esempio, i comandi:

```
$ chmod ug+rw-x,o-rwx text.txt
```

e

```
$ chmod 660 text.txt
```

produrranno esattamente lo stesso output, un file con le autorizzazioni impostate:

```
-rw-rw---- 1 carol carol 765 Dec 20 21:25 text.txt
```

Vediamo ora come funziona ciascuna modalità.

Modalità Simbolica

Quando si descrivono quali permessi cambiare in *modalità simbolica* i primi caratteri indicano quali permessi si modificheranno: quelli per l'utente (u), per il gruppo (g), per gli altri (o) e/o per tutti (a).

Quindi devi dire al comando cosa fare: puoi concedere un'autorizzazione (+), revocare un'autorizzazione (-) o impostarla su un valore specifico (=).

Infine, si specifica su quale autorizzazione si desidera agire: leggere (r), scrivere (w) o eseguire (x).

Per esempio, immagina di avere un file chiamato `text.txt` con il seguente set di autorizzazioni:

```
$ ls -l text.txt
-rw-r--r-- 1 carol carol 765 Dec 20 21:25 text.txt
```

Se desideri concedere i permessi di scrittura ai membri del gruppo che possiede il file, dovresti usare il parametro `g+w`. È più facile se ci pensi in questo modo: “Per il gruppo (g), concedi (+) i permessi di scrittura (w)”. Quindi, il comando sarebbe:

```
$ chmod g+w text.txt
```

Controlliamo il risultato con `ls`:

```
$ ls -l text.txt
-rw-rw-r-- 1 carol carol 765 Dec 20 21:25 text.txt
```

Vuoi rimuovere i permessi di lettura per il proprietario dello stesso file? Pensalo come: “Per l’utente (u), revoca (-) i permessi di lettura (r)”. Quindi il parametro è `u-r`, in questo modo:

```
$ chmod u-r text.txt
$ ls -l text.txt
--w-rw-r-- 1 carol carol 765 Dec 20 21:25 text.txt
```

E se volessimo impostare i permessi esattamente come `rw-` per tutti? Pensalo come: “Per tutti (a), imposta esattamente (=) leggi (r), scrivi (w) e non eseguire (-)”. Così:

```
$ chmod a=rw- text.txt
$ ls -l text.txt
-rw-rw-rw- 1 carol carol 765 Dec 20 21:25 text.txt
```

Ovviamente è possibile modificare più permessi contemporaneamente. In questo caso, separali con una virgola (,):

```
$ chmod u+rw,g-x text.txt
$ ls -lh text.txt
-rwxrw-rw- 1 carol carol 765 Dec 20 21:25 text.txt
```

L’esempio sopra può essere letto come: “Per l’utente (u), concedi (+) i permessi di lettura, scrittura

ed esecuzione (rwx), per il gruppo (g), revoca (-) i permessi di esecuzione (x)”.

Quando viene eseguito su una directory, `chmod` modifica solo i permessi della directory. `chmod` ha anche una modalità ricorsiva, che è utile quando vuoi cambiare i permessi per “tutti i file all’interno di una directory e le sue sottodirectory”. Per usarlo, aggiungi il parametro `-R` dopo il nome del comando, prima dei permessi da modificare:

```
$ chmod -R u+rwx Another_Directory/
```

Questo comando può essere letto come: “Ricorsivamente (`-R`), per l’utente (`u`), concedi (+) i permessi di lettura, scrittura ed esecuzione (rwx)”.

WARNING

Fai attenzione e pensaci due volte prima di usare l’opzione `-R`, poiché è facile cambiare i permessi su file e directory che non vuoi cambiare, specialmente su directory con un gran numero di file e sottodirectory.

Modalità Ottale

Nella *modalità ottale* i permessi sono specificati in un modo diverso: come un valore a tre cifre su notazione ottale (un sistema numerico in base 8).

Ogni permesso ha un valore corrispondente, e sono specificati nel seguente ordine: prima viene *lettura* (r), che è 4, poi *scrittura* (w), che è 2 e l’ultimo è *esecuzione* (x), rappresentato da 1. Se non ci sono permessi, usa il valore zero (0). Quindi, un permesso di rwx sarebbe 7(4+2+1) e r-x sarebbe 5 (4+0+1).

La prima delle tre cifre sul set di autorizzazioni rappresenta i permessi per l’utente (u), la seconda per il gruppo (g) e la terza per gli altri (o). Se volessimo impostare i permessi per un file su `rw-rw----`, il valore ottale sarebbe 660:

```
$ chmod 660 text.txt
$ ls -l text.txt
-rw-rw---- 1 carol carol 765 Dec 20 21:25 text.txt
```

Oltre a questo, la sintassi nella *modalità ottale* è la stessa della *modalità simbolica*: il primo parametro rappresenta i permessi che desideri modificare, e il secondo parametro punta al file o alla directory in cui verrà effettuata la modifica.

TIP | Se un valore di autorizzazione è *dispari*, il file è sicuramente eseguibile!

Quale sintassi dovresti usare? La *modalità ottale* è consigliata se si desidera modificare i permessi

con un valore specifico, per esempio `640` (``rw-r-----``).

La *modalità simbolica* è più utile se si desidera cambiare solo un valore specifico, indipendentemente dalle autorizzazioni correnti per il file. Per esempio, puoi aggiungere i permessi di esecuzione per l'utente usando semplicemente `chmod u+x script.sh` senza considerare, o anche toccare, i permessi correnti per il gruppo e altri.

Modificare la Proprietà di un File

Il comando `chown` viene utilizzato per modificare la proprietà di un file o di una directory. La sintassi è abbastanza semplice:

```
chown USERNAME:GROUPNAME FILENAME
```

Per esempio, controlliamo un file chiamato `text.txt`:

```
$ ls -l text.txt
-rw-rw---- 1 carol carol 1881 Dec 10 15:57 text.txt
```

L'utente che possiede il file è `carol` e anche il gruppo proprietario è `carol`. Ora, cambieremo il gruppo che possiede il file in un altro gruppo, come `students`:

```
$ chown carol:students text.txt
$ ls -l text.txt
-rw-rw---- 1 carol students 1881 Dec 10 15:57 text.txt
```

Tieni presente che l'utente che possiede un file non ha bisogno di appartenere al gruppo che possiede un file. Nell'esempio sopra, l'utente `carol` non ha bisogno di essere un membro del gruppo `students`.

Il set di autorizzazioni utente o gruppo può essere omissso se non si desidera modificarlo. Quindi, per cambiare solo il gruppo che possiede un file dovresti usare `chown :students text.txt`. Per cambiare solo l'utente, il comando sarebbe `chown carol: text.txt` o semplicemente `chown carol text.txt`. In alternativa, potresti usare il comando `chgrp students text.txt`.

A meno che tu non sia l'amministratore di sistema (`root`), non puoi cambiare la proprietà di un file a un altro utente o gruppo a cui non appartieni. Se provi a farlo, riceverai il messaggio di errore `Operation not permitted`.

Interrogare i Gruppi

Prima di modificare la proprietà di un file, potrebbe essere utile sapere quali gruppi esistono nel sistema, quali utenti sono membri di un gruppo e a quali gruppi appartiene un utente.

Per vedere quali gruppi esistono sul tuo sistema, digita `getent group`. L'output sarà simile a questo (l'output è stato abbreviato):

```
$ getent group
root:x:0:
daemon:x:1:
bin:x:2:
sys:x:3:
adm:x:4:syslog,rigues
tty:x:5:rigues
disk:x:6:
lp:x:7:
mail:x:8:
news:x:9:
uucp:x:10:rigues
```

Se vuoi sapere a quali gruppi appartiene un utente, aggiungi il nome utente come parametro a `groups`:

```
$ groups carol
carol : carol students cdrom sudo dip plugdev lpadmin sambashare
```

Per fare il contrario (vedere quali utenti appartengono a un gruppo) usa `groupmems`. Il parametro `-g` specifica il gruppo e `-l` elencherà tutti i suoi membri:

```
# groupmems -g cdrom -l
carol
```

TIP

`groupmems` può essere eseguito solo come root, l'amministratore di sistema. Se non sei attualmente loggato come root, aggiungi `sudo` prima del comando.

Permessi Default

Proviamo un esperimento. Apri una finestra di terminale e crea un file vuoto con il seguente comando:

```
$ touch testfile
```

Ora, diamo un'occhiata alle autorizzazioni per questo file. Potrebbero essere diversi sul tuo sistema, ma supponiamo che abbiano il seguente aspetto:

```
$ ls -lh testfile
-rw-r--r-- 1 carol carol 0 jul 13 21:55 testfile
```

I permessi sono `rw-r--r--`: *lettura e scrittura* per l'utente e *lettura* per il gruppo e gli altri, o `644` in modalità ottale. Ora prova a creare una directory:

```
$ mkdir testdir
$ ls -lhd testdir
drwxr-xr-x 2 carol carol 4,0K jul 13 22:01 testdir
```

Ora i permessi sono `rw-r-xr-x`: *lettura, scrittura ed esecuzione* per l'utente, *lettura ed esecuzione* per il gruppo e altri, o `755` in modalità ottale.

Non importa dove ti trovi nel filesystem, ogni file o directory che crei avrà le stesse autorizzazioni. Ti sei mai chiesto da dove provengono?

Provengono dalla *user mask* o `umask`, che imposta i permessi predefiniti per ogni file creato. Puoi controllare i valori correnti con il comando `umask`:

```
$ umask
0022
```

Ma questo non sembra `rw-r--r--`, o anche `644`. Forse dovremmo provare con il parametro `-S`, per ottenere un output in modalità simbolica:

```
$ umask -S
u=rwx,g=rx,o=rx
```

Queste sono le stesse autorizzazioni ottenute dalla nostra directory di test in uno degli esempi precedenti. Ma perché quando abbiamo creato un file i permessi erano diversi?

Bene, non ha senso impostare le autorizzazioni di esecuzione globali per tutti su qualsiasi file per impostazione predefinita, giusto? Le directory richiedono autorizzazioni di esecuzione (altrimenti

non è possibile accederci), ma i file no, quindi non le ottengono. Da qui il `rw-r--r--`.

Oltre a mostrare i permessi predefiniti, `umask` può anche essere usato per cambiarli per la tua attuale sessione di shell. Per esempio, se usiamo il comando:

```
$ umask u=rwx,g=rwx,o=
```

Ogni nuova directory eredita i permessi `rw-rwxrwx---`, e ogni file `rw-rw----` (dato che non ottengono i permessi di esecuzione). Se ripeti gli esempi sopra per creare un `testfile` e `testdir` e controlli i permessi, dovresti ottenere:

```
$ ls -lhd test*
drwxrwx--- 2 carol carol 4,0K jul 13 22:25 testdir
-rw-rw---- 1 carol carol    0 jul 13 22:25 testfile
```

E se controlli `umask` senza il parametro `-S` (modalità simbolica), ottieni:

```
$ umask
0007
```

Il risultato non sembra familiare perché i valori utilizzati sono diversi. Ecco una tabella con ogni valore e il suo rispettivo significato:

Valore	Permesso per i File	Permesso per le Directory
0	<code>rw-</code>	<code>rwX</code>
1	<code>rw-</code>	<code>rw-</code>
2	<code>r--</code>	<code>r-X</code>
3	<code>r--</code>	<code>r--</code>
4	<code>-w-</code>	<code>-wX</code>
5	<code>-w-</code>	<code>-w-</code>
6	<code>---</code>	<code>--X</code>
7	<code>---</code>	<code>---</code>

Come puoi vedere, `007` corrisponde a `rw-rwx---`, esattamente come abbiamo richiesto. Lo zero iniziale può essere ignorato.

Permessi Speciali

Oltre ai permessi di lettura, scrittura ed esecuzione per utente, gruppo e altri, ogni file può avere altri tre *permessi speciali* che possono alterare il modo in cui funziona una directory o come viene eseguito un programma. Possono essere specificati in modalità simbolica o ottale e sono i seguenti:

Sticky Bit

Lo *sticky bit*, chiamato anche *indicatore di cancellazione ristretta*, ha il valore ottale 1 e in modalità simbolica è rappresentato da una **t** all'interno dei permessi dell'altro. Ciò si applica solo alle directory e non ha alcun effetto sui file normali. Su Linux impedisce agli utenti di rimuovere o rinominare un file in una directory a meno che non siano proprietari di quel file o directory.

Le directory con lo sticky bit impostato mostrano una **t** che sostituisce la **x** sui permessi per gli altri sull'output di `ls -l`:

```
$ ls -ld Sample_Directory/
drwxr-xr-t 2 carol carol 4096 Dec 20 18:46 Sample_Directory/
```

In modalità ottale i permessi speciali vengono specificati usando una notazione a 4 cifre, con la prima cifra che rappresenta il permesso speciale su cui agire. Per esempio, per impostare lo sticky bit (valore 1) per la directory `Another_Directory` in modalità ottale, con i permessi 755, il comando sarebbe:

```
$ chmod 1755 Another_Directory
$ ls -ld Another_Directory
drwxr-xr-t 2 carol carol 4,0K Dec 20 18:46 Another_Directory
```

Set GID

Set GID, noto anche come SGID o bit di *Set Group ID*, ha il valore ottale 2 e in modalità simbolica è rappresentato da una **s** sui permessi di *gruppo*. Questo può essere applicato a file eseguibili o directory. Sui file, eseguirà il processo con i privilegi del gruppo che possiede il file. Quando applicato alle directory, farà in modo che ogni file o directory creato sotto di esso erediti il gruppo dalla directory principale.

I file e le directory con bit SGID mostrano una **s** che sostituisce la **x** sui permessi per il *gruppo* sull'output di `ls -l`:

```
$ ls -l test.sh
-rwxr-sr-x 1 carol root 33 Dec 11 10:36 test.sh
```

Per aggiungere autorizzazioni SGID a un file in modalità simbolica, il comando sarebbe:

```
$ chmod g+s test.sh
$ ls -l test.sh
-rwxr-sr-x 1 carol root 33 Dec 11 10:36 test.sh
```

Il seguente esempio ti aiuterà a comprendere meglio gli effetti di SGID su una directory. Supponiamo di avere una directory chiamata `Sample_Directory`, di proprietà dell'utente `carol` e del gruppo `users`, con la seguente struttura dei permessi

```
$ ls -ldh Sample_Directory/
drwxr-xr-x 2 carol users 4,0K Jan 18 17:06 Sample_Directory/
```

Ora, passiamo a questa directory e, usando il comando `touch`, creiamo un file vuoto al suo interno. Il risultato sarebbe:

```
$ cd Sample_Directory/
$ touch newfile
$ ls -lh newfile
-rw-r--r-- 1 carol carol 0 Jan 18 17:11 newfile
```

Come possiamo vedere il file è di proprietà dell'utente `carol` e del gruppo `carol`. Tuttavia, se la directory avesse il set di autorizzazioni SGID, il risultato sarebbe diverso. Per prima cosa, aggiungiamo il bit SGID a `Sample_Directory` e controlliamo i risultati:

```
$ sudo chmod g+s Sample_Directory/
$ ls -ldh Sample_Directory/
drwxr-sr-x 2 carol users 4,0K Jan 18 17:17 Sample_Directory/
```

La `s` sui permessi del gruppo indica che il bit SGID è impostato. Ora, passeremo a questa directory e, di nuovo, creeremo un file vuoto con il comando `touch`:

```
$ cd Sample_Directory/
$ touch emptyfile
$ ls -lh emptyfile
```



```
-rw-r--r-- 1 carol users 0 Jan 18 17:20 emptyfile
```

Il gruppo che possiede il file è `users`. Questo perché il bit SGID ha fatto in modo che il file ereditasse il proprietario del gruppo della sua directory padre, che è `users`.

Set UID

SUID, noto anche come Set User ID, ha il valore ottale `4` ed è rappresentato da una `s` sui permessi *utente* in modalità simbolica. Si applica solo ai file e non ha alcun effetto sulle directory. Il suo comportamento è simile al bit SGID, ma il processo verrà eseguito con i privilegi dell' *utente* che possiede il file. I file con il bit SUID mostrano una `s` che sostituisce la `x` sui permessi per l'utente sull'output di `ls -l`:

```
$ ls -ld test.sh
-rwsr-xr-x 1 carol carol 33 Dec 11 10:36 test.sh
```

È possibile combinare più autorizzazioni speciali su un parametro. Quindi, per impostare SGID (valore `2`) e SUID (valore `4`) in modalità ottale per lo script `test.sh` con i permessi `755`, dovresti digitare:

```
$ chmod 6755 test.sh
```

E il risultato sarà:

```
$ ls -lh test.sh
-rwsr-sr-x 1 carol carol 66 Jan 18 17:29 test.sh
```

TIP

Se il tuo terminale supporta il colore, e oggi giorno la maggior parte di loro lo fa, puoi vedere rapidamente se questi permessi speciali sono impostati dando uno sguardo all'output di `ls -l`. Per lo sticky bit, il nome della directory potrebbe essere visualizzato in un carattere nero con sfondo blu. Lo stesso vale per i file con i bit SGID (sfondo giallo) e SUID (sfondo rosso). I colori possono essere diversi a seconda della distribuzione Linux e delle impostazioni del terminale che utilizzi.

Esercizi Guidati

1. Crea una directory chiamata `emptydir` usando il comando `mkdir emptydir`. Ora, usando `ls`, elenca i permessi per la directory `emptydir`.

2. Crea un file vuoto chiamato `emptyfile` con il comando `touch emptyfile`. Ora, usando `chmod` in modalità simbolica, aggiungi i permessi di esecuzione per il proprietario del file `emptyfile`, e rimuovi i permessi di scrittura ed esecuzione per tutti gli altri. Fallo usando un solo comando `chmod`.

3. Quali sarebbero i permessi predefiniti per un file se il valore `umask` fosse impostato su `027`?

4. Supponiamo che un file chiamato `test.sh` sia uno script di shell con i seguenti permessi e proprietà:

```
-rwxr-sr-x 1 carol root    33 Dec 11 10:36 test.sh
```

- Quali sono le autorizzazioni per il proprietario del file?
-
- Usando la notazione ottale, quale sarebbe la sintassi di `chmod` per “rimuovere” il permesso speciale concesso a questo file?
-

5. Considera il file:

```
$ ls -l /dev/sdb1  
brw-rw---- 1 root disk 8, 17 Dec 21 18:51 /dev/sdb1
```

Quale tipo di file è `sdb1`? Chi può scriverci?

6. Considera i seguenti quattro file:

```
drwxr-xr-t 2 carol carol 4,0K Dec 20 18:46 Another_Directory
```

```
----r--r-- 1 carol carol    0 Dec 11 10:55 foo.bar
-rw-rw-r-- 1 carol carol 1,2G Dec 20 18:22 HugeFile.zip
drwxr-sr-x 2 carol users   4,0K Jan 18 17:26 Sample_Directory
```

Annotare le autorizzazioni corrispondenti per ogni file e directory utilizzando la modalità ottale utilizzando la notazione a 4 cifre.

Another_Directory	
foo.bar	
HugeFile.zip	
Sample_Directory	

Esercizi Esplorativi

1. Prova questo su un terminale: crea un file vuoto chiamato `emptyfile` con il comando `touch emptyfile`. Ora “azzerà” i permessi per il file con `chmod 000 emptyfile`. Cosa succederà se modifichi i permessi per `emptyfile` passando solo *un* valore per `chmod` in modalità ottale, come per esempio `chmod 4 emptyfile`? E se ne usi due, come in `chmod 44 emptyfile`? Cosa possiamo imparare sul modo in cui `chmod` legge il valore numerico?

2. Considera i permessi per la directory temporanea su un sistema Linux, `/tmp`:

```
$ ls -l /tmp
drwxrwxrwt 19 root root 16K Dec 21 18:58 tmp
```

Utente, gruppo e altri dispongono di autorizzazioni complete. Ma un utente normale può eliminare *qualsiasi* file all'interno di questa directory? Perché?

3. Un file chiamato `test.sh` ha i seguenti permessi: `-rwsr-xr-x`, il che significa che il bit SUID è impostato. Ora, esegui i seguenti comandi:

```
$ chmod u-x test.sh
$ ls -l test.sh
-rwsr-xr-x 1 carol carol 33 Dec 11 10:36 test.sh
```

Che cosa abbiamo fatto? Che cosa significa la `S` maiuscola?

4. Come creereste una directory chiamata `Box` in cui tutti i file sono automaticamente di proprietà del gruppo `users` e possono essere eliminati solo dall'utente che li ha creati?

Sommario

In questa lezione hai imparato come usare `ls` per ottenere (e comprendere) informazioni sui permessi dei file, come controllare o cambiare chi può creare, eliminare o modificare un file con `chmod`, sia in modalità *ottale* sia *simbolica*, come per cambiare la proprietà dei file con `chown` e `chgrp` e come interrogare e modificare la maschera dei permessi predefinita per file e directory con `umask`.

In questa lezione sono stati discussi i seguenti comandi:

`ls`

Elenca i file, includendo facoltativamente dettagli come le autorizzazioni.

`chmod`

Modifica le autorizzazioni di un file o di una directory.

`chown`

Modifica l'utente e/o il gruppo proprietario di un file o di una directory.

`chgrp`

Modifica il gruppo proprietario di un file o di una directory.

`umask`

Interroga o imposta la maschera delle autorizzazioni predefinita per file e directory.

Risposte agli Esercizi Guidati

1. Crea una directory chiamata `emptydir` usando il comando `mkdir emptydir`. Ora, usando `ls`, elenca i permessi per la directory `emptydir`.

Aggiungi il parametro `-d` a `ls` per vedere gli attributi di una directory, invece di elencarne il contenuto. Quindi, la risposta è:

```
ls -l -d emptydir
```

Punti bonus se unisci i due parametri in uno, come in `ls -ld emptydir`.

2. Crea un file vuoto chiamato `emptyfile` con il comando `touch emptyfile`. Ora, usando `chmod` in modalità simbolica, aggiungi i permessi di esecuzione per il proprietario del file `emptyfile`, e rimuovi i permessi di scrittura ed esecuzione per tutti gli altri. Fallo usando un solo comando `chmod`.

Pensaci in questo modo:

- “Per l’utente che possiede il file (u) aggiungi il permesso (+) di esecuzione (x)”, quindi `u+x`.
- “Per il gruppo (g) e altri utenti (o), rimuovi i permessi (-) scrittura (w) ed esecuzione (x)”, quindi `go-wx`.

Per combinare questi due set di autorizzazioni, aggiungiamo una virgola tra di loro. Quindi il risultato finale è:

```
chmod u+x,go-wx emptyfile
```

3. Quali sarebbero i permessi predefiniti per un file se il valore `umask` fosse impostato su `020`?

I permessi sarebbero `rw-r-----`

4. Supponiamo che un file chiamato `test.sh` sia uno script di shell con i seguenti permessi e proprietà:

```
-rwxr-sr-x 1 carol root    33 Dec 11 10:36 test.sh
```

- Quali sono le autorizzazioni per il proprietario del file?

I permessi per il proprietario (dal secondo al quarto carattere nell’output di `ls -l`) sono

`rwX`, quindi la risposta è: “leggere, scrivere ed eseguire il file”.

- Usando la notazione ottale, quale dovrebbe essere la sintassi di `chmod` per “rimuovere” il permesso speciale concesso a questo file?

Possiamo “rimuovere” i permessi speciali passando una quarta cifra, `0`, a “`chmod`”. I permessi correnti sono `755`, quindi il comando dovrebbe essere `chmod 0755`.

5. Considera il file:

```
$ ls -l /dev/sdb1
brw-rw---- 1 root disk 8, 17 Dec 21 18:51 /dev/sdb1
```

Quale tipo di file è `sdb1`? Chi può scriverci?

Il primo carattere sull’output di `ls -l` mostra il tipo di file. `b` è un *device a blocchi*, solitamente un disco (interno o esterno), connesso alla macchina. Il proprietario (`root`) e qualsiasi utente del gruppo `disk` possono scrivere su di esso.

6. Considera i seguenti quattro file:

```
drwxr-xr-t 2 carol carol 4,0K Dec 20 18:46 Another_Directory
----r--r-- 1 carol carol    0 Dec 11 10:55 foo.bar
-rw-rw-r-- 1 carol carol 1,2G Dec 20 18:22 HugeFile.zip
drwxr-sr-x 2 carol users 4,0K Jan 18 17:26 Sample_Directory
```

Annotare le autorizzazioni corrispondenti per ogni file e directory utilizzando la modalità ottale utilizzando la notazione a 4 cifre.

I permessi corrispondenti, in modalità ottale, sono i seguenti:

Another_Directory	1755. 1 per lo sticky bit, 755 per i permessi regolari (<code>rwX</code> per l’utente, <code>r-x</code> per gruppo e altri).
foo.bar	0044. Nessun permesso speciale (quindi la prima cifra è <code>0</code>), nessun permesso per l’utente (<code>---</code>) e sola lettura (<code>r--r--</code>) per il gruppo e altri.

HugeFile.zip	0664. Nessun permesso speciale, quindi la prima cifra è 0. 6 (rw-) per l'utente e il gruppo, 4 (r - -) per gli altri.
Sample_Directory	2755. 2 per il bit SGID, 7 (rwx) per l'utente, 5 (r - x) per il gruppo e altri.

Risposte agli Esercizi Esplorativi

1. Prova questo su un terminale: crea un file vuoto chiamato `emptyfile` con il comando `touch emptyfile`. Ora “azzerà” i permessi per il file con `chmod 000 emptyfile`. Che cosa succederà se modifichi i permessi per `emptyfile` passando solo *un* valore per `chmod` in modalità ottale, come per esempio `chmod 4 emptyfile`? E se ne usi due, come in `chmod 44 emptyfile`? Cosa possiamo imparare sul modo in cui `chmod` legge il valore numerico?

Ricorda che abbiamo “azzerato” i permessi per `emptyfile`. Quindi, il suo stato iniziale sarebbe:

```
----- 1 carol carol  0 Dec 11 10:55 emptyfile
```

Ora, proviamo il primo comando, `chmod 4 emptyfile`:

```
$ chmod 4 emptyfile
$ ls -l emptyfile
-----r-- 1 carol carol 0 Dec 11 10:55 emptyfile
```

Si vede? Le autorizzazioni per gli *altri* sono state modificate. E se provassimo due cifre, come in `chmod 44 emptyfile`?

```
$ chmod 44 emptyfile
$ ls -l emptyfile
----r--r-- 1 carol carol 0 Dec 11 10:55 emptyfile
```

Ora le autorizzazioni per il *gruppo* e gli *altri* sono state interessate. Possiamo concludere che in modalità ottale `chmod` legge il valore “all’indietro”, partendo dalla cifra meno significativa (*others*) a quella più significativa (*user*). Se indichi una cifra, modifichi i permessi per gli *altri*. Con due cifre modifichi *group* e *others*, e con tre modifichi *user*, *group* e *others* e con quattro cifre modifichi *user*, *group*, *others* e le autorizzazioni speciali.

2. Considera i permessi per la directory temporanea su un sistema Linux, `/ tmp`:

```
$ ls -l /tmp
drwxrwxrwt 19 root root 16K Dec 21 18:58 tmp
```

Utente, gruppo e altri dispongono di autorizzazioni complete. Ma un utente normale può eliminare *qualsiasi* file all’interno di questa directory? Perché?

`/tmp` è ciò che chiamiamo una directory *scrivibile dal mondo*, il che significa che qualsiasi utente può scriverci. Ma non vogliamo che un utente giochi con file creati da altri, quindi lo *sticky bit* è impostato (come indicato dalla `t` sui permessi per *others*). Ciò significa che un utente può eliminare i file in `/tmp`, ma solo quelli creati da lui stesso.

- Un file chiamato `test.sh` ha i seguenti permessi: `-rwsr-xr-x`, il che significa che il bit SUID è impostato. Ora, esegui i seguenti comandi:

```
$ chmod u-x test.sh
$ ls -l test.sh
-rwsr-xr-x 1 carol carol 33 Dec 11 10:36 test.sh
```

Che cosa abbiamo fatto? Che cosa significa la `S` maiuscola?

Abbiamo rimosso le autorizzazioni di esecuzione per l'utente proprietario del file. La `s` (o `t`) prende il posto della `x` sull'output di `ls -l`, quindi il sistema ha bisogno di un modo per mostrare se l'utente ha i permessi di esecuzione o meno. Lo fa cambiando il case del carattere speciale.

Una `s` minuscola sul primo gruppo di permessi significa che l'utente che possiede il file ha i permessi di esecuzione e che il bit SUID è impostato. Una `S` maiuscola indica che l'utente che possiede il file non ha (-) i permessi di esecuzione e che il bit SUID è impostato.

Lo stesso si può dire per SGID, una `s` minuscola sul secondo gruppo di permessi indica che il gruppo che possiede il file ha i permessi di esecuzione e che il bit SGID è impostato. Una `S` maiuscola indica invece che il gruppo che possiede il file non ha (-) i permessi di esecuzione e che il bit SGID è impostato.

Questo vale anche per lo sticky bit, rappresentato dalla `t` nel terzo gruppo di permessi. La `t` minuscola significa che è impostato lo sticky bit e che altri hanno i permessi di esecuzione. La `T` maiuscola significa che lo sticky bit è impostato e che gli altri non hanno i permessi di esecuzione.

- Come creereste una directory chiamata `Box` in cui tutti i file sono automaticamente di proprietà del gruppo `users` e possono essere eliminati solo dall'utente che li ha creati?

Questo è un processo in più fasi. Il primo passo è creare la directory:

```
$ mkdir Box
```

Vogliamo che ogni file creato all'interno di questa directory venga automaticamente assegnato

al gruppo `users`. Possiamo farlo impostando questo gruppo come proprietario della directory e quindi impostando il bit SGID su di esso. Dobbiamo anche assicurarci che qualsiasi membro del gruppo possa scrivere in quella directory.

Dato che non ci interessa sapere quali sono gli altri permessi e vogliamo “attivare” solo i bit speciali, ha senso usare la modalità simbolica:

```
$ chown :users Box/
$ chmod g+ws Box/
```

Nota che se il tuo utente attuale non appartiene al gruppo `users`, dovrai usare il comando `sudo` prima dei comandi sopra per eseguire la modifica come root.

Ora per l'ultima parte, assicurandoti che solo l'utente che ha creato un file sia autorizzato a eliminarlo. Questo viene eseguito impostando lo sticky bit (rappresentato da una `t`) sulla directory. Ricorda che è impostato sui permessi per gli altri (`o`).

```
$ chmod o+t Box/
```

I permessi sulla directory `Box` dovrebbero essere i seguenti:

```
drwxrwsr-t 2 carol users 4,0K Jan 18 19:09 Box
```

Ovviamente puoi specificare SGID e lo sticky bit usando un solo comando `chmod`:

```
$ chmod g+ws,o+t Box/
```

Punti bonus se ci hai pensato.



104.6 Creare e modificare collegamenti hard e soft

Obiettivi LPI di riferimento

LPIC-1 version 5.0, Exam 101, Objective 104.6

Peso

2

Aree di Conoscenza Chiave

- Creare collegamenti.
- Identificare collegamenti hard e/o soft.
- Copia e collegamento di file.
- Utilizzare i collegamenti per supportare le attività di amministrazione del sistema.

Di seguito è riportato un elenco parziale dei file, dei termini e dei comandi utilizzati

- `ln`
- `ls`



104.6 Lezione 1

Certificazione:	LPIC-1
Versione:	5.0
Argomento:	104 Dispositivi, Il Filesystem Linux, Filesystem Hierarchy Standard
Obiettivo:	104.6 Creare e modificare gli hard e soft links
Lezione:	1 di 1

Introduzione

Su Linux alcuni file ricevono un trattamento speciale o per il posto in cui sono memorizzati, come i file temporanei, o per il modo in cui interagiscono con il filesystem, come i collegamenti. In questa lezione imparerai cosa sono i collegamenti e come gestirli.

Comprendere i Link

Come già accennato, su Linux tutto viene trattato come un file. Ma esiste un tipo speciale di file, chiamato *link*, e ce ne sono di due tipi in un sistema Linux:

Link simbolici: chiamati anche *soft link*, puntano al percorso di un altro file. Se elimini il file a cui punta il collegamento (chiamato *target*) il collegamento esisterà ancora, ma “smetterà di funzionare”, poiché ora non punterà più a niente.

Hard Link

Pensa a un collegamento fisico (hard) come a un secondo nome per il file originale. *Non* sono duplicati, ma sono invece una voce aggiuntiva nel filesystem che punta alla stessa posizione

(*inode*) sul disco.

TIP

Un *inode* è una struttura dati che memorizza gli attributi per un oggetto (come un file o una directory) su un filesystem. Tra questi attributi ci sono i permessi, la proprietà e su quali blocchi del disco sono archiviati i dati per l'oggetto. Pensala come una voce su un indice, da cui il nome, che deriva da “index node”.

Lavorare con gli Hard Link

Creare Hard Link

Il comando per creare un hard link su Linux è `ln`. La sintassi di base è:

```
$ ln TARGET LINK_NAME
```

Il `TARGET` deve già esistere (questo è il file a cui punterà il collegamento), e se il target non si trova nella directory corrente, o se si desidera creare il collegamento altrove, è necessario specificare il percorso completo a esso. Per esempio, il comando:

```
$ ln target.txt /home/carol/Documents/hardlink
```

creerà un file chiamato `hardlink` nella directory `/home/carol/Documents/`, collegato al file `target.txt` nella directory corrente.

Se tralasci l'ultimo parametro (`LINK_NAME`), verrà creato un collegamento con lo stesso nome del target nella directory corrente.

Gestire Hard Link

Gli hard link sono voci nel filesystem che hanno nomi diversi ma puntano agli stessi dati sul disco. Tutti questi nomi sono uguali e possono essere utilizzati per fare riferimento a un file. Se modifichi il contenuto di uno dei nomi, il contenuto di tutti gli altri nomi che puntano a quel file cambia poiché tutti questi nomi puntano agli stessi dati. Se elimini uno dei nomi (puntamento), gli altri continueranno a funzionare.

Questo accade perché quando si “cancella” un file i dati non vengono effettivamente cancellati dal disco. Il sistema cancella semplicemente la voce sulla tabella del filesystem che punta all'inode corrispondente ai dati sul disco. Ma se hai una seconda voce che punta allo stesso inode, puoi comunque accedere ai dati. Pensa a due strade che convergono allo stesso punto. Anche se blocchi o reindirizzi una delle strade, puoi comunque raggiungere la destinazione utilizzando l'altra.

Puoi verificarlo usando il parametro `-li` di `ls`. Considera i seguenti contenuti di una directory:

```
$ ls -li
total 224
3806696 -r--r--r-- 2 carol carol 111702 Jun  7 10:13 hardlink
3806696 -r--r--r-- 2 carol carol 111702 Jun  7 10:13 target.txt
```

Il numero prima dei permessi è il numero di inode. Vedi che sia il file `hardlink` che il file `target.txt` hanno lo stesso numero (3806696)? Questo perché uno è un hard link dell'altro.

Ma qual è l'originale e qual è il collegamento? Non si può davvero dire, poiché per tutti gli scopi pratici sono la stessa cosa.

Nota che ogni hard link che punta a un file aumenta il *contatore dei collegamenti (link count)* del file. Questo è il numero subito dopo i permessi sull'output di `ls -li`. Per impostazione predefinita, ogni file ha un conteggio dei collegamenti di 1 (le directory hanno un conteggio di 2) e ogni collegamento fisico a esso aumenta il conteggio di uno. Questo è il motivo per il conteggio dei collegamenti pari a 2 sui file nell'elenco sopra.

A differenza dei collegamenti simbolici, è possibile creare solo collegamenti fisici ai file e sia il collegamento che la destinazione devono risiedere nello stesso file system.

Spostare e Rimuovere Hard Link

Poiché gli hard link sono trattati come file normali, possono essere cancellati con `rm` e rinominati o spostati nel filesystem con `mv`. E poiché un hard link punta allo stesso inode, può essere spostato liberamente, senza timore di “spezzare” il collegamento.

Link Simbolici

Creare Link Simbolici

Anche il comando usato per creare un collegamento simbolico è `ln`, ma con l'aggiunta dell'opzione `-s`. Così:

```
$ ln -s target.txt /home/carol/Documents/softlink
```

Questo creerà un file chiamato `softlink` nella directory `/home/carol/Documents/`, che punta al file `target.txt` nella directory corrente.

Come con gli hard link, è possibile omettere il nome del collegamento per creare un collegamento

con lo stesso nome della destinazione nella directory corrente.

Gestire i Link Simbolici

I link simbolici puntano a un altro percorso nel filesystem. È possibile creare collegamenti simbolici (o soft) a file e directory, anche su partizioni diverse. È abbastanza facile individuare un collegamento simbolico con l'output del comando `ls`:

```
$ ls -lh
total 112K
-rw-r--r-- 1 carol carol 110K Jun  7 10:13 target.txt
lrwxrwxrwx 1 carol carol  12 Jun  7 10:14 softlink -> target.txt
```

Nell'esempio sopra, il primo carattere sui permessi per il file `softlink` è `l`, che indica un collegamento simbolico. Inoltre, subito dopo il nome del file si vede il nome della destinazione a cui punta il collegamento, il file `target.txt`.

Si noti che negli elenchi di file e directory, i collegamenti simbolici stessi mostrano sempre i permessi `rwX` per l'utente, il gruppo e altri, ma in pratica i permessi di accesso per loro sono gli stessi di quelli per la destinazione.

Spostare ed Eliminare Link Simbolici

Come gli hard link, i link simbolici possono essere rimossi usando `rm` e spostati o rinominati usando `mv`. Tuttavia, è necessario prestare particolare attenzione durante la creazione, per evitare di “interrompere” il collegamento se viene spostato dalla sua posizione originale.

Quando si creano collegamenti simbolici, è necessario essere consapevoli del fatto che, a meno che un percorso non sia completamente specificato, la posizione della destinazione venga interpretata come *relativa* alla posizione del collegamento. Ciò potrebbe creare problemi se il collegamento o il file a cui punta venisse spostato.

Questo è più facile da capire con un esempio. Supponiamo di avere un file chiamato `original.txt` nella directory corrente e di voler creare un collegamento simbolico ad esso chiamato `softlink`. Potresti usare:

```
$ ln -s original.txt softlink
```

E, a quanto pare, tutto sarebbe andato bene. Controlliamo con `ls`:


```
$ ls -lh
total 112K
-r--r--r-- 1 carol carol 110K Jun  7 10:13 original.txt
lrwxrwxrwx 1 carol carol  12 Jun  7 19:23 softlink -> original.txt
```

Guarda come è costruito il collegamento: `softlink` punta a (`->`) `original.txt`. Tuttavia, vediamo cosa succede se sposti il collegamento nella directory precedente e provi a visualizzarne il contenuto utilizzando il comando `less`:

```
$ mv softlink ../
$ less ../softlink
../softlink: No such file or directory
```

Poiché il percorso di `original.txt` non è stato specificato, il sistema presume che si trovi nella stessa directory del collegamento. Quando questo non è più vero, il collegamento smette di funzionare.

Il modo per evitarlo è specificare sempre il percorso completo della destinazione durante la creazione del collegamento:

```
$ ln -s /home/carol/Documents/original.txt softlink
```

In questo modo, indipendentemente da dove ci si sposta, il collegamento continuerà a funzionare, poiché punta alla posizione assoluta della destinazione. Controlla con `ls`:

```
$ ls -lh
total 112K
lrwxrwxrwx 1 carol carol  40 Jun  7 19:34 softlink -> /home/carol/Documents/original.txt
```

Esercizi Guidati

1. Qual è il parametro per `chmod` in modalità *simbolica* per abilitare lo sticky bit su una directory?

2. Immagina che ci sia un file chiamato `document.txt` nella directory `/home/carol/Documents`. Qual è il comando per creare un link simbolico chiamato `text.txt` nella directory corrente?

3. Spiega la differenza tra un hard link a un file e una copia dello stesso file.

Esercizi Esplorativi

1. Immagina di creare, all'interno di una directory, un file chiamato `recipes.txt`. All'interno di questa directory, creerai anche un hard link a questo file, chiamato `receitas.txt`, e un link simbolico (o *soft*) a questo chiamato `rezepte.txt`.

```
$ touch recipes.txt
$ ln recipes.txt receitas.txt
$ ln -s recipes.txt rezepte.txt
```

Il contenuto della directory dovrebbe essere così:

```
$ ls -lhi
total 160K
5388833 -rw-r--r-- 4 carol carol 77K jun 17 17:25 receitas.txt
5388833 -rw-r--r-- 4 carol carol  0K jun 17 17:25 recipes.txt
5388837 lrwxrwxrwx 1 carol carol  12 jun 17 17:25 rezepte.txt -> receitas.txt
```

Ricorda che, come hard link, `receitas.txt` punta allo stesso inode assegnato a `recipes.txt`. Che cosa succederebbe al collegamento software `rezepte.txt` se il file `receitas.txt` venisse cancellato? Perché?

2. Immagina di avere una chiavetta USB collegata al tuo sistema e montata in `/media/youruser/FlashA`. Si desidera creare un collegamento chiamato `schematics.pdf` nella directory home, che punta al file `esquema.pdf` nella root dell'unità flash. Quindi, digiti il comando:

```
$ ln /media/youruser/FlashA/esquema.pdf ~/schematics.pdf
```

Che cosa succede? Perché

3. Considera il seguente output di `ls -lah`:

```
$ ls -lah
total 3,1M
drwxr-xr-x 2 carol carol 4,0K jun 17 17:27 .
drwxr-xr-x 5 carol carol 4,0K jun 17 17:29 ..
-rw-rw-r-- 1 carol carol 2,8M jun 17 15:45 compressed.zip
-rw-r--r-- 4 carol carol  77K jun 17 17:25 document.txt
-rw-rw-r-- 1 carol carol 216K jun 17 17:25 image.png
```

```
-rw-r--r-- 4 carol carol 77K jun 17 17:25 text.txt
```

- Quanti link puntano al file `document.txt`?
- Sono collegamenti hard o soft?
- Quale parametro dovresti passare a `ls` per vedere quale inode occupa ogni file?

4. Immagina di avere nella tua directory `~/Documents` un file chiamato `clients.txt` contenente alcuni nomi di clienti e una directory chiamata `somedir`. All'interno di questa c'è un file *differente* denominato *anch'esso* `clients.txt` con nomi differenti. Per replicare questa struttura, utilizzare i seguenti comandi.

```
$ cd ~/Documents
$ echo "John, Michael, Bob" > clients.txt
$ mkdir somedir
$ echo "Bill, Luke, Karl" > somedir/clients.txt
```

Quindi crea un collegamento all'interno di `somedir` denominato `partners.txt` che punta a questo file, con i comandi:

```
$ cd somedir/
$ ln -s clients.txt partners.txt
```

Quindi, la struttura della directory sarà:

```
Documents
|-- clients.txt
`-- somedir
    |-- clients.txt
    `-- partners.txt -> clients.txt
```

Ora, sposta `partners.txt` da `somedir` a `~/Documents`, ed elencane il contenuto.

```
$ cd ~/Documents/
$ mv somedir/partners.txt .
```

```
$ less partners.txt
```

Il collegamento funzionerà ancora? In tal caso, quale file avrà il suo contenuto elencato? Perché?

5. Considera i seguenti file:

```
-rw-r--r-- 1 carol carol 19 Jun 24 11:12 clients.txt  
lrwxrwxrwx 1 carol carol 11 Jun 24 11:13 partners.txt -> clients.txt
```

Quali sono i permessi di accesso per `partners.txt`? Perché?

Sommario

In questa lezione abbiamo imparato:

- Cosa sono i collegamenti.
- La differenza tra i collegamenti (o link) *simbolici* (o *soft*) e *hard*.
- Come creare collegamenti.
- Come spostare, rinominare o rimuovere questi collegamenti.

In questa lezione sono stati discussi i seguenti comandi:

- `ln`: Il comando “link”. Di per sé, questo comando crea un hard link. Con l’opzione `-s` è possibile creare un collegamento *simbolico* o *soft*. Ricorda che i collegamenti hard possono risiedere solo sulla stessa partizione e file system, mentre i collegamenti simbolici possono attraversare partizioni e file system (anche l’archiviazione di rete).
- Il parametro `-i` di `ls`, consente di visualizzare il numero di inode per un file.

Risposte agli Esercizi Guidati

1. Qual è il parametro per `chmod` in modalità *simbolica* per abilitare lo sticky bit su una directory?

Il simbolo per lo sticky bit in modalità simbolica è `t`. Dato che vogliamo abilitare (aggiungere) questa autorizzazione alla directory, il parametro dovrebbe essere `+t`.

2. Immagina che ci sia un file chiamato `document.txt` nella directory `/home/carol/Documents`. Qual è il comando per creare un link simbolico chiamato `text.txt` nella directory corrente?

`ln -s` è il comando per creare un collegamento simbolico. Poiché è necessario specificare il percorso completo del file a cui ci si collega, il comando è:

```
$ ln -s /home/carol/Documents/document.txt text.txt
```

3. Spiega la differenza tra un hard link a un file e una copia dello stesso file

Un collegamento fisico è solo un altro nome per un file. Anche se sembra un duplicato del file originale, a tutti gli effetti sia il collegamento sia l'originale sono uguali, poiché puntano agli stessi dati sul disco. Le modifiche apportate ai contenuti del collegamento si rifletteranno sull'originale e viceversa. Una copia è un'entità completamente indipendente, che occupa un posto diverso sul disco. Le modifiche alla copia non si rifletteranno sull'originale e viceversa.

Risposte agli Esercizi Esplorativi

1. Immagina di creare, all'interno di una directory, un file chiamato `recipes.txt`. All'interno di questa directory, crea anche un hard link a questo file, chiamato `receitas.txt`, e un link simbolico (o *soft*) a questo chiamato `rezepte.txt`.

```
$ touch recipes.txt
$ ln recipes.txt receitas.txt
$ ln -s receitas.txt rezepte.txt
```

Il contenuto della directory dovrebbe essere così:

```
$ ls -lhi
total 160K
5388833 -rw-r--r-- 4 carol carol 77K jun 17 17:25 receitas.txt
5388833 -rw-r--r-- 4 carol carol  0K jun 17 17:25 recipes.txt
5388837 lrwxrwxrwx 1 carol carol  12 jun 17 17:25 rezepte.txt -> receitas.txt
```

Ricorda che, come hard link, `receitas.txt` punta allo stesso inode assegnato a `recipes.txt`. Che cosa succederebbe al collegamento software `rezepte.txt` se il file `receitas.txt` venisse cancellato? Perché?

Il collegamento simbolico `rezepte.txt` smetterebbe di funzionare. Questo perché i collegamenti simbolici (o *soft*) puntano ai nomi, non agli inode, e il nome `receitas.txt` non esiste più, anche se i dati sono ancora sul disco con il nome `recipes.txt`.

2. Immagina di avere una chiavetta USB collegata al tuo sistema e montata in `/media/youruser/FlashA`. Si desidera creare un collegamento chiamato `schematics.pdf` nella directory home, che punta al file `esquema.pdf` nella root dell'unità flash. Quindi, digita il comando:

```
$ ln /media/youruser/FlashA/esquema.pdf ~/schematics.pdf
```

Che cosa succede? Perché

Il comando fallirebbe. Il messaggio di errore sarebbe `Invalid cross-device link` e chiarisce il motivo: gli hard link non possono puntare a una destinazione in una partizione o dispositivo diverso. L'unico modo per creare un collegamento come questo è usare un collegamento *simbolico* o *soft*, aggiungendo il parametro `-s` a `ln`.

3. Considera il seguente output di `ls -lah`:

```
$ ls -lah
total 3,1M
drwxr-xr-x 2 carol carol 4,0K jun 17 17:27 .
drwxr-xr-x 5 carol carol 4,0K jun 17 17:29 ..
-rw-rw-r-- 1 carol carol 2,8M jun 17 15:45 compressed.zip
-rw-r--r-- 4 carol carol 77K jun 17 17:25 document.txt
-rw-rw-r-- 1 carol carol 216K jun 17 17:25 image.png
-rw-r--r-- 4 carol carol 77K jun 17 17:25 text.txt
```

- Quanti link puntano al file `document.txt`?

Ogni file inizia con un conteggio dei collegamenti uguale a 1. Poiché il conteggio dei collegamenti per il file è 4, ci sono altri tre collegamenti che puntano a quel file.

- Sono collegamenti hard o soft?

Sono collegamenti fisici (hard link), poiché i collegamenti simbolici non aumentano il contatore dei collegamenti di un file.

- Quale parametro dovresti passare a `ls` per vedere quale inode occupa ogni file?

Il parametro è `-li`. L'inode verrà mostrato come la prima colonna nell'output di `ls`, come di seguito:

```
$ ls -lahi
total 3,1M
5388773 drwxr-xr-x 2 carol carol 4,0K jun 17 17:27 .
5245554 drwxr-xr-x 5 carol carol 4,0K jun 17 17:29 ..
5388840 -rw-rw-r-- 1 carol carol 2,8M jun 17 15:45 compressed.zip
5388833 -rw-r--r-- 4 carol carol 77K jun 17 17:25 document.txt
5388837 -rw-rw-r-- 1 carol carol 216K jun 17 17:25 image.png
5388833 -rw-r--r-- 4 carol carol 77K jun 17 17:25 text.txt
```

4. Immagina di avere nella tua directory `~/Documents` un file chiamato `clients.txt` contenente alcuni nomi di clienti e una directory chiamata `somedir`. All'interno di questa c'è un file *differente* denominato *anch'esso* `clients.txt` con nomi differenti. Per replicare questa struttura, utilizzare i seguenti comandi.

```
$ cd ~/Documents
```

```
$ echo "John, Michael, Bob" > clients.txt
$ mkdir somedir
$ echo "Bill, Luke, Karl" > somedir/clients.txt
```

Quindi crea un collegamento all'interno di `somedir` denominato `partners.txt` che punta a questo file, con i comandi:

```
$ cd somedir/
$ ln -s clients.txt partners.txt
```

Quindi, la struttura della directory sarà:

```
Documents
|-- clients.txt
`-- somedir
    |-- clients.txt
    `-- partners.txt -> clients.txt
```

Ora, sposta `partners.txt` da `somedir` a `~/Documents`, ed elencane il contenuto.

```
$ cd ~/Documents/
$ mv somedir/partners.txt .
$ less partners.txt
```

Il collegamento funzionerà ancora? In tal caso, quale file avrà il suo contenuto elencato? Perché?

Questo è “un po' complicato”, ma il collegamento funzionerà e il file elencato sarà quello in `~/Documents`, contenente i nomi John, Michael, Bob.

Ricorda che poiché non hai specificato il percorso completo della destinazione `clients.txt` durante la creazione del soft link `partners.txt`, il percorso di destinazione sarà interpretato come relativo alla posizione del link, che in questo caso è la directory corrente.

Quando il collegamento è stato spostato da `~/Documents/somedir` a `~/Documents`, dovrebbe smettere di funzionare, poiché la destinazione non era più nella stessa directory del collegamento. Tuttavia, accade che c'è un file chiamato `clients.txt` su `~/Documents`, quindi il collegamento punterà a questo file, invece della destinazione originale all'interno di `~/somedir`.

Per evitare ciò, specificare sempre il percorso completo della destinazione quando si crea un collegamento simbolico.

5. Considera i seguenti file:

```
-rw-r--r-- 1 carol carol 19 Jun 24 11:12 clients.txt  
lrwxrwxrwx 1 carol carol 11 Jun 24 11:13 partners.txt -> clients.txt
```

Quali sono i permessi di accesso per `partners.txt`? Perché?

I permessi di accesso per `partners.txt` sono `rw-r--r--`, poiché i collegamenti ereditano sempre gli stessi permessi di accesso della destinazione.



104.7 Trovare i file di sistema e collocarli nella posizione corretta

Obiettivi LPI di riferimento

LPIC-1 version 5.0, Exam 101, Objective 104.7

Peso

2

Aree di Conoscenza Chiave

- Comprendere la posizione corretta dei file rispetto allo standard FHS.
- Trovare file e comandi su un sistema Linux.
- Conoscere la posizione e lo scopo di file e directory importanti come definito nello standard FHS.

Di seguito è riportato un elenco parziale dei file, dei termini e dei comandi utilizzati

- `find`
- `locate`
- `updatedb`
- `whereis`
- `which`
- `type`
- `/etc/updatedb.conf`



104.7 Lezione 1

Certificazione:	LPIC-1
Versione:	5.0
Argomento:	104 Dispositivi, Il Filesystem Linux, Filesystem Hierarchy Standard
Obiettivo:	104.7 Trovare i file di sistema e collocarli nella posizione corretta
Lezione:	1 di 1

Introduzione

Le distribuzioni Linux sono disponibili in tutte le forme e dimensioni, ma una cosa che quasi tutte condividono è che seguono il *Filesystem Hierarchy Standard* (FHS), che definisce un “layout standard” per il filesystem, rendendo l’interoperabilità e l’amministrazione del sistema molto più semplice. In questa lezione imparerai di più su questo standard e su come trovare file su un sistema Linux.

Il Filesystem Hierarchy Standard

Il Filesystem Hierarchy Standard (FHS) è uno sforzo della Linux Foundation per standardizzare la struttura delle directory e il contenuto delle stesse sui sistemi Linux. La conformità allo standard non è obbligatoria, ma la maggior parte delle distribuzioni la segue.

NOTE

Chi è interessato ai dettagli dell’organizzazione del filesystem può leggere la specifica FHS 3.0, disponibile in più formati su: <http://refspecs.linuxfoundation.org/fhs.shtml>

Secondo lo standard, la struttura delle directory di base è la seguente:

/

Questa è la directory principale, la directory più in alto nella gerarchia. Ogni altra directory si trova al suo interno. Un filesystem è spesso paragonato a un “albero”, quindi questa sarebbe “la radice” / “il tronco” a cui sono collegati tutti i rami.

/bin

Binari essenziali, disponibili per tutti gli utenti.

/boot

File necessari al processo di avvio, incluso il disco RAM iniziale (initrd) e il kernel Linux stesso.

/dev

File di dispositivo. Questi possono essere dispositivi fisici collegati al sistema (per esempio, `/dev/sda` sarebbe il primo disco SCSI o SATA) o dispositivi virtuali forniti dal kernel.

/etc

File di configurazione specifici dell’host. I programmi possono creare sottodirectory in `/etc` per memorizzare più file di configurazione, se necessario.

/home

Ogni utente nel sistema ha una directory “home” per memorizzare i file e le preferenze personali, e la maggior parte di essi si trova sotto `/home`. Di solito, la directory home è la stessa del nome utente, quindi l’utente *John* avrebbe la sua directory in `/home/john`. Le eccezioni sono il superutente (root), che ha una directory separata (`/root`) e alcuni utenti di sistema.

/lib

Librerie condivise necessarie per avviare il sistema operativo e per eseguire i binari in `/bin` e `/sbin`.

/media

I supporti rimovibili montabili dall’utente, come unità flash, lettori di CD e DVD-ROM, dischi floppy, schede di memoria e dischi esterni sono montati qui dentro.

/mnt

Punto di montaggio per filesystem montati temporaneamente.

/opt

Pacchetti software applicativi.

/root

Directory home per il superutente (root).

/run

Dati variabili di runtime.

/sbin

Binari di sistema.

/srv

Dati forniti dal sistema. Per esempio, le pagine servite da un server web potrebbero essere archiviate in `/srv/www`.

/tmp

File temporanei.

/usr

Dati utente di sola lettura, inclusi i dati necessari per alcune utilità e applicazioni secondarie.

/proc

File system virtuale contenente dati relativi ai processi in esecuzione.

/var

Dati variabili scritti durante il funzionamento del sistema, inclusi coda di stampa, dati di registro, caselle di posta, file temporanei, cache del browser, ecc.

Tieni presente che alcune di queste directory, come `/etc`, `/usr` e `/var`, contengono un'intera gerarchia di sottodirectory.

I File Temporanei

I file temporanei sono file utilizzati dai programmi per archiviare dati necessari solo per un breve periodo. Questi possono essere dati di processi in esecuzione, registri di arresto anomalo, file temporanei da un salvataggio automatico, file intermedi utilizzati durante una conversione di file, file di cache e simili.

Posizione dei File Temporanei

La versione 3.0 del *Filesystem Hierarchy Standard* (FHS) definisce le posizioni standard per i file temporanei sui sistemi Linux. Ogni posizione ha uno scopo e un comportamento diverso ed è consigliato agli sviluppatori di seguire le convenzioni impostate dall'FHS quando creano dati

temporanei su disco.

/tmp

Secondo l'FHS i programmi non dovrebbero presumere che i file scritti qui all'interno verranno conservati tra un'esecuzione e l'altra di un programma. La raccomandazione è che questa directory venga ripulita (tutti i file cancellati) durante l'avvio del sistema, sebbene ciò non sia obbligatorio.

/var/tmp

Un'altra posizione per i file temporanei, ma questa *non dovrebbe essere ripulita* durante l'avvio del sistema. I file archiviati qui all'interno, di solito, persistono tra i riavvii.

/run

Questa directory contiene i dati delle variabili di runtime utilizzati dai processi in esecuzione, come i file di identificazione del processo (.pid). I programmi che richiedono più di un file di runtime possono creare sottodirectory qui dentro. Questa posizione *deve essere ripulita* durante l'avvio del sistema. Lo scopo di questa directory una volta era svolto da /var/run, e ancora oggi, su alcuni sistemi, /var/run potrebbe essere un collegamento simbolico a /run.

Notare che nulla impedisce a un programma di creare file temporanei altrove nel sistema, ma è buona norma rispettare le convenzioni impostate dall'FHS.

Ricerca di File

Per cercare file su un sistema Linux, puoi usare il comando `find`. Questo è uno strumento molto potente, ricco di opzioni e parametri che possono adattarne il comportamento e modificare l'output esattamente alle tue esigenze.

Per iniziare, `find` ha bisogno di due argomenti: un punto di partenza e che cosa cercare. Per esempio, per cercare tutti i file nella directory corrente (e nelle sottodirectory) il cui nome termina con `.jpg` puoi usare:

```
$ find . -name '*.jpg'
./pixel_3a_seethrough_1.jpg
./Mate3.jpg
./Expert.jpg
./Pentaro.jpg
./Mate1.jpg
./Mate2.jpg
./Sala.jpg
./Hotbit.jpg
```


Questo corrisponderà a qualsiasi file i cui ultimi quattro caratteri del nome siano “.jpg”, indipendentemente da ciò che viene prima, poiché `*` è un carattere jolly che indica “qualsiasi cosa”. Tuttavia, guarda cosa succede se un altro `*` viene aggiunto alla *fine* della maschera di ricerca (pattern):

```
$ find . -name '*.jpg*'
./pixel_3a_seethrough_1.jpg
./Pentaro.jpg.zip
./Mate3.jpg
./Expert.jpg
./Pentaro.jpg
./Mate1.jpg
./Mate2.jpg
./Sala.jpg
./Hotbit.jpg
```

Il file `Pentaro.jpg.zip` (evidenziato sopra) non era incluso nell’elenco precedente, perché anche se contiene `.jpg` nel nome, non corrispondeva al pattern perché c’erano caratteri extra dopo di esso. Il nuovo modello significa “qualsiasi cosa `.jpg` qualunque cosa”, quindi corrisponde.

TIP

Tieni presente che il parametro `-name` fa distinzione tra maiuscole e minuscole. Se desideri eseguire una ricerca senza questa distinzione, usa `-iname`.

L’espressione `*.jpg` deve essere inserita tra *virgolette singole*, per evitare che la shell interpreti il pattern stesso. Prova senza virgolette e guarda cosa succede.

Per impostazione predefinita, `find` inizierà dal punto di partenza e scenderà attraverso tutte le sottodirectory (e le sottodirectory di quelle sottodirectory) trovate. Puoi limitare questo comportamento con i parametri `-maxdepth N`, dove `N` è il numero massimo di livelli.

Per cercare solo nella directory corrente, dovresti usare `-maxdepth 1`. Supponiamo di avere la seguente struttura di directory:

```
directory
├─ clients.txt
├─ partners.txt -> clients.txt
└─ somedir
   └─ anotherdir
      └─ clients.txt
```

Per cercare all’interno di `somedir`, dovresti usare `-maxdepth 2` (la directory corrente +1 livello in

basso). Per cercare all'interno di `anotherdir`, sarebbe necessario `-maxdepth 3` (la directory corrente +2 livelli in basso). Il parametro `-mindepth N` funziona in modo opposto cercando solo nelle directory *almeno* i livelli inferiori di N.

Il parametro `-mount` può essere usato per evitare che `find` scenda all'interno dei filesystem montati. Puoi anche limitare la ricerca a specifici tipi di filesystem usando il parametro `-fstype`. Quindi `find /mnt -fstype exfat -iname "*report*"` cercherà solo all'interno dei filesystem exFAT montati sotto `/mnt`.

Ricerca di Attributi

È possibile utilizzare i parametri seguenti per cercare file con attributi specifici, come quelli che siano scrivibili dall'utente, abbiano un insieme specifico di autorizzazioni o abbiano una certa dimensione:

-user USERNAME

Corrisponde ai file di proprietà dell'utente `USERNAME`.

-group GROUPNAME

Corrisponde ai file di proprietà del gruppo `GROUPNAME`.

-readable

Corrisponde a file leggibili dall'utente corrente.

-writable

Corrisponde a file scrivibili dall'utente corrente.

-executable

Corrisponde ai file eseguibili dall'utente corrente. Nel caso delle directory, questo corrisponderà a qualsiasi directory a cui l'utente può accedere (autorizzazione `x`).

-perm NNNN

Questo corrisponderà a tutti i file che hanno esattamente l'autorizzazione `NNNN`. Per esempio, `-perm 0664` corrisponderà a qualsiasi file su cui l'utente e il gruppo possono leggere e scrivere e su cui altri possono leggere (o `rw-rw-r-`).

È possibile aggiungere un `-` prima di `NNNN` per verificare la presenza di file che hanno *almeno* l'autorizzazione specificata. Per esempio, `-perm -644` corrisponderebbe a file che hanno almeno i permessi `644` (`rw-r-r-`). Questo include un file con `664` (`rw-rw-r-`) o anche `775` (`rw-rwx-r-x`).

-empty

Corrisponde a file e directory vuote.

-size N

Corrisponde a qualsiasi file di dimensione N, dove N di default è un numero di blocchi da 512 byte. Puoi aggiungere suffissi a N per altre unità: **Nc** conterà la dimensione in byte, **Nk** in kibibyte (KiB, multipli di 1024 byte), **NM** in mebibyte (MiB, multipli di 1024 * 1024) e **NG** per i gibibyte (GiB, multipli di 1024 * 1024 * 1024).

Di nuovo, puoi aggiungere i prefissi **+** o **-** (qui significa *più grande di* e *più piccolo di*) per cercare dimensioni relative. Per esempio, **-size -10M** corrisponderà a qualsiasi file di dimensioni inferiori a 10 MiB.

Per esempio, per cercare i file nella tua home directory che contengono il pattern insensibile al maiuscolo/minuscolo **report** in qualsiasi parte del nome, hanno i permessi **0644**, sono stati acceduti 10 giorni fa e la cui dimensione è almeno 1 Mib, puoi usare:

```
$ find ~ -iname "*report*" -perm 0644 -atime 10 -size +1M
```

Ricerche Temporal

Oltre a cercare gli attributi, puoi anche eseguire ricerche temporali, trovando i file a cui è stato effettuato l'accesso, i cui attributi sono stati modificati o sono stati modificati durante un periodo di tempo specifico. I parametri sono:

-amin N, -cmin N, -mmin N

Per trovare file a cui è stato effettuato l'accesso, che hanno avuto attributi modificati o sono stati modificati (rispettivamente) N minuti fa.

-atime N, -ctime N, -mtime N

Per trovare file a cui è stato effettuato l'accesso, a cui sono stati modificati gli attributi o che sono stati modificati N*24 ore fa.

Per **-cmin N** e **-ctime N**, qualsiasi modifica di attributo causerà una corrispondenza, inclusa una modifica dei permessi, la lettura o la scrittura del file. Ciò rende questi parametri particolarmente potenti, poiché praticamente qualsiasi operazione che coinvolge il file attiverà una corrispondenza.

Il seguente esempio permette una corrispondenza a qualsiasi file nella directory corrente che è stato modificato meno di 24 ore fa ed è più grande di 100 MiB:

```
$ find . -mtime -1 -size +100M
```

Utilizzo di `locate` e `updatedb`

`locate` e `updatedb` sono comandi che possono essere usati per trovare rapidamente un file che corrisponde a un dato modello su un sistema Linux. Ma a differenza di `find`, `locate` non cercherà il pattern nel filesystem: invece, lo cerca su un database costruito eseguendo il comando `updatedb`. Ciò fornisce risultati molto rapidi, ma che potrebbero essere imprecisi a seconda dell'ultimo aggiornamento del database.

Il modo più semplice per usare `locate` è semplicemente dargli un pattern da cercare. Per esempio, per trovare ogni immagine JPEG sul tuo sistema, dovresti usare `locate jpg`. L'elenco dei risultati può essere piuttosto lungo, ma dovrebbe essere simile a questo:

```
$ locate jpg
/home/carol/Downloads/Expert.jpg
/home/carol/Downloads/Hotbit.jpg
/home/carol/Downloads/Mate1.jpg
/home/carol/Downloads/Mate2.jpg
/home/carol/Downloads/Mate3.jpg
/home/carol/Downloads/Pentaro.jpg
/home/carol/Downloads/Sala.jpg
/home/carol/Downloads/pixel_3a_seethrough_1.jpg
/home/carol/Downloads/jpg_specs.doc
```

Quando viene richiesto il pattern `jpg`, `locate` mostrerà tutto ciò che contiene questo pattern, non importa cosa venga prima o dopo di esso. Puoi vedere un esempio di ciò nel file `jpg_specs.doc` nell'elenco sopra: contiene il pattern, ma l'estensione non è `jpg`.

TIP Ricorda che con `locate` stai facendo corrispondere i pattern, non le estensioni di file.

Per impostazione predefinita, il pattern fa distinzione tra maiuscole e minuscole. Ciò significa che i file contenenti `.JPG` non verrebbero visualizzati poiché scritto in minuscolo. Per evitare ciò, passare il parametro `-i` a `locate`. Ripetendo il nostro esempio precedente:

```
$ locate -i .jpg
/home/carol/Downloads/Expert.jpg
/home/carol/Downloads/Hotbit.jpg
/home/carol/Downloads/Mate1.jpg
/home/carol/Downloads/Mate1_old.JPG
```

```

/home/carol/Downloads/Mate2.jpg
/home/carol/Downloads/Mate3.jpg
/home/carol/Downloads/Pentaro.jpg
/home/carol/Downloads/Sala.jpg
/home/carol/Downloads/pixel_3a_seethrough_1.jpg

```

Si noti che il file `Mate1_old.JPG`, in grassetto sopra, non era presente nell'elenco precedente.

Puoi passare più pattern a `locate`, semplicemente separandoli con spazi. L'esempio seguente eseguirà una ricerca senza distinzione tra maiuscole e minuscole per tutti i file che corrispondono a `zip` e `jpg`:

```

$ locate -i zip jpg
/home/carol/Downloads/Expert.jpg
/home/carol/Downloads/Hotbit.jpg
/home/carol/Downloads/Mate1.jpg
/home/carol/Downloads/Mate1_old.JPG
/home/carol/Downloads/Mate2.jpg
/home/carol/Downloads/Mate3.jpg
/home/carol/Downloads/OPENMSXPIHAT.zip
/home/carol/Downloads/Pentaro.jpg
/home/carol/Downloads/Sala.jpg
/home/carol/Downloads/gbs-control-master.zip
/home/carol/Downloads/lineage-16.0-20190711-MOD-quark.zip
/home/carol/Downloads/pixel_3a_seethrough_1.jpg
/home/carol/Downloads/jpg_specs.doc

```

Quando si utilizzano più pattern, è possibile richiedere a `locate` di mostrare solo i file che corrispondono a *tutti* i pattern di ricerca. Questo viene fatto con l'opzione `-A`. Il seguente esempio mostrerebbe qualsiasi file che corrisponde al pattern `.jpg` e al pattern `.zip`:

```

$ locate -A .jpg .zip
/home/carol/Downloads/Pentaro.jpg.zip

```

Se desideri contare il numero di file che corrispondono a un dato pattern invece di mostrare il loro percorso completo, puoi usare l'opzione `-c`. Per esempio, per contare il numero di file `.jpg` su un sistema:

```

$ locate -c .jpg
1174

```

Un problema con `locate` è che mostra solo le voci presenti nel database generato da `updatedb` (situato in `/var/lib/mlocate.db`). Se il database è obsoleto, l'output potrebbe mostrare file che invece sono stati eliminati dall'ultima volta che è stato aggiornato. Un modo per evitare questo è aggiungere il parametro `-e`, che controllerà se il file esiste ancora prima di mostrarlo sull'output.

Ovviamente, questo non risolverà il problema dei file creati *dopo* l'ultimo aggiornamento del database non visualizzato. Per questo dovrai aggiornare il database con il comando `updatedb`. Quanto tempo ci vorrà dipenderà dalla quantità di file nel disco.

Controllare il Comportamento di `updatedb`

Il comportamento di `updatedb` può essere controllato dal file `/etc/updatedb.conf`. Questo è un file di testo in cui ogni riga controlla una variabile. Le linee vuote e quelle che iniziano con il carattere `#`(commenti) vengono ignorate.

PRUNEFS=

Qualsiasi tipo di filesystem indicato dopo questo parametro non verrà scansionato da `updatedb`. L'elenco dei tipi di filesystem dovrebbe essere separato da spazi e i tipi stessi non fanno distinzione tra maiuscole e minuscole, quindi `NFS` e `nfs` saranno considerati allo stesso modo.

PRUNENAMES=

Questo è un elenco separato da spazi di nomi di directory che non dovrebbero essere scansionati da `updatedb`.

PRUNEPATHS=

Questa è una lista di percorsi che dovrebbero essere ignorati da `updatedb`. I nomi dei percorsi devono essere separati da spazi e specificati nello stesso modo in cui verrebbero visualizzati da `updatedb` (per esempio, `/var/spool/media`)

PRUNE_BIND_MOUNTS=

Questa è una semplice variabile `yes` o `no`. Se impostato su `yes` (le directory montate altrove con il comando `mount --bind`) verranno ignorate.

Trovare Binari, Pagine di manuale e Codice Sorgente

`which` è un comando molto utile che mostra il percorso completo di un eseguibile. Per esempio, se vuoi individuare l'eseguibile per `bash`, potresti usare:

```
$ which bash
```

```
/usr/bin/bash
```

Se viene aggiunta l'opzione `-a`, il comando mostrerà tutti i nomi di percorso che corrispondono all'eseguibile. Osserva la differenza:

```
$ which mkfs.ext3
/usr/sbin/mkfs.ext3

$ which -a mkfs.ext3
/usr/sbin/mkfs.ext3
/sbin/mkfs.ext3
```

TIP

Per trovare quali directory sono nella variabile `PATH` usa il comando `echo $PATH`. Questo visualizzerà (echo) il contenuto della variabile `PATH` (`$PATH`) sul tuo terminale.

`type` è un comando simile che mostrerà le informazioni su un binario, incluso dove si trova e il suo tipo. Usa semplicemente `type` seguito dal nome del comando

```
$ type locate
locate is /usr/bin/locate
```

L'opzione `-a` funziona allo stesso modo di `which`, mostrando tutti i nomi di percorso che corrispondono all'eseguibile. Così:

```
$ type -a locate
locate is /usr/bin/locate
locate is /bin/locate
```

E l'opzione `-t` mostrerà il tipo di file del comando che può essere un `alias`, una `keyword`, una `function`, un `builtin` o un `file`. Per esempio:

```
$ type -t locate
file

$ type -t ll
alias

$ type -t type
```

```
type is a built-in shell command
```

Il comando `whereis` è più versatile e oltre ai binari può anche essere usato per mostrare la posizione delle pagine `man` o anche il codice sorgente di un programma (se disponibile nel tuo sistema). Basta digitare `whereis` seguito dal nome del binario:

```
$ whereis locate
locate: /usr/bin/locate /usr/share/man/man1/locate.1.gz
```

I risultati sopra includono binari (`/usr/bin/locate`) e pagine di manuale compresse (`/usr/share/man/man1/locate.1.gz`).

Puoi filtrare rapidamente i risultati usando le opzioni della riga di comando come `-b`, che li limiterà solo ai binari, `-m`, che li limiterà solo alle pagine `man`, o `-s`, che li limiterà solo al codice sorgente. Ripetendo l'esempio sopra, otterrai:

```
$ whereis -b locate
locate: /usr/bin/locate

$ whereis -m locate
locate: /usr/share/man/man1/locate.1.gz
```


Esercizi Guidati

1. Immagina che un programma debba creare un file temporaneo monouso che non sarà mai più necessario dopo la chiusura del programma. Quale sarebbe la directory corretta in cui creare questo file?
2. Qual è la directory temporanea che *deve* essere cancellata durante il processo di avvio?
3. Usando `find`, cerca solo nella directory corrente i file che sono scrivibili dall'utente, sono stati modificati negli ultimi 10 giorni e sono più grandi di 4 GiB.
4. Usando `locate`, trova tutti i file che contengono sia i pattern `report` che `updated`, `update` o `updating` nei loro nomi.
5. Come puoi trovare dove è memorizzata la pagina di manuale di `ifconfig`?
6. Quale variabile deve essere aggiunta a `/etc/updatedb.conf` per fare in modo che `updatedb` ignori i filesystem `ntfs`?
7. Un amministratore di sistema desidera montare un disco interno (`/dev/sdc1`). Secondo FHS, in quale directory dovrebbe essere montato questo disco?

Esercizi Esplorativi

1. Quando viene utilizzato `locate`, i risultati vengono estratti da un database generato da `updatedb`. Tuttavia, questo database può essere obsoleto, facendo sì che `locate` mostri file che non esistono più. Come puoi fare in modo che `locate` mostri solo i file esistenti sul suo output?
2. Trova qualsiasi file nella directory corrente o nelle sottodirectory fino a 2 livelli inferiori, esclusi i filesystem montati, che contengono il pattern `Status` o `statute` nei loro nomi.
3. Limitando la ricerca ai filesystem `ext4`, trova tutti i file all'interno di `/mnt` che hanno almeno i permessi di esecuzione per il gruppo, sono leggibili per l'utente corrente e hanno avuto un attributo cambiato nelle ultime 2 ore.
4. Trova i file vuoti che sono stati creati più di 30 giorni fa e sono almeno due livelli più in basso rispetto alla directory corrente.
5. Considera che gli utenti `carol` e `john` fanno parte del gruppo `mkt`. Trova nella home directory di `john` tutti i file che sono leggibili anche da `carol`.

Sommario

In questa lezione hai imparato l'organizzazione di base del filesystem su una macchina Linux, secondo FHS, e come trovare binari e file, sia per nome che per attributi. In questa lezione sono stati discussi i seguenti comandi:

find

Un comando versatile usato per trovare file e cartelle in base a una varietà di criteri di ricerca.

locate

Un'utilità che utilizza un database locale che contiene le posizioni per i file memorizzati localmente.

updatedb

Aggiorna il database locale utilizzato dal comando `locate`.

which

Visualizza il percorso completo di un eseguibile.

whereis

Visualizza le posizioni delle pagine di manuale, dei binari e del codice sorgente sul sistema.

type

Visualizza la posizione di un binario e che tipo di applicazione è (come un programma installato, un programma Bash integrato e altro).

Risposte agli Esercizi Guidati

1. Immagina che un programma debba creare un file temporaneo monouso che non sarà mai più necessario dopo la chiusura del programma. Quale sarebbe la directory corretta in cui creare questo file?

Dato che non ci interessa il file dopo che il programma avrà terminato l'esecuzione, la directory corretta è `/tmp`.

2. Qual è la directory temporanea che *deve* essere cancellata durante il processo di avvio?

La directory è `/run` o, su alcuni sistemi, `/var/run`.

3. Usando `find`, cerca solo nella directory corrente i file che sono scrivibili dall'utente, sono stati modificati negli ultimi 10 giorni e sono più grandi di 4 GiB.

Per questo avrai bisogno dei parametri `-writable`, `-mtime` e `-size`:

```
find . -writable -mtime -10 -size +4G
```

4. Usando `locate`, trova tutti i file che contengono sia i pattern `report` che `updated`, `update` o `updating` nei loro nomi.

Dal momento che `locate` deve corrispondere a tutti i pattern, usa l'opzione `-A`:

```
locate -A "report" "updat"
```

5. Come puoi trovare dove è memorizzata la pagina di manuale di `ifconfig`?

Usa l'opzione `-m` per `whereis`:

```
whereis -m ifconfig
```

6. Quale variabile deve essere aggiunta a `/etc/updatedb.conf` per fare in modo che `updatedb` ignori i filesystem `ntfs`?

La variabile è `PRUNEFS=` seguita dal tipo di filesystem: `PRUNEFS=ntfs`

7. Un amministratore di sistema desidera montare un disco interno (`/dev/sdc1`). Secondo FHS, in quale directory dovrebbe essere montato questo disco?

In pratica, il disco può essere montato ovunque. Tuttavia, l'FHS raccomanda che i montaggi temporanei vengano eseguiti in `/mnt`.

Risposte agli Esercizi Esplorativi

1. Quando viene utilizzato `locate`, i risultati vengono estratti da un database generato da `updatedb`. Tuttavia, questo database può essere obsoleto, facendo sì che `locate` mostri file che non esistono più. Come puoi fare in modo che `locate` mostri solo i file esistenti sul suo output?

Aggiungi il parametro `-e`, come in `locate -e PATTERN`.

2. Trova qualsiasi file nella directory corrente o nelle sottodirectory fino a 2 livelli inferiori, esclusi i filesystem montati, che contengono il pattern `Status` o `statute` nei loro nomi.

Ricorda che per `-maxdepth` devi considerare anche la directory corrente, quindi vogliamo tre livelli (l'attuale più 2 livelli in basso):

```
find . -maxdepth 3 -mount -iname "*statu*"
```

3. Limitando la ricerca ai filesystem `ext4`, trova tutti i file all'interno di `/mnt` che hanno almeno i permessi di esecuzione per il gruppo, sono leggibili per l'utente corrente e hanno avuto un attributo cambiato nelle ultime 2 ore.

Usa il parametro `-fstype` di `mount` per limitare la ricerca a specifici tipi di filesystem. Un file leggibile dall'utente corrente avrebbe almeno 4 nella prima cifra dei permessi, e un eseguibile dal gruppo avrebbe almeno 1 nella seconda cifra. Dato che non ci interessano le autorizzazioni per gli altri, possiamo usare 0 per la terza cifra. Usa `-cmin N` per filtrare le modifiche recenti agli attributi, ricordando che N è specificato in minuti. Così:

```
find /mnt -fstype ext4 -perm -410 -cmin -120
```

4. Trova i file vuoti che sono stati creati più di 30 giorni fa e sono almeno due livelli più in basso rispetto alla directory corrente.

Il parametro `-mindepth N` può essere usato per limitare la ricerca ad almeno N livelli inferiori, ma ricorda che devi includere la directory corrente nel conteggio. Usa `-empty` per verificare la presenza di file vuoti e `-mtime N` per verificare l'ora di modifica. Così:

```
find . -empty -mtime +30 -mindepth 3
```

5. Considera che gli utenti `carol` e `john` fanno parte del gruppo `mkt`. Trova nella home directory di `john` tutti i file che sono leggibili anche da `carol`.

Considerando che sono membri dello stesso gruppo, abbiamo bisogno di almeno una **r** (4) sui permessi del gruppo e non ci interessa nulla degli altri. Così:

```
find /home/john -perm -040
```

Imprint

© 2025 by Linux Professional Institute: Learning Materials, “LPIC-1 (101) (Version 5.0)”.

PDF generato: 2025-03-12

Questa opera è concessa in licenza con Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 International (CC BY-NC-ND 4.0). Per visualizzare una copia di questa licenza, visitare

<https://creativecommons.org/licenses/by-nc-nd/4.0/>

Sebbene Linux Professional Institute si sia adoperato in buona fede per garantire che le informazioni e le istruzioni contenute in questa opera siano accurate, Linux Professional Institute declina ogni responsabilità per errori od omissioni, inclusa, senza limitazione, la responsabilità per danni derivanti dall’uso di questa opera. L’utilizzo di informazioni e istruzioni contenute in questa opera è a proprio rischio. Se qualche esempio di codice o tecnologia che questa opera contiene o descrive è soggetto a licenze open source o è sotto diritti di proprietà intellettuale di terzi, è tua responsabilità assicurarti che se ne faccia uso rispettando tali licenze e / o diritti.

I materiali didattici LPI (Learning Materials) sono un’iniziativa Linux Professional Institute (<https://lpi.org>). Materiali didattici e loro traduzioni sono su <https://learning.lpi.org>.

Per domande e commenti scrivi una mail a: learning@lpi.org.