Linux Professional Institute

LPIC-1

Versione 5.0
Italiano

102

Table of Contents

ARGOMENTO 105: SHELL E SCRIPT DI SHELL **1**
105.1 Personalizzare e utilizzare l'ambiente di shell **2**
105.1 Lezione 1 4
Introduzione 4
Tipi di shell: Interattiva vs Non-interattiva ; Login vs Senza-Login 5
Esercizi Guidati 18
Esercizi Esplorativi 20
Sommario 22
Risposte agli Esercizi Guidati 24
Risposte agli Esercizi Esplorativi 26
105.1 Lezione 2 28
Introduzione 28
Variabili: Assegnazione e Riferimento 28
Variabili Locali o di Shell 32
Variabili Globali o di Ambiente 35
Esercizi Guidati 45
Esercizi Esplorativi 48
Sommario 50
Risposte agli Esercizi Guidati 52
Risposte agli Esercizi Esplorativi 56
105.1 Lezione 3 58
Introduzione 58
Creazione di Alias 58
Creazione di Funzioni 63
Esercizi Guidati 73
Esercizi Esplorativi 76
Sommario 77
Risposte agli Esercizi Guidati 79
Risposte agli Esercizi Esplorativi 84
105.2 Personalizzare o scrivere semplici script **85**
105.2 Lezione 1 87
Introduzione 87
Struttura degli Script ed Esecuzione 88
Variabili 90
Espressioni Aritmetiche 93
Esecuzione Condizionale 94
Output di uno Script 95
Esercizi Guidati 98

Esercizi Esplorativi . . . 99
Sommario . . . 100
Risposte agli Esercizi Guidati . . . 101
Risposte agli Esercizi Esplorativi . . . 102
105.2 Lezione 2 . . . 103
Introduzione . . . 103
Test Estesi . . . 103
Costrutti di Loop . . . 109
Un Esempio più Elaborato . . . 111
Esercizi Guidati . . . 115
Esercizi Esplorativi . . . 117
Sommario . . . 118
Risposte agli Esercizi Guidati . . . 119
ARGOMENTO 106: INTERFACCE UTENTE E DESKTOP . . . 121
106.1 Installare e configurare X11 . . . 122
106.1 Lezione 1 . . . 123
Introduzione . . . 123
Architettura di Sistema X Window . . . 124
Wayland . . . 132
Esercizi Guidati . . . 134
Esercizi Esplorativi . . . 135
Sommario . . . 136
Risposte agli Esercizi Guidati . . . 137
Risposte agli Esercizi Esplorativi . . . 138
106.2 Desktop grafici . . . 139
106.2 Lezione 1 . . . 140
Introduzione . . . 140
Il Sitema X Window . . . 141
L'ambiente Desktop . . . 141
Ambienti Desktop Popolari . . . 143
Interoperabilità Desktop . . . 145
Accesso Non Locale . . . 146
Esercizi Guidati . . . 149
Esercizi Esplorativi . . . 150
Sommario . . . 151
Risposte agli Esercizi Guidati . . . 152
Risposte agli Esercizi Esplorativi . . . 153
106.3 Accessibilità . . . 154
106.3 Lezione 1 . . . 155
Introduzione . . . 155

Impostazioni di Accessibilità . . . 155
Assistenza a Tastiera e Mouse . . . 156
Disabilità Visive . . . 158
Esercizi Guidati . . . 160
Esercizi Esplorativi . . . 161
Sommario . . . 162
Risposte agli Esercizi Guidati . . . 163
Risposte agli Esercizi Esplorativi . . . 164
ARGOMENTO 107: ATTIVITÀ AMMINISTRATIVE . . . 165
107.1 Gestire account utente e gruppo e file di sistema correlati . . . 166
107.1 Lezione 1 . . . 168
Introduzione . . . 168
Aggiungere un Account Utente . . . 168
Modificare un Account Utente . . . 170
Eliminare un Account Utente . . . 172
Aggiungere, Modificare e Rimuovere i Gruppi . . . 172
La Directory Skeleton . . . 173
Il File `/etc/login.defs` . . . 173
Il Comando `passwd` . . . 174
Il Comando `chage` . . . 176
Esercizi Guidati . . . 177
Esercizi Esplorativi . . . 178
Sommario . . . 179
Risposte agli Esercizi Guidati . . . 181
Risposte agli Esercizi Esplorativi . . . 183
107.1 Lezione 2 . . . 185
Introduzione . . . 185
/etc/passwd . . . 186
/etc/group . . . 186
/etc/shadow . . . 187
/etc/gshadow . . . 188
Filtrare i Database delle Password e dei Gruppi . . . 188
Esercizi Guidati . . . 190
Esercizi Esplorativi . . . 192
Sommario . . . 193
Risposte agli Esercizi Guidati . . . 194
Risposte agli Esercizi Esplorativi . . . 196
107.2 Automatizzare le attività di amministrazione del sistema attraverso la pianificazione . . . 198
107.2 Lezione 1 . . . 200

Introduzione . . . 200
Pianificare i Lavori con Cron . . . 200
Crontab Utente . . . 201
Crontab di Sistema . . . 202
Particolari Specifiche di Tempo . . . 203
Variabili Crontab . . . 203
Creare Attività di Cron Utente . . . 204
Creare Attività di Cron di Sistema . . . 205
Configurare l'Accesso alla Pianificazione delle Attività . . . 206
Una Alternativa a Cron . . . 206
Esercizi Guidati . . . 209
Esercizi Esplorativi . . . 211
Sommario . . . 212
Risposte agli Esercizi Guidati . . . 213
Risposte agli Esercizi Esplorativi . . . 215
107.2 Lezione 2 . . . 217
Introduzione . . . 217
Programmare attività con `at` . . . 217
Mostrare i Lavori Pianificati con `atq` . . . 218
Rimuovere i Lavori Pianificati con `atrm` . . . 219
Configurare l'Accesso alla Pianificazione delle Attività . . . 219
Specifiche di Tempo . . . 220
Una Alternativa a `at` . . . 220
Esercizi Guidati . . . 222
Esercizi Esplorativi . . . 223
Sommario . . . 224
Risposte agli Esercizi Guidati . . . 225
Risposte agli Esercizi Esplorativi . . . 226
107.3 Localizzazione e internazionalizzazione . . . 228
107.3 Lezione 1 . . . 230
Introduzione . . . 230
Fusi Orari . . . 231
Ora Legale . . . 235
Lingua e Codifica dei Caratteri . . . 236
Conversione della Codifica . . . 239
Esercizi Guidati . . . 240
Esercizi Esplorativi . . . 241
Sommario . . . 242
Risposte agli Esercizi Guidati . . . 243
Risposte agli Esercizi Esplorativi . . . 244

ARGOMENTO 108: SERVIZI ESSENZIALI DI SISTEMA . . . **245**
108.1 Mantenere l'orario di sistema . . . **246**
108.1 Lezione 1 . . . 248
Introduzione . . . 248
Tempo Locale e Tempo Universale a Confronto . . . 249
La Data . . . 249
Hardware Clock . . . 251
Configurare il Fuso Oraio senza timedatectl . . . 253
Impostare Data e Ora senza timedatectl . . . 254
Esercizi Guidati . . . 257
Esercizi Esplorativi . . . 259
Sommario . . . 260
Risposte agli Esercizi Guidati . . . 262
Risposte agli Esercizi Esplorativi . . . 264
108.1 Lezione 2 . . . 265
Introduzione . . . 265
timedatectl . . . 267
Il Demone NTP . . . 268
La Configurazione NTP . . . 269
pool.ntp.org . . . 270
ntpdate . . . 270
ntpq . . . 270
chrony . . . 271
Esercizi Guidati . . . 276
Esercizi Esplorativi . . . 278
Sommario . . . 279
Risposte agli Esercizi Guidati . . . 280
Risposte agli Esercizi Esplorativi . . . 282
108.2 Logging di sistema . . . **283**
108.2 Lezione 1 . . . 285
Introduzione . . . 285
Il Log di Sistema . . . 285
Esercizi Guidati . . . 306
Esercizi Esplorativi . . . 308
Sommario . . . 309
Risposte agli Esercizi Guidati . . . 310
Risposte agli Esercizi Esplorativi . . . 313
108.2 Lezione 2 . . . 314
Introduzione . . . 314
Fondamenti di `systemd` . . . 314

Il Sistema Journal: `systemd-journald` 315
Esercizi Guidati 333
Esercizi Esplorativi 335
Sommario 336
Risposte agli Esercizi Guidati 337
Risposte agli Esercizi Esplorativi 339
108.3 Concetti base dei Mail Transfer Agent (MTA) 340
108.3 Lezione 1 341
Introduzione 341
MTA Locale e Remoto 342
MTA in Linux 343
Il Comando `mail` e i Mail User Agent (MUA) 348
Personalizzazione della Consegna 349
Esercizi Guidati 352
Esercizi Esplorativi 353
Sommario 354
Risposte agli Esercizi Guidati 355
Risposte agli Esercizi Esplorativi 356
108.4 Gestire stampa e stampanti 357
108.4 Lezione 1 358
Introduzione 358
Il Servizio CUPS 359
Installare una Stampante 363
Gestire le Stampanti 365
Inviare Lavori di Stampa 366
Gestire i Lavori di Stampa 369
Rimuovere le Stampanti 370
Esercizi Guidati 372
Esercizi Esplorativi 373
Sommario 374
Risposte agli Esercizi Guidati 376
Risposte agli Esercizi Esplorativi 377
ARGOMENTO 109: FONDAMENTI DI NETWORKING 379
109.1 Fondamenti dei protocolli Internet 380
109.1 Lezione 1 381
Introduzione 381
IP (Internet Protocol) 381
Esercizi Guidati 390
Esercizi Esplorativi 391
Summario 392

Risposte agli Esercizi Guidati . . . 393
Risposte agli Esercizi Esplorativi . . . 394
109.1 Lezione 2 . . . 395
Introduzione . . . 395
Transmission Control Protocol (TCP) . . . 397
User Datagram Protocol (UDP) . . . 397
Internet Control Message Protocol (ICMP) . . . 397
IPv6 . . . 398
Esercizi Guidati . . . 401
Esercizi Esplorativi . . . 402
Sommario . . . 403
Risposte agli Esercizi Guidati . . . 404
Risposte agli Esercizi Esplorativi . . . 405
109.2 Configurazione di rete persistente . . . 406
109.2 Lezione 1 . . . 407
Introduzione . . . 407
L'Interfaccia di Rete . . . 407
Nomi di Interfaccia . . . 409
Gestione dell'Interfaccia . . . 410
Nomi Locali e Remoti . . . 412
Esercizi Guidati . . . 417
Esercizi Esplorativi . . . 418
Sommario . . . 419
Risposte agli Esercizi Guidati . . . 420
Risposte agli Esercizi Esplorativi . . . 421
109.2 Lezione 2 . . . 422
Introduzione . . . 422
NetworkManager . . . 422
systemd-networkd . . . 427
Esercizi Guidati . . . 430
Esercizi Esplorativi . . . 431
Sommario . . . 432
Risposte agli Esercizi Guidati . . . 433
Risposte agli Esercizi Esplorativi . . . 434
109.3 Risoluzione dei problemi di base di una rete . . . 435
109.3 Lezione 1 . . . 437
Introduzione . . . 437
Il Comando `ip` . . . 438
Controllo della Maschera di Rete e dell'Instradamento . . . 439
Configurare un'Interfaccia . . . 440

La Tabella di Routing . . . 442
Esercizi Guidati . . . 446
Esercizi Esplorativi . . . 447
Sommario . . . 448
Risposte agli Esercizi Guidati . . . 449
Risposte agli Esercizi Esplorativi . . . 451
109.3 Lezione 2 . . . 453
Introduzione . . . 453
Fare Test di Connessione con `ping` . . . 453
Tracciare le Rotte . . . 454
Trovare le MTU con `tracepath` . . . 457
Creare Connessioni Arbitrarie . . . 457
Visualizzazione delle Connessioni Attive e/o in Ascolto . . . 459
Esercizi Guidati . . . 461
Esercizi Esplorativi . . . 462
Sommario . . . 463
Risposte agli Esercizi Guidati . . . 465
Risposte agli Esercizi Esplorativi . . . 467
109.4 Configurare un client DNS . . . 469
109.4 Lezione 1 . . . 470
Introduzione . . . 470
Il Processo di Risoluzione dei Nomi . . . 470
Le Classi DNS . . . 471
Strumenti per la Risoluzione dei Nomi . . . 474
Esercizi Guidati . . . 480
Esercizi Esplorativi . . . 481
Sommario . . . 482
Risposte agli Esercizi Guidati . . . 483
Risposte agli Esercizi Esplorativi . . . 484
ARGOMENTO 110: SICUREZZA . . . 486
110.1 Eseguire attività di amministrazione della sicurezza . . . 487
110.1 Lezione 1 . . . 489
Introduzione . . . 489
Controllo dei File con SUID e SGID Attivo . . . 489
Gestione e Scadenza delle Password . . . 492
Scoprire le Porte Aperte . . . 495
Limiti ai Login degli Utenti, ai Processi e all'Uso della Memoria . . . 502
Trattare con gli Utenti in Sessione . . . 505
Configurazione e Utilizzo di Base di `sudo` . . . 507
Esercizi Guidati . . . 513

Esercizi Esplorativi . . . 516
Sommario . . . 517
Risposte agli Esercizi Guidati . . . 519
Risposte agli Esercizi Esplorativi . . . 523
110.2 Configurare la sicurezza dell'host . . . 524
110.2 Lezione 1 . . . 525
Introduzione . . . 525
Migliorare la Sicurezza dell'Autenticazione con le Shadow Password . . . 525
Come Mettere in Ascolto un Superdaemon sulle Connessioni di Rete in Entrata. . . . 527
Controllare i Servizi per i Demoni non Necessari . . . 532
TCP Wrapper come una Sorta di Semplice Firewall . . . 534
Esercizi Guidati . . . 535
Esercizi Esplorativi . . . 536
Sommario . . . 537
Risposte agli Esercizi Guidati . . . 539
Risposte agli Esercizi Esplorativi . . . 540
110.3 Proteggere i dati con la crittografia . . . 541
110.3 Lezione 1 . . . 543
Introduzione . . . 543
Configurazione e Uso di Base del Client OpenSSH . . . 544
Il Ruolo delle Chiavi Host del Server OpenSSH . . . 549
Tunnel SSH . . . 551
Esercizi Guidati . . . 555
Esercizi Esplorativi . . . 557
Sommario . . . 558
Risposte agli Esercizi Guidati . . . 559
Risposte agli Esercizi Esplorativi . . . 561
110.3 Lezione 2 . . . 562
Introduzione . . . 562
Effettuare la Configurazione di Base, Utilizzare ed Eseguire Attività di Revoca con GnuPG . . . 562
Usare GPG per Criptare, Cecriptare, Firmare e Verificare i File . . . 568
Esercizi Guidati . . . 573
Esercizi Esplorativi . . . 575
Sommario . . . 576
Risposte agli Esercizi Guidati . . . 577
Risposte agli Esercizi Esplorativi . . . 579
Imprint . . . 580



Linux Professional Institute

Argomento 105: Shell e Script di Shell



105.1 Personalizzare e utilizzare l'ambiente di shell

Obiettivi LPI di riferimento

LPIC-1 version 5.0, Exam 102, Objective 105.1

Peso

4

Aree di Conoscenza Chiave

- Impostare le variabili di ambiente (per esempio PATH) al login o quando si genera una nuova shell.
- Scrivere funzioni Bash per sequenze di comandi usate frequentemente.
- Mantenere le directory scheletro per i nuovi account utente.
- Impostare il percorso di ricerca dei comandi con la directory corretta.

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

- `.`
- `source`
- `/etc/bash.bashrc`
- `/etc/profile`
- `env`
- `export`
- `set`
- `unset`
- `~/.bash_profile`
- `~/.bash_login`

- `~/.profile`
- `~/.bashrc`
- `~/.bash_logout`
- `function`
- `alias`



105.1 Lezione 1

Certificazione:	LPIC-1
Versione:	5.0
Argomento:	105 Shell e Script di Shell
Obiettivo:	105.1 Personalizzare e utilizzare l'ambiente di Shell
Lezione:	1 di 3

Introduzione

La shell è probabilmente lo strumento più potente in un sistema Linux e può essere definita come un'interfaccia tra l'utente e il kernel del sistema operativo che interpreta i comandi inseriti dall'utente. Tutti gli amministratori di sistema devono essere esperti nell'uso della shell. Come ormai sappiamo, la *Bourne Again Shell* (*Bash*) è la shell *de facto* per la stragrande maggioranza delle distribuzioni Linux.

Una volta avviata, la prima cosa che fa Bash - o qualsiasi altra shell - è eseguire una serie di script di avvio. Questi script personalizzano l'ambiente di sessione. Sono disponibili sia script a livello di sistema sia specifici dell'utente. Possiamo inserire le nostre preferenze o impostazioni personali che meglio si adattano alle esigenze dei nostri utenti in questi script sotto forma di variabili, alias e funzioni.

La serie esatta di file di avvio dipende da un parametro molto importante: il tipo di shell. Diamo uno sguardo alle varie shell esistenti.

Tipi di shell: Interattiva vs Non-interattiva ; Login vs Senza-Login

Per cominciare, chiariamo i concetti di *interattivo* e *accesso* nel contesto delle shell:

Shell Interattive / Non-interattive

Ci si riferisce all'interazione che avviene tra l'utente e la shell: l'utente fornisce l'input digitando i comandi nel terminale utilizzando la tastiera; la shell fornisce l'output visualizzando messaggi sullo schermo.

Shell con Login / Senza-Login

Ci si riferisce all'evento in cui un utente accede a un sistema informatico fornendo le sue credenziali, come nome utente e password.

Sia le shell interattive sia quelle non-interattive possono essere con login o senza-login e ogni possibile combinazione di questi tipi ha i suoi usi specifici.

Le shell di Login interattive vengono eseguite quando gli utenti accedono al sistema e vengono utilizzate per personalizzare le configurazioni degli utenti in base alle loro esigenze. Un buon esempio di questo tipo di shell sarebbe quella di un gruppo di utenti appartenenti allo stesso dipartimento che necessitano di un particolare set di variabili nelle loro sessioni.

Con *shell interattive non di login* si fa riferimento a qualsiasi altra shell aperta dall'utente dopo aver effettuato l'accesso al sistema. Gli utenti utilizzano queste shell durante le sessioni per eseguire attività di manutenzione e amministrative come l'impostazione di variabili, l'ora, la copia di file, la scrittura di script, ecc.

D'altra parte, le *shell non interattive* non richiedono alcun tipo di interazione umana. Pertanto, queste shell non chiedono input all'utente e il loro output, se presente, nella maggior parte dei casi viene scritto in un registro.

Le *shell di login non interattive* sono piuttosto rare e poco pratiche. I loro usi sono praticamente inesistenti e ne parleremo solo per motivi di comprensione del comportamento della shell. Alcuni strani esempi includono la forzatura di uno script da eseguire da una shell di login con `/bin/bash --login <some_script>` o il *piping* dell'output standard (*stdout*) di un comando nello standard input (*stdin*) di una connessione ssh:

```
<some_command> | ssh <some_user>@<some_server>
```

Per quanto riguarda la *shell non interattiva non di login* non c'è né interazione né login per conto dell'utente, quindi ci riferiamo qui all'uso di script automatizzati. Questi script vengono utilizzati principalmente per eseguire attività amministrative e di manutenzione ripetitive come quelle

incluse in *cronjob*. In questi casi, `bash` non legge alcun file di avvio.

Apertura di un Terminale

Quando ci troviamo in un ambiente desktop possiamo aprire un'applicazione terminale o passare a una delle console di sistema. Pertanto, una nuova shell è una shell `pts` quando viene aperta da un emulatore di terminale nella GUI o una shell `tty` quando viene eseguita da una console di sistema. Nel primo caso non abbiamo a che fare con un terminale ma con un emulatore di terminale. Come parte delle sessioni grafiche, gli emulatori di terminale come *gnome-terminal* o *konsole* sono molto ricchi di funzionalità e facili da usare rispetto ai terminali con interfaccia utente basata su testo. Emulatori di terminale meno ricchi di funzionalità includono, tra gli altri, *XTerm* e *sakura*.

Usando le combinazioni di `Ctrl` + `Alt` + `F1`-`F6` possiamo andare ai login della console che aprono una shell di login interattiva basata su testo. `Ctrl` + `Alt` + `F7` ci riporterà nelle sessione desktop (GUI).

NOTE `tty` sta per teletypewritter; `pts` sta per *pseudo terminal slave*. Per ulteriori informazioni: `man tty` e `man pts`.

Avviare una Shell con `bash`

Dopo il login, digitare `bash` in un terminale per aprire una nuova shell. Tecnicamente, questa shell è un processo figlio della shell corrente.

Durante l'avvio del processo figlio `bash`, possiamo specificare vari parametri per definire quale tipo di shell vogliamo avviare. Ecco alcune importanti opzioni di invocazione di `bash`:

`bash -l` or `bash --login`
: invocherà una shell di login.

`bash -i`
: invocherà una shell interattiva.

`bash --noprofile`
: insieme alle shell di login, ignoreranno sia il file di avvio a livello di sistema `/etc/profile` che i file di avvio a livello utente `~/.bash_profile`, `~/.bash_login` e `~ /.profile`.

`bash --norc`
: ignorerà sia il file di avvio a livello di sistema `/ etc / bash.bashrc` che il file di avvio a livello utente `~/.bashrc`.

`bash --rcfile <file>`

ignorerà sia il file di avvio a livello di sistema `/etc/bash.bashrc` che il file di avvio a livello utente `~/.bashrc`.

Discuteremo i vari file di avvio in seguito.

Avvio di Shell con `su` e `sudo`

Attraverso l'uso di questi due programmi simili possiamo ottenere specifici tipi di shell:

`su`

Cambia l'ID utente o diventa superutente (`root`). Con questo comando possiamo richiamare sia shell di login che shell non di login:

- `su - user2`, `su -l user2` o `su --login user2` avvierà una shell di login interattiva come `user2`.
- `su user2` avvierà una shell interattiva non di login come `user2`.
- `su - root` o `su -` avvieranno una shell di login interattiva come `root`.
- `su root` o `su` avvieranno una shell interattiva non di login come `root`.

`sudo`

Consente di eseguire i comandi come un altro utente (incluso il supersuser). Poiché questo comando viene utilizzato principalmente per ottenere temporaneamente i privilegi di root, l'utente che lo utilizza deve trovarsi nel file `sudoers`. Per aggiungere utenti a `sudoers` dobbiamo diventare `root` ed eseguire:

```
root@debian:~# usermod -aG sudo user2
```

Proprio come `su`, `sudo` ci permette di invocare sia shell di login che shell non di login:

- `sudo su - user2`, `sudo su -l user2` o `sudo su --login utente2` avvieranno una shell di login interattiva come `user2`.
- `sudo su user2` avvierà una shell interattiva non di login come `user2`.
- `sudo -u user2 -s` avvierà una shell interattiva non di login come `user2`.
- `sudo su - root` o `sudo su -` avvierà una shell di login interattiva come `root`.
- `sudo -i` avvierà una shell di login interattiva come `root`.
- `sudo -i <some_command>` avvierà una shell di login interattiva come `root`, eseguirà il comando e tornerà all'utente originale.

- `sudo su root ` o `sudo su` avvieranno una shell interattiva non di login come `root`.
- `sudo -s` o `sudo -u root -s` avvierà una shell non di login come `root`.

Quando si usa `su` o `sudo` è importante considerare il nostro caso particolare per l'avvio di una nuova shell: abbiamo bisogno dell'ambiente dell'utente di destinazione, o no? Se è così, useremmo le opzioni che invocano le shell di login; in caso contrario, quelli che invocano shell non di login.

Che Tipi di Shell Abbiamo?

Per scoprire su quale tipo di shell stiamo lavorando, possiamo digitare `echo $0` nel terminale e ottenere il seguente output:

Login interattivo

`-bash` o `-su`

Non-login interattivo

`bash` o `/bin/bash`

Non-interattivo non-login (scripts)

`<nome_dello_script>`

Quante Shell abbiamo?

Per vedere quante shell `bash` abbiamo attive e funzionanti nel sistema, possiamo usare il comando `ps aux | grep bash`:

```
user2@debian:~$ ps aux | grep bash
user2       5270  0.1  0.1  25532  5664 pts/0    Ss   23:03   0:00 bash
user2       5411  0.3  0.1  25608  5268 tty1     S+   23:03   0:00 -bash
user2       5452  0.0  0.0  16760   940 pts/0    S+   23:04   0:00 grep --color=auto bash
```

`user2` in `debian` ha effettuato l'accesso a una sessione GUI (o X Window System) e ha aperto *gnome-terminal*, quindi ha premuto `Ctrl` + `Alt` + `F1` per entrare in una sessione di terminale `tty`. Infine, è tornato alla sessione della GUI premendo `Ctrl` + `Alt` + `F7` e digitato il comando `ps aux | grep bash`. Pertanto, l'output mostra una shell interattiva non di login tramite l'emulatore di terminale (`pts/0`) e una shell di login interattiva tramite l'appropriato terminale basato su testo (`tty1`). Nota anche come l'ultimo campo di ogni riga (il comando) sia `bash` per la prima e `-bash` per la seconda.

Da Dove le Shell Ottengono la Loro Configurazione: File di Avvio

Bene, ora che conosciamo i tipi di shell che possiamo trovare in un sistema Linux, è giunto il momento di vedere quali file di avvio vengano eseguiti da quale shell. Notare che gli script globali o globali di sistema sono posti nella directory `/etc/`, mentre quelli locali o a livello utente si trovano nella home dell'utente (~). Inoltre, quando è presente più di un file da cercare, una volta trovato ed eseguito gli altri vengono ignorati. Esplora e studia questi file tu stesso con il tuo editor di testo preferito o digitando `less <startup_file>`.

NOTE I file di avvio possono essere suddivisi in *specifici di Bash* (quelli limitati solo alle configurazioni e ai comandi di `bash`) e *generali* (relativi alla maggior parte delle shell).

Shell Interattive di Login

Livello Globale

`/etc/profile`

Questo è il file `.profile` a livello di sistema per la shell Bourne e per le shell compatibili con Bourne (`bash` inclusa). Attraverso una serie di istruzioni `if`, questo file imposta di conseguenza un numero di variabili come `PATH` e `PS1` così come il reperimento - qualora esistano - sia del file `/etc/bash.bashrc` sia di altri file all'interno della directory `/etc/profile.d`.

`/etc/profile.d/*`

Questa directory può contenere script che vengono eseguiti da `/etc/profile`.

Livello Locale

`~/.bash_profile`

Questo file specifico di Bash viene utilizzato per configurare l'ambiente utente. Può anche essere usato per eseguire sia `~/.bash_login` sia `~/.profile`.

`~/.bash_login`

Anch'esso specifico per Bash, questo file verrà eseguito solo se non c'è un file `~/.bash_profile`. Il suo nome suggerisce che dovrebbe essere usato per eseguire i comandi necessari all'accesso.

`~/.profile`

Questo file non è specifico di Bash e viene preso in considerazione solo se non esistono né `~/.bash_profile` né `~/.bash_login`, il che è generalmente la norma. Quindi, lo scopo principale di `~/.profile` è quello di controllare se una shell Bash è in esecuzione e, in tal caso,

di cercare se esiste `~/.bashrc`. Di solito imposta la variabile `PATH` in modo che includa la directory privata `~/bin` dell'utente, se esiste.

`~/.bash_logout`

Se esiste, questo file specifico di Bash esegue alcune operazioni di pulizia quando si esce dalla shell. Ciò può essere utile in casi come quelli dove si attivano sessioni remote.

Esplorazione dei File di Configurazione di una Shell di Login Interattiva

Mostriamo alcuni di questi file in azione modificando `/etc/profile` e `/home/user2/.profile`. Aggiungeremo a ciascuno una riga per ricordarci il file in esecuzione:

```
root@debian:~# echo 'echo Hello from /etc/profile' >> /etc/profile
root@debian:~# echo 'echo Hello from ~/.profile' >> ~/.profile
```

NOTE Due operatori di reindirizzamento `>>` aggiungono l'output di un comando in un file esistente, senza sovrascriverlo. Se il file non esiste, tuttavia, verrà creato.

Quindi, attraverso l'output dei rispettivi comandi `echo` sapremo quando ciascuno di questi file viene letto ed eseguito. Per dimostrarlo, vediamo cosa succede quando `user2` accede tramite `ssh` da un'altra macchina:

```
user2@debian:~$ ssh user2@192.168.1.6
user2@192.168.1.6's password:
Linux debian 4.9.0-8-amd64 #1 SMP Debian 4.9.130-2 (2018-10-27) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Tue Nov 27 19:57:19 2018 from 192.168.1.10
Hello from /etc/profile
Hello from /home/user2/.profile
```

Come mostrano le ultime due righe, ha funzionato. Inoltre, nota tre cose:

- Il file globale è stato eseguito per primo.
- Non c'erano file `.bash_profile` o `.bash_login` nella directory home di `user2`.

- La tilde (`~`) espansa al percorso assoluto del file (`/home/utente2/.profile`).

Shell Interattiva Non di Login

Livello Globale

`/etc/bash.bashrc`

Questo è il file `.bashrc` a livello di sistema per le shell interattive `bash`. Attraverso la sua esecuzione `bash` si assicura che venga eseguito in modo interattivo, controlla la dimensione della finestra dopo ogni comando (aggiornando i valori di `LINES` e `COLUMNS` se necessario) e imposta alcune variabili.

Livello Locale

`~/.bashrc`

Oltre a svolgere attività simili a quelle descritte per `/etc/bash.bashrc` a livello utente (come controllare la dimensione della finestra o se viene eseguito in modo interattivo), questo file specifico di Bash di solito imposta alcune variabili di cronologia e esegue `~/.bash_aliases` se esiste. A parte ciò, questo file viene normalmente utilizzato per memorizzare gli alias e le funzioni specifich degli utenti.

Allo stesso modo, vale anche la pena notare che `~/.bashrc` viene letto se `bash` rileva che `<stdin>` è una connessione di rete (come nel caso della connessione *Secure Shell* (SSH) nell'esempio sopra) .

Esplorazione dei File di Configurazione della Shell Interattiva non di Login

Ora modifichiamo `/etc/bash.bashrc` e `/home/user2/.bashrc`:

```
root@debian:~# echo 'echo Hello from /etc/bash.bashrc' >> /etc/bash.bashrc
root@debian:~# echo 'echo Hello from ~/.bashrc' >> ~/.bashrc
```

E questo è ciò che accade quando `user2` avvia una nuova shell:

```
user2@debian:~$ bash
Hello from /etc/bash.bashrc
Hello from /home/user2/.bashrc
```

Anche in questo caso, i due file sono stati letti ed eseguiti.

WARNING Ricorda, a causa dell'ordine in cui vengono eseguiti i file, i file locali hanno la precedenza su quelli globali.

Shell Non-Interattive di Login

Una shell non interattiva con le opzioni `-l` o `--login` è costretta a comportarsi come una shell di login e quindi i file di avvio da eseguire saranno gli stessi delle shell di login interattive.

Per dimostrarlo, scriviamo un semplice script e rendiamolo eseguibile. Non includeremo alcun *shebang* perché invocheremo l'eseguibile *bash* (`/bin/bash` con l'opzione di login) dalla riga di comando.

1. Creiamo lo script `test.sh` contenente la riga `echo 'Hello from a script'` in modo da poter provare che lo script viene eseguito correttamente:

```
user2@debian:~$ echo "echo 'Hello from a script'" > test.sh
```

2. Rendiamo eseguibile il nostro script:

```
user2@debian:~$ chmod +x ./test.sh
```

3. Infine, invochiamo `bash` con l'opzione `-l` per eseguire lo script:

```
user2@debian:~$ bash -l ./test.sh
Hello from /etc/profile
Hello from /home/user2/.profile
Hello from a script
```

   Funziona! Prima di eseguire lo script, è stato eseguito il login e sono stati eseguiti sia `/etc/profile` che `~/.profile`.

NOTE Impareremo a conoscere *shebang* e tutti gli altri aspetti dello scripting di shell nelle prossime lezioni.

Facciamo ora lo standard output (*stdout*) del comando `echo` nello standard input (*stdin*) di una connessione `ssh` per mezzo di una pipe (`|`):

```
user2@debian:~$ echo "Hello-from-a-noninteractive-login-shell" | ssh user2@192.168.1.6
Pseudo-terminal will not be allocated because stdin is not a terminal.
user2@192.168.1.6's password:
```

```
Linux debian 4.9.0-8-amd64 #1 SMP Debian 4.9.130-2 (2018-10-27) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Hello from /etc/profile
Hello from /home/user2/.profile
-bash: line 1: Hello-from-a-noninteractive-login-shell: command not found
```

Di nuovo, vengono eseguiti `/etc/profile` e `~/.profile`. Oltre a questo, la prima e l'ultima riga dell'output sono abbastanza indicative per quanto riguarda il comportamento della shell.

Shell Non-Interattiva non di Login

Gli script non leggono nessuno dei file sopra elencati ma cercano la variabile d'ambiente `BASH_ENV`, espanderne il valore se necessario e usarla come nome di un file di avvio per leggere ed eseguire comandi. Impareremo di più sulle *variabili d'ambiente* nella prossima lezione.

Come accennato in precedenza, tipicamente `/etc/profile` e `~/.profile` si assicurano che sia `/etc/bash.bashrc` sia `~/.bashrc` vengano eseguiti dopo un login riuscito. L'output del seguente comando mostra questo fenomeno:

```
root@debian:~# su - user2
Hello from /etc/bash.bashrc
Hello from /etc/profile
Hello from /home/user2/.bashrc
Hello from /home/user2/.profile
```

Tenendo presente le righe precedentemente aggiunte agli script di avvio — e invocando una shell di login interattiva a livello utente con `su - user2` — le quattro righe di output possono essere spiegate come segue:

1. `Hello from /etc/bash.bashrc` indica che `/etc/profile` ha eseguito `/etc/bash.bashrc`.
2. `Hello from /etc/profile` indica che `/etc/profile` è stato completamente letto ed eseguito.
3. `Hello from /home/user2/.bashrc` indica che `~/.profile` ha eseguito `~/.bashrc`.
4. `Hello from /home/user2/.profile` indica che `~/.profile` è stato completamente letto ed

eseguito.

Nota come con `su - <username>` (anche `su -l <username>` e `su --login <username>`) garantiamo l'invocazione di una shell di login, mentre `su <username>` avrebbe solo invocato `/etc/bash.bashrc` e `~/.bashrc`.

Sourcing di File

Nelle sezioni precedenti abbiamo discusso del fatto che alcuni script di avvio includono o eseguono altri script. Questo meccanismo è chiamato "sourcing" ed è spiegato in questa sezione.

Sourcing di File con `.`

Il punto (`.`) si trova normalmente nei file di avvio.

Nel file `.profile` del nostro server Debian possiamo trovare, per esempio, il seguente blocco:

```
# include .bashrc if it exists
if [ -f "$HOME/.bashrc" ]; then
. "$HOME/.bashrc"
fi
```

Abbiamo già visto come l'esecuzione di uno script possa portare a quella di un altro. Pertanto, l'istruzione `if` garantisce che il file `$HOME/.bashrc` — se esiste (`-f`) — verrà prelevato (cioè letto ed eseguito) al login:

```
. "$HOME/.bashrc"
```

NOTE Come impareremo nella prossima lezione, `$HOME` è una variabile d'ambiente che si espande nel percorso assoluto della directory home dell'utente.

Inoltre, possiamo usare il carattere `.` ogni volta che abbiamo modificato un file di avvio e vogliamo rendere effettive le modifiche senza un riavvio. Per esempio possiamo:

- aggiungere un alias a `~/.bashrc`:

```
user2@debian:~$ echo "alias hi='echo We salute you.'" >> ~/.bashrc
```

WARNING Quando si invia l'output di un comando in un file, ricordarsi di non confondere *accodare* (`>>`) con *sovrascrivere* (`>`).

- visualizza l'ultima riga di `~/.bashrc` per controllare che tutto sia andato bene:

```
user2@debian:~$ tail -n 1 !$
tail -n 1 ~/.bashrc
alias hi='echo We salute you.'
```

NOTE `!$` si espande all'ultimo argomento del comando precedente, nel nostro caso: `~/.bashrc`.

- richiama (*source*) il file a mano:

```
user2@debian:~$ . ~/.bashrc
```

- e richiama l'alias per dimostrare che funziona:

```
user2@debian:~$ hi
We salute you.
```

NOTE Fare riferimento alla lezione successiva per informazioni su *alias* e *variabili*.

Richiamare (*Sourcing*) i file con `source`

Il comando `source` è equivalente di `.`. Quindi per richiamare `~/.bashrc` possiamo anche fare in questo modo:

```
user2@debian:~$ source ~/.bashrc
```

L'Origine dei File di Avvio della Shell: `SKEL`

`SKEL` è una variabile il cui valore è il percorso assoluto della directory `skel`. Questa directory funge da modello per la struttura del file system delle directory home degli utenti. Include i file che verranno ereditati da qualsiasi nuovo account utente creato (inclusi, ovviamente, i file di configurazione per le shell). `SKEL` e altre variabili correlate sono memorizzate in `/etc/adduser.conf`, che è il file di configurazione per `adduser`:

```
user2@debian:~$ grep SKEL /etc/adduser.conf
# The SKEL variable specifies the directory containing "skeletal" user
SKEL=/etc/skel
# If SKEL_IGNORE_REGEX is set, adduser will ignore files matching this
```

```
SKEL_IGNORE_REGEX="dpkg-(old|new|dist|save)"
```

`SKEL` è impostato su `/etc/skel`; quindi, gli script di avvio che configurano le nostre shell si trovano qui:

```
user2@debian:~$ ls -a /etc/skel/
.  ..  .bash_logout  .bashrc  .profile
```

WARNING Ricorda, i file che iniziano con `.` Sono nascosti, quindi dobbiamo usare `ls -a` per vederli quando si elencano i contenuti della directory.

Creiamo ora una directory in `/etc/skel` in cui memorizzare i file per i nuovi utenti:

1. Come `root` ci spostiamo in `/etc/skel`:

```
root@debian:~# cd /etc/skel/
root@debian:/etc/skel#
```

2. Visualizziamo il contenuto:

```
root@debian:/etc/skel# ls -a
.  ..  .bash_logout  .bashrc  .profile
```

3. Creiamo la nostra directory e controlliamo che tutto sia andato come previsto:

```
root@debian:/etc/skel# mkdir my_personal_scripts
root@debian:/etc/skel# ls -a
.  ..  .bash_logout  .bashrc  my_personal_scripts  .profile
```

4. Adesso cancelliamo `user2` insieme alla sua directory `home`:

```
root@debian:~# deluser --remove-home user2
Looking for files to backup/remove ...
Removing files ...
Removing user `user2' ...
Warning: group `user2' has no more members.
Done.
```

5. Aggiungiamo di nuovo `user2` in modo che ottenga una nuova directory home:

```
root@debian:~# adduser user2
Adding user `user2' ...
Adding new group `user2' (1001) ...
Adding new user `user2' (1001) with group `user2' ...
Creating home directory `/home/user2' ...
Copying files from `/etc/skel' ...
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
Changing the user information for user2
Enter the new value, or press ENTER for the default
    Full Name []:
    Room Number []:
    Work Phone []:
    Home Phone []:
    Other []:
Is the information correct? [Y/n] y
```

6. Infine, accediamo come `user2` ed elenchiamo tutti i file in `/home/user2` per vedere se tutto è andato come previsto:

```
root@debian:~# su - user2
user2@debian:~$ pwd
/home/user2
user2@debian:~$ ls -a
.  ..  .bash_history  .bash_logout  .bashrc  my_personal_scripts  .profile
```

Tutto come previsto.

Esercizi Guidati

1. Studia come sono state avviate le shell sotto la colonna “Shell Avviata con...” e completa con le informazioni richieste:

Shell Avviata con...	**Interactive?**	**Login?**	**Risultato di `echo $0`**
`sudo ssh user2@machine2`			
`Ctrl` + `Alt` + `F2`			
`su - user2`			
gnome-terminal			
Un utente normale utilizza *konsole* per avviare un’istanza di *sakura*			
Uno script chiamato `test.sh` contenente il comando `echo $0`			

2. Scrivi i comandi `su` e `sudo` per avviare la shell specificata:

   Shell di login interattiva come `user2`

   `su`:

   `sudo`:

   Shell di login interattiva come `root`

   `su`:

   `sudo`:

   Shell non di login interattiva come `root`

   `su`:

   `sudo`:

   Shell non di login interattiva come `user2`

   `su`:

`sudo`:

3. Quale file di avvio viene letto quando viene avviata la shell presente in "Tipo di Shell"?

Tipo di Shell	`/etc/profile`	`/etc/bash.bashrc`	`~/.profile`	`~/.bashrc`
Shell di login interattiva come `user2`				
Shell di login interattiva come `root`				
Shell non di login interattiva come `root`				
Shell non di login interattiva come `user2`				

Esercizi Esplorativi

1. In Bash possiamo scrivere una semplice funzione `Hello world!` includendo il seguente codice in un file vuoto:

```
function hello() {
     echo "Hello world!"
}
```

   - Che cosa dovremmo fare dopo per rendere la funzione disponibile alla shell?

   - Una volta che è disponibile per la shell corrente, come la invocheresti?

   - Per automatizzare le cose, in quale file metteresti la funzione e la sua invocazione in modo che venga eseguita quando `user2` apre un terminale da una sessione X Window? Che tipo di Shell è?

   - In quale file metteresti la funzione e la sua invocazione in modo che venga eseguita quando `root` lancia una nuova Shell interattiva indipendentemente dal fatto che sia login o meno?

2. Dai un'occhiata al seguente script di base `bash` `Hello world!` :

```
#!/bin/bash

#hello_world: a simple bash script to discuss interaction in scripts.

echo "Hello world!"
```

   - Supponiamo di rendere eseguibile lo script e di eseguirlo. Sarebbe uno script interattivo? Perché?

   - Che cosa rende interattivo uno script?

3. Immagina di aver cambiato i valori di alcune variabili in `~/.bashrc` e vuoi che tali modifiche

abbiano effetto senza un riavvio. Dalla tua home directory, come puoi ottenerlo in due modi diversi?

4. John ha appena avviato una sessione X Window su un server Linux. Apre un emulatore di terminale per svolgere alcune attività amministrative ma, sorprendentemente, la sessione si blocca e ha bisogno di aprire una shell di testo.
   - Come può aprire quella shell `tty`?
   - Quali file di avvio verranno richiamati?
5. Linda è un utente di un server Linux. Chiede gentilmente all'amministratore di avere un file `~/.bash_login` in modo che possa avere l'ora e la data stampate sullo schermo quando accede. Ad altri utenti piace l'idea e seguono l'esempio. L'amministratore ha difficoltà a creare il file per tutti gli altri utenti sul server, quindi decide di aggiungere una nuova policy e di creare `~/.bash_login` per tutti i potenziali nuovi utenti. Come può l'amministratore svolgere tale compito?

Sommario

In questa lezione abbiamo imparato:

- Le shell impostano l'ambiente degli utenti in un sistema Linux.
- *Bash* è la shell numero uno nelle distribuzioni GNU/Linux.
- Il primo lavoro che una shell esegue è quello di leggere ed eseguire uno o più file di avvio.
- I concetti di *interattivo* e *login* relativi alle shell.
- Come avviare diversi tipi di shell con `bash`, `su`, `sudo` e kbd: [Ctrl+Alt+F1]-F6.
- Come controllare il tipo di shell con `echo $0`.
- I file di avvio locali `~/.bash_profile`, `~/.profile`, `~/.bash_login`, `~/.bash_logout` e `~/.bashrc`.
- I file di avvio globali `/etc/profile`, `/etc/profile.d/*`, `/etc/bash.bashrc`.
- I file locali hanno la precedenza su quelli globali.
- Come reindirizzare l'output di un comando con `>` (sovrascrivi) e `>>` (accoda).
- Il significato della directory `skel`.
- Come richiamare i file.

Comandi utilizzati in questa lezione:

`bash`
Avvia una nuova Shell.

`su`
Avvia una nuova Shell.

`sudo`
Avvia una nuova Shell.

`usermod`
Modifica un account utente.

`echo`
Visualizza una riga di testo.

`ps`

Mostra un'istantanea dei processi in corso.

`less`

Un paginatore di file lunghi.

`ssh`

Avvia una connessione Open SSH (remotamente).

`chmod`

Modifica i bit di modalità di un file, per esempio, renderlo eseguibile.

`grep`

Stampa le linee che corrispondono a un *pattern*.

`ls`

Elenca il contenuto della directory.

`cd`

Cambia directory.

`mkdir`

Crea una directory.

`deluser`

Elimina un utente.

`adduser`

Aggiunge un utente.

`.`

Richiama (*Source*) a file.

`source`

Richiama (*Source*) a file.

`tail`

Visualizza l'ultima parte di un file.

Risposte agli Esercizi Guidati

1. Studia come sono state avviate le shell sotto la colonna “Shell Avviata con...” e completa con le informazioni richieste:

Shell Avviata con...	**Interactive?**	**Login?**	**Risultato di `echo $0`**
`sudo ssh user2@machine2`	Sì	Sì	`-bash`
`Ctrl` + `Alt` + `F2`	Sì	Sì	`-bash`
`su - user2`	Sì	Sì	`-bash`
gnome-terminal	Sì	No	`bash`
Un utente normale utilizza *konsole* per avviare un’istanza di *sakura*	Sì	No	`/bin/bash`
Uno script chiamato `test.sh` contenente il comando `echo $0`	No	No	`./test.sh`

2. Scrivi i comandi `su` e `sudo` per avviare la shell specificata:

   Shell di login interattiva come `user2`

   `su`

   `su - user2`, `su -l user2` o `su --login user2`

   `sudo`

   `sudo su - user2`, `sudo su -l user2` o `sudo su --login user2`

   Shell di login interattiva come `root`

   `su`

   `su - root` or `su -`

   `sudo`

   `sudo su - root`, `sudo su -` o `sudo -i`

Shell non di login interattiva come `root`

`su`

`su root` o `su`

`sudo`

`sudo su root`, `sudo su`, `sudo -s` o `sudo -u root -s`

Shell non di login interattiva come `user2`

`su`

`su user2`

`sudo`

`sudo su user2` o `sudo -u user2 -s`

3. Quale file di avvio viene letto quando viene avviata la shell presente in "Tipo di Shell"?

Tipo di Shell	`/etc/profile`	`/etc/bash.bashrc`	`~/.profile`	`~/.bashrc`
Shell di login interattiva come `user2`	Sì	Sì	Sì	Sì
Shell di login interattiva come `root`	Sì	Sì	No	No
Shell non di login interattiva come `root`	No	Sì	No	No
Shell non di login interattiva come `user2`	No	Sì	No	Sì

Risposte agli Esercizi Esplorativi

1. In Bash possiamo scrivere una semplice funzione `Hello world!` includendo il seguente codice in un file vuoto:

```
function hello() {
     echo "Hello world!"
}
```

   - Che cosa dovremmo fare dopo per rendere la funzione disponibile alla shell?

     Per rendere la funzione disponibile alla shell corrente, dobbiamo generare un file che la includa.

   - Una volta che è disponibile per la shell corrente, come la invocheresti?

     La invocheremmo digitando il suo nome nel terminale.

   - Per automatizzare le cose, in quale file metteresti la funzione e la sua invocazione in modo che venga eseguita quando `user2` apre un terminale da una sessione X Window? Che tipo di Shell è?

     Il miglior file da utilizzare è `/home/user2/.bashrc`. La shell invocata sarebbe una shell interattiva non di login.

   - In quale file metteresti la funzione e la sua invocazione in modo che venga eseguita quando `root` lancia una nuova Shell interattiva indipendentemente dal fatto che sia login o meno?

     In `/etc/bash.bashrc` poiché questo file viene eseguito per tutte le shell interattive, indipendentemente dal fatto che siano di login o meno.

2. Dai un'occhiata al seguente script di base `bash` `Hello world!` :

```
#!/bin/bash

#hello_world: a simple bash script to discuss interaction in scripts.

echo "Hello world!"
```

   - Supponiamo di rendere eseguibile lo script e di eseguirlo. Sarebbe uno script interattivo? Perché?

No, poiché non vi è alcuna interazione umana e nessun comando viene digitato dall'utente.

- Che cosa rende interattivo uno script?

  Il fatto che richieda l'input dell'utente.

3. Immagina di aver cambiato i valori di alcune variabili in `~/.bashrc` e vuoi che tali modifiche abbiano effetto senza un riavvio. Dalla tua home directory, come puoi ottenerlo in due modi diversi?

```
$ source .bashrc
```

o

```
$ . .bashrc
```

4. John ha appena avviato una sessione X Window su un server Linux. Apre un emulatore di terminale per svolgere alcune attività amministrative ma, sorprendentemente, la sessione si blocca e ha bisogno di aprire una shell di testo.

- Come può aprire quella shell `tty`?

  Potrebbe farlo premendo `Ctrl` + `Alt` + `F1`-`F6` per entrare in una delle sei shell tty.

- Quali file di avvio verranno richiamati?

  `/etc/profile`
  `/home/john/.profile`

5. Linda è un utente di un server Linux. Chiede gentilmente all'amministratore di avere un file `~/.bash_login` in modo che possa avere l'ora e la data stampate sullo schermo quando accede. Ad altri utenti piace l'idea e seguono l'esempio. L'amministratore ha difficoltà a creare il file per tutti gli altri utenti sul server, quindi decide di aggiungere una nuova policy e di creare `~/.bash_login` per tutti i potenziali nuovi utenti. Come può l'amministratore svolgere tale compito?

   Potrebbe ottenerlo inserendo `.bash_login` nella directory `/etc/skel`.



105.1 Lezione 2

Certificazione:	LPIC-1
Versione:	5.0
Argomento:	105 Shell e Script di Shell
Obiettivo:	105.1 Personalizzare e utilizzare l'ambiente di Shell
Lezione:	2 di 3

Introduzione

Pensa a una variabile come a una scatola immaginaria in cui collocare temporaneamente un'informazione. Come con i suoi script di inizializzazione, Bash classifica le variabili come *di shell/locali* (quelle che vivono solo entro i limiti della shell in cui sono state create) o *di ambiente/globali* (quelle ereditate dalle shell secondarie e/o dai processi). Nella lezione precedente abbiamo esaminato la shell e i loro script di configurazione/inizializzazione. È ora importante sottolineare che la potenza di questi file di avvio risiede nel fatto che ci consentono di utilizzare variabili - così come alias e funzioni - che ci aiutano a creare e personalizzare l'ambiente shell di nostra scelta.

Variabili: Assegnazione e Riferimento

Una variabile può essere definita come un nome contenente un valore.

In Bash, dare un valore a un nome è chiamato *assegnazione variabile* ed è il modo in cui creiamo o impostiamo le variabili. Il processo di accesso al valore contenuto nel nome è chiamato *riferimento variabile*.

La sintassi per l'assegnazione delle variabili è:

```
<variable_name>=<variable_value>
```

Per esempio:

```
$ distro=zorinos
```

La variabile `distro` è uguale a `zorinos`, cioè c'è una porzione di memoria che contiene il valore `zorinos`, con `distro` che è il puntatore a essa.

Da notare, tuttavia, che non può esserci spazio su entrambi i lati del segno di uguale quando si assegna una variabile:

```
$ distro =zorinos
-bash: distro: command not found
$ distro= zorinos
-bash: zorinos: command not found
```

A causa del nostro errore, Bash ha letto `distro` e `zorinos` come comandi.

Per fare riferimento a una variabile (ovvero, per verificarne il valore) usiamo il comando `echo` facendo precedere al nome della variabile un segno `$`:

```
$ echo $distro
zorinos
```

Nomi di Variabili

Quando si sceglie il nome delle variabili, ci sono alcune regole che dobbiamo prendere in considerazione.

Il nome di una variabile può contenere lettere (`a-z`, `A-Z`), numeri (`0-9`) e trattini bassi (`_`):

```
$ distro=zorinos
$ echo $distro
zorinos
$ DISTRO=zorinos
$ echo $DISTRO
```

```
zorinos
$ distro_1=zorinos
$ echo $distro_1
zorinos
$ _distro=zorinos
$ echo $_distro
zorinos
```

Il nome di una variabile non può iniziare con un numero, altrimenti Bash si confonde:

```
$ 1distro=zorinos
-bash: 1distro=zorinos: command not found
```

Non può contenere spazi (nemmeno usando le virgolette); per convenzione, vengono utilizzati i trattini bassi:

```
$ "my distro"=zorinos
-bash: my: command not found
$ my_distro=zorinos
$ echo $my_distro
zorinos
```

Valori di Variabili

Per quanto riguarda il riferimento o il valore delle variabili è anche importante considerare una serie di regole.

Le variabili possono contenere qualsiasi carattere alfanumerico (`a-z`, `A-Z`, `0-9`) così come la maggior parte degli altri caratteri (`?`, `!`, `*`, `.`, `/`, ecc.):

```
$ distro=zorin12.4?
$ echo $distro
zorin12.4?
```

I valori delle variabili devono essere racchiusi tra virgolette se contengono spazi singoli:

```
$ distro=zorin 12.4
-bash: 12.4: command not found
$ distro="zorin 12.4"
$ echo $distro
```

```
zorin 12.4
$ distro='zorin 12.4'
$ echo $distro
zorin 12.4
```

I valori delle variabili devono anche essere racchiusi tra virgolette se contengono caratteri come quelli usati per il reindirizzamento (`<`, `>`) o il simbolo di pipe (`|`). L'unica cosa che fa il seguente comando è creare un file vuoto chiamato `zorin`:

```
$ distro=>zorin
$ echo $distro

$ ls zorin
zorin
```

Funziona, però, quando usiamo le virgolette:

```
$ distro=">zorin"
$ echo $distro
>zorin
```

Tuttavia, le virgolette singole e doppie non sono sempre intercambiabili. A seconda di ciò che stiamo facendo con una variabile (assegnazione o riferimento), l'uso delle une o delle altre ha implicazioni e produrrà risultati diversi. Nel contesto dell'assegnazione di variabili le virgolette singole prendono tutti i caratteri del valore della variabile *letteralmente*, mentre le virgolette doppie consentono la sostituzione delle variabili:

```
$ lizard=uromastyx
$ animal='My $lizard'
$ echo $animal
My $lizard
$ animal="My $lizard"
$ echo $animal
My uromastyx
```

D'altra parte, quando si fa riferimento a una variabile il cui valore include alcuni spazi iniziali (o extra) — a volte combinati con asterischi — è obbligatorio utilizzare le virgolette doppie dopo il comando `echo` per evitare *field splitting* e *pathname expansion*:

```
$ lizard="   genus   |   uromastyx"
$ echo $lizard
genus | uromastyx
$ echo "$lizard"
   genus   |   uromastyx
```

Se il riferimento della variabile contiene un punto esclamativo di chiusura, questo deve essere l'ultimo carattere della stringa (altrimenti Bash penserà che ci riferiamo a un evento `history`):

```
$ distro=zorin.?/!os
-bash: !os: event not found
$ distro=zorin.?/!
$ echo $distro
zorin.?/!
```

Qualsiasi carattere di *backslash* deve essere preceduto da un'altro *backslash* . Inoltre, se il *backslash* è l'ultimo carattere della stringa e non ne mettiamo un'altro, Bash interpreterà che vogliamo un'interruzione di riga e ci darà una nuova riga:

```
$ distro=zorinos\
>
$ distro=zorinos\\
$ echo $distro
zorinos\
```

Nelle prossime due sezioni riassumeremo le principali differenze tra le variabili *locali* e di *ambiente*.

Variabili Locali o di Shell

Le variabili locali o di shell esistono *solo* nella shell in cui vengono create. Per convenzione, le variabili locali sono scritte in lettere minuscole.

Creiamo una variabile locale per eseguire alcuni test. Come spiegato sopra, scegliamo un nome di variabile appropriato e lo associamo a un valore appropriato. Per esempio:

```
$ reptile=tortoise
```

Usiamo ora il comando `echo` per fare riferimento alla nostra variabile e controllare che tutto sia

andato come previsto:

```
$ echo $reptile
tortoise
```

In alcuni scenari, come durante la scrittura di script, l'immutabilità può essere una caratteristica interessante delle variabili. Se vogliamo che le nostre variabili siano immutabili, possiamo crearle `readonly`:

```
$ readonly reptile=tortoise
```

Oppure in due parti:

```
$ reptile=tortoise
$ readonly reptile
```

Ora, se proviamo a cambiare il valore di `reptile`, Bash si rifiuterà:

```
$ reptile=lizard
-bash: distro: readonly variable
```

NOTE Per elencare tutte le variabili di sola lettura nella nostra sessione corrente, digita `readonly` o `readonly -p` nel terminale.

Un comando utile quando si ha a che fare con le variabili locali è `set`.

`set` restituisce tutte le variabili e le funzioni della shell attualmente assegnate. Dal momento che possono essere molte righe (provalo tu stesso!), si consiglia di usarlo in combinazione con un paginatore come `less`:

```
$ set | less
BASH=/bin/bash
BASHOPTS=checkwinsize:cmdhist:complete_fullquote:expand_aliases:extglob:extquote:force_figno
re:histappend:interactive_comments:login_shell:progcomp:promptvars:sourcepath
BASH_ALIASES=()
BASH_ARGC=()
BASH_ARGV=()
BASH_CMDS=()
BASH_COMPLETION_COMPAT_DIR=/etc/bash_completion.d
```

```
BASH_LINENO=()
BASH_SOURCE=()
BASH_VERSINFO=([0]="4" [1]="4" [2]="12" [3]="1" [4]="release" [5]="x86_64-pc-linux-gnu")
BASH_VERSION='4.4.12(1)-release'
(...)
```

C'è la nostra variabile `reptile`?

```
$ set | grep reptile
reptile=tortoise
```

Sì, c'è!

Tuttavia, `reptile`, essendo una variabile locale, non verrà trasmessa da alcun processo "figlio" generato dalla shell corrente:

```
$ bash
$ set | grep reptile
$
```

E, naturalmente, non possiamo nemmeno visualizzare il suo valore:

```
$ echo $reptile
$
```

NOTE Digitando il comando `bash` nel terminale apriamo una nuova shell (figlia).

Per annullare l'impostazione di qualsiasi variabile (sia locale che globale), usiamo il comando `unset`:

```
$ echo $reptile
tortoise
$ unset reptile
$ echo $reptile
$
```

NOTE `unset` deve essere seguito dal solo nome della variabile (non preceduto dal simbolo $).

Variabili Globali o di Ambiente

Esistono variabili globali o di ambiente per la shell corrente e per tutti i processi successivi generati da essa. Per convenzione, le variabili d'ambiente sono scritte in lettere maiuscole:

```
$ echo $SHELL
/bin/bash
```

Possiamo passare ricorsivamente il valore di queste variabili ad altre variabili e il valore di queste ultime si espanderà infine a quello delle prime:

```
$ my_shell=$SHELL
$ echo $my_shell
/bin/bash
$ your_shell=$my_shell
$ echo $your_shell
/bin/bash
$ our_shell=$your_shell
$ echo $our_shell
/bin/bash
```

Affinché una variabile di shell locale diventi una variabile di ambiente, è necessario utilizzare il comando `export`:

```
$ export reptile
```

Con `export reptile` abbiamo trasformato la nostra variabile locale in una variabile d'ambiente in modo che le shell secondarie (figlie) possano riconoscerla e usarla:

```
$ bash
$ echo $reptile
tortoise
```

Allo stesso modo, `export` può essere utilizzato per impostare ed esportare una variabile, tutto in una volta:

```
$ export amphibian=frog
```

Ora possiamo aprire una nuova istanza di Bash e fare riferimento con successo alla nuova variabile:

```
$ bash
$ echo $amphibian
frog
```

NOTE Con `export -n <VARIABLE-NAME>` la variabile verrà nuovamente trasformata in una variabile di shell locale.

Il comando `export` ci darà anche un elenco di tutte le variabili d'ambiente esistenti quando digitato da solo (o con l'opzione `-p`):

```
$ export
declare -x HOME="/home/user2"
declare -x LANG="en_GB.UTF-8"
declare -x LOGNAME="user2"
(...)
declare -x PATH="/usr/local/bin:/usr/bin:/bin:/usr/local/games:/usr/games"
declare -x PWD="/home/user2"
declare -x SHELL="/bin/bash"
declare -x SHLVL="1"
declare -x SSH_CLIENT="192.168.1.10 49330 22"
declare -x SSH_CONNECTION="192.168.1.10 49330 192.168.1.7 22"
declare -x SSH_TTY="/dev/pts/0"
declare -x TERM="xterm-256color"
declare -x USER="user2"
declare -x XDG_RUNTIME_DIR="/run/user/1001"
declare -x XDG_SESSION_ID="8"
declare -x reptile="tortoise"
```

NOTE Il comando `declare -x` è equivalente a `export`.

Altri due comandi che possono essere usati per stampare un elenco di tutte le variabili d'ambiente. Sono `env` e `printenv`:

```
$ env
SSH_CONNECTION=192.168.1.10 48678 192.168.1.7 22
LANG=en_GB.UTF-8
XDG_SESSION_ID=3
USER=user2
```

```
PWD=/home/user2
HOME=/home/user2
SSH_CLIENT=192.168.1.10 48678 22
SSH_TTY=/dev/pts/0
MAIL=/var/mail/user2
TERM=xterm-256color
SHELL=/bin/bash
SHLVL=1
LOGNAME=user2
XDG_RUNTIME_DIR=/run/user/1001
PATH=/usr/local/bin:/usr/bin:/bin:/usr/local/games:/usr/games
_=/usr/bin/env
```

Oltre a essere un'alternativa a `env`, possiamo usare `printenv` in un modo simile a come usiamo il comando `echo` per verificare il valore di una variabile:

```
$ echo $PWD
/home/user2
$ printenv PWD
/home/user2
```

Nota, tuttavia, che con `printenv` il nome della variabile non è preceduto da `$`.

NOTE `PWD` memorizza il percorso della directory di lavoro attuale. Apprenderemo in seguito questa e altre variabili d'ambiente comuni.

Esecuzione di un Programma in un Ambiente Modificato

`env` può essere usato per modificare l'ambiente della shell al momento dell'esecuzione di un programma.

Per avviare una nuova sessione Bash con un ambiente il più vuoto possibile — cancellando la maggior parte delle variabili (così come le funzioni e gli alias) — useremo `env` con l'opzione `-i`:

```
$ env -i bash
```

Ora la maggior parte delle nostre variabili di ambiente sono scomparse:

```
$ echo $USER
$
```

E ne rimangono solo poche:

```
$ env
LS_COLORS=
PWD=/home/user2
SHLVL=1
_=/usr/bin/printenv
```

Possiamo anche usare `env` per impostare una particolare variabile per un particolare programma.

Nella nostra lezione precedente, parlando di *shell non di login non interattive*, abbiamo visto come gli script non leggano alcun file di avvio standard ma cerchino invece il valore della variabile `BASH_ENV` e lo usano come file di avvio se esiste.

Dimostriamo questo processo:

1. Creiamo il nostro file di avvio chiamato `.startup_script` con il seguente contenuto:

   ```
   CROCODILIAN=caiman
   ```

2. Scriviamo uno script Bash chiamato `test_env.sh` con il seguente contenuto:

   ```
   #!/bin/bash

   echo $CROCODILIAN
   ```

3. Abbiamo impostato il bit eseguibile per il nostro script `test_env.sh`:

   ```
   $ chmod +x test_env.sh
   ```

4. Infine, usiamo `env` per impostare `BASH_ENV` su `.startup_script` per l'esecuzione di `test_env.sh`:

   ```
   $ env BASH_ENV=/home/user2/.startup_script ./test_env.sh
   caiman
   ```

   Il comando `env` è implicito anche se lo eliminiamo:

   ```
   $ BASH_ENV=/home/user2/.startup_script ./test_env.sh
   ```

```
caiman
```

NOTE

Se non capisci la riga `#!/bin/bash` o il comando `chmod+x`, niente panico! Impareremo tutto il necessario sullo scripting della shell nelle prossime lezioni. Per ora, ricorda che per eseguire uno script dalla sua directory usiamo `./some_script`.

Variabili d'Ambiente Comuni

È giunto il momento di rivedere alcune delle variabili di ambiente più rilevanti impostate nei file di configurazione di Bash.

`DISPLAY`

Relativo al server *X*, il valore di questa variabile è normalmente costituito da tre elementi:

- Il nome host (la sua assenza significa `localhost`) dove è in esecuzione il server X.
- I due punti come delimitatore.
- Un numero (normalmente è `0` e si riferisce al display del computer).

```
$ printenv DISPLAY
:0
```

Un valore vuoto per questa variabile indica un server senza un sistema X Window. Un numero extra — come in `my.xserver:0:1` — farebbe riferimento al numero dello schermo se ne esistesse più di uno.

`HISTCONTROL`

Questa variabile controlla quali comandi vengono salvati in `HISTFILE` (vedi sotto). Esistono tre possibili valori:

`ignorespace`

I comandi che iniziano con uno spazio non verranno salvati.

`ignoredups`

Un comando uguale al precedente non verrà salvato.

`ignoreboth`

I comandi che rientrano in una delle due categorie precedenti non verranno salvati.

```
$ echo $HISTCONTROL
ignoreboth
```

HISTSIZE

Questo imposta il numero di comandi da salvare in memoria durante la sessione della shell.

```
$ echo $HISTSIZE
1000
```

HISTFILESIZE

Questo imposta il numero di comandi da salvare in `HISTFILE` dall'inizio alla fine della sessione:

```
$ echo $HISTFILESIZE
2000
```

HISTFILE

Il nome del file che memorizza tutti i comandi mentre vengono digitati. Per impostazione predefinita, questo file si trova in `~/.bash_history`:

```
$ echo $HISTFILE
/home/user2/.bash_history
```

NOTE

Per visualizzare il contenuto di `HISTFILE`, è sufficiente digitare `history`. In alternativa, possiamo specificare il numero di comandi che vogliamo vedere passando un argomento (il numero dei comandi più recenti) a `history` come in `history 3`.

HOME

Questa variabile memorizza il percorso assoluto della directory home dell'utente corrente e viene impostata quando l'utente effettua il login.

Il seguente codice — da `~/.profile` — è autoesplicativo (fornisce `"$HOME/.bashrc"` se esiste):

```
# include .bashrc if it exists
if [ -f "$HOME/.bashrc" ]; then
. "$HOME/.bashrc"
```

```
    fi
```

NOTE Se non capisci bene l'istruzione `if`, non preoccuparti: fai riferimento alle lezioni sullo scripting di shell.

Ricorda che `~` è equivalente a `$HOME`:

```
$ echo ~; echo $HOME
/home/carol
/home/carol
```

NOTE I comandi possono essere concatenati con un punto e virgola (`;`).

Possiamo anche dimostrarlo con un'istruzione `if`:

```
$ if [ ~ == "$HOME" ]; then echo "true"; else echo "false"; fi
true
```

NOTE Ricorda: il segno di uguale `=` viene utilizzato per l'assegnazione delle variabili. `==` viene utilizzato per verificare l'uguaglianza.

HOSTNAME

Questa variabile memorizza il nome TCP/IP del computer host:

```
$ echo $HOSTNAME
debian
```

HOSTTYPE

Memorizza l'architettura del processore del computer host:

```
$ echo $HOSTTYPE
x86_64
```

LANG

Questa variabile salva le impostazioni internazionali del sistema:

```
$ echo $LANG
en_UK.UTF-8
```

`LD_LIBRARY_PATH`

Questa variabile è costituita da un insieme di directory separate da due punti dove si trovano le librerie condivise:

```
$ echo $LD_LIBRARY_PATH
/usr/local/lib
```

`MAIL`

Questa variabile memorizza il file in cui Bash cerca la casella di posta dell'utente:

```
$ echo $MAIL
/var/mail/carol
```

Un altro valore comune per questa variabile è `/var/spool/mail/$USER`.

`MAILCHECK`

Questa variabile memorizza un valore numerico che indica in secondi la frequenza con cui Bash controlla la nuova posta:

```
$ echo $MAILCHECK
60
```

`PATH`

Questa variabile di ambiente memorizza l'elenco delle directory in cui Bash cerca i file eseguibili quando gli viene chiesto di eseguire qualsiasi programma. Nella nostra macchina di esempio questa variabile è impostata tramite il file `/etc/profile` a livello di sistema:

```
if [ "`id -u`" -eq 0 ]; then
  PATH="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin"
else
  PATH="/usr/local/bin:/usr/bin:/bin:/usr/local/games:/usr/games"
fi
export PATH
```

Attraverso l'istruzione `if` viene testata l'identità dell'utente e — a seconda del risultato del test (`root` o altro) — otterremo un `PATH` o l'altro. Infine il `PATH` scelto viene propagato con `export`.

Osserva due cose riguardo al valore di `PATH`:

- I nomi delle directory vengono scritti utilizzando percorsi assoluti.
- I due punti vengono utilizzati come delimitatori.

Se volessimo includere la cartella `/usr/local/sbin` nel `PATH` per gli utenti regolari, modificheremo la riga in modo che assomigli a questo:

```
(...)
else
  PATH="/usr/local/bin:/usr/bin:/bin:/usr/local/games:/usr/games:/usr/local/sbin"
fi
export PATH
```

Ora possiamo vedere come cambia il valore della variabile quando accediamo come utente normale:

```
# su - carol
$ echo $PATH
/usr/local/bin:/usr/bin:/bin:/usr/local/games:/usr/games:/usr/local/sbin
```

NOTE Avremmo potuto anche aggiungere `/usr/local/sbin` al `PATH` dell'utente dalla riga di comando digitando `PATH=/usr/local/sbin:$PATH` o `PATH=$PATH:/usr/local/sbin` Nel primo caso rendiamo `/usr/local/sbin` la prima directory in cui cercare i file eseguibili, nel secondo lo rendiamo l'ultimo.

`PS1`

Questa variabile memorizza il valore del prompt di Bash. Nella seguente porzione di codice (anche da `/etc/profile`), l'istruzione `if` verifica l'identità dell'utente e fornisce di conseguenza un prompt specifico (`#` per `root` o `$` per utenti regolari):

```
if [ "`id -u`" -eq 0 ]; then
  PS1='# '
else
  PS1='$ '
fi
```

NOTE L `id` di `root` è `0`. Diventa `root` e provalo tu stesso con `id -u`.

Altre variabili del prompt includono:

PS2

Normalmente impostato su `>` e usato come prompt di continuazione per lunghi comandi multilinea.

PS3

Usato come prompt per il comando `select`.

PS4

Normalmente impostato su `+` e utilizzato per il debug.

SHELL

Questa variabile memorizza il percorso assoluto della shell corrente:

```
$ echo $SHELL
/bin/bash
```

USER

Memorizza il nome dell'utente corrente:

```
$ echo $USER
carol
```

Esercizi Guidati

1. Osserva l'assegnazione della variabile nella colonna "Comando(i)" e indica se la variabile risultante è "Locale" o "Globale":

Comando(i)	Locale	Globale
`debian=mother`		
`ubuntu=deb-based`		
`mint=ubuntu-based; export mint`		
`export suse=rpm-based`		
`zorin=ubuntu-based`		

2. Studia il "Comando" e l' "Output" e spiegane il significato:

Comando	Output	Significato
`echo $HISTCONTROL`	`ignoreboth`	
`echo ~`	`/home/carol`	
`echo $DISPLAY`	`reptilium:0:2`	
`echo $MAILCHECK`	`60`	
`echo $HISTFILE`	`/home/carol/.bash_history`	

3. Le variabili vengono impostate in modo errato nella colonna "Command Errato". Fornisci le informazioni mancanti sotto "Comando Giusto" e "Riferimento della Variabile" in modo da ottenere l'"Output Previsto":

Comando Errato	Comando Giusto	Riferimento della Variabile	Output Previsto
`lizard =chameleon`			`chameleon`
`cool lizard=chameleon`			`chameleon`
`lizard=cha|me|leon`			`cha|me|leon`

Comando Errato	Comando Giusto	Riferimento della Variabile	Output Previsto
`lizard=/** chameleon **/`			`/** chamelon **/`
`win_path=C:\path\to\dir\`			`C:\path\to\dir\`

4. Considera lo scopo e scrivi il comando appropriato:

Scopo	Comando
Imposta la lingua della shell corrente sullo spagnolo UTF-8 (`es_ES.UTF-8`).	
Stampa il nome della directory di lavoro corrente.	
Fai riferimento alla variabile d'ambiente che memorizza le informazioni sulle connessioni `ssh`.	
Imposta `PATH` per includere `/home/carol/scripts` come ultima directory in cui cercare gli eseguibili.	
Imposta il valore di `my_path` come `PATH`.	
Imposta il valore di `my_path` a quello della variabile `PATH`.	

5. Crea una variabile locale chiamata `mammal` e assegnale il valore `gnu`:

6. Usando la sostituzione delle variabili, crea un'altra variabile locale chiamata `var_sub` con il valore appropriato in modo che quando ci si richiama tramite `echo $var_sub` otteniamo: `The value of mammal is gnu`:

7. Trasforma `mammal` in una variabile d'ambiente:

8. Cercalo con `set` e `grep`:

9. Cercalo con `env` e `grep`:

10. Crea, in due comandi consecutivi, una variabile d'ambiente chiamata `BIRD` il cui valore è `penguin`:

11. Crea, in un unico comando, una variabile d'ambiente chiamata `NEW_BIRD` il cui valore è `yellow-eyed penguin`:

12. Supponendo che tu sia `user2`, crea una cartella chiamata `bin` nella tua directory home:

13. Digita il comando per aggiungere la cartella `~/bin` al tuo `PATH` in modo che sia la prima directory dove `bash` cerca i binari:

14. Per garantire che il valore di `PATH` rimanga inalterato durante i riavvii, quale parte di codice — sotto forma di un'istruzione `if` — inseriresti in `~/.profile`?

Esercizi Esplorativi

1. `let`: valuta le espressioni aritmetiche:
   - Cerca nella pagine di `man` o una ricerca web per `let` e le sue implicazioni quando si impostano le variabili e crea una nuova variabile locale denominata `my_val` il cui valore è `10` — come risultato della somma di 5 più 5:

     Ora crea un'altra variabile chiamata `your_val` il cui valore è `5` — come risultato della divisione del valore di "my_val" per 2:

2. Il risultato di un comando in una variabile? Ovviamente questo è possibile; si chiama *sostituzione di comando*. Investigalo e studia la seguente funzione chiamata `music_info`:

```
music_info(){
latest_music=`ls -l1t ~/Music | head -n 6`
echo -e "Your latest 5 music files:\n$latest_music"
}
```

   Il risultato del comando `ls -l1t ~/Music | head -n 6` diventa il valore della variabile `latest_music`. Quindi si fa riferimento alla variabile `latest_music` nel comando `echo` (che restituisce il numero totale di byte occupati dalla cartella `Music` e gli ultimi cinque file musicali memorizzati nella cartella `Music` — uno per riga).

   Quale dei seguenti è un'alternativa valida a:

```
latest_music=`ls -l1t ~/Music | head -n 6`
```

   Opzione A:

```
latest_music=$(ls -l1t ~/Music| head -n 6)
```

   Opzione B:

```
latest_music="(ls -l1t ~/Music| head -n 6)"
```

   Opzione C:

```
latest_music=((ls -l1t ~/Music| head -n 6))
```

Sommario

In questa lezione abbiamo imparato:

- Le variabili sono una parte molto importante dell'ambiente della shell poiché vengono utilizzate dalla shell stessa e da altri programmi.
- Come assegnare e fare riferimento alle variabili.
- Le differenze tra le variabili *locali* e *globali* (o di *ambiente*).
- Come creare variabili *readonly*.
- Come trasformare una variabile locale in una variabile d'ambiente con il comando `export`.
- Come elencare tutte le variabili d'ambiente.
- Come eseguire un programma in un ambiente modificato.
- Come rendere le variabili persistenti con l'aiuto degli script di avvio.
- Alcune variabili d'ambiente comuni: `DISPLAY`, `HISTCONTROL`, `HISTSIZE`, `HISTFILESIZE`, `HISTFILE`, `HOME`, `HOSTNAME`, `HOSTTYPE`, `LANG`, `LD_LIBRARY_PATH`, `MAIL`, `MAILCHECK`, `PATH`, `PS1` (e altre variabili di prompt), `SHELL` e `USER`.
- Il significato della tilde (`~`).
- Le basi delle istruzioni `if`.

Comandi utilizzati in questa lezione:

`echo`
: Mostra il valore di una variabile

`ls`
: Elenca il contenuto della directory.

`readonly`
: Rende le variabili immutabili. Elenca tutte le variabili di sola lettura nella sessione corrente.

`set`
: Elenca tutte le variabili e le funzioni nella sessione corrente.

`grep`
: Stampa le linee che corrispondono a un *pattern*.

`bash`

Esegue una nuova shell Bash

`unset`

Annulla le variabili.

`export`

Trasforma una variabile locale in una variabile d'ambiente. Elenca le variabili d'ambiente.

`env`

Elenca le variabili d'ambiente. Esegue un programma in un ambiente modificato.

`printenv`

Elenca le variabili d'ambiente. Fai riferimento a una variabile.

`chmod`

Modifica i bit di modalità di un file, per esempio, renderlo eseguibile.

`history`

Elenca i comandi precedenti.

`su`

Cambia lo user ID o diventa superutente.

`id`

Visualizza lo user ID.

Risposte agli Esercizi Guidati

1. Osserva l'assegnazione della variabile nella colonna "Comando(i)" e indica se la variabile risultante è "Locale" o "Globale":

Comando(i)	Locale	Globale
`debian=mother`	Sì	No
`ubuntu=deb-based`	Sì	No
`mint=ubuntu-based; export mint`	No	Sì
`export suse=rpm-based`	No	Sì
`zorin=ubuntu-based`	Sì	No

2. Studia il "Comando" e l' "Output" e spiegane il significato:

Comando	Output	Significato
`echo $HISTCONTROL`	`ignoreboth`	Sia i comandi duplicati che quelli che iniziano con uno spazio non verranno salvati nell' `history`.
`echo ~`	`/home/carol`	La `HOME` di `carol` è `/home/carol`.
`echo $DISPLAY`	`reptilium:0:2`	Il sistema `reptilium` ha X come server in esecuzione e stiamo usando la seconda schermata del display .
`echo $MAILCHECK`	`60`	La posta verrà controllata ogni minuto.
`echo $HISTFILE`	`/home/carol/.bash_history`	L'`history` sarà salvata all'interno di `/home/carol/.bash_history`.

3. Le variabili vengono impostate in modo errato nella colonna "Command Errato". Fornisci le informazioni mancanti sotto "Comando Giusto" e "Riferimento della Variabile" in modo da ottenere l'"Output Previsto":

Comando Errato	Comando Giusto	Riferimento della Variabile	Output Previsto
`lizard =chameleon`	`lizard=chameleon`	`echo $lizard`	`chameleon`
`cool lizard=chameleon`	`cool_lizard=chameleon` (per esempio)	`echo $cool_lizard`	`chameleon`
`lizard=cha|me|leon`	`lizard="cha|me|leon"` o `lizard='cha|me|leon'`	`echo $lizard`	`cha|me|leon`
`lizard=/** chameleon **/`	`lizard="/** chameleon **/"` o `lizard='/** chameleon **/'`	`echo "$lizard"`	`/** chamelon **/`
`win_path=C:\path\to\dir\`	`win_path=C:\\path\\to\\dir\\`	`echo $win_path`	`C:\path\to\dir\`

4. Considera lo scopo e scrivi il comando appropriato:

Scopo	Comando
Imposta la lingua della shell corrente sullo spagnolo UTF-8 (`es_ES.UTF-8`).	`LANG=es_ES.UTF-8`
Stampa il nome della directory di lavoro corrente.	`echo $PWD` o `pwd`
Fare riferimento alla variabile d'ambiente che memorizza le informazioni sulle connessioni `ssh`.	`echo $SSH_CONNECTION`
Imposta `PATH` per includere `/home/carol/scripts` come ultima directory in cui cercare gli eseguibili.	`PATH=$PATH:/home/carol/scripts`
Imposta il valore di `my_path` come `PATH`.	`my_path=PATH`
Imposta il valore di `my_path` a quello della variabile `PATH`.	`my_path=$PATH`

5. Crea una variabile locale chiamata `mammal` e assegnale il valore `gnu`:

```
mammal=gnu
```

6. Usando la sostituzione delle variabili, crea un'altra variabile locale chiamata `var_sub` con il valore appropriato in modo che, quando si richiama tramite `echo $var_sub`, si ottenga: `The value of mammal is gnu`:

```
var_sub="The value of mammal is $mammal"
```

7. Trasforma `mammal` in una variabile d'ambiente:

```
export mammal
```

8. Cercalo con `set` e `grep`:

```
set | grep mammal
```

9. Cercalo con `env` e `grep`:

```
env | grep mammal
```

10. Crea, in due comandi consecutivi, una variabile d'ambiente chiamata `BIRD` il cui valore è `penguin`:

```
BIRD=penguin; export BIRD
```

11. Crea, in un unico comando, una variabile d'ambiente chiamata `NEW_BIRD` il cui valore è `yellow-eyed penguin`:

```
export NEW_BIRD="yellow-eyed penguin"
```

o

```
export NEW_BIRD='yellow-eyed penguin'
```

12. Supponendo che tu sia `user2`, crea una cartella chiamata `bin` nella tua directory home:

```
mkdir ~/bin
```

o

```
mkdir /home/user2/bin
```

o

```
mkdir $HOME/bin
```

13. Digita il comando per aggiungere la cartella `~/bin` al tuo `PATH` in modo che sia la prima directory dove `bash` cerca i binari:

```
PATH="$HOME/bin:$PATH"
```

`PATH=~/bin:$PATH` o `PATH=/home/user2/bin:$PATH` sono egualmente validi.

14. Per garantire che il valore di `PATH` rimanga inalterato durante i riavvii, quale parte di codice — sotto forma di un'istruzione `if` — inseriresti in `~/.profile`?

```
if [ -d "$HOME/bin" ] ; then
    PATH="$HOME/bin:$PATH"
fi
```

Risposte agli Esercizi Esplorativi

1. `let`: valuta le espressioni aritmetiche:
   - Cerca nella pagine di `man` o una ricerca web per `let` e le sue implicazioni quando si impostano le variabili e crea una nuova variabile locale denominata `my_val` il cui valore è `10` — come risultato della somma di 5 più 5:

     ```
     let "my_val = 5 + 5"
     ```

     o

     ```
     let 'my_val = 5 + 5'
     ```

   - Ora crea un'altra variabile chiamata `your_val` il cui valore è `5` — come risultato della divisione del valore di "my_val" per 2:

     ```
     let "your_val = $my_val / 2"
     ```

     o

     ```
     let 'your_val = $my_val / 2'
     ```

2. Il risultato di un comando in una variabile? Ovviamente questo è possibile; si chiama *sostituzione di comando*. Investigalo e studia la seguente funzione chiamata `music_info`:

   ```
   music_info(){
   latest_music=`ls -l1t ~/Music | head -n 6`
   echo -e "Your latest 5 music files:\n$latest_music"
   }
   ```

   Il risultato del comando `ls -l1t ~/Music | head -n 6` diventa il valore della variabile `latest_music`. Quindi si fa riferimento alla variabile `latest_music` nel comando `echo` (che restituisce il numero totale di byte occupati dalla cartella `Music` e gli ultimi cinque file musicali memorizzati nella cartella `Music` — uno per riga).

   Quale dei seguenti è un'alternativa valida a:

```
latest_music=`ls -l1t ~/Music | head -n 6`
```

È l'opzione A:

```
latest_music=$(ls -l1t ~/Music| head -n 6)
```



105.1 Lezione 3

Certificazione:	LPIC-1
Versione:	5.0
Argomento:	105 Shell e Script di Shell
Obiettivo:	105.1 Personalizzare e utilizzare l'ambiente di Shell
Lezione:	3 di 3

Introduzione

Dopo aver esaminato le shell, gli script di avvio e le variabili nelle lezioni precedenti, completeremo l'intero argomento della personalizzazione della shell dando un'occhiata a due elementi di shell molto interessanti: *alias* e *funzioni*. In effetti l'intero gruppo di variabili, alias e funzioni — e la loro influenza reciproca — è ciò che costituisce l'ambiente della shell.

Il principale punto di forza di queste due caratteristiche di Shell ha a che fare con il concetto di *incapsulamento*: offrono la possibilità di mettere insieme — sotto un unico comando — una serie di ulteriori comandi ripetitivi o ricorrenti.

Creazione di Alias

Un alias è un nome sostitutivo per altro/i comando/i. Può essere eseguito come fosse un qualsiasi altro eseguibile, ma esegue invece uno o più comandi e relative opzioni base della definizione dell'alias stesso.

La sintassi per la dichiarazione degli alias è abbastanza semplice. Gli alias vengono dichiarati

scrivendo la parola chiave `alias` seguita dall'assegnazione dello stesso. Nello specifico, l'assegnazione dell'alias consiste nel *nome dell'alias*, un *carattere di uguale* e uno o più *commandi*:

```
alias alias_name=command(s)
```

Per esempio:

```
$ alias oldshell=sh
```

Questo strano alias avvierà un'istanza della shell `sh` originale quando l'utente digita `oldshell` sul terminale:

```
$ oldshell
$
```

Il potere degli alias sta nel fatto che ci permette di scrivere versioni brevi di comandi lunghi:

```
$ alias ls='ls --color=auto'
```

NOTE Per informazioni su `ls` e sui suoi colori, digita `man dir_colors` sul terminale.

Allo stesso modo, possiamo creare alias per una serie di comandi concatenati - il punto e virgola (`;`) è usato come delimitatore. Possiamo, per esempio, avere un alias che ci fornisce informazioni sulla posizione dell'eseguibile `git` e sulla sua versione:

```
$ alias git_info='which git;git --version'
```

Per richiamare un alias, digitiamo il suo nome nel terminale:

```
$ git_info
/usr/bin/git
git version 2.7.4
```

Il comando `alias` produrrà un elenco di tutti gli alias disponibili nel sistema:

```
$ alias
alias git-info='which git;git --version'
```

```
alias ls='ls --color=auto'
alias oldshell='sh'
```

Il comando `unalias` rimuove gli alias. Possiamo, per esempio, `unalias git-info` e vedere come scompare dall'elenco:

```
$ unalias git-info
$ alias
alias ls='ls --color=auto'
alias oldshell='sh'
```

Come abbiamo visto con l'`alias hi='echo We salute you.'` in una lezione precedente, dobbiamo racchiudere i comandi tra virgolette (singole o doppie) quando — a causa di argomenti o parametri — contengono spazi :

```
$ alias greet='echo Hello world!'
$ greet
Hello world!
```

I comandi con spazi includono anche quelli con opzioni:

```
$ alias ll='ls -al'
```

Ora `ll` elencherà tutti i file, inclusi quelli nascosti (`a`), nel formato lungo (`l`).

Possiamo utilizzare le variabili negli alias:

```
$ reptile=uromastyx
$ alias greet='echo Hello $reptile!'
$ greet
Hello uromastyx!
```

La variabile può essere assegnata anche all'interno dell'alias:

```
$ alias greet='reptile=tortoise; echo Hello $reptile!'
$ greet
Hello tortoise!
```

Possiamo sfuggire (*escape*) a un alias con `\`:

```
$ alias where?='echo $PWD'
$ where?
/home/user2
$ \where?
-bash: where?: command not found
```

L'escape di un alias è utile quando quest'ultimo ha lo stesso nome di un normale comando. In questo caso, l'alias ha la precedenza sul comando originale, che però è ancora accessibile mediante l'escape dell'alias.

Allo stesso modo, possiamo inserire un alias all'interno di un altro alias:

```
$ where?
/home/user2
$ alias my_home=where?
$ my_home
/home/user2
```

Inoltre, possiamo anche inserire una funzione all'interno di un alias come verrà mostrato di seguito.

Espansione e Valutazione delle Virgolette negli Alias

Quando si utilizzano virgolette con variabili di ambiente, le virgolette singole rendono dinamica l'espansione:

```
$ alias where?='echo $PWD'
$ where?
/home/user2
$ cd Music
$ where?
/home/user2/Music
```

Differentemente, con le virgolette doppie l'espansione viene eseguita in modo statico:

```
$ alias where?="echo $PWD"
$ where?
/home/user2
```

```
$ cd Music
$ where?
/home/user2
```

Persistenza degli Alias: Script di Avvio

Proprio come con le variabili, affinché i nostri alias acquisiscano persistenza, dobbiamo inserirli negli script di inizializzazione. Come già sappiamo, un buon file in cui gli utenti possono inserire i propri alias personali è `~/.bashrc`. Probabilmente troverai già alcuni alias (la maggior parte dei quali commentati e pronti per essere utilizzati rimuovendo il `#` iniziale):

```
$ grep alias .bashrc
# enable color support of ls and also add handy aliases
    alias ls='ls --color=auto'
    #alias dir='dir --color=
    #alias vdir='vdir --color=
    #alias grep='grep --color=
    #alias fgrep='fgrep --color'
    #alias egrep='egrep --color=
# some more ls aliases
#ll='ls -al'
#alias la='ls -A'
#alias l='ls -CF'
# ~/.bash_aliases, instead of adding them here directly.
if [ -f ~/.bash_aliases ]; then
   . ~/.bash_aliases
```

Come puoi leggere nelle ultime tre righe, ci viene offerta la possibilità di avere il nostro file dedicato agli alias — `~/.bash_aliases` — e di farlo richiamare da `.bashrc`. Quindi possiamo seguire questa strada, popolando tale file:

```
############
# .bash_aliases:
# a file to be populated by the user's personal aliases (and sourced by ~/.bashrc).
############
alias git_info='which git;git --version'
alias greet='echo Hello world!'
alias ll='ls -al'
alias where?='echo $PWD'
```

Creazione di Funzioni

Rispetto agli alias, le funzioni sono più programmatiche e flessibili, specialmente quando si tratta di sfruttare tutto il potenziale delle variabili incorporate speciali di *Bash* e dei parametri posizionali. Sono anche ottimi per lavorare con strutture di controllo del flusso come loop o condizionali. Possiamo pensare a una funzione come a un comando che include la logica attraverso blocchi o raccolte di altri comandi.

Due Sintassi per la Creazione di Funzioni

Esistono due sintassi valide per definire le funzioni.

Usando la parola chiave `function`

Da un lato, possiamo usare la parola chiave `function`, seguita dal nome della funzione e dai comandi tra parentesi graffe:

```
function function_name {
command #1
command #2
command #3
.
.
.
command #n
}
```

Usando `()`

Dall'altro, possiamo tralasciare la parola chiave `function` e utilizzare invece due parentesi subito dopo il nome della funzione:

```
function_name() {
command #1
command #2
command #3
.
.
.
command #n
}
```

È normale inserire funzioni in file o script. Tuttavia, possono anche essere scritti direttamente nel prompt della shell con ogni comando su una riga diversa — nota `PS2(>)` che indica una nuova riga dopo un'interruzione di riga:

```
$ greet() {
> greeting="Hello world!"
> echo $greeting
> }
```

In ogni caso e indipendentemente dalla sintassi che scegliamo, se decidiamo di saltare le interruzioni di riga e scrivere una funzione in una sola riga, i comandi devono essere separati da punto e virgola (notare anche il punto e virgola dopo l'ultimo comando):

```
$ greet() { greeting="Hello world!"; echo $greeting; }
```

`bash` non ha dato errori quando abbiamo premuto Invio, quindi la nostra funzione è pronta per essere invocata. Per richiamare una funzione, dobbiamo digitare il suo nome nel terminale:

```
$ greet
Hello world!
```

Proprio come con le variabili e gli alias, se vogliamo che le funzioni siano persistenti durante i riavvii del sistema, dobbiamo inserirle negli script di inizializzazione della shell come `/etc/bash.bashrc` (globale) o `~/.bashrc` (locale).

WARNING Dopo aver aggiunto alias o funzioni a qualsiasi file di script di avvio, è necessario creare tali file con `.` o `source` affinché le modifiche abbiano effetto se non si desidera disconnettersi e rientrare di nuovo o riavviare il sistema.

Variabili Speciali Integrate di Bash

La *Bourne Again Shell* viene fornita con una serie di variabili speciali particolarmente utili per funzioni e script. Sono "speciali" perché possono essere solo referenziate, non assegnate. Ecco un elenco di quelle più rilevanti:

`$?`

Il riferimento di questa variabile si espande al risultato dell'ultimo comando eseguito. Un valore di `0` significa successo:

```
$ ps aux | grep bash
user2      420  0.0  0.4  21156  5012 pts/0    Ss   17:10   0:00 -bash
user2      640  0.0  0.0  12784   936 pts/0    S+   18:04   0:00 grep bash
$ echo $?
0
```

Un valore diverso da `0` indica un errore:

```
user1@debian:~$ ps aux |rep bash
-bash: rep: command not found
user1@debian:~$ echo $?
127
```

`$$`

Si espande al PID della shell (ID processo):

```
$ ps aux | grep bash
user2      420  0.0  0.4  21156  5012 pts/0    Ss   17:10   0:00 -bash
user2      640  0.0  0.0  12784   936 pts/0    S+   18:04   0:00 grep bash
$ echo $$
420
```

`$!`

Si espande al PID dell'ultimo *job* in background:

```
$ ps aux | grep bash &
[1] 663
$ user2      420  0.0  0.4  21156  5012 pts/0    Ss+  17:10   0:00 -bash
user2      663  0.0  0.0  12784   972 pts/0    S    18:08   0:00 grep bash
^C
[1]+  Done                    ps aux | grep bash
$ echo $!
663
```

NOTE Ricorda, la e commerciale (&) viene utilizzata per avviare i processi in background.

Parametri posizionali da `$0` a `$9`

Si espandono ai parametri o agli argomenti passati alla funzione (alias o script) — `$0` si espande

al nome dello script o della shell.

Creiamo una funzione per dimostrare i parametri posizionali — nota `PS2(>)` che indica nuove righe dopo le interruzioni di riga:

```
$ special_vars() {
> echo $0
> echo $1
> echo $2
> echo $3
}
```

Ora, invocheremo la funzione (`special_vars`) passandole tre parametri (`debian`, `ubuntu`, `zorin`):

```
$ special_vars debian ubuntu zorin
-bash
debian
ubuntu
zorin
```

Ha funzionato come previsto.

WARNING

Sebbene il passaggio di parametri posizionali agli alias sia tecnicamente possibile, non è affatto funzionale poiché - con gli alias - i parametri posizionali vengono sempre passati alla fine:

```
$ alias great_editor='echo $1 is a great text editor'
$ great_editor emacs
is a great text editor emacs
```

Altre variabili integrate speciali di Bash includono:

`$#`

Si espande al numero di argomenti passati al comando.

`$@`, `$*`

Si espandono agli argomenti passati al comando.

`$_`

Si espande all'ultimo parametro o al nome dello script (fare riferimento a `man bash` per saperne di più!):

Variabili all'interno di Funzioni

Naturalmente, le variabili possono essere utilizzate all'interno delle funzioni.

Dimostriamolo creando un nuovo file vuoto chiamato `funed` e inserendo la seguente funzione:

```
editors() {

editor=emacs

echo "My editor is: $editor. $editor is a fun text editor."
}
```

Come avrai già intuito, dobbiamo prima cercare il file per poter invocare la funzione:

```
$ . funed
```

E ora possiamo testarlo:

```
$ editors
My editor is emacs. emacs is a fun text editor.
```

Come puoi notare, affinché la funzione `editors` possa lavorare correttamente, la variabile `editor` deve prima essere impostata. L'ambito di quella variabile è locale rispetto alla shell corrente e possiamo farvi riferimento finché dura la sessione:

```
$ echo $editor
emacs
```

Insieme alle variabili locali possiamo anche includere nella nostra funzione variabili d'ambiente:

```
editors() {

editor=emacs
```

```
echo "The text editor of $USER is: $editor."
}

editors
```

Nota come questa volta abbiamo deciso di chiamare la funzione dall'interno del file stesso (`editors` nell'ultima riga). In questo modo, quando richiamiamo il file, questo avverrà anche per la funzione senza soluzione di continuità:

```
$ . funed
The text editor of user2 is: emacs.
```

Parametri Posizionali nelle Funzioni

Qualcosa di simile si verifica con i parametri posizionali.

Possiamo passarli alle funzioni dall'interno del file o dello script (nota l'ultima riga: `editors tortoise`):

```
editors() {

editor=emacs

echo "The text editor of $USER is: $editor."
echo "Bash is not a $1 shell."
}

editors tortoise
```

Richiamiamo il file e dimostriamo che funziona:

```
$ . funed
The text editor of user2 is: emacs.
Bash is not a tortoise shell.
```

E possiamo anche passare parametri posizionali alle funzioni dalla riga di comando. Per dimostrarlo, ci liberiamo dell'ultima riga del file:

```
editors() {
```

```
editor=emacs

echo "The text editor of $USER is: $editor."
echo "Bash is not a $1 shell."
}
```

Quindi, dobbiamo richiamare il file:

```
$ . funed
```

Infine, invochiamo la funzione con `tortoise` come parametro posizionale `$1` nella riga di comando:

```
$ editors tortoise
The text editor of user2 is: emacs.
Bash is not a tortoise shell.
```

Funzioni negli Script

Le funzioni si trovano principalmente negli script Bash.

Trasformare il nostro file `funed` in uno script (lo chiameremo `funed.sh`) è davvero un gioco da ragazzi:

```
#!/bin/bash

editors() {

editor=emacs

echo "The text editor of $USER is: $editor."
echo "Bash is not a $1 shell."
}

editors tortoise
```

Questo è tutto! Abbiamo aggiunto solo due righe:

- La prima riga è *shebang* e definisce quale programma interpreterà lo script: `#!/bin/bash`. Curiosamente, quel programma è lo stesso `bash`.

- L'ultima riga è semplicemente l'invocazione della funzione.

Ora resta solo una cosa: dobbiamo rendere eseguibile lo script:

```
$ chmod +x funed.sh
```

E ora è pronto per essere eseguito:

```
$ ./funed.sh
The text editor of user2 is: emacs.
Bash is not a tortoise shell.
```

NOTE Imparerai tutto sullo *shell scripting* nelle prossime lezioni.

Una Funzione all'Interno di un Alias

Come detto sopra, possiamo inserire una funzione all'interno di un alias:

```
$ alias great_editor='gr8_ed() { echo $1 is a great text editor; unset -f gr8_ed; }; gr8_ed'
```

Questo lungo valore alias merita una spiegazione. Cerchiamo di scomporlo:

- Prima c'è la funzione stessa: `gr8_ed() {echo $1 is a great text editor; unset -f gr8_ed; }`
- L'ultimo comando nella funzione — `unset -f gr8_ed` — ripristina la funzione in modo che non rimanga nella sessione attuale di `bash` dopo che l'alias è stato richiamato.
- Ultimo ma non meno importante, per avere una corretta invocazione dell'alias, dobbiamo prima invocare anche la funzione: `gr8_ed`.

Richiamiamo l'alias e dimostriamo che funziona:

```
$ great_editor emacs
emacs is a great text editor
```

Come mostrato sopra in `unset -f gr8_ed`, il comando `unset` non è usato solo per annullare l'impostazione delle variabili, ma anche per le funzioni. In effetti, ci sono opzioni specifiche:

`unset -v`
: per le variabili

`unset -f`
: per le funzioni

Se usato senza interruttori, `unset` proverà prima a annullare l'impostazione di una variabile e, se fallisce, proverà a disattivare una funzione.

Una Funzione all'Interno di una Funzione

Diciamo ora di voler comunicare due cose a `user2` ogni volta che accede al sistema:

- Saluta e consiglia/elogia un editor di testo.
- Dato che sta iniziando a mettere molti file video `Matroska` nella sua cartella `$HOME/Video`, vogliamo anche darle un avvertimento.

Per raggiungere questo scopo, abbiamo inserito le seguenti due funzioni in `/home/user2/.bashrc`:

La prima funzione (`check_vids`) esegue il controllo dei file `.mkv` e l'avviso:

```
check_vids() {
    ls -1 ~/Video/*.mkv > /dev/null 2>&1
    if [ "$?" = "0" ];then
        echo -e "Remember, you must not keep more than 5 video files in your Video
folder.\nThanks."
    else
    echo -e "You do not have any videos in the Video folder. You can keep up to 5.\nThanks."
    fi
}
```

`check_vids` svolge tre compiti:

- Elenca i file `mkv` in `~/Video` inviando l'output — e qualsiasi errore — al cosiddetto *bit-bucket* (`/dev/null`).
- Verifica il successo dell'output del comando precedente.
- A seconda del risultato del test, riproduce uno dei due messaggi.

La seconda funzione è una versione modificata della nostra funzione `editors`:

```
editors() {

editor=emacs

echo "Hi, $USER!"
echo "$editor is more than a text editor!"

check_vids
}

editors
```

È importante osservare due cose:

- L'ultimo comando di `editors` invoca `check_vids` in modo che entrambe le funzioni vengano concatenate: il saluto, la lode, il controllo e l'avvertimento vengono eseguiti in sequenza.
- `editors` stesso è il punto di ingresso alla sequenza di funzioni, quindi viene invocato nell'ultima riga (`editors`).

Ora, accediamo come `user2` e dimostriamo che funziona:

```
# su - user2
Hi, user2!
emacs is more than a text editor!
Remember, you must not keep more than 5 video files in your Video folder.
Thanks.
```

Esercizi Guidati

1. Completa la tabella con "Sì" o "No" considerando le capacità di alias e funzioni:

Caratteristica	**Alias?**	**Funzioni?**
È possibile utilizzare variabili locali		
È possibile utilizzare variabili d'ambiente		
Può essere evitato con `\`		
Può essere ricorsivo		
Molto produttivo se utilizzato con parametri posizionali		

2. Immetti il comando che elenca tutti gli alias nel sistema:

3. Scrivi un alias chiamato `logg` che elenchi tutti i file `ogg` in `~/Music` — uno per riga:

4. Richiama l'alias per dimostrare che funziona:

5. Ora, modifica l'alias in modo che mostri l'utente di sessione e i due punti prima dell'elenco:

6. Richiamalo di nuovo per dimostrare che anche questa nuova versione funziona:

7. Elenca di nuovo tutti gli alias e controlla che il tuo alias `logg` appaia nell'elenco:

8. Rimuovi l'alias

9. Studia le colonne "Nome alias" e "Comandi resi alias" e assegna correttamente gli alias ai loro valori:

Nome Alias	Commando(i) resi Alias	Assegnazione Alias
`b`	`bash`	
`bash_info`	`which bash` + `echo "$BASH_VERSION"`	
`kernel_info`	`uname -r`	
`greet`	`echo Hi, $USER!`	
`computer`	`pc=slimbook` + `echo My computer is a $pc`	

10. Come `root`, scrivi una funzione chiamata `my_fun` in `/etc/bash.bashrc`. La funzione deve salutare l'utente e dirgli qual è il suo percorso personale. Richiamalo in modo che l'utente riceva entrambi i messaggi ogni volta che accede:

11. Accedi come `user2` per verificare che funzioni:

12. Scrivi la stessa funzione in una sola riga:

13. Richiama la funzione

14. Rimuovi la funzione:

15. Questa è una versione modificata della funzione `special_vars`:

```
$ special_vars2() {
> echo $#
> echo $_
> echo $1
> echo $4
> echo $6
> echo $7
> echo $_
> echo $@
> echo $?
```

```
> }
```

Questo è il comando che usiamo per invocarlo:

```
$ special_vars2 crying cockles and mussels alive alive oh
```

Indovina il risultato:

Riferimento	**Valore**
`echo $#`	
`echo $_`	
`echo $1`	
`echo $4`	
`echo $6`	
`echo $7`	
`echo $_`	
`echo $@`	
`echo $?`	

16. Basandosi sulla funzione di esempio (`check_vids`) nella sezione "Una funzione all'interno di una funzione", scrivi una funzione chiamata `check_music` da includere in uno script di avvio `bash` che accetta parametri posizionali in modo che possiamo modificare facilmente:
    - il tipo di file da controllare: `ogg`
    - la directory in cui vengono salvati i file: `~/Music`
    - il tipo di file da conservare: `music`
    - il numero di file salvati: `7`

Esercizi Esplorativi

1. Le funzioni di sola lettura sono quelle il cui contenuto non è possibile modificare. Fai una ricerca sulle *funzioni di sola lettura* e completa la seguente tabella:

Nome Funzione	**Rendila readonly**	**Mostra tutte le Funzioni readonly**
`my_fun`		

2. Cerca nel web come modificare `PS1` e qualsiasi altra cosa di cui potresti aver bisogno per scrivere una funzione chiamata `fyi` (da inserire in uno script di avvio) che fornisce all'utente le seguenti informazioni:
   - nome dell'utente
   - home directory
   - nome dell'host
   - tipo di sistema operativo
   - percorso di ricerca per eseguibili
   - directory di posta
   - quanto spesso viene controllata la posta
   - quanteo è profonda la sessione corrente in termine di numero di Shell
   - prompt (dovresti modificarlo in modo che mostri `<user>@<host-date>`)

Sommario

In questa lezione abbiamo imparato:

- Sia gli alias sia le funzioni sono caratteristiche importanti della shell che ci consentono di incapsulare blocchi ricorrenti di codice.
- Gli alias sono utili per avere versioni più brevi di comandi lunghi e/o complicati.
- Le funzioni sono procedure che implementano la logica e ci consentono di automatizzare le attività, specialmente se utilizzate negli script.
- La sintassi per scrivere alias e funzioni.
- Come concatenare vari comandi mediante il punto e virgola (`;`).
- Come utilizzare correttamente le virgolette con alias.
- Come rendere persistenti alias e funzioni.
- Variabili speciali *built-in* in Bash: `$?`, `$$`, `$!`, parametri posizionali (`$0`-`$9`), `$#`, `$@`, `$*` e `$_`.
- Come utilizzare variabili e parametri posizionali con le funzioni.
- Come utilizzare le funzioni negli script.
- Come richiamare una funzione da un alias.
- Come richiamare una funzione da un'altra funzione.
- Le basi per creare uno script `bash`.

Comandi utilizzati in questa lezione: `alias`:: Crea alias.

`unalias`
: Rimuove alias.

`cd`
: Cambia directory.

`grep`
: Stampa le linee che corrispondono a un *pattern*.

`function`
: Parola chiave della Shell per creare funzioni.

`.`
: Richiama (*Source*) a file.

`source`

Richiama (*Source*) a file.

`ps`

Mostra un'istantanea dei processi in corso.

`echo`

Visualizza una riga di testo.

`chmod`

Modifica i bit di modalità di un file, per esempio, renderlo eseguibile.

`unset`

Annulla le variabili e le funzioni.

`su`

Cambia lo user ID o diventa superutente.

Risposte agli Esercizi Guidati

1. Completa la tabella con "Sì" o "No" considerando le capacità di alias e funzioni:

Caratteristica	**Alias?**	**Funzioni?**
È possibile utilizzare variabili locali	Sì	Sì
È possibile utilizzare variabili d'ambiente	Sì	Sì
Può essere evitato con `\`	Sì	No
Può essere ricorsivo	Sì	Sì
Molto produttivo se utilizzato con parametri posizionali	No	Sì

2. Immettere il comando che elenca tutti gli alias nel sistema:

```
alias
```

3. Scrivi un alias chiamato `logg` che elenchi tutti i file `ogg` in `~/Music` — uno per riga:

```
alias logg='ls -1 ~/Music/*ogg'
```

4. Richiama l'alias per dimostrare che funziona:

```
logg
```

5. Ora, modifica l'alias in modo che mostri l'utente di sessione e i due punti prima dell'elenco:

```
alias logg='echo $USER:; ls -1 ~/Music/*ogg'
```

6. Richiamalo di nuovo per dimostrare che anche questa nuova versione funziona:

```
logg
```

7. Elenca di nuovo tutti gli alias e controlla che il tuo alias `logg` appaia nell'elenco:

```
alias
```

8. Rimuovi l'alias:

```
unalias logg
```

9. Studia le colonne "Nome alias" e "Comandi resi alias" e assegna correttamente gli alias ai loro valori:

Nome Alias	Commando(i) resi Alias	Assegnazione Alias
`b`	`bash`	`alias b=bash`
`bash_info`	`which bash` + `echo "$BASH_VERSION"`	`alias bash_info='which bash; echo "$BASH_VERSION"'`
`kernel_info`	`uname -r`	`alias kernel_info='uname -r'`
`greet`	`echo Hi, $USER!`	`alias greet='echo Hi, $USER'`
`computer`	`pc=slimbook` + `echo My computer is a $pc`	`alias computer='pc=slimbook; echo My computer is a $pc'`

NOTE Le virgolette singole possono anche essere sostituite da doppie.

10. Come `root`, scrivi una funzione chiamata `my_fun` in `/etc/bash.bashrc`. La funzione deve salutare l'utente e dirgli qual è il suo percorso personale. Richiamalo in modo che l'utente riceva entrambi i messaggi ogni volta che accede:

    Opzione A:

```
my_fun() {
echo Hello, $USER!
echo Your path is: $PATH
}
my_fun
```

Opzione B:

```
function my_fun {
echo Hello, $USER!
echo Your path is: $PATH
}
my_fun
```

11. Accedi come `user2` per verificare che funzioni:

```
su - user2
```

12. Scrivi la stessa funzione in una sola riga:

Opzione A:

```
my_fun() { echo "Hello, $USER!"; echo "Your path is: $PATH"; }
```

Opzione B:

```
function my_fun { echo "Hello, $USER!"; echo "Your path is: $PATH"; }
```

13. Richiama la funzione

```
my_fun
```

14. Rimuovi la funzione:

```
unset -f my_fun
```

15. Questa è una versione modificata della funzione `special_vars`:

```
$ special_vars2() {
> echo $#
> echo $_
> echo $1
> echo $4
> echo $6
```

```
> echo $7
> echo $_
> echo $@
> echo $?
> }
```

Questo è il comando che usiamo per invocarlo:

```
$ special_vars2 crying cockles and mussels alive alive oh
```

Indovina il risultato:

Riferimento	**Valore**
`echo $#`	`7`
`echo $_`	`7`
`echo $1`	`crying`
`echo $4`	`mussels`
`echo $6`	`alive`
`echo $7`	`oh`
`echo $_`	`oh`
`echo $@`	`crying cockles and mussels alive alive oh`
`echo $?`	`0`

16. Basandosi sulla funzione di esempio (`check_vids`) nella sezione “Una funzione all’interno di una funzione”, scrivi una funzione chiamata `check_music` da includere in uno script di avvio `bash` che accetta parametri posizionali in modo che possiamo modificare facilmente:
    - il tipo di file da controllare: `ogg`
    - la directory in cui vengono salvati i file: `~/Music`
    - il tipo di file da conservare: `music`
    - il numero di file salvati: `7`

    ```
    check_music() {
        ls -1 ~/$1/*.$2 > ~/.mkv.log 2>&1
        if [ "$?" = "0" ];then
    ```

```
        echo -e "Remember, you must not keep more than $3 $4 files in your $1
folder.\nThanks."
    else
     echo -e "You do not have any $4 files in the $1 folder. You can keep up to
$3.\nThanks."
    fi
}

check_music Music ogg 7 music
```

Risposte agli Esercizi Esplorativi

1. Le funzioni di sola lettura sono quelle il cui contenuto non è possibile modificare. Fai una ricerca sulle *funzioni di sola lettura* e completa la seguente tabella:

Nome Funzione	**Rendila readonly**	**Mostra tutte le Funzioni readonly**
`my_fun`	`readonly -f my_fun`	`readonly -f`

2. Cerca nel web come modificare `PS1` e qualsiasi altra cosa di cui potresti aver bisogno per scrivere una funzione chiamata `fyi` (da inserire in uno script di avvio) che fornisce all'utente le seguenti informazioni:
   - nome dell'utente
   - home directory
   - nome dell'host
   - tipo di sistema operativo
   - percorso di ricerca per eseguibili
   - directory di posta
   - quanto spesso viene controllata la posta
   - quanteo è profonda la sessione corrente in termine di numero di Shell
   - prompt (dovresti modificarlo in modo che mostri `<user>@<host-date>`)PP

```
fyi() {
    echo -e "For your Information:\n
    Username: $USER
    Home directory: $HOME
    Host: $HOSTNAME
    Operating System: $OSTYPE
    Path for executable files: $PATH
    Your mail directory is $MAIL and is searched every $MAILCHECK seconds.
    The current level of your shell is: $SHLVL"
    PS1="\u@\h-\d "
}

fyi
```



105.2 Personalizzare o scrivere semplici script

Obiettivi LPI di riferimento

LPIC-1 5.0, Exam 102, Objective 105.2

Peso

4

Aree di Conoscenza Chiave

- Usare la sintassi sh standard (loop, test).
- Usare la sostituzione dei comandi.
- Testare i valori di uscita in base al risultato o altre informazioni fornite da un comando.
- Eseguire comandi concatenati.
- Eseguire il mailing condizionale al superuser.
- Selezionare correttamente l'interprete di script tramite la riga shebang (`#!`).
- Gestire la posizione, la proprietà, l'esecuzione e i diritti suid degli script.

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

- `for`
- `while`
- `test`
- `if`
- `read`
- `seq`
- `exec`

- `||`
- `&&`



105.2 Lezione 1

Certificazione:	LPIC-1
Versione:	5.0
Argomento:	105 Shell e Script di Shell
Obiettivo:	105.2 Personalizzare o scrivere semplici script
Lezione:	1 di 2

Introduzione

L'ambiente shell di Linux consente l'uso di file — chiamati *script* — contenenti comandi disponibili nel sistema in combinazione con comandi incorporati nella shell per automatizzare le attività di un utente e/o di un sistema. In effetti, molte delle attività di manutenzione del sistema operativo vengono eseguite da script costituiti da sequenze di comandi, strutture decisionali e cicli condizionali. Sebbene gli script siano per la maggior parte del tempo destinati ad attività relative al sistema operativo stesso, sono utili anche per attività orientate all'utente, come la ridenominazione di massa dei file, la raccolta e l'analisi dei dati o qualsiasi attività da riga di comando altrimenti ripetitiva.

Gli script non sono altro che file di testo che si comportano come programmi. L'*interprete*, un programma vero e proprio, legge ed esegue le istruzioni elencate nel file di script. L'interprete può anche avviare una sessione interattiva in cui i comandi, inclusi gli script, vengono letti ed eseguiti non appena vengono immessi, come nel caso delle sessioni della shell di Linux. I file di script possono raggruppare tali istruzioni e comandi quando diventano troppo complessi per essere implementati come alias o come funzione di shell personalizzata. Inoltre, i file di script possono essere mantenuti come i programmi convenzionali e, essendo solo file di testo, possono essere creati e modificati con qualsiasi semplice editor di testo.

Struttura degli Script ed Esecuzione

Fondamentalmente, un file di script è una sequenza ordinata di comandi che deve essere eseguita da un interprete di comandi corrispondente. Il modo in cui un interprete legge un file di script varia e ci sono modi diversi per farlo in una sessione della shell Bash, ma l'interprete predefinito per un file di script sarà quello indicato nella prima riga dello script, subito dopo i caratteri `#!`(noto come *shebang*). In uno script con le istruzioni per la shell Bash, la prima riga dovrebbe essere `#!/bin/bash`. Indicando questa riga, l'interprete per tutte le istruzioni nel file sarà `/bin/bash`. Ad eccezione della prima riga, tutte le altre righe che iniziano con il carattere cancelletto "#" verranno ignorate, quindi possono essere utilizzate per inserire promemoria e commenti. Anche le righe vuote vengono ignorate. Un file script di shell molto conciso può quindi essere scritto come segue:

```
#!/bin/bash

# Uno script molto semplice

echo "Cheers from the script file! Current time is: "

date +%H:%M
```

Questo script ha solo due istruzioni per l'interprete `/bin/bash`: il comando integrato `echo` e il comando `date`. Il modo più semplice per eseguire un file di script è eseguire l'interprete con il percorso dello script come argomento. Quindi, supponendo che l'esempio precedente sia stato salvato in un file di script chiamato `script.sh` nella directory corrente, verrà letto e interpretato da Bash con il seguente comando:

```
$ bash script.sh
Cheers from the script file! Current time is:
10:57
```

Il comando `echo` aggiungerà automaticamente una nuova riga dopo aver visualizzato il contenuto, ma l'opzione `-n` sopprimerà questo comportamento. Quindi, usando `echo -n` nello script, l'output di entrambi i comandi apparirà nella stessa riga:

```
$ bash script.sh
Cheers from the script file! Current time is: 10:57
```

Sebbene non sia richiesto, il suffisso `.sh` aiuta a identificare gli script di shell quando si elencano

e si cercano i file.

TIP

Bash chiamerà qualunque comando sia indicato dopo `#!` come interprete per il file di script. Può essere utile, per esempio, utilizzare lo *shebang* per altri linguaggi di scripting, come *Python* (`#!/usr/bin/python`), *Perl* (`#!/usr/bin/perl`) o *awk* (`#!/usr/bin/awk`).

Se il file di script deve essere eseguito da altri utenti nel sistema, è importante verificare se sono impostate le autorizzazioni di lettura corrette. Il comando `chmod o+r script.sh` darà il permesso di lettura a tutti gli utenti nel sistema, consentendo loro di eseguire `script.sh` inserendo il percorso del file script come argomento del comando `bash`. In alternativa, il file di script può avere il permesso di esecuzione impostato in modo che il file possa essere eseguito come un comando convenzionale. Il bit di esecuzione viene attivato sul file di script con il comando `chmod`:

```
$ chmod +x script.sh
```

Con il bit di esecuzione abilitato, il file di script denominato `script.sh` nella directory corrente può essere eseguito direttamente con il comando `./script.sh`. Gli script inseriti in una directory elencata nella variabile d'ambiente `PATH` saranno accessibili anche senza il loro percorso completo.

WARNING

Uno script che esegue azioni limitate può avere il suo permesso SUID attivato, quindi gli utenti ordinari possono anche eseguire lo script con i privilegi di root. In questo caso, è molto importante assicurarsi che nessun utente diverso da root abbia il permesso di scrivere nel file. In caso contrario, un utente normale potrebbe modificare il file per eseguire operazioni arbitrarie e potenzialmente dannose.

Il posizionamento e il rientro dei comandi nei file di script non sono troppo rigidi. Ogni riga in uno script di shell verrà eseguita come un normale comando di shell, nella stessa sequenza in cui la riga appare nel file di script e le stesse regole che si applicano al prompt della shell si applicano anche a ciascuna riga di script individualmente. È possibile inserire due o più comandi nella stessa riga, separati da punto e virgola:

```
echo "Cheers from the script file! Current time is:" ; date +%H:%M
```

Sebbene questo formato possa essere a volte conveniente, il suo utilizzo è facoltativo, poiché i comandi sequenziali possono essere inseriti un comando per riga e verranno eseguiti proprio come se fossero separati da punto e virgola. In altre parole, il punto e virgola può essere sostituito

da un nuovo carattere di riga nei file di script Bash.

Quando viene eseguito uno script, i comandi in esso contenuti non vengono eseguiti direttamente nella sessione corrente, ma vengono invece eseguiti da un nuovo processo Bash, chiamato *sub-shell*. Impedisce allo script di sovrascrivere le variabili di ambiente della sessione corrente e di lasciare le modifiche automatiche nella sessione corrente. Se l'obiettivo è eseguire il contenuto dello script nella sessione corrente della shell, allora dovrebbe essere eseguito con `source script.sh` o `. script.sh` (nota che c'è uno spazio tra il punto e il nome dello script).

Come accade con l'esecuzione di qualsiasi altro comando, il prompt della shell sarà nuovamente disponibile solo quando lo script avrà terminato la sua esecuzione e il suo codice di stato di uscita sarà disponibile nella variabile `$?`. Per cambiare questo comportamento, in modo che anche la shell corrente termini quando lo script finisce, lo script — o qualsiasi altro comando — può essere preceduto dal comando `exec`. Questo comando sostituirà anche il codice dello stato di uscita della sessione di shell corrente con il proprio.

Variabili

Le variabili negli script di shell si comportano allo stesso modo delle sessioni interattive, dato che l'interprete è lo stesso. Per esempio, il formato `SOLUTION=42` (senza spazi attorno al segno di uguale) assegnerà il valore `42` alla variabile denominata `SOLUTION`. Per convenzione, le lettere maiuscole vengono utilizzate per i nomi delle variabili, ma non è obbligatorio. Tuttavia, i nomi delle variabili non possono iniziare con caratteri non alfabetici.

Oltre alle normali variabili create dall'utente, gli script Bash hanno anche una serie di variabili speciali chiamate parametri o *parameters*. A differenza delle variabili ordinarie, i nomi dei parametri iniziano con un carattere non alfabetico che ne designa la funzione. Gli argomenti passati a uno script e altre informazioni utili sono memorizzati in parametri come `$0`, `$*`, `$?`, ecc., Dove il carattere che segue il segno del dollaro indica le informazioni da recuperare:

`$*`

Tutti gli argomenti passati allo script.

`$@`

Tutti gli argomenti passati allo script. Se usato con virgolette doppie, come in `"$@"`, ogni argomento sarà racchiuso tra virgolette doppie.

`$#`

Il numero degli argomenti.

`$0`

Il nome del file di script.

`$!`

PID dell'ultimo programma eseguito.

`$$`

PID della shell corrente.

`$?`

Codice numerico di stato di uscita dell'ultimo comando terminato. Per i processi standard POSIX, un valore numerico di `0` significa che l'ultimo comando è stato eseguito con successo, il che si applica anche agli script di shell.

Un *parametro posizionale* è un parametro indicato da una o più cifre, diverse dalla singola cifra `0`. Per esempio, la variabile `$1` corrisponde al primo argomento fornito allo script (parametro posizionale uno), `$2` corrisponde al secondo argomento e così via. Se la posizione di un parametro è maggiore di nove, deve essere referenziato con parentesi graffe, come in `${10}`, `${11}`, ecc.

Le variabili ordinarie, invece, hanno lo scopo di memorizzare valori inseriti manualmente o l'output generato da altri comandi. Il comando `read`, per esempio, può essere utilizzato all'interno dello script per chiedere all'utente un input durante l'esecuzione dello script:

```
echo "Do you want to continue (y/n)?"
read ANSWER
```

Il valore restituito verrà memorizzato nella variabile `ANSWER`. Se il nome della variabile non viene fornito, per impostazione predefinita verrà utilizzato il nome della variabile `REPLY`. È anche possibile utilizzare il comando `read` per leggere più di una variabile contemporaneamente:

```
echo "Type your first name and last name:"
read NAME SURNAME
```

In questo caso, ogni termine separato da spazi verrà assegnato rispettivamente alle variabili `NAME` e `SURNAME`. Se il numero di termini dati è maggiore del numero di variabili, i termini in eccesso verranno memorizzati nell'ultima variabile. Lo stesso `read` può mostrare il messaggio all'utente con l'opzione `-p`, rendendo ridondante il comando `echo` in questo caso:

```
read -p "Type your first name and last name:" NAME SURNAME
```

Gli script che eseguono attività di sistema richiedono spesso informazioni fornite da altri programmi. La notazione *backtick* può essere utilizzata per memorizzare l'output di un comando in una variabile:

```
$ OS=`uname -o`
```

Nell'esempio, l'output del comando `uname -o` sarà memorizzato nella variabile `OS`. Un risultato identico verrà prodotto con `$()`:

```
$ OS=$(uname -o)
```

La lunghezza di una variabile, ovvero la quantità di caratteri che contiene, viene restituita anteponendo un cancelletto `#` prima del nome della variabile. Questa caratteristica, tuttavia, richiede l'uso della sintassi delle parentesi graffe per indicare la variabile:

```
$ OS=$(uname -o)
$ echo $OS
GNU/Linux
$ echo ${#OS}
9
```

Bash dispone anche di *variabili array unidimensionali*, quindi un insieme di elementi correlati può essere memorizzato con un unico nome di variabile. Ogni elemento in un array ha un indice numerico, che deve essere utilizzato per scrivere e leggere i valori nell'elemento corrispondente. Diversamente dalle variabili ordinarie, gli array devono essere dichiarati con il comando integrato di Bash `declare`. Per esempio, per dichiarare una variabile denominata `SIZES` come un array:

```
$ declare -a SIZES
```

Gli array possono anche essere dichiarati implicitamente quando popolati da un elenco predefinito di elementi, utilizzando la notazione tra parentesi:

```
$ SIZES=( 1048576 1073741824 )
```

Nell'esempio, i due valori interi grandi sono stati memorizzati nell'array `SIZES`. Gli elementi della matrice devono essere referenziati utilizzando parentesi graffe e parentesi quadre, altrimenti Bash non modificherà o visualizzerà l'elemento correttamente. Poiché gli indici dell'array iniziano

da 0, il contenuto del primo elemento è in `${SIZES[0]}`, il secondo elemento è in `${SIZES[1]}` e così via:

```
$ echo ${SIZES[0]}
1048576
$ echo ${SIZES[1]}
1073741824
```

A differenza della lettura, la modifica del contenuto di un elemento dell'array viene eseguita senza le parentesi graffe (Per esempio, `SIZES[0]=1048576`). Come con le variabili ordinarie, la lunghezza di un elemento in un array viene restituita con il carattere cancelletto (per esempio, `${#SIZES[0]}` per la lunghezza del primo elemento nell'array `SIZES`). Il numero totale di elementi in un array viene restituito se `@` o `*` sono usati come indice:

```
$ echo ${#SIZES[@]}
2
$ echo ${#SIZES[*]}
2
```

Gli array possono anche essere dichiarati utilizzando l'output di un comando come elementi iniziali tramite la sostituzione del comando. Il seguente esempio mostra come creare un array Bash i cui elementi sono i filesystem supportati dal sistema corrente:

```
$ FS=( $(cut -f 2 < /proc/filesystems) )
```

Il comando `cut -f 2 < /proc/filesystems` mostrerà tutti i filesystem attualmente supportati dal kernel in esecuzione (come elencato nella seconda colonna del file `/proc/filesystems`), quindi l'array `FS` ora contiene un elemento per ogni filesystem supportato. Qualsiasi contenuto di testo può essere utilizzato per inizializzare un array poiché, per impostazione predefinita, qualsiasi termine delimitato da caratteri *space*, *tab* o *newline* diventerà un elemento dell'array.

TIP

Bash tratta ogni carattere di `$IFS` (*Input Field Separator*) di una variabile d'ambiente come un delimitatore. Per cambiare il delimitatore di campo solo con caratteri di nuova riga, per esempio, la variabile IFS dovrebbe essere reimpostata con il comando `IFS=$'\n'`.

Espressioni Aritmetiche

Bash fornisce un metodo pratico per eseguire operazioni aritmetiche su interi con il comando

incorporato `expr`. Due variabili numeriche, per esempio `$VAL1` e `$VAL2`, possono essere aggiunte insieme al seguente comando:

```
$ SUM=`expr $VAL1 + $VAL2`
```

Il valore risultante dell'esempio sarà disponibile nella variabile `$SUM`. Il comando `expr` può essere sostituito da `$(())`, quindi l'esempio precedente può essere riscritto come `SUM=$(($VAL1+$VAL2))`. Le espressioni di potenza sono consentite anche con l'operatore doppio asterisco, quindi la precedente dichiarazione di array `SIZES=(1048576 1073741824)` potrebbe essere riscritta come `SIZES=( $((1024**2)) $((1024**3)) )`.

La sostituzione dei comandi può essere utilizzata anche nelle espressioni aritmetiche. Per esempio, il file `/proc/meminfo` contiene informazioni dettagliate sulla memoria di sistema, incluso il numero di byte liberi nella RAM:

```
$ FREE=$(( 1000 * `sed -nre '2s/[^[:digit:]]//gp' < /proc/meminfo` ))
```

L'esempio mostra come utilizzare il comando `sed` per analizzare il contenuto di `/proc/meminfo` all'interno dell'espressione aritmetica. La seconda riga del file `/proc/meminfo` contiene la quantità di memoria libera in migliaia di byte, quindi l'espressione aritmetica la moltiplica per 1000 per ottenere il numero di byte liberi nella RAM.

Esecuzione Condizionale

Alcuni script di solito non hanno lo scopo di eseguire tutti i comandi nel file di script, ma solo quei comandi che corrispondono a criteri predefiniti. Per esempio, uno script di manutenzione può inviare un messaggio di avviso all'e-mail dell'amministratore solo se l'esecuzione di un comando non riesce. Bash fornisce metodi specifici per valutare il successo dell'esecuzione dei comandi e strutture condizionali generali, più simili a quelle che si trovano nei linguaggi di programmazione più diffusi.

Separando i comandi con `&&`, il comando a destra verrà eseguito solo se il comando a sinistra non ha riscontrato un errore, cioè se il suo stato di uscita era uguale a `0`:

```
COMMAND A && COMMAND B && COMMAND C
```

Il comportamento opposto si verifica se i comandi sono separati da `||`. In questo caso, il comando seguente verrà eseguito solo se il comando precedente ha riscontrato un errore, cioè se il suo codice di stato restituito è diverso da 0.

Una delle caratteristiche più importanti di tutti i linguaggi di programmazione è la capacità di eseguire comandi in base a condizioni definite in precedenza. Il modo più semplice per eseguire comandi in modo condizionale è usare il comando incorporato di Bash `if`, che esegue uno o più comandi solo se il comando dato come argomento restituisce un codice di stato 0 (successo). Un altro comando, `test`, può essere usato per valutare molti differenti criteri speciali, quindi è usato principalmente insieme a `if`. Nell'esempio seguente, il messaggio `Confirmed: /bin/bash is executable.` verrà visualizzato solo se il file `/bin/bash` esiste ed è eseguibile:

```
if test -x /bin/bash ; then
  echo "Confirmed: /bin/bash is executable."
fi
```

L'opzione `-x` fa sì che il comando `test` restituisca un codice di stato 0 solo se il percorso specificato è un file eseguibile. L'esempio seguente mostra un altro modo per ottenere lo stesso identico risultato, poiché le parentesi quadre possono essere utilizzate in sostituzione di `test`:

```
if [ -x /bin/bash ] ; then
  echo "Confirmed: /bin/bash is executable."
fi
```

L'istruzione `else` è opzionale alla struttura `if` e può, se presente, definire un comando o una sequenza di comandi da eseguire se l'espressione condizionale non è vera:

```
if [ -x /bin/bash ] ; then
  echo "Confirmed: /bin/bash is executable."
else
  echo "No, /bin/bash is not executable."
fi
```

Le strutture `if` devono sempre finire con `fi`, quindi l'interprete di Bash sa dove terminano i comandi condizionali.

Output di uno Script

Anche quando lo scopo di uno script coinvolge solo operazioni orientate ai file, è importante visualizzare i messaggi relativi allo stato di avanzamento nell'*output standard*, in modo che l'utente sia informato di eventuali problemi e possa eventualmente utilizzare quei messaggi per generare i log delle operazioni.

Il comando integrato di Bash `echo` è comunemente usato per visualizzare semplici stringhe di testo, ma fornisce anche alcune funzionalità estese. Con l'opzione `-e`, il comando `echo` è in grado di visualizzare caratteri speciali usando sequenze di *escape* (una sequenza di *backslash* che designa un carattere speciale). Per esempio:

```
#!/bin/bash

# Get the operating system's generic name
OS=$(uname -o)

# Get the amount of free memory in bytes
FREE=$(( 1000 * `sed -nre '2s/[^[:digit:]]//gp' < /proc/meminfo` ))

echo -e "Operating system:\t$OS"
echo -e "Unallocated RAM:\t$(( $FREE / 1024**2 )) MB"
```

Sebbene l'uso delle virgolette sia opzionale quando si usa `echo` senza opzioni, è necessario aggiungerle quando si usa l'opzione `-e`, altrimenti i caratteri speciali potrebbero non essere visualizzati correttamente. Nello script precedente, entrambi i comandi `echo` usano il carattere di tabulazione `\t` per allineare il testo, ottenendo il seguente output:

```
Operating system:       GNU/Linux
Unallocated RAM:        1491 MB
```

Il carattere di nuova riga `\n` può essere utilizzato per separare le righe di output, quindi lo stesso identico output si ottiene combinando i due comandi `echo` in uno solo:

```
echo -e "Operating system:\t$OS\nUnallocated RAM:\t$(( $FREE / 1024**2 )) MB"
```

Sebbene sia adatto per visualizzare la maggior parte dei messaggi di testo, il comando `echo` potrebbe non essere adatto per visualizzare modelli di testo più specifici. Il comando integrato di Bash `printf` dà più controllo su come visualizzare le variabili. Il comando `printf` usa il primo argomento come formato dell'output, dove i segnaposto saranno sostituiti dai seguenti argomenti nell'ordine in cui appaiono nella riga di comando. Per esempio, il messaggio dell'esempio precedente potrebbe essere generato con il seguente comando `printf`:

```
printf "Operating system:\t%s\nUnallocated RAM:\t%d MB\n" $OS $(( $FREE / 1024**2 ))
```

Il segnaposto `%s` è destinato per il contenuto di testo (sarà rimpiazzato dalla variabile `$OS`) e il

segnaposto `%d` è destinato a numeri interi (sarà rimpiazzato dal numero risultante di megabyte liberi nella RAM). `printf` non aggiunge un carattere di nuova riga alla fine del testo, quindi il carattere di nuova riga `\n` dovrebbe essere posizionato alla fine del modello se necessario. L'intero modello deve essere interpretato come un singolo argomento, quindi deve essere racchiuso tra virgolette.

TIP

Il formato di sostituzione del segnaposto eseguita da `printf` può essere personalizzato usando lo stesso formato usato dalla funzione `printf` dal linguaggio di programmazione C. Il riferimento completo per la funzione `printf` può essere trovato nella sua pagina di manuale, accessibile con il comando `man 3 printf`.

Con `printf`, le variabili sono poste al di fuori del *pattern* di testo, il che rende possibile memorizzarlo in una variabile separata:

```
MSG='Operating system:\t%s\nUnallocated RAM:\t%d MB\n'
printf "$MSG" $OS $(( $FREE / 1024**2 ))
```

Questo metodo è particolarmente utile per visualizzare formati di output distinti, a seconda delle esigenze dell'utente. Rende più semplice, per esempio, scrivere uno script che utilizza un *pattern* di testo distinto se l'utente richiede un elenco CSV (*Comma Separated Values*) piuttosto che un messaggio di output predefinito.

Esercizi Guidati

1. L'opzione `-s` per il comando `read` è utile per inserire le password, poiché non mostrerà il contenuto digitato sullo schermo. Come potrebbe essere usato il comando `read` per memorizzare l'input dell'utente nella variabile `PASSWORD` nascondendo il contenuto digitato?

2. L'unico scopo del comando `whoami` è quello di visualizzare il nome utente dell'utente che lo ha eseguito, quindi viene utilizzato principalmente all'interno degli script per identificare l'utente che lo sta eseguendo. All'interno di uno script Bash, come potrebbe essere memorizzato l'output del comando `whoami` nella variabile chiamata `WHO`?

3. Quale operatore Bash dovrebbe esserci tra i comandi `apt-get dist-upgrade` e `systemctl reboot` se l'utente root vuole eseguire `systemctl reboot` solo se `apt-get dist-upgrade` è terminato con successo?

Esercizi Esplorativi

1. Dopo aver tentato di eseguire uno script Bash appena creato, un utente riceve il seguente messaggio di errore:

   ```
   bash: ./script.sh: Permission denied
   ```

   Considerando che il file `./script.sh` è stato creato dallo stesso utente, quale potrebbe essere la probabile causa di questo errore?

2. Supponiamo che un file di script chiamato `do.sh` sia eseguibile e il collegamento simbolico chiamato `undo.sh` punti ad esso. Dall'interno dello script, come potresti identificare se il nome del file chiamante era `do.sh` o `undo.sh`?

3. In un sistema con un servizio di posta elettronica configurato correttamente, il comando `mail -s "Maintenance Error" root <<<"Scheduled task error"` invia il messaggio di posta elettronica di avviso all'utente root. Tale comando potrebbe essere utilizzato in attività non assistite, come *cronjob*, per informare l'amministratore di sistema di un problema imprevisto. Scrivete un costrutto *if* che eseguirà il suddetto comando `mail` se lo stato di uscita del comando precedente - -qualunque esso sia — non ha successo.

Sommario

Questa lezione copre i concetti base per la comprensione e la scrittura di script nella shell Bash. Gli script di shell sono una parte fondamentale di qualsiasi distribuzione Linux in quanto offrono un modo molto flessibile per automatizzare le attività dell'utente e del sistema. La lezione segue i seguenti passaggi:

- Struttura dello script di shell e autorizzazioni corrette del file di script
- Parametri di script
- Utilizzo di variabili per leggere l'input dell'utente e memorizzare l'output dei comandi
- Bash array
- Test semplici ed esecuzione condizionale
- Formattazione dell'output

I comandi e le procedure affrontati sono stati:

- Notazione incorporata di Bash per la sostituzione dei comandi, espansione di array e espressioni aritmetiche
- Esecuzione condizionale di comandi con gli operatori `||` e `&&`
- `echo`
- `chmod`
- `exec`
- `read`
- `declare`
- `test`
- `if`
- `printf`

Risposte agli Esercizi Guidati

1. L'opzione `-s` per il comando `read` è utile per inserire le password, poiché non mostrerà il contenuto digitato sullo schermo. Come potrebbe essere usato il comando `read` per memorizzare l'input dell'utente nella variabile `PASSWORD` nascondendo il contenuto digitato?

   `read -s PASSWORD`

2. L'unico scopo del comando `whoami` è quello di visualizzare il nome utente dell'utente che lo ha eseguito, quindi viene utilizzato principalmente all'interno degli script per identificare l'utente che lo sta eseguendo. All'interno di uno script Bash, come potrebbe essere memorizzato l'output del comando `whoami` nella variabile chiamata `WHO`?

   `WHO=\`whoami\`` o `WHO=$(whoami)`

3. Quale operatore Bash dovrebbe esserci tra i comandi `apt-get dist-upgrade` e `systemctl reboot` se l'utente root vuole eseguire `systemctl reboot` solo se `apt-get dist-upgrade` è terminato con successo?

   L'operatore `&&`, come in `apt-get dist-upgrade && systemctl reboot`.

Risposte agli Esercizi Esplorativi

1. Dopo aver tentato di eseguire uno script Bash appena creato, un utente riceve il seguente messaggio di errore:

   ```
   bash: ./script.sh: Permission denied
   ```

   Considerando che il file `./script.sh` è stato creato dallo stesso utente, quale potrebbe essere la probabile causa di questo errore?

   Il file `./script.sh` non possiede il permesso di esecuzione attivo.

2. Supponiamo che un file di script chiamato `do.sh` sia eseguibile e il collegamento simbolico chiamato `undo.sh` punti ad esso. Dall'interno dello script, come potresti identificare se il nome del file chiamante era `do.sh` o `undo.sh`?

   La variabile speciale `$0` contiene il nome del file utilizzato per richiamare lo script.

3. In un sistema con un servizio di posta elettronica configurato correttamente, il comando `mail -s "Maintenance Error" root <<<"Scheduled task error"` invia il messaggio di posta elettronica di avviso all'utente root. Tale comando potrebbe essere utilizzato in attività non assistite, come *cronjob*, per informare l'amministratore di sistema di un problema imprevisto. Scrivete un costrutto *if* che eseguirà il suddetto comando `mail` se lo stato di uscita del comando precedente - -qualunque esso sia — non ha successo.

   ```
   if [ "$?" -ne 0 ]; then mail -s "Maintenance Error" root <<<"Scheduled task error"; fi
   ```

Linux Professional Institute

105.2 Lezione 2

Certificazione:	LPIC-1
Versione:	5.0
Argomento:	105 Shell e Script di Shell
Obiettivo:	105.2 Personalizzare o scrivere semplici script
Lezione:	2 di 2

Introduzione

Gli script di shell sono generalmente destinati ad automatizzare le operazioni relative a file e directory, le stesse operazioni che potrebbero essere eseguite manualmente dalla riga di comando. Tuttavia, la copertura degli script di shell non è limitata ai soli documenti di un utente, poiché anche la configurazione e l'interazione con molti aspetti di un sistema operativo Linux vengono eseguite tramite file di script. La *shell Bash* offre molti utili comandi *builtin* per scrivere script di shell, ma la piena potenza di questi script si basa sulla combinazione dei comandi incorporati di Bash con le numerose utilità della riga di comando disponibili su un sistema Linux.

Test Estesi

Bash, come linguaggio di script, è principalmente orientato a lavorare con i file, quindi il comando integrato di Bash `test` ha molte opzioni per valutare le proprietà degli oggetti del filesystem (essenzialmente file e directory). I test incentrati su file e directory sono utili, per esempio, per verificare se i file e le directory necessari per eseguire una determinata attività sono presenti e possono essere letti. Quindi, associato a un costrutto condizionale *if*, il set di azioni appropriato viene eseguito se il test ha esito positivo.

Il comando `test` può valutare espressioni usando due diverse sintassi: le espressioni di prova possono essere fornite come argomento del comando `test` oppure possono essere inserite tra parentesi quadre, dove il comando `test` è dato implicitamente. Quindi, il test per valutare se `/etc` è una directory valida può essere scritto come `test -d /etc` o come `[-d /etc]`:

```
$ test -d /etc
$ echo $?
0
$ [ -d /etc ]
$ echo $?
0
```

Come confermato dai codici di stato di uscita nella variabile speciale `$?` — un valore di 0 significa che il test ha avuto successo — entrambe le forme hanno valutato `/etc` come una directory valida. Supponendo che il percorso di un file o di una directory sia stato memorizzato nella variabile `$VAR`, le seguenti espressioni possono essere utilizzate come argomenti per `test` o racchiuse tra parentesi quadre:

`-a "$VAR"`
: Valuta se il percorso in `VAR` esiste nel filesystem ed è un file.

`-b "$VAR"`
: Valuta se il percorso in `VAR` è un file speciale a blocchi.

`-c "$VAR"`
: Valuta se il percorso in `VAR` è un file speciale di caratteri.

`-d "$VAR"`
: Valuta se il percorso in `VAR` è una directory.

`-e "$VAR"`
: Valuta se il percorso in `VAR` esiste nel filesystem.

`-f "$VAR"`
: Valuta se il percorso in `VAR` esiste ed è un file normale.

`-g "$VAR"`
: Valuta se il percorso in `VAR` ha il permesso SGID.

`-h "$VAR"`
Valuta se il percorso in `VAR` è un collegamento simbolico.

`-L "$VAR"`
Valuta se il percorso in `VAR` è un collegamento simbolico. (come `-h`).

`-k "$VAR"`
Valuta se il percorso in `VAR` ha il permesso *sticky* bit.

`-p "$VAR"`
Valuta se il percorso in `VAR` è un file *pipe*.

`-r "$VAR"`
Valuta se il percorso in `VAR` è leggibile dall'utente corrente.

`-s "$VAR"`
Valuta se il percorso in `VAR` esiste e non è vuoto.

`-S "$VAR"`
Valuta se il percorso in `VAR` è un file socket.

`-t "$VAR"`
Valuta se il percorso in `VAR` è aperto in un terminale.

`-u "$VAR"`
Valuta se il percorso in `VAR` ha il permesso SUID.

`-w "$VAR"`
Valuta se il percorso in `VAR` è scrivibile dall'utente corrente.

`-x "$VAR"`
Valuta se il percorso in `VAR` è eseguibile dall'utente corrente.

`-O "$VAR"`
Valuta se il percorso in `VAR` è di proprietà dell'utente corrente.

`-G "$VAR"`
Valuta se il percorso in `VAR` appartiene al gruppo effettivo dell'utente corrente.

`-N "$VAR"`

Valuta se il percorso in `VAR` è stato modificato dall'ultimo accesso.

`"$VAR1" -nt "$VAR2"`

Valuta se il percorso in `VAR1` è più recente del percorso in `VAR2`, in base alle date di modifica.

`"$VAR1" -ot "$VAR2"`

Valuta se il percorso in `VAR1` è più vecchio di `VAR2`.

`"$VAR1" -ef "$VAR2"`

Questa espressione restituisce *True* se il percorso in `VAR1` è un hardlink a `VAR2`.

Si consiglia di utilizzare le virgolette doppie attorno a una variabile *testata* perché, se la variabile risulta essere vuota, potrebbe causare un errore di sintassi per il comando `test`. Le opzioni di test richiedono un argomento e una variabile vuota non messa tra virgolette causerebbe un errore a causa di un argomento obbligatorio mancante. Esistono anche test per variabili di testo arbitrarie, descritti come segue:

`-z "$TXT"`

Valuta se la variabile `TXT` è vuota (dimensione zero).

`-n "$TXT"` or `test "$TXT"`

Valuta se la variabile `TXT` non è vuota.

`"$TXT1" = "$TXT2"` o `"$TXT1" == "$TXT2"`

Valuta se `TXT1` e `TXT2` sono uguali.

`"$TXT1" != "$TXT2"`

Valuta se `TXT1` e `TXT2` non sono uguali.

`"$TXT1" < "$TXT2"`

Valuta se `TXT1` viene prima di `TXT2`, in ordine alfabetico.

`"$TXT1" > "$TXT2"`

Valuta se `TXT1` viene dopo `TXT2`, in ordine alfabetico.

Lingue diverse possono avere regole diverse per l'ordine alfabetico. Per ottenere risultati coerenti, indipendentemente dalle impostazioni di localizzazione del sistema in cui viene eseguito lo script, si consiglia di impostare la variabile d'ambiente `LANG` su `C`, come in `LANG=C`, prima di eseguire operazioni che implicano l'ordine alfabetico . Questa definizione manterrà anche i messaggi di sistema nella lingua originale, quindi dovrebbe essere utilizzata solo nell'ambito dello script.

I confronti numerici hanno il proprio set di opzioni di test:

`$NUM1 -lt $NUM2`
Valuta se `NUM1` è minore di `NUM2`.

`$NUM1 -gt $NUM2`
Valuta se `NUM1` è maggiore di `NUM2`.

`$NUM1 -le $NUM2`
Valuta se `NUM1` è minore o uguale a `NUM2`.

`$NUM1 -ge $NUM2`
Valuta se `NUM1` è maggiore o uguale a `NUM2`.

`$NUM1 -eq $NUM2`
Valuta se `NUM1` è uguale a `NUM2`.

`$NUM1 -ne $NUM2`
Valuta se `NUM1` non è uguale a `NUM2`.

Tutti i test possono ricevere i seguenti modificatori:

`! EXPR`
Valuta se l'espressione `EXPR` è falsa.

`EXPR1 -a EXPR2`
Valuta se sia `EXPR1` che `EXPR2` sono veri.

`EXPR1 -o EXPR2`
Valuta se almeno una delle due espressioni è vera.

Un altro costrutto condizionale, `case`, può essere visto come una variazione del costrutto *if*. L'istruzione `case` eseguirà un elenco di comandi dati se un elemento specificato — il contenuto di una variabile, per esempio — può essere trovato in un elenco di elementi separati da *pipes* e terminato da `)`. Il seguente script di esempio mostra come il costrutto `case` pussa essere usato per indicare il formato di pacchettizzazione del software corrispondente per una data distribuzione Linux:

```
#!/bin/bash

DISTRO=$1
```

```
echo -n "Distribution $DISTRO uses "
case "$DISTRO" in
    debian | ubuntu | mint)
    echo -n "the DEB"
  ;;
    centos | fedora | opensuse )
    echo -n "the RPM"
  ;;
    *)
    echo -n "an unknown"
  ;;
esac
echo " package format."
```

Ogni elenco di pattern e comandi associati deve terminare con `;;`, `;&`, o `;;&`. L'ultimo pattern, un asterisco, corrisponderà se nessun altro pattern precedente corrispondeva in precedenza. L'istruzione `esac` (*case* al contrario) termina il costrutto `case`. Supponendo che lo script di esempio precedente fosse chiamato `script.sh` e che venga eseguito su *OpenSuse* come primo argomento, verrà generato il seguente output:

```
$ ./script.sh opensuse
Distribution opensuse uses the RPM package format.
```

TIP

Bash ha un'opzione chiamata `nocasematch` che abilita il *pattern matching* senza distinzione tra maiuscole e minuscole per il costrutto `case` e altri comandi condizionali. Il comando integrato `shopt` alterna i valori delle impostazioni che controllano il comportamento opzionale della shell: `shopt -s` abiliterà (*set*) l'opzione data e `shopt -u` disabiliterà (*unset*) l'opzione data. Pertanto, posizionare `shopt -s nocasematch` prima del costrutto case abiliterà la corrispondenza dei modelli senza distinzione tra maiuscole e minuscole. Le opzioni modificate da `shopt` influenzeranno solo la sessione corrente, quindi le opzioni modificate all'interno di script in esecuzione in una sub-shell — che è il modo standard per eseguire uno script — non influenzano le opzioni della sessione *parent*.

L'elemento ricercato e gli schemi subiscono l'espansione della tilde (~), l'espansione dei parametri, la sostituzione dei comandi e l'espansione aritmetica. Se l'elemento cercato è specificato tra virgolette, verranno rimossi prima che venga tentata la corrispondenza.

Costrutti di Loop

Gli script vengono spesso utilizzati come strumento per automatizzare attività ripetitive, eseguendo lo stesso set di comandi fino a quando non viene verificato un criterio di interruzione. Bash ha tre istruzioni di ciclo — `for`, `until` e `while` — progettate per specifici costrutti di loop.

Il costrutto `for` percorre un dato elenco di elementi — di solito un elenco di parole o qualsiasi altro segmento di testo separato da spazi — eseguendo lo stesso insieme di comandi su ciascuno di quegli elementi. Prima di ogni iterazione, l'istruzione `for` assegna l'elemento corrente a una variabile, che può quindi essere utilizzata dai comandi inclusi. Il processo viene ripetuto fino a quando non ci sono più elementi rimasti. La sintassi del costrutto `for` è:

```
for VARNAME in LIST
do
    COMMANDS
done
```

`VARNAME` è un nome di variabile di shell arbitrario e `LIST` è una qualsiasi sequenza di termini separati. I caratteri di delimitazione validi che dividono gli elementi nell'elenco sono definiti dalla variabile d'ambiente `IFS`, e sono i caratteri *space*, *tab* e *newline* per impostazione predefinita. L'elenco dei comandi da eseguire è delimitato dalle istruzioni `do` e `done`, quindi i comandi possono occupare tutte le righe necessarie.

Nell'esempio seguente, il comando `for` prenderà ogni elemento dall'elenco fornito — una sequenza di numeri — e lo assegnerà alla variabile `NUM`, un elemento alla volta:

```
#!/bin/bash

for NUM in 1 1 2 3 5 8 13
do
    echo -n "$NUM is "
    if [ $(( $NUM % 2 )) -ne 0 ]
    then
        echo "odd."
    else
        echo "even."
  fi
done
```

Nell'esempio, un costrutto `if` nidificato viene utilizzato insieme a un'espressione aritmetica per valutare se il numero nella variabile `NUM` corrente è pari o dispari. Supponendo che lo script di

esempio precedente sia denominato `script.sh` e si trovi nella directory corrente, verrà generato il seguente output:

```
$ ./script.sh
1 is odd.
1 is odd.
2 is even.
3 is odd.
5 is odd.
8 is even.
13 is odd.
```

Bash supporta anche un formato alternativo ai costrutti `for`, con la notazione delle doppie parentesi. Questa notazione ricorda la sintassi dell'istruzione `for` del linguaggio di programmazione C ed è particolarmente utile per lavorare con gli *array*:

```
#!/bin/bash

SEQ=( 1 1 2 3 5 8 13 )

for (( IDX = 0; IDX < ${#SEQ[*]}; IDX++ ))
do
    echo -n "${SEQ[$IDX]} is "
    if [ $(( ${SEQ[$IDX]} % 2 )) -ne 0 ]
    then
        echo "odd."
    else
        echo "even."
  fi
done
```

Questo script di esempio genererà lo stesso identico output dell'esempio precedente. Tuttavia, invece di usare la variabile `NUM` per memorizzare un elemento alla volta, la variabile `IDX` viene impiegata per tracciare l'indice dell'array corrente in ordine crescente, partendo da 0 e aggiungendo continuamente ad esso il numero di elementi nell'array `SEQ`. L'elemento effettivo viene recuperato dalla sua posizione nell'array con `${SEQ[$IDX]}`.

Allo stesso modo, il costrutto `until` esegue una sequenza di comandi fino a quando un comando di test — come il comando `test` stesso — termina con lo stato 0 (successo). Per esempio, la stessa struttura di loop dell'esempio precedente può essere implementata con `until` come segue:

```
#!/bin/bash

SEQ=( 1 1 2 3 5 8 13 )

IDX=0

until [ $IDX -eq ${#SEQ[*]} ]
do
    echo -n "${SEQ[$IDX]} is "
    if [ $(( ${SEQ[$IDX]} % 2 )) -ne 0 ]
    then
        echo "odd."
    else
        echo "even."
  fi
  IDX=$(( $IDX + 1 ))
done
```

Il costrutto `until` può richiedere più istruzioni rispetto ai costrutti `for`, ma può essere più adatto a criteri di arresto non numerici forniti dalle espressioni `test` o da qualsiasi altro comando. È importante includere azioni che assicurino un criterio di arresto valido, come l'incremento di una variabile contatore, altrimenti il ciclo potrebbe essere eseguito indefinitamente.

L'istruzione `while` è simile all'istruzione `until`, ma `while` continua a ripetere l'insieme di comandi se il comando di test termina con lo stato 0 (successo). Pertanto, l'istruzione `until [ $IDX -eq ${#SEQ[*]} ]` dell'esempio precedente è equivalente a `while [ $IDX -lt ${#SEQ[*]} ]`, come ciclo che dovrebbe ripetersi mentre l'indice dell'array è *minore del* il totale degli elementi nell'array.

Un Esempio più Elaborato

Immagina che un utente desideri sincronizzare periodicamente una raccolta dei propri file e directory con un altro dispositivo di archiviazione, montato in un punto di montaggio arbitrario nel filesystem, e che un sistema di backup completo sia considerato eccessivo. Poiché si tratta di un'attività che deve essere eseguita periodicamente, uno script di shell è una buon candidato per una simile automazione.

Il compito è semplice: sincronizzare ogni file e directory contenuto in un elenco, da una directory di origine come primo argomento dello script a una directory di destinazione come secondo argomento dello script. Per semplificare l'aggiunta o la rimozione di elementi dall'elenco, verrà conservato in un file separato, `~/sync.list`, un elemento per riga:

```
$ cat ~/.sync.list
Documents
To do
Work
Family Album
.config
.ssh
.bash_profile
.vimrc
```

Il file contiene una combinazione di file e directory, alcuni con spazi vuoti nei nomi. Questo è uno scenario adatto per il comando Bash *builtin* `mapfile`, che analizzerà qualsiasi contenuto di testo e creerà una variabile di array da esso, posizionando ogni riga come un singolo elemento di array. Il file di script sarà chiamato `sync.sh`, contenente il seguente codice:

```
#!/bin/bash

set -ef

# List of items to sync
FILE=~/.sync.list

# Origin directory
FROM=$1

# Destination directory
TO=$2

# Check if both directories are valid
if [ ! -d "$FROM" -o ! -d "$TO" ]
then
  echo Usage:
  echo "$0 <SOURCEDIR> <DESTDIR>"
  exit 1
fi

# Create array from file
mapfile -t LIST < $FILE

# Sync items
for (( IDX = 0; IDX < ${#LIST[*]}; IDX++ ))
do
```

```
    echo -e "$FROM/${LIST[$IDX]} \u2192 $TO/${LIST[$IDX]}";
    rsync -qa --delete "$FROM/${LIST[$IDX]}" "$TO";
done
```

La prima azione che lo script esegue è ridefinire due parametri della shell con il comando `set`: l'opzione `-e` fa terminare immediatamente l'esecuzione se un comando esce con uno stato diverso da zero e l'opzione `-f` disabilita il *globbing* del nome del file. Entrambe le opzioni possono essere abbreviate come `-ef`. Questo non è un passaggio obbligatorio, ma aiuta a diminuire la probabilità di comportamenti imprevisti.

Le istruzioni effettive orientate all'applicazione del file di script possono essere suddivise in tre parti:

1. *Raccoglie e controlla i parametri dello script*

   La variabile `FILE` è il percorso del file contenente l'elenco degli elementi da copiare: `~/.sync.list`. Le variabili `FROM` e `TO` sono rispettivamente i percorsi di origine e di destinazione. Poiché questi ultimi due parametri sono forniti dall'utente, passano attraverso un semplice test di convalida eseguito dal costrutto `if`: se uno dei due non è una directory valida, valutato dal test `[ ! -d "$FROM" -o ! -d "$TO" ]`, lo script mostrerà un breve messaggio di aiuto e poi terminerà con uno stato di uscita di 1.

2. *Carica l'elenco di file e directory*

   Dopo che tutti i parametri sono stati definiti, viene creato un array contenente l'elenco degli elementi da copiare con il comando `mapfile -t LIST < $FILE`. L'opzione `-t` di `mapfile` rimuoverà il carattere di nuova riga finale da ogni riga prima di includerlo nella variabile array denominata `LIST`. Il contenuto del file indicato dalla variabile `FILE` — `~/.sync.list` — viene letto tramite il reindirizzamento dell'input.

3. *Esegue la copia e informa l'utente*

   Un ciclo `for` che utilizza la notazione con doppie parentesi attraversa l'array di elementi, con la variabile `IDX` che tiene traccia dell'incremento dell'indice. Il comando `echo` informerà l'utente di ogni elemento che viene copiato. Il carattere Unicode di escape — `\u2192` — per il carattere *freccia destra* è presente nel messaggio di output, quindi deve essere utilizzata l'opzione `-e` del comando `echo`. Il comando `rsync` copierà selettivamente solo i file modificati dall'origine, quindi il suo utilizzo è raccomandato per tali attività. Le opzioni `rsync` `-q` e `-a`, condensate in `-qa`, inibiranno i messaggi `rsync` e attiveranno la modalità *archive*, dove vengono preservate tutte le proprietà dei file. L'opzione `--delete` farà in modo che `rsync` cancelli un elemento nella destinazione che non esiste più nell'origine, quindi dovrebbe essere usato con attenzione.

Supponendo che tutti gli elementi nell'elenco esistano nella directory home dell'utente `carol`, `/home/carol`, e la directory di destinazione `/media/carol/backup` punta a un dispositivo di archiviazione esterno montato, il comando `sync. sh /home/carol /media/carol/backup` genererà il seguente output:

```
$ sync.sh /home/carol /media/carol/backup
/home/carol/Documents → /media/carol/backup/Documents
/home/carol/"To do" → /media/carol/backup/"To do"
/home/carol/Work → /media/carol/backup/Work
/home/carol/"Family Album" → /media/carol/backup/"Family Album"
/home/carol/.config → /media/carol/backup/.config
/home/carol/.ssh → /media/carol/backup/.ssh
/home/carol/.bash_profile → /media/carol/backup/.bash_profile
/home/carol/.vimrc → /media/carol/backup/.vimrc
```

L'esempio presuppone anche che lo script venga eseguito da root o dall'utente `carol`, poiché la maggior parte dei file sarebbe illeggibile da altri utenti. Se `script.sh` non è all'interno di una directory elencata nella variabile d'ambiente `PATH`, allora dovrebbe essere specificato con il suo percorso completo.

Esercizi Guidati

1. Come potrebbe essere usato il comando `test` per verificare se il percorso del file memorizzato nella variabile `FROM` è più recente di un file il cui percorso è memorizzato nella variabile `TO`?

2. Il seguente script dovrebbe stampare una sequenza numerica da 0 a 9, ma invece stampa indefinitamente 0. Cosa si dovrebbe fare per ottenere l'output atteso?

```
#!/bin/bash

COUNTER=0

while [ $COUNTER -lt 10 ]
do
  echo $COUNTER
done
```

3. Supponiamo che un utente abbia scritto uno script che genera un elenco ordinato di nomi utente. L'elenco ordinato risultante viene presentato come il seguente sul suo computer:

```
carol
Dave
emma
Frank
Grace
henry
```

   Tuttavia, lo stesso elenco è ordinato come segue sul computer del suo collega:

```
Dave
Frank
Grace
carol
emma
henry
```

   Cosa potrebbe spiegare le differenze tra i due elenchi ordinati?

Esercizi Esplorativi

1. Come possono essere usati tutti gli argomenti della riga di comando dello script per inizializzare un array Bash?

2. Perché, controintuitivamente, il comando `test 1 > 2` viene valutato come vero?

3. In che modo un utente potrebbe cambiare temporaneamente il separatore di campo predefinito solo con il carattere di nuova riga, pur essendo ancora in grado di ripristinarlo al suo contenuto originale?

Sommario

Questa lezione dà uno sguardo più approfondito ai test disponibili del comando `test` e ad altri costrutti condizionali e di ciclo, necessari per scrivere script di shell più elaborati. Un semplice script di sincronizzazione dei file viene fornito come esempio di una pratica applicazione di script di shell. La lezione segue i seguenti passaggi:

- Test estesi per i costrutti condizionali `if` e `case`.
- Costrutti di loop della shell: `for`, `until` e `while`.
- Iterazione attraverso array e parametri.

I comandi e le procedure affrontati sono stati:

`test`
: Esegue un confronto tra gli elementi forniti al comando.

`if`
: Un costrutto logico utilizzato negli script per valutare qualcosa come vero o falso, quindi l'esecuzione del comando successivo in base ai risultati.

`case`
: Valuta diversi valori rispetto a una singola variabile. L'esecuzione del comando script viene quindi eseguita a seconda del risultato del comando `case`.

`for`
: Ripete l'esecuzione di un comando in base a un dato criterio.

`until`
: Ripete l'esecuzione di un comando finché un'espressione non restituisce false.

`while`
: Ripete l'esecuzione di un comando mentre una data espressione restituisce true.

Risposte agli Esercizi Guidati

1. Come potrebbe essere usato il comando `test` per verificare se il percorso del file memorizzato nella variabile `FROM` è più recente di un file il cui percorso è memorizzato nella variabile `TO`?

   Il comando `test "$FROM" -nt "$TO"` restituirà un codice di stato 0 se il file nella variabile `FROM` è più recente del file nella variabile `TO`.

2. Il seguente script dovrebbe stampare una sequenza numerica da 0 a 9, ma invece stampa indefinitamente 0. Cosa si dovrebbe fare per ottenere l'output atteso?

   ```
   #!/bin/bash

   COUNTER=0

   while [ $COUNTER -lt 10 ]
   do
     echo $COUNTER
   done
   ```

   La variabile `COUNTER` dovrebbe essere incrementata, cosa che potrebbe essere eseguita con l'espressione aritmetica `COUNTER=$(( $COUNTER + 1 ))`, per raggiungere i criteri di stop e terminare il ciclo.

3. Supponiamo che un utente abbia scritto uno script che genera un elenco ordinato di nomi utente. L'elenco ordinato risultante viene presentato come il seguente sul suo computer:

   ```
   carol
   Dave
   emma
   Frank
   Grace
   henry
   ```

   Tuttavia, lo stesso elenco è ordinato come segue sul computer del suo collega:

   ```
   Dave
   Frank
   Grace
   carol
   emma
   ```

```
henry
```

Cosa potrebbe spiegare le differenze tra i due elenchi ordinati?

L'ordinamento è basato sulle impostazioni internazionali del sistema corrente. Per evitare incongruenze, le attività di ordinamento dovrebbero essere eseguite con la variabile d'ambiente `LANG` impostata su `C`.

```
 [[sec.105.2_02-AEE]]
<<<
== Risposte agli Esercizi Esplorativi
```

4. Come possono essere usati tutti gli argomenti della riga di comando dello script per inizializzare un array Bash?

   I comandi `PARAMS=( $* )` o `PARAMS=( "$@" )` creeranno un array chiamato `PARAMS` con tutti gli argomenti.

5. Perché, controintuitivamente, il comando `test 1 > 2` viene valutato come vero?

   L'operatore `>` deve essere utilizzato con i test di stringa, non con i test numerici.

6. In che modo un utente potrebbe cambiare temporaneamente il separatore di campo predefinito solo con il carattere di nuova riga, pur essendo ancora in grado di ripristinarlo al suo contenuto originale?

   Una copia della variabile `IFS` può essere memorizzata in un'altra variabile: `OLDIFS=$IFS`. Quindi il nuovo separatore di riga è definito con `IFS=$'\n'` e la variabile IFS può essere ripristinata con `IFS=$OLDIFS`.



Linux Professional Institute

Argomento 106: Interfacce Utente e Desktop



106.1 Installare e configurare X11

Obiettivi LPI di riferimento

LPIC-1 version 5.0, Exam 102, Objective 106.1

Peso

2

Aree di Conoscenza Chiave

- Comprendere l'architettura di X11.
- Comprensione e conoscenza di base del file di configurazione di X Window.
- Sovrascrivere aspetti specifici della configurazione di Xorg, come il layout della tastiera.
- Comprendere i componenti degli ambienti desktop, come display manager e window manager.
- Gestire l'accesso al server X e visualizzare le applicazioni sui server X remoti.
- Conoscenza di Wayland.

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

- `/etc/X11/xorg.conf`
- `/etc/X11/xorg.conf.d/`
- `~/.xsession-errors`
- `xhost`
- `xauth`
- `DISPLAY`
- `X`



106.1 Lezione 1

Certificazione:	LPIC-1
Versione:	5.0
Argomento:	106 Interfacce Utente e Desktop
Obiettivo:	106.1 Installare e configurare X11
Lezione:	1 di 1

Introduzione

Il sistema X Window è uno *stack* software utilizzato per visualizzare testo e grafica su uno schermo. L'aspetto e il design complessivi di un *client X* non sono dettati dal sistema X Window, ma sono invece gestiti da ogni singolo client X, da un *window manager* (per esempio Window Maker, Tab Window Manager) o da un *desktop environment* completo come KDE, GNOME o Xfce. Gli ambienti desktop verranno trattati in una lezione successiva. Questa lezione si concentrerà sull'architettura sottostante e sugli strumenti comuni per il sistema X Window che un amministratore utilizzerebbe per configurare X.

Il sistema X Window è multipiattaforma e funziona su vari sistemi operativi come Linux, BSD, Solaris e altri sistemi *Unix-like*. Sono disponibili anche implementazioni per macOS di Apple e Microsoft Windows.

La versione principale del protocollo X utilizzato nelle moderne distribuzioni Linux è *X.org* versione 11, comunemente scritta come *X11*. Il protocollo X è il meccanismo di comunicazione tra il client X e il server X. Le differenze tra il client X e il *server X* verranno discusse di seguito.

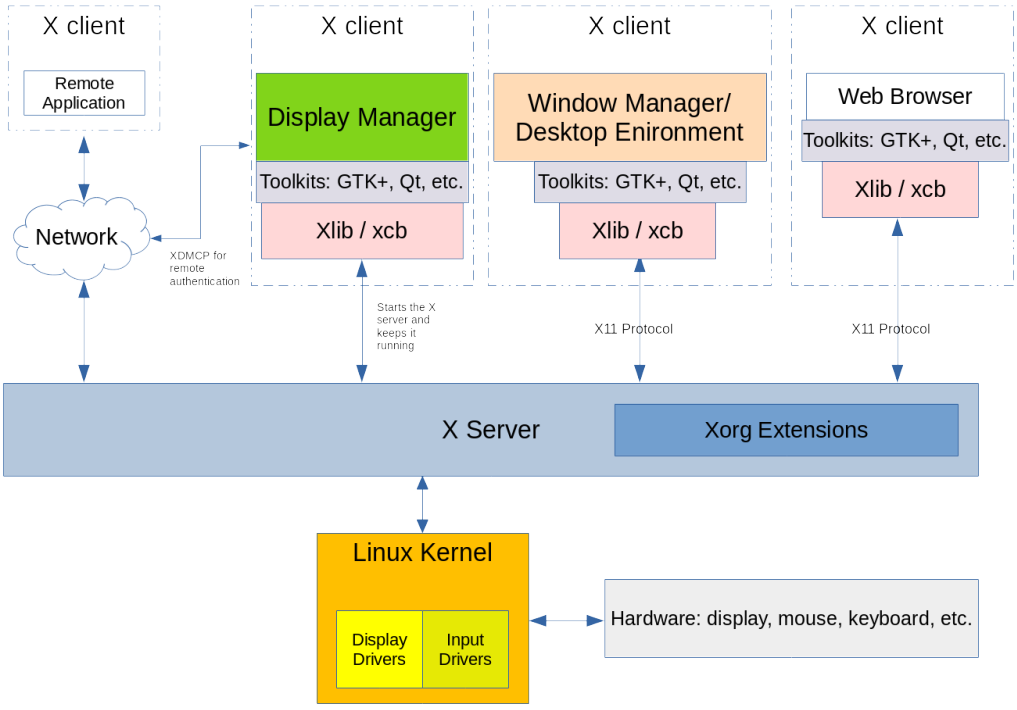
NOTE Il predecessore di X Window System era un sistema a finestre chiamato *W* e fu uno

sforzo di sviluppo congiunto tra IBM, DEC e MIT. Questo software nacque da *Project Athena* nel 1984. Quando gli sviluppatori hanno iniziato a lavorare su un nuovo server di visualizzazione, hanno scelto la lettera successiva dell'alfabeto inglese: "X". L'evoluzione del sistema X Window è attualmente controllata dal *MIT X Consortium*.

Architettura di Sistema X Window

Il sistema X Window fornisce i meccanismi per disegnare forme bidimensionali di base (e forme tridimensionali tramite estensioni) su un display. È diviso in un client e un server e nella maggior parte delle installazioni in cui è necessario un desktop grafico, entrambi questi componenti si trovano sullo stesso computer. Il componente client assume la forma di un'applicazione, quali un emulatore di terminale, un gioco o un browser web. Ogni applicazione client informa il server X sulla posizione e le dimensioni della finestra sullo schermo di un computer. Il client gestisce anche ciò che entra in quella finestra e il server X mette sullo schermo il disegno richiesto. Il sistema X Window gestisce anche l'input da dispositivi come mouse, tastiere, trackpad e altro.

Basic Structure of an X Window System



Il sistema X Window è compatibile con la rete e più client X da diversi computer su una rete possono effettuare richieste di disegno a un singolo server X remoto. Il ragionamento alla base di

ciò è che un amministratore o un utente può avere accesso a un'applicazione grafica su un sistema remoto che potrebbe non essere disponibile sul proprio sistema locale.

Una caratteristica fondamentale del sistema X Window è che è *modulare*. Nel corso dell'esistenza di X Window System sono state sviluppate e aggiunte al suo framework nuove funzionalità. Questi nuovi componenti sono stati aggiunti solo come estensioni al server X, lasciando intatto il protocollo X11 di base. Queste estensioni sono contenute nei file di libreria *Xorg*. Esempi di librerie Xorg includono: `libXrandr`, `libXcursor`, `libX11`, `libxkbfile` e molte altre, ognuna delle quali fornisce funzionalità estese al server X.

Un *display manager* fornisce un login grafico a un sistema. Questo sistema può essere un computer locale o un computer in rete. Il display manager viene avviato dopo l'avvio del computer e avvia una sessione del server X per l'utente autenticato. Il display manager è anche responsabile di mantenere il server X attivo e funzionante. I display manager di esempio includono GDM, SDDM e LightDM.

Ogni istanza di un server X in esecuzione ha un *display name* per identificarla. Il nome visualizzato contiene quanto segue:

```
hostname:displaynumber.screennumber
```

Il *display name* indica anche a un'applicazione grafica dove l'output debba essere visualizzato e su quale host (se si utilizza una connessione X remota).

L'`hostname` si riferisce al nome del sistema che visualizzerà l'applicazione. Se un nome host manca dal nome visualizzato, viene assunto l'host locale.

Il `displaynumber` fa riferimento alla raccolta di "schermi" che sono in uso, sia che si tratti di un singolo schermo di laptop o di più schermi su una workstation. A ogni sessione del server X in esecuzione viene assegnato un numero di visualizzazione a partire da `0`.

Lo `screennumber` predefinito è `0`. Questo può essere il caso se solo uno schermo fisico o più schermi fisici sono configurati per funzionare come uno schermo. Quando tutte le schermate in una configurazione multi-monitor sono combinate in un'unica schermata logica, le finestre dell'applicazione possono essere spostate liberamente tra le schermate. In situazioni in cui ogni schermata è configurata per funzionare indipendentemente l'una dall'altra, ogni schermata ospiterà le finestre dell'applicazione che si aprono al loro interno e le finestre non possono essere spostate da una schermata all'altra. Ogni schermo indipendente avrà il proprio numero assegnato. Se è in uso un solo schermo logico, il punto e il numero dello schermo vengono omessi.

Il nome visualizzato di una sessione X in esecuzione è memorizzato nella variabile d'ambiente

`DISPLAY`:

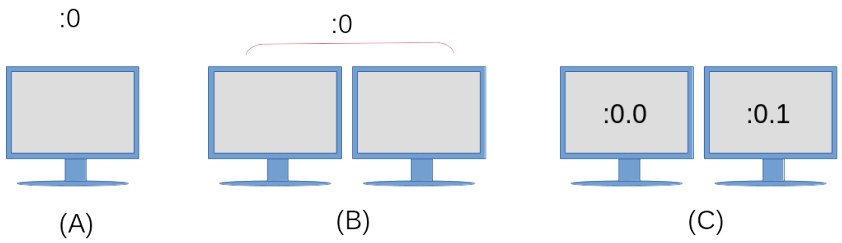
```
$ echo $DISPLAY
:0
```

L'output descrive in dettaglio quanto segue:

1. Il server X in uso è sul sistema locale, quindi non c'è nulla stampato a sinistra dei due punti.
2. L'attuale sessione del server X è la prima come indicato dallo `0` subito dopo i due punti.
3. È in uso una sola schermata logica, quindi il numero di schermata non è visibile.

Per illustrare ulteriormente questo concetto, fare riferimento al diagramma seguente:

Example Display Configurations



(A)

Un unico monitor, con un'unica configurazione del display e un solo schermo.

(B)

Configurato come un unico display, con due monitor fisici configurati come uno schermo. Le finestre dell'applicazione possono essere spostate liberamente tra i due monitor.

(C)

Una singola configurazione del display (come indicato da `:0`) tuttavia ogni monitor è uno schermo indipendente. Entrambi gli schermi condivideranno ancora gli stessi dispositivi di input come tastiera e mouse, tuttavia un'applicazione aperta sullo schermo `:0.0` non può essere spostata sullo schermo `:0.1` e viceversa.

Per avviare un'applicazione su uno schermo specifico, assegna il numero dello schermo alla variabile d'ambiente `DISPLAY` prima di avviare l'applicazione:

```
$ DISPLAY=:0.1 firefox &
```

Questo comando avvierà il browser Web Firefox sullo schermo a destra nel diagramma sopra. Alcuni toolkit forniscono anche opzioni della riga di comando per istruire un'applicazione da eseguire su una schermata specificata. Vedere `--screen` e `--display` nella pagina man `gtk-options (7)` per un esempio. === Configurazione X Server

Tradizionalmente, il file di configurazione principale utilizzato per configurare un server X è il file `/etc/X11/xorg.conf`. Sulle moderne distribuzioni Linux, il server X si configurerà in fase di esecuzione quando il server X viene avviato e quindi non può esistere alcun file `xorg.conf`.

Il file `xorg.conf` è diviso in parti separate chiamate *sections*. Ogni sezione inizia con il termine `Section` e dopo questo termine c'è il *nome della sezione* che si riferisce alla configurazione di un componente. Ogni `Section` viene terminata da una `EndSection` corrispondente. Il tipico file `xorg.conf` contiene le seguenti sezioni:

`InputDevice`

Per configurare un modello specifico di tastiera o mouse.

`InputClass`

Nelle moderne distribuzioni Linux questa sezione si trova in genere in un file di configurazione separato situato in `/etc/X11/xorg.conf.d/`. La `InputClass` è usata per configurare una classe di dispositivi hardware come tastiere e mouse piuttosto che un a parte hardware specifica. Di seguito è riportato un file di esempio `/etc/X11/xorg.conf.d/00-keyboard.conf`:

```
Section "InputClass"
        Identifier "system-keyboard"
        MatchIsKeyboard "on"
        Option "XkbLayout" "us"
        Option "XkbModel" "pc105"
EndSection
```

L'opzione per `XkbLayout` determina il layout dei tasti su una tastiera, come Dvorak, mano sinistra o destra, QWERTY e lingua. L'opzione per `XkbModel` è usata per definire il tipo di tastiera in uso. Una tabella di modelli, layout e le loro descrizioni può essere trovata in `xkeyboard-config(7)`. I file associati ai layout di tastiera possono essere trovati in `/usr/share/X11/xkb`. Un esempio di layout di tastiera greca Polytonic su un computer Chromebook sarebbe simile al seguente:

```
Section "InputClass"
        Identifier "system-keyboard"
```

```
        MatchIsKeyboard "on"
        Option "XkbLayout" "gr(polytonic)"
        Option "XkbModel" "chromebook"
EndSection
```

In alternativa, il layout di una tastiera può essere modificato durante una sessione X in esecuzione con il comando `setxkbmap`. Ecco un esempio di questo comando che imposta il layout Greek Polytonic su un computer Chromebook:

```
$ setxkbmap -model chromebook -layout "gr(polytonic)"
```

Questa impostazione permane solo fino a quando la sessione X rimane in uso. Per rendere permanenti tali modifiche, modificare il file `/etc/X11/xorg.conf.d/00-keyboard.conf` per includere le impostazioni richieste.

NOTE Il comando `setxkbmap` utilizza la *X Keyboard Extension* (XKB). Questo è un esempio della funzionalità aggiuntiva del sistema X Window attraverso l'uso di estensioni.

Le moderne distribuzioni Linux forniscono il comando `localectl` tramite `systemd` che può essere utilizzato anche per modificare un layout di tastiera e creerà automaticamente il file di configurazione `/etc/X11/xorg.conf.d/00-keyboard.conf`. Ancora una volta, ecco un esempio di configurazione di una tastiera greca Polytonic su un Chromebook, questa volta utilizzando il comando `localectl`:

```
$ localectl --no-convert set-x11-keymap "gr(polytonic)" chromebook
```

L'opzione `--no-convert` è usata qui per impedire a `localectl` di modificare la mappa dei tasti della console dell'host.

`Monitor`

La sezione `Monitor` descrive il monitor fisico utilizzato e dove è collegato. Di seguito è riportato un esempio di configurazione che mostra un monitor hardware collegato alla seconda porta del display e utilizzato come monitor principale.

```
Section "Monitor"
        Identifier  "DP2"
        Option      "Primary" "true"
EndSection
```

`Device`

La sezione `Device` descrive la scheda video fisica utilizzata. La sezione conterrà anche il modulo del kernel utilizzato come driver per la scheda video, insieme alla sua posizione fisica sulla scheda madre.

```
Section "Device"
        Identifier  "Device0"
        Driver      "i915"
        BusID       "PCI:0:2:0"
EndSection
```

`Screen`

La sezione `Screen` unisce le sezioni `Monitor` e `Device`. Un esempio di sezione `Screen` potrebbe essere simile alla seguente:

```
Section "Screen"
        Identifier "Screen0"
        Device     "Device0"
        Monitor    "DP2"
EndSection
```

`ServerLayout`

La sezione `ServerLayout` raggruppa tutte le sezioni come mouse, tastiera e schermi in un'unica interfaccia del sistema X Window.

```
Section "ServerLayout"
    Identifier   "Layout-1"
    Screen       "Screen0" 0 0
    InputDevice  "mouse1"  "CorePointer"
    InputDevice  "system-keyboard"  "CoreKeyboard"
EndSection
```

NOTE Non tutte le sezioni possono essere trovate all'interno di un file di configurazione. Nei casi in cui manca una sezione, i valori predefiniti sono invece forniti dall'istanza del server X in esecuzione.

I file di configurazione specificati dall'utente risiedono anche in `/etc/X11/xorg.conf.d/`. I file di configurazione forniti dalla distribuzione si trovano in `/usr/share/X11/xorg.conf.d/`. I file di configurazione che si trovano all'interno di `/etc/X11/xorg.conf.d/` vengono analizzati

prima del file `/etc/X11/xorg.conf` se quest'ultimo esiste nel sistema.

Il comando `xdpyinfo` viene utilizzato su un computer per visualizzare le informazioni su un'istanza del server X in esecuzione. Di seguito è riportato un esempio di output del comando:

```
$ xdpyinfo
name of display:    :0
version number:    11.0
vendor string:    The X.Org Foundation
vendor release number:    12004000
X.Org version: 1.20.4
maximum request size:  16777212 bytes
motion buffer size:  256
bitmap unit, bit order, padding:    32, LSBFirst, 32
image byte order:    LSBFirst
number of supported pixmap formats:    7
supported pixmap formats:
    depth 1, bits_per_pixel 1, scanline_pad 32
    depth 4, bits_per_pixel 8, scanline_pad 32
    depth 8, bits_per_pixel 8, scanline_pad 32
    depth 15, bits_per_pixel 16, scanline_pad 32
    depth 16, bits_per_pixel 16, scanline_pad 32
    depth 24, bits_per_pixel 32, scanline_pad 32
    depth 32, bits_per_pixel 32, scanline_pad 32
keycode range:    minimum 8, maximum 255
focus:  None
number of extensions:    25
    BIG-REQUESTS
    Composite
    DAMAGE
    DOUBLE-BUFFER
    DRI3
    GLX
    Generic Event Extension
    MIT-SCREEN-SAVER
    MIT-SHM
    Present
    RANDR
    RECORD
    RENDER
    SECURITY
    SHAPE
    SYNC
    X-Resource
```

```
    XC-MISC
    XFIXES
    XFree86-VidModeExtension
    XINERAMA
    XInputExtension
    XKEYBOARD
    XTEST
    XVideo
default screen number:    0
number of screens:    1

screen #0:
  dimensions:    3840x1080 pixels (1016x286 millimeters)
  resolution:    96x96 dots per inch
  depths (7):    24, 1, 4, 8, 15, 16, 32
  root window id:    0x39e
  depth of root window:    24 planes
  number of colormaps:    minimum 1, maximum 1
  default colormap:    0x25
  default number of colormap cells:    256
  preallocated pixels:    black 0, white 16777215
  options:    backing-store WHEN MAPPED, save-unders NO
  largest cursor:    3840x1080
  current input event mask:    0xda0033
    KeyPressMask            KeyReleaseMask          EnterWindowMask
    LeaveWindowMask         StructureNotifyMask     SubstructureNotifyMask
    SubstructureRedirectMask PropertyChangeMask      ColormapChangeMask
  number of visuals:    270
...
```

Le parti più rilevanti dell'output sono in grassetto, come il nome del display (che è lo stesso del contenuto della variabile d'ambiente `DISPLAY`), le informazioni sulla versione del server X in uso, il numero e l'elenco di Estensioni Xorg in uso e ulteriori informazioni sullo schermo stesso.

Creazione di un File di Configurazione Base di Xorg

Anche se X creerà la sua configurazione dopo l'avvio del sistema sulle moderne installazioni Linux, può ancora essere usato un file `xorg.conf`. Per generare un file permanente `/etc/X11/xorg.conf`, esegui il seguente comando:

```
$ sudo Xorg -configure
```

NOTE

Se è già in esecuzione una sessione X, dovrai specificare un diverso `DISPLAY` nel tuo comando, per esempio:

```
$ sudo Xorg :1 -configure
```

Su alcune distribuzioni Linux, il comando `X` può essere usato al posto di `Xorg`, poiché `X` è un collegamento simbolico a `Xorg`.

Verrà creato un file `xorg.conf.new` nella directory di lavoro corrente. I contenuti di questo file derivano da ciò che il server X ha trovato disponibile nell'hardware e nei driver del sistema locale. Per usare questo file, dovrà essere spostato nella directory `/etc/X11/` e rinominato in `xorg.conf`:

```
$ sudo mv xorg.conf.new /etc/X11/xorg.conf
```

NOTE

Le seguenti pagine di manuale forniscono ulteriori informazioni sui componenti del sistema X Window: `xorg.conf(5)`, `Xserver(1)`, `X(1)` e `Xorg(1)`.

Wayland

Wayland è il nuovo protocollo di visualizzazione progettato per sostituire il sistema X Window. Molte moderne distribuzioni Linux lo utilizzano come server di visualizzazione predefinito. È pensato per essere più leggero sulle risorse di sistema e avere un ingombro di installazione inferiore rispetto a X. Il progetto è iniziato nel 2010 ed è ancora in fase di sviluppo attivo, compreso il lavoro di sviluppatori attivi ed ex. di X.org

A differenza del sistema X Window, non esiste alcuna istanza del server che gira tra il client e il kernel. Invece, una finestra client funziona con il proprio codice o con quello di un toolkit (come Gtk+ o Qt) per fornire il *rendering*. Per eseguire il rendering, viene effettuata una richiesta al kernel Linux tramite il protocollo Wayland. Il kernel inoltra la richiesta tramite il protocollo Wayland al Wayland *compositor*, che gestisce l'input del dispositivo, la gestione delle finestre e la composizione. Il compositore è la parte del sistema che combina gli elementi renderizzati nell'output visivo sullo schermo.

La maggior parte dei toolkit moderni come Gtk+ 3 e Qt 5 sono stati aggiornati per consentire il rendering su un sistema X Window o su un computer con Wayland. Non tutte le applicazioni autonome sono state ancora scritte per supportare il rendering in Wayland. Per l'esecuzione di applicazioni e framework ancora destinati al sistema X Window, l'applicazione può essere eseguita all'interno di *XWayland*. Il sistema XWayland è un server X separato che viene eseguito all'interno di un client Wayland e quindi esegue il rendering dei contenuti di una finestra client

all'interno di un'istanza di server X autonoma.

Proprio come il sistema X Window usa una variabile d'ambiente `DISPLAY` per tenere traccia degli schermi in uso, il protocollo Wayland usa una variabile d'ambiente `WAYLAND_DISPLAY`. Di seguito è riportato un esempio di output da un sistema che esegue un display Wayland:

```
$ echo $WAYLAND_DISPLAY
wayland-0
```

Questa variabile di ambiente non è disponibile sui sistemi che eseguono X.

Esercizi Guidati

1. Quale comando usereste per determinare quali estensioni Xorg sono disponibili su un sistema?

2. Hai appena ricevuto un nuovissimo mouse a 10 pulsanti per il tuo computer, tuttavia è richiesta una configurazione aggiuntiva per far funzionare correttamente tutti i pulsanti. Senza modificare il resto della configurazione del server X, quale directory useresti per creare un nuovo file di configurazione per questo mouse e quale specifica sezione di configurazione verrebbe usata in questo file?

3. Quale componente di un'installazione Linux è responsabile del mantenimento in esecuzione di un server X?

4. Quale opzione della riga di comando viene utilizzata con il comando `X` per creare un nuovo file di configurazione `xorg.conf`?

Esercizi Esplorativi

1. Quale sarebbe il contenuto della variabile d'ambiente `DISPLAY` su un sistema chiamato `lab01` usando una configurazione di visualizzazione singola? Supponiamo che la variabile d'ambiente `DISPLAY` venga visualizzata in un emulatore di terminale sulla terza schermata indipendente.

2. Quale comando può essere utilizzato per creare un file di configurazione della tastiera per l'utilizzo da parte del sistema X Window?

3. In una tipica installazione Linux un utente può passare da un terminale virtuale all'altro premendo i tasti `Ctrl` + `Alt` + `F1` - `F6` su una tastiera. Ti è stato chiesto di configurare un sistema *kiosk* con un'interfaccia grafica e devi disabilitare questa funzione per evitare manomissioni non autorizzate del sistema. Decidi di creare un file di configurazione `/etc/X11/xorg.conf.d/10-kiosk.conf`. Utilizzando una sezione `ServerFlags` (utilizzata per impostare le opzioni Xorg globali sul server), quale opzione dovrebbe essere specificata? Riguarda la pagina `man xorg (1)` per individuare l'opzione.

Sommario

Questa lezione ha riguardato il sistema X Window utilizzato su Linux. Il sistema X Window viene utilizzato per disegnare immagini e testo sugli schermi, così come sono definiti in vari file di configurazione. Il sistema X Window viene spesso utilizzato per configurare dispositivi di input come mouse e tastiere. Questa lezione ha discusso i seguenti punti:

- L'architettura del sistema X Window ad alto livello.
- Quali file di configurazione sono usati per configurare un X Window System e la loro posizione nel filesystem.
- Come usare la variabile d'ambiente `DISPLAY` su un sistema che esegue X.
- Una breve introduzione al protocollo di visualizzazione Wayland.

I comandi e i file di configurazione trattati sono stati:

- Modifica del layout di una tastiera all'interno di un'installazione Xorg con `setxkbmap` e `localectl`.
- Il comando `Xorg` per creare un nuovo file di configurazione `/etc/X11/xorg.conf`.
- Il contenuto dei file di configurazione di Xorg in: `/etc/X11/xorg.conf`, `/etc/X11/xorg.conf.d/` e `/usr/share/X11/xorg.conf.d/`.
- Il comando `xdpyinfo` per visualizzare informazioni generali su una sessione del server X in esecuzione.

Risposte agli Esercizi Guidati

1. Quale comando usereste per determinare quali estensioni Xorg sono disponibili su un sistema?

   ```
   $ xdpyinfo
   ```

2. Hai appena ricevuto un nuovissimo mouse a 10 pulsanti per il tuo computer, tuttavia richiederà una configurazione aggiuntiva per far funzionare correttamente tutti i pulsanti. Senza modificare il resto della configurazione del server X, quale directory useresti per creare un nuovo file di configurazione per questo mouse e quale specifica sezione di configurazione verrebbe usata in questo file?

   Le configurazioni definite dall'utente dovrebbero trovarsi in `/etc/X11/xorg.conf.d/` e la sezione specifica necessaria per questa configurazione del mouse sarebbe `InputDevice`.

3. Quale componente di un'installazione Linux è responsabile del mantenimento in esecuzione di un server X?

   Il display manager.

4. Quale opzione della riga di comando viene utilizzata con il comando `X` per creare un nuovo file di configurazione `xorg.conf`?

   ```
   -configure
   ```

   Ricorda che il comando `X` è un collegamento simbolico al comando `Xorg`.

Risposte agli Esercizi Esplorativi

1. Quale sarebbe il contenuto della variabile d'ambiente `DISPLAY` su un sistema chiamato `lab01` usando una configurazione di visualizzazione singola? Supponiamo che la variabile d'ambiente `DISPLAY` venga visualizzata in un emulatore di terminale sulla terza schermata indipendente.

```
$ echo $DISPLAY
lab01:0.2
```

2. Quale comando può essere utilizzato per creare un file di configurazione della tastiera per l'utilizzo da parte del sistema X Window?

```
$ localectl
```

3. In una tipica installazione Linux un utente può passare da un terminale virtuale all'altro premendo i tasti `Ctrl` + `Alt` + `F1` - `F6` su una tastiera. Ti è stato chiesto di configurare un sistema *kiosk* con un'interfaccia grafica e devi disabilitare questa funzione per evitare manomissioni non autorizzate del sistema. Decidi di creare un file di configurazione `/etc/X11/xorg.conf.d/10-kiosk.conf`. Utilizzando una sezione `ServerFlags` (utilizzata per impostare le opzioni Xorg globali sul server), quale opzione dovrebbe essere specificata? Riguarda la pagina `man xorg (1)` per individuare l'opzione.

```
Section "ServerFlags"
    Option  "DontVTSwitch"  "True"
EndSection
```



106.2 Desktop grafici

Obiettivi LPI di riferimento

LPIC-1 version 5.0, Exam 102, Objective 106.2

Peso

1

Aree di Conoscenza Chiave

- Conoscenza dei principali ambienti desktop.
- Conoscenza dei protocolli per accedere alle sessioni di desktop remoto.

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

- KDE
- Gnome
- Xfce
- X11
- XDMCP
- VNC
- Spice
- RDP



106.2 Lezione 1

Certificazione:	LPIC-1
Versione:	5.0
Argomento:	106 Interfacce Utente e Desktop
Obiettivo:	106.2 Desktop grafici
Lezione:	1 di 1

Introduzione

I sistemi operativi basati su Linux sono noti per la loro *interfaccia a riga di comando* avanzata: essa però può intimidire gli utenti non esperti. Con l'obiettivo di rendere l'utilizzo del computer più intuitivo, la combinazione di display ad alta risoluzione con dispositivi di puntamento ha dato vita a interfacce utente grafiche. Dato che l'interfaccia della riga di comando richiede una conoscenza preliminare dei nomi dei programmi e delle relative opzioni di configurazione, con una *graphical user interface* (GUI) la funzionalità del programma può essere attivata puntando a elementi visivi familiari, rendendo la curva di apprendimento meno ripida. Inoltre, l'interfaccia grafica è più adatta per attività multimediali e orientata ad altre attività visive.

In effetti, l'interfaccia utente grafica è quasi ormai un sinonimo di interfaccia per computer e la maggior parte delle distribuzioni Linux viene fornita con l'interfaccia grafica installata come impostazione predefinita. Non esiste, tuttavia, un singolo programma monolitico responsabile per i desktop grafici completi disponibili nei sistemi Linux. Invece, ogni desktop grafico è di fatto un'ampia raccolta di programmi e delle loro dipendenze, che variano dalle scelte della distribuzione al gusto personale dell'utente.

Il Sitema X Window

In Linux e in altri sistemi operativi *Unix-like* in cui è impiegato, *X Window System* (noto anche come *X11* o più semplicemente *X*) fornisce le risorse di basso livello relative al rendering dell'interfaccia grafica e all'interazione dell'utente con essa, come:

- La gestione degli eventi di input, quali i movimenti del mouse o le sequenze di tasti.
- La possibilità di tagliare, copiare e incollare contenuto di testo tra applicazioni separate.
- L'interfaccia di programmazione alla quale i programmi ricorrono per disegnare gli elementi grafici.

Sebbene il sistema X Window sia responsabile del controllo del display grafico (il driver video stesso fa parte di X), questo non è destinato a disegnare elementi visivi complessi da solo. Forme, colori, sfumature e qualsiasi altro effetto visivo sono generati dall'applicazione in esecuzione su X. Questo approccio offre alle applicazioni molto spazio per creare interfacce personalizzate, ma può anche portare a un sovraccarico di sviluppo che va oltre l'ambito dell'applicazione e a incongruenze nell'aspetto e nel comportamento rispetto ad altre interfacce di programmazione.

Dal punto di vista dello sviluppatore, l'introduzione dell'*ambiente desktop* facilita la programmazione GUI legata allo sviluppo dell'applicazione sottostante, mentre dal punto di vista dell'utente fornisce un'esperienza coerente tra applicazioni distinte. Gli ambienti desktop riuniscono interfacce di programmazione, librerie e programmi di supporto che cooperano per fornire concetti di design tradizionali ma ancora in evoluzione.

L'ambiente Desktop

La GUI tradizionale del computer desktop è costituita da varie finestre - -il termine *window* è usato qui per fare riferimento a qualsiasi area dello schermo autonoma — associata ai processi in esecuzione. Poiché il sistema X Window da solo offre solo funzionalità interattive di base, l'esperienza utente completa dipende dai componenti forniti dall'ambiente desktop.

Probabilmente il componente più importante di un ambiente desktop, il *window manager* controlla il posizionamento e le decorazioni delle finestre. È il window manager che aggiunge la barra del titolo alla finestra, i pulsanti di controllo — generalmente associati alle azioni di minimizzazione, ingrandimento e chiusura — e gestisce il passaggio tra le finestre aperte.

NOTE

I concetti di base che si trovano nelle interfacce grafiche dei computer desktop derivano da idee prese da veri e propri spazi di lavoro d'ufficio. Metaforicamente parlando, lo schermo del computer è il desktop, dove vengono posizionati oggetti come documenti e cartelle. Una finestra dell'applicazione con il contenuto di un

> documento imita atti fisici come compilare un modulo o disegnare un'immagine. Come i desktop *reali*, quelli dei computer hanno anche accessori software come blocchi note, orologi, calendari, ecc., La maggior parte dei quali basati sulle loro controparti *reali*.

Tutti gli ambienti desktop forniscono un gestore di finestre che corrisponde all'aspetto del suo *widget toolkit*. I widget sono elementi visivi informativi o interattivi, come pulsanti o campi di input di testo, distribuiti all'interno della finestra dell'applicazione. I componenti desktop standard, come il *launcher* dell'applicazione, la barra delle applicazioni, ecc., e lo stesso window manager si affidano a tali toolkit widget per assemblare le loro interfacce.

Le librerie software, come *GTK+* e *Qt*, forniscono widget che i programmatori possono utilizzare per creare elaborate interfacce grafiche per le loro applicazioni. Storicamente, le applicazioni sviluppate con GTK+ non sembravano applicazioni realizzate con Qt e viceversa, ma il migliore supporto dei temi degli ambienti desktop odierni rende la distinzione meno ovvia.

Nel complesso, GTK+ e Qt offrono le stesse funzionalità per quanto riguarda i widget. Semplici elementi interattivi possono essere indistinguibili, mentre i widget compositi, come la finestra di dialogo utilizzata dalle applicazioni per aprire o salvare i file, possono, tuttavia, avere un aspetto molto diverso. Tuttavia, le applicazioni create con toolkit distinti possono essere eseguite simultaneamente, indipendentemente dal toolkit widget utilizzato dagli altri componenti desktop.

Oltre ai componenti desktop di base, che potrebbero essere considerati singoli programmi isolati, gli ambienti desktop perseguono la metafora del desktop fornendo un insieme minimo di applicazioni accessorie sviluppate secondo le stesse linee guida di progettazione. Variazioni delle seguenti applicazioni sono comunemente fornite da tutti i principali ambienti desktop:

Applicazioni relative al sistema

Emulatore di terminale, gestore di file, gestore di installazione di pacchetti, strumenti di configurazione del sistema.

Comunicazione e Internet

Gestore contatti, client di posta elettronica, browser web.

Applicazioni d'ufficio

Calendario, calcolatrice, editor di testo.

Gli ambienti desktop possono includere molti altri servizi e applicazioni: la schermata di benvenuto all'accesso, il gestore della sessione, la comunicazione tra processi, l'agente di gestione portachiavi, ecc. Incorporano anche funzionalità fornite da servizi di sistema di terze parti, come *PulseAudio* per l'audio e *CUPS* per la stampa. Queste funzionalità non richiedono l'ambiente

grafico per funzionare, ma l'ambiente desktop fornisce *frontend* grafici per facilitare la configurazione e il funzionamento di tali risorse.

Ambienti Desktop Popolari

Molti sistemi operativi proprietari supportano solo un singolo ambiente desktop ufficiale, che è legato alla loro particolare versione e non dovrebbe essere modificato. A differenza di loro, i sistemi operativi basati su Linux supportano diverse opzioni di ambiente desktop che possono essere utilizzate insieme a X. Ogni ambiente desktop ha le proprie caratteristiche, ma di solito condividono alcuni concetti di progettazione comuni:

- Un esecutore di applicazioni (*launcher*) che elenca le applicazioni integrate e di terze parti disponibili nel sistema.
- Regole che definiscono le applicazioni predefinite associate ai tipi di file e ai protocolli.
- Strumenti di configurazione per personalizzare l'aspetto e il comportamento dell'ambiente desktop.

Gnome è uno degli ambienti desktop più popolari, essendo la prima scelta in distribuzioni come Fedora, Debian, Ubuntu, SUSE Linux Enterprise, Red Hat Enterprise Linux, CentOS, ecc. Nella sua versione 3, Gnome ha apportato importanti cambiamenti nel suo aspetto e struttura, allontanandosi dalla metafora del desktop e introducendo *Gnome Shell* come nuova interfaccia.



Figure 1. Gnome Shell Activities

Il *launcher* generico a schermo intero *Gnome Shell Activities* ha sostituito il tradizionale esecutore di applicazioni e la barra delle applicazioni. Tuttavia, è ancora possibile utilizzare Gnome 3 con il

vecchio aspetto scegliendo l'opzione *Gnome Classic* nella schermata di accesso.

KDE è un ampio ecosistema di applicazioni e una piattaforma di sviluppo. La sua ultima versione dell'ambiente desktop, *KDE Plasma*, è usata di default in openSUSE, Mageia, Kubuntu, ecc. L'impiego della libreria Qt è la caratteristica principale di KDE, che conferisce il suo aspetto inconfondibile e una varietà di applicazioni originali. KDE fornisce anche uno strumento di configurazione per garantire la coesione visiva con le applicazioni GTK+.

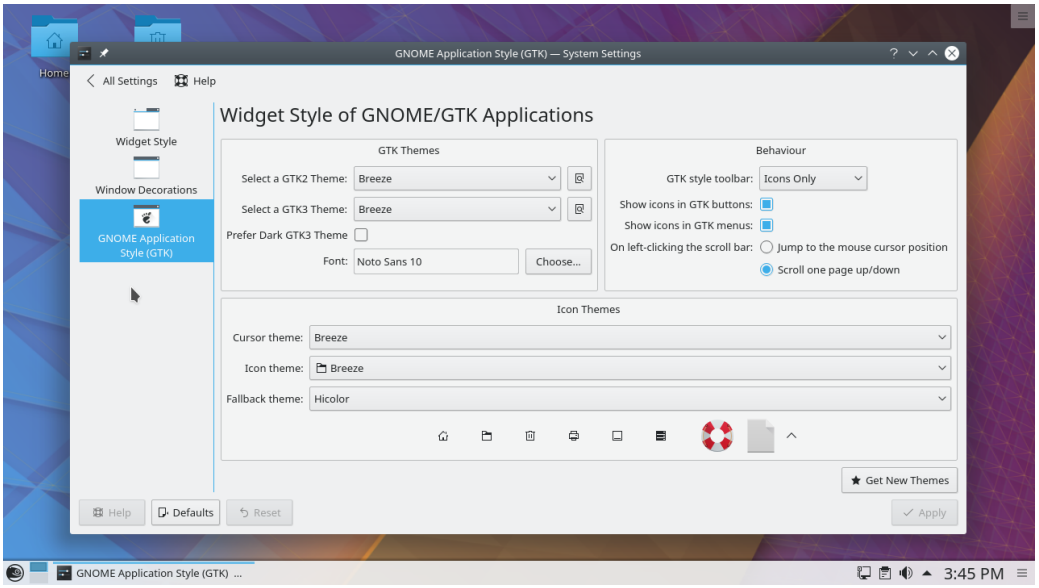


Figure 2. KDE GTK Settings

Xfce è un ambiente desktop che mira a essere esteticamente gradevole senza consumare molte risorse della macchina. La sua struttura è altamente modulare, consentendo all'utente di attivare e disattivare i componenti in base esigenze e preferenze.

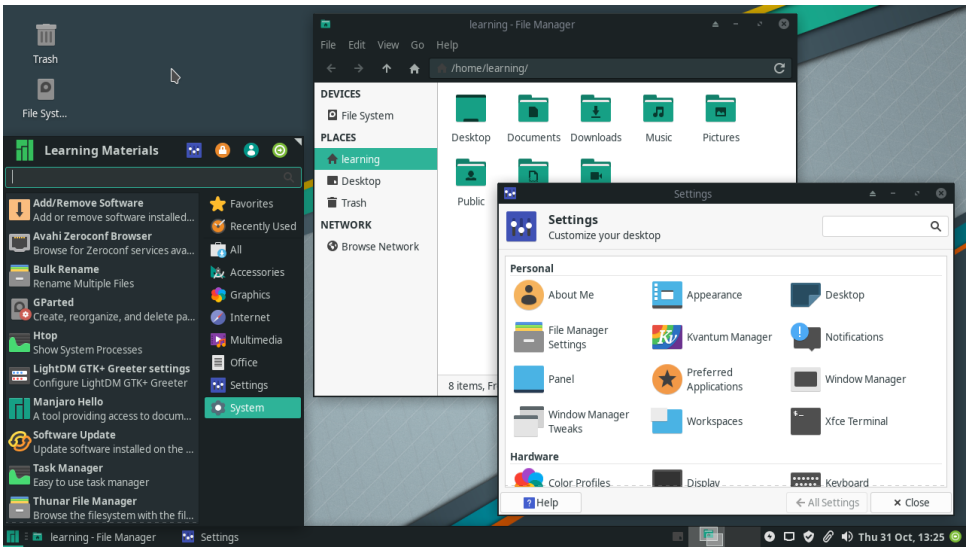


Figure 3. The Xfce desktop

Esistono molti altri ambienti desktop per Linux, solitamente forniti da distribuzioni alternative derivate. La distribuzione Linux Mint, per esempio, fornisce due ambienti desktop originali: *Cinnamon* (un fork di Gnome 3) e *MATE* (un fork di Gnome 2). *LXDE* è un ambiente desktop su misura per un basso consumo di risorse, che lo rende una buona scelta per l'installazione su apparecchiature meno recenti o computer a scheda singola. Pur non offrendo tutte le funzionalità degli ambienti desktop più blasonati, LXDE offre tutte le funzionalità di base che ci si aspetta da una moderna interfaccia utente grafica.

TIP

Le scorciatoie da tastiera svolgono un ruolo importante nell'interazione con l'ambiente desktop. Alcune scorciatoie da tastiera — come `Alt` + `Tab` per passare da una finestra all'altra o `Ctrl` + `c` per copiare il testo — sono universali in tutti gli ambienti desktop, ma ogni ambiente desktop ha anche le proprie scorciatoie da tastiera. Le scorciatoie da tastiera si trovano nello strumento di configurazione della tastiera fornito dall'ambiente desktop, dove le scorciatoie possono essere aggiunte, rimosse o modificate.

Interoperabilità Desktop

La diversità degli ambienti desktop nei sistemi operativi basati su Linux impone una sfida: come farli funzionare correttamente con applicazioni grafiche o servizi di sistema di terze parti senza dover implementare un supporto specifico per ciascuno di essi. Metodi e specifiche condivisi tra ambienti desktop migliorano notevolmente l'esperienza dell'utente e risolvono molti problemi di sviluppo, poiché le applicazioni grafiche devono interagire con l'ambiente desktop corrente indipendentemente dall'ambiente desktop per cui sono state originariamente progettate. Inoltre, è importante mantenere le impostazioni generali del desktop se l'utente alla fine cambia la scelta

dell'ambiente desktop.

L'organizzazione *freedesktop.org* gestisce un ampio corpus di specifiche per l'interoperabilità desktop. L'adozione della specifica completa non è obbligatoria, ma molte di esse sono ampiamente utilizzate:

Posizioni delle directory

Dove si trovano le impostazioni personali e altri file specifici dell'utente.

Voci sul desktop

Le applicazioni della riga di comando possono essere eseguite nell'ambiente desktop tramite qualsiasi emulatore di terminale, ma sarebbe troppo confuso renderle tutte disponibili all'avvio dell'applicazione. Le voci del desktop sono file di testo che terminano con `.desktop` che vengono utilizzati dall'ambiente desktop per raccogliere informazioni sulle applicazioni desktop disponibili e su come usarle.

Avvio automatico dell'applicazione

Voci del desktop che indicano l'applicazione che dovrebbe avviarsi automaticamente dopo che l'utente ha effettuato l'accesso.

Drag and drop

In che modo le applicazioni dovrebbero gestire gli eventi di trascinamento della selezione.

Cestino

La posizione comune dei file eliminati dal file manager, nonché i metodi per archiviare e rimuovere i file da lì.

Temi delle icone

Il formato comune per le librerie di icone intercambiabili.

La facilità d'uso fornita dagli ambienti desktop ha uno svantaggio rispetto alle interfacce di testo come la shell: la possibilità di fornire accesso remoto. Sebbene sia possibile accedere facilmente all'ambiente shell della riga di comando di una macchina remota con strumenti come `ssh`, l'accesso remoto agli ambienti grafici richiede metodi diversi e potrebbe non ottenere prestazioni soddisfacenti su connessioni più lente.

Accesso Non Locale

I sistemi X Window adottano un design basato su *display* autonomi, in cui lo stesso *X display manager* può controllare più di una sessione desktop grafica allo stesso tempo. In sostanza, un display è analogo a un terminale di testo: entrambi si riferiscono a una macchina o

un'applicazione software utilizzata come punto di ingresso per stabilire una sessione del sistema operativo indipendente. Sebbene la configurazione più comune implichi una singola sessione grafica in esecuzione nella macchina locale, sono possibili anche altre configurazioni meno convenzionali:

- Passare da una sessione desktop grafica attiva all'altra nella stessa macchina.
- Più di un set di dispositivi di visualizzazione (es. Schermo, tastiera, mouse) collegati alla stessa macchina, ognuno dei quali controlla la propria sessione di desktop grafico.
- Sessioni desktop grafiche remote, in cui l'interfaccia grafica viene inviata attraverso la rete a un display remoto.

Le sessioni di desktop remoto sono supportate in modo nativo da X, che utilizza *X Display Manager Control Protocol* (XDMCP) per comunicare con i display remoti. A causa dell'elevato utilizzo della larghezza di banda, XDMCP viene utilizzato raramente tramite Internet o in LAN a bassa velocità. Anche i problemi di sicurezza sono un limite di XDMCP: il display locale comunica con un gestore di display X remoto privilegiato per eseguire procedure remote, quindi un'eventuale vulnerabilità potrebbe rendere possibile l'esecuzione di comandi privilegiati arbitrari sulla macchina remota.

Inoltre, XDMCP richiede istanze X in esecuzione su entrambe le estremità della connessione, il che può essere irrealizzabile se il sistema X Windows non è disponibile per tutte le macchine coinvolte. In pratica, vengono utilizzati altri metodi più efficienti e meno invasivi per stabilire sessioni desktop grafiche remote. *Virtual Network Computing* (VNC) è uno strumento indipendente dalla piattaforma per visualizzare e controllare ambienti desktop remoti utilizzando il protocollo *Remote Frame Buffer* (RFB). Attraverso esso gli eventi prodotti dalla tastiera e dal mouse locali vengono trasmessi al desktop remoto, che a sua volta rinvia gli eventuali aggiornamenti dello schermo da visualizzare localmente. È possibile eseguire molti server VNC sulla stessa macchina, ma ogni server VNC necessita di una porta TCP esclusiva nell'interfaccia di rete che accetti le richieste di sessione in entrata. Per convenzione, il primo server VNC dovrebbe utilizzare la porta TCP 5900, il secondo dovrebbe utilizzare la TCP 5901 e così via.

Il server VNC non necessita di privilegi speciali per essere eseguito. Un utente normale può, per esempio, accedere al proprio account remoto e avviare da lì il proprio server VNC. Quindi, nella macchina locale, qualsiasi applicazione client VNC può essere utilizzata per accedere al desktop remoto (supponendo che le porte di rete corrispondenti siano raggiungibili). Il file `~/.vnc/xstartup` è uno script di shell eseguito dal server VNC all'avvio e può essere utilizzato per definire quale ambiente desktop il server VNC renderà disponibile per il client VNC. È importante notare che VNC non fornisce metodi di crittografia e autenticazione moderni in modo nativo, quindi dovrebbe essere utilizzato insieme a un'applicazione di terze parti che fornisce tali funzionalità. I metodi che coinvolgono VPN e tunnel SSH vengono spesso utilizzati per proteggere

le connessioni VNC.

Il *Remote Desktop Protocol* (RDP) viene utilizzato principalmente per accedere in remoto al desktop di un sistema operativo *Microsoft Windows* tramite la porta di rete TCP 3389. Sebbene utilizzi il protocollo RDP proprietario di Microsoft, l'implementazione client utilizzata nei sistemi Linux sono programmi open source concessi in licenza con la *GNU General Public License* (GPL) e non ha limitazioni legali sull'uso.

Simple Protocol for Independent Computing Environments (Spice) comprende una suite di strumenti finalizzati all'accesso all'ambiente desktop di sistemi *virtualizzati*, sia nella macchina locale sia in una posizione remota. In aggiunta a ciò, il protocollo Spice offre funzionalità native per integrare i sistemi locale e remoto, come la possibilità di accedere ai dispositivi locali (per esempio, gli altoparlanti e i dispositivi USB collegati) dalla macchina remota e la condivisione di file tra i due sistemi .

Esistono comandi client specifici per connettersi a ciascuno di questi protocolli di desktop remoto, ma il client di desktop remoto *Remmina* fornisce un'interfaccia grafica integrata che facilita il processo di connessione, memorizzando facoltativamente le impostazioni di connessione per un uso successivo. Remmina ha plugin per ogni singolo protocollo e ci sono plugin per XDMCP, VNC, RDP e Spice. La scelta dello strumento giusto dipende dai sistemi operativi coinvolti, dalla qualità della connessione di rete e dalle funzionalità dell'ambiente desktop remoto che dovrebbero essere disponibili.

Esercizi Guidati

1. Che tipo di applicazione fornisce sessioni di shell con finestre nell'ambiente desktop?

2. A causa della varietà di ambienti desktop Linux, la stessa applicazione può avere più di una versione, ognuna delle quali più adatta per un particolare *widget toolkit* . Per esempio, il client bittorrent *Transmission* ha due versioni: *transmission-gtk* e *transmission-qt*. Quale dei due dovrebbe essere installato per garantire la massima integrazione con KDE?

3. Quali ambienti desktop Linux sono consigliati per computer a scheda singola a basso costo con poca potenza di elaborazione?

Esercizi Esplorativi

1. Ci sono due modi per copiare e incollare il testo nel sistema X Window: usare il tradizionale `Ctrl` + `c` e `Ctrl` + `v` (disponibile anche nel menu della finestra) o usare il pulsante centrale del mouse facendo clic per incollare il testo attualmente selezionato. Qual è il modo appropriato per copiare e incollare il testo da un emulatore di terminale?

2. La maggior parte degli ambienti desktop assegna la scorciatoia da tastiera `Alt` + `F2` alla finestra *Esegui programma*, dove i programmi possono essere eseguiti in una modalità simil riga di comando. In KDE, quale comando eseguirà l'emulatore di terminale predefinito?

3. Qual è il protocollo più adatto per accedere a un desktop Windows remoto da un ambiente desktop Linux?

Sommario

Questa lezione è una panoramica sui desktop grafici disponibili per i sistemi Linux. Il sistema X Window da solo fornisce solo semplici funzionalità di interfaccia, quindi gli ambienti desktop estendono l'esperienza dell'utente nell'interfaccia grafica a finestre. La lezione affronta i seguenti argomenti:

- Interfaccia grafica e concetti del sistema X Window.
- Ambienti desktop disponibili per Linux.
- Somiglianze e differenze tra ambienti desktop.
- Come accedere a un ambiente desktop remoto.

I concetti e i programmi affrontati sono stati:

- Il sistema X Window.
- Ambienti desktop popolari: KDE, Gnome, Xfce.
- Protocolli di accesso remoto: XDMCP, VNC, RDP, Spice.

Risposte agli Esercizi Guidati

1. Che tipo di applicazione fornisce sessioni di shell con finestre nell'ambiente desktop?

   Qualsiasi emulatore di terminale come Konsole, Gnome terminal, xterm, etc, darà accesso a una sessione di shell interattiva locale.

2. A causa della varietà di ambienti desktop Linux, la stessa applicazione può avere più di una versione, ognuna delle quali più adatta per un particolare *widget toolkit* . Per esempio, il client bittorrent *Transmission* ha due versioni: *transmission-gtk* e *transmission-qt*. Quale dei due dovrebbe essere installato per garantire la massima integrazione con KDE?

   KDE è costruito sfruttando la libreria Qt, quindi la versione Qt — *transmission-qt* — dovrebbe essere installata.

3. Quali ambienti desktop Linux sono consigliati per computer a scheda singola a basso costo con poca potenza di elaborazione?

   Ambienti desktop di base che non utilizzano troppi effetti visivi, come Xfce e LXDE.

Risposte agli Esercizi Esplorativi

1. Ci sono due modi per copiare e incollare il testo nel sistema X Window: usare il tradizionale `Ctrl` + `c` e `Ctrl` + `v` (disponibile anche nel menu della finestra) o usare il pulsante centrale del mouse facendo clic per incollare il testo attualmente selezionato. Qual è il modo appropriato per copiare e incollare il testo da un emulatore di terminale?

   Le sessioni di shell interattive assegnano la sequenza di tasti `Ctrl` + `c` per interrompere l'esecuzione del programma, quindi si consiglia il metodo del pulsante centrale.

2. La maggior parte degli ambienti desktop assegna la scorciatoia da tastiera `Alt` + `F2` alla finestra *Esegui programma*, dove i programmi possono essere eseguiti in una modalità simil riga di comando. In KDE, quale comando eseguirà l'emulatore di terminale predefinito?

   Il comando `konsole` esegue l'emulatore di terminale di KDE, ma funzionano anche termini generici come *terminal*.

3. Qual è il protocollo più adatto per accedere a un desktop Windows remoto da un ambiente desktop Linux?

   Il *Remote Desktop Protocol* (RDP), in quanto è supportato nativamente sia da Windows sia da Linux.



106.3 Accessibilità

Obiettivi LPI di riferimento

LPIC-1 version 5.0, Exam 102, Objective 106.3

Peso

1

Aree di Conoscenza Chiave

- Conoscenza di base delle impostazioni visive e dei temi.
- Conoscenza di base delle tecnologie assistive.

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

- High Contrast/Large Print Desktop Themes.
- Screen Reader.
- Braille Display.
- Screen Magnifier.
- On-Screen Keyboard.
- Sticky/Repeat keys.
- Slow/Bounce/Toggle keys.
- Mouse keys.
- Gestures.
- Voice recognition.

106.3 Lezione 1

Certificazione:	LPIC-1
Versione:	5.0
Argomento:	106 Interfacce Utente e Desktop
Obiettivo:	106.3 Accessibilità
Lezione:	1 di 1

Introduzione

L'ambiente desktop Linux ha molte impostazioni e strumenti per adattare l'interfaccia utente alle esigenze di persone con disabilità. I normali dispositivi di interfaccia umana — schermo, tastiera e mouse/touchpad — possono essere riconfigurati individualmente per superare problemi visivi o mobilità ridotta.

È possibile, per esempio, regolare la combinazione di colori del desktop per facilitare le persone daltoniche. Inoltre, le persone che soffrono di lesioni da sforzo ripetuto possono trarre vantaggio da metodi di digitazione e puntamento alternativi.

Alcune di queste funzioni di accessibilità sono fornite dall'ambiente desktop stesso, come Gnome o KDE; altre sono fornite da programmi aggiuntivi. In quest'ultimo caso, è importante scegliere lo strumento che meglio integra l'ambiente desktop, in modo che l'aiuto sia di migliore qualità.

Impostazioni di Accessibilità

Tutte le principali distribuzioni Linux forniscono grosso modo le stesse funzionalità di accessibilità, che possono essere personalizzate con un modulo di configurazione disponibile nel

gestore delle impostazioni fornito con l'ambiente desktop. Il modulo delle impostazioni di accessibilità si chiama *Accesso Universale* nel desktop di Gnome, mentre in KDE si trova in *Impostazioni di sistema, Personalizzazione, Accessibilità*. Anche altri ambienti desktop, come *Xfce*, lo chiamano *Accessibilità* nel loro gestore delle impostazioni grafiche. Tuttavia, offrono una serie ridotta di funzionalità rispetto a Gnome e KDE.

Gnome, per esempio, può essere configurato per mostrare in modo permanente il menu Accesso universale nell'angolo in alto a destra dello schermo, dove le opzioni di accessibilità possono essere attivate rapidamente. Può essere utilizzato, per esempio, per sostituire l'avviso sonoro con uno visivo, in modo che gli utenti con problemi di udito possano percepire più facilmente gli avvisi di sistema. Sebbene KDE non disponga di un menu di accesso rapido simile, è disponibile anche la funzione di avviso visivo, chiamata invece *campanello visivo*.

Assistenza a Tastiera e Mouse

Il comportamento predefinito della tastiera e del mouse può essere modificato per aggirare specifiche difficoltà di mobilità. Le combinazioni di tasti, la frequenza di ripetizione automatica dei tasti e le pressioni involontarie dei tasti possono essere ostacoli significativi per gli utenti con mobilità ridotta della mano. Queste carenze di battitura sono risolte da tre funzioni di accessibilità relative alla tastiera: *Tasti permanenti*, *Tasti di rimbalzo* e *Tasti lenti*.

La funzione *Sticky keys*, che si trova nella sezione *Assistenza alla digitazione* della configurazione di Accesso Universale di Gnome, consente all'utente di digitare le scorciatoie da tastiera un tasto alla volta. Quando abilitate, le combinazioni di tasti come `Ctrl` + `C` non devono essere tenute premute contemporaneamente. L'utente può prima premere il tasto `Ctrl`, rilasciarlo e quindi premere il tasto `C`. In KDE, questa opzione si trova nella scheda *Modifica tasti* della finestra delle impostazioni di accessibilità. KDE offre anche l'opzione *Blocco tasti*: se abilitata, i tasti `Alt`, `Ctrl` e `Shift` rimarranno "premuti" se l'utente li preme due volte, in modo simile al comportamento del tasto Caps Lock. Come per quest'ultima, l'utente dovrà premere nuovamente il tasto corrispondente per rilasciarlo.

La funzione *Tasti di rimbalzo (Bounce keys)* cerca di inibire le pressioni involontarie dei tasti inserendo un ritardo tra di loro, ovvero, una nuova pressione del tasto verrà accettata solo dopo che è trascorso un periodo di tempo specificato dall'ultima pressione del tasto. Gli utenti con tremori alle mani potrebbero trovare utile la funzione Bounce keys per evitare di premere un tasto più volte quando intendono premerlo solo una volta. In Gnome, questa funzionalità riguarda solo le stesse ripetizioni dei tasti, mentre in KDE riguarda anche qualsiasi altro tasto premuto e si trova nella scheda *Filtri tastiera*.

La funzione *Tasti lenti (Slow keys)* aiuta anche a evitare la pressione accidentale dei tasti. Se abilitati, i tasti lenti richiedono all'utente di tenere premuto il tasto per un periodo di tempo

specificato prima che l'input venga accettato. A seconda delle esigenze dell'utente, può anche essere utile regolare la ripetizione automatica tenendo premuto un tasto, opzione disponibile nelle impostazioni della tastiera.

Le funzioni di accessibilità dei tasti permanenti e dei tasti lenti possono essere attivate e disattivate con *Gesti di attivazione* eseguiti sulla tastiera. In KDE, l'opzione *Usa gesti per l'attivazione di tasti permanenti e tasti lenti* dovrebbe essere selezionata per abilitare i gesti di attivazione, mentre in Gnome questa funzionalità è chiamata *Abilita da tastiera* nella finestra di configurazione *Supporto alla Digitazione*. Una volta abilitati i gesti di attivazione, la funzione Tasti permanenti verrà attivata dopo aver premuto il tasto `Shift` cinque volte consecutive. Per attivare la funzione Tasti lenti è necessario tenere premuto il tasto `Shift` per otto secondi consecutivi.

Gli utenti che trovano più comodo utilizzare la tastiera al posto del mouse o del touchpad possono ricorrere a scorciatoie da tastiera per spostarsi nell'ambiente desktop. Inoltre, una funzionalità chiamata *Mouse Keys* consente all'utente di controllare il puntatore del mouse stesso con il tastierino numerico, presente nelle tastiere desktop di dimensioni standard e nei laptop più grandi.

Il tastierino numerico è disposto in una griglia quadrata, quindi ogni numero corrisponde a una direzione: `2` sposta il cursore verso il basso, `4` sposta il cursore a sinistra, `7` sposta il cursore verso nord-ovest, ecc. Per impostazione predefinita, il numero `5` corrisponde al clic sinistro del mouse.

Mentre in Gnome c'è solo un comando per abilitare l'opzione Tasti del mouse nella finestra delle impostazioni di Accesso universale, in KDE le impostazioni dei tasti del mouse si trovano in *Impostazioni di sistema*, *Mouse*, *Navigazione d tastiera* e opzioni come la velocità e l'accelerazione possono essere personalizzate.

TIP

I Tasti Lenti, i Tasti Stick, i Tasti Bounce e i tasti mouse sono funzioni di accessibilità fornite da AccessX, una risorsa all'interno dell'estensione della tastiera X del sistema X Window. Le impostazioni di AccessX possono essere modificate anche dalla riga di comando, con il comando `xkbset`.

Il mouse o il touchpad possono essere utilizzati per generare input da tastiera quando l'utilizzo della tastiera è troppo scomodo o non è possibile. Se lo switch *Screen Keyboard* nelle impostazioni di Accesso Universale di Gnome è abilitato, verrà visualizzata una tastiera su schermo ogni volta che il cursore si trova in un campo di testo e viene inserito un nuovo testo facendo clic sui tasti con il mouse o con il touchscreen, proprio come con la tastiera virtuale di uno smartphone.

KDE e altri ambienti desktop potrebbero non fornire la tastiera su schermo per impostazione predefinita, ma il pacchetto *onboard* può essere installato manualmente per fornire una semplice tastiera su schermo che può essere utilizzata in qualsiasi ambiente desktop. Dopo l'installazione, sarà disponibile come normale applicazione nel *launcher* delle applicazioni.

Il comportamento del puntatore può essere modificato anche se fare clic e trascinare il mouse provoca dolore o è impraticabile per qualsiasi altro motivo. Se l'utente non è in grado di fare clic con il pulsante del mouse abbastanza velocemente da attivare un evento di doppio clic, per esempio, l'intervallo di tempo per premere il pulsante del mouse una seconda volta per fare doppio clic può essere aumentato nelle *Preferenze del mouse* nella finestra di configurazione del sistema.

Se l'utente non è in grado di premere uno o nessuno dei pulsanti del mouse, i clic del mouse possono essere simulati utilizzando tecniche diverse. Nella sezione *Assistenza clic* di *Accesso universale Gnome*, l'opzione *Simula un clic destro del mouse* genererà un clic destro se l'utente tiene premuto il pulsante sinistro del mouse. Con l'opzione *Simula clic al passaggio del mouse* abilitata, verrà attivato un evento clic quando l'utente tiene fermo il mouse. In KDE, l'applicazione *KMouseTool* fornirà queste stesse funzionalità per assistere nelle azioni del mouse.

Disabilità Visive

Gli utenti con problemi di vista potrebbero comunque essere in grado di utilizzare lo schermo del monitor per interagire con il computer. A seconda delle esigenze dell'utente, è possibile apportare molte regolazioni visive per migliorare i dettagli altrimenti difficili da vedere del desktop grafico standard.

La sezione *Seeing* di Gnome delle impostazioni di *Accesso universale* fornisce opzioni che possono aiutare le persone con problemi di vista:

Contrasto elevato

renderà le finestre e i pulsanti più facili da vedere disegnandoli con colori più nitidi.

Testo grande

ingrandirà la dimensione del carattere dello schermo standard.

Dimensioni cursore

permette di scegliere un cursore del mouse più grande, facilitandone così la localizzazione sullo schermo.

Alcune di queste regolazioni non sono strettamente correlate alle funzioni di accessibilità, quindi possono essere trovate nella sezione dell'aspetto dell'utilità di configurazione fornita da altri ambienti desktop. Un utente che ha difficoltà a discernere tra gli elementi visivi può scegliere un tema ad alto contrasto per facilitare l'identificazione di pulsanti, finestre sovrapposte, ecc.

Se le regolazioni dell'aspetto da sole non sono sufficienti per migliorare la leggibilità dello schermo, è possibile utilizzare un programma di ingrandimento dello schermo. Questa funzione è

chiamata *Zoom* nelle impostazioni di *Accesso universale* di Gnome, dove è possibile personalizzare opzioni come il rapporto di ingrandimento, la posizione della lente d'ingrandimento e le regolazioni del colore.

In KDE, il programma *KMagnifier* fornisce le stesse funzionalità, ma è disponibile come una normale applicazione tramite il lanciatore di applicazioni. Altri ambienti desktop possono fornire i propri ingranditori dello schermo. Xfce, per esempio, ingrandirà e rimpicciolirà lo schermo ruotando la rotellina del mouse mentre il tasto `Alt` è premuto.

Infine, gli utenti per i quali l'interfaccia grafica non è un'opzione possono utilizzare uno *screen reader* per interagire con il computer. Indipendentemente dall'ambiente desktop scelto, lo screen reader più popolare per i sistemi Linux è *Orca,* che è installato di default nella maggior parte delle distribuzioni. Orca genera una voce sintetizzata per segnalare eventi sullo schermo e leggere il testo sotto il cursore del mouse. Orca funziona anche con *schermi braille aggiornabili,* dispositivi speciali che visualizzano caratteri braille sollevando piccoli spilli che possono essere sentiti con la punta delle dita. Non tutte le applicazioni desktop sono completamente adattate per i lettori di schermo e non tutti gli utenti troveranno facile utilizzarle, quindi è importante fornire agli utenti il maggior numero di strategie di lettura dello schermo tra cui scegliere.

Esercizi Guidati

1. Quale funzione di accessibilità potrebbe aiutare un utente ad alternare le finestre aperte utilizzando la tastiera, considerando che l'utente non è in grado di premere contemporaneamente i tasti `Alt` e `Tab`?

2. In che modo la funzione di accessibilità *Bounce keys* può aiutare gli utenti i cui tremori involontari alle mani disturbano la digitazione?

3. Quali sono i gesti di attivazione più comuni per la funzione di accessibilità *Tasti permanenti*?

Esercizi Esplorativi

1. Le funzioni di accessibilità potrebbero non essere fornite da una singola applicazione e possono variare da un ambiente desktop all'altro. In KDE, quale applicazione aiuta le persone con lesioni da sforzo ripetitivo facendo clic con il mouse ogni volta che il cursore del mouse si ferma brevemente?

2. Quali tra gli aspetti dell'ambiente grafico possono essere modificati per rendere più facile per le persone leggere il testo sullo schermo?

3. In che modo l'applicazione *Orca* può aiutare gli utenti ipovedenti a interagire con l'ambiente desktop?

Sommario

Questa lezione tratta le funzioni generali di accessibilità disponibili nei sistemi Linux. Tutti i principali ambienti desktop, specialmente Gnome e KDE, forniscono molte applicazioni integrate e di terze parti per assistere le persone con disabilità visive o mobilità ridotta. La lezione affronta i seguenti argomenti:

- Come modificare le impostazioni di accessibilità.
- Modi alternativi per utilizzare la tastiera e il mouse.
- Miglioramenti del desktop per i non vedenti.

I comandi e le procedure affrontati sono stati:

- Impostazioni di accessibilità della tastiera: tasti permanenti, tasti lenti, tasti di rimbalzo.
- Generare artificialmente eventi del mouse.
- Tastiera sullo schermo.
- Impostazioni visive per migliorare la leggibilità.
- Temi desktop ad alto contrasto / stampa di grandi dimensioni.
- Lenti di ingrandimento dello schermo.
- Orca screen reader.

 | Questa opera è concessa in licenza CC BY-NC-ND 4.0. |

Risposte agli Esercizi Guidati

1. Quale funzione di accessibilità potrebbe aiutare un utente ad alternare le finestre aperte utilizzando la tastiera, considerando che l'utente non è in grado di premere contemporaneamente i tasti `Alt` e `Tab`?

   La funzione Tasti permanenti, che consente all'utente di digitare le scorciatoie da tastiera un tasto alla volta.

2. In che modo la funzione di accessibilità *Bounce keys* può aiutare gli utenti i cui tremori involontari alle mani disturbano la digitazione?

   Con i tasti Bounce abilitati, una nuova pressione di un tasto sarà accettata solo dopo che è trascorso un periodo di tempo specificato dall'ultima pressione del precedente.

3. Quali sono i gesti di attivazione più comuni per la funzione di accessibilità *Tasti permanenti*?

   Se i Gesti di attivazione sono abilitati, la funzione dei tasti permanenti verrà attivata dopo aver premuto il tasto `Shift` cinque volte consecutive.

Risposte agli Esercizi Esplorativi

1. Le funzioni di accessibilità potrebbero non essere fornite da una singola applicazione e possono variare da un ambiente desktop all'altro. In KDE, quale applicazione aiuta le persone con lesioni da sforzo ripetitivo facendo clic con il mouse ogni volta che il cursore del mouse si ferma brevemente?

   L'applicazione *KMouseTool* .

2. Quali tra gli aspetti dell'ambiente grafico possono essere modificati per rendere più facile per le persone leggere il testo sullo schermo?

   L'impostazione di una dimensione grande del carattere dello schermo nella configurazione del desktop renderà tutti i testi dello schermo più facili da leggere.

3. In che modo l'applicazione *Orca* può aiutare gli utenti ipovedenti a interagire con l'ambiente desktop?

   Orca è uno screen reader che genera una voce sintetizzata per segnalare eventi sullo schermo e leggere il testo sotto il cursore del mouse. Funziona anche con dispositivi chiamati *display braille aggiornabili*, in modo che l'utente possa identificare il testo con schemi tattili.



Linux Professional Institute

Argomento 107: Attività Amministrative



107.1 Gestire account utente e gruppo e file di sistema correlati

Obiettivi LPI di riferimento

LPIC-1 version 5.0, Exam 102, Objective 107.1

Peso

5

Aree di Conoscenza Chiave

- Aggiungere, modificare e rimuovere utenti e gruppi.
- Gestire le informazioni utente/gruppo nei database di password/gruppo.
- Creare e gestire scopi speciali e account limitati.

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

- `/etc/passwd`
- `/etc/shadow`
- `/etc/group`
- `/etc/skel/`
- `chage`
- `getent`
- `groupadd`
- `groupdel`
- `groupmod`
- `passwd`
- `useradd`

- `userdel`
- `usermod`



107.1 Lezione 1

Certificazione:	LPIC-1
Versione:	5.0
Argomento:	107 Attività Amministrative
Obiettivo:	107.1 Gestire account utente e gruppo e file di sistema correlati
Lezione:	1 di 2

Introduzione

L'amministrazione di utenti e gruppi è una parte molto importante del lavoro di qualsiasi amministratore di sistema. Le moderne distribuzioni Linux implementano interfacce grafiche che permettono di gestire tutte le attività relative a questo aspetto chiave in modo rapido e semplice. Queste interfacce sono diverse l'una dall'altra in termini di layout grafico, ma le caratteristiche sono le stesse. Con questi strumenti è possibile visualizzare, modificare, aggiungere ed eliminare utenti e gruppi locali. Tuttavia, per una gestione più avanzata è necessario lavorare attraverso la riga di comando.

Aggiungere un Account Utente

In Linux è possibile aggiungere un nuovo account utente con il comando `useradd`. Per esempio, agendo con i privilegi di *root*, è possibile creare un nuovo account utente chiamato `michael` con un'impostazione predefinita, come segue:

```
# useradd michael
```

Quando si esegue il comando `useradd`, le informazioni sull'utente e sul gruppo memorizzate nei database delle password e dei gruppi vengono aggiornate per l'account utente appena creato e, se specificato, viene creata anche la home directory del nuovo utente. Viene anche creato un gruppo con lo stesso nome del nuovo account utente.

Una volta creato il nuovo utente, è possibile impostare la sua password usando il comando `passwd`. È possibile rivedere i propri User ID (UID), Group ID (GID) e i gruppi a cui appartiene attraverso i comandi `id` e `groups`.

```
# passwd michael
Changing password for user michael.
New UNIX password:
Retype new UNIX password:
passwd: all authentication tokens updated successfully.
# id michael
uid=1000(michael) gid=100(michael) groups=100(michael)
# groups michael
michael : michael
```

NOTE

Ricorda che qualsiasi utente può rivedere il proprio UID, GID e i gruppi a cui appartiene semplicemente usando i comandi `id` e `groups` senza argomenti, e che qualsiasi utente può cambiare la propria password usando il comando `passwd`. Tuttavia solo gli utenti con i privilegi di root possono cambiare la password di *qualsiasi* utente.

Le opzioni più importanti che si applicano al comando `useradd` sono:

`-c`

Crea un nuovo account utente con commenti personalizzati (per esempio il nome completo dell'utente).

`-d`

Crea un nuovo account utente con una home directory personalizzata.

`-e`

Crea un nuovo account utente impostando una data specifica in cui sarà disabilitato.

`-f`

Crea un nuovo account utente impostando il numero di giorni dopo la scadenza della password durante i quali l'utente deve aggiornare la password (altrimenti l'account sarà disabilitato).

`-g`
Crea un nuovo account utente con un GID specifico.

`-G`
Crea un nuovo account utente aggiungendolo a più gruppi secondari.

`-k`
Crea un nuovo account utente copiando i file *skeleton* da una specifica directory personalizzata (questa opzione è valida solo se viene specificata l'opzione `-m` o `--create-home`).

`-m`
Crea un nuovo account utente con la sua directory home (se non esiste).

`-M`
Crea un nuovo account utente senza la sua home directory.

`-s`
Crea un nuovo account utente con una specifica shell di login.

`-u`
Crea un nuovo account utente con uno specifico UID.

Vedi le pagine di manuale del comando `useradd` per la lista completa delle opzioni.

Modificare un Account Utente

A volte è necessario cambiare un attributo di un account utente esistente, come il nome di login, la shell di login, la data di scadenza della password e così via. In questi casi, è necessario utilizzare il comando `usermod`.

```
# usermod -s /bin/tcsh michael
# usermod -c "Michael User Account" michael
```

Proprio come il comando `useradd`, il comando `usermod` richiede i privilegi di root.

Negli esempi qui sopra, viene cambiata prima la shell di login di `michael` e poi viene aggiunta una breve descrizione a questo account utente. Ricorda che puoi modificare più attributi contemporaneamente, specificandoli in un unico comando.

Le opzioni più importanti che si applicano al comando `usermod` sono:

`-c`

Aggiunge un breve commento all'account utente specificato.

`-d`

Cambia la home directory dell'account utente specificato. Quando viene usato con l'opzione `-m`, il contenuto della home directory corrente viene spostato nella nuova home directory, che viene creata se non esiste già.

`-e`

Imposta la data di scadenza dell'account utente specificato.

`-f`

Imposta il numero di giorni dopo la scadenza di una password durante i quali l'utente deve aggiornare la password (altrimenti l'account verrà disabilitato).

`-g`

Cambia il gruppo primario dell'account utente specificato (il gruppo deve esistere).

`-G`

Aggiunge gruppi secondari all'account utente specificato. Ogni gruppo deve esistere e deve essere separato dal successivo da una virgola, senza spazi intermedi. Se usata da sola, questa opzione rimuove tutti i gruppi esistenti a cui l'utente appartiene, mentre se usata con l'opzione `-a`, aggiunge semplicemente nuovi gruppi secondari a quelli esistenti.

`-l`

Cambia il nome di login dell'account utente specificato.

`-L`

Blocca l'account utente specificato. Inserisce un punto esclamativo davanti alla password criptata nel file `/etc/shadow`, disabilitando così l'accesso con password per quell'utente.

`-s`

Cambia la shell di login dell'account utente specificato.

`-u`

Cambia l'UID dell'account utente specificato.

`-U`

Sblocca l'account utente specificato. Rimuove il punto esclamativo davanti alla password criptata con il file `/etc/shadow`.

Consultare le pagine di manuale del comando `usermod` per la lista completa delle opzioni.

TIP

Ricorda che quando cambiate il nome di login di un account utente dovresti probabilmente rinominare la home directory di quell'utente e altri elementi relativi all'utente come i file di *spool* della posta. Ricorda anche che quando cambia l'UID di un account utente dovresti probabilmente fissare la proprietà dei file e delle directory al di fuori della home directory dell'utente (l'ID utente viene cambiato automaticamente per la casella di posta dell'utente e per tutti i file di proprietà dell'utente e situati nella home directory dell'utente).

Eliminare un Account Utente

Se vuoi eliminare un account utente, puoi usare il comando `userdel`. In particolare, questo comando aggiorna le informazioni memorizzate nei database degli account, cancellando tutte le voci che si riferiscono all'utente specificato. L'opzione `-r` rimuove anche la directory home dell'utente e tutto il suo contenuto, insieme allo spool di posta dell'utente. Altri file, situati altrove, devono essere cercati e cancellati manualmente.

```
# userdel -r michael
```

Come per `useradd` e `usermod`, hai bisogno dell'autorità di root per cancellare gli account utente.

Aggiungere, Modificare e Rimuovere i Gruppi

Proprio come per la gestione degli utenti, puoi aggiungere, modificare e cancellare gruppi usando i comandi `groupadd`, `groupmod` e `groupdel` con privilegi di root. Se vuoi creare un nuovo gruppo chiamato `developer`, puoi eseguire il seguente comando:

```
# groupadd -g 1090 developer
```

L'opzione `-g` di questo comando crea un gruppo con un GID specifico.

WARNING

Ricorda che quando aggiungi un nuovo account utente, il gruppo primario e i gruppi secondari a cui appartiene devono esistere prima di lanciare il comando `useradd`.

In seguito, se vuoi rinominare il gruppo da `developer` a `web-developer` e cambiare il suo GID, puoi eseguire quanto segue:

```
# groupmod -n web-developer -g 1050 developer
```

TIP Ricorda che se cambi il GID usando l'opzione `-g`, devi cambiare il GID di tutti i file e le directory che devono continuare ad appartenere al gruppo.

Infine, se vuoi eliminare il gruppo `web-developer`, puoi eseguire quanto segue:

```
# groupdel web-developer
```

Non puoi eliminare un gruppo se è il gruppo principale di un account utente. Pertanto, è necessario rimuovere l'utente prima di rimuovere il gruppo. Come per gli utenti, se si elimina un gruppo, i file appartenenti a quel gruppo rimangono nel filesystem e non vengono cancellati o assegnati a un altro gruppo.

La Directory Skeleton

Quando si aggiunge un nuovo account utente, anche creando la sua home directory, la home directory appena creata viene popolata con file e cartelle che vengono copiati dalla directory *skeleton* (di default `/etc/skel`). L'idea alla base è semplice: un amministratore di sistema vuole aggiungere nuovi utenti che abbiano gli stessi file e cartelle nella loro home directory. Quindi, se vuoi personalizzare i file e le cartelle che vengono creati automaticamente nella home directory dei nuovi account utente, devi aggiungere questi nuovi file e cartelle alla directory skeleton.

TIP Nota che se vuoi elencare tutti i file e le directory nella directory skeleton, devi usare il comando `ls -al`.

Il File `/etc/login.defs`

In Linux, il file `/etc/login.defs` specifica i parametri di configurazione che controllano la creazione di utenti e gruppi. Inoltre, i comandi mostrati nelle sezioni precedenti prendono i valori predefiniti da questo file.

Le direttive più importanti sono:

`UID_MIN` e `UID_MAX`
: L'intervallo di ID utente che può essere assegnato ai nuovi utenti ordinari.

`GID_MIN` e `GID_MAX`
: L'intervallo di ID di gruppo che può essere assegnato ai nuovi gruppi ordinari.

`CREATE_HOME`

Specifica se una home directory deve essere creata di default per i nuovi utenti.

`USERGROUPS_ENAB`

Specifica se il sistema debba creare di default un nuovo gruppo per ogni nuovo account utente con lo stesso nome dell'utente, e se, eliminando l'account utente, debba rimuovere anche il gruppo primario dell'utente qualora non contenga più membri.

`MAIL_DIR`

La directory di spool della posta.

`PASS_MAX_DAYS`

Il numero massimo di giorni in cui una password può essere usata.

`PASS_MIN_DAYS`

Il numero minimo di giorni consentito tra un cambio di password e l'altro.

`PASS_MIN_LEN`

La lunghezza minima accettabile della password.

`PASS_WARN_AGE`

Il numero di giorni di preavviso prima che una password scada.

TIP Quando si gestiscono utenti e gruppi, controllare sempre questo file per visualizzare ed eventualmente cambiare il comportamento predefinito del sistema se necessario.

Il Comando `passwd`

Questo comando è usato principalmente per cambiare la password di un utente. Come descritto prima, ogni utente può cambiare la propria password, ma solo root può cambiare la password di *qualsiasi* utente. Questo accade perché il comando `passwd` ha il bit SUID impostato (una `s` al posto del flag eseguibile per il proprietario), il che significa che viene eseguito con i privilegi del proprietario del file (quindi root).

```
# ls -l /usr/bin/passwd
-rwsr-xr-x 1 root root 42096 mag 17  2015 /usr/bin/passwd
```

A seconda delle opzioni `passwd` usate, si possono controllare aspetti specifici della scadenza delle password:

`-d`

Cancella la password di un account utente (disabilitando così l'utente).

`-e`

Forza l'account utente a cambiare la password.

`-i`

Imposta il numero di giorni di inattività dopo la scadenza di una password durante i quali l'utente dovrebbe aggiornare la password (altrimenti l'account sarà disabilitato).

`-l`

Blocca l'account utente (la password criptata è preceduta da un punto esclamativo nel file `/etc/shadow`).

`-n`

Imposta la durata minima della password.

`-S`

Mostra informazioni sullo stato della password di uno specifico account utente.

`-u`

Sblocca l'account utente (il punto esclamativo viene rimosso dal campo password nel file `/etc/shadow`).

`-x`

Imposta la durata massima della password.

`-w`

Imposta il numero di giorni di preavviso prima che la password scada, durante i quali l'utente viene avvisato che la password deve essere cambiata.

NOTE

I gruppi possono anche avere una password, che può essere impostata usando il comando `gpasswd`. Gli utenti che non sono membri di un gruppo ma ne conoscono la password possono unirsi temporaneamente a esso usando il comando `newgrp`. Ricorda che `gpasswd` è usato anche per aggiungere e rimuovere utenti da un gruppo e per impostare la lista degli amministratori e dei membri ordinari del gruppo.

Il Comando `chage`

Questo comando, che sta per “change age”, è usato per cambiare le informazioni sulla scadenza della password di un utente. Il comando `chage` è limitato a root, tranne che per l’opzione `-l`, che può essere usata dagli utenti comuni per elencare le informazioni sulla password del proprio account.

Le altre opzioni che si applicano al comando `chage` sono:

`-d`

Imposta l’ultimo cambio di password per un account utente.

`-E`

Imposta la data di scadenza per un account utente.

`-I`

Imposta il numero di giorni di inattività dopo la scadenza di una password durante i quali l’utente dovrebbe aggiornare la password (altrimenti l’account sarà disabilitato).

`-m`

Imposta la durata minima della password per un account utente.

`-M`

Imposta la durata massima della password per un account utente.

`-W`

Imposta il numero di giorni di preavviso prima della scadenza della password durante i quali l’utente viene avvisato che la password deve essere cambiata.

Esercizi Guidati

1. Per ciascuno dei seguenti comandi, identifica lo scopo corrispondente:

`usermod -L`	
`passwd -u`	
`chage -E`	
`groupdel`	
`useradd -s`	
`groupadd -g`	
`userdel -r`	
`usermod -l`	
`groupmod -n`	
`useradd -m`	

2. Per ciascuno dei seguenti comandi `passwd`, identifica il corrispondente comando `chage`:

`passwd -n`	
`passwd -x`	
`passwd -w`	
`passwd -i`	
`passwd -S`	

3. Spiega in dettaglio lo scopo dei comandi nella domanda precedente:

4. Quali comandi si possono usare per bloccare un account utente? E quali comandi per sbloccarlo?

Esercizi Esplorativi

1. Usando il comando `groupadd`, crea i gruppi `administrators` e `developers`. Supponiamo che tu stia lavorando come root.

2. Ora che hai creato questi gruppi, esegui il seguente comando: `useradd -G administrators,developers kevin`. Quali operazioni esegue questo comando? Assumiamo che `CREATE_HOME` e `USERGROUPS_ENAB` in `/etc/login.defs` siano impostati su `yes`.

3. Crea un nuovo gruppo chiamato `designers`, rinominalo in `web-designers` e aggiungi questo nuovo gruppo ai gruppi secondari dell'account utente `kevin`. Identifica tutti i gruppi a cui appartiene `kevin` e i loro ID.

4. Rimuovi solo il gruppo `developers` dai gruppi secondari di `kevin`.

5. Imposta la password per l'account utente `kevin`.

6. Usando il comando `chage`, prima controlla la data di scadenza dell'account utente `kevin` e poi cambiala al 31 dicembre 2022. Quale altro comando puoi usare per cambiare la data di scadenza di un account utente?

7. Aggiungi un nuovo account utente chiamato `emma` con UID 1050 e imposta `administrators` come gruppo primario e `developers` e `web-designer` come gruppi secondari.

8. Cambia la shell di login di `emma` in `/bin/sh`.

9. Elimina gli account utente `emma` e `kevin` e i gruppi `administrators`, `developers` e `web-designers`.

Sommario

In questa lezione abbiamo imparato:

- I fondamenti della gestione utenti e gruppi in Linux.
- Come aggiungere, modificare e rimuovere gli account utente.
- Come aggiungere, modificare e rimuovere gli account dei gruppi.
- Mantenere la directory skeloton.
- Modificare il file che controlla la creazione di utenti e gruppi.
- Cambiare le password degli account utente.
- Cambiare le informazioni sulla scadenza delle password degli account utente.

I seguenti file e comandi sono stati discussi in questa lezione:

`useradd`
: Crea un nuovo account utente.

`usermod`
: Modifica un account utente.

`userdel`
: Elimina un account utente.

`groupadd`
: Crea un nuovo account di gruppo.

`groupmod`
: Modifica un account di gruppo.

`groupdel`
: Elimina un account di gruppo.

`passwd`
: Cambia la password degli account utente e controlla tutti gli aspetti della scadenza delle password.

`chage`
: Cambia le informazioni sulla scadenza della password dell'utente.

`/etc/skel`

La posizione predefinita della directory skeleton.

`/etc/login.defs`

Il file che controlla la creazione di utenti e gruppi e fornisce valori predefiniti per diversi parametri dell'account utente.

 Questa opera è concessa in licenza CC BY-NC-ND 4.0. |

Risposte agli Esercizi Guidati

1. Per ciascuno dei seguenti comandi, identifica lo scopo corrispondente:

`usermod -L`	Blocca l'account dell'utente
`passwd -u`	Sblocca l'account utente
`chage -E`	Imposta la data di scadenza per l'account utente
`groupdel`	Elimina il gruppo
`useradd -s`	Crea un nuovo account utente con una specifica shell di login
`groupadd -g`	Crea un nuovo gruppo con un GID specifico
`userdel -r`	Rimuove l'account utente e tutti i file nella sua home directory, la home directory stessa e lo spool di posta dell'utente
`usermod -l`	Cambia il nome di login dell'account utente
`groupmod -n`	Cambiare il nome del gruppo
`useradd -m`	Creare un nuovo account utente e la sua home directory

2. Per ciascuno dei seguenti comandi `passwd`, identifica il corrispondente comando `chage`:

`passwd -n`	`chage -m`
`passwd -x`	`chage -M`
`passwd -w`	`chage -W`
`passwd -i`	`chage -I`
`passwd -S`	`chage -l`

3. Spiega in dettaglio lo scopo dei comandi nella domanda precedente:

   In Linux, puoi usare il comando `passwd -n` (o `chage -m`) per impostare il numero minimo di giorni tra i cambi di password, il comando `passwd -x` (o `chage -M`) per impostare il numero massimo di giorni durante i quali una password è valida, il comando `passwd -w` (o `chage -W`) per impostare il numero di giorni di avviso prima che la password scada, il comando `passwd -i` (o `chage -I`) per impostare il numero di giorni di inattività durante i quali l'utente

dovrebbe cambiare la password; infine il comando `passwd -S` (o `chage -l`) per mostrare brevi informazioni sulla password dell'account utente.

4. Quali comandi si possono usare per bloccare un account utente? E quali comandi per sbloccarlo?

   Se vuoi bloccare un account utente, puoi usare uno di questi comandi: `usermod -L`, `usermod --lock` e `passwd -l`. Invece, se vuoi sbloccarlo, puoi usare `usermod -U`, `usermod --unlock` e `passwd -u`.

Risposte agli Esercizi Esplorativi

1. Usando il comando `groupadd`, crea i gruppi `administrators` e `developers`. Supponiamo che tu stia lavorando come root.

```
# groupadd administrators
# groupadd developers
```

2. Ora che hai creato questi gruppi, esegui il seguente comando: `useradd -G administrators,developers kevin`. Quali operazioni esegue questo comando? Assumiamo che `CREATE_HOME` e `USERGROUPS_ENAB` in `/etc/login.defs` siano impostati su `yes`.

   Il comando aggiunge un nuovo utente, chiamato `kevin`, alla lista degli utenti nel sistema, crea la sua home directory (`CREATE_HOME` è impostato su yes e quindi si può omettere l'opzione -m) e crea un nuovo gruppo, chiamato `kevin`, come gruppo primario di questo account utente (`USERGROUPS_ENAB` è impostato su yes). Infine, i file e le cartelle contenuti nella directory skeleton vengono copiati nella home directory di `kevin`.

3. Crea un nuovo gruppo chiamato `designers`, rinominalo in `web-designers` e aggiungi questo nuovo gruppo ai gruppi secondari dell'account utente `kevin`. Identifica tutti i gruppi a cui appartiene `kevin` e i loro ID.

```
# groupadd designers
# groupmod -n web-designers designers
# usermod -a -G web-designers kevin
# id kevin
uid=1010(kevin) gid=1030(kevin)
groups=1030(kevin),1028(administrators),1029(developers),1031(web-designers)
```

4. Rimuovi solo il gruppo `developers` dai gruppi secondari di `kevin`.

```
# usermod -G administrators,web-designers kevin
# id kevin
uid=1010(kevin) gid=1030(kevin) groups=1030(kevin),1028(administrators),1031(web-
designers)
```

   Il comando `usermod` non ha un'opzione per rimuovere un solo gruppo; quindi, è necessario specificare tutti i gruppi secondari a cui l'utente appartiene.

5. Imposta la password per l'account utente `kevin`.

```
# passwd kevin
Changing password for user kevin.
New UNIX password:
Retype new UNIX password:
passwd: all authentication tokens updated successfully.
```

6. Usando il comando `chage`, prima controlla la data di scadenza dell'account utente `kevin` e poi cambiala al 31 dicembre 2022. Quale altro comando puoi usare per cambiare la data di scadenza di un account utente?

```
# chage -l kevin | grep "Account expires"
Account expires     : never
# chage -E 2022-12-31 kevin
# chage -l kevin | grep "Account expires"
Account expires     : dec 31, 2022
```

   Il comando `usermod` con l'opzione `-e` è equivalente a `chage -E`.

7. Aggiungi un nuovo account utente chiamato `emma` con UID 1050 e imposta `administrators` come gruppo primario e `developers` e `web-designer` come gruppi secondari.

```
# useradd -u 1050 -g administrators -G developers,web-designers emma
# id emma
uid=1050(emma) gid=1028(administrators)
groups=1028(administrators),1029(developers),1031(web-designers)
```

8. Cambia la shell di login di `emma` in `/bin/sh`.

```
# usermod -s /bin/sh emma
```

9. Elimina gli account utente `emma` e `kevin` e i gruppi `administrators`, `developers` e `web-designers`.

```
# userdel -r emma
# userdel -r kevin
# groupdel administrators
# groupdel developers
# groupdel web-designers
```

107.1 Lezione 2

Certificazione:	LPIC-1
Versione:	5.0
Argomento:	107 Attività Amministrative
Obiettivo:	107.1 Gestire account utente e gruppo e file di sistema correlati
Lezione:	2 di 2

Introduzione

Gli strumenti a riga di comando discussi nella lezione precedente e le applicazioni grafiche che eseguono gli stessi compiti, fornite da ogni distribuzione, aggiornano una serie di file che memorizzano informazioni su utenti e gruppi.

Questi file si trovano sotto la directory `/etc/` e sono:

`/etc/passwd`
: Un file di sette campi delimitati da due punti contenente informazioni di base sugli utenti.

`/etc/group`
: Un file di quattro campi delimitati da due punti contenente informazioni di base sui gruppi.

`/etc/shadow`
: Un file di nove campi delimitati da due punti contenenti password utente criptate.

`/etc/gshadow`

Un file di quattro campi delimitati da due punti contenenti password di gruppo criptate.

Anche se questi quattro file sono in testo semplice, non dovrebbero essere modificati direttamente, ma sempre attraverso gli strumenti forniti dalla distribuzione che state usando.

/etc/passwd

Questo è un file leggibile da tutti e contiene una lista di utenti, ognuno su una linea separata. Ogni linea consiste di sette campi delimitati da due punti:

Username

Il nome usato quando l'utente accede al sistema.

Password

La password criptata (o una `x` se si usano le *shadow* password).

User ID (UID)

Il numero ID assegnato all'utente nel sistema.

Group ID (GID)

Il numero del gruppo primario dell'utente nel sistema.

GECOS

Un campo di commento opzionale, che è usato per aggiungere informazioni extra sull'utente (come il nome completo). Il campo può contenere più voci separate da virgola.

Home Directory

Il percorso assoluto della *home directory* dell'utente.

Shell

Il percorso assoluto del programma che viene lanciato automaticamente quando l'utente accede al sistema (di solito una shell interattiva come `/bin/bash`).

/etc/group

Questo è un file leggibile da tuti che contiene una lista di gruppi, ognuno su una linea separata. Ogni linea consiste di quattro campi delimitati da due punti:

Group Name

Il nome del gruppo.

Group Password

La password criptata del gruppo (o una `x` se si usano le *shadow* password).

Group ID (GID)

Il numero ID assegnato al gruppo nel sistema.

Member List

Una lista delimitata da virgole di utenti appartenenti al gruppo, eccetto quelli per i quali questo è il gruppo principale.

/etc/shadow

Questo è un file leggibile solo da *root* e da utenti con privilegi di root che contiene password utente criptate, ciascuna su una linea separata. Ogni linea consiste di nove campi delimitati da due punti:

Username

Il nome usato quando l'utente accede al sistema.

Encrypted Password

La password criptata dell'utente (se il valore inizia con `!`, l'account è bloccato).

Date of Last Password Change

La data dell'ultimo cambio di password, come numero di giorni dal 01/01/1970 (un valore di 0 significa che l'utente deve cambiare la password al suo prossimo login).

Minimum Password Age

Il numero minimo di giorni, dopo un cambio di password, che devono trascorrere prima che l'utente sia autorizzato a cambiare nuovamente la password.

Maximum Password Age

Il numero massimo di giorni che devono trascorrere prima che sia richiesto un cambio di password.

Password Warning Period

Il numero di giorni, prima che la password scada, durante i quali l'utente viene avvertito che la password deve essere cambiata.

Password Inactivity Period

Il numero di giorni dopo la scadenza della password durante i quali l'utente dovrebbe aggiornare la password. Dopo questo periodo, se l'utente non cambia la password, l'account sarà disabilitato.

Account Expiration Date

La data, espressa come numero di giorni dal 01/01/1970, in cui l'account utente sarà disattivato (un campo vuoto significa che l'account utente non scadrà mai).

A reserved field

Un campo che è riservato per un uso futuro.

/etc/gshadow

Questo è un file leggibile solo da root e da utenti con privilegi di root che contiene password di gruppo criptate, ciascuna su una linea separata. Ogni linea consiste di quattro campi delimitati da due punti:

Group Name

Il nome del gruppo.

Encrypted Password

La password criptata per il gruppo (è usata quando un utente, che non è membro del gruppo, vuole unirsi al gruppo usando il comando `newgrp` — se la password inizia con `!`, a nessuno è permesso accedere al gruppo con `newgrp`).

Group Administrators

Una lista delimitata da virgole degli amministratori del gruppo (possono cambiare la password del gruppo e possono aggiungere o rimuovere membri del gruppo con il comando `gpasswd`).

Group Members

Una lista delimitata da virgole dei membri del gruppo.

Filtrare i Database delle Password e dei Gruppi

Molto spesso può essere necessario rivedere le informazioni su utenti e gruppi memorizzati in questi quattro file e cercare record specifici. Per eseguire questo compito, puoi usare il comando `grep` o in alternativa concatenare `cat` e `grep`.

```
# grep emma /etc/passwd
```

```
emma:x:1020:1020:User Emma:/home/emma:/bin/bash
# cat /etc/group | grep db-admin
db-admin:x:1050:grace,frank
```

Un altro modo per accedere a questi database è usare il comando `getent`. In generale, questo comando visualizza le voci dei database supportati dalle librerie *Name Service Switch* (NSS) e richiede il nome del database e una chiave di ricerca. Se non viene fornito alcun argomento chiave, vengono visualizzate tutte le voci del database specificato (a meno che il database non supporti l'enumerazione). Altrimenti, se vengono forniti uno o più argomenti chiave, il database viene filtrato di conseguenza.

```
# getent passwd emma
emma:x:1020:1020:User Emma:/home/emma:/bin/bash
# getent group db-admin
db-admin:x:1050:grace,frank
```

Il comando `getent` non richiede l'autorità di root; devi solo essere in grado di leggere il database da cui vuoi recuperare i record.

NOTE Ricorda che `getent` può solo accedere ai database configurati nel file `/etc/nsswitch.conf`.

Esercizi Guidati

1. Osserva il seguente output e rispondi alle seguenti domande:

```
# cat /etc/passwd | grep '\(root\|mail\|catherine\|kevin\)'
root:x:0:0:root:/root:/bin/bash
mail:x:8:8:mail:/var/spool/mail:/sbin/nologin
catherine:x:1030:1025:User Chaterine:/home/catherine:/bin/bash
kevin:x:1040:1015:User Kevin:/home/kevin:/bin/bash
# cat /etc/group | grep '\(root\|mail\|db-admin\|app-developer\)'
root:x:0:
mail:x:8:
db-admin:x:1015:emma,grace
app-developer:x:1016:catherine,dave,christian
# cat /etc/shadow | grep '\(root\|mail\|catherine\|kevin\)'
root:$6$1u36Ipok$ljt8ooPMLewAhkQPf.lYgGopAB.jClTO6ljsdczxvkLPkpi/amgp.zyfAN680zrLLp2avvpd
KA0llpssdfcPppOp:18015:0:99999:7:::
mail:*:18015:0:99999:7:::
catherine:$6$ABCD25jlld14hpPthEFGnnssEWw1234yioMpliABCdef1f3478kAfhhAfgbAMjY1/BAeeAsl/FeE
dddKd12345g6kPACcik:18015:20:90:5:::
kevin:$6$DEFGabc123WrLp223fsvp0ddx3dbA7pPPc4LMaa123u6Lp02Lpvm123456pyphhh5ps012vbArL245.P
R1345kkA3Gas12P:18015:0:60:7:2::
# cat /etc/gshadow | grep '\(root\|mail\|db-admin\|app-developer\)'
root:*::
mail:*::
db-admin:!:emma:emma,grace
app-developer:!::catherine,dave,christian
```

   - Qual è l'ID utente (UID) e l'ID gruppo (GID) di `root` e `catherine`?

   - Qual è il nome del gruppo principale di `kevin`? Ci sono altri membri in questo gruppo?

   - Quale shell è impostata per `mail`? Che cosa significa?

   - Chi sono i membri del gruppo `app-developer`? Quali tra questi membri sono amministratori del gruppo e quali sono membri ordinari?

- Qual è la durata minima della password per `catherine`? E qual è la durata massima della password?

- Qual è il periodo di inattività della password per `kevin`?

2. Per convenzione, quali ID sono assegnati agli account di sistema e quali agli utenti ordinari?

3. Come si fa a scoprire se un account utente, che prima era in grado di accedere al sistema, ora è bloccato? Supponiamo che il vostro sistema utilizzi le *shadow* password.

Esercizi Esplorativi

1. Crea un account utente chiamato `christian` usando il comando `useradd -m` e identifica il suo User ID (UID), Group ID (GID) e la shell.

2. Identifica il nome del gruppo primario di `christian`. Che cosa puoi dedurre?

3. Usando il comando `getent`, esamina le informazioni sulla scadenza della password per l'account utente `christian`.

4. Aggiungi il gruppo `editor` ai gruppi secondari di `christian`. Assumi che questo gruppo contenga già `emma`, `dave` e `frank` come membri ordinari. Come puoi verificare che non ci siano amministratori per questo gruppo?

5. Esegui il comando `ls -l /etc/passwd /etc/group /etc/shadow /etc/gshadow` e descrivi l'output che ricevi in termini di permessi dei file. Quali di questi quattro file sono "mascherati" per ragioni di sicurezza? Supponiamo che il tuo sistema usi le *shadow* password.

Sommario

In questa lezione abbiamo imparato:

- La posizione dei file che memorizzano le informazioni su utenti e gruppi.
- La gestione delle informazioni su utenti e gruppi memorizzate nei database di password e gruppi.
- Il recupero delle informazioni dai database di password e gruppi.

I seguenti file e comandi sono stati discussi in questa lezione:

`/etc/passwd`
: Il file contenente informazioni di base sugli utenti.

`/etc/group`
: Il file contenente informazioni di base sui gruppi.

`/etc/shadow`
: Il file contenente le password utente criptate.

`/etc/gshadow`
: Il file contenente le password di gruppo criptate.

`getent`
: Filtra i database delle password e dei gruppi.

Risposte agli Esercizi Guidati

1. Osservate il seguente output e rispondi alle seguenti domande:

```
# cat /etc/passwd | grep '\(root\|mail\|catherine\|kevin\)'
root:x:0:0:root:/root:/bin/bash
mail:x:8:8:mail:/var/spool/mail:/sbin/nologin
catherine:x:1030:1025:User Chaterine:/home/catherine:/bin/bash
kevin:x:1040:1015:User Kevin:/home/kevin:/bin/bash
# cat /etc/group | grep '\(root\|mail\|db-admin\|app-developer\)'
root:x:0:
mail:x:8:
db-admin:x:1015:emma,grace
app-developer:x:1016:catherine,dave,christian
# cat /etc/shadow | grep '\(root\|mail\|catherine\|kevin\)'
root:$6$1u36Ipok$ljt8ooPMLewAhkQPf.lYgGopAB.jClTO6ljsdczxvkLPkpi/amgp.zyfAN680zrLLp2avvpd
KA0llpssdfcPppOp:18015:0:99999:7:::
mail:*:18015:0:99999:7:::
catherine:$6$ABCD25jlld14hpPthEFGnnssEWw1234yioMpliABCdef1f3478kAfhhAfgbAMjY1/BAeeAsl/FeE
dddKd12345g6kPACcik:18015:20:90:5:::
kevin:$6$DEFGabc123WrLp223fsvp0ddx3dbA7pPPc4LMaa123u6Lp02Lpvm123456pyphhh5ps012vbArL245.P
R1345kkA3Gas12P:18015:0:60:7:2::
# cat /etc/gshadow | grep '\(root\|mail\|db-admin\|app-developer\)'
root:*::
mail:*::
db-admin:!:emma:emma,grace
app-developer:!::catherine,dave,christian
```

   - Qual è l'ID utente (UID) e l'ID gruppo (GID) di `root` e `catherine`?

     L'UID e il GID di `root` sono 0 e 0, mentre l'UID e il GID di `catherine` sono 1030 e 1025.

   - Qual è il nome del gruppo principale di `kevin`? Ci sono altri membri in questo gruppo?

     Il nome del gruppo è `db-admin`. Anche `emma` e `grace` sono in questo gruppo.

   - Quale shell è impostata per `mail`? Che cosa significa?

     `mail` è un account utente di sistema e la sua shell è `/sbin/nologin`. Infatti, gli account utente di sistema come `mail`, `ftp`, `news` e `daemon` sono usati per eseguire compiti amministrativi e quindi il normale login dovrebbe essere impedito per questi account. Questo è il motivo per cui la shell è solitamente impostata su `/sbin/nologin` o

`/bin/false`.

- Chi sono i membri del gruppo `app-developer`? Quali tra questi membri sono amministratori del gruppo e quali sono membri ordinari?

  I membri sono `catherine`, `dave` e `christian` e sono tutti membri ordinari.

- Qual è la durata minima della password per `catherine`? E qual è la durata massima della password?

  La durata minima della password è di 20 giorni, mentre quella massima è di 90 giorni.

- Qual è il periodo di inattività della password per `kevin`?

  Il periodo di inattività della password è di 2 giorni. Durante questo periodo `kevin` deve aggiornare la password, altrimenti l'account verrà disabilitato.

2. Per convenzione, quali ID sono assegnati agli account di sistema e quali agli utenti ordinari?

   Gli account di sistema di solito hanno UID inferiori a 100 o tra 500 e 1000, mentre gli utenti ordinari hanno UID a partire da 1000, anche se alcuni sistemi *legacy* possono iniziare la numerazione da 500. L'utente `root` ha UID 0. Ricorda che i valori `UID_MIN` e `UID_MAX` in `/etc/login.defs` definiscono la gamma di UID usati per la creazione degli utenti ordinari. Dal punto di vista di LPI Linux Essentials e LPIC-1, gli account di sistema hanno UID inferiori a 1000 e gli utenti ordinari hanno UID superiori a 1000.

3. Come si fa a scoprire se un account utente, che prima era in grado di accedere al sistema, ora è bloccato? Supponiamo che il vostro sistema utilizzi le *shadow* password.

   Quando si usano le password *shadow*, il secondo campo in `/etc/passwd` contiene il carattere `x` per ogni account utente, perché le password utente criptate sono memorizzate in `/etc/shadow`. In particolare, la password criptata di un account utente è memorizzata nel secondo campo di questo file e, se inizia con un punto esclamativo, l'account è bloccato.

Risposte agli Esercizi Esplorativi

1. Crea un account utente chiamato `christian` usando il comando `useradd -m` e identifica il suo User ID (UID), Group ID (GID) e la shell.

   ```
   # useradd -m christian
   # cat /etc/passwd | grep christian
   christian:x:1050:1060::/home/christian:/bin/bash
   ```

   L'UID e il GID di `christian` sono rispettivamente 1050 e 1060 (il terzo e il quarto campo in `/etc/passwd`). `/bin/bash` è la shell impostata per questo account utente (il settimo campo in `/etc/passwd`).

2. Identifica il nome del gruppo primario di `christian`. Che cosa puoi dedurre?

   ```
   # cat /etc/group | grep 1060
   christian:x:1060:
   ```

   Il nome del gruppo primario di `christian` è `christian` (il primo campo in `/etc/group`). Pertanto `USERGROUPS_ENAB` in `/etc/login.defs` è impostato su *yes* in modo che `useradd` crei di default un gruppo con lo stesso nome dell'account utente.

3. Usando il comando `getent`, esamina le informazioni sulla scadenza della password per l'account utente `christian`.

   ```
   # getent shadow christian
   christian:!:18015:0:99999:7:::
   ```

   L'account utente `christian` non ha la password impostata ed è ora bloccato (il secondo campo in `/etc/shadow` contiene un punto esclamativo). Non c'è un'età minima e massima della password per questo account utente (il quarto e il quinto campo in `/etc/shadow` sono impostati a 0 e 99999 giorni), mentre il periodo di avviso della password è impostato a 7 giorni (il sesto campo in `/etc/shadow`). Infine, non c'è un periodo di inattività (il settimo campo in `/etc/shadow`) e l'account non scade mai (l'ottavo campo in `/etc/shadow`).

4. Aggiungi il gruppo `editor` ai gruppi secondari di `christian`. Assumi che questo gruppo contenga già `emma`, `dave` e `frank` come membri ordinari. Come puoi verificare che non ci siano amministratori per questo gruppo?

```
# cat /etc/group | grep editor
editor:x:1100:emma,dave,frank
# usermod -a -G editor christian
# cat /etc/group | grep editor
editor:x:1100:emma,dave,frank,christian
# cat /etc/gshadow | grep editor
editor:!::emma,dave,frank,christian
```

Il terzo e il quarto campo in `/etc/ghadow` contengono amministratori e membri ordinari per il gruppo specificato. Quindi, poiché il terzo campo è vuoto per `editor`, non ci sono amministratori per questo gruppo (`emma`, `dave`, `frank` e `christian` sono tutti membri ordinari).

5. Esegui il comando `ls -l /etc/passwd /etc/group /etc/shadow /etc/gshadow` e descrivi l'output che ti dà in termini di permessi dei file. Quali di questi quattro file sono "mascherati" per ragioni di sicurezza? Supponiamo che il tuo sistema usi le password *shadow*.

```
# ls -l /etc/passwd /etc/group /etc/shadow /etc/gshadow
-rw-r--r-- 1 root root    853 mag  1 08:00 /etc/group
-rw-r----- 1 root shadow 1203 mag  1 08:00 /etc/gshadow
-rw-r--r-- 1 root root   1354 mag  1 08:00 /etc/passwd
-rw-r----- 1 root shadow 1563 mag  1 08:00 /etc/shadow
```

I file `/etc/passwd` e `/etc/group` sono leggibili da tutti e sono "mascherati" per ragioni di sicurezza. Quando si usano le password *shadow*, si può vedere una `x` nel secondo campo di questi file, perché le password criptate per utenti e gruppi sono memorizzate in `/etc/shadow` e `/etc/gshadow`, che sono leggibili solo da root e, su alcuni sistemi, anche dai membri appartenenti al gruppo `shadow`.

107.2 Automatizzare le attività di amministrazione del sistema attraverso la pianificazione

Obiettivi LPI di riferimento

LPIC-1 version 5.0, Exam 102, Objective 107.2

Peso

4

Aree di Conoscenza Chiave

- Gestire pianificazioni con cron e at.
- Configurare l'accesso utente ai servizi di cron e at.
- Comprendere le timer unit di systemd.

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

- `/etc/cron.{d,daily,hourly,monthly,weekly}/`
- `/etc/at.deny`
- `/etc/at.allow`
- `/etc/crontab`
- `/etc/cron.allow`
- `/etc/cron.deny`
- `/var/spool/cron/`
- `crontab`
- `at`
- `atq`

- `atrm`
- `systemctl`
- `systemd-run`



107.2 Lezione 1

Certificazione:	LPIC-1
Versione:	5.0
Argomento:	107 Attività Amministrative
Obiettivo:	107.2 Automatizzare le attività di amministrazione del sistema attraverso la pianificazione
Lezione:	1 di 2

Introduzione

Uno dei compiti più importanti di un buon amministratore di sistema è quello di programmare i *job* che devono essere eseguiti regolarmente. Per esempio, un amministratore può creare e automatizzare job per i backup, gli aggiornamenti del sistema e l'esecuzione di molte altre attività ripetitive. Per fare questo si può usare la funzione `cron`, che è utile per automatizzare la programmazione di job periodici.

Pianificare i Lavori con Cron

In Linux, `cron` è un demone che è in esecuzione continua e ogni minuto controlla un insieme di tabelle per trovare compiti da eseguire. Queste tabelle sono note come *crontabs* e contengono i cosiddetti *cron jobs*. Cron è adatto a server e sistemi che sono costantemente accesi, perché ogni lavoro cron viene eseguito solo se il sistema è in funzione all'ora prevista. Può essere usato dagli utenti ordinari, ognuno dei quali ha il proprio `crontab`, così come l'utente root che gestisce i crontab di sistema.

NOTE

In Linux c'è anche la funzione `anacron`, che è adatta a sistemi che possono essere spenti (come desktop o laptop). Può essere usata solo da root. Se la macchina è spenta quando i lavori `anacron` devono essere eseguiti, saranno eseguiti la prossima volta che la macchina sarà accesa. `anacron` è fuori dallo scopo della certificazione LPIC-1.

Crontab Utente

I crontab utente sono file di testo che gestiscono la programmazione di lavori cron definiti dall'utente. Hanno sempre il nome dell'account utente che li ha creati, ma la posizione di questi file dipende dalla distribuzione utilizzata (generalmente una sottodirectory di `/var/spool/cron`).

Ogni linea in un crontab utente contiene sei campi separati da uno spazio:

- Il minuto dell'ora (0-59).
- L'ora del giorno (0-23).
- Il giorno del mese (1-31).
- Il mese dell'anno (1-12).
- Il giorno della settimana (0-7 con domenica=0 o domenica=7).
- Il comando da eseguire.

Per il mese dell'anno e il giorno della settimana puoi usare le prime tre lettere del nome invece del numero corrispondente.

I primi cinque campi indicano quando eseguire il comando specificato nel sesto campo e possono contenere uno o più valori. In particolare, è possibile specificare più valori utilizzando:

`*` (asterisco)
: Si riferisce a qualsiasi valore.

`,` (virgola)
: Specifica una lista di valori possibili.

`-` (trattino)
: Specifica un intervallo di valori possibili.

`/` (slash)
: Specifica valori spezzati.

Molte distribuzioni includono il file `/etc/crontab` che può essere usato come riferimento per il layout di un file `cron`. Ecco un esempio di file `/etc/crontab` da un'installazione Debian:

```
SHELL=/bin/sh
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin

# Example of job definition:
# .---------------- minute (0 - 59)
# |  .------------- hour (0 - 23)
# |  |  .---------- day of month (1 - 31)
# |  |  |  .------- month (1 - 12) OR jan,feb,mar,apr ...
# |  |  |  |  .---- day of week (0 - 6) (Sunday=0 or 7) OR sun,mon,tue,wed,thu,fri,sat
# |  |  |  |  |
# *  *  *  *  * user-name command to be executed
```

Crontab di Sistema

I crontab di sistema sono file di testo che gestiscono la programmazione dei cron job di sistema e possono essere modificati solo dall'utente root. `/etc/crontab` e tutti i file nella directory `/etc/cron.d` sono crontab di sistema.

Molte distribuzioni includono anche le directory `/etc/cron.hourly`, `/etc/cron.daily`, `/etc/cron.weekly` e `/etc/cron.monthly` che contengono script da eseguire con la frequenza appropriata. Per esempio, se vuoi eseguire uno script ogni giorno puoi metterlo in `/etc/cron.daily`.

WARNING

Alcune distribuzioni usano `/etc/cron.d/hourly`, `/etc/cron.d/daily`, `/etc/cron.d/weekly` e `/etc/cron.d/monthly`. Ricordati sempre di controllare le directory corrette in cui inserire gli script che vorresti far eseguire da cron.

La sintassi dei crontab di sistema è simile a quella dei crontab utente, tuttavia richiede anche un ulteriore campo obbligatorio che specifica quale utente eseguirà il *cron job*. Pertanto, ogni linea in un crontab di sistema contiene sette campi separati da uno spazio:

- Il minuto dell'ora (0-59).
- L'ora del giorno (0-23).
- Il giorno del mese (1-31).
- Il mese dell'anno (1-12).
- Il giorno della settimana (0-7 con domenica=0 o domenica=7).

- Il nome dell'account utente che esegue il comando.
- Il comando da eseguire.

Come per i crontab utente, è possibile specificare valori multipli per i campi dell'ora usando gli operatori `*`, `,` , `-` e `/`. È possibile anche indicare il mese dell'anno e il giorno della settimana con le prime tre lettere del nome invece del numero corrispondente.

Particolari Specifiche di Tempo

Quando si modificano i file crontab, si possono anche usare scorciatoie speciali nelle prime cinque colonne al posto delle specifiche di tempo:

`@reboot`
: Esegue il compito specificato una volta dopo il riavvio.

`@hourly`
: Esegue il task specificato una volta all'ora all'inizio dell'ora.

`@daily` (o `@midnight`)
: Esegue il task specificato una volta al giorno a mezzanotte.

`@weekly`
: Esegue il task specificato una volta alla settimana alla mezzanotte della domenica.

`@monthly`
: Esegue il task specificato una volta al mese alla mezzanotte del primo giorno del mese.

`@yearly` (o `@annually`)
: Esegue il task specificato una volta all'anno alla mezzanotte del 1° Gennaio.

Variabili Crontab

All'interno di un file crontab, ci sono talvolta assegnazioni di variabili definite prima che i compiti programmati siano dichiarati. Le variabili d'ambiente comunemente impostate sono:

`HOME`
: La directory dove `cron` invoca i comandi (di default la home directory dell'utente).

`MAILTO`
: Il nome dell'utente o l'indirizzo a cui lo standard output ed error sono inviati per posta (di

default il proprietario di crontab). Sono ammessi anche valori multipli separati da virgola e un valore vuoto indica che non deve essere inviata alcun messaggio.

`PATH`

Il percorso dove i comandi possono essere trovati.

`SHELL`

La shell da usare (di default `/bin/sh`).

Creare Attività di Cron Utente

Il comando `crontab` è usato per mantenere i file crontab per i singoli utenti. In particolare, è possibile eseguire il comando `crontab -e` per modificare il proprio file crontab o per crearne uno se non esiste già.

```
$ crontab -e
no crontab for frank - using an empty one

Select an editor.  To change later, run 'select-editor'.
  1. /bin/ed
  2. /bin/nano        < ---- easiest
  3. /usr/bin/emacs24
  4. /usr/bin/vim.tiny

Choose 1-4 [2]:
```

Per impostazione predefinita, il comando `crontab` apre l'editor specificato dalle variabili d'ambiente `VISUAL` o `EDITOR`, così puoi iniziare a modificare il tuo file crontab con l'editor preferito. Alcune distribuzioni, come mostrato nell'esempio sopra, ti permettono di scegliere l'editor da una lista quando `crontab` viene eseguito per la prima volta.

Se vuoi eseguire lo script `foo.sh` situato nella tua directory home ogni giorno alle 10:00, puoi aggiungere la seguente linea al tuo file crontab:

```
0 10 * * * /home/frank/foo.sh
```

Considerate le seguenti voci di crontab di esempio:

```
0,15,30,45 08 * * 2 /home/frank/bar.sh
```

```
30 20 1-15 1,6 1-5 /home/frank/foobar.sh
```

Nella prima linea lo script `bar.sh` viene eseguito ogni martedì alle 08:00, alle 08:15, alle 08:30 e alle 08:45. Nella seconda linea lo script `foobar.sh` viene eseguito alle 08:30 da lunedì a venerdì per i primi quindici giorni di gennaio e giugno.

WARNING Anche se i file crontab possono essere modificati manualmente, si raccomanda sempre di usare il comando `crontab`. I permessi sui file crontab di solito permettono modifiche solo tramite il comando `crontab`.

Oltre all'opzione `-e` menzionata sopra, il comando `crontab` ha altre opzioni utili:

`-l`
: Visualizza il crontab corrente sullo standard output.

`-r`
: Rimuove il crontab corrente.

`-u`
: Specifica il nome dell'utente il cui crontab deve essere modificato. Questa opzione richiede i privilegi di root e permette all'utente root di modificare i file crontab degli utenti.

Creare Attività di Cron di Sistema

A differenza dei crontab degli utenti, i crontab di sistema sono aggiornati usando un editor: quindi non è necessario eseguire il comando `crontab` per modificare `/etc/crontab` e i file in `/etc/cron.d`. Ricorda che, quando modifichi i crontab di sistema, devi specificare l'account che sarà usato per eseguire il cron job (di solito l'utente root).

Per esempio: se vuoi eseguire lo script `barfoo.sh` situato nella directory `/root` ogni giorno alle 01:30, puoi aprire `/etc/crontab` con il tuo editor preferito e aggiungere la seguente linea:

```
30 01 * * * root /root/barfoo.sh >>/root/output.log 2>>/root/error.log
```

Nell'esempio precedente, lo *standard output* del lavoro viene aggiunto a `/root/output.log`, mentre gli errori vengono aggiunti a `/root/error.log`.

WARNING A meno che l'output sia reindirizzato a un file come nell'esempio precedente (o la variabile `MAILTO` sia impostata su un valore vuoto), tutto l'output di un *cron job* sarà inviato all'utente via e-mail. Una pratica comune è quella di

> reindirizzare l'output standard a `/dev/null` (o a un file per una revisione successiva, se necessario) e di non reindirizzare lo *standard error*. In questo modo l'utente sarà avvisato immediatamente via e-mail di qualsiasi errore.

Configurare l'Accesso alla Pianificazione delle Attività

In Linux i file `/etc/cron.allow` e `/etc/cron.deny` sono usati per impostare le restrizioni di `crontab`. In particolare, sono usati per permettere o meno la pianificazione delle attività `cron` agli utenti. Se `/etc/cron.allow` esiste, solo gli utenti non root elencati al suo interno possono creare *cron job* usando il comando `crontab`. Se `/etc/cron.allow` non esiste ma esiste `/etc/cron.deny`, solo gli utenti non-root elencati in questo file non possono creare *cron job* usando il comando `crontab` (in questo caso un `/etc/cron.deny` vuoto significa che ogni utente è autorizzato). Se nessuno di questi file esiste, l'accesso dell'utente alla pianificazione delle attività di cron dipende dalla distribuzione usata.

NOTE I file `/etc/cron.allow` e `/etc/cron.deny` contengono una lista di nomi utente, ognuno su una linea separata.

Una Alternativa a Cron

Usando *systemd* come gestore di sistemi e servizi, puoi impostare *timers* come alternativa a `cron` per pianificare delle attività. I timer sono file di unità di systemd identificati dal suffisso `.timer`, e per ognuno di questi deve esserci un corrispondente file di unità che descrive l'unità da attivare allo scadere del timer. Per impostazione predefinita, un `timer` attiva un servizio con lo stesso nome, tranne che per il suffisso.

Un *timer* include una sezione `[Timer]` che specifica quando le attività programmate dovrebbero essere eseguite. In particolare, si può usare l'opzione `OnCalendar=` per definire dei *temporizzatori in tempo reale* che funzionano allo stesso modo dei cron job (sono basati su espressioni di eventi del calendario). L'opzione `OnCalendar=` richiede la seguente sintassi:

```
DayOfWeek Year-Month-Day Hour:Minute:Second
```

con `DayOfWeek` opzionale. Gli operatori `*`, `/` e `,` hanno lo stesso significato di quelli usati per i *cron job*, mentre puoi usare `..` tra due valori per indicare un intervallo contiguo. Per la specifiche di `DayOfWeek`, puoi usare le prime tre lettere del nome o il nome completo.

NOTE È possibile anche definire dei *temporizzatori monotonici* che si attivano dopo che è trascorso un certo tempo da un punto di partenza specifico (per esempio, quando la macchina è stata avviata o quando il timer stesso viene attivato).

Per esempio, se vuoi eseguire il servizio chiamato `/etc/systemd/system/foobar.service` alle 05:30 del primo lunedì di ogni mese, puoi aggiungere le seguenti linee nel corrispondente file `/etc/systemd/system/foobar.timer`.

```
[Unit]
Description=Run the foobar service

[Timer]
OnCalendar=Mon *-*-1..7 05:30:00
Persistent=true

[Install]
WantedBy=timers.target
```

Una volta creato il nuovo timer, puoi abilitarlo e avviarlo eseguendo i seguenti comandi come root:

```
# systemctl enable foobar.timer
# systemctl start foobar.timer
```

Puoi cambiare la frequenza del tuo lavoro programmato, modificando il valore `OnCalendar` e poi digitando il comando `systemctl daemon-reload`.

Infine, se vuoi vedere la lista dei timer attivi ordinati in base al tempo, puoi usare il comando `systemctl list-timers`. Puoi aggiungere l'opzione `--all` per vedere anche i timer inattivi.

NOTE Ricorda che i timer sono registrati nel *journal* di systemd e puoi rivedere i log delle diverse unità usando il comando `journalctl`. Ricorda anche che se stai agendo come utente ordinario, devi usare l'opzione `--user` dei comandi `systemctl` e `journalctl`.

Invece della forma normalizzata più lunga di cui sopra, si possono usare alcune espressioni speciali che descrivono particolari frequenze di esecuzione dell'attività:

`hourly`
: Esegue il compito specificato una volta all'ora all'inizio dell'ora.

`daily`
: Esegue il task specificato una volta al giorno a mezzanotte.

`weekly`

Esegue l'attività specificata una volta alla settimana alla mezzanotte del lunedì.

`monthly`

Esegue l'attività specificata una volta al mese alla mezzanotte del primo giorno del mese.

`yearly`

Esegue l'attività specificata una volta all'anno alla mezzanotte del primo giorno di gennaio.

Puoi vedere le pagine del manuale per la lista completa delle specifiche di ora e data in `systemd.timer(5)`.

Esercizi Guidati

1. Per ciascuna delle seguenti scorciatoie `crontab`, indicare la specifica temporale corrispondente (cioè le prime cinque colonne in un file `crontab` utente):

`@hourly`	
`@daily`	
`@weekly`	
`@monthly`	
`@annually`	

2. Per ciascuna delle seguenti scorciatoie `OnCalendar`, indicare la specifica temporale corrispondente (la forma normalizzata più lunga):

`hourly`	
`daily`	
`weekly`	
`monthly`	
`yearly`	

3. Spiega il significato delle seguenti specifiche temporali che si trovano in un file `crontab`:

`30 13 * * 1-5`	
`00 09-18 * * *`	
`30 08 1 1 *`	
`0,20,40 11 * * Sun`	
`00 09 10-20 1-3 *`	
`*/20 * * * *`	

4. Spiega il significato delle seguenti specifiche temporali usate nell'opzione `OnCalendar` di un file *timer*:

`*-*-* 08:30:00`	
`Sat,Sun *-*-* 05:00:00`	

`*-*-01 13:15,30,45:00`	
`Fri *-09..12-* 16:20:00`	
`Mon,Tue *-*-1,15 08:30:00`	
`*-*-* *:00/05:00`	

Esercizi Esplorativi

1. Supponendo che tu sia autorizzato a pianificate attività con `cron` come utente ordinario, quale comando useresti per creare il tuo file crontab?

2. Crea un semplice lavoro pianificato che esegua il comando `date` ogni venerdì alle 01:00 pm. Dove puoi vedere l'output di questo lavoro?

3. Crea un altro lavoro pianificato che esegua lo script `foobar.sh` ogni minuto, reindirizzando l'output al file `output.log` nella tua home directory in modo che solo lo *standard error* ti venga inviato per e-mail.

4. Guarda la voce `crontab` dell'attività pianificata appena creata. Perché non è necessario specificare il percorso assoluto del file in cui viene salvato lo *standard output* ? E perché puoi usare il comando `./foobar.sh` per eseguire lo script?

5. Modifica la voce precedente di `crontab` rimuovendo il reindirizzamento dell'output e disabilita il primo cron job che hai creato.

6. Come puoi inviare lo *standard output* ed *error* del tuo lavoro pianificato all'account utente `emma` via e-mail? E come si può invece evitarlo?

7. Esegui il comando `ls -l /usr/bin/crontab`. Quale bit speciale è impostato e qual è il suo significato?

Sommario

In questa lezione abbiamo imparato:

- Usare `cron` per eseguire attività a intervalli regolari.
- Gestire i lavori cron.
- Configurare l'accesso degli utenti alla pianificazione dei lavori cron.
- Comprendere il ruolo dei *timer* di systemd come alternativa a `cron`.

I seguenti comandi e file sono stati discussi in questa lezione:

`crontab`

Mantiene i file `crontab` per i singoli utenti.

`/etc/cron.allow` e **`/etc/cron.deny`**

File particolari usati per impostare le restrizioni di `crontab`.

`/etc/crontab`

File crontab di sistema.

`/etc/cron.d`

La directory che contiene i file crontab di sistema.

`systemctl`

Controlla systemd e il gestore dei servizi. In relazione ai timer, può essere usato per abilitarli e avviarli.

Risposte agli Esercizi Guidati

1. Per ciascuna delle seguenti scorciatoie `crontab`, indicare la specifica temporale corrispondente (cioè le prime cinque colonne in un file `crontab` utente):

`@hourly`	`0 * * * *`
`@daily`	`0 0 * * *`
`@weekly`	`0 0 * * 0`
`@monthly`	`0 0 1 * *`
`@annually`	`0 0 1 1 *`

2. Per ciascuna delle seguenti scorciatoie `OnCalendar`, indicare la specifica temporale corrispondente (la forma normalizzata più lunga):

`hourly`	`*-*-* *:00:00`
`daily`	`*-*-* 00:00:00`
`weekly`	`Mon *-*-* 00:00:00`
`monthly`	`*-*-01 00:00:00`
`yearly`	`*-01-01 00:00:00`

3. Spiega il significato delle seguenti specifiche temporali per un file `crontab`:

`30 13 * * 1-5`	Alle 01:30 pm tutti i giorni della settimana da lunedì a venerdì
`00 09-18 * * *`	Ogni giorno e ogni ora dalle 09.00 am alle 06.00 pm
`30 08 1 1 *`	Alle 08:30 am del primo giorno di gennaio
`0,20,40 11 * * Sun`	Ogni domenica alle 11:00, 11:20 e 11:40 am
`00 09 10-20 1-3 *`	Alle 09:00 am dal 10 al 20 gennaio, febbraio e marzo
`*/20 * * * *`	Ogni 20 minuti

4. Spiega il significato delle seguenti specifiche temporali usate nell'opzione `OnCalendar` di un file *timer*:

`*-*-* 08:30:00`	Ogni giorno alle 08:30 am
`Sat,Sun *-*-* 05:00:00`	Alle 05:00 am il sabato e la domenica
`*-*-01 13:15,30,45:00`	Alle 01:15 pm, 01:30 pm e 01:45 pm il primo giorno del mese
`Fri *-09..12-* 16:20:00`	Alle 04:20 pm tutti i venerdì a settembre, ottobre, novembre e dicembre
`Mon,Tue *-*-1,15 08:30:00`	Alle 08.30 am il primo e il quindicesimo giorno di ogni mese solo se il giorno è un lunedì o un martedì
`*-*-* *:00/05:00`	Ogni 5 minuti

Risposte agli Esercizi Esplorativi

1. Supponendo che tu sia autorizzato a pianificate attività con `cron` come utente ordinario, quale comando useresti per creare il tuo file crontab?

```
dave@hostname ~ $ crontab -e
no crontab for dave - using an empty one

Select an editor.  To change later, run 'select-editor'.
  1. /bin/ed
  2. /bin/nano        < ---- easiest
  3. /usr/bin/emacs24
  4. /usr/bin/vim.tiny

Choose 1-4 [2]:
```

2. Crea un semplice lavoro pianificato che esegua il comando `date` ogni venerdì alle 01:00 pm. Dove puoi vedere l'output di questo lavoro?

```
00 13 * * 5 date
```

   L'output viene inviato per posta all'utente; per visualizzarlo, puoi usare il comando `mail`.

3. Crea un altro lavoro pianificato che esegua lo script `foobar.sh` ogni minuto, reindirizzando l'output al file `output.log` nella tua home directory in modo che solo lo *standard error* ti venga inviato per e-mail.

```
*/1 * * * * ./foobar.sh >> output.log
```

4. Guarda la voce `crontab` dell'attività pianificata appena creata. Perché non è necessario specificare il percorso assoluto del file in cui viene salvato lo *standard output* ? E perché puoi usare il comando `./foobar.sh` per eseguire lo script?

   `cron` invoca i comandi dalla home directory dell'utente, a meno che un'altra posizione sia specificata dalla variabile d'ambiente `HOME` all'interno del file `crontab`. Per questo motivo, puoi usare il percorso relativo del file di output ed eseguire lo script con `./foobar.sh`.

5. Modifica la voce precedente di `crontab` rimuovendo il reindirizzamento dell'output e disabilita il primo cron job che hai creato.

```
#00 13 * * 5 date
*/1 * * * * ./foobar.sh
```

Per disabilitare un *cron job*, puoi semplicemente commentare la linea corrispondente nel file `crontab`.

6. Come puoi inviare lo *standard output* ed *error* del tuo lavoro pianificato all'account utente `emma` via e-mail? E come si può invece evitarlo?

   Per inviare lo standard output ed error, devi impostare la variabile d'ambiente `MAILTO` nel tuo file `crontab` come segue:

   ```
   MAILTO="emma"
   ```

   Per dire a `cron` che non deve essere inviata alcuna mail, puoi assegnare un valore vuoto alla variabile d'ambiente `MAILTO`.

   ```
   MAILTO=""
   ```

7. Esegui il comando `ls -l /usr/bin/crontab`. Quale bit speciale è impostato e qual è il suo significato?

   ```
   $ ls -l /usr/bin/crontab
   -rwxr-sr-x 1 root crontab 25104 feb 10  2015 /usr/bin/crontab
   ```

   Il comando `crontab` ha il bit SGID impostato (il carattere `s` al posto del flag eseguibile per il gruppo), il che significa che viene eseguito con i privilegi del gruppo (quindi `crontab`). Questo è il motivo per cui gli utenti ordinari possono modificare il loro file `crontab` usando il comando `crontab`. Si noti che molte distribuzioni hanno i permessi sui file impostati in modo tale che i file `crontab` possano essere modificati solo tramite il comando `crontab`.



107.2 Lezione 2

Certificazione:	LPIC-1
Versione:	5.0
Argomento:	107 Attività Amministrative
Obiettivo:	107.2 Automatizzare le attività di amministrazione del sistema attraverso la pianificazione
Lezione:	2 di 2

Introduzione

Come hai imparato nella lezione precedente, puoi pianificare dei lavori regolari usando `cron` o i *timer* di systemd; a volte però potresti aver bisogno di eseguire un'attività solo per una volta in un momento specifico nel futuro. Per fare questo, puoi usare un'altra potente utility: il comando `at`.

Programmare attività con `at`

Il comando `at` è usato per la pianificazione di un'attività *una tantum* e richiede solo che si specifichi *quando* il lavoro deve essere eseguito in futuro. Dopo aver inserito `at` sulla linea di comando seguito dalla specifica del tempo, si accede al prompt `at` dove si possono definire i comandi da eseguire. È possibile uscire dal prompt con la sequenza di tasti `Ctrl` + `D`.

```
$ at now +5 minutes
warning: commands will be executed using /bin/sh
at> date
at> Ctrl+D
```

```
job 12 at Sat Sep 14 09:15:00 2019
```

L'attività `at` nell'esempio precedente esegue semplicemente il comando `date` dopo cinque minuti. Simile a `cron`, lo *standard output* ed *error* vengono inviati via e-mail. Si noti che il demone `atd` dovrà essere in esecuzione sul sistema per poter utilizzare la pianificazione `at`.

NOTE In Linux, il comando `batch` è simile ad `at`, tuttavia le attività `batch` vengono eseguite solo quando il carico del sistema è sufficientemente basso da permetterlo.

Le opzioni più importanti che si applicano al comando `at` sono:

`-c`
Stampa i comandi di uno specifico ID lavoro allo standard output.

`-d`
Elimina i lavori in base al loro ID. È un alias di `atrm`.

`-f`
Legge il lavoro da un file invece che dallo standard input.

`-l`
Elenca i lavori in sospeso dell'utente. Se l'utente è root, vengono elencati tutti i lavori di tutti gli utenti. È un alias di `atq`.

`-m`
Invia una mail all'utente alla fine del lavoro anche se non c'è stato alcun output.

`-q`
Specifica una coda sotto forma di una singola lettera da `a` a `z` e da `A` a `Z` (di default `a` per `at` e `b` per `batch`). I lavori nelle code con le lettere più alte sono eseguiti con priorità più alta. I lavori inviati a una coda con una lettera maiuscola sono trattati come lavori `batch`.

`-v`
Mostra il tempo in cui il lavoro verrà eseguito prima della lettura del lavoro.

Mostrare i Lavori Pianificati con `atq`

Programmiamo altri due lavori `at`: il primo esegue lo script `foo.sh` alle 09:30, mentre il secondo esegue lo script `bar.sh` dopo un'ora.

```
$ at 09:30 AM
warning: commands will be executed using /bin/sh
at> ./foo.sh
at> Ctrl+D
job 13 at Sat Sep 14 09:30:00 2019
$ at now +2 hours
warning: commands will be executed using /bin/sh
at> ./bar.sh
at> Ctrl+D
job 14 at Sat Sep 14 11:10:00 2019
```

Per elencare i lavori in sospeso, puoi usare il comando `atq` che mostra le seguenti informazioni per ogni lavoro: ID del lavoro, data di esecuzione del lavoro, tempo di esecuzione del lavoro, coda e nome utente.

```
$ atq
14      Sat Sep  14 11:10:00 2019 a frank
13      Sat Sep  14 09:30:00 2019 a frank
12      Sat Sep  14 09:15:00 2019 a frank
```

Ricorda che il comando `at -l` è un alias di `atq`.

NOTE Se si esegue `atq` come root, mostrerà i lavori in coda per tutti gli utenti.

Rimuovere i Lavori Pianificati con `atrm`

Se vuoi cancellare un lavoro `at`, puoi usare il comando `atrm` seguito dall'ID del lavoro. Per esempio, per cancellare il lavoro con *ID 14*, puoi eseguire quanto segue:

```
$ atrm 14
```

Puoi cancellare più lavori con `atrm` specificando più ID separati da spazi. Ricorda che il comando `at -d` è un alias di `atrm`.

NOTE Se esegui `atrm` come root puoi cancellare i lavori di tutti gli utenti.

Configurare l'Accesso alla Pianificazione delle Attività

L'autorizzazione per gli utenti ordinari a pianificare lavori con `at` è determinata dai file `/etc/at.allow` e `/etc/at.deny`. Se `/etc/at.allow` esiste, solo gli utenti non root elencati in

esso possono pianificare lavori con `at`. Se `/etc/at.allow` non esiste ma esiste `/etc/at.deny`, solo gli utenti non-root elencati al suo interno non possono pianificare lavori con `at` (in questo caso un file `/etc/at.deny` vuoto significa che ogni utente è autorizzato a pianificare lavori con `at`). Se nessuno di questi file esiste, l'accesso dell'utente alla pianificazione dei lavori `at` dipende dalla distribuzione usata.

Specifiche di Tempo

Puoi specificare quando eseguire un particolare lavoro `at` usando il formato `HH:MM`, opzionalmente seguito da AM o PM nel caso del formato a 12 ore. Se l'ora specificata è già passata, il lavoro si attiverà il giorno successivo. Se vuoi programmare una data particolare in cui il lavoro venga eseguito, devi aggiungere le informazioni sulla data dopo l'ora usando una delle seguenti forme: `nome mese giorno del mese`, `nome mese giorno del mese anno`, `MMDDYY`, `MM/DD/YYY`, `DD.MM.YY` e `YYYY-MM-DD`).

Sono accettate anche le seguenti parole chiave: `midnight`, `noon`, `teatime` (4 pm) e `now` seguita da un segno più (`+`) e un periodo di tempo (minuti, ore, giorni e settimane). Infine, puoi dire a `at` di eseguire il lavoro oggi o domani aggiungendo al tempo le parole `today` o `tomorrow`. Per esempio, puoi usare `at 07:15 AM Jan 01` per eseguire un lavoro alle 07:15 del 01 Gennaio e `at now +5 minutes` per eseguire un lavoro tra cinque minuti. Puoi leggere il file `timespec` all'interno di `/usr/share` per maggiori informazioni sulla definizione esatta delle specifiche temporali.

Una Alternativa a `at`

Usando systemd come gestore del sistema e dei servizi, si possono anche programmare compiti una tantum con il comando `systemd-run`. È tipicamente usato per creare un'unità *timer* transitoria in modo che un comando venga eseguito in un momento specifico senza la necessità di creare un file di servizio. Per esempio, agendo come root, è possibile eseguire il comando `date` alle 11:30 AM del 2019/10/06 utilizzando quanto segue:

```
# systemd-run --on-calendar='2019-10-06 11:30' date
```

Se vuoi eseguire lo script `foo.sh`, situato nella tua directory di lavoro corrente dopo due minuti puoi usare:

```
# systemd-run --on-active="2m" ./foo.sh
```

Consulta le pagine del manuale per imparare tutti i possibili usi di `systemd-run` con `systemd-run(1)`.

NOTE

Ricorda che i timer sono registrati nel *journal* di systemd e che puoi rivedere i log delle diverse unità usando il comando `journalctl`. Ricorda anche che, se stai agendo come utente ordinario, devi usare l'opzione `--user` dei comandi `systemd-run` e `journalctl`.

Esercizi Guidati

1. Per ciascuna delle seguenti specifiche temporali, indicare quale è valida e quale non è valida per `at`:

`at 08:30 AM next week`	
`at midday`	
`at 01-01-2020 07:30 PM`	
`at 21:50 01.01.20`	
`at now +4 days`	
`at 10:15 PM 31/03/2021`	
`at tomorrow 08:30 AM`	

2. Una volta che hai programmato un lavoro con `at`, come puoi rivedere i suoi comandi?

3. Quali comandi puoi usare per rivedere i lavori `at` in sospeso? Quali comandi useresti per cancellarli?

4. Con systemd, quale comando è usato come alternativa a `at`?

Esercizi Esplorativi

1. Crea un lavoro `at` che esegua lo script `foo.sh`, situato nella tua home directory, alle 10:30 del 31 Ottobre prossimo. Supponiamo che tu stia agendo come un utente ordinario.

2. Accedi al sistema come un altro utente ordinario e crea un altro lavoro `at` che esegua lo script `bar.sh` domani alle 10:00. Supponiamo che lo script si trovi nella home directory dell'utente.

3. Accedi al sistema come un altro utente ordinario e crea un altro lavoro `at` che esegue lo script `foobar.sh` dopo 30 minuti. Supponiamo che lo script si trovi nella home directory dell'utente.

4. Ora, come root, esegui il comando `atq` per esaminare i lavori programmati `at` di tutti gli utenti. Che cosa succede se un utente normale esegue questo comando?

5. Come root, cancella tutti questi lavori in sospeso `at` con un solo comando.

6. Esegui il comando `ls -l /usr/bin/at` ed esamina i suoi permessi.

Sommario

In questa lezione abbiamo imparato:

- Usare `at` per eseguire lavori una tantum in un momento specifico.
- Gestire i lavori `at`.
- Configurare l'accesso degli utenti alla programmazione dei lavori `at`.
- Usare `systemd-run` come alternativa a `at`.

I seguenti comandi e file sono stati discussi in questa lezione:

`at`
: Esegue i comandi in un momento specifico.

`atq`
: Elenca i lavori `at` in sospeso dell'utente, a meno che l'utente non sia il superutente.

`atrm`
: Elimina i lavori `at`, identificati dal loro numero di lavoro.

`/etc/at.allow` and `/etc/at.deny`
: File particolari usati per impostare le restrizioni `at`.

`systemd-run`
: Crea e avvia un'unità transitoria `timer` come alternativa a `at` per la programmazione una tantum.

Risposte agli Esercizi Guidati

1. Per ciascuna delle seguenti specifiche temporali, indicare quale è valida e quale non è valida per `at`:

`at 08:30 AM next week`	Valida
`at midday`	Non Valida
`at 01-01-2020 07:30 PM`	Non Valida
`at 21:50 01.01.20`	Valida
`at now +4 days`	Valida
`at 10:15 PM 31/03/2021`	Non Valida
`at tomorrow 08:30 AM monotonic`	Non Valida

2. Una volta che hai programmato un lavoro con `at`, come puoi rivedere i suoi comandi?

   Puoi usare il comando `at -c` seguito dall'ID del lavoro di cui vuoi esaminare i comandi. Nota che l'output contiene anche la maggior parte dell'ambiente che era attivo nel momento in cui il lavoro è stato pianificato. Ricorda che root può esaminare i lavori di tutti gli utenti.

3. Quali comandi puoi usare per rivedere i lavori `at` in sospeso? Quali comandi useresti per cancellarli?

   Puoi usare il comando `at -l` per rivedere i tuoi lavori in sospeso, e puoi usare il comando `at -d` per cancellare i tuoi lavori. `at -l` è un alias per `atq` e `at -d` è un alias per `atrm`. Ricorda che root può elencare e cancellare i lavori di tutti gli utenti.

4. Con systemd, quale comando è usato come alternativa a `at`?

   Il comando `systemd-run` può essere usato come alternativa a `at` per programmare lavori una tantum. Per esempio, puoi usarlo per eseguire comandi in un momento specifico, definendo un *calendar timer* o un *monotonic timer* relativo a diversi punti di partenza.

Risposte agli Esercizi Esplorativi

1. Crea un lavoro `at` che esegua lo script `foo.sh`, situato nella tua home directory, alle 10:30 del 31 Ottobre prossimo. Supponiamo che tu stia agendo come un utente ordinario.

```
$ at 10:30 AM October 31
warning: commands will be executed using /bin/sh
at> ./foo.sh
at> Ctrl+D
job 50 at Thu Oct 31 10:30:00 2019
```

2. Accedi al sistema come un altro utente ordinario e crea un altro lavoro `at` che esegue lo script `bar.sh` domani alle 10:00. Supponiamo che lo script si trovi nella home directory dell'utente.

```
$ at 10:00 AM tomorrow
warning: commands will be executed using /bin/sh
at> ./bar.sh
at> Ctrl+D
job 51 at Sun Oct 6 10:00:00 2019
```

3. Accedi al sistema come un altro utente ordinario e crea un altro lavoro `at` che esegue lo script `foobar.sh` dopo 30 minuti. Supponiamo che lo script si trovi nella home directory dell'utente.

```
$ at now +30 minutes
warning: commands will be executed using /bin/sh
at> ./foobar.sh
at> Ctrl+D
job 52 at Sat Oct 5 10:19:00 2019
```

4. Ora, come root, esegui il comando `atq` per esaminare i lavori programmati `at` di tutti gli utenti. Cosa succede se un utente normale esegue questo comando?

```
# atq
52      Sat Oct  5 10:19:00 2019 a dave
50      Thu Oct 31 10:30:00 2019 a frank
51      Sun Oct  6 10:00:00 2019 a emma
```

   Se si esegue il comando `atq` come root, vengono elencati tutti i lavori `at` pendenti di tutti gli utenti. Se lo esegui come utente ordinario, vengono elencati solo i tuoi lavori `at` in sospeso.

5. Come root, cancella tutti questi lavori in sospeso `at` con un solo comando.

```
# atrm 50 51 52
```

6. Come root, esegui il comando `ls -l /usr/bin/at` ed esamina i suoi permessi.

```
# ls -l /usr/bin/at
-rwsr-sr-x 1 daemon daemon 43762 Dec  1  2015 /usr/bin/at
```

   In questa distribuzione, il comando `at` ha entrambi i bit SUID (il carattere `s` invece del flag eseguibile per il proprietario) e SGID (il carattere `s` invece del flag eseguibile per il gruppo) impostati, il che significa che viene eseguito con i privilegi del proprietario e del gruppo del file (`daemon` per entrambi). Questo è il motivo per cui gli utenti ordinari sono in grado di pianificare lavori con `at`.



107.3 Localizzazione e internazionalizzazione

Obiettivi LPI di riferimento

LPIC-1 version 5.0, Exam 102, Objective 107.3

Peso

3

Aree di Conoscenza Chiave

- Configurare le impostazioni locali e le relative variabili di ambiente.
- Configurare le impostazioni del fuso orario e le relative variabili d'ambiente.

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

- `/etc/timezone`
- `/etc/localtime`
- `/usr/share/zoneinfo/`
- `LC_*`
- `LC_ALL`
- `LANG`
- `TZ`
- `/usr/bin/locale`
- `tzselect`
- `timedatectl`
- `date`
- `iconv`

- `UTF-8`
- `ISO-8859`
- ASCII
- Unicode

107.3 Lezione 1

Certificazione:	LPIC-1
Versione:	5.0
Argomento:	107 Attività Amministrative
Obiettivo:	107.3 Localizzazione e internazionalizzazione
Lezione:	1 di 1

Introduzione

Tutte le principali distribuzioni Linux possono essere configurate per utilizzare le impostazioni personalizzate di localizzazione. Queste impostazioni includono definizioni relative alla regione e alla lingua, quali fuso orario, lingua dell'interfaccia e codifica dei caratteri, impostazioni che possono essere modificate durante l'installazione del sistema operativo o in qualsiasi momento successivo.

Le applicazioni si basano su variabili d'ambiente, file di configurazione di sistema e comandi per decidere l'ora e la lingua appropriate da usare; quindi la maggior parte delle distribuzioni Linux condividono un modo standardizzato per regolare le impostazioni di ora e localizzazione. Queste regolazioni sono importanti non solo per migliorare l'esperienza dell'utente, ma anche per assicurare che la tempistica degli eventi di sistema — importante, per esempio, per segnalare problemi legati alla sicurezza — sia calcolata correttamente.

Per essere in grado di rappresentare qualsiasi testo scritto, indipendentemente dalla lingua parlata, i sistemi operativi moderni hanno bisogno di uno *standard di codifica dei caratteri* di riferimento, e i sistemi Linux non sono diversi. Poiché i computer sono in grado di trattare solo numeri, un carattere di testo non è altro che un numero associato ad un simbolo grafico.

Piattaforme di computer diverse possono associare valori numerici diversi allo stesso carattere, quindi uno standard comune di codifica dei caratteri è necessario per renderli compatibili. Un documento di testo creato in un sistema sarà leggibile in un altro sistema *solo se* entrambi concordano sul formato di codifica e su quale numero è associato a quale carattere, o almeno se sanno gestire la conversione tra i due standard.

La natura eterogenea delle impostazioni di localizzazione nei sistemi basati su Linux si traduce in piccole differenze tra le distribuzioni. Nonostante ciò, *tutte* le distribuzioni condividono gli stessi strumenti e concetti di base per impostare gli aspetti di internazionalizzazione di un sistema.

Fusi Orari

I fusi orari sono bande discrete approssimativamente proporzionali della superficie terrestre che coprono l'equivalente di un'ora, cioè regioni del mondo che vivono la stessa ora del giorno nello stesso momento. Poiché non esiste una sola longitudine che possa essere considerata l'inizio del giorno per tutto il mondo, i fusi orari sono relativi al *primo meridiano*, dove l'angolo di longitudine della Terra è definito come 0. L'ora al primo meridiano è chiamata *Tempo Universale Coordinato*, per convenzione abbreviato in UTC. Per ragioni pratiche, i fusi orari non seguono l'esatta distanza longitudinale dal punto di riferimento (il primo meridiano). Invece, i fusi orari sono adattati artificialmente per seguire i confini dei paesi o altre suddivisioni significative.

Le suddivisioni politiche sono così rilevanti che i fusi orari prendono il nome da qualche importante elemento geografico in quella particolare area, di solito basato sul nome di un grande paese o città all'interno della zona. Tuttavia, i fusi orari sono divisi in base al loro *offset* temporale rispetto all'UTC e questo offset può anche essere usato per indicare la zona in questione. Il fuso orario *GMT-5*, per esempio, indica una regione per la quale l'ora UTC è cinque ore avanti, cioè quella regione è 5 ore indietro rispetto all'UTC. Allo stesso modo, il fuso orario *GMT+3* indica una regione per la quale l'ora UTC è tre ore indietro. Il termine GMT — da *Greenwich Mean Time* — è usato come sinonimo di UTC.

Una macchina collegata può essere raggiunta da diverse parti del mondo, quindi è buona pratica impostare l'orologio hardware su UTC (il fuso orario GMT+0) e lasciare la scelta del fuso orario per ogni caso particolare. I servizi cloud, per esempio, sono comunemente configurati per utilizzare UTC, in quanto può aiutare a mitigare le incongruenze occasionali tra l'ora locale e quella dei client o di altri server. Al contrario, gli utenti che aprono una sessione remota sul server potrebbero voler usare il loro fuso orario locale. Quindi, sarà compito del sistema operativo impostare il fuso orario corretto a seconda del caso.

Oltre alla data e all'ora correnti, il comando `date` visualizzerà anche il fuso orario attualmente configurato:

```
$ date
Mon Oct 21 10:45:21 -03 2019
```

L'offset rispetto all'UTC è dato dal valore `-03`, che significa che l'ora visualizzata è tre ore meno dell'UTC. Pertanto, l'ora UTC è tre ore avanti, rendendo *GMT-3* il fuso orario corrispondente all'ora data. Il comando `timedatectl`, che è disponibile nelle distribuzioni che usano systemd, mostra maggiori dettagli sull'ora e la data del sistema:

```
$ timedatectl
                      Local time: Sat 2019-10-19 17:53:18 -03
                  Universal time: Sat 2019-10-19 20:53:18 UTC
                        RTC time: Sat 2019-10-19 20:53:18
                       Time zone: America/Sao_Paulo (-03, -0300)
       System clock synchronized: yes
systemd-timesyncd.service active: yes
                 RTC in local TZ: no
```

Come mostrato nella voce `Time zone`, sono accettati anche nomi di fusi orari basati su località — come `America/Sao_Paulo` --. Il fuso orario predefinito per il sistema è mantenuto nel file `/etc/timezone`, sia dal nome descrittivo completo della zona che dall'offset. I nomi generici dei fusi orari dati dall'offset UTC devono includere `Etc` come prima parte del nome. Quindi, per impostare il fuso orario di default su GMT+3, il nome del fuso orario deve essere `Etc/GMT+3`.

```
$ cat /etc/timezone
Etc/GMT+3
```

Anche se i nomi dei fusi orari basati sulle località non richiedono l'offset temporale per funzionare, non sono così semplici da scegliere. La stessa zona può avere più di un nome, il che può renderla difficile da ricordare. Per facilitare questo problema, il comando `tzselect` offre un metodo interattivo che guida l'utente verso la corretta definizione del fuso orario. Il comando `tzselect` dovrebbe essere disponibile di default in tutte le distribuzioni Linux, poiché è fornito dal pacchetto che contiene i necessari programmi di utilità relativi alla GNU C Library.

Il comando `tzselect` sarà utile, per esempio, per un utente che vuole identificare il fuso orario per "São Paulo City" in "Brazil". `tzselect` inizia chiedendo la macro regione della località desiderata:

```
$ tzselect
Please identify a location so that time zone rules can be set correctly.
```

```
Please select a continent, ocean, "coord", or "TZ".
 1) Africa
 2) Americas
 3) Antarctica
 4) Asia
 5) Atlantic Ocean
 6) Australia
 7) Europe
 8) Indian Ocean
 9) Pacific Ocean
10) coord - I want to use geographical coordinates.
11) TZ - I want to specify the time zone using the Posix TZ format.
#? 2
```

L'opzione `2` è per località (nord e sud) americane, non necessariamente nello stesso fuso orario. È anche possibile specificare il fuso orario con coordinate geografiche o con la notazione di offset, nota anche come *formato Posix TZ*. Il passo successivo è quello di scegliere il paese:

```
Please select a country whose clocks agree with yours.
 1) Anguilla              19) Dominican Republic    37) Peru
 2) Antigua & Barbuda     20) Ecuador               38) Puerto Rico
 3) Argentina             21) El Salvador           39) St Barthelemy
 4) Aruba                 22) French Guiana         40) St Kitts & Nevis
 5) Bahamas               23) Greenland             41) St Lucia
 6) Barbados              24) Grenada               42) St Maarten (Dutch)
 7) Belize                25) Guadeloupe            43) St Martin (French)
 8) Bolivia               26) Guatemala             44) St Pierre & Miquelon
 9) Brazil                27) Guyana                45) St Vincent
10) Canada                28) Haiti                 46) Suriname
11) Caribbean NL          29) Honduras              47) Trinidad & Tobago
12) Cayman Islands        30) Jamaica               48) Turks & Caicos Is
13) Chile                 31) Martinique            49) United States
14) Colombia              32) Mexico                50) Uruguay
15) Costa Rica            33) Montserrat            51) Venezuela
16) Cuba                  34) Nicaragua             52) Virgin Islands (UK)
17) Curaçao               35) Panama                53) Virgin Islands (US)
18) Dominica              36) Paraguay
#? 9
```

Il territorio del Brasile si estende su quattro fusi orari, quindi le sole informazioni sul paese non sono sufficienti per impostare il fuso orario. Nel prossimo passo `tzselect` richiederà all'utente di specificare la regione locale:

```
Please select one of the following time zone regions.
 1) Atlantic islands
 2) Pará (east); Amapá
 3) Brazil (northeast: MA, PI, CE, RN, PB)
 4) Pernambuco
 5) Tocantins
 6) Alagoas, Sergipe
 7) Bahia
 8) Brazil (southeast: GO, DF, MG, ES, RJ, SP, PR, SC, RS)
 9) Mato Grosso do Sul
10) Mato Grosso
11) Pará (west)
12) Rondônia
13) Roraima
14) Amazonas (east)
15) Amazonas (west)
16) Acre
#? 8
```

Non tutti i nomi delle località sono disponibili, ma scegliere la regione più vicina sarà sufficiente. Le informazioni date saranno poi utilizzate da `tzselect` per visualizzare il fuso orario corrispondente:

```
The following information has been given:

        Brazil
        Brazil (southeast: GO, DF, MG, ES, RJ, SP, PR, SC, RS)

Therefore TZ='America/Sao_Paulo' will be used.
Selected time is now:   sex out 18 18:47:07 -03 2019.
Universal Time is now:  sex out 18 21:47:07 UTC 2019.
Is the above information OK?
1) Yes
2) No
#? 1

You can make this change permanent for yourself by appending the line
        TZ='America/Sao_Paulo'; export TZ
to the file '.profile' in your home directory; then log out and log in again.

Ecco di nuovo quel valore TZ, questa volta su standard output in modo da
potete usare il comando /usr/bin/tzselect negli script di shell:
```

```
America/Sao_Paulo
```

Il nome del fuso orario risultante, `America/Sao_Paulo`, può anche essere usato come contenuto di `/etc/timezone` per informare il fuso orario predefinito del sistema:

```
$ cat /etc/timezone
America/Sao_Paulo
```

Come indicato dall'output di `tzselect`, la variabile d'ambiente `TZ` definisce il fuso orario per la sessione di shell, qualunque sia il fuso orario di default del sistema. Aggiungendo la linea `TZ='America/Sao_Paulo'; export TZ` al file `~/.profile` renderà `America/Sao_Paulo` il fuso orario per le future sessioni dell'utente. La variabile `TZ` può anche essere temporaneamente modificata durante la sessione corrente, per visualizzare l'ora in un diverso fuso orario:

```
$ env TZ='Africa/Cairo' date
Mon Oct 21 15:45:21 EET 2019
```

Nell'esempio, il comando `env` esegue il comando dato in una nuova sessione sub-shell con le stesse variabili d'ambiente della sessione corrente, tranne la variabile `TZ`, modificata dall'argomento `TZ='Africa/Cairo'`.

Ora Legale

Molte regioni adottano l'ora legale per una parte dell'anno — quando gli orologi sono in genere portati avanti di un'ora — che potrebbe portare un sistema mal configurato a segnalare l'ora sbagliata durante quel periodo dell'anno.

Il file `/etc/localtime` contiene i dati usati dal sistema operativo per regolare il suo orologio di conseguenza. I sistemi Linux standard hanno file per tutti i fusi orari nella directory `/usr/share/zoneinfo/`, quindi `/etc/localtime` è solo un collegamento simbolico al file di dati effettivo all'interno di quella directory. I file in `/usr/share/zoneinfo/` sono organizzati per il nome del fuso orario corrispondente, quindi il file di dati per il fuso orario `America/Sao_Paulo` sarà `/usr/share/zoneinfo/America/Sao_Paulo`.

Poiché le definizioni per l'ora legale possono cambiare, è importante mantenere aggiornati i file in `/usr/share/zoneinfo/`. Il comando di aggiornamento dello strumento di gestione dei pacchetti fornito dalla distribuzione dovrebbe aggiornarli ogni volta che è disponibile una nuova versione.

Lingua e Codifica dei Caratteri

I sistemi Linux possono lavorare con un'ampia varietà di lingue e codifiche di caratteri non occidentali, definizioni note come *locales*. La configurazione più basilare di ciò è la definizione della variabile d'ambiente `LANG`, dalla quale la maggior parte dei programmi di shell identificano la lingua da usare.

Il contenuto della variabile `LANG` segue il formato `ab_CD`, dove `ab` è il codice della lingua e `CD` è il codice della regione. Il codice della lingua dovrebbe seguire lo standard ISO-639 e il codice della regione dovrebbe seguire lo standard ISO-3166. Un sistema configurato per usare il portoghese brasiliano, per esempio, dovrebbe avere la variabile `LANG` definita a `pt_BR.UTF-8`:

```
$ echo $LANG
pt_BR.UTF-8
```

Come si vede nell'output di esempio, la variabile `LANG` contiene anche la codifica dei caratteri prevista per il sistema. ASCII, abbreviazione di *American Standard Code for Information Interchange*, è stato il primo standard di codifica dei caratteri ampiamente utilizzato per la comunicazione elettronica. Tuttavia, poiché ASCII ha una gamma molto limitata di valori numerici disponibili ed era basato sull'alfabeto inglese, non contiene caratteri usati da altre lingue o un insieme esteso di simboli non alfabetici. La codifica UTF-8 è uno standard *Unicode* per i normali caratteri occidentali, più molti altri simboli non convenzionali. Come dichiarato dallo *Unicode Consortium*, il manutentore dello *Standard Unicode*, questa codifica dovrebbe essere adottata di base per assicurare la compatibilità tra le piattaforme di computer:

> Lo standard Unicode fornisce un numero unico per ogni carattere, non importa quale piattaforma, dispositivo, applicazione o lingua. È stato adottato da tutti i moderni fornitori di software e ora permette ai dati di essere trasportati attraverso molte piattaforme, dispositivi e applicazioni diverse senza corruzione. Il supporto di Unicode costituisce la base per la rappresentazione di lingue e simboli in tutti i principali sistemi operativi, motori di ricerca, browser, computer portatili e smartphone - oltre a Internet e al World Wide Web (URL, HTML, XML, CSS, JSON, ecc.). (...) lo standard Unicode e la disponibilità di strumenti che lo supportano sono tra le più significative tendenze recenti della tecnologia software globale.
>
> — The Unicode Consortium, Cos'è Unicode?

Alcuni sistemi possono ancora usare standard definiti da ISO — come lo standard ISO-8859-1 — per la codifica dei caratteri non-ASCII. Tuttavia, tali standard di codifica dei caratteri dovrebbero essere deprecati in favore degli standard di codifica Unicode. Tuttavia, tutti i maggiori sistemi operativi tendono ad adottare lo standard Unicode di default.

Le impostazioni di localizzazione a livello di sistema sono configurate nel file `/etc/locale.conf`. La variabile `LANG` e altre variabili relative al *locale* sono assegnate in questo file come una normale variabile di shell, per esempio:

```
$ cat /etc/locale.conf
LANG=pt_BR.UTF-8
```

Gli utenti possono usare una configurazione personalizzata del locale ridefinendo la variabile d'ambiente `LANG`. Può essere fatto solo per la sessione corrente o per le sessioni future, aggiungendo la nuova definizione al profilo Bash dell'utente in `~/.bash_profile` o `~/.profile`. Finché l'utente non effettua il login, comunque, il locale di sistema predefinito sarà ancora usato da programmi indipendenti dall'utente, come la schermata di login del display manager.

TIP Il comando `localectl`, disponibile sui sistemi che impiegano *systemd* come gestore di sistema, può anche essere usato per interrogare e cambiare il *locale* di sistema. Per esempio: `localectl set-locale LANG=en_US.UTF-8`.

Oltre alla variabile `LANG`, altre variabili d'ambiente hanno un effetto su aspetti specifici del *locale*, come il simbolo di valuta da usare o il corretto separatore delle migliaia per i numeri:

`LC_COLLATE`
: Imposta l'ordine alfabetico. Uno dei suoi scopi è quello di definire l'ordine in cui file e directory sono elencati.

`LC_CTYPE`
: Imposta come il sistema tratterà certi insiemi di caratteri. Definisce, per esempio, quali caratteri considerare come *uppercase* o *lowercase*.

`LC_MESSAGES`
: Imposta il linguaggio per visualizzare i messaggi dei programmi (per lo più programmi GNU).

`LC_MONETARY`
: Imposta l'unità monetaria e il formato della valuta.

`LC_NUMERIC`
: Imposta il formato numerico per i valori non monetari. Il suo scopo principale è quello di definire le migliaia e i separatori decimali.

`LC_TIME`
: Imposta il formato dell'ora e della data.

`LC_PAPER`
: Imposta il formato standard della carta.

`LC_ALL`
: Sovrascrive tutte le altre variabili, incluso `LANG`.

Il comando `locale` mostrerà tutte le variabili definite nella configurazione del *locale* corrente:

```
$ locale
LANG=pt_BR.UTF-8
LC_CTYPE="pt_BR.UTF-8"
LC_NUMERIC=pt_BR.UTF-8
LC_TIME=pt_BR.UTF-8
LC_COLLATE="pt_BR.UTF-8"
LC_MONETARY=pt_BR.UTF-8
LC_MESSAGES="pt_BR.UTF-8"
LC_PAPER=pt_BR.UTF-8
LC_NAME=pt_BR.UTF-8
LC_ADDRESS=pt_BR.UTF-8
LC_TELEPHONE=pt_BR.UTF-8
LC_MEASUREMENT=pt_BR.UTF-8
LC_IDENTIFICATION=pt_BR.UTF-8
LC_ALL=
```

L'unica variabile non definita è `LC_ALL`, che può essere usata per sovrascrivere temporaneamente tutte le altre impostazioni di localizzazione. L'esempio seguente mostra come il comando `date` — eseguito in un sistema configurato con il locale `pt_BR.UTF-8` — modificherà il suo output per conformarsi alla nuova variabile `LC_ALL`:

```
$ date
seg out 21 10:45:21 -03 2019
$ env LC_ALL=en_US.UTF-8 date
Mon Oct 21 10:45:21 -03 2019
```

La modifica della variabile `LC_ALL` ha reso entrambe le abbreviazioni per il giorno della settimana e il nome del mese da mostrare in inglese americano (`en_US`). Non è obbligatorio, tuttavia, impostare lo stesso locale per tutte le variabili. È possibile, per esempio, avere la lingua definita a `pt_BR` e il formato numerico (`LC_NUMERIC`) impostato allo standard americano.

Alcune impostazioni di localizzazione cambiano il modo in cui i programmi trattano l'ordine alfabetico e i formati dei numeri. Mentre i programmi convenzionali sono solitamente preparati a

scegliere correttamente un *locale* comune per queste situazioni, gli script possono comportarsi in modo inaspettato quando cercano, per esempio, di ordinare in ordine alfabetico una lista di elementi. Per questo motivo, si raccomanda di impostare la variabile d'ambiente `LANG` al *locale* comune `C`, come in `LANG=C`, così lo script produrrà risultati non ambigui, indipendentemente dalle definizioni di localizzazione usate nel sistema in cui viene eseguito. Il *locale* C conduce solo un semplice confronto *bytewise*, quindi avrà anche prestazioni migliori degli altri.

Conversione della Codifica

Il testo può apparire con caratteri incomprensibili quando viene visualizzato su un sistema con una configurazione di codifica dei caratteri diversa dal sistema in cui il testo è stato creato. Il comando `iconv` può essere usato per risolvere questo problema, convertendo il file dalla sua codifica di caratteri originale a quella desiderata. Per esempio, per convertire un file chiamato `original.txt` dalla codifica ISO-8859-1 al file chiamato `converted.txt` usando la codifica UTF-8 si può usare il seguente comando:

```
$ iconv -f ISO-8859-1 -t UTF-8 original.txt > converted.txt
```

L'opzione `-f ISO-8859-1` (o `--from-code=ISO-8859-1`) imposta la codifica del file originale e l'opzione `-t UTF-8` (o `--to-code=UTF-8`) imposta la codifica per il file da convertire. Tutte le codifiche supportate dal comando `iconv` sono elencate con il comando `iconv -l` o `iconv --list`. Invece di usare il reindirizzamento dell'output, come nell'esempio, si possono usare anche le opzioni `-o converted.txt` o `--output converted.txt`.

Esercizi Guidati

1. In base al seguente output del comando `date`, qual è il fuso orario del sistema in notazione GMT?

```
$ date
Mon Oct 21 18:45:21 +05 2019
```

2. A quale file dovrebbe puntare il link simbolico `/etc/localtime` per rendere `Europe/Brussels` l'ora locale di default del sistema?

3. I caratteri nei file di testo potrebbero non essere resi correttamente in un sistema con una codifica dei caratteri diversa da quella usata nel documento di testo. Come si può usare `iconv` per convertire il file codificato WINDOWS-1252 `old.txt` nel file `new.txt` usando la codifica UTF-8?

Esercizi Esplorativi

1. Quale comando renderà `Pacific/Auckland` il fuso orario predefinito per la sessione corrente della shell?

2. Il comando `uptime` mostra, tra le altre cose, il *carico medio* del sistema in numeri frazionari. Utilizza le impostazioni locali correnti per decidere se il separatore decimale debba essere un punto o una virgola. Se, per esempio, il *locale* corrente è impostato su `de_DE.UTF-8` (lo standard della Germania), `uptime` userà una virgola come separatore. Sapendo che nella lingua inglese americana si usa il punto come separatore, quale comando farà sì che `uptime` visualizzi i frazionari usando un punto invece di una virgola per il resto della sessione corrente?

3. Il comando `iconv` sostituirà tutti i caratteri al di fuori dell'insieme di caratteri di destinazione con un punto interrogativo. Se `//TRANSLIT` è aggiunto alla codifica di destinazione, i caratteri non rappresentati nel set di caratteri di destinazione saranno sostituiti (traslitterati) da uno o più caratteri simili. Come potrebbe essere usato questo metodo per convertire un file di testo UTF-8 chiamato `readme.txt` in un semplice file ASCII chiamato `ascii.txt`?

Sommario

Questa lezione tratta del come configurare un sistema Linux per lavorare con lingue e impostazioni temporali personalizzate. Vengono trattati anche i concetti e le impostazioni di codifica dei caratteri, che sono molto importanti per rendere correttamente il contenuto del testo. La lezione tratta i seguenti argomenti:

- Come i sistemi Linux selezionano la lingua per visualizzare i messaggi della shell.
- Capire come i fusi orari influenzano l'ora locale.
- Identificare il fuso orario appropriato e modificare di conseguenza le impostazioni di sistema.
- Cosa sono le codifiche dei caratteri e come convertirle.

I comandi e le procedure affrontati sono stati:

- Variabili d'ambiente relative alla lingua e al tempo, come `LC_ALL`, `LANG` e `TZ`.
- `/etc/timezone`
- `/etc/localtime`
- `/usr/share/zoneinfo/`
- `locale`
- `tzselect`
- `timedatectl`
- `date`
- `iconv`

Risposte agli Esercizi Guidati

1. In base al seguente output del comando `date`, qual è il fuso orario del sistema in notazione GMT?

   ```
   $ date
   Mon Oct 21 18:45:21 +05 2019
   ```

   È il fuso orrio `Etc/GMT+5`.

2. A quale file dovrebbe puntare il link simbolico `/etc/localtime` per rendere `Europe/Brussels` l'ora locale di default del sistema?

   Il link `/etc/localtime` dovrebbe puntare a `/usr/share/zoneinfo/Europe/Brussels`.

3. I caratteri nei file di testo potrebbero non essere resi correttamente in un sistema con una codifica dei caratteri diversa da quella usata nel documento di testo. Come si può usare `iconv` per convertire il file codificato WINDOWS-1252 `old.txt` nel file `new.txt` usando la codifica UTF-8?

   Il comando `iconv -f WINDOWS-1252 -t UTF-8 -o new.txt old.txt` eseguirà la conversione desiderata.

Risposte agli Esercizi Esplorativi

1. Quale comando renderà `Pacific/Auckland` il fuso orario predefinito per la sessione corrente della shell?

   `export TZ=Pacific/Auckland`

2. Il comando `uptime` mostra, tra le altre cose, il *carico medio* del sistema in numeri frazionari. Utilizza le impostazioni locali correnti per decidere se il separatore decimale debba essere un punto o una virgola. Se, per esempio, il *locale* corrente è impostato su `de_DE.UTF-8` (lo standard della Germania), `uptime` userà una virgola come separatore. Sapendo che nella lingua inglese americana si usa il punto come separatore, quale comando farà sì che `uptime` visualizzi i frazionari usando un punto invece di una virgola per il resto della sessione corrente?

   Il comando `export LC_NUMERIC=en_US.UTF-8` o `export LC_ALL=en_US.UTF-8`.

3. Il comando `iconv` sostituirà tutti i caratteri al di fuori dell'insieme di caratteri di destinazione con un punto interrogativo. Se `//TRANSLIT` è aggiunto alla codifica di destinazione, i caratteri non rappresentati nel set di caratteri di destinazione saranno sostituiti (traslitterati) da uno o più caratteri simili. Come potrebbe essere usato questo metodo per convertire un file di testo UTF-8 chiamato `readme.txt` in un semplice file ASCII chiamato `ascii.txt`?

   Il comando `iconv -f UTF-8 -t ASCII//TRANSLIT -o ascii.txt readme.txt` effettuerà la conversione desiderata.



Linux Professional Institute

Argomento 108: Servizi Essenziali di Sistema



108.1 Mantenere l'orario di sistema

Obiettivi LPI di riferimento

LPIC-1 version 5.0, Exam 102, Objective 108.1

Peso

3

Aree di Conoscenza Chiave

- Impostare la data e l'ora del sistema.
- Impostare l'orologio hardware sull'ora corretta in UTC.
- Configurare il fuso orario corretto.
- Configurare base di NTP utilizzando ntpd e chrony.
- Conoscenza dell'utilizzo del servizio pool.ntp.org.
- Conoscenza del comando ntpq.

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

- `/usr/share/zoneinfo/`
- `/etc/timezone`
- `/etc/localtime`
- `/etc/ntp.conf`
- `/etc/chrony.conf`
- `date`
- `hwclock`
- `timedatectl`

- `ntpd`
- `ntpdate`
- `chronyc`
- `pool.ntp.org`

108.1 Lezione 1

Certificazione:	LPIC-1 (102)
Versione:	5.0
Argomento:	108 Servizi Essenziali di Sistema
Obiettivo:	108.1 Mantenere l'orario di sistema
Lezione:	1 di 2

Introduzione

La misurazione accurata del tempo è assolutamente cruciale per l'informatica moderna. L'implementazione del mantenimento del tempo, tuttavia, è materia sorprendentemente complessa. La pratica di tenere il tempo sembra banale per un utente finale, ma il sistema deve essere in grado di gestire in modo intelligente molte idiosincrasie e casi limite. Considerate che i fusi orari non sono statici, ma possono essere cambiati da una decisione amministrativa o politica. Un paese può scegliere di smettere di osservare l'ora legale. Qualsiasi programma deve essere in grado di gestire questi cambiamenti in modo logico. Fortunatamente per gli amministratori di sistema, le soluzioni per la gestione del tempo sul sistema operativo Linux sono mature, robuste e di solito funzionano senza molte interferenze.

Quando un computer Linux si avvia, inizia a *tenere il tempo*. Ci riferiamo a questo come a un *system clock*, poiché è aggiornato dal sistema operativo. Inoltre, i computer moderni avranno anche un *hardware* o *real time clock*. Questo orologio hardware è spesso una caratteristica della scheda madre e tiene il tempo indipendentemente dal fatto che il computer sia in funzione o meno. Durante l'avvio, l'ora del sistema è impostata dall'orologio hardware, ma per la maggior parte questi due orologi funzionano indipendentemente l'uno dall'altro. In questa lezione discuteremo i metodi per interagire sia con l'orologio di sistema sia con quello hardware.

Sulla maggior parte dei moderni sistemi Linux, l'ora del sistema e l'ora dell'hardware sono sincronizzati con il *network time*, che è implementato dal *Network Time Protocol* (NTP). Nella stragrande maggioranza dei casi, l'unica configurazione che un normale utente dovrà fare è impostare il proprio fuso orario e l'NTP si occuperà del resto. Comunque, tratteremo alcuni modi manuali di lavorare con il tempo; le specifiche della configurazione del *network time* saranno trattate nella prossima lezione.

Tempo Locale e Tempo Universale a Confronto

L'orologio del sistema è impostato sul Coordinated Universal Time (UTC), che è l'ora locale di Greenwich, Regno Unito. Di solito un utente vuole sapere il suo *local time*. L'ora locale è calcolata prendendo l'ora UTC e applicando un *offset* basato sul fuso orario e sull'ora legale. In questo modo, si possono evitare parecchie complessità.

L'orologio di sistema può essere impostato sia sull'ora UTC sia sull'ora locale, ma si raccomanda che sia impostato sull'ora UTC.

La Data

`date` è un'utilità di base che visualizza semplicemente l'ora locale:

```
$ date
Sun Nov 17 12:55:06 EST 2019
```

Modificando le opzioni del comando `date` si cambia il formato dell'output.

Per esempio, un utente può usare `date -u` per vedere l'ora UTC corrente.

```
$ date -u
Sun Nov 17 18:02:51 UTC 2019
```

Alcune altre opzioni comunemente usate restituiranno l'ora locale in un formato che aderisce a un formato RFC accettato:

`-I`

Data/ora nel formato ISO 8601. L'aggiunta di `date` (`-Idate`) limiterà l'output alla sola data. Altri formati sono `hours`, `minutes`, `seconds` e `ns` per i nanosecondi.

`-R`

Restituisce data e ora nel formato RFC 5322.

`--rfc-3339`

Restituisce data e ora nel formato RFC 3339.

Il formato di `date` può essere personalizzato dall'utente con sequenze specificate indicate nella sua pagina `man`. Per esempio, l'ora corrente può essere formattata come ora Unix in questo modo:

```
$ date +%s
1574014515
```

Dalla pagina man di `date` possiamo vedere che `%s` si riferisce al tempo di Unix.

Lo *Unix time* è usato internamente sulla maggior parte dei sistemi Unix-like. Memorizza il tempo UTC come numero di secondi dall'*Epoch*, corrispondente al 1 gennaio 1970.

NOTE

Al momento attuale il numero di bit richiesto per memorizzare il tempo di Unix è di 32 bit. C'è un problema futuro quando 32 bit diventeranno insufficienti per contenere l'ora corrente in formato Unix. Questo causerà seri problemi per qualsiasi sistema Linux a 32 bit. Fortunatamente, questo non accadrà fino al 19 gennaio 2038.

Usando queste sequenze siamo in grado di formattare data e ora in quasi tutti i formati richiesti da qualsiasi applicazione. Naturalmente, nella maggior parte dei casi è di gran lunga preferibile attenersi a uno standard accettato.

Inoltre, `date --date` può essere usato per formattare un'ora che non è quella corrente. In questo scenario, un utente può specificare la data da applicare al sistema usando l'ora di Unix, per esempio:

```
$ date --date='@1564013011'
Wed Jul 24 20:03:31 EDT 2019
```

L'uso dell'opzione `--debug` può essere molto utile per assicurarsi che una data possa essere analizzata con successo. Osserva cosa succede quando si passa una data valida al comando:

```
$ date --debug --date="Fri, 03 Jan 2020 14:00:17 -0500"
date: parsed day part: Fri (day ordinal=0 number=5)
date: parsed date part: (Y-M-D) 2020-01-03
```

```
date: parsed time part: 14:00:17 UTC-05
date: input timezone: parsed date/time string (-05)
date: using specified time as starting value: '14:00:17'
date: warning: day (Fri) ignored when explicit dates are given
date: starting date/time: '(Y-M-D) 2020-01-03 14:00:17 TZ=-05'
date: '(Y-M-D) 2020-01-03 14:00:17 TZ=-05' = 1578078017 epoch-seconds
date: timezone: system default
date: final: 1578078017.000000000 (epoch-seconds)
date: final: (Y-M-D) 2020-01-03 19:00:17 (UTC)
date: final: (Y-M-D) 2020-01-03 14:00:17 (UTC-05)
```

Questo può essere uno strumento utile nella risoluzione dei problemi di un'applicazione che genera una data.

Hardware Clock

Un utente può eseguire il comando `hwclock` per vedere l'ora mantenuta nell'orologio in tempo reale. Questo comando richiede privilegi elevati, quindi useremo `sudo` per eseguire il comando:

```
$ sudo hwclock
2019-11-20 11:31:29.217627-05:00
```

Usando l'opzione `--verbose` si ottengono più output, il che potrebbe essere utile per la risoluzione di problemi:

```
$ sudo hwclock --verbose
hwclock from util-linux 2.34
System Time: 1578079387.976029
Trying to open: /dev/rtc0
Using the rtc interface to the clock.
Assuming hardware clock is kept in UTC time.
Waiting for clock tick...
...got clock tick
Time read from Hardware Clock: 2020/01/03 19:23:08
Hw clock time : 2020/01/03 19:23:08 = 1578079388 seconds since 1969
Time since last adjustment is 1578079388 seconds
Calculated Hardware Clock drift is 0.000000 seconds
2020-01-03 14:23:07.948436-05:00
```

Nota il `Calculated Hardware Clock drift`. Questo output può dirti se il tempo di sistema e il tempo dell'hardware stanno deviando l'uno dall'altro. === timedatectl

`timedatectl` è un comando che può essere usato per controllare lo stato generale dell'ora e della data, incluso se l'ora della rete è stata sincronizzata o meno (il Network Time Protocol sarà trattato nella prossima lezione).

Per default `timedatectl` restituisce informazioni simili a quelle di `date`, ma con l'aggiunta dell'ora RTC (hardware) e dello stato del servizio NTP:

```
$ timedatectl
               Local time: Thu 2019-12-05 11:08:05 EST
           Universal time: Thu 2019-12-05 16:08:05 UTC
                 RTC time: Thu 2019-12-05 16:08:05
                Time zone: America/Toronto (EST, -0500)
System clock synchronized: yes
              NTP service: active
          RTC in local TZ: no
```

Impostare l'Ora con timedatectl

Se NTP non è disponibile, si raccomanda di usare `timedatectl` piuttosto che `date` o `hwclock` per impostare data e ora:

```
# timedatectl set-time '2011-11-25 14:00:00'
```

Il processo è simile a quello di `date`. L'utente può anche impostare il tempo indipendentemente dalla data usando il formato HH:MM:SS.

Impostare il Fuso Orario con timedatectl

`timedatectl` è il modo preferito per impostare il fuso orario locale sui sistemi Linux basati su `systemd` quando non esiste una GUI. `timedatectl` elencherà i possibili fusi orari e il fuso orario può essere impostato di conseguenza usando uno di essi come argomento.

Per prima cosa elenchiamo i possibili fusi orari:

```
$ timedatectl list-timezones
Africa/Abidjan
Africa/Accra
Africa/Algiers
Africa/Bissau
Africa/Cairo
```

```
...
```

La lista dei possibili fusi orari è lunga, quindi l'uso del comando `grep` è raccomandato in questo caso.

Poi possiamo impostare il fuso orario usando uno degli elementi della lista che è stata restituita:

```
$ timedatectl set-timezone Africa/Cairo
$ timedatectl
               Local time: Thu 2019-12-05 18:18:10 EET
           Universal time: Thu 2019-12-05 16:18:10 UTC
                 RTC time: Thu 2019-12-05 16:18:10
                Time zone: Africa/Cairo (EET, +0200)
System clock synchronized: yes
              NTP service: active
          RTC in local TZ: no
```

Tieni presente che il nome del fuso orario deve essere esatto. `Africa/Cairo` per esempio cambierà il fuso orario, ma `cairo` o `africa/cairo` no.

Disabilitare NTP Utilizzando timedatectl

In alcuni casi potrebbe essere necessario disabilitare NTP. Questo potrebbe essere fatto usando `systemctl`, ma noi lo dimostreremo usando `timedatectl`:

```
# timedatectl set-ntp no
$ timedatectl
             Local time: Thu 2019-12-05 18:19:04 EET Universal time: Thu 2019-12-05 16:19:04
UTC
               RTC time: Thu 2019-12-05 16:19:04
              Time zone: Africa/Cairo (EET, +0200)
            NTP enabled: no
       NTP synchronized: no
        RTC in local TZ: no
             DST active: n/a
```

Configurare il Fuso Oraio senza timedatectl

Impostare le informazioni sul fuso orario è un passo standard quando si installa Linux su una nuova macchina. Se c'è un processo di installazione grafico, questo sarà molto probabilmente

gestito senza alcun ulteriore input da parte dell'utente.

La directory `/usr/share/zoneinfo` contiene informazioni per i diversi fusi orari possibili. Nella directory `zoneinfo`, ci sono sottodirectory che contengono i nomi dei continenti così come altri collegamenti simbolici. Si raccomanda di trovare la `zoneinfo` della tua regione partendo dal tuo continente.

I file `zoneinfo` contengono le regole necessarie per calcolare l'offset dell'ora locale rispetto all'UTC, e sono anche importanti se la tua regione osserva l'ora legale. Il contenuto di `/etc/localtime` sarà letto quando Linux ha bisogno di determinare il fuso orario locale. Per impostare il fuso orario senza l'uso di una GUI, l'utente dovrebbe creare un collegamento simbolico per la propria posizione da `/usr/share/zoneinfo` a `/etc/localtime`. Per esempio:

```
$ ln -s /usr/share/zoneinfo/Canada/Eastern /etc/localtime
```

Dopo aver impostato il fuso orario corretto, si raccomanda di eseguire:

```
# hwclock --systohc
```

Questo imposterà l'*hardware clock* dal *system clock* (cioè, l'orologio *real-time* sarà impostato alla stessa ora di `date`). Si noti che questo comando viene eseguito con i privilegi di root, in questo caso facendo il login come root.

`/etc/timezone` è simile a `/etc/localtime`. È una rappresentazione di dati del fuso orario locale, e come tale può essere letto usando `cat`:

```
$ cat /etc/timezone
America/Toronto
```

Si noti che questo file *non è* usato da tutte le distribuzioni Linux.

Impostare Data e Ora senza timedatectl

NOTE

La maggior parte dei moderni sistemi Linux usa `systemd` per la sua configurazione e i suoi servizi e quindi non si raccomanda di usare `date` o `hwclock` per impostare l'ora. `systemd` usa `timedatectl` per questo. Tuttavia è importante conoscere questi comandi *legacy* nel caso in cui si debba amministrare un vecchio sistema.

Utilizzare date

`date` ha un'opzione per impostare l'ora di sistema. Usa `--set` o `-s` per impostare la data e l'ora. Puoi anche scegliere di usare `--debug` per verificare la corretta analisi del comando:

```
# date --set="11 Nov 2011 11:11:11"
```

Si noti che sono necessari i privilegi di root per impostare la data qui. Possiamo anche scegliere di cambiare l'ora o la data in modo indipendente:

```
# date +%Y%m%d -s "20111125"
```

Qui dobbiamo specificare le sequenze in modo che la nostra stringa sia elaborata correttamente. Per esempio `%Y` si riferisce all'anno, e quindi le prime quattro cifre `2011` saranno interpretate come l'anno 2011. Allo stesso modo, `%T` è la sequenza per il tempo, ed è dimostrato qui impostando il tempo:

```
# date +%T -s "13:11:00"
```

Dopo aver cambiato l'orario di sistema, si raccomanda di impostare anche l'orologio hardware in modo che entrambi gli orologi, di sistema e hardware, siano sincronizzati:

```
# hwclock --systohc
```

`systohc` significa "da system clock a hardware clock".

Utilizzare hwclock

Invece di impostare l'orologio di sistema e aggiornare l'orologio hardware, si può scegliere di invertire il processo. Inizieremo impostando l'orologio hardware:

```
# hwclock --set --date "4/12/2019 11:15:19"
# hwclock
Fri 12 Apr 2019 6:15:19 AM EST -0.562862 seconds
```

Nota che `hwclock` si aspetta l'ora UTC, ma restituisce l'ora locale per default.

Dopo aver impostato l'orologio hardware, avremo bisogno di aggiornare l'orologio di sistema da

esso. `hctosys` può essere inteso come “da orologio hardware a orologio di sistema”.

```
# hwclock --hctosys
```

Esercizi Guidati

1. Indica se i seguenti comandi stanno visualizzando o modificando il *system time* o l'*hardware time*:

Comando(i)	System	Hardware	Entrambi
`date -u`			
`hwclock --set --date "12:00:00"`			
`timedatectl`			
`timedatectl | grep RTC`			
`hwclock --hctosys`			
`date +%T -s "08:00:00"`			
`timedatectl set-time 1980-01-10`			

2. Osserva il seguente output, e poi correggi il formato dell'argomento in modo che il comando abbia successo:

```
$ date --debug --date "20/20/12 0:10 -3"

date: warning: value 20 has less than 4 digits. Assuming MM/DD/YY[YY]
date: parsed date part: (Y-M-D) 0002-20-20
date: parsed time part: 00:10:00 UTC-03
date: input timezone: parsed date/time string (-03)
date: using specified time as starting value: '00:10:00'
date: error: invalid date/time value:
date:     user provided time: '(Y-M-D) 0002-20-20 00:10:00 TZ=-03'
date:        normalized time: '(Y-M-D) 0003-08-20 00:10:00 TZ=-03'
date:                                 ---- --
date:      possible reasons:
date:        numeric values overflow;
date:        incorrect timezone
date: invalid date '20/20/2 0:10 -3'
```

3. Usa il comando `date` e le sequenze in modo che il mese del sistema sia impostato su febbraio. Lascia il resto della data e dell'ora invariati.

4. Supponendo che il comando precedente abbia avuto successo, usa `hwclock` per impostare l'orologio hardware dall'orologio di sistema.

5. C'è una località chiamata `eucla`. Di quale continente fa parte? Usa il comando `grep` per scoprirlo.

6. Imposta il tuo fuso orario attuale su quello di `eucla`.

Esercizi Esplorativi

1. Quale metodo di impostazione del tempo è ottimale? In quale scenario il metodo preferito potrebbe essere impossibile?

2. Perché è necessario che ci siano così tanti metodi per realizzare la stessa cosa, cioè impostare il tempo del sistema?

3. Dopo il 19 gennaio 2038, il Linux System Time richiederà un numero a 64 bit da memorizzare. Tuttavia, è possibile che si possa semplicemente scegliere di impostare una "Nuova Epoca (Epoch)". Per esempio, il 1° gennaio 2038 a mezzanotte potrebbe essere impostato su un New Epoch Time di `0`. Perché non è che questa non sia diventata la soluzione preferita?

Sommario

In questa lezione abbiamo imparato:

- Come visualizzare l'ora in diversi formati dalla riga di comando.
- La differenza tra l'orologio di sistema e l'orologio hardware in Linux.
- Come impostare manualmente l'orologio di sistema.
- Come impostare manualmente l'orologio hardware.
- Come cambiare il fuso orario del sistema.

Comandi utilizzati in questa lezione:

`date`
Visualizza o cambia l'orologio di sistema. Altre opzioni:

`-u`
Visualizza l'ora UTC.

`+%s`
Utilizza una sequenza per visualizzare l'ora e data Unix.

`--date=`
Specifica un'ora da visualizzare, invece dell'ora corrente.

`--debug`
Visualizza messaggi di debug quando analizza una data inserita dall'utente.

`-s`
Imposta manualmente l'orologio di sistema.

`hwclock`
Visualizza o cambia l'orologio hardware.

`--systohc`
Utilizza l'orologio di sistema per impostare l'orologio hardware.

`--hctosys`
Utilizza l'orologio hardware per impostare l'orologio di sistema.

`--set --date`

Imposta manualmente l'orologio hardware.

`timedatectl`

Visualizza gli orologi di sistema e hardware, così come la configurazione NTP sui sistemi Linux basati su systemd.

`set-time`

Imposta l'ora manualmente.

`list-timezones`

Elenca i possibili fusi orari.

`set-timezone`

Imposta il fuso orario manualmente.

`set-ntp`

Abilita/disabilita NTP.

Risposte agli Esercizi Guidati

1. Indica se i seguenti comandi stanno visualizzando o modificando il *system time* o l'*hardware time*:

Comando(i)	System	Hardware	Entrambi
`date -u`	X		
`hwclock --set --date "12:00:00"`		X	
`timedatectl`			X
`timedatectl | grep RTC`		X	
`hwclock --hctosys`	X		
`date +%T -s "08:00:00"`	X		
`timedatectl set-time 1980-01-10`			X

2. Osserva il seguente output, e poi correggi il formato dell'argomento in modo che il comando abbia successo:

```
$ date --debug --date "20/20/12 0:10 -3"

date: warning: value 20 has less than 4 digits. Assuming MM/DD/YY[YY]
date: parsed date part: (Y-M-D) 0002-20-20
date: parsed time part: 00:10:00 UTC-03
date: input timezone: parsed date/time string (-03)
date: using specified time as starting value: '00:10:00'
date: error: invalid date/time value:
date:     user provided time: '(Y-M-D) 0002-20-20 00:10:00 TZ=-03'
date:        normalized time: '(Y-M-D) 0003-08-20 00:10:00 TZ=-03'
date:                                 ---- --
date:      possible reasons:
date:        numeric values overflow;
date:        incorrect timezone
date: invalid date '20/20/2 0:10 -3'
```

`date --debug --set "12/20/20 0:10 -3"`

3. Usate il comando `date` e le sequenze in modo che il mese del sistema sia impostato su febbraio. Lascia il resto della data e dell'ora invariati.

   ```
   date +%m -s "2"
   ```

4. Supponendo che il comando precedente abbia avuto successo, usa `hwclock` per impostare l'orologio hardware dall'orologio di sistema.

   ```
   hwclock -systohc
   ```

5. C'è una località chiamata `eucla`. Di quale continente fa parte? Usa il comando `grep` per scoprirlo.

   ```
   timedatectl list-timezones \| grep -i eucla
   ```

   o

   ```
   grep -ri eucla /usr/share/zoneinfo
   ```

6. Imposta il tuo fuso orario attuale su quello di `eucla`.

   ```
   timedatectl set-timezone 'Australia/Eucla'
   ```

   o

   ```
   ln -s /usr/share/zoneinfo/Australia/Eucla /etc/localtime
   ```

Risposte agli Esercizi Esplorativi

1. Quale metodo di impostazione del tempo è ottimale? In quale scenario il metodo preferito potrebbe essere impossibile?

   Nella maggior parte delle distribuzioni Linux, NTP è abilitato di default e dovrebbe essere lasciato per impostare l'ora del sistema senza interferenze. Tuttavia, se c'è un sistema Linux che non è connesso a Internet, NTP sarà inaccessibile. Per esempio, un sistema Linux *embedded* che gira su apparecchiature industriali potrebbe non avere connettività di rete.

2. Perché è necessario che ci siano così tanti metodi per realizzare la stessa cosa, cioè impostare il tempo del sistema?

   Poiché l'impostazione dell'ora è stata un requisito di tutti i sistemi *nix per decenni, ci sono molti metodi legacy per l'impostazione dell'ora che sono ancora mantenuti.

3. Dopo il 19 gennaio 2038, il Linux System Time richiederà un numero a 64 bit da memorizzare. Tuttavia, è possibile che si possa semplicemente scegliere di impostare una "Nuova Epoca (Epoch)". Per esempio, il 1° gennaio 2038 a mezzanotte potrebbe essere impostato su un New Epoch Time di `0`. Perché pensi che questa non sia diventata la soluzione preferita?

   Entro il 2038 la stragrande maggioranza dei computer sarà già dotata di CPU a 64 bit, e l'uso di un numero a 64 bit non degraderà le prestazioni in modo significativo. Tuttavia, sarebbe impossibile stimare i rischi di "resettare" l'Epoch time in questo modo. C'è molto software legacy che potrebbe esserne colpito. Le banche e le grandi aziende, per esempio, hanno spesso una grande quantità di vecchi programmi su cui fanno affidamento per uso interno. Quindi questo scenario, come molti altri, è uno studio sui compromessi. Qualsiasi sistema a 32 bit ancora in funzione nel 2038 sarebbe colpito da un *overflow* di Epoch Time, ma il software legacy sarebbe colpito dal cambiamento del valore di Epoch.

 | Questa opera è concessa in licenza CC BY-NC-ND 4.0. |

108.1 Lezione 2

Certificazione:	LPIC-1 (102)
Versione:	5.0
Argomento:	108 Servizi Essenziali di Sistema
Obiettivo:	108.1 Mantenere l'orario di sistema
Lezione:	2 di 2

Introduzione

Mentre i personal computer sono in grado di mantenere un tempo ragionevolmente accurato da soli, gli ambienti informatici di produzione e quelli di rete richiedono il mentenimento di un tempo molto preciso. Il tempo più accurato è misurato da *orologi di riferimento*, che sono tipicamente orologi atomici. Il mondo moderno ha ideato un sistema per cui tutti i computer connessi a Internet possono essere sincronizzati con questi orologi di riferimento usando quello che è conosciuto come *Network Time Protocol* (NTP). Un sistema di computer con NTP sarà in grado di sincronizzare i propri orologi di sistema al'orario fornito dagli orologi di riferimento. Se il tempo di sistema e il tempo misurato su questi server sono diversi, allora il computer accelererà o rallenterà il suo tempo di sistema interno in modo incrementale fino a quando il tempo di sistema coinciderà con quello degli orologi di riferimento.

NTP utilizza una struttura gerarchica per diffondere l'orario. Gli orologi di riferimento sono collegati ai server in cima alla gerarchia. Questi server sono macchine *Stratum 1* e tipicamente non sono accessibili al pubblico. Le macchine dello strato 1 sono tuttavia accessibili alle macchine dello strato 2, che sono accessibili alle macchine dello strato 3 e così via. I server Stratum 2 sono accessibili al pubblico, così come tutte le macchine più in basso nella gerarchia. Quando si imposta l'NTP per una grande rete è buona pratica avere un piccolo numero di computer connessi ai

server Stratum 2, e poi avere quelle macchine che forniscono l'NTP a tutte le altre macchine. In questo modo, le richieste sulle macchine Stratum 2 possono essere minimizzate.

Ci sono alcuni termini importanti che è importante conoscere quando si parla di NTP. Alcuni di questi termini sono usati nei comandi che useremo per tracciare lo stato di NTP sulle nostre macchine:

Offset

Questo si riferisce alla differenza assoluta tra l'ora di sistema e l'ora NTP. Per esempio, se l'orologio di sistema indica le 12:00:02 e l'ora NTP indica le 11:59:58, allora l'offset tra i due orologi è di quattro secondi.

Step

Se l'offset temporale tra il fornitore NTP e un utilizzatore è maggiore di 128ms, allora NTP eseguirà un singolo cambiamento significativo al tempo del sistema, invece di rallentare o accelerare il tempo del sistema. Questo è chiamato *stepping*.

Slew

Slewing si riferisce alle modifiche apportate all'ora del sistema quando l'offset tra l'ora del sistema e l'NTP è inferiore a 128ms. Se questo è il caso, allora i cambiamenti saranno fatti gradualmente.

Insane Time

Se l'offset tra l'ora di sistema e l'ora NTP è maggiore di 17 minuti, allora l'ora di sistema è considerata *insane* e il demone NTP non introdurrà alcuna modifica all'ora di sistema. Dovranno essere prese misure speciali per portare l'ora di sistema entro 17 minuti dall'ora corretta.

Drift

Il *drift* si riferisce al fenomeno per cui due orologi diventano fuori sincrono nel tempo. In sostanza, se due orologi sono inizialmente sincronizzati ma poi diventano fuori sincrono nel tempo, allora si sta verificando un drift.

Jitter

Il *jitter* si riferisce alla quantità di *drift* dall'ultima volta che un orologio è stato interrogato. Quindi se l'ultima sincronizzazione NTP è avvenuta 17 minuti fa, e l'offset tra il provider NTP e l'utilizzatore è di 3 millisecondi, allora 3 millisecondi è il jitter.

Ora discuteremo alcuni dei modi specifici in cui Linux implementa NTP.

timedatectl

Se la tua distribuzione Linux usa `timedatectl`, per default implementa un client *SNTP* invece di un'implementazione NTP completa. Questa è un'implementazione meno complessa del *network time* e significa che la tua macchina non servirà NTP ad altri computer connessi.

In questo caso SNTP non funzionerà a meno che il servizio `timesyncd` sia in esecuzione. Come per tutti i servizi systemd, possiamo verificare che sia in esecuzione con:

```
$ systemctl status systemd-timesyncd
● systemd-timesyncd.service - Network Time Synchronization
   Loaded: loaded (/lib/systemd/system/systemd-timesyncd.service; enabled; vendor preset:
enabled)
  Drop-In: /lib/systemd/system/systemd-timesyncd.service.d
           └─disable-with-time-daemon.conf
   Active: active (running) since Thu 2020-01-09 21:01:50 EST; 2 weeks 1 days ago
     Docs: man:systemd-timesyncd.service(8)
 Main PID: 1032 (systemd-timesyn)
   Status: "Synchronized to time server for the first time 91.189.89.198:123
(ntp.ubuntu.com)."
    Tasks: 2 (limit: 4915)
   Memory: 3.0M
   CGroup: /system.slice/systemd-timesyncd.service
           └─1032 /lib/systemd/systemd-timesyncd

Jan 11 13:06:18 NeoMex systemd-timesyncd[1032]: Synchronized to time server for the first
time 91.189.91.157:123 (ntp.ubuntu.com).
...
```

Lo stato della sincronizzazione SNTP di `timedatectl` può essere verificato usando `show-timesync`:

```
$ timedatectl show-timesync --all
LinkNTPServers=
SystemNTPServers=
FallbackNTPServers=ntp.ubuntu.com
ServerName=ntp.ubuntu.com
ServerAddress=91.189.89.198
RootDistanceMaxUSec=5s
PollIntervalMinUSec=32s
PollIntervalMaxUSec=34min 8s
PollIntervalUSec=34min 8s
```

```
NTPMessage={ Leap=0, Version=4, Mode=4, Stratum=2, Precision=-23, RootDelay=8.270ms,
RootDispersion=18.432ms, Reference=91EECB0E, OriginateTimestamp=Sat 2020-01-25 18:35:49 EST,
ReceiveTimestamp=Sat 2020-01-25 18:35:49 EST, TransmitTimestamp=Sat 2020-01-25 18:35:49 EST,
DestinationTimestamp=Sat 2020-01-25 18:35:49 EST, Ignored=no PacketCount=263, Jitter=2.751ms
}
Frequency=-211336
```

Questa configurazione potrebbe essere adeguata per la maggior parte delle situazioni, ma, come notato prima, sarà insufficiente se si spera di sincronizzare diversi client in una rete. In questo caso si raccomanda di installare un client NTP completo.

Il Demone NTP

L'ora del sistema viene confrontata con l'ora della rete a intervalli regolari. Affinché questo funzioni dobbiamo avere un *demone* in esecuzione in background. Per molti sistemi Linux il nome di questo demone è `ntpd`. `ntpd` permetterà a una macchina non solo di essere un *consumatore di tempo* (cioè, in grado di sincronizzare il proprio orologio da una fonte esterna), ma anche di *fornire* il tempo ad altre macchine.

Assumiamo che il nostro computer sia basato su systemd e che usi `systemctl` per controllare i demoni. Installeremo i pacchetti `ntp` usando il gestore di pacchetti appropriato e poi ci assicureremo che il nostro demone `ntpd` sia in esecuzione controllando il suo stato:

```
$ systemctl status ntpd

●  ntpd.service - Network Time Service
   Loaded: loaded (/usr/lib/systemd/system/ntpd.service; enabled; vendor preset: disabled)
   Active: active (running) since Fri 2019-12-06 03:27:21 EST; 7h ago
  Process: 856 ExecStart=/usr/sbin/ntpd -u ntp:ntp $OPTIONS (code=exited, status=0/SUCCESS)
 Main PID: 867 (ntpd)
   CGroup: /system.slice/ntpd.service
           `-867 /usr/sbin/ntpd -u ntp:ntp -g
```

In alcuni casi potrebbe essere necessario avviare e abilitare `ntpd`. Sulla maggior parte delle macchine Linux questo viene fatto con:

```
# systemctl enable ntpd && systemctl start ntpd
```

Le interrogazioni NTP avvengono sulla porta UDP 123. Se NTP fallisce, assicurati che questa porta sia aperta e in ascolto.

La Configurazione NTP

NTP è in grado di interrogare diverse fonti e di selezionare i migliori candidati da utilizzare per impostare l'ora del sistema. Se una connessione di rete viene persa, NTP utilizza le regolazioni precedenti per stimare le regolazioni future.

A seconda della vostra distribuzione Linux, la lista dei server NTP di rete sarà memorizzata in posti diversi. Supponiamo che `ntp` sia installato sulla tua macchina.

Il file `/etc/ntp.conf` contiene informazioni di configurazione su come il tuo sistema si sincronizza con l'ora di rete. Questo file può essere letto e modificato usando `vi` o `nano`.

Per impostazione predefinita i server NTP utilizzati sono specificati in una sezione come questa:

```
# Use public servers from the pool.ntp.org project.
# Please consider joining the pool (http://www.pool.ntp.org/join.html).
server 0.centos.pool.ntp.org iburst
server 1.centos.pool.ntp.org iburst
server 2.centos.pool.ntp.org iburst
server 3.centos.pool.ntp.org iburst
```

La sintassi per aggiungere server NTP è simile a questa:

```
server (IP Address)
server server.url.localhost
```

Gli indirizzi dei server possono essere indirizzi IP o URL se il DNS è stato configurato correttamente. In questo caso, il server sarà sempre interrogato.

Un amministratore di rete potrebbe anche considerare l'uso (o la creazione) di un *pool*. In questo caso, assumiamo che ci siano diversi provider NTP, tutti con demoni NTP in esecuzione e con lo stesso orario. Quando un client interroga un pool, un provider viene selezionato a caso. Questo aiuta a distribuire il carico di rete tra molte macchine in modo che nessuna macchina nel pool stia gestendo tutte le query NTP.

Comunemente, `/etc/ntp.conf` sarà popolato con un pool di server chiamato `pool.ntp.org`. Così, per esempio, `server 0.centos.pool.ntp.org` è un pool NTP predefinito fornito alle macchine CentOS.

pool.ntp.org

I server NTP usati di default sono un progetto open source. Maggiori informazioni possono essere trovate su ntppool.org.

> Considerate se il pool NTP sia appropriato per le vostre necessità o meno. Se gli affari, l'organizzazione o la vita umana dipendono dall'avere un orario corretto o possono essere danneggiati dal fatto che sia sbagliato, non dovreste "prendere l'ora da Internet". L'NTP Pool è generalmente di alta qualità, ma è un servizio gestito da volontari nel loro tempo libero. Parlate con i vostri fornitori di attrezzature e di servizi per ottenere un servizio locale e affidabile per voi. Vedi anche i nostri termini di servizio. Noi raccomandiamo i time server di Meinberg, ma sono consigliabili anche server di End Run, Spectracom e molti altri.
>
> — ntppool.org

ntpdate

Durante la configurazione iniziale, l'ora del sistema e l'NTP potrebbero essere seriamente desincronizzati. Se lo *offset* tra l'ora di sistema e l'ora NTP è maggiore di 17 minuti, allora il demone NTP non apporterà modifiche all'ora di sistema. In questo scenario sarà necessario un intervento manuale.

Per prima cosa, se `ntpd` è in esecuzione, sarà necessario *fermare* il servizio. Usa `systemctl stop ntpd` per farlo.

Poi, usa `ntpdate pool.ntp.org` per eseguire una sincronizzazione iniziale *una tantum*, dove `pool.ntp.org` si riferisce all'indirizzo IP o all'URL di un server NTP. Può essere necessaria più di una sincronizzazione.

ntpq

`ntpq` è un'utilità per monitorare lo stato di NTP. Una volta che il demone NTP è stato avviato e configurato, `ntpq` può essere usato per controllare il suo stato:

```
$ ntpq -p
     remote           refid      st t when poll reach   delay   offset  jitter
==============================================================================
+37.44.185.42    91.189.94.4      3 u   86  128  377  126.509  -20.398   6.838
+ntp2.0x00.lv    193.204.114.233  2 u   82  128  377  143.885   -8.105   8.478
*inspektor-vlan1 121.131.112.137  2 u   17  128  377  112.878  -23.619   7.959
 b1-66er.matrix. 18.26.4.105      2 u  484  128   10   34.907   -0.811  16.123
```

In questo caso `-p` sta per *print* e stamperà a schermo un riassunto dei *peer*. Gli indirizzi degli host possono anche essere restituiti come indirizzi IP usando `-n`.

`remote`
: hostname del provider NTP.

`refid`
: ID di riferimento del provider NTP.

`st`
: Stratum del provider.

`when`
: Numero di secondi dall'ultima interrogazione.

`poll`
: Numero di secondi tra le query.

`reach`
: ID di stato che indica se un server è stato raggiunto. Le connessioni riuscite aumenteranno questo numero di 1.

`delay`
: Tempo in ms tra la query e la risposta del server.

`offset`
: Tempo in ms tra il tempo di sistema e il tempo NTP.

`jitter`
: Offset in ms tra l'ora di sistema e l'ora NTP nell'ultima interrogazione.

`ntpq` ha anche una modalità interattiva, a cui si accede quando viene eseguito senza opzioni o argomenti. L'opzione `?` restituirà una lista di comandi che `ntpq` riconoscerà.

chrony

`chrony` è un altro modo per implementare NTP. È installato di default su alcuni sistemi Linux, ma è disponibile per il download su tutte le principali distribuzioni. `chronyd` è il demone chrony, e `chronyc` è l'interfaccia a riga di comando. Potrebbe essere richiesto di avviare e abilitare `chronyd` prima di interagire con `chronyc`.

Se l'installazione di chrony ha una configurazione predefinita, l'uso del comando `chronyc tracking` fornirà informazioni sull'orario NTP e di sistema:

```
$ chronyc tracking
Reference ID    : 3265FB3D (bras-vprn-toroon2638w-lp130-11-50-101-251-61.dsl.)
Stratum         : 3
Ref time (UTC)  : Thu Jan 09 19:18:35 2020
System time     : 0.000134029 seconds fast of NTP time
Last offset     : +0.000166506 seconds
RMS offset      : 0.000470712 seconds
Frequency       : 919.818 ppm slow
Residual freq   : +0.078 ppm
Skew            : 0.555 ppm
Root delay      : 0.006151616 seconds
Root dispersion : 0.010947504 seconds
Update interval : 129.8 seconds
Leap status     : Normal
```

Questo output contiene molte informazioni, più di quelle disponibili in altre implementazioni.

`Reference ID`
: L'ID e il nome di riferimento a cui il computer è attualmente sincronizzato.

`Stratum`
: Numero di "salti" verso un computer con un orologio di riferimento collegato.

`Ref time`
: L'ora UTC in cui è stata fatta l'ultima misurazione dalla sorgente di riferimento.

`System time`
: Ritardo dell'orologio di sistema dal server sincronizzato.

`Last offset`
: Offset stimato dell'ultimo aggiornamento dell'orologio.

`RMS offset`
: Media a lungo termine del valore di offset.

`Frequency`
: Il grado di errore con cui l'orologio del sistema si troverebbe se *chronyd* non lo correggesse. È fornito in ppm (parti per milione).

`Residual freq`

Frequenza residua che indica la differenza tra le misure della sorgente di riferimento e la frequenza attualmente in uso.

`Skew`

Limite di errore stimato della frequenza.

`Root delay`

Totale dei ritardi del percorso di rete verso il computer *stratum*, dal quale il computer riceve la sincronizzazione.

`Leap status`

Lo stato di salto che può avere uno dei seguenti valori: normale, inserisci secondo, cancella secondo o non sincronizzato.

Possiamo anche osservare una serie di informazioni dettagliate sull'ultimo aggiornamento NTP valido:

```
# chrony ntpdata
Remote address  : 172.105.97.111 (AC69616F)
Remote port     : 123
Local address   : 192.168.122.81 (C0A87A51)
Leap status     : Normal
Version         : 4
Mode            : Server
Stratum         : 2
Poll interval   : 6 (64 seconds)
Precision       : -25 (0.000000030 seconds)
Root delay      : 0.000381 seconds
Root dispersion : 0.000092 seconds
Reference ID    : 61B7CE58 ()
Reference time  : Mon Jan 13 21:50:03 2020
Offset          : +0.000491960 seconds
Peer delay      : 0.004312567 seconds
Peer dispersion : 0.000000068 seconds
Response time   : 0.000037078 seconds
Jitter asymmetry: +0.00
NTP tests       : 111 111 1111
Interleaved     : No
Authenticated   : No
TX timestamping : Daemon
RX timestamping : Kernel
```

```
Total TX        : 15
Total RX        : 15
Total valid RX  : 15
```

Infine, `chronyc sources` restituirà informazioni sui server NTP utilizzati per sincronizzare il tempo:

```
$ chronyc sources
210 Number of sources = 0
MS Name/IP address         Stratum Poll Reach LastRx Last sample
===============================================================================
```

Al momento, questa macchina non ha fonti configurate. Possiamo aggiungere fonti da `pool.ntp.org` aprendo il file di configurazione di *chrony*. Esso si trova di solito in `/etc/chrony.conf`. Quando apriamo questo file, dovremmo vedere che alcuni server sono elencati di default:

```
210 Number of sources = 0
MS Name/IP address         Stratum Poll Reach LastRx Last sample
===============================================================================
# Most computers using chrony will send measurement requests to one or
# more 'NTP servers'.  You will probably find that your Internet Service
# Provider or company have one or more NTP servers that you can specify.
# Failing that, there are a lot of public NTP servers.  There is a list
# you can access at http://support.ntp.org/bin/view/Servers/WebHome or
# you can use servers from the 3.arch.pool.ntp.org project.

! server 0.arch.pool.ntp.org iburst iburst
! server 1.arch.pool.ntp.org iburst iburst
! server 2.arch.pool.ntp.org iburst iburst

! pool 3.arch.pool.ntp.org iburst
```

I server presenti nel file servono anche da guida alla sintassi quando ne inseriremo di nostri. Tuttavia, in questo caso rimuoveremo semplicemente i punti escalmativi `!` all'inizio di ogni linea, decommentando così queste linee e usando i server di default del progetto `pool.ntp.org`.

Inoltre, in questo file possiamo scegliere di cambiare la configurazione di default per quanto riguarda skew e drift, così come la posizione del driftfile e del keyfile.

Su questa macchina abbiamo bisogno di fare un'importante correzione iniziale del clock.

Sceglieremo di decommentare la seguente linea:

```
! makestep 1.0 3
```

Dopo aver apportato le modifiche al file di configurazione, riavvia il servizio `chronyd` e poi usa `chronyc makestep` per far passare manualmente l'orologio di sistema:

```
# chronyc makestep
200 OK
```

Poi usa il `chronyc tracking` come prima per verificare che le modifiche abbiano avuto luogo.

Esercizi Guidati

1. Inserisci il termine appropriato per ogni definizione:

Definizione	**Termine**
Un computer che condividerà l'orario di rete	
Distanza da un orologio di riferimento, in salti o passi	
Differenza tra l'ora di sistema e l'ora di rete	
Differenza tra l'ora di sistema e l'ora di rete dall'ultimo polling NTP	
Gruppo di server che forniscono l'orario di rete e condividono il carico tra di loro	

2. Specificate quale dei comandi usereste per produrre i seguenti valori:

Valore	`chronyc tracking`	`timedatectl show-timesync --all`	`ntpq -pn`	`chrony ntpdata`	`chronyc sources`
Jitter					
Drift					
Interval of Poll					
Offset					
Stratum					
IP Address of Provider					
Root Delay					

3. Stai impostando una rete aziendale che consiste in un server e diversi desktop Linux. Il server ha un indirizzo IP statico di 192.168.0.101. Decidi che il server si connetterà a `pool.ntp.org` e quindi fornirà l'orario NTP ai desktop. Descrivi la configurazione del server e dei desktop.

4. Una macchina Linux ha l'ora sbagliata. Descrivi i passi che faresti per risolvere i problemi di

NTP.

Esercizi Esplorativi

1. Indica le differenze tra SNTP e NTP.

SNTP	NTP

2. Perché un amministratore di sistema potrebbe scegliere di non usare `pool.ntp.org`?

3. Come potrebbe un amministratore di sistema scegliere di unirsi o altrimenti contribuire al progetto `pool.ntp.org`?

Sommario

In questa lezione abbiamo imparato:

- Che cos'è NTP e perché è importante.
- Configurazione del demone NTP dal progetto `pool.ntp.org`.
- Uso di `ntpq` per verificare la configurazione NTP.
- Uso di `chrony` come servizio NTP alternativo.

Comandi utilizzati in questa lezione:

`timedatectl show-timesync --all`
Visualizza informazioni SNTP se si usa `timedatectl`.

`ntpdate <address>`
Esegue un aggiornamento NTP manuale una tantum.

`ntpq -p`
Stampa una cronologia delle rchieste recenti di NTP. `-n` sostituirà le URL con gli indirizzi IP.

`chronyc tracking`
Visualizza lo stato di NTP se si usa *chrony*.

`chronyc ntpdata`
Visualizza le informazioni NTP sull'ultimo polling.

`chronyc sources`
Visualizza le informazioni sui provider NTP.

`chronyc makestep`
Esegue un aggiornamento NTP manuale una tantum se si usa chrony.

Risposte agli Esercizi Guidati

1. Inserisci il termine appropriato per ogni definizione:

Definizione	**Termine**
Un computer che condividerà l'orario di rete	Provider
Distanza da un orologio di riferimento, in salti o passi	Stratum
Differenza tra l'ora di sistema e l'ora di rete	Offset
Differenza tra l'ora di sistema e l'ora di rete dall'ultimo polling NTP	Jitter
Gruppo di server che forniscono l'orario di rete e condividono il carico tra loro	Pool

2. Specificate quale dei comandi usereste per produrre i seguenti valori:

Valore	`chronyc tracking`	`timedatectl show-timesync --all`	`ntpq -pn`	`chrony ntpdata`	`chronyc sources`
Jitter		X	X		
Drift					
Interval of Poll	X	X	X (colonna `when`)	X	X
Offset	X		X	X	
Stratum	X	X	X	X	X
IP Address of Provider		X	X	X	X
Root Delay	X			X	

3. State impostando una rete aziendale che consiste in un server e diversi desktop Linux. Il server ha un indirizzo IP statico di 192.168.0.101. Decidi che il server si connetterà a `pool.ntp.org` e quindi fornirà l'orario NTP ai desktop. Descrivi la configurazione del server e dei desktop.

   Assicurati che il server abbia un servizio ntpd in funzione, piuttosto che SNTP. Usa i pool `pool.ntp.org` nel file `/etc/ntp.conf` o `/etc/chrony.conf`. Per ogni client, specificare

`192.168.0.101` in ogni file `/etc/ntp.conf` o `/etc/chrony.conf`.

4. Un sistema Linux ha l'ora sbagliata. Descrivi i passi che fareste per risolvere i problemi di NTP.

   Per prima cosa, assicurati che il sistema sia connesso a Internet. Usa `ping` per verificarlo. Controlla che un servizio ntpd o SNTP sia in esecuzione usando `systemctl status ntpd` o `systemctl status systemd-timesyncd`. Potreste vedere messaggi di errore che forniscono informazioni utili. Infine, usa un comando come `ntpq -p` o `chrony tracking` per verificare se sono state fatte delle richieste. Se l'ora di sistema è drasticamente diversa dall'ora di rete, può essere che l'ora di sistema sia considerata "insana" e non verrà cambiata senza un intervento manuale. In questo caso, usa un comando della lezione precedente o un comando come `ntpdate pool.ntp.org` per eseguire una sincronizzazione ntp una tantum.

Risposte agli Esercizi Esplorativi

1. Indica le differenze tra SNTP e NTP.

SNTP	**NTP**
meno accurato	più accurato
richiede meno risorse	richiede più risorse
non può agire come fornitore di tempo	può agire come fornitore di tempo
solo piccole modifiche temporali	fa passi o slitta il tempo
richiede il tempo da una singola fonte	può monitorare più server NTP e utilizzare il fornitore ottimale

2. Perché un amministratore di sistema potrebbe scegliere di non usare `pool.ntp.org`?

   Da ntppool.org: *se è assolutamente cruciale avere un orario corretto, dovresti considerare un'alternativa. Allo stesso modo, se il tuo Internet provider ha un time server, si raccomanda di usare quello.*

3. Come potrebbe un amministratore di sistema scegliere di unirsi o altrimenti contribuire al progetto `pool.ntp.org`?

   Da www.ntppool.org: *il tuo server deve avere un indirizzo IP statico e una connessione internet permanente. L'indirizzo IP statico non deve cambiare affatto o almeno meno di una volta all'anno. Oltre a questo, i requisiti di banda sono modesti: 384 - 512 Kbit di banda. I server Stratum 3 o 4 sono i benvenuti.*



108.2 Logging di sistema

Obiettivi LPI di riferimento

LPIC-1 version 5.0, Exam 102, Objective 108.2

Peso

4

Aree di Conoscenza Chiave

- Configurazione di base di rsyslog.
- Comprensione di strutture standard, priorità e azioni.
- Interrogare il log di sistema.
- Filtrare i dati del log di sistema in base a criteri quali data, servizio o priorità.
- Configurare l'archiviazione persistente del journal di systemd e la sua dimensione.
- Eliminare i vecchi dati nel journal di systemd.
- Recuperare i dati del journal di systemd da un sistema di ripristino o da una copia del file system.
- Comprendere l'interazione di rsyslog con systemd-journald.
- Configurazione di logrotate.
- Conoscenza di syslog e syslog-ng.

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

- `/etc/rsyslog.conf`
- `/var/log/`
- `logger`
- `logrotate`

- `/etc/logrotate.conf`
- `/etc/logrotate.d/`
- `journalctl`
- `systemd-cat`
- `/etc/systemd/journald.conf`
- `/var/log/journal/`

 | Questa opera è concessa in licenza CC BY-NC-ND 4.0. |

108.2 Lezione 1

Certificazione:	LPIC-1
Versione:	5.0
Argomento:	108 Servizi Essenziali di Sistema
Obiettivo:	108.2 Logging di sistema
Lezione:	1 di 2

Introduzione

I *log* possono essere i migliori amici di un amministratore di sistema. I log sono file (di solito file di testo) dove tutti gli eventi di sistema e di rete sono cronologicamente registrati dal momento in cui il sistema viene avviato. La gamma di informazioni che si possono trovare nei log include virtualmente ogni aspetto del sistema: tentativi di autenticazione falliti, errori di programma e di servizio, host bloccati dal firewall e altri ancora. Come si può immaginare, i log rendono la vita degli amministratori di sistema molto più facile quando si parla di risoluzione dei problemi, controllo delle risorse, rilevamento di comportamenti anomali dei programmi e così via.

In questa lezione discuteremo uno dei più comuni ambienti di logging attualmente presenti nelle distribuzioni GNU/Linux: `rsyslog`. Studieremo i diversi tipi di log che esistono, dove sono memorizzati, quali informazioni includono e come queste informazioni possono essere ottenute e filtrate. Discuteremo anche di come i log possono essere conservati in server centralizzati attraverso reti IP, la rotazione dei log e il *ring buffer* del kernel.

Il Log di Sistema

Nel momento in cui il kernel e i diversi processi nel sistema iniziano a funzionare e a comunicare

tra loro, vengono generate molte informazioni sotto forma di messaggi che sono — per la maggior parte — inviati ai log.

Senza i log, la ricerca di un evento accaduto su un server genererebbe di sicuro grandi mal di testa agli amministratori di sistema: da qui l'importanza di avere un modo standardizzato e centralizzato di tenere traccia di qualsiasi evento di sistema. I registri sono determinanti e rivelatori quando si tratta di risoluzione dei problemi e sicurezza e sono fonti di dati affidabili per comprendere le statistiche di sistema e fare previsioni di tendenza.

Lasciando da parte `systemd-journald` (che discuteremo nella prossima lezione), il logging è stato tradizionalmente gestito da tre principali servizi dedicati: `syslog`, `syslog-ng` (syslog new generation) e `rsyslog` ("the rocket-fast system for log processing"). `rsyslog` ha portato importanti miglioramenti (come il supporto RELP) ed è diventato la scelta più popolare al giorno d'oggi. Ognuno di questi servizi raccoglie messaggi da altri servizi e programmi e li memorizza in file di log, tipicamente sotto `/var/log`. Tuttavia, alcuni servizi si prendono cura dei propri log (prendi come esempio il server web Apache HTTPD o il sistema di stampa CUPS). Allo stesso modo, il kernel Linux usa un ring buffer *in-memory* per memorizzare i suoi messaggi di log.

NOTE `RELP` sta per *Reliable Event Logging Protocol* ed estende la funzionalità del protocollo syslog per fornire una consegna affidabile dei messaggi.

Dato che `rsyslog` è diventato lo strumento di logging *de facto* in tutte le principali distro, ci concentreremo su di esso per questa lezione. `rsyslog` utilizza un modello client-server. Il client e il server possono vivere sullo stesso host o su macchine diverse. I messaggi sono inviati e ricevuti in un particolare formato e possono essere conservati in server centralizzati `rsyslog` attraverso reti IP. Il demone di rsyslog — `rsyslogd` — lavora insieme a `klogd` (che gestisce i messaggi del kernel). Nelle prossime sezioni verranno discussi `rsyslog` e la sua infrastruttura di logging.

NOTE Un demone è un servizio che funziona in background. Si noti la `d` finale nei nomi dei demoni: `klogd` o `rsyslogd`.

Tipi di Log

Poiché i log sono dati *variabili*, normalmente si trovano in `/var/log`. Approssimativamente, possono essere classificati in *system logs* e *service or program logs*.

Vediamo alcuni log di sistema e le informazioni che conservano:

`/var/log/auth.log`

Attività relative ai processi di autenticazione: utenti registrati, notifiche `sudo`, *cron job*, tentativi di login falliti, ecc.

`/var/log/syslog`

Un file centralizzato per praticamente tutti i log catturati da `rsyslogd`. Poiché include così tante informazioni, i log sono distribuiti in altri file secondo la configurazione fornita in `/etc/rsyslog.conf`.

`/var/log/debug`

Informazioni di debug dai programmi.

`/var/log/kern.log`

Messaggi del kernel.

`/var/log/messages`

Messaggi informativi che non sono relativi al kernel ma ad altri servizi. È anche la destinazione predefinita del log del client remoto in un'implementazione centralizzata del server di log.

`/var/log/daemon.log`

Informazioni relative a demoni o servizi in esecuzione in background.

`/var/log/mail.log`

Informazioni relative al server di posta elettronica, per esempio postfix.

`/var/log/Xorg.0.log`

Informazioni relative all'ambiente grafico Xorg.

`/var/run/utmp` and `/var/log/wtmp`

Accessi riusciti

`/var/log/btmp`

Tentativi di login falliti, per esempio attacco brute force via ssh.

`/var/log/faillog`

Tentativi di autenticazione falliti.

`/var/log/lastlog`

Data e ora degli ultimi accessi dell'utente.

Vediamo invece adesso alcuni esempi di log di servizio:

`/var/log/cups/`

Directory per i log del *Common Unix Printing System*. Include comunemente i seguenti file di log predefiniti: `error_log`, `page_log` e `access_log`.

`/var/log/apache2/` or `/var/log/httpd`

Directory per i log di *Apache Web Server*. Di solito include i seguenti file di log predefiniti: `access.log`, `error_log`, e `other_vhosts_access.log`.

`/var/log/mysql`

Directory per i log di *MySQL - Relational Database Management System*. Include comunemente i seguenti file di log predefiniti: `error_log`, `mysql.log` e `mysql-slow.log`.

`/var/log/samba/`

Directory per i log del protocollo *Session Message Block* (SMB). Di solito include i seguenti file di log predefiniti: `log.`, `log.nmbd` e `log.smbd`.

NOTE

Il nome esatto e il contenuto dei file di log possono differire nelle varie distribuzioni Linux. Esistono anche log particolari per distribuzioni specifiche come `/var/log/dpkg.log` (contenente informazioni relative ai pacchetti `dpkg`) in Debian GNU/Linux e le sue derivate.

Leggere i Log

Per leggere i file di log, prima devi assicurarti di essere l'utente root o di avere i permessi di lettura sul file. Puoi usare una varietà di utility come:

`less` o `more`

Paginatori che permettono di visualizzare e scorrere una pagina alla volta:

```
root@debian:~# less /var/log/auth.log
Sep 12 18:47:56 debian sshd[441]: Received SIGHUP; restarting.
Sep 12 18:47:56 debian sshd[441]: Server listening on 0.0.0.0 port 22.
Sep 12 18:47:56 debian sshd[441]: Server listening on :: port 22.
Sep 12 18:47:56 debian sshd[441]: Received SIGHUP; restarting.
Sep 12 18:47:56 debian sshd[441]: Server listening on 0.0.0.0 port 22.
Sep 12 18:47:56 debian sshd[441]: Server listening on :: port 22.
Sep 12 18:49:46 debian sshd[905]: Accepted password for carol from 192.168.1.65 port
44296 ssh2
Sep 12 18:49:46 debian sshd[905]: pam_unix(sshd:session): session opened for user carol
by (uid=0)
Sep 12 18:49:46 debian systemd-logind[331]: New session 2 of user carol.
Sep 12 18:49:46 debian systemd: pam_unix(systemd-user:session): session opened for user
carol by (uid=0)
(...)
```

`zless` o `zmore`

Lo stesso di `less` e `more`, ma usato per i log che sono compressi con `gzip` (una funzione comune di `logrotate`):

```
root@debian:~# zless /var/log/auth.log.3.gz
Aug 19 20:05:57 debian sudo:    carol : TTY=pts/0 ; PWD=/home/carol ; USER=root ;
COMMAND=/sbin/shutdown -h now
Aug 19 20:05:57 debian sudo: pam_unix(sudo:session): session opened for user root by
carol(uid=0)
Aug 19 20:05:57 debian lightdm: pam_unix(lightdm-greeter:session): session closed for
user lightdm
Aug 19 23:50:49 debian systemd-logind[333]: Watching system buttons on /dev/input/event2
(Power Button)
Aug 19 23:50:49 debian systemd-logind[333]: Watching system buttons on /dev/input/event3
(Sleep Button)
Aug 19 23:50:49 debian systemd-logind[333]: Watching system buttons on /dev/input/event4
(Video Bus)
Aug 19 23:50:49 debian systemd-logind[333]: New seat seat0.
Aug 19 23:50:49 debian sshd[409]: Server listening on 0.0.0.0 port 22.
(...)
```

`tail`

Visualizza le ultime righe di un file (il valore predefinito è 10 righe). La potenza di tail risiede - in larga misura - nell'opzione `-f`, che mostrerà dinamicamente le nuove righe man mano che vengono aggiunte al file:

```
root@suse-server:~# tail -f /var/log/messages
2019-09-14T13:57:28.962780+02:00 suse-server sudo: pam_unix(sudo:session): session closed
for user root
2019-09-14T13:57:38.038298+02:00 suse-server sudo:    carol : TTY=pts/0 ; PWD=/home/carol
; USER=root ; COMMAND=/usr/bin/tail -f /var/log/messages
2019-09-14T13:57:38.039927+02:00 suse-server sudo: pam_unix(sudo:session): session opened
for user root by carol(uid=0)
2019-09-14T14:07:22+02:00 debian carol: appending new message from client to remote
server...
```

`head`

Visualizza le prime righe di un file (il valore predefinito è 10 righe):

```
root@suse-server:~# head -5 /var/log/mail
```

```
2019-06-29T11:47:59.219806+02:00 suse-server postfix/postfix-script[1732]: the Postfix
mail system is not running
2019-06-29T11:48:01.355361+02:00 suse-server postfix/postfix-script[1925]: starting the
Postfix mail system
2019-06-29T11:48:01.391128+02:00 suse-server postfix/master[1930]: daemon started --
version 3.3.1, configuration /etc/postfix
2019-06-29T11:55:39.247462+02:00 suse-server postfix/postfix-script[3364]: stopping the
Postfix mail system
2019-06-29T11:55:39.249375+02:00 suse-server postfix/master[1930]: terminating on signal
15
```

`grep`

Utilità di filtraggio che permette di cercare stringhe specifiche:

```
root@debian:~# grep "dhclient" /var/log/syslog
Sep 13 11:58:48 debian dhclient[448]: DHCPREQUEST of 192.168.1.4 on enp0s3 to 192.168.1.1
port 67
Sep 13 11:58:49 debian dhclient[448]: DHCPACK of 192.168.1.4 from 192.168.1.1
Sep 13 11:58:49 debian dhclient[448]: bound to 192.168.1.4 -- renewal in 1368 seconds.
(...)
```

Come avrai notato, l'output viene stampato nel seguente formato:

- Riferimento temporale
- Nome dell'host da cui è partito il messaggio
- Nome del programma/servizio che ha generato il messaggio
- Il PID del programma che ha generato il messaggio
- Descrizione dell'azione che ha avuto luogo

Ci sono alcuni esempi in cui i log non sono testo, ma file binari e — di conseguenza — è necessario usare comandi speciali per analizzarli:

`/var/log/wtmp`

Utilizza `who` (o `w`):

```
root@debian:~# who
root     pts/0        2020-09-14 13:05 (192.168.1.75)
root     pts/1        2020-09-14 13:43 (192.168.1.75)
```

`/var/log/btmp`

Utilizza `utmpdump` o `last -f`:

```
root@debian:~# utmpdump /var/log/btmp
Utmp dump of /var/log/btmp
[6] [01287] [    ] [dave     ] [ssh:notty   ] [192.168.1.75        ] [192.168.1.75   ]
[2019-09-07T19:33:32,000000+0000]
```

`/var/log/faillog`

Utilizza `faillog`:

```
root@debian:~# faillog -a | less
Login       Failures Maximum Latest                   On

root            0        0   01/01/70 01:00:00 +0100
daemon          0        0   01/01/70 01:00:00 +0100
bin             0        0   01/01/70 01:00:00 +0100
sys             0        0   01/01/70 01:00:00 +0100
sync            0        0   01/01/70 01:00:00 +0100
games           0        0   01/01/70 01:00:00 +0100
man             0        0   01/01/70 01:00:00 +0100
lp              0        0   01/01/70 01:00:00 +0100
mail            0        0   01/01/70 01:00:00 +0100
(...)
```

`/var/log/lastlog`

Utilizza `lastlog`:

```
root@debian:~# lastlog | less
Username         Port     From             Latest
root                                       Never logged in
daemon                                     Never logged in
bin                                        Never logged in
sys                                        Never logged in
(...)
sync                                       Never logged in
avahi                                      Never logged in
colord                                     Never logged in
saned                                      Never logged in
hplip                                      Never logged in
carol            pts/1    192.168.1.75     Sat Sep 14 13:43:06 +0200 2019
```

```
dave             pts/3    192.168.1.75     Mon Sep  2 14:22:08 +0200 2019
```

NOTE Ci sono anche strumenti grafici per leggere i file di log, per esempio: `gnome-logs` e `KSystemLog`.

Come i messaggi vengono trasformati in registri

Il seguente processo illustra come un messaggio venga scritto in un file di log:

1. Applicazioni, servizi e kernel scrivono messaggi in file speciali (*socket* e buffer di memoria), per esempio `/dev/log` o `/dev/kmsg`.
2. `rsyslogd` ottiene le informazioni dai socket o dai buffer di memoria.
3. A seconda delle regole trovate in `/etc/rsyslog.conf` e/o nei file in `/etc/ryslog.d/`, `rsyslogd` sposta le informazioni nel file di log corrispondente (tipicamente in `/var/log`).

NOTE Un *socket* è un file speciale usato per trasferire informazioni tra diversi processi. Per elencare tutti i socket sul tuo sistema, puoi usare il comando `systemctl list-sockets --all`.

Strutture, Priorità e Azioni

Il file di configurazione di `rsyslog` è `/etc/rsylog.conf` (in alcune distribuzioni puoi anche trovare i file di configurazione in `/etc/rsyslog.d/`). È normalmente diviso in tre sezioni: `MODULES`, `GLOBAL DIRECTIVES` e `RULES`. Diamo loro un'occhiata esplorando il file `rsyslog.conf` nel nostro host Debian GNU/Linux 10 (buster). Puoi usare `sudo less /etc/rsyslog.conf`.

`MODULES` include il supporto del modulo per la registrazione, la gestione dei messaggi e la ricezione dei log via UDP/TCP:

```
#################
#### MODULES ####
#################

module(load="imuxsock") # provides support for local system logging
module(load="imklog")   # provides kernel logging support
#module(load="immark")  # provides --MARK-- message capability

# provides UDP syslog reception
#module(load="imudp")
#input(type="imudp" port="514")
```

```
# provides TCP syslog reception
#module(load="imtcp")
#input(type="imtcp" port="514")
```

`GLOBAL DIRECTIVES` ci permettono di configurare un certo numero di cose come i log e i permessi delle directory di log:

```
###########################
#### GLOBAL DIRECTIVES ####
###########################

#
# Use traditional timestamp format.
# To enable high precision timestamps, comment out the following line.
#
$ActionFileDefaultTemplate RSYSLOG_TraditionalFileFormat

#
# Set the default permissions for all log files.
#
$FileOwner root
$FileGroup adm
$FileCreateMode 0640
$DirCreateMode 0755
$Umask 0022

#
# Where to place spool and state files
#
$WorkDirectory /var/spool/rsyslog

#
# Include all config files in /etc/rsyslog.d/
#
$IncludeConfig /etc/rsyslog.d/*.conf
```

`RULES` è dove entrano in gioco *strutture (facilities)*, *priorità (priorities)* e *azioni (actions)*. Le impostazioni in questa sezione dicono al demone di logging di filtrare i messaggi secondo certe regole e di registrarli o inviarli dove richiesto. Per capire queste regole, dovremmo prima spiegare i concetti di strutture e priorità di `rsyslog`. A ogni messaggio di log viene dato un numero di *facility* e una parola chiave associati al sottosistema interno di Linux che produce il messaggio:

Numero	Parola Chiave	Descrizione
`0`	`kern`	Linux kernel messages
`1`	`user`	User-level messages
`2`	`mail`	Mail system
`3`	`daemon`	System daemons
`4`	`auth`, `authpriv`	Security/Authorization messages
`5`	`syslog`	syslogd messages
`6`	`lpr`	Line printer subsystem
`7`	`news`	Network news subsystem
`8`	`uucp`	UUCP (Unix-to-Unix Copy Protocol) subsystem
`9`	`cron`	Clock daemon
`10`	`auth`, `authpriv`	Security/Authorization messages
`11`	`ftp`	FTP (File Transfer Protocol) daemon
`12`	`ntp`	NTP (Network Time Protocol) daemon
`13`	`security`	Log audit
`14`	`console`	Log alert
`15`	`cron`	Clock daemon
`16 - 23`	`local0` through `local7`	Local use 0 - 7

Inoltre, a ogni messaggio viene assegnato un livello di *priority*:

Codice	Gravità	Parola chiave	Descrizione
`0`	Emergency	`emerg`, `panic`	System is unusable
`1`	Alert	`alert`	Action must be taken immediately
`2`	Critical	`crit`	Critical conditions
`3`	Error	`err`, `error`	Error conditions

Codice	Gravità	Parola chiave	Descrizione
4	Warning	`warn`, `warning`	Warning conditions
5	Notice	`notice`	Normal but significant condition
6	Informational	`info`	Informational messages
7	Debug	`debug`	Debug-level messages

Ecco un estratto di `rsyslog.conf` dal nostro sistema Debian GNU/Linux 10 (buster) che include alcune regole di esempio:

```
###############
#### RULES ####
###############

# First some standard log files.  Log by facility.
#
auth,authpriv.*                 /var/log/auth.log
*.*;auth,authpriv.none          -/var/log/syslog
#cron.*                         /var/log/cron.log
daemon.*                        -/var/log/daemon.log
kern.*                          -/var/log/kern.log
lpr.*                           -/var/log/lpr.log
mail.*                          -/var/log/mail.log
user.*                          -/var/log/user.log

#
# Logging for the mail system.  Split it up so that
# it is easy to write scripts to parse these files.
#
mail.info                       -/var/log/mail.info
mail.warn                       -/var/log/mail.warn
mail.err                        /var/log/mail.err

#
# Some "catch-all" log files.
#
*.=debug;\
        auth,authpriv.none;\
    news.none;mail.none     -/var/log/debug
*.=info;*.=notice;*.=warn;\
```

```
    auth,authpriv.none;\
    cron,daemon.none;\
    mail,news.none          -/var/log/messages
```

Il formato della regola è il seguente: `<facility>.<priority> <action>`

Il selettore `<facility>.<priority>` filtra i messaggi da abbinare. I livelli di priorità sono *gerarchicamente inclusivi*, il che significa che rsyslog troverà una corrispondenza per i messaggi dalla priorità specificata o superiore. `<action>` mostra quale azione intraprendere (dove inviare il messaggio di log). Ecco alcuni esempi per chiarezza:

```
auth,authpriv.*                 /var/log/auth.log
```

Indipendentemente dalla loro priorità (`*`), tutti i messaggi dalle strutture `auth` o `authpriv` saranno inviati a `/var/log/auth.log`.

```
*.*;auth,authpriv.none          -/var/log/syslog
```

Tutti i messaggi — indipendentemente dalla loro priorità (`*`) — da tutte le *facility* (`*`) — scartando quelli da `auth` o `authpriv` (da qui il suffisso `.none`) — saranno scritti in `/var/log/syslog` (il segno meno (`-`) prima del percorso previene eccessive scritture su disco). Nota il punto e virgola (`;`) per dividere il selettore e la virgola (`,`) per concatenare due strutture nella stessa regola (`auth,authpriv`).

```
mail.err                        /var/log/mail.err
```

I messaggi della *facility* `mail` con un livello di priority di `error` o superiore (`critical`, `alert` o `emergency`) saranno inviati a `/var/log/mail.err`.

```
*.=debug;\
        auth,authpriv.none;\
    news.none;mail.none     -/var/log/debug
```

I messaggi provenienti da tutte le *facility* con *priority* `debug` e nessun'altra (`=`) saranno scritti in `/var/log/debug` — escludendo qualsiasi messaggio proveniente dalle *facility* `auth`, `authpriv`, `news` e `mail` (nota la sintassi: `;\`).

Inserimenti Manuali nel Registro di Sistema: `logger`

Il comando `logger` è utile per lo scripting di shell o per scopi di test. `logger` aggiungerà ogni messaggio che riceve a `/var/log/syslog` (o a `/var/log/messages` quando registra su un server di log centrale remoto, come vedrai più avanti in questa lezione):

```
carol@debian:~$ logger this comment goes into "/var/log/syslog"
```

Per visualizzare l'ultima riga in `/var/log/syslog`, usa il comando `tail` con l'opzione `-1`:

```
root@debian:~# tail -1 /var/log/syslog
Sep 17 17:55:33 debian carol: this comment goes into /var/log/syslog
```

`rsyslog` come Server di Log Centralizzato

Per spiegare questo argomento aggiungeremo un nuovo host alla nostra configurazione. Il layout è il seguente:

Ruolo	Hostname	OS	Indirizzo IP
Central Log Server	`suse-server`	openSUSE Leap 15.1	192.168.1.6
Client	`debian`	Debian GNU/Linux 10 (buster)	192.168.1.4

Iniziamo a configurare il server. Prima di tutto, assicuriamoci che `rsyslog` sia attivo e funzionante:

```
root@suse-server:~# systemctl status rsyslog
 rsyslog.service - System Logging Service
   Loaded: loaded (/usr/lib/systemd/system/rsyslog.service; enabled; vendor preset: enabled)
   Active: active (running) since Thu 2019-09-17 18:45:58 CEST; 7min ago
     Docs: man:rsyslogd(8)
           http://www.rsyslog.com/doc/
 Main PID: 832 (rsyslogd)
    Tasks: 5 (limit: 4915)
   CGroup: /system.slice/rsyslog.service
           └─832 /usr/sbin/rsyslogd -n -iNONE
```

openSUSE dispone di un file di configurazione dedicato alla registrazione remota: `/etc/rsyslog.d/remote.conf`. Abilitiamo la ricezione dei messaggi dai client (host remoti) via

TCP. Dobbiamo decommentare le linee che caricano il modulo e avviano il server TCP sulla porta 514:

```
# ######### Receiving Messages from Remote Hosts ##########
# TCP Syslog Server:
# provides TCP syslog reception and GSS-API (if compiled to support it)
$ModLoad imtcp.so  # load module
##$UDPServerAddress 10.10.0.1  # force to listen on this IP only
$InputTCPServerRun 514  # Starts a TCP server on selected port

# UDP Syslog Server:
#$ModLoad imudp.so  # provides UDP syslog reception
##$UDPServerAddress 10.10.0.1  # force to listen on this IP only
#$UDPServerRun 514  # start a UDP syslog server at standard port 514
```

Una volta fatto questo, dobbiamo riavviare il servizio *rsyslog* e controllare che il server sia in ascolto sulla porta 514:

```
root@suse-server:~# systemctl restart rsyslog
root@suse-server:~# netstat -nltp | grep 514
[sudo] password for root:
tcp        0      0 0.0.0.0:514             0.0.0.0:*               LISTEN
2263/rsyslogd
tcp6       0      0 :::514                  :::*                    LISTEN
2263/rsyslogd
```

Successivamente, dovremmo aprire le porte nel firewall e ricaricare la configurazione:

```
root@suse-server:~# firewall-cmd --permanent --add-port 514/tcp
success
root@suse-server:~# firewall-cmd --reload
success
```

NOTE Con l'arrivo di openSUSE Leap 15.0, `firewalld` ha sostituito completamente il classico `SuSEFirewall2`.

Modelli e Condizioni di Filtro

Per default, i log del client saranno scritti nel file `/var/log/messages` del server — insieme a quelli del server stesso. Tuttavia, creeremo un *modello (template)* e una *condizioni di filtro* per fare in modo che i log del nostro client siano immagazzinati in cartelle proprie ben definite. Per fare

ciò, aggiungeremo quanto segue a `/etc/rsyslog.conf` (o `/etc/rsyslog.d/remote.conf`):

```
$template RemoteLogs,"/var/log/remotehosts/%HOSTNAME%/%$NOW%.%syslogseverity-text%.log"
if $FROMHOST-IP=='192.168.1.4' then ?RemoteLogs
& stop
```

Template

Il modello corrisponde alla prima riga e permette di specificare un formato per i nomi dei log utilizzando la generazione dinamica dei nomi dei file. Un template consiste in:

- Direttiva template (`$template`)
- Nome del template (`RemoteLogs`)
- Testo del template (`"/var/log/remotehosts/%HOSTNAME%/%$NOW%.%syslogseverity-text%.log"`)
- Opzioni (facoltativo)

Il nostro modello si chiama `RemoteLogs` e il suo testo consiste nel percorso `/var/log`. Tutti i log del nostro host remoto andranno nella directory `remotehosts`, dove verrà creata una sottodirectory basata sull'hostname della macchina (`%HOSTNAME%`). Ogni nome di file in questa directory sarà composto dalla data (`%$NOW%`), la gravità (o priorità) del messaggio in formato testo (`%syslogseverity-text%`) e il suffisso `.log`. Le parole tra i segni di percentuale sono *proprietà* e ti permettono di accedere al contenuto del messaggio di log (data, priorità, ecc.). Un messaggio `syslog` ha un certo numero di proprietà ben definite che possono essere usate nei modelli. Queste proprietà sono accessibili — e possono essere modificate — dal cosiddetto *property replacer* che implica la loro scrittura tra i segni di percentuale.

Condizioni di Filtro

Le due righe rimanenti corrispondono alla condizione del filtro e alla sua azione associata:

- Filtro basato sull'espressione (`if $FROMHOST-IP=='192.168.1.4'`)
- Azione (`then ?RemoteLogs`, `& stop`)

La prima linea controlla l'indirizzo IP dell'host remoto che invia il log e — se è uguale a quello del nostro client Debian — applica il modello `RemoteLogs`. L'ultima linea (`& stop`) garantisce che i messaggi non vengano inviati contemporaneamente a `/var/log/messages` (ma solo ai file nella directory `/var/log/remotehosts`).

NOTE Per saperne di più su modelli, proprietà e regole, puoi consultare la pagina di manuale di `rsyslog.conf`.

Con la configurazione aggiornata, riavviamo nuovamente `rsyslog` e confermiamo che non c'è ancora una directory `remotehosts` in `/var/log`:

```
root@suse-server:~# systemctl restart rsyslog
root@suse-server:~# ls /var/log/
acpid             chrony     localmessages   pbl.log          Xorg.0.log
alternatives.log  cups       mail            pk_backend_zypp  Xorg.0.log.old
apparmor          firebird   mail.err        samba            YaST2
audit             firewall   mail.info       snapper.log      zypp
boot.log          firewalld  mail.warn       tallylog         zypper.log
boot.msg          krb5       messages        tuned
boot.omsg         lastlog    mysql           warn
btmp              lightdm    NetworkManager  wtmp
```

Il server è stato ora configurato. Ora configureremo il client.

Di nuovo, dobbiamo assicurarci che `rsyslog` sia installato e funzionante:

```
root@debian:~# sudo systemctl status rsyslog
 rsyslog.service - System Logging Service
   Loaded: loaded (/lib/systemd/system/rsyslog.service; enabled; vendor preset:
   Active: active (running) since Thu 2019-09-17 18:47:54 CEST; 7min ago
     Docs: man:rsyslogd(8)
           http://www.rsyslog.com/doc/
 Main PID: 351 (rsyslogd)
    Tasks: 4 (limit: 4915)
   CGroup: /system.slice/rsyslog.service
           └─351 /usr/sbin/rsyslogd -n
```

Nel nostro ambiente di esempio abbiamo implementato la risoluzione del nome sul client aggiungendo la linea `192.168.1.6 suse-server` a `/etc/hosts`. Così, possiamo fare riferimento al server sia per nome (`suse-server`) sia per indirizzo IP (`192.168.1.6`).

Il nostro client Debian non ha un file `remote.conf` in `/etc/rsyslog.d/`, quindi applicheremo le nostre configurazioni in `/etc/rsyslog.conf`. Scriveremo la seguente linea alla fine del file:

```
*.* @@suse-server:514
```

Infine, riavviamo `rsyslog`.

```
root@debian:~# systemctl restart rsyslog
```

Ora, torniamo alla nostra macchina `suse-server` e controlliamo l'esistenza di `remotehosts` in `/var/log`:

```
root@suse-server:~# ls /var/log/remotehosts/debian/
2019-09-17.info.log  2019-09-17.notice.log
```

Abbiamo già due log in `/var/log/remotehosts` come descritto nel nostro modello. Per completare questa sezione eseguiamo `tail -f 2019-09-17.notice.log` su `suse-server` mentre inviamo *manualmente* un log dal nostro client Debian e confermiamo che i messaggi vengono aggiunti al file di log come previsto (l'opzione `-t` fornisce un tag per il nostro messaggio):

```
root@suse-server:~# tail -f /var/log/remotehosts/debian/2019-09-17.notice.log
2019-09-17T20:57:42+02:00 debian dbus[323]: [system] Successfully activated service
'org.freedesktop.nm_dispatcher'
2019-09-17T21:01:41+02:00 debian anacron[1766]: Anacron 2.3 started on 2019-09-17
2019-09-17T21:01:41+02:00 debian anacron[1766]: Normal exit (0 jobs run)
```

```
carol@debian:~$ logger -t DEBIAN-CLIENT Hi from 192.168.1.4
```

```
root@suse-server:~# tail -f /var/log/remotehosts/debian/2019-09-17.notice.log
2019-09-17T20:57:42+02:00 debian dbus[323]: [system] Successfully activated service
'org.freedesktop.nm_dispatcher'
2019-09-17T21:01:41+02:00 debian anacron[1766]: Anacron 2.3 started on 2019-09-17
2019-09-17T21:01:41+02:00 debian anacron[1766]: Normal exit (0 jobs run)
2019-09-17T21:04:21+02:00 debian DEBIAN-CLIENT: Hi from 192.168.1.4
```

Il Meccanismo di Rotazione dei Log

I Log vengono ruotati regolarmente, il che serve a due scopi principali:

- Evitare che i vecchi file di log usino su disco più spazio del necessario.
- Mantenere i log a una lunghezza gestibile per facilitare la consultazione.

L'utility incaricata della rotazione dei log è `logrotate` e il suo lavoro include azioni come spostare i file di log in un nuovo nome, archiviarli e/o comprimerli, a volte inviarli via email all'amministratore di sistema e infine cancellarli quando diventano vecchi. Ci sono varie

convenzioni per nominare questi file di log ruotati (aggiungendo un suffisso con la data al nome del file, per esempio); tuttavia, aggiungere semplicemente un suffisso con un numero intero è la pratica più comune:

```
root@debian:~# ls /var/log/messages*
/var/log/messages  /var/log/messages.1  /var/log/messages.2.gz  /var/log/messages.3.gz
/var/log/messages.4.gz
```

Spieghiamo ora che cosa succederà nella prossima rotazione del registro:

1. `messages.4.gz` sarà cancellato e perso definitivamente.
2. Il contenuto di `messages.3.gz` sarà spostato in `messages.4.gz`.
3. Il contenuto di `messages.2.gz` verrà spostato in `messages.3.gz`.
4. Il contenuto di `messages.1` sarà spostato in `messages.2.gz`.
5. Il contenuto di `messages` sarà spostato in `messages.1` e `messages` sarà vuoto e pronto a registrare nuove voci di log.

Nota come — secondo le direttive `logrotate` che vedrai tra poco — i tre file di log più vecchi siano compressi, mentre i due più recenti no. Inoltre, conserveremo i log delle ultime 4-5 settimane. Per leggere i messaggi vecchi di 1 settimana, consulteremo `messages.1` (e così via).

`logrotate` viene eseguito come processo automatico o *cron job* giornaliero attraverso lo script `/etc/cron.daily/logrotate` e legge il file di configurazione `/etc/logrotate.conf`. Questo file include alcune opzioni globali ed è ben commentato con ogni opzione introdotta da una breve spiegazione del suo scopo:

```
carol@debian:~$ sudo less /etc/logrotate.conf
# see "man logrotate" for details
# rotate log files weekly
weekly

# keep 4 weeks worth of backlogs
rotate 4

# create new (empty) log files after rotating old ones
create

# uncomment this if you want your log files compressed
#compress
```

```
# packages drop log rotation information into this directory
include /etc/logrotate.d

(...)
```

Come puoi vedere sono inclusi anche i file di configurazione in `/etc/logrotate.d` per pacchetti specifici. Questi file contengono — per la maggior parte — definizioni locali e specificano i file di log da ruotare (ricorda, le definizioni locali hanno la precedenza su quelle globali, e le definizioni successive sovrascrivono quelle precedenti). Quello che segue è un estratto di una definizione in `/etc/logrotate.d/rsyslog`:

```
/var/log/messages
{
        rotate 4
        weekly
        missingok
        notifempty
        compress
        delaycompress
        sharedscripts
        postrotate
                invoke-rc.d rsyslog rotate > /dev/null
        endscript
}
```

Come puoi vedere, ogni direttiva è separata dal suo valore da spazi vuoti e/o da un segno opzionale di uguale (=). Le linee tra `postrotate` e `endscript` devono però apparire su linee a sé stanti. La spiegazione è la seguente:

`rotate 4`

Conserva 4 settimane di log.

`weekly`

Ruota i file di registro settimanalmente.

`missingok`

Non emette un messaggio di errore se il file di log è mancante; passa semplicemente a quello successivo.

`notifempty`

Non ruota il registro se è vuoto.

`compress`

Comprime i file di log con `gzip` (predefinito).

`delaycompress`

Posticipa la compressione del file di log precedente al prossimo ciclo di rotazione (efficace solo se usato in combinazione con *compress*). È utile quando non si può dire a un programma di chiudere il suo file di log e quindi potrebbe continuare a scrivere sul file di log precedente per un certo tempo.

`sharedscripts`

Relativo agli script *prerotate* e *postrotate*. Per evitare che uno script venga eseguito più volte, esegue gli script solo una volta indipendentemente da quanti file di log corrispondono a un dato schema (per esempio `/var/log/mail/*`). Gli script non verranno eseguiti se nessuno dei log nello schema richiede la rotazione. Inoltre, se gli script escono con errori, qualsiasi azione successiva non verrà eseguita per nessun log.

`postrotate`

Indica l'inizio di uno script *postrotate*.

`invoke-rc.d rsyslog rotate > /dev/null`

Usa `/bin/sh` per eseguire `invoke-rc.d rsyslog rotate > /dev/null` dopo la rotazione dei log

`endscript`

Indica la fine dello script *postrotate*.

NOTE Per una lista completa di direttive e spiegazioni, consultare la pagina di manuale di `logrotate.conf`.

Il Kernel Ring Buffer

Poiché il kernel genera diversi messaggi prima che `rsyslogd` sia disponibile all'avvio, è necessario un meccanismo per registrare questi messaggi. È qui che entra in gioco il *kernel ring buffer*. È una struttura dati a dimensione fissa e — quindi — man mano che cresce con nuovi messaggi, i più vecchi spariranno.

Il comando `dmesg` visualizza il ring buffer del kernel. A causa della dimensione del buffer, questo comando è normalmente usato in combinazione con l'utilità di filtraggio del testo `grep`. Per esempio, per cercare i messaggi relativi ai dispositivi Universal Serial Bus:

```
root@debian:~# dmesg | grep "usb"
```

```
[    1.241182] usbcore: registered new interface driver usbfs
[    1.241188] usbcore: registered new interface driver hub
[    1.250968] usbcore: registered new device driver usb
[    1.339754] usb usb1: New USB device found, idVendor=1d6b, idProduct=0001, bcdDevice=
4.19
[    1.339756] usb usb1: New USB device strings: Mfr=3, Product=2, SerialNumber=1
(...)
```

Esercizi Guidati

1. Quali utility/comandi usereste nei seguenti scenari:

Scopo e file di log	**Utility**
Leggere `/var/log/syslog.7.gz`	
Leggere `/var/log/syslog`	
Cercare la parola `renewal` in `/var/log/syslog`	
Leggere `/var/log/faillog`	
Leggere `/var/log/syslog` dinamicamente	

2. Riorganizza le seguenti voci di log in modo che rappresentino un messaggio di log valido con la struttura appropriata:
   - `debian-server`
   - `sshd`
   - `[515]:`
   - `Sep 13 21:47:56`
   - `Server listening on 0.0.0.0 port 22`

   L'ordine corretto è:

3. Quali regole aggiungereste a `/etc/rsyslog.conf` per realizzare ciascuna delle seguenti:
   - Inviare tutti i messaggi dalla *facility* `mail` e una priorità/severità di `crit` (e superiore) a `/var/log/mail.crit`:
   - Invia tutti i messaggi della *facility* `mail` con priorità `alert` e `emergency` a `/var/log/mail.urgent`:
   - Eccetto quelli provenienti dai servizi `cron` e `ntp`, invia tutti i messaggi — indipendentemente dalla loro *facility* e dalla priorità — a `/var/log/allmessages`:

- Con tutte le impostazioni necessarie configurate prima, invia tutti i messaggi dalla *facility* `mail` a un host remoto il cui indirizzo IP è `192.168.1.88` usando TCP e specificando la porta predefinita:

- Indipendentemente dalla loro *facility*, inviare tutti i messaggi con la priorità `warning` (solo con la priorità `warning`) a `/var/log/warnings` evitando un'eccessiva scrittura sul disco:

4. Considera la seguente parte di `/etc/logrotate.d/samba` e spiega le diverse opzioni:

```
carol@debian:~$ sudo head -n 11 /etc/logrotate.d/samba
/var/log/samba/log.smbd {
        weekly
        missingok
        rotate 7
        postrotate
                [ ! -f /var/run/samba/smbd.pid ] || /etc/init.d/smbd reload > /dev/null
        endscript
        compress
        delaycompress
        notifempty
}
```

Opzione	Significato
`weekly`	
`missingok`	
`rotate 7`	
`postrotate`	
`endscript`	
`compress`	
`delaycompress`	
`notifyempty`	

Esercizi Esplorativi

1. Nella sezione “Modelli e Condizioni di Filtro” abbiamo usato un *filtro basato su espressione* come condizione di filtro. I *filtri basati su proprietà* sono un altro tipo di filtro esclusivo di `rsyslogd`. Traduci il nostro filtro *basato su espressione* in un filtro *basato su proprietà*:

Filtro Basato su Espressione	**Filtro Basato su Proprietà**
`if $FROMHOST-IP=='192.168.1.4' then ?RemoteLogs`	

2. `omusrmsg` è un modulo integrato di `rsyslog` che facilita la notifica agli utenti (invia messaggi di log al terminale dell’utente). Scrivi una regola per inviare tutti i messaggi di tipo *emergency* di tutte le *facility* sia a `root` che all’utente standard `carol`.

Sommario

In questa lezione abbiamo imparato:

- La registrazione dei log è cruciale per l'amministrazione del sistema.
- `rsyslogd` è l'utility incaricata di mantenere i log ordinati e puliti.
- Alcuni servizi si occupano dei propri log.
- Sommariamente, i log possono essere classificati in log di sistema e log di servizio/programma.
- Ci sono un certo numero di utility che sono convenienti per la lettura dei log: `less`, `more`, `zless`, `zmore`, `grep`, `head` e `tail`.
- La maggior parte dei file di log sono file di testo semplice; tuttavia, c'è un piccolo numero di file di log binari.
- Per quanto riguarda i log, `rsyslogd` riceve le informazioni rilevanti da file speciali (socket e buffer di memoria) prima di elaborarle.
- Per classificare i log, `rsyslogd` usa regole in `/etc/rsyslog.conf` o `/etc/rsyslog.d/*`.
- Ogni utente può inserire i propri messaggi nel log di sistema manualmente con l'utilità `logger`.
- `rsyslog` permette di mantenere tutti i log attraverso le reti IP in un server di log centralizzato.
- I modelli sono utili per formattare dinamicamente i nomi dei file di log.
- Lo scopo della rotazione dei log è duplice: evitare che i vecchi log usino troppo spazio su disco e rendere gestibile la consultazione dei log.

Risposte agli Esercizi Guidati

1. Quali utility/comandi usereste nei seguenti scenari:

Scopo e file di log	**Utility**
Leggere `/var/log/syslog.7.gz`	`zmore` o `zless`
Leggere `/var/log/syslog`	`more` o `less`
Cercare la parola `renewal` in `/var/log/syslog`	`grep`
Leggere `/var/log/faillog`	`faillog -a`
Leggere `/var/log/syslog` dinamicamente	`tail -f`

2. Riorganizza le seguenti voci di log in modo che rappresentino un messaggio di log valido con la struttura appropriata:
   - `debian-server`
   - `sshd`
   - `[515]:`
   - `Sep 13 21:47:56`
   - `Server listening on 0.0.0.0 port 22`

   L'ordine corretto è:

   ```
   Sep 13 21:47:56 debian-server sshd[515]: Server listening on 0.0.0.0 port 22
   ```

3. Quali regole aggiungereste a `/etc/rsyslog.conf` per realizzare ciascuna delle seguenti:
   - Inviare tutti i messaggi dalla *facility* `mail` e una priorità/severità di `crit` (e superiore) a `/var/log/mail.crit`:

     ```
     mail.crit                /var/log/mail.crit
     ```

   - Invia tutti i messaggi della *facility* `mail` con priorità `alert` e `emergency` a `/var/log/mail.urgent`:

     ```
     mail.alert                       /var/log/mail.urgent
     ```

- Eccetto quelli provenienti dai servizi `cron` e `ntp`, invia tutti i messaggi — indipendentemente dalla loro *facility* e dalla priorità — a `/var/log/allmessages`:

```
*.*;cron.none;ntp.none                  /var/log/allmessages
```

- Con tutte le impostazioni necessarie configurate prima, invia tutti i messaggi dalla *facility* `mail` a un host remoto il cui indirizzo IP è `192.168.1.88` usando TCP e specificando la porta predefinita:

```
mail.* @@192.168.1.88:514
```

- Indipendentemente dalla loro *facility*, inviare tutti i messaggi con la priorità `warning` (solo con la priorità `warning`) a `/var/log/warnings` evitando un'eccessiva scrittura sul disco:

```
*.=warning                          -/var/log/warnings
```

4. Considera la seguente parte di `/etc/logrotate.d/samba` e spiega le diverse opzioni:

```
carol@debian:~$ sudo head -n 11 /etc/logrotate.d/samba
/var/log/samba/log.smbd {
        weekly
        missingok
        rotate 7
        postrotate
                [ ! -f /var/run/samba/smbd.pid ] || /etc/init.d/smbd reload > /dev/null
        endscript
        compress
        delaycompress
        notifempty
}
```

Opzione	Significato
`weekly`	Ruota i file di log su base settimanale.
`missingok`	Non emette un messaggio di errore se il log è mancante; continua con quello successivo.
`rotate 7`	Mantiene 7 settimane di arretrati.

Opzione	Significato
`postrotate`	Esegue lo script sulla linea seguente dopo aver ruotato i log.
`endscript`	Indica la fine dello script *postrotate*.
`compress`	Comprime i log con `gzip`.
`delaycompress`	In combinazione con `compress`, rimanda la compressione al prossimo ciclo di rotazione.
`notifyempty`	Non ruotare il log se è vuoto.

Risposte agli Esercizi Esplorativi

1. Nella sezione “Modelli e Condizioni di Filtro” abbiamo usato un *filtro basato su espressione* come condizione di filtro. I *filtri basati su proprietà* sono un altro tipo di filtro esclusivo di `rsyslogd`. Traduci il nostro filtro *basato su espressione* in un filtro *basato su proprietà*:

Filtro Basato su Espressione	**Filtro Basato su Proprietà**
`if $FROMHOST-IP=='192.168.1.4' then ?RemoteLogs`	`:fromhost-ip, isequal, "192.168.1.4" ?RemoteLogs`

2. `omusrmsg` è un modulo integrato di `rsyslog` che facilita la notifica agli utenti (invia messaggi di log al terminale dell’utente). Scrivi una regola per inviare tutti i messaggi di tipo *emergency* di tutte le *facility* sia a `root` che all’utente standard `carol`.

```
*.emerg                         :omusrmsg:root,carol
```

Linux Professional Institute

108.2 Lezione 2

Certificazione:	LPIC-1
Versione:	5.0
Argomento:	108 Servizi Essenziali di Sistema
Obiettivo:	108.2 Logging di sistema
Lezione:	2 di 2

Introduzione

Con l'adozione generale di `systemd` da parte di tutte le principali distribuzioni, il demone *journal* (`systemd-journald`) è diventato il servizio di log standard. In questa lezione discuteremo come funziona e come puoi usarlo per fare un certo numero di cose: interrogarlo, filtrare le sue informazioni con diversi criteri, configurare la sua memorizzazione e la sua dimensione, cancellare i vecchi dati, recuperare i suoi dati da un sistema di salvataggio o da una copia del filesystem e — ultimo ma non meno importante — capire la sua interazione con `rsyslogd`.

Fondamenti di `systemd`

Introdotto per la prima volta in Fedora, `systemd` ha progressivamente sostituito *SysV Init* come system e service manager nella maggior parte delle principali distribuzioni Linux. Tra i suoi punti di forza:

- Facilità di configurazione: *unit file* rispetto agli script SysV Init.
- Gestione versatile: oltre a demoni e processi, gestisce anche dispositivi, socket e punti di montaggio.

- Retrocompatibilità con SysV Init e *Upstart*.
- Caricamento parallelo durante l'avvio (al contrario che in Sysv Init che li carica in modo sequenziale).
- Presenta un servizio di *logging* chiamato *journal* che presenta i seguenti vantaggi:
  - Centralizza tutti i log in un unico posto.
  - Non richiede la rotazione dei log.
  - I log possono essere disabilitati, caricati in RAM o resi persistenti.

Unit e Target

`systemd` opera su *unit*. Una unità è una qualsiasi risorsa che `systemd` può gestire (per esempio, rete, bluetooth, ecc.). Le unità, a loro volta, sono governate da *unit file*. Questi sono file di testo semplice che risiedono in `/lib/systemd/system` e includono le impostazioni di configurazione — sotto forma di *sezioni* e *direttive* — per una particolare risorsa da gestire. Ci sono diversi tipi di unità: `service`, `mount`, `automount`, `swap`, `timer`, `device`, `socket`, `path`, `timer`, `snapshot`, `slice`, `scope` e `target`. Così, ogni nome di file di unità segue lo schema `<nome_risorsa>.<tipo_unità>` (per esempio, `reboot.service`).

Un *target* è un tipo speciale di unità che assomiglia ai classici *runlevel* di SysV Init. Questo perché un *target unit* riunisce varie risorse per rappresentare un particolare stato del sistema (per esempio, `graphical.target` è simile a `runlevel 5`, ecc.) Per controllare il target corrente nel tuo sistema, usa il comando `systemctl get-default`:

```
carol@debian:~$ systemctl get-default
graphical.target
```

D'altra parte, i target e i runlevel differiscono in quanto i primi sono reciprocamente inclusivi, mentre i secondi no. Così un target può portarne con se altri, cosa che non è possibile con i runlevel.

NOTE Una spiegazione di come funzionano le unità `systemd` va oltre lo scopo di questa lezione.

Il Sistema Journal: `systemd-journald`

`systemd-journald` è il servizio di sistema che si occupa di ricevere informazioni di log da una varietà di fonti: messaggi del kernel, messaggi di sistema semplici e strutturati, standard output e standard error dei servizi, nonché record di audit dal sottosistema di audit del kernel (per

ulteriori dettagli vedere la pagina di manuale per `systemd-journald`). La sua missione è quella di creare e mantenere un diario strutturato e indicizzato.

Il suo file di configurazione è `/etc/systemd/journald.conf` e — come per qualsiasi altro servizio — si può usare il comando `systemctl` per avviarlo, riavviarlo, fermarlo o — semplicemente — controllare il suo stato:

```
root@debian:~# systemctl status systemd-journald
 systemd-journald.service - Journal Service
   Loaded: loaded (/lib/systemd/system/systemd-journald.service; static; vendor preset:
enabled)
   Active: active (running) since Sat 2019-10-12 13:43:06 CEST; 5min ago
     Docs: man:systemd-journald.service(8)
           man:journald.conf(5)
 Main PID: 178 (systemd-journal)
   Status: "Processing requests..."
    Tasks: 1 (limit: 4915)
   CGroup: /system.slice/systemd-journald.service
           └─178 /lib/systemd/systemd-journald
(...)
```

Sono anche possibili file di configurazione del tipo `journal.conf.d/*.conf` — che possono includere configurazioni specifiche del servizio — (consultare la pagina di manuale di `journald.conf` per saperne di più).

Se abilitato, il journal può essere memorizzato in modo persistente su disco o in modo volatile su un filesystem basato sulla RAM. Il journal non è un file di testo semplice, è binario. Quindi non si possono usare strumenti di analisi del testo come `less` o `more` per leggere il suo contenuto; si usa invece il comando `journalctl`.

Interrogare il Contenuto del Journal

`journalctl` è l'utilità che si usa per interrogare il journal di `systemd`. Devi essere *root* o usare `sudo` per invocarlo. Se interrogato senza opzioni, mostrerà l'intero journal in ordine cronologico (con le voci più vecchie elencate per prime):

```
root@debian:~# journalctl
-- Logs begin at Sat 2019-10-12 13:43:06 CEST, end at Sat 2019-10-12 14:19:46 CEST. --
Oct 12 13:43:06 debian kernel: Linux version 4.9.0-9-amd64 (debian-kernel@lists.debian.org)
(...)
Oct 12 13:43:06 debian kernel: Command line: BOOT_IMAGE=/boot/vmlinuz-4.9.0-9-amd64
root=UUID=b6be6117-5226-4a8a-bade-2db35ccf4cf4 ro qu
```

```
(...)
```

Si possono fare ricerche più specifiche usando un certo numero di opzioni:

`-r`

I messaggi del journal saranno mostrati in ordine inverso:

```
root@debian:~# journalctl -r
-- Logs begin at Sat 2019-10-12 13:43:06 CEST, end at Sat 2019-10-12 14:30:30 CEST. --
Oct 12 14:30:30 debian sudo[1356]: pam_unix(sudo:session): session opened for user root
by carol(uid=0)
Oct 12 14:30:30 debian sudo[1356]:    carol : TTY=pts/0 ; PWD=/home/carol ; USER=root ;
COMMAND=/bin/journalctl -r
Oct 12 14:19:53 debian sudo[1348]: pam_unix(sudo:session): session closed for user root
(...)
```

`-f`

Mostrerà i messaggi del journal più recenti e continuerà a farlo con le nuove voci che man mano verranno aggiunte — in modo simile a `tail -f`:

```
root@debian:~# journalctl -f
-- Logs begin at Sat 2019-10-12 13:43:06 CEST. --
(...)
Oct 12 14:44:42 debian sudo[1356]: pam_unix(sudo:session): session closed for user root
Oct 12 14:44:44 debian sudo[1375]:    carol : TTY=pts/0 ; PWD=/home/carol ; USER=root ;
COMMAND=/bin/journalctl -f
Oct 12 14:44:44 debian sudo[1375]: pam_unix(sudo:session): session opened for user root
by carol(uid=0)

(...)
```

`-e`

Salterà alla fine del journal in modo che le ultime voci siano visibili all'interno della visualizzazione:

```
root@debian:~# journalctl -e
(...)
Oct 12 14:44:44 debian sudo[1375]:    carol : TTY=pts/0 ; PWD=/home/carol ; USER=root ;
COMMAND=/bin/journalctl -f
Oct 12 14:44:44 debian sudo[1375]: pam_unix(sudo:session): session opened for user root
```

```
by carol(uid=0)
Oct 12 14:45:57 debian sudo[1375]: pam_unix(sudo:session): session closed for user root
Oct 12 14:48:39 debian sudo[1378]:    carol : TTY=pts/0 ; PWD=/home/carol ; USER=root ;
COMMAND=/bin/journalctl -e
Oct 12 14:48:39 debian sudo[1378]: pam_unix(sudo:session): session opened for user root
by carol(uid=0)
```

`-n <value>, --lines=<value>`

Mostrerà le *n* linee più recenti (se non viene specificato `<value>`, il valore predefinito è 10):

```
root@debian:~# journalctl -n 5
(...)
Oct 12 14:44:44 debian sudo[1375]:    carol : TTY=pts/0 ; PWD=/home/carol ; USER=root ;
COMMAND=/bin/journalctl -f
Oct 12 14:44:44 debian sudo[1375]: pam_unix(sudo:session): session opened for user root
by carol(uid=0)
Oct 12 14:45:57 debian sudo[1375]: pam_unix(sudo:session): session closed for user root
Oct 12 14:48:39 debian sudo[1378]:    carol : TTY=pts/0 ; PWD=/home/carol ; USER=root ;
COMMAND=/bin/journalctl -e
Oct 12 14:48:39 debian sudo[1378]: pam_unix(sudo:session): session opened for user root
by carol(uid=0)
```

`-k`, `--dmesg`

Equivalente all'uso del comando `dmesg`:

```
root@debian:~# journalctl -k
-- Logs begin at Sat 2019-10-12 13:43:06 CEST, end at Sat 2019-10-12 14:53:20 CEST. --
Oct 12 13:43:06 debian kernel: Linux version 4.9.0-9-amd64 (debian-
kernel@lists.debian.org) (gcc version 6.3.0 20170516 (Debian 6.3.0-18
Oct 12 13:43:06 debian kernel: Command line: BOOT_IMAGE=/boot/vmlinuz-4.9.0-9-amd64
root=UUID=b6be6117-5226-4a8a-bade-2db35ccf4cf4 ro qu
Oct 12 13:43:06 debian kernel: x86/fpu: Supporting XSAVE feature 0x001: 'x87 floating
point registers'
Oct 12 13:43:06 debian kernel: x86/fpu: Supporting XSAVE feature 0x002: 'SSE registers'
(...)
```

Muoversi e Cercare all'Interno del Journal

Si può navigare nell'outuput del journal con:

- PageUp, PageDown e i tasti freccia per muoversi in alto, in basso, a sinistra e a destra.

- `>` per andare alla fine dell'output.
- `<` per andare all'inizio dell'output.

È possibile cercare stringhe sia in avanti che indietro rispetto alla posizione di visualizzazione corrente:

- Ricerca in avanti: Premere `/` e inserire la stringa da cercare, poi premere `Enter`.
- Ricerca a ritroso: Premere `?` e inserire la stringa da cercare, poi premere `Enter`.

Per navigare tra le corrispondenze nelle ricerche, usa `N` per andare alla prossima occorrenza della corrispondenza e `Shift` + `N` per andare alla precedente.

Filtrare i Dati del Journal

Il journal permette di filtrare i dati di log secondo diversi criteri:

Boot number

`--list-boots`

Elenca tutti i boot disponibili. L'output consiste di tre colonne; la prima specifica il numero di avvio (`0` si riferisce all'avvio corrente, `-1` è quello precedente, `-2` quello prima del precedente e così via); la seconda colonna è l'ID dell'avvio; la terza mostra gli indicatori di tempo:

```
root@debian:~# journalctl --list-boots
 0 83df3e8653474ea5aed19b41cdb45b78 Sat 2019-10-12 18:55:41 CEST—Sat 2019-10-12
19:02:24 CEST
```

`-b`, `--boot`

Mostra *tutti* i messaggi dell'avvio corrente. Per vedere i messaggi di log dei boot precedenti, basta aggiungere un parametro di *offset* come spiegato sopra. Per esempio, per avere i messaggi dell'avvio precedente stampati, digiterai `journalctl -b -1`. Ricorda, però, che per recuperare informazioni dai log precedenti, la persistenza del journal deve essere abilitata (imparerai come farlo nella prossima sezione):

```
root@debian:~# journalctl -b -1
Specifying boot ID has no effect, no persistent journal was found
```

Priority

`-p`

È interessante notare che si può anche filtrare per gravità/priorità con l'opzione `-p`:

```
root@debian:~# journalctl -b -0 -p err
-- No entries --
```

Il journal ci informa che — finora — non ci sono stati messaggi con una priorità di `errore` (o superiore) dall'avvio corrente. Nota: `-b -0` può essere omesso quando ci si riferisce all'avvio corrente.

NOTE Far riferimento alla lezione precedente per una lista completa di tutte le severità (o priorità) del `syslog`.

Time Interval

Puoi fare in modo che `journalctl` stampi solo i messaggi registrati in uno specifico lasso di tempo usando gli interruttori `--since` e `--until`. La specifica della data dovrebbe seguire il formato `YYYY-MM-DD HH:MM:SS`. La mezzanotte sarà assunta se si omette la componente temporale. Allo stesso modo, se la data viene omessa, si assume il giorno corrente. Per esempio, per vedere i messaggi registrati dalle 19:00 alle 19:01, si digiterà:

```
root@debian:~# journalctl --since "19:00:00" --until "19:01:00"
-- Logs begin at Sat 2019-10-12 18:55:41 CEST, end at Sat 2019-10-12 20:10:50 CEST. --
Oct 12 19:00:14 debian systemd[1]: Started Run anacron jobs.
Oct 12 19:00:14 debian anacron[1057]: Anacron 2.3 started on 2019-10-12
Oct 12 19:00:14 debian anacron[1057]: Normal exit (0 jobs run)
Oct 12 19:00:14 debian systemd[1]: anacron.timer: Adding 2min 47.988096s random time.
```

Allo stesso modo puoi usare una specifica temporale leggermente diversa: `"integer time-unit ago"`. Così, per vedere i messaggi registrati due minuti fa si digiterà `sudo journalctl --since "2 minutes ago"`. È anche possibile usare `+` e `-` per specificare tempi relativi al tempo corrente, quindi `--since "-2 minutes"` e `--since "-2 minutes ago"` sono equivalenti.

Oltre alle espressioni numeriche, puoi specificare un certo numero di parole chiave:

`yesterday`

A partire dalla mezzanotte del giorno precedente il giorno corrente.

`today`

A partire dalla mezzanotte del giorno corrente.

`tomorrow`

A partire dalla mezzanotte del giorno dopo il giorno corrente.

`now`

L'ora corrente.

Vediamo tutti i messaggi dalla mezzanotte scorsa fino alle 21:00 di oggi:

```
root@debian:~# journalctl --since "today" --until "21:00:00"
-- Logs begin at Sat 2019-10-12 20:45:29 CEST, end at Sat 2019-10-12 21:06:15 CEST. --
Oct 12 20:45:29 debian sudo[1416]:    carol : TTY=pts/0 ; PWD=/home/carol ; USER=root
; COMMAND=/bin/systemctl r
Oct 12 20:45:29 debian sudo[1416]: pam_unix(sudo:session): session opened for user
root by carol(uid=0)
Oct 12 20:45:29 debian systemd[1]: Stopped Flush Journal to Persistent Storage.
(...)
```

NOTE Per saperne di più sulle diverse sintassi per le specifiche di tempo, consultare la pagina di manuale `systemd.time`.

Program

Per vedere i messaggi del journal relativi a uno specifico eseguibile si usa la seguente sintassi: `journalctl /path/to/executable`:

```
root@debian:~# journalctl /usr/sbin/sshd
-- Logs begin at Sat 2019-10-12 20:45:29 CEST, end at Sat 2019-10-12 21:54:49 CEST. --
Oct 12 21:16:28 debian sshd[1569]: Accepted password for carol from 192.168.1.65 port
34050 ssh2
Oct 12 21:16:28 debian sshd[1569]: pam_unix(sshd:session): session opened for user carol
by (uid=0)
Oct 12 21:16:54 debian sshd[1590]: Accepted password for carol from 192.168.1.65 port
34052 ssh2
Oct 12 21:16:54 debian sshd[1590]: pam_unix(sshd:session): session opened for user carol
by (uid=0)
```

Unit

Ricorda: una *unit* è qualsiasi risorsa gestita da `systemd`; è possibile filtrare anche per unit.

`-u`

Mostra i messaggi di un'unità specifica:

```
root@debian:~# journalctl -u ssh.service
-- Logs begin at Sun 2019-10-13 10:50:59 CEST, end at Sun 2019-10-13 12:22:59 CEST. --
Oct 13 10:51:00 debian systemd[1]: Starting OpenBSD Secure Shell server...
Oct 13 10:51:00 debian sshd[409]: Server listening on 0.0.0.0 port 22.
Oct 13 10:51:00 debian sshd[409]: Server listening on :: port 22.
(...)
```

NOTE Per stampare tutte le unità caricate e attive, usa `systemctl list-units`; per vedere tutti i file delle unità installate usa `systemctl list-unit-files`.

Fields

Il journal può anche essere filtrato da specifici *campi* attraverso una delle seguenti sintassi:

- `<field-name>=<value>`
- `_<field-name>=<value>_`
- `__<field-name>=<value>`

`PRIORITY=`

Uno degli otto possibili valori di priorità `syslog` formattati come una stringa decimale:

```
root@debian:~# journalctl PRIORITY=3
-- Logs begin at Sun 2019-10-13 10:50:59 CEST, end at Sun 2019-10-13 14:30:50 CEST.
--
Oct 13 10:51:00 debian avahi-daemon[314]: chroot.c: open() failed: No such file or
directory
```

Nota come avresti potuto ottenere lo stesso risultato usando il comando `sudo journalctl -p err` che abbiamo visto precedentemente.

`SYSLOG_FACILITY=`

Uno qualsiasi dei possibili numeri di codice della *facility* formattato come una stringa decimale. Per esempio, per vedere tutti i messaggi a livello utente:

```
root@debian:~# journalctl SYSLOG_FACILITY=1
-- Logs begin at Sun 2019-10-13 10:50:59 CEST, end at Sun 2019-10-13 14:42:52 CEST.
--
Oct 13 10:50:59 debian mtp-probe[227]: checking bus 1, device 2:
"/sys/devices/pci0000:00/0000:00:06.0/usb1/1-1"
```

```
Oct 13 10:50:59 debian mtp-probe[227]: bus: 1, device: 2 was not an MTP device
Oct 13 10:50:59 debian mtp-probe[238]: checking bus 1, device 2:
"/sys/devices/pci0000:00/0000:00:06.0/usb1/1-1"
Oct 13 10:50:59 debian mtp-probe[238]: bus: 1, device: 2 was not an MTP device
```

`_PID=`

Mostra i messaggi prodotti da uno specifico ID di processo. Per vedere tutti i messaggi prodotti da `systemd`, si dovrebbe digitare:

```
root@debian:~# journalctl _PID=1
-- Logs begin at Sun 2019-10-13 10:50:59 CEST, end at Sun 2019-10-13 14:50:15 CEST.
--
Oct 13 10:50:59 debian systemd[1]: Mounted Debug File System.
Oct 13 10:50:59 debian systemd[1]: Mounted POSIX Message Queue File System.
Oct 13 10:50:59 debian systemd[1]: Mounted Huge Pages File System.
Oct 13 10:50:59 debian systemd[1]: Started Remount Root and Kernel File Systems.
Oct 13 10:50:59 debian systemd[1]: Starting Flush Journal to Persistent Storage...
(...)
```

`_BOOT_ID`

In base al suo ID di avvio si possono individuare i messaggi da un avvio specifico, per esempio: `sudo journalctl _BOOT_ID=83df3e8653474ea5aed19b41cdb45b78`.

`_TRANSPORT`

Mostra i messaggi ricevuti da un *trasporto specifico*. I valori possibili sono: `audit` (sottosistema di audit del kernel), `driver` (generato internamente), `syslog` (socket syslog), `journal` (protocollo journal nativo), `stdout` (output standard dei servizi o errore standard), `kernel` (ring buffer del kernel — lo stesso di `dmesg`, `journalctl -k` o `journalctl --dmesg`):

```
root@debian:~# journalctl _TRANSPORT=journal
-- Logs begin at Sun 2019-10-13 20:19:58 CEST, end at Sun 2019-10-13 20:46:36 CEST.
--
Oct 13 20:19:58 debian systemd[1]: Started Create list of required static device
nodes for the current kernel.
Oct 13 20:19:58 debian systemd[1]: Starting Create Static Device Nodes in /dev...
Oct 13 20:19:58 debian systemd[1]: Started Create Static Device Nodes in /dev.
Oct 13 20:19:58 debian systemd[1]: Starting udev Kernel Device Manager...
(...)
```

Combinare Campi

I campi non si escludono a vicenda, quindi puoi usarne più di uno nella stessa query. Tuttavia, saranno mostrati *solo* i messaggi che corrispondono al valore di entrambi i campi contemporaneamente:

```
root@debian:~# journalctl PRIORITY=3 SYSLOG_FACILITY=0
-- No entries --
root@debian:~# journalctl PRIORITY=4 SYSLOG_FACILITY=0
-- Logs begin at Sun 2019-10-13 20:19:58 CEST, end at Sun 2019-10-13 20:21:55 CEST. --
Oct 13 20:19:58 debian kernel: acpi PNP0A03:00: fail to add MMCONFIG information, can't
access extended PCI configuration (...)
```

A meno che tu non usi il separatore `+` per combinare due espressioni alla maniera di un *OR* logico:

```
root@debian:~# journalctl PRIORITY=3 + SYSLOG_FACILITY=0
-- Logs begin at Sun 2019-10-13 20:19:58 CEST, end at Sun 2019-10-13 20:24:02 CEST. --
Oct 13 20:19:58 debian kernel: Linux version 4.9.0-9-amd64 (debian-kernel@lists.debian.org)
(...9
Oct 13 20:19:58 debian kernel: Command line: BOOT_IMAGE=/boot/vmlinuz-4.9.0-9-amd64
root=UUID= (...)
(...)
```

D'altra parte, puoi fornire due valori per lo stesso campo e tutte le voci che corrispondono a uno dei due valori vengano mostrate:

```
root@debian:~# journalctl PRIORITY=1
-- Logs begin at Sun 2019-10-13 17:16:24 CEST, end at Sun 2019-10-13 17:30:14 CEST. --
-- No entries --
root@debian:~# journalctl PRIORITY=1 PRIORITY=3
-- Logs begin at Sun 2019-10-13 17:16:24 CEST, end at Sun 2019-10-13 17:32:12 CEST. --
Oct 13 17:16:27 debian connmand[459]: __connman_inet_get_pnp_nameservers: Cannot read /pro
Oct 13 17:16:27 debian connmand[459]: The name net.connman.vpn was not provided by any .se
```

NOTE

I campi Journal rientrano in una delle seguenti categorie: "Campi journal dell'utente", "Campi journal di fiducia", "Campi journal del kernel", "Campi per conto di un programma diverso" e "Campi indirizzo". Per maggiori informazioni su questo argomento — inclusa una lista completa dei campi — si veda la pagina `man` per `systemd.journal-fields(7)`.

Voci Manuali nel System Journal: `systemd-cat`

Proprio come il comando `logger` è usato per inviare messaggi dalla linea di comando al log di sistema (come abbiamo visto nella lezione precedente), il comando `systemd-cat` ha uno scopo simile, ma più completo, con il journal di sistema. Ci permette di inviare input standard (*stdin*), output (*stdout*) ed errori (*stderr*) al diario.

Se invocato senza parametri, invierà tutto ciò che legge da *stdin* al journal. Una volta finito, premi `Ctrl` + `C`:.

```
carol@debian:~$ systemd-cat
This line goes into the journal.
^C
```

Se gli viene passato l'output di un comando, anche questo sarà inviato al diario:

```
carol@debian:~$ echo "And so does this line." | systemd-cat
```

Se seguito da un comando, anche l'output di quel comando sarà inviato al journal — insieme a *stderr* (se presente):

```
carol@debian:~$ systemd-cat echo "And so does this line too."
```

C'è anche la possibilità di specificare un livello di priorità con l'opzione `-p`:

```
carol@debian:~$ systemd-cat -p emerg echo "This is not a real emergency."
```

Fate riferimento alla pagina man di `systemd-cat` per conoscere le altre opzioni.

Per vedere le ultime quattro righe del journal:

```
carol@debian:~$ journalctl -n 4
(...)
-- Logs begin at Sun 2019-10-20 13:43:54 CEST. --
Nov 13 23:14:39 debian cat[1997]: This line goes into the journal.
Nov 13 23:19:16 debian cat[2027]: And so does this line.
Nov 13 23:23:21 debian echo[2030]: And so does this line too.
Nov 13 23:26:48 debian echo[2034]: This is not a real emergency.
```

NOTE Le voci del journal con un livello di priorità di *emergency* saranno mostrate in grassetto rosso sulla maggior parte dei sistemi.

Salvataggio Persistente del Journal

Come detto in precedenza, si hanno tre opzioni quando si tratta della posizione del journal:

- Il journaling può essere disattivato del tutto (il reindirizzamento ad altre strutture come la console è però ancora possibile).
- Mantenerlo in memoria — il che lo rende volatile — e sbarazzarsi dei log ad ogni riavvio del sistema. In questo scenario, la directory `/run/log/journal` sarà creata e utilizzata.
- Renderlo persistente in modo che scriva i log su disco. In questo caso, i messaggi di log andranno nella directory `/var/log/journal`.

Il comportamento predefinito è il seguente: se `/var/log/journal/` non esiste, i log saranno salvati in modo volatile in una directory in `/run/log/journal/` e — quindi — persi al riavvio. Il nome della directory — il `/etc/machine-id` — è una stringa esadecimale di 32 caratteri minuscoli terminata da una linea nuova:

```
carol@debian:~$ ls /run/log/journal/8821e1fdf176445697223244d1dfbd73/
system.journal
```

Se provi a leggerlo con `less` riceverai un avvertimento, quindi usa invece il comando `journalctl`:

```
root@debian:~# less /run/log/journal/9a32ba45ce44423a97d6397918de1fa5/system.journal
"/run/log/journal/9a32ba45ce44423a97d6397918de1fa5/system.journal" may be a binary file.
See it anyway?
root@debian:~# journalctl
-- Logs begin at Sat 2019-10-05 21:26:38 CEST, end at Sat 2019-10-05 21:31:27 CEST. --
(...)
Oct 05 21:26:44 debian systemd-journald[1712]: Runtime journal
(/run/log/journal/9a32ba45ce44423a97d6397918de1fa5) is 4.9M, max 39.5M, 34.6M free.
Oct 05 21:26:44 debian systemd[1]: Started Journal Service.
(...)
```

Se `/var/log/journal/` esiste, i log vi saranno memorizzati in modo persistente. Se questa directory venisse cancellata, `systemd-journald` non la ricreerebbe ma scriverebbe invece su `/run/log/journal`. Non appena creeremo nuovamente `/var/log/journal/` e riavvieremo il demone, la registrazione persistente sarà ristabilita:

```
root@debian:~# mkdir /var/log/journal/
root@debian:~# systemctl restart systemd-journald
root@debian:~# journalctl
(...)
Oct 05 21:33:49 debian systemd-journald[1712]: Received SIGTERM from PID 1 (systemd).
Oct 05 21:33:49 debian systemd[1]: Stopped Journal Service.
Oct 05 21:33:49 debian systemd[1]: Starting Journal Service...
Oct 05 21:33:49 debian systemd-journald[1768]: Journal started
Oct 05 21:33:49 debian systemd-journald[1768]: System journal
(/var/log/journal/9a32ba45ce44423a97d6397918de1fa5) is 8.0M, max 1.1G, 1.1G free.
Oct 05 21:33:49 debian systemd[1]: Started Journal Service.
Oct 05 21:33:49 debian systemd[1]: Starting Flush Journal to Persistent Storage...
(...)
```

NOTE Per impostazione predefinita, ci saranno file di journal specifici per ogni utente loggato, situati in `/var/log/journal/`, quindi — insieme ai file `system.journal` — troverai anche file del tipo `user-1000.journal`.

Oltre a ciò che abbiamo appena menzionato, il modo in cui il demone journal si occupa della memorizzazione dei log può essere cambiato dopo l'installazione modificando il suo file di configurazione: `/etc/systemd/journald.conf`. L'opzione chiave è `Storage=` e può avere i seguenti valori:

`Storage=volatile`

I dati di log saranno memorizzati esclusivamente in memoria — sotto `/run/log/journal/`. Se non è presente, la directory verrà creata.

`Storage=persistent`

Per default i dati di log saranno memorizzati su disco — sotto `/var/log/journal/` — con un fallback alla memoria (`/run/log/journal/`) durante le prime fasi di avvio e se il disco non è scrivibile. Entrambe le directory saranno create se necessario.

`Storage=auto`

`auto` è simile a `persistent`, ma la directory `/var/log/journal` non viene creata se necessario. Questo è il default.

`Storage=none`

Tutti i dati di log saranno scartati. L'inoltro ad altre destinazioni come la console, il buffer di log del kernel o un *socket syslog* sono comunque possibili.

Per esempio, per far sì che `systemd-journald` crei `/var/log/journal/` e passi alla

memorizzazione persistente, si dovrebbe modificare `/etc/systemd/journald.conf` e impostare `Storage=persistent`, salvare il file e riavviare il demone con `sudo systemctl restart systemd-journald`. Per assicurarti che il riavvio sia avvenuto senza problemi, puoi sempre controllare lo stato del demone:

```
root@debian:~# systemctl status systemd-journald
 systemd-journald.service - Journal Service
   Loaded: loaded (/lib/systemd/system/systemd-journald.service; static; vendor preset:
enabled)
   Active: active (running) since Wed 2019-10-09 10:03:40 CEST; 2s ago
     Docs: man:systemd-journald.service(8)
           man:journald.conf(5)
 Main PID: 1872 (systemd-journal)
   Status: "Processing requests..."
    Tasks: 1 (limit: 3558)
   Memory: 1.1M
   CGroup: /system.slice/systemd-journald.service
           └─1872 /lib/systemd/systemd-journald

Oct 09 10:03:40 debian10 systemd-journald[1872]: Journal started
Oct 09 10:03:40 debian10 systemd-journald[1872]: System journal
(/var/log/journal/9a32ba45ce44423a97d6397918de1fa5) is 8.0M, max 1.2G, 1.2G free.
```

NOTE

I file del diario in `/var/log/journal/<machine-id>/` o `/run/log/journal/<machine-id>/` hanno il suffisso `.journal` (es. `system.journal`). Tuttavia, se vengono trovati corrotti o se il demone viene fermato in modo poco pulito, saranno rinominati con l'aggiunta di `~` (per esempio, `system.journal~`) e il demone inizierà a scrivere su un nuovo file pulito.

Cancellare i Dati del Vecchio Journal: Dimensione del Journal

I log sono salvati in *journal file* i cui nomi finiscono con `.journal` o `.journal~` e si trovano nella directory appropriata (`/run/log/journal` o `/var/log/journal` come configurato). Per controllare quanto spazio su disco è attualmente occupato dai file di journal (sia archiviati sia attivi), usa lo switch `--disk-usage`:

```
root@debian:~# journalctl --disk-usage
Archived and active journals take up 24.0M in the filesystem.
```

I log di `systemd` hanno di default un massimo del 10% della dimensione del filesystem in cui sono memorizzati. Per esempio su un filesystem da 1GB non occuperanno più di 100MB. Una volta

raggiunto questo limite, i vecchi log inizieranno a scomparire così che si rimanga vicini a questo valore.

Tuttavia, l'applicazione dei limiti di dimensione sui file di journal memorizzati può essere gestita modificando una serie di opzioni di configurazione in `/etc/systemd/journald.conf`. Queste opzioni rientrano in due categorie a seconda del tipo di filesystem usato: persistente (`/var/log/journal`) o in-memoria (`/run/log/journal`). Il primo usa opzioni che sono precedute dalla parola `System` e si applicano solo se il logging persistente è correttamente abilitato e una volta che il sistema è completamente avviato. I nomi delle opzioni nel secondo iniziano con la parola `Runtime` e si applicano nei seguenti scenari:

`SystemMaxUse=`, **`RuntimeMaxUse=`**

Controllano la quantità di spazio su disco che può essere occupata dal journal. Il valore predefinito è il 10% della dimensione del filesystem, ma può essere modificato (per esempio, `SystemMaxUse=500M`) purché non superi un massimo di 4GiB.

`SystemKeepFree=`, **`RuntimeKeepFree=`**

Controllano la quantità di spazio su disco che dovrebbe essere lasciato libero per altri utenti. Il valore predefinito è il 15% della dimensione del filesystem, ma può essere modificato (per esempio, `SystemKeepFree=500M`) finché non supera un massimo di 4GiB.

Per quanto riguarda la precedenza di `*MaxUse` e `*KeepFree`, `systemd-journald` li soddisferà entrambi usando il più piccolo dei due valori. Allo stesso modo, si tenga presente che solo i file di journal archiviati (al contrario di quelli attivi) vengono cancellati.

`SystemMaxFileSize=`, **`RuntimeMaxFileSize=`**

Controllano la dimensione massima a cui possono crescere i singoli file del diario. Il valore predefinito è 1/8 di `*MaxUse`. La riduzione della dimensione viene effettuata in modo sincrono e i valori possono essere specificati in byte o usando `K`, `M`, `G`, `T`, `P`, `E` per Kibibyte, Mebibyte, Gibibyte, Tebibyte, Pebibyte e Exbibyte, rispettivamente.

`SystemMaxFiles=`, **`RuntimeMaxFiles=`**

Stabiliscono il numero massimo di file di journal individuali e archiviati da memorizzare (i file di journal attivi non sono interessati). Il valore predefinito è 100.

Oltre alla cancellazione basata sulle dimensioni e alla rotazione dei messaggi di log, `systemd-journald` permette anche criteri basati sul tempo utilizzando le seguenti due opzioni: `MaxRetentionSec=` e `MaxFileSec=`. Fai riferimento alla pagina di manuale di `journald.conf` per maggiori informazioni su queste e altre opzioni.

NOTE Ogni volta che si modifica il comportamento predefinito di `systemd-journald`

> decommentando e modificando le opzioni in `/etc/systemd/journald.conf`, è necessario riavviare il demone perché le modifiche abbiano effetto.

Pulire il Journal

Puoi pulire manualmente i file di journal archiviati in qualsiasi momento con una delle seguenti tre opzioni:

`--vacuum-time=`

Questa opzione basata sul tempo eliminerà tutti i messaggi nei file di journal con un timestamp più vecchio dell'intervallo di tempo specificato. I valori devono essere scritti con uno dei seguenti suffissi: `s`, `m`, `h`, `days` (o `d`), `months`, `weeks` (o `w`) e `years` (o `y`). Per esempio, per sbarazzarsi di tutti i messaggi nei file di journal archiviati che sono più vecchi di 1 mese:

```
root@debian:~# journalctl --vacuum-time=1months
Deleted archived journal
/var/log/journal/7203088f20394d9c8b252b64a0171e08/system@27dd08376f71405a91794e632ede97ed
-0000000000000001-00059475764d46d6.journal (16.0M).
Deleted archived journal /var/log/journal/7203088f20394d9c8b252b64a0171e08/user-
1000@e7020d80d3af42f0bc31592b39647e9c-000000000000008e-00059479df9677c8.journal (8.0M).
```

`--vacuum-size=`

Questa opzione basata sulla dimensione cancellerà i file di journal archiviati fino a quando non occuperanno un valore inferiore alla dimensione specificata. I valori devono essere scritti con uno dei seguenti suffissi: `K`, `M`, `G` o `T`. Per esempio, per eliminare i file di journal archiviati finché non sono al di sotto di 100 Mebibytes:

```
root@debian:~# journalctl --vacuum-size=100M
Vacuuming done, freed 0B of archived journals from
/run/log/journal/9a32ba45ce44423a97d6397918de1fa5.
```

`--vacuum-files=`

Questa opzione farà in modo che non rimangano più file di journal archiviati del numero specificato. Il valore è un numero intero. Per esempio, per limitare il numero di file archiviati a 10:

```
root@debian:~# journalctl --vacuum-files=10
Vacuuming done, freed 0B of archived journals from
/run/log/journal/9a32ba45ce44423a97d6397918de1fa5.
```

La cancellazione rimuove *solo* i file di journal archiviati. Se vuoi sbarazzarti di tutto (inclusi i file attivi), devi usare un *segnale* (`SIGUSR2`) che inneschi la rotazione immediata dei file di journal con l'opzione `--rotate`. Altri segnali importanti possono essere invocati con le seguenti opzioni:

`--flush` (`SIGUSR1`)
: Richiede lo svuotamento dei file del journal da `/run/` a `/var/` per rendere il journal persistente. Richiede che il logging persistente sia abilitato e che `/var/` sia montato.

`--sync` (`SIGRTMIN+1`)
: Si usa per richiedere che tutti i dati di registro non ancora scritti siano scritti su disco.

NOTE Per controllare la coerenza interna del file di journal, usa `journalctl` con l'opzione `--verify`. Vedrai una barra di progresso man mano che il controllo viene fatto e ogni possibile problema verrà mostrato.

Recuperare i Dati del Journal da un Rescue System

Come amministratore di sistema, potresti trovarti in una situazione in cui hai bisogno di accedere ai file del journal sul disco rigido di una macchina difettosa attraverso un *rescue system* (un CD avviabile o una chiave USB contenente una distribuzione Linux live).

`journalctl` cerca i file del journal in `/var/log/journal/<machine-id>/`. Poiché gli ID delle macchine sul sistema *rescue* e su quello guasto saranno diversi, devi usare la seguente opzione:

`-D </path/to/dir>`, `--directory=</path/to/dir>`
: Con questa opzione si specifica un percorso di directory in cui `journalctl` cercherà i file del journal invece delle posizioni predefinite di runtime e di sistema.

Quindi, è necessario montare il `rootfs` del sistema difettoso (`/dev/sda1`) sul filesystem del sistema di salvataggio e procedere alla lettura dei file del diario in questo modo:

```
root@debian:~# journalctl -D /media/carol/faulty.system/var/log/journal/
-- Logs begin at Sun 2019-10-20 12:30:45 CEST, end at Sun 2019-10-20 12:32:57 CEST. --
oct 20 12:30:45 suse-server kernel: Linux version 4.12.14-lp151.28.16-default
(geeko@buildhost) (...)
oct 20 12:30:45 suse-server kernel: Command line: BOOT_IMAGE=/boot/vmlinuz-4.12.14-
lp151.28.16-default root=UUID=7570f67f-4a08-448e-aa09-168769cb9289 splash=>
oct 20 12:30:45 suse-server kernel: x86/fpu: Supporting XSAVE feature 0x001: 'x87 floating
point registers'
oct 20 12:30:45 suse-server kernel: x86/fpu: Supporting XSAVE feature 0x002: 'SSE registers'
(...)
```

Altre opzioni che possono essere utili in questo scenario sono:

`-m`, `--merge`
: Unisce le voci di tutti i journal disponibili sotto `/var/log/journal`, inclusi quelli remoti.

`--file`
: Mostra le voci in un file specifico, per esempio: `journalctl --file /var/log/journal/64319965bda04dfa81d3bc4e7919814a/user-1000.journal`.

`--root`
: Un percorso che indichi la directory principale è passato come argomento. `journalctl` cercherà lì i file del journal (per esempio `journalctl --root /faulty.system/`).

Vedere la pagina man di `journalctl` per maggiori informazioni.

Inoltrare i Dati di Log a un Demone Standard `syslog`

I dati di log del journal possono essere resi disponibili a un demone standard `syslog`:

- Inoltro dei messaggi al file socket `/run/systemd/journal/syslog` per la lettura di `syslogd`. Questa funzione è abilitata con l'opzione `ForwardToSyslog=yes`.
- Avere un demone `syslog` che si comporti come `journalctl`, quindi leggere i messaggi di log direttamente dai file di journal. In questo caso, l'opzione rilevante è quella di `Storage`; deve avere un valore diverso da `none`.

NOTE

Allo stesso modo, puoi inoltrare i messaggi di log ad altre destinazioni con le seguenti opzioni: `ForwardToKMsg` (*kernel log buffer* — `kmsg`), `ForwardToConsole` (la console di sistema) o `ForwardToWall` (tutti gli utenti connessi tramite `wall`). Per maggiori informazioni, consulta la pagina man di `journald.conf`.

Esercizi Guidati

1. Supponendo che tu sia `root`, completa la tabella con il comando appropriato "journalctl":

Scopo	**Comando**
Mostra le voci del kernel	
Mostra i messaggi del secondo avvio a partire dall'inizio del journal	
Mostra i messaggi del secondo avvio a partire dalla fine del journal	
Mostra i messaggi di log più recenti e continua a guardare se ce ne sono di nuovi	
Mostra solo i nuovi messaggi da ora, e aggiorna l'output continuamente	
Mostra i messaggi dell'avvio precedente con una priorità di `warning` e in ordine inverso	

2. Il comportamento del demone journal riguardo allo storage è controllato principalmente dal valore dell'opzione `Storage` in `/etc/systemd/journald.conf`. Indica quale comportamento è legato a quale valore nella tabella seguente:

Behaviour	`Storage=auto`	`Storage=none`	`Storage=persistent`	`Storage=volatile`
I dati di log vengono buttati via ma l'inoltro è possibile.				
Una volta che il sistema si è avviato, i dati di log saranno memorizzati in `/var/log/journal`. Se non è già presente, la directory verrà creata.				

Behaviour	`Storage=auto`	`Storage=none`	`Storage=persistent`	`Storage=volatile`
Una volta che il sistema si è avviato, i dati di log saranno memorizzati in `/var/log/journal`. Se non è già presente, la directory non verrà creata.				
I dati di log saranno memorizzati sotto `/var/run/journal` ma non sopravviveranno ai riavvii.				

3. Come hai ormai imparato, il journal può essere ripulito manualmente in base al tempo, alla dimensione e al numero di file. Completa i seguenti compiti usando `journalctl` e le opzioni appropriate:
   - Controlla quanto spazio su disco è occupato dai file del journal:
   - Riduci la quantità di spazio riservato ai file di journal archiviati e imposta a 200MiB:
   - Controlla di nuovo lo spazio su disco e spiega i risultati:

Esercizi Esplorativi

1. Quali opzioni dovresti modificare in `/etc/systemd/journald.conf` in modo che i messaggi siano inoltrati a `/dev/tty5`? Quali valori dovrebbero avere le opzioni?

2. Fornisci il corretto filtro `journalctl` per mostrare quanto segue:

Scopo	**Filtro + Valore**
Mostra i messaggi che appartengono ad un utente specifico.	
Mostra i messaggi da un host chiamato `debian`.	
Mostra i messaggi che appartengono a un gruppo specifico.	
Mostra i messaggi appartenenti a `root`	
In base al percorso dell'eseguibile, mostra i messaggi di `sudo`	
In base al nome del comando, mostra i messaggi di `sudo`	

3. Quando si filtra per priorità, anche i log con una priorità più alta di quella indicata saranno inclusi nell'elenco; per esempio `journalctl -p err` stamperà i messaggi *error*, *critical*, *alert* e *emergency*. Tuttavia, puoi fare in modo che `journalctl` mostri solo un intervallo specifico. Quale comando useresti per far mostrare a `journalctl` solo i messaggi nei livelli di priorità *warning*, *error* e *critical*?

4. I livelli di priorità possono anche essere specificati numericamente. Riscrivi il comando dell'esercizio precedente usando la rappresentazione numerica dei livelli di priorità:

Sommario

In questa lezione abbiamo imparato:

- I vantaggi di usare `systemd` come gestore di sistemi e servizi.
- Le basi delle *unit* e dei *target* di `systemd`.
- Da dove `systemd-journald` prende i dati di logging.
- Le opzioni che puoi passare a `systemctl` per controllare `systemd-journald`: `start`, `status`, `restart` e `stop`.
- Dove si trova il file di configurazione del journal — `/etc/systemd/journald.conf` — e le sue opzioni principali.
- Come interrogare il journal in modo generale e per dati specifici utilizzando i filtri.
- Come navigare e cercare nel journal.
- Come gestire la memorizzazione dei file di journal: in memoria o su disco.
- Come disabilitare del tutto il journaling.
- Come controllare lo spazio su disco occupato dal journal, imporre limiti di dimensione ai file di journal archiviati e pulire manualmente i file di journal archiviati (*vacumming*).
- Come recuperare i dati del journal da un sistema di *rescue*.
- Come inoltrare i dati di log a un demone standard `syslog`.

Comandi utilizzati in questa lezione:

`systemctl`
Controlla il sistema `systemd` e il gestore dei servizi.

`journalctl`
Interroga il journal di `systemd`.

`ls`
Elenca il contenuto della directory.

`less`
Visualizza il contenuto del file.

`mkdir`
Crea directories.

Risposte agli Esercizi Guidati

1. Supponendo che tu sia `root`, completa la tabella con il comando appropriato "journalctl":

Scopo	**Comando**
Mostra le voci del kernel	`journalctl -k` o `journalctl --dmesg`
Mostra i messaggi del secondo avvio a partire dall'inizio del journal	`journalctl -b 2`
Mostra i messaggi del secondo avvio a partire dalla fine del journal	`journalctl -b -2 -r` o `journalctl -r -b -2`
Mostra i messaggi di log più recenti e continua a guardare se ce ne sono di nuovi	`journalctl -f`
Mostra solo i nuovi messaggi da ora, e aggiorna l'output continuamente	`journalctl --since "now" -f`
Mostra i messaggi dell'avvio precedente con una priorità di `warning` e in ordine inverso	`journalctl -b -1 -p warning -r`

2. Il comportamento del demone journal riguardo allo storage è controllato principalmente dal valore dell'opzione `Storage` in `/etc/systemd/journald.conf`. Indica quale comportamento è legato a quale valore nella tabella seguente:

Behaviour	`Storage=auto`	`Storage=none`	`Storage=persistent`	`Storage=volatile`
I dati di log vengono buttati via ma l'inoltro è possibile.		x		
Una volta che il sistema si è avviato, i dati di log saranno memorizzati in `/var/log/journal`. Se non è già presente, la directory verrà creata.			x	

Behaviour	`Storage=auto`	`Storage=none`	`Storage=persistent`	`Storage=volatile`
Una volta che il sistema si è avviato, i dati di log saranno memorizzati in `/var/log/journal`. Se non è già presente, la directory non verrà creata.	x			
I dati di log saranno memorizzati sotto `/var/run/journal` ma non sopravviveranno ai riavvii.				x

3. Come hai imparato, il journal può essere ripulito manualmente in base al tempo, alla dimensione e al numero di file. Completa i seguenti compiti usando `journalctl` e le opzioni appropriate:
   - Controlla quanto spazio su disco è occupato dai file del journal:

     ```
     journalctl --disk-usage
     ```

   - Riduci la quantità di spazio riservato ai file di journal archiviati e imposta a 200MiB:

     ```
     journalctl --vacuum-size=200M
     ```

   - Controlla di nuovo lo spazio su disco e spiega i risultati:

     ```
     journalctl --disk-usage
     ```

     Non c'è correlazione perché `--disk-usage` mostra lo spazio occupato da entrambi i file di giornale attivi e archiviati mentre `--vacuum-size` si applica solo ai file archiviati.

Risposte agli Esercizi Esplorativi

1. Quali opzioni dovresti modificare in `/etc/systemd/journald.conf` in modo che i messaggi siano inoltrati a `/dev/tty5`? Quali valori dovrebbero avere le opzioni?

```
ForwardToConsole=yes
TTYPath=/dev/tty5
```

2. Fornisci il corretto filtro `journalctl` per mostrare quanto segue:

Scopo	**Filtro + Valore**
Mostra i messaggi che appartengono ad un utente specifico.	`_ID=<user-id>`
Mostra i messaggi da un host chiamato `debian`.	`_HOSTNAME=debian`
Mostra i messaggi che appartengono a un gruppo specifico.	`_GID=<group-id>`
Mostra i messaggi appartenenti a `root`.	`_UID=0`
In base al percorso dell'eseguibile, mostra i messaggi di `sudo`.	`_EXE=/usr/bin/sudo`
In base al nome del comando, mostra i messaggi di `sudo`.	`_COMM=sudo`

3. Quando si filtra per priorità, anche i log con una priorità più alta di quella indicata saranno inclusi nell'elenco; per esempio `journalctl -p err` stamperà i messaggi *error*, *critical*, *alert* e *emergency*. Tuttavia, puoi fare in modo che `journalctl` mostri solo un intervallo specifico. Quale comando useresti per far mostrare a `journalctl` solo i messaggi nei livelli di priorità *warning*, *error* e *critical*?

```
journalctl -p warning..crit
```

4. I livelli di priorità possono anche essere specificati numericamente. Riscrivi il comando dell'esercizio precedente usando la rappresentazione numerica dei livelli di priorità:

```
journalctl -p 4..2
```



108.3 Concetti base dei Mail Transfer Agent (MTA)

Obiettivi LPI di riferimento

LPIC-1 version 5.0, Exam 102, Objective 108.3

Peso

3

Aree di Conoscenza Chiave

- Creare un alias di posta elettronica.
- Configurare l'inoltro della posta elettronica.
- Conoscenza dei programmi MTA comunemente disponibili (postfix, sendmail, exim) (nessuna configurazione)

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

- `~/.forward`
- sendmail emulation layer commands
- `newaliases`
- `mail`
- `mailq`
- `postfix`
- `sendmail`
- `exim`

108.3 Lezione 1

Certificazione:	LPIC-1
Versione:	5.0
Argomento:	108 Servizi Essenziali di Sistema
Obiettivo:	108.3 Concetti base dei Mail Transfer Agent
Lezione:	1 di 1

Introduzione

Nei sistemi operativi *Unix-like*, come Linux, ogni utente ha la propria *inbox*: una posizione speciale sul filesystem che è inaccessibile da altri utenti non root e che memorizza i messaggi personali dell'utente. I nuovi messaggi in arrivo vengono aggiunti alla casella di posta dell'utente dal *Mail Transfer Agent* (MTA). L'MTA è un programma in esecuzione come servizio di sistema che raccoglie i messaggi inviati da altri account locali così come i messaggi ricevuti dalla rete, inviati da utenti remoti.

Lo stesso MTA è anche responsabile dell'invio di messaggi alla rete, se l'indirizzo di destinazione si riferisce a un account remoto. Lo fa usando una locazione del filesystem come *outbox* di posta elettronica per tutti gli utenti del sistema: non appena un utente mette un nuovo messaggio nella outbox, l'MTA identificherà il nodo di rete di destinazione dal nome di dominio dato dall'indirizzo email di destinazione — la parte dopo il segno @ — e poi proverà a trasferire il messaggio all'MTA remoto usando il *Simple Mail Transfer Protocol* (SMTP). SMTP è stato progettato con reti inaffidabili in mente, quindi cercherà di stabilire percorsi di consegna alternativi se il nodo primario di destinazione della posta non è raggiungibile.

MTA Locale e Remoto

Gli account utente tradizionali nelle macchine collegate in rete costituiscono lo scenario di scambio di posta elettronica più semplice, dove ogni nodo della rete esegue il proprio demone MTA. Nessun altro software oltre all'MTA è richiesto per inviare e ricevere messaggi di posta elettronica. In pratica, tuttavia, è più comune usare un account di posta elettronica remoto e non avere un servizio MTA locale attivo (cioè usare invece un'applicazione client di posta elettronica per accedere all'account remoto).

A differenza degli account locali, un account di posta elettronica remoto — chiamato anche *remote mailbox* — richiede l'autenticazione dell'utente per garantire l'accesso alla casella di posta dell'utente e all'MTA remoto (in questo caso, chiamato semplicemente *SMTP server*). Mentre l'utente che interagisce con una casella di posta e un MTA locali è già identificato dal sistema, un sistema remoto deve verificare l'identità dell'utente prima di gestire i suoi messaggi attraverso IMAP o POP3.

NOTE

Al giorno d'oggi il metodo più comune per inviare e ricevere email è attraverso un account ospitato su un server remoto, per esempio il server email centralizzato di una società che ospita tutti gli account dei dipendenti o un servizio email personale, come *Gmail* di Google. Invece di raccogliere i messaggi consegnati localmente, l'applicazione client di posta elettronica si connette alla casella di posta remota e recupera i messaggi da lì. I protocolli POP3 e IMAP sono comunemente usati per recuperare i messaggi dal server remoto, ma possono essere usati anche altri protocolli proprietari non standard.

Quando un demone MTA è in esecuzione sul sistema locale, gli utenti locali possono inviare un'email ad altri utenti locali o a utenti su una macchina remota, a condizione che il loro sistema abbia anche un servizio MTA che accetti connessioni di rete. La porta TCP 25 è la porta standard per la comunicazione SMTP, ma possono essere usate anche altre porte, a seconda dello schema di autenticazione e/o crittografia utilizzato (se presenti).

Lasciando da parte i metodi che coinvolgono l'accesso a caselle di posta remote, una rete di scambio di e-mail tra normali account utente Linux può essere implementata a condizione che tutti i nodi della rete abbiano un MTA attivo che sia in grado di eseguire i seguenti compiti:

- Mantenere la coda di messaggi in uscita da inviare. Per ogni messaggio in coda, l'MTA locale valuterà l'MTA di destinazione dall'indirizzo del destinatario.
- Comunicare con demoni MTA remoti usando SMTP. L'MTA locale dovrebbe essere in grado di usare il Simple Mail Transfer Protocol (SMTP) sullo stack TCP/IP per ricevere, inviare e reindirizzare messaggi da/a altri demoni MTA remoti.

- Mantenere una casella di posta individuale per ogni account locale. L'MTA di solito memorizza i messaggi nel formato *mbox*: un singolo file di testo contenente tutti i messaggi di posta elettronica in sequenza.

Normalmente, gli indirizzi email specificano un nome di dominio come luogo, per esempio `lpi.org` in `info@lpi.org`. Quando questo è il caso, l'MTA del mittente interrogherà il servizio DNS per il corrispondente record *MX*. Il record DNS MX contiene l'indirizzo IP dell'MTA che gestisce l'email per quel dominio. Se lo stesso dominio ha più di un record MX specificato nel DNS, l'MTA dovrebbe provare a contattarli secondo i loro valori di priorità. Se l'indirizzo del destinatario non specifica un nome di dominio o il dominio non ha un record MX, allora la parte dopo il simbolo `@` sarà trattata come l'host dell'MTA di destinazione.

Gli aspetti di sicurezza devono essere considerati se gli host MTA saranno visibili agli host su Internet. Per esempio, è possibile per un utente sconosciuto usare l'MTA locale per impersonare un altro utente e inviare email potenzialmente dannose. Un MTA che inoltra ciecamente un'email è conosciuto come un *open relay*, quando può essere usato come intermediario per mascherare potenzialmente il vero mittente del messaggio. Per prevenire questi abusi, la raccomandazione è di accettare connessioni solo da domini autorizzati e di implementare uno schema di autenticazione sicuro.

Inoltre, ci sono molte diverse implementazioni di MTA per Linux, ognuna delle quali si concentra su aspetti specifici come compatibilità, prestazioni, sicurezza, ecc. Tuttavia, tutti gli MTA seguono gli stessi principi di base e forniscono caratteristiche simili.

MTA in Linux

Il tradizionale MTA disponibile per i sistemi Linux è *Sendmail*, un MTA molto flessibile e di uso generale usato da molti sistemi operativi *Unix-like*. Alcuni degli altri MTA comuni sono *Postfix*, *qmail* e *Exim*. La ragione principale per scegliere un MTA alternativo è quella di implementare funzionalità avanzate più facilmente, poiché configurare server di posta elettronica personalizzati in Sendmail può essere un compito complicato. Inoltre, ogni distribuzione può avere il suo MTA preferito, con impostazioni predefinite appropriate per le configurazioni più comuni. Tutti gli MTA sono intesi come sostituti di Sendmail, quindi tutte le applicazioni compatibili con Sendmail dovrebbero poter funzionare indipendentemente dall'MTA utilizzato.

Se l'MTA è in funzione ma non accetta connessioni di rete, sarà in grado di consegnare i messaggi email solo sulla macchina locale. Per l'MTA `sendmail`, il file `/etc/mail/sendmail.mc` dovrebbe essere modificato per accettare connessioni non locali. Per farlo, la voce

```
DAEMON_OPTIONS(`Port=smtp,Addr=127.0.0.1, Name=MTA')dnl
```

dovrebbe essere modificata con l'indirizzo di rete corretto e il servizio dovrebbe essere riavviato. Alcune distribuzioni Linux, come Debian, possono offrire strumenti di configurazione per aiutare a portare il server di posta elettronica con un set predefinito di funzioni comunemente usate.

TIP

A causa di problemi di sicurezza la maggior parte delle distribuzioni Linux non installa un MTA di default. Per testare gli esempi dati in questa lezione, assicurati che ci sia un MTA in funzione su ogni macchina e che accetti connessioni sulla porta TCP 25. Per motivi di sicurezza, questi sistemi non dovrebbero essere esposti a connessioni in entrata da Internet durante i test.

Una volta che l'MTA è in funzione e accetta connessioni dalla rete, i nuovi messaggi email gli vengono passati con comandi SMTP che vengono inviati attraverso una connessione TCP. Il comando `nc` — un'utilità di rete che legge e scrive dati generici attraverso la rete — può essere usato per inviare comandi SMTP direttamente all'MTA. Se il comando `nc` non è disponibile, sarà installato con il pacchetto *ncat* o *nmap-ncat*, a seconda del sistema di gestione dei pacchetti in uso. Scrivere comandi SMTP direttamente all'MTA ti aiuterà a capire meglio il protocollo e altri concetti generali di posta elettronica, ma può anche aiutare a diagnosticare problemi nel processo di consegna della posta.

Se per esempio l'utente `emma` sull'host `lab1.campus` vuole inviare un messaggio all'utente `dave` sull'host `lab2.campus`, allora può usare il comando `nc` per connettersi direttamente all'MTA `lab2.campus`, assumendo che sia in ascolto sulla porta TCP 25:

```
$ nc lab2.campus 25
220 lab2.campus ESMTP Sendmail 8.15.2/8.15.2; Sat, 16 Nov 2019 00:16:07 GMT
HELO lab1.campus
250 lab2.campus Hello lab1.campus [10.0.3.134], pleased to meet you
MAIL FROM: emma@lab1.campus
250 2.1.0 emma@lab1.campus... Sender ok
RCPT TO: dave@lab2.campus
250 2.1.5 dave@lab2.campus... Recipient ok
DATA
354 Enter mail, end with "." on a line by itself
Subject: Recipient MTA Test

Hi Dave, this is a test for your MTA.
.
250 2.0.0 xAG0G7Y0000595 Message accepted for delivery
QUIT
221 2.0.0 lab2.campus closing connection
```

Una volta stabilita la connessione, l'MTA remoto si identifica e poi è pronto a ricevere i comandi

SMTP. Il primo comando SMTP nell'esempio, `HELO lab1.campus`, indica `lab1.campus` come iniziatore dello scambio. I prossimi due comandi, `MAIL FROM: emma@lab1.campus` e `RCPT TO: dave@lab2.campus`, indicano il mittente e il destinatario. Il messaggio email vero e proprio inizia dopo il comando `DATA` e finisce con un punto su una linea a sé. Per aggiungere un campo `subject` all'email, dovrebbe essere nella prima riga dopo il comando `DATA`, come mostrato nell'esempio. Quando il campo oggetto è usato, ci deve essere una linea vuota che lo separa dal contenuto dell'email. Il comando `QUIT` termina la connessione con l'MTA sull'host `lab2.campus`.

Sull'host `lab2.campus`, l'utente `dave` riceverà un messaggio simile a `You have new mail in /var/spool/mail/dave` non appena entri in una sessione di shell. Questo file conterrà il messaggio email grezzo inviato da `emma` e le intestazioni aggiunte dall'MTA:

```
$ cat /var/spool/mail/dave
From emma@lab1.campus  Sat Nov 16 00:19:13 2019
Return-Path: <emma@lab1.campus>
Received: from lab1.campus (lab1.campus [10.0.3.134])
    by lab2.campus (8.15.2/8.15.2) with SMTP id xAG0G7Y0000595
    for dave@lab2.campus; Sat, 16 Nov 2019 00:17:06 GMT
Date: Sat, 16 Nov 2019 00:16:07 GMT
From: emma@lab1.campus
Message-Id: <201911160017.xAG0G7Y0000595@lab2.campus>
Subject: Recipient MTA Test

Hi Dave, this is a test for your MTA.
```

L'intestazione `Received:` mostra che il messaggio da `lab1.campus` è stato ricevuto direttamente da `lab2.campus`. Per impostazione predefinita, gli MTA accettano solo messaggi a destinatari locali. Il seguente errore si verificherà probabilmente se l'utente `emma` cerca di inviare un'email all'utente `henry` sull'host `lab3.campus`, ma usando l'MTA di `lab2.campus` invece dell'MTA corretto di `lab3.campus`:

```
$ nc lab2.campus 25
220 lab2.campus ESMTP Sendmail 8.15.2/8.15.2; Sat, 16 Nov 2019 00:31:44 GMT
HELO lab1.campus
250 lab2.campus Hello lab1.campus [10.0.3.134], pleased to meet you
MAIL FROM: emma@lab1.campus
250 2.1.0 emma@lab1.campus... Sender ok
RCPT TO: henry@lab3.campus
550 5.7.1 henry@lab3.campus... Relaying denied
```

I codici di risposta SMTP che iniziano con 5, come il messaggio `Relaying denied`, indicano un

errore. Ci sono situazioni legittime in cui il relaying è desiderabile, come quando gli host che inviano e ricevono email non sono sempre connessi: un MTA intermedio può essere configurato per accettare email destinate ad altri host, agendo come un server SMTP *relay* che può inoltrare messaggi tra gli MTA.

La capacità di instradare il traffico e-mail attraverso server SMTP intermedi scoraggia il tentativo di connettersi direttamente all'host indicato dall'indirizzo e-mail del destinatario, come mostrato negli esempi precedenti. Inoltre, gli indirizzi email spesso hanno un nome di dominio come posizione (dopo la `@`), quindi il nome effettivo dell'host MTA corrispondente deve essere recuperato tramite DNS. Pertanto, si raccomanda di delegare il compito di identificare l'host di destinazione appropriato all'MTA locale o al server SMTP remoto, quando si usano caselle di posta remote.

Sendmail fornisce il comando `sendmail` per eseguire molte operazioni relative alla posta elettronica, inclusa l'assistenza nella composizione di nuovi messaggi. Richiede anche all'utente di digitare a mano le intestazioni delle email, ma in un modo più amichevole rispetto all'usare direttamente i comandi SMTP. Quindi un metodo più adeguato per l'utente `emma@lab1.campus` per inviare un messaggio email a `dave@lab2.campus` sarebbe:

```
$ sendmail dave@lab2.campus
From: emma@lab1.campus
To: dave@lab2.campus
Subject: Sender MTA Test

Hi Dave, this is a test for my MTA.
.
```

Anche qui, il punto in una riga, da solo, termina il messaggio. Il messaggio dovrebbe essere immediatamente inviato al destinatario, a meno che l'MTA locale non sia stato in grado di contattare l'MTA remoto. Il comando `mailq`, se eseguito da root, mostrerà tutti i messaggi non consegnati. Se per esempio l'MTA a `lab2.campus` non ha risposto, allora il comando `mailq` elencherà il messaggio non consegnato e la causa del fallimento:

```
# mailq
        /var/spool/mqueue (1 request)
-----Q-ID----- --Size-- -----Q-Time----- ------------Sender/Recipient-----------
xAIK3D9S000453       36 Mon Nov 18 20:03 <emma@lab1.campus>
                (Deferred: Connection refused by lab2.campus.)
                    <dave@lab2.campus>
        Total requests: 1
```

La posizione predefinita per la coda di posta in uscita è `/var/spool/mqueue/`, ma diversi MTA possono usare diverse posizioni nella directory `/var/spool/`. *Postfix*, per esempio, creerà un albero di directory sotto `/var/spool/postfix/` per gestire la coda. Il comando `mailq` è equivalente a `sendmail -bp`, e dovrebbero essere presenti indipendentemente dall'MTA installato nel sistema. Per assicurare la retrocompatibilità, la maggior parte degli MTA fornisce questi tradizionali comandi di amministrazione della posta.

Se l'host primario di destinazione dell'email — quando è fornito da un record DNS MX per il dominio — non è raggiungibile, l'MTA proverà a contattare le voci con priorità inferiore (se ce ne sono specificate). Se nessuna di esse è raggiungibile, il messaggio rimarrà nella coda di posta in uscita locale per essere inviato in seguito. Se configurato per farlo, l'MTA può controllare periodicamente la disponibilità degli host remoti ed eseguire un nuovo tentativo di consegna. Se si usa un MTA compatibile con Sendmail, un nuovo tentativo avrà luogo immediatamente con il comando `sendmail -q`.

Sendmail memorizzerà i messaggi in arrivo in un file chiamato come il proprietario della casella di posta corrispondente, per esempio `/var/spool/mail/dave`. Altri MTA, come Postfix, possono memorizzare i messaggi in arrivo in posizioni come `/var/mail/dave`, ma il contenuto del file è lo stesso. Nell'esempio, il comando `sendmail` è stato usato nell'host del mittente per comporre il messaggio, quindi le intestazioni grezze del messaggio mostrano che l'email ha fatto dei passi in più prima di raggiungere la destinazione finale:

```
$ cat /var/spool/mail/dave
From emma@lab1.campus  Mon Nov 18 20:07:39 2019
Return-Path: <emma@lab1.campus>
Received: from lab1.campus (lab1.campus [10.0.3.134])
    by lab2.campus (8.15.2/8.15.2) with ESMTPS id xAIK7clC000432
    (version=TLSv1.3 cipher=TLS_AES_256_GCM_SHA384 bits=256 verify=NOT)
    for <dave@lab2.campus>; Mon, 18 Nov 2019 20:07:38 GMT
Received: from lab1.campus (localhost [127.0.0.1])
    by lab1.campus (8.15.2/8.15.2) with ESMTPS id xAIK3D9S000453
    (version=TLSv1.3 cipher=TLS_AES_256_GCM_SHA384 bits=256 verify=NOT)
    for <dave@lab2.campus>; Mon, 18 Nov 2019 20:03:13 GMT
Received: (from emma@localhost)
    by lab1.campus (8.15.2/8.15.2/Submit) id xAIK0doL000449
    for dave@lab2.campus; Mon, 18 Nov 2019 20:00:39 GMT
Date: Mon, 18 Nov 2019 20:00:39 GMT
Message-Id: <201911182000.xAIK0doL000449@lab1.campus>
From: emma@lab1.campus
To: dave@lab2.campus
Subject: Sender MTA Test
```

```
Hi Dave, this is a test for my MTA.
```

Dal basso verso l'alto, le linee che iniziano con `Received:` mostrano il percorso del messaggio. Il messaggio è stato inviato dall'utente `emma` con il comando `sendmail dave@lab2.campus` emesso su `lab1.campus`, come indicato dal primo *header* `Received:`. Poi, sempre su `lab1.campus`, l'MTA usa ESMTPS — un superset dell'SMTP, che aggiunge estensioni di crittografia — per inviare il messaggio all'MTA su `lab2.campus`, come indicato dall'ultimo (in alto) header `Received:`

L'MTA finisce il suo lavoro dopo che il messaggio viene salvato nella casella di posta dell'utente. È comune eseguire qualche tipo di filtraggio delle e-mail, come il blocco dello spam o l'applicazione di regole di filtraggio definite dall'utente. Questi compiti sono eseguiti da applicazioni di terze parti che lavorano insieme all'MTA. L'MTA potrebbe per esempio chiamare l'utilità *SpamAssassin* per contrassegnare i messaggi sospetti usando le sue caratteristiche di analisi del testo.

Anche se possibile, non è conveniente leggere direttamente il file della casella di posta. Si raccomanda invece di utilizzare un programma client di posta elettronica (per esempio Thunderbird, Evolution o KMail), che analizzerà il file e gestirà in modo appropriato i messaggi. Tali programmi offrono anche caratteristiche extra, come scorciatoie per azioni comuni, sottocartelle della posta in arrivo, ecc.

Il Comando `mail` e i Mail User Agent (MUA)

È possibile scrivere un messaggio di posta elettronica direttamente nel suo formato grezzo, ma è molto più pratico usare un'applicazione client — conosciuta anche come MUA (*Mail User Agent*) — per velocizzare il processo ed evitare errori. Il MUA si occupa del lavoro "sotto il cofano", cioè, il client di posta elettronica presenta e organizza i messaggi ricevuti e gestisce la corretta comunicazione con l'MTA dopo che l'utente ha composto un'email.

Ci sono molti tipi distinti di Mail User Agent. Applicazioni desktop come *Mozilla Thunderbird* e *Evolution* di Gnome supportano sia account email locali sia remoti. Anche le interfacce *Webmail* possono essere viste come un tipo di MUA, in quanto fanno da intermediari tra l'utente e il sottostante MTA. I client di posta elettronica non sono limitati alle interfacce grafiche: i client di posta elettronica per console sono ampiamente utilizzati per accedere alle caselle di posta non integrate con un'interfaccia grafica e per automatizzare i compiti relativi alla posta elettronica all'interno di script di shell.

Originariamente, il comando Unix `mail` era inteso solo per condividere messaggi tra utenti del sistema locale (il primo comando `mail` risale alla prima edizione di Unix, rilasciata nel 1971). Quando gli scambi di email in rete divennero più importanti, altri programmi furono creati per gestire il nuovo sistema di consegna e gradualmente sostituirono il vecchio programma `mail`.

Al giorno d'oggi, il comando `mail` più comunemente usato è fornito dal pacchetto *mailx*, che è compatibile con tutte le moderne funzionalità di posta elettronica. Nella maggior parte delle distribuzioni Linux, il comando `mail` è solo un collegamento simbolico al comando `mailx`. Altre implementazioni, come il pacchetto *GNU Mailutils*, forniscono fondamentalmente le stesse caratteristiche di `mailx`. Ci sono, tuttavia, leggere differenze tra loro, specialmente per quanto riguarda le opzioni della riga di comando.

Indipendentemente dalla loro implementazione, tutte le varianti moderne del comando `mail` operano in due modalità: *modalità normale* e *modalità di invio*. Se viene fornito un indirizzo email come argomento al comando `mail`, esso entrerà in modalità di invio, altrimenti entrerà in modalità normale (lettura). In questa ultima modalità, i messaggi ricevuti sono elencati con un indice numerico, così l'utente può fare riferimento a essi individualmente quando digita comandi nel prompt interattivo. Per esempio il comando `print 1` può essere usato per visualizzare il contenuto del messaggio numero 1. I comandi interattivi possono essere abbreviati, così comandi come `print`, `delete` o `reply` possono essere sostituiti rispettivamente da `p`, `d` o `r`. Il comando `mail` considererà sempre l'ultimo messaggio ricevuto o l'ultimo visualizzato quando il numero di indice del messaggio è omesso. Il comando `quit` o `q` terminerà il programma.

La modalità *invio* è particolarmente utile per inviare messaggi email automatici. Può essere usata per esempio per inviare un'email all'amministratore di sistema se uno script di manutenzione programmata non riesce ad eseguire il suo compito. In modalità invio, `mail` userà il contenuto dello *standard input* come corpo del messaggio:

```
$ mail -s "Maintenance fail" henry@lab3.campus <<<"The maintenance script failed at `date`"
```

In questo esempio l'opzione `-s` è stata aggiunta per includere un campo *subject* al messaggio. Il corpo del messaggio è stato fornito dal reindirizzamento di *Hereline* allo standard input, ma il contenuto di un file o l'output di un comando potrebbe anche essere convogliato nell'*stdin* del programma. Se nessun contenuto è fornito da un reindirizzamento allo standard input, allora il programma aspetterà che l'utente inserisca il corpo del messaggio. In questo caso, la pressione del tasto `Ctrl` + `D` terminerà il messaggio. Il comando `mail` uscirà immediatamente dopo che il messaggio sarà stato aggiunto alla coda di uscita.

Personalizzazione della Consegna

Per impostazione predefinita, gli account di posta elettronica su un sistema Linux sono associati agli account di sistema standard. Per esempio, se l'utente Carol ha il nome di login `carol` sull'host `lab2.campus`, allora il suo indirizzo email sarà `carol@lab2.campus`. Questa associazione *uno-a-uno* tra account di sistema e caselle di posta può essere estesa con metodi standard forniti dalla maggior parte delle distribuzioni Linux, in particolare il meccanismo di *routing* delle email fornito

dal file `/etc/aliases`.

Un *alias* di posta elettronica è un destinatario di posta elettronica "virtuale" i cui messaggi in ricezione sono reindirizzati a caselle di posta locali esistenti o ad altri tipi di destinazioni di archiviazione o elaborazione dei messaggi. Gli alias sono utili per esempio per mettere i messaggi inviati a `postmaster@lab2.campus` nella mailbox di Carol, che è una normale mailbox locale nel sistema `lab2.campus`. Per fare ciò, la linea `postmaster: carol` dovrebbe essere aggiunta al file `/etc/aliases` in `lab2.campus`. Dopo aver modificato il file `/etc/aliases`, il comando `newaliases` dovrebbe essere eseguito per aggiornare il database degli alias del MTA e rendere effettive le modifiche. I comandi `sendmail -bi` o `sendmail -I` possono anche essere usati per aggiornare il database degli alias.

Gli alias sono definiti uno per riga, nel formato `<alias>: <destinazione>`. Oltre alle normali caselle di posta locali, indicate dal nome utente corrispondente, sono disponibili altri tipi di destinazione:

- Un percorso completo (che inizia con `/`) a un file. I messaggi inviati all'alias corrispondente saranno aggiunti al file.
- Un comando per elaborare il messaggio. La `<destinazione>` deve iniziare con un carattere *pipe* e, se il comando contiene caratteri speciali (come spazi vuoti), deve essere racchiuso tra doppi apici. Per esempio, l'alias `subscribe: |subscribe.sh` in `lab2.campus` inoltrerà tutti i messaggi inviati a `subscribe@lab2.campus` allo standard input del comando `subscribe.sh`. Se sendmail è in esecuzione in *restricted shell mode*, i comandi consentiti — o i collegamenti ad essi — dovrebbero essere in `/etc/smrsh/`.
- Un file di inclusione. Un singolo alias può avere più destinazioni (separate da virgole), quindi può essere più pratico tenerle in un file esterno. La parola chiave `:include:` deve indicare il percorso del file, come in `:include:/var/local/destinations`.
- Un indirizzo esterno. Gli alias possono anche inoltrare i messaggi a indirizzi e-mail esterni.
- Un altro alias.

Un utente locale senza privilegi può definire degli alias per la propria posta elettronica modificando il file `.forward` nella sua directory principale. Poiché gli alias possono influenzare solo la propria casella di posta, è necessaria solo la parte `<destination>`. Per inoltrare tutte le email in arrivo ad un indirizzo esterno, per esempio, l'utente `dave` in `lab2.campus` potrebbe creare il seguente file `~/.forward`:

```
$ cat ~/.forward
emma@lab1.campus
```

Inoltrerà tutti i messaggi email inviati a `dave@lab2.campus` a `emma@lab1.campus`. Come per il file `/etc/aliases`, altre regole di reindirizzamento possono essere aggiunte a `.forward`, una per riga. Tuttavia, il file `.forward` deve essere scrivibile solo dal suo proprietario e non è necessario eseguire il comando `newaliases` dopo averlo modificato. I file che iniziano con un punto non appaiono nei normali elenchi di file, il che potrebbe rendere l'utente ignaro degli alias attivi. Pertanto, è importante verificare se il file esiste quando si diagnosticano problemi di consegna delle email.

Esercizi Guidati

1. Senza ulteriori opzioni o argomenti, il comando `mail henry@lab3.campus` entra nella modalità di input in modo che l'utente possa digitare il messaggio a `henry@lab3.campus`. Dopo aver finito il messaggio, quale tasto chiuderà la modalità di input e invierà l'email?

2. Quale comando può eseguire l'utente root per elencare i messaggi non consegnati che hanno avuto origine sul sistema locale?

3. Come può un utente senza privilegi usare il metodo standard MTA per inoltrare automaticamente tutta la sua posta in arrivo all'indirizzo `dave@lab2.campus`?

 Questa opera è concessa in licenza CC BY-NC-ND 4.0. |

Esercizi Esplorativi

1. Usando il comando `mail` fornito da `mailx`, quale comando invierà un messaggio a `emma@lab1.campus` con il file `logs.tar.gz` come allegato e l'output del comando `uname -a` come corpo della mail?

2. L'amministratore di un servizio di posta elettronica vuole monitorare i trasferimenti di e-mail attraverso la rete, ma non vuole riempire la sua casella di posta con messaggi di prova. Come potrebbe questo amministratore configurare un alias di posta elettronica a livello di sistema per reindirizzare tutte le e-mail inviate all'utente `test` al file `/dev/null`?

3. Quale comando, oltre a `newaliases`, potrebbe essere usato per aggiornare il database degli alias dopo aver aggiunto un nuovo alias a `/etc/aliases`?

Sommario

Questa lezione tratta il ruolo e l'uso dei Mail Transfer Agent nei sistemi Linux. L'MTA fornisce un metodo standard per la comunicazione tra gli account utente e può essere combinato con altri software per fornire funzionalità extra. La lezione ha discusso i seguenti argomenti:

- Concetti sulle tecnologie, caselle di posta e protocolli relativi alla posta elettronica.
- Come gli MTA di Linux scambiano messaggi in rete.
- I client di posta elettronica della console e i MUA (Mail User Agent).
- Aliasing e inoltro delle email locali.

Le tecnologie, i comandi e le procedure affrontate sono state:

- SMTP e protocolli correlati.
- MTA disponibili per Linux: Sendmail, Postfix, qmail, Exim.
- Comandi MTA e MUA: `sendmail` e `mail`.
- File e comandi amministrativi: `mailq`, `/etc/aliases`, `newaliases`, `~/.forward`.

Risposte agli Esercizi Guidati

1. Senza ulteriori opzioni o argomenti, il comando `mail henry@lab3.campus` entra nella modalità di input in modo che l'utente possa digitare il messaggio a `henry@lab3.campus`. Dopo aver finito il messaggio, quale tasto chiuderà la modalità di input e invierà l'email?

   Premendo `Ctrl` + `D` il programma si chiuderà e invierà l'e-mail.

2. Quale comando può eseguire l'utente root per elencare i messaggi non consegnati che hanno avuto origine sul sistema locale?

   Il comando `mailq` o `sendmail -bp`.

3. Come può un utente senza privilegi usare il metodo standard MTA per inoltrare automaticamente tutta la sua posta in arrivo all'indirizzo `dave@lab2.campus`?

   L'utente dovrebbe aggiungere `dave@lab2.campus` al file `~/.forward`.

Risposte agli Esercizi Esplorativi

1. Usando il comando `mail` fornito da `mailx`, quale comando invierà un messaggio a `emma@lab1.campus` con il file `logs.tar.gz` come allegato e l'output del comando `uname -a` come corpo della mail?

   ```
   uname -a | mail -a logs.tar.gz emma@lab1.campus
   ```

2. L'amministratore di un servizio di posta elettronica vuole monitorare i trasferimenti di e-mail attraverso la rete, ma non vuole riempire la sua casella di posta con messaggi di prova. Come potrebbe questo amministratore configurare un alias di posta elettronica a livello di sistema per reindirizzare tutte le e-mail inviate all'utente `test` al file `/dev/null`?

   La linea `test: /dev/null` all'interno di `/etc/aliases` reindirizzerà tutti i messaggi inviati alla casella di posta locale `test` al file `/dev/null`.

3. Quale comando, oltre a `newaliases`, potrebbe essere usato per aggiornare il database degli alias dopo aver aggiunto un nuovo alias a `/etc/aliases`?

   Il comando `sendmail -bi` o `sendmail -I`.



108.4 Gestire stampa e stampanti

Obiettivi LPI di riferimento

LPIC-1 version 5.0, Exam 102, Objective 108.4

Peso

2

Aree di Conoscenza Chiave

- Configurazione di base di CUPS (per stampanti locali e remote).
- Gestire le code di stampa degli utenti.
- Risolvere i problemi generali di stampa.
- Aggiungere e rimuovere lavori dalle code di stampa configurate.

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

- I file di configurazione di CUPS, strumenti e utilità
- `/etc/cups/`
- lpd legacy interface (`lpr`, `lprm`, `lpq`)



108.4 Lezione 1

Certificazione:	LPIC-1
Versione:	5.0
Argomento:	108 Servizi Essenziali di Sistema
Obiettivo:	108.4 Gestire stampa e stampanti
Lezione:	1 di 1

Introduzione

Le dichiarazioni di una “società senza carta” portate dall’avvento dei computer si sono dimostrate, fino a oggi, false. Molte organizzazioni si basano ancora su pagine di informazioni stampate, o “copie cartacee”. Con questo in mente possiamo capire quanto sia importante per un utente di computer sapere come stampare da un sistema, così come per un amministratore che ha bisogno di sapere come mantenere la capacità di un computer di lavorare con le stampanti.

Su Linux, così come su molti altri sistemi operativi *Unix-like*, lo stack software *Common Unix Printing System* (CUPS) permette la stampa e la gestione della stampante da un computer. Ecco uno schema molto semplificato di come un file viene stampato in Linux usando CUPS:

1. Un utente invia un file da stampare.
2. Il demone CUPS, `cupsd`, *accoda* il lavoro di stampa. Questo lavoro di stampa riceve un numero di lavoro da CUPS, insieme alle informazioni su quale coda di stampa contiene il lavoro e il nome del documento da stampare.
3. CUPS utilizza dei *filtri* installati sul sistema per generare un file formattato che la stampante può utilizzare.

4. CUPS invia quindi il file riformattato alla stampante per la stampa.

Vedremo questi passi in modo più dettagliato, così come installare e gestire una stampante in Linux.

Il Servizio CUPS

La maggior parte delle installazioni desktop di Linux avrà i pacchetti CUPS già installati. Nelle installazioni Linux minimali i pacchetti CUPS potrebbero non essere installati, a seconda della distribuzione. Un'installazione di base di CUPS può essere eseguita su un sistema Debian con quanto segue:

```
$ sudo apt install cups
```

Sui sistemi Fedora il processo di installazione è altrettanto facile. Sarà necessario avviare manualmente il servizio CUPS dopo l'installazione su Fedora e altre distribuzioni basate su Red Hat:

```
$ sudo dnf install cups
...
$ sudo systemctl start cups.service
```

Dopo che l'installazione è stata completata, puoi verificare che il servizio CUPS sia in esecuzione con l'uso del comando `systemctl`:

```
$ systemctl status cups.service
● cups.service - CUPS Scheduler
   Loaded: loaded (/lib/systemd/system/cups.service; enabled; vendor preset: enabled)
   Active: active (running) since Thu 2020-06-25 14:35:47 EDT; 41min ago
     Docs: man:cupsd(8)
 Main PID: 3136 (cupsd)
    Tasks: 2 (limit: 1119)
   Memory: 3.2M
   CGroup: /system.slice/cups.service
           ├─3136 /usr/sbin/cupsd -l
           └─3175 /usr/lib/cups/notifier/dbus dbus://
```

Come molti altri demoni Linux, CUPS si basa su un insieme di file di configurazione per le sue operazioni. Di seguito sono elencati i principali d'interesse per l'amministratore di sistema:

`/etc/cups/cupsd.conf`

Questo file contiene le impostazioni di configurazione per il servizio CUPS stesso. Se hai familiarità con il file di configurazione del web server Apache, allora il file di configurazione di CUPS ti sembrerà abbastanza simile, dato che usa una sintassi molto simile. Il file `cupsd.conf` contiene impostazioni per cose come il controllo dell'accesso alle varie code di stampa in uso sul sistema, se l'interfaccia web di CUPS è abilitata o meno, così come il livello di log che il demone utilizzerà.

`/etc/printcap`

Questo è il file *legacy* che veniva usato dal protocollo LPD (*Line Printer Daemon*) prima dell'avvento di CUPS. CUPS creerà ancora questo file sui sistemi per retro compatibilità e spesso è un link simbolico a `/run/cups/printcap`. Ogni riga di questo file contiene una stampante a cui il sistema ha accesso.

`/etc/cups/printers.conf`

Questo file contiene ogni stampante configurata per essere usata dal sistema CUPS. Ogni stampante e la sua coda di stampa associata in questo file è racchiusa in una sezione: `<Printer></Printer>`. Questo file fornisce gli elenchi delle singole stampanti che si trovano in `/etc/printcap`.

WARNING Nessuna modifica al file `/etc/cups/printers.conf` dovrebbe essere fatta dalla linea di comando mentre il servizio CUPS è in esecuzione.

`/etc/cups/ppd/`

Questo non è un file di configurazione ma una directory che contiene i file *PostScript Printer Description* (PPD) per le stampanti che li utilizzano. Le capacità operative di ogni stampante saranno memorizzate in un file PPD (che termina con l'estensione `.ppd`). Questi sono file di testo semplice e seguono un formato specifico.

Il servizio CUPS utilizza anche il logging allo stesso modo del servizio *Apache 2*. I log sono memorizzati in `/var/log/cups/` e contengono un `access_log`, un `page_log` e un `error_log`. l' `access_log` tiene traccia degli accessi all'interfaccia web di CUPS e delle azioni intraprese al suo interno, come la gestione della stampante. Il `page_log` tiene traccia dei lavori di stampa che sono stati inviati alle code di stampa gestite dall'installazione CUPS. L' `error_log` contiene messaggi sui lavori di stampa che sono falliti e altri errori registrati dall'interfaccia web.

Vedremo poi gli strumenti e le utilità che sono utilizzati per gestire il servizio CUPS.

Utilizzo dell'Interfaccia Web

Come detto prima, il file di configurazione `/etc/cups/cupsd.conf` determina se l'interfaccia

web per il sistema CUPS è abilitata. L'opzione di configurazione è nella norma qualche cosa del genere:

```
# Web interface setting...
WebInterface Yes
```

Se l'interfaccia web è abilitata, allora CUPS può essere gestito da un browser collegandosi alla URL predefinita: `http://localhost:631`. Per default un utente del sistema può visualizzare le stampanti e le code di stampa, ma qualsiasi forma di modifica della configurazione richiede un utente con accesso di tipo root . La sezione di configurazione all'interno del file `/etc/cups/cupsd.conf` per limitare l'accesso alle capacità amministrative sarà simile alla seguente:

```
# All administration operations require an administrator to authenticate...
<Limit CUPS-Add-Modify-Printer CUPS-Delete-Printer CUPS-Add-Modify-Class CUPS-Delete-Class
CUPS-Set-Default>
  AuthType Default
  Require user @SYSTEM
  Order deny,allow
</Limit>
```

Ecco una ripartizione di queste opzioni:

`AuthType Default`
: userà una richiesta di autenticazione di base quando un'azione richiede l'accesso di root.

`Require user @SYSTEM`
: indica che un utente con privilegi amministrativi sarà richiesto per l'operazione. Questo potrebbe essere cambiato in `@groupname` dove i membri di `groupname` possono amministrare il servizio CUPS o i singoli utenti potrebbero essere forniti con una lista come in `Require user carol, tim`.

`Order deny,allow`
: funziona come l'opzione di configurazione di Apache 2 dove l'azione è negata per default a meno che un utente (o un membro di un gruppo) sia autenticato.

L'interfaccia web di CUPS può essere disabilitata fermando prima il servizio CUPS, cambiando l'opzione `WebInterface` da `Yes` a `No`, quindi riavviando il servizio CUPS.

L'interfaccia web di CUPS è costruita come un sito web standard con schede di navigazione per

varie sezioni del sistema CUPS. Le schede dell'interfaccia web includono quanto segue:

Home

La pagina iniziale elenca la versione corrente di CUPS che è installata. Inoltre suddivide CUPS in sezioni come:

CUPS for Users

Fornisce una descrizione di CUPS, opzioni a riga di comando per lavorare con stampanti e code di stampa, e un link al forum degli utenti di CUPS.

CUPS for Administrators

Fornisce collegamenti nell'interfaccia per installare e gestire le stampanti e collegamenti a informazioni su come lavorare con le stampanti in una rete.

CUPS for Developers

Fornisce collegamenti per sviluppare per CUPS stesso e per creare file PPD per le stampanti.

Administration

Anche la pagina di amministrazione è suddivisa in sezioni:

Printers

Qui un amministratore può aggiungere nuove stampanti al sistema, localizzare le stampanti connesse al sistema e gestire le stampanti che sono già installate.

Classes

Le *classi* sono un meccanismo in cui le stampanti possono essere aggiunte a gruppi con politiche specifiche. Per esempio, una classe può contenere un gruppo di stampanti che appartengono a un piano specifico di un edificio su cui possono stampare solo gli utenti di un particolare dipartimento. Un'altra classe può avere limitazioni su quante pagine un utente può stampare. Le classi non sono create di default in un'installazione CUPS e devono essere definite da un amministratore. Questa è la sezione dell'interfaccia web di CUPS dove si possono creare e gestire nuove classi.

Jobs

Dove un amministratore può visualizzare tutti i lavori di stampa che sono attualmente in coda per tutte le stampanti che questa installazione CUPS gestisce.

Server

Dove un amministratore può apportare modifiche al file `/etc/cups/cupsd.conf`. Inoltre, ulteriori opzioni di configurazione sono disponibili tramite caselle di controllo come il

permettere alle stampanti connesse a questa installazione CUPS di essere condivise in rete, autenticazione avanzata e permettere l'amministrazione remota della stampante.

Classes

Se le classi di stampanti sono configurate sul sistema, saranno elencate in questa pagina. Ogni classe di stampanti avrà delle opzioni per gestire tutte le stampanti della classe in una volta sola, così come visualizzare tutti i lavori che sono in coda per le stampanti di questa classe.

Help

Questa scheda fornisce i link per tutta la documentazione disponibile di CUPS.

Jobs

La scheda Jobs permette la ricerca di singoli lavori di stampa così come la lista di tutti i lavori di stampa correnti gestiti dal server.

Printers

La scheda Printers elenca tutte le stampanti attualmente gestite dal sistema, nonché una rapida panoramica dello stato di ogni stampante. Ogni stampante elencata può essere cliccata e l'amministratore sarà portato alla pagina dove la singola stampante può essere ulteriormente gestita. Le informazioni per le stampanti in questa scheda provengono dal file `/etc/cups/printers.conf`.

Installare una Stampante

Aggiungere una coda di stampa al sistema è un processo semplice all'interno dell'interfaccia web di CUPS:

1. Clicca sulla scheda **Administration** e poi sul pulsante **Add Printer**.
2. La pagina successiva fornirà varie opzioni a seconda di come la stampante è collegata al sistema. Se si tratta di una stampante locale, seleziona l'opzione più rilevante, come per esempio a quale porta è collegata la stampante o quale software per stampanti di terze parti può essere installato. CUPS cercherà anche di rilevare le stampanti che sono connesse alla rete e le mostrerà qui. Puoi anche scegliere un'opzione di connessione diretta a una stampante di rete a seconda dei protocolli di stampa di rete che la stampante supporta. Seleziona l'opzione appropriata e clicca sul pulsante **Continue**.
3. La pagina successiva ti permetterà di fornire un nome, una descrizione e una posizione (come "back office" o "front desk" ecc.) per la stampante. Se desideri condividere questa stampante in rete, puoi selezionare la casella di controllo per questa opzione anche in questa pagina. Una volta inserite le impostazioni, clicca sul pulsante **Continue**.

4. Nella pagina successiva è possibile selezionare la marca e il modello della stampante. Questo permette a CUPS di cercare, nel suo database installato localmente, i driver e i file PPD più adatti da usare con la stampante. Se hai un file PPD fornito dal produttore della stampante, cerca la sua posizione e selezionalo per usarlo qui. Una volta fatto questo, clicca sul pulsante **Add Printer**.
5. La pagina finale è dove imposterai le opzioni predefinite come la dimensione della pagina che la stampante userà e la risoluzione dei caratteri stampati sulla pagina. Clicca sul pulsante **Set Default Options** e la tua stampante è ora installata sul tuo sistema.

NOTE

Molte installazioni desktop di Linux hanno diversi strumenti che possono essere usati per installare una stampante. Gli ambienti desktop GNOME e KDE hanno le loro applicazioni integrate che possono essere utilizzate per installare e gestire le stampanti. Inoltre, alcune distribuzioni forniscono applicazioni separate per la gestione delle stampanti. Tuttavia, quando si ha a che fare con un'installazione server su cui molti utenti stamperanno, l'interfaccia web di CUPS può fornire i migliori strumenti per questo compito.

La coda di una stampante può anche essere installata usando i comandi *legacy* LPD/LPR. Ecco un esempio usando il comando `lpadmin`:

```
$ sudo lpadmin -p ENVY-4510 -L "office" -v socket://192.168.150.25 -m everywhere
```

Scomponiamo il comando per illustrare le opzioni usate qui:

- Poiché l'aggiunta di una stampante al sistema richiede un utente con privilegi amministrativi, anteponiamo al comando `lpadmin` il comando `sudo`.
- L'opzione `-p` permette di indicare la destinazione dei lavori di stampa. È essenzialmente un nome amichevole per l'utente per sapere dove arriveranno i lavori di stampa. Tipicamente puoi fornire il nome della stampante.
- L'opzione `-L` è la posizione della stampante. Questa opzione è facoltativa ma utile nel caso in cui tu abbia bisogno di gestire un certo numero di stampanti in varie località.
- L'opzione `-v` è per l'*URI* del dispositivo della stampante. L'URI del dispositivo è ciò di cui la coda di stampa CUPS ha bisogno per inviare lavori di stampa renderizzati a una specifica stampante. Nel nostro esempio, stiamo usando una posizione di rete utilizzando l'indirizzo IP fornito.
- L'ultima opzione, `-m`, è impostata su "everywhere". Questo imposta il modello della stampante per CUPS per determinare quale file PPD usare. Nelle versioni moderne di CUPS, è meglio usare "everywhere" in modo che CUPS possa controllare l'URI del dispositivo (impostato con la

precedente opzione `-v`) per determinare automaticamente il corretto file PPD da usare per la stampante. Nelle situazioni moderne, CUPS userà semplicemente *IPP* come spiegato di seguito.

Come detto in precedenza, è meglio lasciare che CUPS determini automaticamente quale file PPD usare per una particolare coda di stampa. Tuttavia, il comando *legacy* `lpinfo` può essere usato per interrogare i file PPD installati localmente per vedere quali sono disponibili. Basta fornire l'opzione `--make-and-model` per la stampante che si desidera installare e l'opzione `-m`:

```
$ lpinfo --make-and-model "HP Envy 4510" -m
hplip:0/ppd/hplip/HP/hp-envy_4510_series-hpijs.ppd HP Envy 4510 Series hpijs, 3.17.10
hplip:1/ppd/hplip/HP/hp-envy_4510_series-hpijs.ppd HP Envy 4510 Series hpijs, 3.17.10
hplip:2/ppd/hplip/HP/hp-envy_4510_series-hpijs.ppd HP Envy 4510 Series hpijs, 3.17.10
drv:///hpcups.crv/hp-envy_4510_series.ppd HP Envy 4510 Series, hpcups 3.17.10
everywhere IPP Everywhere
```

Si noti che il comando `lpinfo` è deprecato. Viene mostrato qui come esempio per elencare quali file di driver di stampa potrebbe usare una stampante.

WARNING

Le versioni future di CUPS elimineranno la necessità di driver specifici e si concentreranno invece sull'uso di IPP (*Internet Printing Protocol*) e dei formati di file standard. L'output del comando precedente illustra questo con la capacità di stampa `everywhere IPP Everywhere`. IPP può eseguire gli stessi compiti per cui viene usato un driver di stampa. IPP, proprio come l'interfaccia web CUPS, utilizza la porta di rete 631 con il protocollo TCP.

Una stampante predefinita può essere impostata usando il comando `lpoptions`. In questo modo, se la maggior parte (o tutti) i lavori di stampa vengono inviati a una particolare stampante, quella specificata con il comando `lpoptions` sarà quella di default. Basta specificare la stampante insieme all'opzione `-d`:

```
$ lpoptions -d ENVY-4510
```

Gestire le Stampanti

Una volta che una stampante è stata installata, un amministratore può usare l'interfaccia web per gestire le opzioni disponibili per la stampante. Un approccio più diretto alla gestione di una stampante è attraverso l'uso del comando `lpadmin`.

È possibile consentire la condivisione di una stampante in rete. Questo può essere ottenuto con l'opzione `printer-is-shared` e specificando la stampante con l'opzione `-p`:

```
$ sudo lpadmin -p FRONT-DESK -o printer-is-shared=true
```

Un amministratore può anche configurare una coda di stampa per accettare solo lavori da utenti specifici con ogni utente separato da una virgola:

```
$ sudo lpadmin -p FRONT-DESK -u allow:carol,frank,grace
```

Allo stesso modo, solo a specifici utenti potrebbe essere negato l'accesso a una specifica coda di stampa:

```
$ sudo lpadmin -p FRONT-DESK -u deny:dave
```

Anche i gruppi di utenti potrebbero essere usati per permettere o negare l'accesso alla coda di una stampante, purché il nome del gruppo sia preceduto da un carattere "at" (@):

```
$ sudo lpadmin -p FRONT-DESK -u deny:@sales,@marketing
```

Una coda di stampa può anche avere una politica specifica di gestione errore nel caso in cui incontri problemi nella stampa di un lavoro. Con l'uso delle politiche, un lavoro di stampa può essere interrotto (`abort-job`) o un altro tentativo di stampa può avvenire in un secondo momento (`retry-job`). Altre politiche includono la capacità di fermare immediatamente la stampante in caso di errore (`stop-printer`) così come la capacità di riprovare il lavoro immediatamente dopo che è stato rilevato un errore (`retry-current-job`). Ecco un esempio in cui la policy della stampante è impostata per interrompere il lavoro di stampa se si verifica un errore sulla stampante `FRONT-DESK`:

```
$ sudo lpadmin -p FRONT-DESK -o printer-error-policy=abort-job
```

Assicurati di rivedere le pagine di manuale per il comando `lpadmin` situate in `lpadmin(8)` per ulteriori dettagli sull'uso di questo comando.

Inviare Lavori di Stampa

Molte applicazioni desktop ti permetteranno di inviare lavori di stampa da una voce di menu o usando la scorciatoia da tastiera `Ctrl` + `p`. Se ti trovi su un sistema Linux che non utilizza un ambiente desktop, puoi ancora inviare file a una stampante per mezzo dei comandi *legacy* LPD/LPR.

Il comando `lpr` ("line printer remote") è usato per inviare un lavoro di stampa alla coda di una stampante. Nella forma più elementare del comando, un nome di file insieme al comando `lpr` è tutto ciò che è richiesto:

```
$ lpr report.txt
```

Il comando precedente invierà il file `report.txt` alla coda di stampa predefinita del sistema (come identificato dal file `/etc/cups/printers.conf`).

Se un'installazione di CUPS ha più stampanti installate, il comando `lpstat` può essere usato per stampare una lista di stampanti disponibili usando l'opzione `-p` e l'opzione `-d` indicherà qual è la stampante predefinita:

```
$ lpstat -p -d
printer FRONT-DESK is idle.  enabled since Mon 03 Aug 2020 10:33:07 AM EDT
printer PostScript_oc0303387803 disabled since Sat 07 Mar 2020 08:33:11 PM EST -
    reason unknown
printer ENVY-4510 is idle.  enabled since Fri 31 Jul 2020 10:08:31 AM EDT
system default destination: ENVY-4510
```

Quindi, nel nostro esempio, il file `report.txt` sarà inviato alla stampante `ENVY-4510`, che è impostata come predefinita. Se il file deve essere stampato su una stampante diversa, specifica la stampante con l'opzione `-P`:

```
$ lpr -P FRONT-DESK report.txt
```

Quando un lavoro di stampa viene sottoposto a CUPS, il demone capirà quale backend è più adatto a gestire il compito. CUPS può fare uso di vari driver di stampa, filtri, monitor di porte hardware e altri software per rendere correttamente il documento. Ci saranno momenti in cui un utente che stampa un documento avrà bisogno di fare modifiche a *come* il documento dovrebbe essere stampato. Molte applicazioni grafiche rendono questo compito piuttosto facile. Ci sono anche opzioni della riga di comando che possono essere utilizzate per cambiare il modo in cui un documento dovrebbe essere stampato. Quando un lavoro di stampa viene inviato tramite la riga di comando, l'opzione `-o` (per "options") può essere usata insieme a termini specifici per regolare il layout del documento per la stampa. Ecco una breve lista di opzioni comunemente usate:

`landscape`

Il documento viene stampato con la pagina ruotata di 90 gradi in senso orario. L'opzione `orientation-requested=4` otterrà lo stesso risultato.

`two-sided-long-edge`

La stampante stamperà il documento in modalità verticale su entrambi i lati della carta, a condizione che la stampante supporti questa capacità.

`two-sided-short-edge`

La stampante stamperà il documento in modalità orizzontale su entrambi i lati della carta, a condizione che la stampante supporti questa capacità.

`media`

La stampante stamperà il lavoro sul supporto specificato. I formati dei supporti disponibili per un lavoro di stampa dipendono dalla stampante. Ecco una lista di formati comuni:

Dimensione	Scopo
`A4`	ISO A4
`Letter`	US Letter
`Legal`	US Legal
`DL`	ISO DL Envelope
`COM10`	US #10 Envelope

`collate`

Raccogli il documento stampato. Questo è utile se hai un documento a più pagine che verrà stampato più di una volta, poiché tutte le pagine di ogni documento verranno stampate in ordine. Imposta questa opzione su `true` per abilitarla o `false` per disabilitarla.

`page-ranges`

Questa opzione può essere usata per selezionare una singola pagina da stampare o un insieme specifico di pagine da stampare da un documento. Un esempio potrebbe essere il seguente: `-o page-ranges=5-7,9,15`. Questo stamperebbe le pagine 5, 6 e 7 e poi le pagine 9 e 15.

`fit-to-page`

Stampa il documento in modo che il file sia scalato per adattarsi alla carta. Se nessuna informazione sulla dimensione della pagina è fornita dal file da stampare, è possibile che il lavoro stampato sia scalato in modo errato e porzioni del documento potrebbero essere scalate fuori dalla pagina o il documento potrebbe risultare troppo piccolo.

`outputorder`

Stampa il documento in ordine `inverso` o `normale` per iniziare la stampa dalla prima pagina. Se una stampante stampa le sue pagine a faccia in giù, l'ordine predefinito è `-o`

`outputorder=normal`, mentre le stampanti che stampano con le pagine rivolte verso l'alto stampano con `-o outputorder=reverse`.

Prendendo un campione delle opzioni di cui sopra, il seguente esempio di comando può essere costruito:

```
$ lpr -P ACCOUNTING-LASERJET -o landscape -o media=A4 -o two-sided-short-edge finance-
report.pdf
```

Più di una copia di un documento può essere stampata usando l'opzione numero nel seguente formato: `-#N` dove `N` è uguale al numero di copie da stampare. Ecco un esempio, con l'opzione `collate`, dove sette copie di un rapporto devono essere stampate sulla stampante predefinita:

```
$ lpr -#7 -o collate=true status-report.pdf
```

Oltre al comando `lpr`, può essere usato anche il comando `lp`. Molte delle opzioni usate con `lpr` possono essere usate anche con il comando `lp`, anche se con alcune differenze. Assicurati di consultare la pagina man di `lp(1)` per riferimento. Ecco come possiamo eseguire il precedente comando `lpr` di esempio usando la sintassi del comando `lp` specificando anche la stampante di destinazione con l'opzione `-d`:

```
$ lp -d ACCOUNTING-LASERJET -n 7 -o collate=true status-report.pdf
```

Gestire i Lavori di Stampa

Come detto prima, ogni lavoro inviato alla coda di stampa riceve un ID lavoro da CUPS. Un utente può visualizzare i lavori di stampa che ha inviato con il comando `lpq`. Passando l'opzione `-a` verranno mostrate le code di tutte le stampanti gestite dall'installazione di CUPS:

```
$ lpq -a
Rank        Owner         Job       File(s)                 Total Size
1st         carol         20        finance-report.pdf      5072 bytes
```

Lo stesso comando `lpstat` usato in precedenza ha anche un'opzione per visualizzare le code di stampa. L'opzione `-o` da sola mostrerà tutte le code di stampa, oppure una coda di stampa può essere specificata per nome:

```
$ lp -o
```

```
ACCOUNTING-LASERJET-4                  carol        19456   Wed 05 Aug 2020 04:29:44 PM EDT
```

L'ID del lavoro di stampa sarà preceduto dal nome della coda dove il lavoro è stato inviato, poi il nome dell'utente che ha inviato il lavoro, la dimensione del file e l'ora in cui è stato inviato.

Se un lavoro di stampa si blocca su una stampante o un utente desidera annullare il suo lavoro di stampa, usa il comando `lprm` insieme all'ID del lavoro trovato dal comando `lpq`:

```
$ lprm 20
```

Tutti i lavori in una coda di stampa potrebbero essere cancellati in una volta sola fornendo solo un trattino `-`:

```
$ lprm -
```

In alternativa, il comando CUPS `cancel` potrebbe anche essere usato da un utente per fermare il suo lavoro di stampa corrente:

```
$ cancel
```

Un lavoro di stampa specifico può essere cancellato attraverso il suo ID lavoro preceduto dal nome della stampante:

```
$ cancel ACCOUNTING-LASERJET-20
```

Un lavoro di stampa può anche essere spostato da una coda di stampa ad un'altra. Questo è spesso utile se una stampante smette di rispondere o il documento da stampare richiede caratteristiche disponibili su una stampante diversa. Nota che questa procedura richiede tipicamente un utente con privilegi elevati. Usando lo stesso lavoro di stampa dell'esempio precedente, potremmo spostarlo nella coda della stampante `FRONT-DESK`:

```
$ sudo lpmove ACCOUNTING-LASERJET-20 FRONT-DESK
```

Rimuovere le Stampanti

Per rimuovere una stampante è spesso utile elencare prima tutte le stampanti che sono attualmente gestite dal servizio CUPS. Questo può essere fatto con il comando `lpstat`:

```
$ lpstat -v
device for FRONT-DESK: socket://192.168.150.24
device for ENVY-4510: socket://192.168.150.25
device for PostScript_oc0303387803: ///dev/null
```

L'opzione `-v` non solo elenca le stampanti ma anche dove (e come) sono collegate. In fase di eliminazione di una stampante, è buona pratica iniziando con il rifiutare tutti i nuovi lavori che vanno alla stampante e fornire una ragione del perché la stampante non accetterà nuovi lavori. Questo può essere fatto nella maniera seguente:

```
$ sudo cupsreject -r "Printer to be removed" FRONT-DESK
```

Notate l'uso di `sudo` poiché questo compito richiede un utente con privilegi elevati.

Per rimuovere una stampante, utilizziamo il comando `lpadmin` con l'opzione `-x` per cancellare la stampante:

```
$ sudo lpadmin -x FRONT-DESK
```

Esercizi Guidati

1. Una nuova stampante è stata appena installata su una stazione di lavoro locale chiamata `office-mgr`. Quale comando potrebbe essere usato per impostare questa stampante come predefinita per questa stazione di lavoro?

2. Quale comando e quale opzione verrebbero usati per determinare quali stampanti sono disponibili per la stampa da una stazione di lavoro?

3. Usando il comando `cancel`, come potreste rimuovere un lavoro di stampa con ID 15 che è bloccato nella coda della stampante chiamata `office-mgr`?

4. Un lavoro di stampa destinato a una stampante che non ha abbastanza carta per stampare l'intero file. Quale comando usereste per spostare il lavoro con ID 2 dalla coda di stampa dalla stampante `FRONT-DESK` alla coda di stampa della stampante `ACCOUNTING-LASERJET`?

Esercizi Esplorativi

Usando il gestore di pacchetti della tua distribuzione, installa i pacchetti `cups` e `printer-driver-cups-pdf`. Si noti che se si sta utilizzando una distribuzione basata su Red Hat (come Fedora) il driver PDF di CUPS è chiamato `cups-pdf`. Installare anche il pacchetto `cups-client` per utilizzare i comandi di stampa in stile System V. Useremo questi pacchetti per fare pratica nella gestione di una stampante CUPS senza installare fisicamente una vera stampante.

1. Verifica che il demone CUPS sia in esecuzione, poi verifica che la stampante PDF sia abilitata e impostata come predefinita.

2. Esegui un comando che stampi il file `/etc/services`. Ora dovresti avere una directory chiamata `PDF` nella tua home directory.

3. Utilizza un comando che disabiliti solo la stampante, poi esegui un comando separato che mostri tutte le informazioni di stato per verificare che la stampante PDF sia disabilitata. Poi prova a stampare una copia del tuo file `/etc/fstab`. Che cosa succede?

4. Ora prova a stampare una copia del file `/etc/fstab` sulla stampante PDF. Che cosa succede?

5. Annulla il lavoro di stampa, poi rimuovi la stampante PDF.

Sommario

Il demone CUPS è una piattaforma ampiamente utilizzata per stampare su stampanti locali e remote. Mentre sostituisce il protocollo *legacy* LPD, fornisce ancora la retro compatibilità per i suoi strumenti.

I file e i comandi discussi in questa lezione sono stati:

`/etc/cups/cupsd.conf`
: Il file di configurazione principale per il servizio CUPS stesso. Questo file controlla anche l'accesso all'interfaccia web di CUPS.

`/etc/printcap`
: Un file *legacy* usato da LPD che contiene una riga per ogni stampante collegata al sistema.

`/etc/cups/printers.conf`
: Il file di configurazione usato da CUPS per le informazioni sulle stampanti.

L'interfaccia web di CUPS in un'installazione predefinita si trova su `http://localhost:631`. Ricorda che la porta di rete predefinita per l'interfaccia web è 631/TCP.

Sono stati discussi anche i seguenti comandi *legacy* LPD/LPR:

`lpadmin`
: usato per installare e rimuovere stampanti e classi di stampanti.

`lpoptions`
: Usato per mostrare e/o modificare le opzioni della stampante.

`lpstat`
: Usato per visualizzare informazioni sullo stato delle stampanti connesse a un'installazione CUPS.

`lpr`
: Usato per inviare lavori di stampa alla coda di una stampante.

`lp`
: Usato per inviare lavori di stampa alla coda di una stampante.

`lpq`
: Questo comando elenca i lavori nella coda di stampa.

`lprm`

Usato per cancellare i lavori di stampa in base all'ID. L'ID di un lavoro può essere ottenuto con l'output del comando `lpq`.

`cancel`

Un'alternativa al comando `lprm` per cancellare i lavori di stampa in base al loro ID.

Assicurati di esaminare le seguenti pagine man per i vari strumenti e utilità di cups: `lpadmin(8)`, `lpoptions(1)`, `lpr(1)`, `lpq(1)`, `lprm(1)`, `cancel(1)`, `lpstat(1)`, `cupsenable(8)` e `cupsaccept(8)`. Si raccomanda inoltre di consultare la documentazione dell'aiuto online su `http://localhost:631/help`.

Risposte agli Esercizi Guidati

1. Una nuova stampante è stata appena installata su una stazione di lavoro locale chiamata `office-mgr`. Quale comando potrebbe essere usato per impostare questa stampante come predefinita per questa stazione di lavoro?

   ```
   $ lpoptions -d office-mgr
   ```

2. Quale comando e quale opzione verrebbero usati per determinare quali stampanti sono disponibili per la stampa da una stazione di lavoro?

   ```
   $ lpstat -p
   ```

   L'opzione `-p` elenca tutte le stampanti disponibili e se sono abilitate alla stampa.

3. Usando il comando `cancel`, come potreste rimuovere un lavoro di stampa con ID 15 che è bloccato nella coda della stampante chiamata `office-mgr`?

   ```
   $ cancel office-mgr-15
   ```

4. Hai un lavoro di stampa destinato a una stampante che non ha abbastanza carta per stampare l'intero file. Quale comando usereste per spostare il lavoro con ID 2 dalla coda di stampa dalla stampante `FRONT-DESK` alla coda di stampa della stampante `ACCOUNTING-LASERJET`?

   ```
   $ sudo lpmove FRONT-DESK-2 ACCOUNTING-LASERJET
   ```

Risposte agli Esercizi Esplorativi

Usando il gestore di pacchetti della tua distribuzione, installa i pacchetti `cups` e `printer-driver-cups-pdf`. Si noti che se si sta utilizzando una distribuzione basata su Red Hat (come Fedora) il driver PDF di CUPS è chiamato `cups-pdf`. Installare anche il pacchetto `cups-client` per utilizzare i comandi di stampa in stile System V. Useremo questi pacchetti per fare pratica nella gestione di una stampante CUPS senza installare fisicamente una vera stampante.

1. Verifica che il demone CUPS sia in esecuzione, poi verifica che la stampante PDF sia abilitata e impostata come predefinita.

   Un metodo per controllare la disponibilità e lo stato della stampante PDF sarebbe quello di eseguire il seguente comando:

   ```
   $ lpstat -p -d

   printer PDF is idle.  enabled since Thu 25 Jun 2020 02:36:07 PM EDTi

   system default destination: PDF
   ```

2. Esegui un comando che stampi il file `/etc/services`. Ora dovresti avere una directory chiamata `PDF` nella tua home directory.

   ```
   $ lp -d PDF /etc/services
   ```

   funzionerebbe. Ora hai una versione PDF di questo file nella directory PDF.

3. Utilizza un comando che disabiliti solo la stampante, poi esegui un comando separato che mostri tutte le informazioni di stato per verificare che la stampante PDF sia disabilitata.

   ```
   $ sudo cupsdisable PDF
   ```

   disabiliterà la stampante.

   Poi esegui il comando `lpstat -t` per ottenere un elenco completo delle condizioni della stampante. Dovrebbe essere simile al seguente output:

   ```
   $ scheduler is running
   ```

```
system default destination: PDFi

device for PDF: cups-pdf:/

PDF accepting requests since Wed 05 Aug 2020 04:19:15 PM EDTi

printer PDF disabled since Wed 05 Aug 2020 04:19:15 PM EDT -

Paused
```

4. Ora prova a stampare una copia del file `/etc/fstab` sulla stampante PDF. Che cosa succede?

   Dopo aver tentato il comando `lp -d PDF /etc/fstab` dovresti ottenere un output che mostra le informazioni sull'ID del lavoro. Tuttavia, se controlli la cartella PDF nella tua home directory, il nuovo file non c'è. Puoi quindi controllare la coda di stampa con il comando `lpstat -o`, e troverai il tuo lavoro elencato lì.

5. Annulla il lavoro di stampa, poi rimuovi la stampante PDF.

   Usando l'output del precedente comando `lp`, usa il comando `cancel` per cancellare il lavoro. Per esempio:

   ```
   $ cancel PDF-4
   ```

   Poi esegui il comando `lpstat -o` per verificare che il lavoro sia stato cancellato.

   Rimuovi la stampante PDF con il seguente comando: `sudo lpadmin -x PDF`. Poi verifica che la stampante sia stata rimossa: `lpstat -a`.



Linux Professional Institute

Argomento 109: Fondamenti di Networking



109.1 Fondamenti dei protocolli Internet

Obiettivi LPI di riferimento

LPIC-1 version 5.0, Exam 102, Objective 109.1

Peso

4

Aree di Conoscenza Chiave

- Dimostrare una comprensione delle maschere di sottorete e della notazione CIDR.
- Conoscenza delle differenze tra indirizzi IP privati e pubblici.
- Conoscenza delle porte e dei servizi TCP e UDP più comuni (20, 21, 22, 23, 25, 53, 80, 110, 123, 139, 143, 161, 162, 389, 443, 465, 514, 636, 993, 995).
- Conoscenza delle differenze e delle principali caratteristiche di UDP, TCP e ICMP.
- Conoscenza delle principali differenze tra IPv4 e IPv6.
- Conoscenza delle caratteristiche di base di IPv6.

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

- `/etc/services`
- IPv4, IPv6
- Subnetting
- TCP, UDP, ICMP

109.1 Lezione 1

Certificazione:	LPIC-1
Versione:	5.0
Argomento:	109 Fondamenti di Networking
Obiettivo:	109.1 Fondamenti dei protocolli Internet
Lezione:	1 di 2

Introduzione

Il TCP/IP (*Transmission Control Protocol/Internet Protocol*) è uno *stack* di protocolli usato per permettere la comunicazione tra computer. Nonostante il nome, lo stack consiste di diversi protocolli come IP, TCP, UDP, ICMP, DNS, SMTP, ARP e altri.

IP (Internet Protocol)

L'IP è il protocollo responsabile dell'indirizzamento logico di un host, che permette di inviare il pacchetto da un host all'altro. Per questo ad ogni dispositivo della rete viene assegnato un indirizzo IP unico, ed è possibile assegnare più di un indirizzo allo stesso dispositivo.

Nella versione 4 del protocollo IP, solitamente chiamata IPv4, l'indirizzo è formato da un insieme di 32 bit separati in 4 gruppi di 8 bit, rappresentati in forma decimale, chiamati "ottetti". Per esempio:

Formato binario (4 gruppi di 8 bit)

```
11000000.10101000.00001010.00010100
```

Formato decimale

`192.168.10.20`

In IPv4, i valori per ogni ottetto possono andare da 0 a 255, che è l'equivalente di 11111111 in formato binario.

Classi di Indirizzi

Teoricamente, gli indirizzi IP sono separati in classi, che sono definite dall'intervallo del primo ottetto come mostrato nella tabella sottostante:

Classe	Primo Ottetto	Intervallo	Esempio
A	1-126	`1.0.0.0` – `126.255.255.255`	`10.25.13.10`
B	128-191	`128.0.0.0` – `191.255.255.255`	`141.150.200.1`
C	192-223	`192.0.0.0` – `223.255.255.255`	`200.178.12.242`

IP Pubblici e Privati

Come accennato in precedenza, affinché la comunicazione avvenga, ogni dispositivo in rete deve essere associato ad almeno un indirizzo IP unico. Tuttavia, se ogni dispositivo connesso a Internet nel mondo avesse un indirizzo IP unico, non ci sarebbero abbastanza IP (v4) per tutti. Per questo, sono stati definiti gli indirizzi IP *privati*.

Gli IP privati sono intervalli di indirizzi IP che sono stati riservati per l'uso nelle reti interne (private) di aziende, istituzioni, case, ecc. All'interno della stessa rete, l'uso di un indirizzo IP rimane unico. Tuttavia, lo stesso indirizzo IP privato può essere usato all'interno di qualsiasi rete privata.

Così, su Internet abbiamo un traffico di dati che utilizza indirizzi IP pubblici, che sono riconoscibili e instradabili su Internet, mentre all'interno delle reti private vengono utilizzati questi intervalli di IP riservati. Il *router* è responsabile della conversione del traffico dalla rete privata alla rete pubblica e viceversa.

Gli intervalli di IP privati, separati per classi, possono essere visti nella tabella sottostante:

Classe	Primo Ottetto	Intervallo	IP Privati
A	1-126	`1.0.0.0` – `126.255.255.255`	`10.0.0.0` – `10.255.255.255`
B	128-191	`128.0.0.0` – `191.255.255.255`	`172.16.0.0` – `172.31.255.255`
C	192-223	`192.0.0.0` – `223.255.255.255`	`192.168.0.0` – `192.168.255.255`

Conversione dal Formato Decimale a quello Binario

Per gli argomenti di questa sezione, è importante sapere come convertire gli indirizzi IP tra i formati binario e decimale.

La conversione dal formato decimale a quello binario avviene attraverso divisioni consecutive per 2. Come esempio, convertiamo il valore 105 attraverso i seguenti passi:

1. Dividendo il valore 105 per 2 abbiamo:

```
105/2
Quoziente = 52
Resto = 1
```

2. Dividiamo il quoziente in modo sequenziale per 2, finché il quoziente finale sia uguale a 1:

```
52/2
Resto = 0
Quoziente = 26
```

```
26/2
Resto = 0
Quoziente = 13
```

```
13/2
Resto = 1
Quoziente = 6
```

```
6/2
```

```
Resto = 0
Quoziente = 3
```

```
3/2
Resto = 1
Quoziente = 1
```

3. Raggruppa l'ultimo quoziente seguito dal resto di tutte le divisioni:

```
1101001
```

4. Riempite da sinistra di zeri fino a completare gli 8 bit:

```
01101001
```

5. Alla fine, abbiamo che il valore 105 in decimale è uguale a 01101001 in binario.

Conversione dal Formato Binario a quello Decimale

In questo esempio, useremo il valore binario `10110000`.

1. Ogni bit è associato a un valore con una potenza base di due. Le potenze partono da 0 e vengono incrementate da destra a sinistra. In questo esempio avremo:

1	0	1	1	0	0	0	0
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0

2. Quando il bit è 1, assegniamo il valore della rispettiva potenza, quando il bit è 0 il risultato è 0.

1	0	1	1	0	0	0	0
2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
128	0	32	16	0	0	0	0

3. Sommiamo tutti i valori:

   128 + 32 + 16 = 176

4. Così, 10110000 in binario è uguale a 176 in decimale.

Netmask

La maschera di rete (o *netmask*) è usata insieme all'indirizzo IP per determinare quale parte dell'IP rappresenta la rete e quale l'host. Ha lo stesso formato dell'indirizzo IP, cioè ci sono 32 bit in 4 gruppi di 8. Per esempio:

Decimale	Binario	CIDR
`255.0.0.0`	`11111111.00000000.00000000.00000000`	`8`
`255.255.0.0`	`11111111.11111111.00000000.00000000`	`16`
`255.255.255.0`	`11111111.11111111.11111111.00000000`	`24`

Usare la maschera `255.255.0.0` come esempio, indica che nell'IP associato, i primi 16 bit (2 primi decimali) identificano la rete/sottorete e gli ultimi 16 bit sono usati per identificare univocamente gli host all'interno della rete.

Il CIDR (*Classless Inter-Domain Routing*) menzionato sopra è legato a una notazione semplificata della maschera, che indica il numero di bit (1) associato alla rete/sottorete. Questa notazione è comunemente usata per sostituire il formato decimale, per esempio `/24` invece di `255.255.255.0`.

È interessante notare che ogni classe di IP ha una maschera standard, come la seguente:

Classe	Primo Ottetto	Intervallo	Maschera Default
A	1-126	`1.0.0.0` – `126.255.255.255`	`255.0.0.0 / 8`
B	128-191	`128.0.0.0` – `191.255.255.255`	`255.255.0.0 / 16`
C	192-223	`192.0.0.0` – `223.255.255.255`	`255.255.255.0 / 24`

Tuttavia, questo modello non significa che questa è la maschera che sarà sempre usata. È possibile utilizzare qualsiasi maschera con qualsiasi indirizzo IP, come vedremo di seguito.

Ecco alcuni esempi di utilizzo di IP e Maschere:

```
192.168.8.12 / 255.255.255.0 / 24
```

Intervallo

`192.168.8.0` - `192.168.8.255`

Indirizzo di Rete

`192.168.8.0`

Indirizzo di Broadcast

`192.168.8.255`

Host

`192.168.8.1` - `192.168.8.254`

In questo caso le prime 3 cifre (i primi 24 bit) dell'indirizzo IP definiscono la rete e l'ultima cifra identifica gli indirizzi degli host, cioè la gamma di questa rete va da `192.168.8.0` a `192.168.8.255`.

Ora definiamo due concetti importanti: Ogni rete/sottorete ha 2 indirizzi riservati, il primo indirizzo nella gamma è chiamato *indirizzo di rete*. In questo caso `192.168.8.0`, che è usato per identificare la rete/sottorete stessa. L'ultimo indirizzo nell'intervallo è chiamato l'indirizzo *broadcast*, in questo caso `192.168.8.255`. Questo indirizzo di destinazione è usato per inviare lo stesso messaggio (pacchetto) a tutti gli host IP su quella rete/sottorete.

Gli indirizzi di rete e di broadcast non possono essere usati dalle macchine in rete. Pertanto, la lista degli IP che possono essere effettivamente configurati va da `192.168.8.1` a `192.168.8.254`.

Ora l'esempio dello stesso IP, ma con una maschera diversa:

```
192.168.8.12 / 255.255.0.0 / 16
```

Intervallo

`192.168.0.0` - `192.168.255.255`

Indirizzo di Rete

`192.168.0.0`

Indirizzo di Broadcast

`192.168.255.255`

Host

`192.168.0.1` – `192.168.255.254`

Guarda come una maschera diversa cambia la gamma di IP che sono all'interno della stessa rete/sottorete.

La suddivisione delle reti attraverso le maschere non è limitata ai valori predefiniti (8, 16, 24). Possiamo creare suddivisioni a piacere, aggiungendo o togliendo bit nell'identificazione della rete, creando le nuove sottoreti.

Per esempio:

```
11111111.11111111.11111111.00000000 = 255.255.255.0 = 24
```

Se vogliamo suddividere la rete di cui sopra in 2, basta aggiungere un altro bit all'identificazione della rete nella maschera, in questo modo:

```
11111111.11111111.11111111.10000000 = 255.255.255.128 = 25
```

Abbiamo quindi le seguenti sottoreti:

```
192.168.8.0   - 192.168.8.127
192.168.8.128 - 192.168.8.255
```

Se aumentiamo ulteriormente la suddivisione della rete:

```
11111111.11111111.11111111.11000000 = 255.255.255.192 = 26
```

Avremo:

```
192.168.8.0   - 192.168.8.63
192.168.8.64  - 192.168.8.127
192.168.8.128 - 192.168.8.191
192.168.8.192 - 192.168.8.255
```

Si noti che in ogni sottorete avremo gli indirizzi di rete riservati (il primo dell'intervallo) e di broadcast (l'ultimo dell'intervallo), quindi più la rete è suddivisa, meno IP possono essere effettivamente utilizzati dagli host.

Identificare gli Indirizzi di Rete e di Broadcast

Attraverso un indirizzo IP e una maschera, possiamo identificare l'indirizzo di rete e l'indirizzo broadcast, e quindi definire la gamma di IP per la rete/sottorete.

L'indirizzo di rete si ottiene usando un "Logical AND" tra l'indirizzo IP e la maschera nei loro formati binari. Prendiamo per esempio come IP `192.168.8.12` e come maschera `255.255.255.192`.

Convertendo dal formato decimale a quello binario, come abbiamo visto prima, abbiamo:

```
11000000.10101000.00001000.00001100 (192.168.8.12)
11111111.11111111.11111111.11000000 (255.255.255.192)
```

Con l'"AND logico", abbiamo 1 e 1 = 1, 0 e 0 = 0, 1 e 0 = 0, quindi:

```
11000000.10101000.00001000.00001100 (192.168.8.12)
11111111.11111111.11111111.11000000 (255.255.255.192)
11000000.10101000.00001000.00000000
```

Quindi l'indirizzo di rete per quella sottorete è `192.168.8.0`.

Ora per ottenere l'indirizzo di broadcast dobbiamo usare l'indirizzo di rete dove tutti i bit relativi all'indirizzo dell'host sono a 1:

```
11000000.10101000.00001000.00000000 (192.168.8.0)
11111111.11111111.11111111.11000000 (255.255.255.192)
11000000.10101000.00001000.00111111
```

L'indirizzo di broadcast è quindi: `192.168.8.63`.

In conclusione, abbiamo:

```
192.168.8.12 / 255.255.255.192 / 26
```

Intervallo

`192.168.8.0` - `192.168.8.63`

Indirizzo di Rete

`192.168.8.0`

Indirizzo di Broadcast

`192.168.8.63`

Host

`192.168.8.1` – `192.168.8.62`

Rotta Predefinita

Come abbiamo visto finora, le macchine che si trovano nella stessa rete/sottorete logica possono comunicare direttamente tramite il protocollo IP.

Ma consideriamo l'esempio qui sotto:

Network 1

`192.168.10.0/24`

Network 2

`192.168.200.0/24`

In questo caso, la macchina `192.168.10.20` non può inviare direttamente un pacchetto alla `192.168.200.100`, poiché sono su reti logiche diverse.

Per permettere questa comunicazione si usa un router (o un insieme di router). Un router in questa configurazione può anche essere chiamato un gateway poiché fornisce un passaggio tra due reti. Questo dispositivo ha accesso a entrambe le reti perché è configurato con gli IP di entrambe le reti. Per esempio `192.168.10.1` e `192.168.200.1`, e per questo motivo riesce ad essere l'intermediario in questa comunicazione.

Per abilitarlo, ogni host della rete deve aver configurato quello che viene chiamato *default route*. La rotta predefinita indica l'IP a cui devono essere inviati tutti i pacchetti la cui destinazione è un IP che non fa parte della rete logica dell'host.

Nell'esempio sopra, la rotta predefinita per le macchine sulla rete `192.168.10.0/24` sarà l'IP `192.168.10.1`, che è l'IP del router/gateway, mentre la rotta predefinita per le macchine sulla rete `192.168.200.0/24` sarà `192.168.200.1`.

Il percorso predefinito è anche utilizzato in modo che le macchine della rete privata (LAN) abbiano accesso a Internet (WAN), attraverso un router.

Esercizi Guidati

1. Utilizzando l'IP `172.16.30.230` e la netmask `255.255.255.224`, identifica:

La notazione CIDR per la maschera di rete	
Indirizzo di rete	
Indirizzo di broadcast	
Numero di IP che possono essere usati per gli host in questa sottorete	

2. Quale impostazione è necessaria su un host per permettere una comunicazione IP con un host in una rete logica diversa?

Esercizi Esplorativi

1. Perché gli intervalli IP che iniziano con `127` e l'intervallo dopo `224` non sono inclusi nelle classi di indirizzi IP A, B o C?

2. Uno dei campi di un pacchetto IP che è molto importante è il TTL (*Time To Live*). Qual è la funzione di questo campo e come funziona?

3. Spiega la funzione del NAT e quando viene utilizzato.

Summario

Questa lezione ha trattato i concetti principali del protocollo IPv4, che è responsabile della comunicazione tra gli host di una rete.

Sono state studiate anche le principali operazioni che devi conoscere per convertire gli IP in diversi formati e per poter analizzare ed eseguire le configurazioni logiche su reti e sottoreti.

Sono stati affrontati i seguenti argomenti:

- Classi di indirizzi IP
- IP Pubblici e Privati
- Come convertire gli IP dal formato decimale a quello binario e viceversa
- La maschera di rete (netmask)
- Come identificare gli indirizzi di rete e di broadcast da IP e netmask
- La rotta predefinita

 Questa opera è concessa in licenza CC BY-NC-ND 4.0. |

Risposte agli Esercizi Guidati

1. Utilizzando l'IP `172.16.30.230` e la netmask `255.255.255.224`, identifica:

La notazione CIDR per la maschera di rete	`27`
Indirizzo di rete	`172.16.30.224`
Indirizzo di broadcast	`172.16.30.255`
Numero di IP che possono essere usati per gli host in questa sottorete	`30`

2. Quale impostazione è necessaria su un host per permettere una comunicazione IP con un host in una rete logica diversa?

   La rotta predefinita

Risposte agli Esercizi Esplorativi

1. Perché gli intervalli IP che iniziano con `127` e l'intervallo dopo `224` non sono inclusi nelle classi di indirizzi IP A, B o C?

   La gamma che inizia con `127` è riservata agli indirizzi di *loopback*, usati per test e comunicazioni interne tra processi, come l'indirizzo `127.0.0.1`. Inoltre, gli indirizzi sopra `224` non sono usati come indirizzi host, ma per il *multicast* e altri scopi.

2. Uno dei campi di un pacchetto IP che è molto importante è il TTL (*Time To Live*). Qual è la funzione di questo campo e come funziona?

   Il TTL definisce la durata di un pacchetto. Questo è implementato attraverso un contatore in cui il valore iniziale definito all'origine è diminuito in ogni gateway/router attraverso cui passa il pacchetto, che è anche chiamato "hop". Se questo contatore raggiunge lo 0 il pacchetto viene scartato.

3. Spiega la funzione del NAT e quando viene utilizzato.

   La funzione NAT (*Network Address Translation*) permette agli host di una rete interna, che usa IP privati, di avere accesso a Internet come se fossero collegati direttamente ad essa, con l'IP pubblico usato sul gateway.



109.1 Lezione 2

Certificazione:	LPIC-1
Versione:	5.0
Argomento:	109 Fondamenti di Networking
Obiettivo:	109.1 Fondamenti dei protocolli Internet
Lezione:	2 di 2

Introduzione

Abbiamo visto che lo stack TCP/IP è composto da una serie di diversi protocolli. Finora abbiamo studiato il protocollo IP, che permette la comunicazione tra macchine attraverso indirizzi IP, maschere, rotte, ecc.

Affinché un host possa accedere a un servizio disponibile su un altro host, oltre al protocollo di indirizzamento IP a livello di rete, è anche necessario utilizzare un *protocollo a livello di trasporto* come i protocolli TCP e UDP.

Questi protocolli realizzano questa comunicazione attraverso porte di rete. Così, oltre a definire un IP di origine e di destinazione, le porte di origine e di destinazione vengono utilizzate per accedere a un servizio.

La porta è identificata da un campo di 16 bit che fornisce un limite di 65.535 porte possibili. I servizi (destinazione) usano generalmente le porte da 1 a 1023, chiamate anche *porte privilegiate*. L'origine della connessione usa invece la gamma di porte da 1024 a 65.535, chiamate *porte non privilegiate*, o porte *socket*.

Le porte usate da ogni tipo di servizio sono standardizzate e controllate dalla IANA (*Internet*

Assigned Numbers Authority). Questo significa che, di norma, su un sistema la porta 22 è usata dal servizio SSH, la porta 80 dal servizio HTTP e così via.

La tabella seguente contiene i principali servizi e le loro rispettive porte.

Port	**Service**
20	FTP (data)
21	FTP (control)
22	SSH (Secure Socket Shell)
23	Telnet (Remote connection without encryption)
25	SMTP (Simple Mail Transfer Protocol), Sending Mails
53	DNS (Domain Name System)
80	HTTP (Hypertext Transfer Protocol)
110	POP3 (Post Office Protocol), Receiving Mails
123	NTP (Network Time Protocol)
139	Netbios
143	IMAP (Internet Message Access Protocol), Accessing Mails
161	SNMP (Simple Network Management Protocol)
162	SNMPTRAP, SNMP Notifications
389	LDAP (Lightweight Directory Access Protocol)
443	HTTPS (Secure HTTP)
465	SMTPS (Secure SMTP)
514	RSH (Remote Shell)
636	LDAPS (Secure LDAP)
993	IMAPS (Secure IMAP)
995	POP3S (Secure POP3)

Su un sistema Linux, le porte di servizio standard sono elencate nel file `/etc/services`.

L'identificazione della porta di destinazione desiderata in una connessione è eseguita usando il carattere `:` (due punti) dopo l'indirizzo IPv4. Così, quando si cerca di accedere al servizio HTTPS

che è servito dall'host IP `200.216.10.15`, il client deve inviare la richiesta alla destinazione `200.216.10.15:443`.

I servizi elencati sopra, e tutti gli altri, usano un protocollo di trasporto secondo le caratteristiche richieste dal servizio, dove TCP e UDP sono i principali.

Transmission Control Protocol (TCP)

TCP è un protocollo di trasporto orientato alla connessione. Ciò significa che una connessione viene stabilita tra il client attraverso la porta socket e il servizio attraverso la porta standard del servizio. Il protocollo si assicura che tutti i pacchetti siano consegnati correttamente, verificandone l'integrità e l'ordine di consegna, compresa la ritrasmissione dei pacchetti persi a causa di errori di rete.

Così facendo le applicazioni non hanno bisogno di implementare questo controllo del flusso di dati perché è già garantito dal protocollo TCP.

User Datagram Protocol (UDP)

UDP stabilisce una connessione tra il client e il servizio, ma non controlla la trasmissione dei dati di questa connessione. In altre parole, non controlla se i pacchetti sono stati persi, se l'ordine di consegna sia stato rispettato e così via. In questo caso quindi le applicazioni diventano responsabili dell'eventuale implementazione di questi controlli.

Poiché c'è meno controllo, UDP permette una migliore performance nel flusso di dati: una necessità rilevante per alcuni tipi di servizi.

Internet Control Message Protocol (ICMP)

ICMP è un protocollo di livello di rete nello *stack* TCP/IP e la sua funzione principale è quella di analizzare e controllare gli elementi della rete, rendendo possibile, per esempio:

- Controllo del volume di traffico
- Rilevamento delle destinazioni non raggiungibili
- Reindirizzamento delle rotte
- Contro dello stato degli host remoti

È il protocollo usato dal comando `ping`, che sarà studiato in un altro capitolo.

IPv6

Finora abbiamo studiato la versione 4 del protocollo IP, cioè IPv4. Questa è stata la versione standard utilizzata in tutti gli ambienti di rete e Internet. Tuttavia ha delle limitazioni soprattutto per quanto riguarda il numero di indirizzi disponibili e in considerazione del sempre crescente numero di dispositivi in qualche modo connessi a Internet (vedi *IoT*), sta diventando sempre più comune utilizzare la versione 6 del protocollo IP, comunemente scritta come IPv6.

IPv6 porta una serie di cambiamenti, nuove implementazioni e caratteristiche, così come una nuova rappresentazione dell'indirizzo stesso.

Ogni indirizzo IPv6 è formato da 128 bit, divisi in 8 gruppi di 16 bit ciascuno, rappresentati da valori esadecimali.

Per esempio:

```
2001:0db8:85a3:08d3:1319:8a2e:0370:7344
```

Abbreviazioni

IPv6 definisce dei modi per accorciare gli indirizzi nella loro rappresentazione scritta. Esaminiamo il seguente indirizzo:

```
2001:0db8:85a3:0000:0000:0000:0000:7344
```

La prima possibilità è quella di ridurre le stringhe da `0000` a solo `0`, con il risultato di:

```
2001:0db8:85a3:0:0:0:0:7344
```

Inoltre, nel caso di stringhe di gruppo con un valore di `0`, esse possono essere omesse, come segue:

```
2001:0db8:85a3::7344
```

Tuttavia, quest'ultima abbreviazione può essere fatta solo una volta nell'indirizzo. Vedi l'esempio:

```
2001:0db8:85a3:0000:0000:1319:0000:7344

2001:0db8:85a3:0:0:1319:0:7344
```

```
2001:0db8:85a3::1319:0:7344
```

Tipi di Indirizzo IPv6

IPv6 classifica gli indirizzi in 3 tipi:

Unicast

Identifica una singola interfaccia di rete. Per impostazione predefinita, i 64 bit a sinistra identificano la rete e i 64 bit a destra identificano l'interfaccia.

Multicast

Identifica un insieme di interfacce di rete. Un pacchetto inviato a un indirizzo *multicast* sarà inviato a tutte le interfacce che appartengono a quel gruppo. Anche se simile, non deve essere confuso con *broadcast*, che *non* esiste nel protocollo IPv6.

Anycast

Identifica anche un insieme di interfacce sulla rete, ma il pacchetto inoltrato a un indirizzo *anycast* sarà consegnato solo a un indirizzo in quell'insieme, non a tutti.

Differenze tra IPv4 e IPv6

Oltre all'indirizzo si possono evidenziare altre differenze tra le versioni 4 e 6 del protocollo IP. Eccone alcune:

- Le porte di servizio seguono gli stessi standard e protocolli (TCP, UDP), la differenza è solo nella rappresentazione dell'IP e del set di porte. In IPv6 l'indirizzo IP deve essere protetto da `[]` (parentesi quadre):

  IPv4

  `200.216.10.15:443`

  IPv6

  `[2001:0db8:85a3:08d3:1319:8a2e:0370:7344]:443`

- IPv6 non implementa la funzione di broadcast esattamente come esiste in IPv4. Tuttavia lo stesso risultato può essere ottenuto inviando il pacchetto all'indirizzo `ff02::1`, raggiungendo tutti gli host della rete locale. Qualcosa di simile all'uso di `224.0.0.1` su IPv4 per il *multicasting* come destinazione.
- Attraverso la funzione SLAAC (*Stateless Address Autoconfiguration*), gli host IPv6 sono in grado di autoconfigurarsi.

- Il campo TTL (*Time to Live*) di IPv4 è stato sostituito dal "*Hop Limit*" nell'intestazione IPv6.
- Tutte le interfacce IPv6 hanno un indirizzo locale, chiamato indirizzo *link-local*, con prefisso `fe80::/10`.
- IPv6 implementa il *Neighbor Discovery Protocol* (NDP), che è simile all'ARP usato da IPv4, ma con molte più funzionalità.

Esercizi Guidati

1. Quale porta è quella predefinita per il protocollo SMTP?

2. Quante porte sono disponibili in un sistema?

3. Quale protocollo di trasporto assicura che tutti i pacchetti siano consegnati correttamente, verificando l'integrità e l'ordine dei pacchetti?

4. Quale tipo di indirizzo IPv6 viene usato per inviare un pacchetto a tutte le interfacce che appartengono a un gruppo di host?

Esercizi Esplorativi

1. Cita 4 esempi di servizi che usano il protocollo TCP di default.

2. Qual è il nome del campo sul pacchetto di intestazione IPv6 che ha lo stesso scopo del *TTL* su IPv4?

3. Che tipo di informazioni è in grado di scoprire il *Neighbor Discovery Protocol* (NDP)?

Sommario

Questa lezione ha trattato i principali protocolli di trasporto e i servizi utilizzati sullo *stack* TCP/IP.

Un altro argomento importante ha riguardato la versione 6 del protocollo IP, compresi gli indirizzi IPv6 e le principali differenze con IPv4.

Sono stati affrontati i seguenti argomenti:

- La correlazione tra numeri di porta e servizi
- TCP (Transmission Control Protocol)
- UDP (User Datagram Protocol)
- ICMP (Internet Control Message Protocol)
- L'indirizzo IPv6 e come può essere abbreviato
- Tipi di indirizzi IPv6
- Principali differenze tra IPv4 e IPv6

Risposte agli Esercizi Guidati

1. Quale porta è quella predefinita per il protocollo SMTP?

   25.

2. Quante porte sono disponibili in un sistema?

   65535.

3. Quale protocollo di trasporto assicura che tutti i pacchetti siano consegnati correttamente, verificando l'integrità e l'ordine dei pacchetti?

   TCP.

4. Quale tipo di indirizzo IPv6 viene usato per inviare un pacchetto a tutte le interfacce che appartengono a un gruppo di host?

   Multicast.

Risposte agli Esercizi Esplorativi

1. Cita 4 esempi di servizi che usano il protocollo TCP di default.

   FTP, SMTP, HTTP, POP3, IMAP, SSH.

2. Qual è il nome del campo sul pacchetto di intestazione IPv6 che ha lo stesso scopo del *TTL* su IPv4?

   Hop Limit.

3. Che tipo di informazioni è in grado di scoprire il *Neighbor Discovery Protocol* (NDP)?

   NDP è in grado di ottenere varie informazioni dalla rete, compresi altri nodi, indirizzi duplicati, percorsi, server DNS, gateway e così via..



109.2 Configurazione di rete persistente

Obiettivi LPI di riferimento

LPIC-1 version 5.0, Exam 102, Objective 109.2

Peso

4

Aree di Conoscenza Chiave

- Comprendere la configurazione TCP/IP di base su un host.
- Configurazione di una rete ethernet e di una wi-fi utilizzando NetworkManager.
- Conoscenza di systemd-networkd.

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

- `/etc/hostname`
- `/etc/hosts`
- `/etc/nsswitch.conf`
- `/etc/resolv.conf`
- `nmcli`
- `hostnamectl`
- `ifup`
- `ifdown`

109.2 Lezione 1

Certificazione:	LPIC-1
Versione:	5.0
Argomento:	109 Fondamenti di Networking
Obiettivo:	109.2 Configurazione di rete persistente
Lezione:	1 di 2

Introduzione

In qualsiasi rete TCP/IP ogni nodo deve aver la scheda di rete correttamente configurata per soddisfare i requisiti del *network* a cui accede, altrimenti non sarà in grado di effettuare nessuna comunicazione. Pertanto, l'amministratore di sistema deve fornire la configurazione di base in modo che il sistema operativo sia in grado di impostare l'interfaccia di rete appropriata, nonché di identificare a ogni avvio se stesso e le caratteristiche di base della rete.

Le impostazioni di rete sono *agnostiche* rispetto ai sistemi operativi, ma questi ultimi hanno i loro metodi per memorizzare e applicare queste impostazioni. I sistemi Linux si basano su configurazioni memorizzate in file di testo sotto la directory `/etc`. È importante conoscere come questi file vengono utilizzati per evitare la perdita di connettività a causa di un'errata configurazione locale.

L'Interfaccia di Rete

"*Interfaccia di rete*" è il termine con cui il sistema operativo si riferisce al canale di comunicazione configurato per lavorare con l'hardware di rete collegato al sistema, come un dispositivo *ethernet* o *Wi-Fi*. L'eccezione è rappresentata dall'interfaccia di *loopback*, che il sistema operativo utilizza

quando ha bisogno di stabilire una connessione con se stesso, ma lo scopo principale di un'interfaccia di rete è quello di fornire un percorso attraverso il quale i dati locali possono essere inviati e i dati remoti possono essere ricevuti. Se l'interfaccia di rete non è configurata correttamente, il sistema operativo non sarà in grado di comunicare con le altre macchine della rete.

Nella maggior parte dei casi le impostazioni corrette dell'interfaccia sono definite di default o personalizzate durante l'installazione del sistema operativo. Tuttavia, queste impostazioni spesso devono essere controllate o addirittura modificate quando la comunicazione non funziona correttamente o quando il comportamento dell'interfaccia richiede una personalizzazione.

Ci sono molti comandi Linux per elencare quali interfacce di rete sono presenti sul sistema, ma non tutti sono disponibili in tutte le distribuzioni. Il comando `ip`, tuttavia, fa parte dell'insieme di base degli strumenti di rete forniti nelle moderne distribuzioni Linux e può essere usato per elencare le interfacce di rete. Il comando completo per mostrare le interfacce è `ip link show`:

```
$ ip link show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN mode DEFAULT group
default qlen 1000
        link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: enp3s5: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP mode DEFAULT
group default qlen 1000
        link/ether 00:16:3e:8d:2b:5b brd ff:ff:ff:ff:ff:ff
```

Se disponibile, può essere usato anche il comando `nmcli device`:

```
$ nmcli device
DEVICE      TYPE      STATE      CONNECTION
enp3s5      ethernet  connected  Gigabit Powerline Adapter
lo          loopback  unmanaged  --
```

I comandi mostrati negli esempi *non* modificano alcuna impostazione nel sistema, quindi possono essere eseguiti da un utente senza privilegi. Entrambi i comandi elencano due interfacce di rete: `lo` (l'interfaccia di loopback) e `enp3s5` (un'interfaccia ethernet).

I desktop e i portatili con Linux hanno di solito due o tre interfacce di rete predefinite, una per l'interfaccia virtuale di loopback e le altre assegnate all'hardware di rete trovato dal sistema. I server e le apparecchiature di rete con Linux, d'altra parte, possono avere decine di interfacce di rete, ma gli stessi principi si applicano senza differenze sostanziali. L'astrazione fornita dal sistema operativo permette la configurazione delle interfacce di rete utilizzando gli stessi metodi,

indipendentemente dall'hardware sottostante.

Tuttavia, conoscere i dettagli sull'hardware sottostante di un'interfaccia può essere utile per capire meglio che cosa stia succedendo quando la comunicazione non funziona come previsto. In un sistema in cui sono disponibili molte interfacce di rete, potrebbe, per esempio, non essere ovvio quale corrisponde al wi-fi e quale all'ethernet. Per questo motivo, Linux utilizza una *convenzione di denominazione delle interfacce* che aiuta a identificare quale interfaccia di rete corrisponde a quale dispositivo e porta.

Nomi di Interfaccia

Le vecchie distribuzioni Linux chiamavano le interfacce di rete ethernet `eth0`, `eth1` e così via, numerate secondo l'ordine in cui il kernel le identificava. Le interfacce wireless erano chiamate `wlan0`, `wlan1` e così via. Questa convenzione di denominazione, tuttavia, non chiarisce quale specifica porta ethernet corrisponde all'interfaccia `eth0`. A seconda di come veniva rilevato l'hardware, era anche possibile che due interfacce di rete si scambiassero i nomi dopo un riavvio.

Per superare questa ambiguità, i sistemi Linux più recenti impiegano una specifica convenzione di denominazione per le interfacce di rete, creando una relazione più stretta tra il nome dell'interfaccia e la connessione hardware sottostante.

Nelle distribuzioni Linux che usano lo schema di denominazione *systemd,* tutti i nomi delle interfacce iniziano con un prefisso di due caratteri che indica il tipo di interfaccia:

`en`
: Ethernet

`ib`
: InfiniBand

`sl`
: Serial line IP (slip)

`wl`
: Wireless local area network (WLAN)

`ww`
: Wireless wide area network (WWAN)

Dalla più alta alla più bassa priorità, vengono utilizzate dal sistema operativo per nominare e numerare le interfacce di rete le seguenti regole:

1. Denominare l'interfaccia rispetto all'indice fornito dal BIOS/UEFI o dal firmware dei dispositivi *embedded*, per esempio `eno1`.
2. Denominare l'interfaccia rispetto all'indice dello *slot PCI express*, fornito dal BIOS/UEFI o dal firmware, per esempio `ens1`.
3. Denominare l'interfaccia rispetto al suo *indirizzo sul bus corrispondente*, per esempio `enp3s5`.
4. Denominare l'interfaccia rispetto al suo *MAC address* , per esempio `enx78e7d1ea46da`.
5. Denominare l'interfaccia usando la *convenzione legacy*, per esempio `eth0`.

È corretto supporre, per esempio, che l'interfaccia di rete `enp3s5` sia stata chiamata così perché non si adattava ai primi due metodi di denominazione, quindi è stato usato il suo indirizzo nel bus e nello slot corrispondente. L'indirizzo del dispositivo `03:05.0`, trovato nell'output del comando `lspci`, rivela il dispositivo associato:

```
$ lspci | fgrep Ethernet
03:05.0 Ethernet controller: Realtek Semiconductor Co., Ltd. RTL-8110SC/8169SC Gigabit
Ethernet (rev 10)
```

Le interfacce di rete sono create dal kernel Linux stesso, ma ci sono molti comandi che possono essere utilizzati per interagire con esse. Normalmente, la configurazione avviene automaticamente e non c'è bisogno di cambiare le impostazioni manualmente. Tuttavia, con il nome dell'interfaccia, è possibile dire al kernel, se necessario, come procedere nella configurazione.

Gestione dell'Interfaccia

Nel corso degli anni, sono stati sviluppati diversi programmi per interagire con le funzionalità di rete fornite dal kernel Linux. Anche se il vecchio comando `ifconfig` può ancora essere usato per fare semplici configurazioni e interrogazioni di interfacce, è ora deprecato a causa del suo limitato supporto di interfacce "non-ethernet". Il comando `ifconfig` è stato sostituito dal comando `ip`, che è in grado di gestire molti altri aspetti delle interfacce, come rotte e tunnel.

Le molte capacità del comando `ip` possono essere eccessive per la maggior parte dei compiti ordinari, quindi ci sono comandi ausiliari per facilitare l'attivazione e la configurazione delle interfacce di rete. I comandi `ifup` e `ifdown` possono essere usati per configurare le interfacce di rete in base alle definizioni di interfaccia che si trovano nel file `/etc/network/interfaces`. Sebbene possano essere invocati manualmente, questi comandi sono normalmente eseguiti automaticamente durante l'avvio del sistema.

Tutte le interfacce di rete gestite da `ifup` e `ifdown` dovrebbero essere elencate nel file

`/etc/network/interfaces`. Il formato usato nel file è semplice: le linee che iniziano con la parola `auto` sono usate per identificare le interfacce fisiche che devono essere attivate quando `ifup` viene eseguito con l'opzione `-a`. Il nome dell'interfaccia dovrebbe seguire la parola `auto` sulla stessa linea. Tutte le interfacce marcate `auto` sono attivate all'avvio, nell'ordine in cui sono elencate.

WARNING

I metodi di configurazione della rete usati da `ifup` e `ifdown` *non sono* standardizzati in tutte le distribuzioni Linux. CentOS, per esempio, mantiene le impostazioni di interfaccia in file individuali nella directory `/etc/sysconfig/network-scripts/` e il formato di configurazione usato in essi è leggermente diverso da quello usato in `/etc/network/interfaces`.

La configurazione vera e propria dell'interfaccia viene scritta in un'altra riga che inizia con la parola `iface`, seguita dal nome dell'interfaccia, dal nome della famiglia di indirizzi che l'interfaccia utilizza e dal nome del metodo utilizzato per configurare l'interfaccia. L'esempio seguente mostra un file di configurazione di base per le interfacce `lo` (loopback) e `enp3s5`:

```
auto lo
iface lo inet loopback

auto enp3s5
iface enp3s5 inet dhcp
```

La famiglia di indirizzi dovrebbe essere `inet` per la rete IPv4, ma c'è anche il supporto per la rete IPX (`ipx`), e la rete IPv6 (`inet6`). Le interfacce loopback usano il metodo di configurazione `loopback`. Con il metodo `dhcp`, l'interfaccia userà le impostazioni IP fornite dal server DHCP della rete. Le impostazioni della configurazione di esempio permettono l'esecuzione del comando `ifup` utilizzando come argomento il nome dell'interfaccia `enp3s5`:

```
# ifup enp3s5
Internet Systems Consortium DHCP Client 4.4.1
Copyright 2004-2018 Internet Systems Consortium.
All rights reserved.
For info, please visit https://www.isc.org/software/dhcp/

Listening on LPF/enp3s5/00:16:3e:8d:2b:5b
Sending on   LPF/enp3s5/00:16:3e:8d:2b:5b
Sending on   Socket/fallback
DHCPDISCOVER on enp3s5 to 255.255.255.255 port 67 interval 4
DHCPOFFER of 10.90.170.158 from 10.90.170.1
DHCPREQUEST for 10.90.170.158 on enp3s5 to 255.255.255.255 port 67
```

```
DHCPACK of 10.90.170.158 from 10.90.170.1
bound to 10.90.170.158 -- renewal in 1616 seconds.
```

In questo esempio il metodo scelto per l'interfaccia `enp3s5` era `dhcp`, quindi il comando `ifup` richiama un programma client DHCP per ottenere le impostazioni IP dal server DHCP. Allo stesso modo, il comando `ifdown enp3s5` può essere usato per disattivare l'interfaccia.

In reti senza un server DHCP il metodo `static` potrebbe essere usato inserendo manualmente le impostazioni IP nel file `/etc/network/interfaces`:

```
iface enp3s5 inet static
    address 192.168.1.2/24
    gateway 192.168.1.1
```

Le interfacce che usano il metodo `static` non hanno bisogno di una direttiva `auto` corrispondente, dato che sono avviate ogni volta che viene rilevato l'hardware di rete.

Se la stessa interfaccia ha più di una voce `iface`, allora tutti gli indirizzi e le opzioni configurati saranno applicati quando si avvia quell'interfaccia. Questo è utile per configurare sia indirizzi IPv4 che IPv6 sulla stessa interfaccia, così come la configurazione di più indirizzi dello stesso tipo su una singola interfaccia.

Nomi Locali e Remoti

Una configurazione TCP/IP funzionante è solo il primo passo verso la piena usabilità della rete. Oltre a essere in grado di identificare i nodi della rete tramite i loro indirizzi IP, il sistema deve essere in grado di identificarli con nomi più facilmente comprensibili per gli esseri umani.

Il nome con cui il sistema si identifica è personalizzabile ed è buona pratica definirlo, anche se la macchina non è destinata a entrare in una rete. Il nome locale spesso corrisponde al nome di rete della macchina, ma questo non è necessariamente sempre vero. Se il file `/etc/hostname` esiste, il sistema operativo userà il contenuto della prima linea come nome locale, in seguito chiamato semplicemente *hostname*. Le linee che iniziano con `#` all'interno di `/etc/hostname` vengono ignorate.

Il file `/etc/hostname` può essere modificato direttamente, ma l'hostname della macchina può anche essere definito attraverso il comando `hostnamectl`. Quando viene fornito con il sottocomando `set-hostname`, il comando `hostnamectl` prenderà il nome dato come argomento e lo scriverà in `/etc/hostname`:

```
# hostnamectl set-hostname storage
# cat /etc/hostname
storage
```

L'hostname definito in `/etc/hostname` è l'hostname *statico*, cioè il nome che viene usato per inizializzare l'hostname del sistema all'avvio. L'hostname statico può essere una stringa a forma libera lunga fino a 64 caratteri. Si raccomanda di utilizzare solo caratteri ASCII minuscoli senza spazi o punti. Ci si dovrebbe anche limitare nella scelta del nome al formato consentito per i nomi di dominio DNS, anche se questo non è un requisito necessario.

Il comando `hostnamectl` può impostare altri due tipi di hostname oltre all'hostname statico:

Pretty hostname

A differenza dell'hostname statico, il *pretty hostname* può includere tutti i tipi di caratteri speciali. Può essere usato per impostare un nome più descrittivo per la macchina, per esempio "LAN Shared Storage":

```
# hostnamectl --pretty set-hostname "LAN Shared Storage"
```

Transient hostname

Usato quando l'hostname statico non è impostato o quando è il nome di default `localhost`. Il *transient hostname* è di norma il nome impostato insieme ad altre configurazioni automatiche, ma può anche essere modificato dal comando `hostnamectl` come nel seguente esempio:

```
# hostnamectl --transient set-hostname generic-host
```

Se non vengono usate né l'opzione `--pretty` né l'opzione `--transient`, allora *tutti e tre i tipi* di hostname saranno impostati al nome dato. Per impostare l'hostname statico, *ma non* gli altri due, dovrebbe invece essere usata l'opzione `--static`. In tutti i casi, *solo* l'hostname statico è memorizzato nel file `/etc/hostname`. Il comando `hostnamectl` può anche essere usato per visualizzare varie informazioni descrittive e di identità sul sistema in esecuzione:

```
$ hostnamectl status
     Static hostname: storage
     Pretty hostname: LAN Shared Storage
  Transient hostname: generic-host
           Icon name: computer-server
             Chassis: server
          Machine ID: d91962a957f749bbaf16da3c9c86e093
```

```
             Boot ID: 8c11dcab9c3d4f5aa53f4f4e8fdc6318
    Operating System: Debian GNU/Linux 10 (buster)
              Kernel: Linux 4.19.0-8-amd64
        Architecture: x86-64
```

Questa è l'azione predefinita del comando `hostnamectl`, quindi il sottocomando `status` può essere omesso.

Per quanto riguarda il nome dei nodi di rete remoti, ci sono due modi fondamentali che il sistema operativo può implementare per abbinare nomi e indirizzi IP: utilizzare una fonte locale o utilizzare un server remoto per tradurre i nomi in indirizzi IP e viceversa. I metodi possono essere complementari tra loro e l'ordine di priorità è definito nel file di configurazione *Name Service Switch*: `/etc/nsswitch.conf`. Questo file è usato dal sistema e dalle applicazioni per determinare non solo le fonti per le corrispondenze nome-IP, ma anche le fonti da cui ottenere informazioni sul servizio dei nomi in una serie di categorie, chiamate *database*.

Il database *hosts* tiene traccia della mappatura tra nomi di host e indirizzi di host. La linea dentro `/etc/nsswitch.conf` che inizia con `hosts` definisce i servizi responsabili nel fornire le associazioni a tal scopo:

```
hosts: files dns
```

In questa voce di esempio `files` e `dns` sono i nomi dei servizi che specificano come funzionerà il processo di ricerca dei nomi host. Per prima cosa, il sistema cercherà le corrispondenze nei file locali, poi chiederà al servizio DNS le corrispondenze.

Il file locale per il database degli host è `/etc/hosts`, un semplice file di testo che associa indirizzi IP a nomi di host, una linea per indirizzo IP. Per esempio:

```
127.0.0.1 localhost
```

L'indirizzo IP *127.0.0.1* è l'indirizzo predefinito dell'interfaccia di loopback, da cui la sua associazione con il nome *localhost*.

È anche possibile legare alias opzionali allo stesso IP. Gli alias possono fornire denominazioni alternative, nomi di host più brevi e dovrebbero essere aggiunti alla fine della linea come di seguito:

```
192.168.1.10 foo.mydomain.org foo
```

Le regole di formattazione per il file `/etc/hosts` sono:

- I campi della singola voce sono separati da un numero qualsiasi di spazi vuoti e/o caratteri di tabulazione.
- Le linee che iniziano con `#` sono un commento e vengono ignorate.
- I nomi degli host possono contenere solo caratteri alfanumerici, segni di meno (`-`) e punti (`.`).
- I nomi degli host devono iniziare con un carattere alfabetico e finire con un carattere alfanumerico.

Gli indirizzi IPv6 possono anche essere aggiunti a `/etc/hosts`. La seguente voce si riferisce all'indirizzo di loopback IPv6:

```
::1 localhost ip6-localhost ip6-loopback
```

Dopo la specifica del servizio `files`, la specifica `dns` dice al sistema di chiedere a un servizio DNS l'associazione nome macchina/IP. L'insieme delle routine responsabili di questo metodo è chiamato *resolver* e il suo file standard di configurazione è `/etc/resolv.conf`. L'esempio seguente mostra un generico `/etc/resolv.conf` contenente voci per i server DNS pubblici di Google:

```
nameserver 8.8.4.4
nameserver 8.8.8.8
```

Come mostrato nell'esempio, la parola chiave `nameserver` indica l'indirizzo IP del server DNS. È richiesto *un solo* nameserver, ma possono esserne indicati fino a tre. Quelli supplementari saranno usati come *fallback*. Se non sono presenti voci di nameserver, il comportamento predefinito è quello di utilizzare il server dei nomi sulla macchina locale (se presente).

Il resolver può essere configurato anche per aggiungere automaticamente un nome di dominio ai nomi host brevi. Per esempio:

```
nameserver 8.8.4.4
nameserver 8.8.8.8
domain mydomain.org
search mydomain.net mydomain.com
```

La voce `domain` imposta `mydomain.org` come nome di dominio locale, così le query per i nomi all'interno di questo dominio saranno autorizzate ad usare nomi brevi relativi al dominio locale.

La voce `search` ha uno scopo simile, ma accetta una lista di domini da provare quando viene fornito un nome breve. Per impostazione predefinita, contiene solo il nome del dominio locale.

Esercizi Guidati

1. Quali comandi possono essere usati per elencare le schede di rete presenti nel sistema?

2. Qual è il tipo di adattatore di rete il cui nome di interfaccia è `wlo1`?

3. Che ruolo ha il file `/etc/network/interfaces` durante l'avvio?

4. Quale voce in `/etc/network/interfaces` configura l'interfaccia `eno1` per ottenere le sue impostazioni IP attraverso un DHCP server?

Esercizi Esplorativi

1. Come si potrebbe usare il comando `hostnamectl` per cambiare in `firewall` solo l'hostname *statico* della macchina locale?

2. Quali dettagli diversi dagli hostname possono essere modificati dal comando `hostnamectl`?

3. Quale voce in `/etc/hosts` associa entrambi i nomi `firewall` e `router` all'IP `10.8.0.1`?

4. Come si potrebbe modificare il file `/etc/resolv.conf` per inviare tutte le richieste DNS a `1.1.1.1`?

 Questa opera è concessa in licenza CC BY-NC-ND 4.0. |

Sommario

Questa lezione mostra i cambiamenti persistenti possibili alla configurazione della rete locale usando file e comandi standard di Linux. Linux si aspetta che le impostazioni TCP/IP siano in percorsi specifici e può essere necessario effettuare cambiamenti quando le impostazioni predefinite non sono appropriate. La lezione tratta i seguenti argomenti:

- Come Linux identifica le interfacce di rete.
- L'attivazione delle interfacce durante l'avvio e la configurazione IP di base.
- Come il sistema operativo associa i nomi agli host.

I concetti, i comandi e le procedure affrontati sono stati:

- Convenzioni relative alla denominazione delle interfacce.
- Elenco delle interfacce di rete con `ip` e `nmcli`.
- Attivazione delle interfacce con `ifup` e `ifdown`.
- Il comando `hostnamectl` e il file `/etc/hostname`.
- I file `/etc/nsswitch.conf`, `/etc/hosts` e `/etc/resolv.conf`.

Risposte agli Esercizi Guidati

1. Quali comandi possono essere usati per elencare le schede di rete presenti nel sistema?

   I comandi `ip link show`, `nmcli device` e il *legacy* `ifconfig`.

2. Qual è il tipo di adattatore di rete il cui nome di interfaccia è `wlo1`?

   Il nome inizia con `wl`, quindi è un adattatore LAN wireless.

3. Che ruolo ha il file `/etc/network/interfaces` durante l'avvio?

   Include le configurazioni utilizzate dal comando `ifup` per attivare le interfacce corrispondenti durante l'avvio.

4. Quale voce in `/etc/network/interfaces` configura l'interfaccia `eno1` per ottenere le sue impostazioni IP attraverso un DHCP server?

   La voce `iface eno1 inet dhcp`.

Risposte agli Esercizi Esplorativi

1. Come si potrebbe usare il comando `hostnamectl` per cambiare in `firewall` solo l'hostname *statico* della macchina locale?

   Con l'opzione `--static`: `hostnamectl --static set-hostname firewall`.

2. Quali dettagli diversi dagli hostname possono essere modificati dal comando `hostnamectl`?

   `hostnamectl` può anche impostare il nome dell'icona predefinita per la macchina locale, il tipo di sistema, la posizione e l'ambiente di distribuzione.

3. Quale voce in `/etc/hosts` associa entrambi i nomi `firewall` e `router` all'IP `10.8.0.1`?

   La linea `10.8.0.1 firewall router`.

4. Come si potrebbe modificare il file `/etc/resolv.conf` per inviare tutte le richieste DNS a `1.1.1.1`?

   Usando `nameserver 1.1.1.1` come sua unica voce *nameserver*.



109.2 Lezione 2

Certificazione:	LPIC-1
Versione:	5.0
Argomento:	109 Fondamenti di Networking
Obiettivo:	109.2 Configurazione di rete persistente
Lezione:	2 di 2

Introduzione

Linux supporta virtualmente ogni tecnologia di rete utilizzata per collegare server, container, macchine virtuali, desktop e dispositivi *mobile*. Le connessioni tra tutti questi nodi di rete possono essere dinamiche ed eterogenee, richiedendo così una gestione appropriata da parte del sistema operativo che vi gira.

In passato le distribuzioni sviluppavano le proprie soluzioni personalizzate per gestire l'infrastruttura di rete dinamica. Oggi, strumenti come *NetworkManager* e *systemd* forniscono caratteristiche più complete e integrate per soddisfare tutte le richieste specifiche.

NetworkManager

La maggior parte delle distribuzioni Linux adotta il demone di servizio *NetworkManager* per configurare e controllare le connessioni di rete del sistema. Lo scopo di NetworkManager è di rendere la configurazione della rete il più semplice e automatica possibile. Quando si usa il DHCP, per esempio, NetworkManager organizza i cambiamenti della *default route*, il recupero degli indirizzi IP e gli aggiornamenti alla lista locale dei server DNS, se necessario. Quando sono disponibili sia connessioni cablate sia wireless, NetworkManager dà per impostazione predefinita

la priorità alla connessione cablata. NetworkManager cercherà di mantenere almeno una connessione attiva per tutto il tempo, quando è possibile.

NOTE

Una richiesta tramite DHCP (*Dynamic Host Configuration Protocol*) viene solitamente inviata attraverso la scheda di rete non appena viene stabilito il collegamento. Il server DHCP attivo sulla rete risponde quindi con le impostazioni (indirizzo IP, maschera di rete, rotta predefinita, ecc.) che il richiedente deve utilizzare per comunicare tramite il protocollo IP.

Per impostazione predefinita, il demone NetworkManager controlla le interfacce di rete non menzionate nel file `/etc/network/interfaces`. Lo fa per non interferire con altri metodi di configurazione che possono essere presenti, modificando così *solo* le interfacce *non* controllate.

Il demone NetworkManager viene eseguito in background con privilegi di root e avvia le azioni necessarie per mantenere il sistema online. Gli utenti ordinari possono creare e modificare connessioni di rete con applicazioni client che, pur non avendo privilegi di root, sono in grado di comunicare con il servizio sottostante per eseguire le azioni richieste.

Le applicazioni client per NetworkManager sono disponibili sia per la linea di comando sia per l'ambiente grafico. Per quest'ultimo, l'applicazione client viene fornita come accessorio dell'ambiente desktop (sotto nomi come *nm-tray*, *network-manager-gnome*, *nm-applet* o *plasma-nm*) ed è solitamente accessibile attraverso un'icona indicatore nell'angolo della barra del desktop o dall'utilità di configurazione del sistema.

Nella linea di comando NetworkManager stesso fornisce due programmi client: `nmcli` e `nmtui`. Entrambi i programmi hanno le stesse caratteristiche di base, ma `nmtui` ha un'interfaccia basata su *curses* mentre `nmcli` è un comando più completo che può essere usato anche negli script. Il comando `nmcli` separa tutte le proprietà relative alla rete controllate da NetworkManager in categorie chiamate *objects*:

`general`

Stato e operazioni generali di NetworkManager.

`networking`

Controllo generale della rete.

`radio`

Interruttori per wireless di NetworkManager.

`connection`

Connessioni di NetworkManager.

`device`

Dispositivi gestiti da NetworkManager.

`agente`

Agente password o `polkit` di NetworkManager.

`monitor`

Monitor di cambiamenti di NetworkManager.

Il nome dell'oggetto è l'argomento principale del comando `nmcli`. Per mostrare lo stato generale di connettività del sistema, per esempio, l'oggetto `general` dovrebbe essere dato come argomento:

```
$ nmcli general
STATE      CONNECTIVITY  WIFI-HW  WIFI     WWAN-HW  WWAN
connected  full          enabled  enabled  enabled  enabled
```

La colonna `STATE` dice se il sistema è connesso a una rete oppure no. Se la connessione è limitata a causa di una cattiva configurazione esterna o di restrizioni di accesso, allora la colonna `CONNECTIVITY` non riporterà lo stato di connettività `full`. Se nella colonna `CONNECTIVITY` appare `Portal`, significa che sono richiesti passaggi di autenticazione extra (di solito attraverso il browser web) per completare il processo di connessione. Le colonne rimanenti riportano lo stato delle connessioni wireless (se presenti), sia `WIFI` che `WWAN` (Wide Wireless Area Network, cioè reti cellulari). Il suffisso `HW` indica lo stato corrispondente al dispositivo di rete piuttosto che alla connessione di rete di sistema, quindi dice anche se l'hardware è abilitato o disabilitato per risparmiare energia.

Oltre all'argomento *oggetto*, `nmcli` ha bisogno per essere eseguito anche di un argomento. Il comando `status` è usato per default se non è presente alcun argomento di comando, quindi il comando `nmcli general` è effettivamente interpretato come `nmcli general status`.

Di rado è necessario intraprendere azioni specifiche quando l'adattatore di rete è collegato direttamente al punto di accesso tramite cavi, le reti wireless al contrario richiedono specifiche configurazioni per poter funzionare. `nmcli` facilita il processo di connessione e salva le impostazioni per connettersi automaticamente in futuro, quindi è molto utile per i computer portatili o qualsiasi altro *device* mobile.

Prima di connettersi al wi-fi, è utile prima elencare le reti disponibili. Se il sistema ha un adattatore wi-fi funzionante, allora l'oggetto `device` potrà essere usato per scansionare le reti disponibili con il comando `nmcli device wifi list`:

```
$ nmcli device wifi list
IN-USE  BSSID              SSID        MODE   CHAN  RATE        SIGNAL  BARS  SECURITY
        90:F6:52:C5:FA:12  Hypnotoad   Infra  11    130 Mbit/s  67      ▂▄▆_  WPA2
        10:72:23:C7:27:AC  Jumbao      Infra  1     130 Mbit/s  55      ▂▄__  WPA2
        00:1F:33:33:E9:BE  NETGEAR     Infra  1     54 Mbit/s   35      ▂▄__  WPA1 WPA2
        A4:33:D7:85:6D:B0  AP53        Infra  11    130 Mbit/s  32      ▂▄__  WPA1 WPA2
        98:1E:19:1D:CC:3A  Bruma       Infra  1     195 Mbit/s  22      ▂___  WPA1 WPA2
```

La maggior parte degli utenti probabilmente userà il nome nella colonna `SSID` per identificare la rete di interesse. Per esempio, il comando `nmcli` può connettersi alla rete chiamata `Hypnotoad` usando ancora l'oggetto `device`:

```
$ nmcli device wifi connect Hypnotoad
```

Se il comando viene eseguito all'interno di un emulatore di terminale in ambiente grafico, allora apparirà una finestra di dialogo che chiede la *passphrase* della rete. Quando viene eseguito in una console di solo testo, la password può essere fornita insieme agli altri argomenti:

```
$ nmcli device wifi connect Hypnotoad password MyPassword
```

Se la rete wi-fi nasconde il suo nome SSID, `nmcli` può ancora connettersi ad essa con gli argomenti extra `hidden yes`:

```
$ nmcli device wifi connect Hypnotoad password MyPassword hidden yes
```

Se il sistema ha *più di un* adattatore wi-fi, quello da usare può essere indicato con `ifname`. Per esempio, per connettersi usando l'adattatore chiamato `wlo1`:

```
$ nmcli device wifi connect Hypnotoad password MyPassword ifname wlo1
```

Dopo che la connessione ha successo, NetworkManager assegnerà alla connessione il nome dell'SSID corrispondente (se è una connessione wi-fi) e lo conserverà per le connessioni future. I nomi delle connessioni e i loro UUID sono elencati dal comando `nmcli connection show`:

```
$ nmcli connection show
NAME                UUID                                  TYPE      DEVICE
Ethernet            53440255-567e-300d-9922-b28f0786f56e  ethernet  enp3s5
tun0                cae685e1-b0c4-405a-8ece-6d424e1fb5f8  tun       tun0
```

```
Hypnotoad          6fdec048-bcc5-490a-832b-da83d8cb7915  wifi      wlo1
4G                 a2cf4460-0cb7-42e3-8df3-ccb927f2fd88  gsm       --
```

Viene mostrato il tipo di ogni connessione — che può essere `ethernet`, `wifi`, `tun`, `gsm`, `bridge`, ecc. — così come il dispositivo a cui sono associate. Per eseguire azioni su una specifica connessione è necessario fornire il nome o UUID. Per disattivare la connessione `Hypnotoad`, per esempio:

```
$ nmcli connection down Hypnotoad
Connection 'Hypnotoad' successfully deactivated
```

Allo stesso modo, il comando `nmcli connection up Hypnotoad` può essere usato per attivare la connessione, dato che ora è salvata da NetworkManager. Il nome dell'interfaccia può anche essere usato per riconnettersi, ma in questo caso l'oggetto `device` dovrebbe invece essere usato nel modo seguente:

```
$ nmcli device disconnect wlo2
Device 'wlo1' successfully disconnected.
```

Il nome dell'interfaccia può anche essere usato per ristabilire la connessione:

```
$ nmcli device connect wlo2
Device 'wlo1' successfully activated with '833692de-377e-4f91-a3dc-d9a2b1fcf6cb'.
```

Si noti che l'UUID della connessione cambia *ogni volta* che la connessione viene richiamata, quindi è preferibile usare il suo nome per coerenza.

Se l'adattatore wireless è disponibile ma non viene utilizzato, allora può essere spento per risparmiare energia. Questa volta, l'oggetto *radio* dovrebbe essere passato a `nmcli`:

```
$ nmcli radio wifi off
```

Naturalmente il dispositivo wireless può essere riacceso con il comando `nmcli radio wifi on`.

Una volta che le connessioni sono stabilite non sarà necessaria alcuna interazione manuale in futuro, poiché NetworkManager identifica le reti conosciute disponibili e si connette automaticamente ad esse. Se necessario, NetworkManager ha dei plugin che possono estendere le sue funzionalità, come il plugin per supportare le connessioni VPN.

systemd-networkd

I sistemi che eseguono systemd possono opzionalmente usare i suoi demoni integrati per gestire la connettività di rete: `systemd-networkd` per controllare le interfacce di rete e `systemd-resolved` per gestire la risoluzione dei nomi locali. Questi servizi sono retro compatibili con i precedenti metodi di configurazione di Linux, ma la configurazione delle interfacce di rete in particolare ha caratteristiche che vale la pena conoscere.

I file di configurazione usati da systemd-networkd per impostare le interfacce di rete possono essere trovati in una delle seguenti tre directory:

`/lib/systemd/network`
: La directory della rete di sistema.

`/run/systemd/network`
: La directory non persistente della rete di runtime.

`/etc/systemd/network`
: La directory di amministrazione locale della rete.

I file sono processati in ordine *lessicografico*, quindi si raccomanda di iniziare i loro nomi con dei numeri per rendere l'ordine più facile da leggere e da impostare.

I file in `/etc` hanno la massima priorità, mentre i file in `/run` hanno la precedenza sui file con lo stesso nome in `/lib`. Questo significa che se i file di configurazione in diverse directory hanno lo stesso nome, allora systemd-networkd ignorerà i file con priorità minore. Separare i file in questo modo è un metodo per cambiare le impostazioni dell'interfaccia senza dover modificare i file originali: le modifiche possono essere apportate in `/etc/systemd/network` per sovrascrivere quelle in `/lib/systemd/network`.

Lo scopo di ogni file di configurazione dipende dal suo suffisso. I nomi dei file che terminano in `.netdev` sono usati da systemd-networkd per creare dispositivi di rete virtuali, come i dispositivi *bridge* o *tun*. I file che terminano in `.link` impostano configurazioni di basso livello per l'interfaccia di rete corrispondente. systemd-networkd rileva e configura automaticamente i dispositivi di rete non appena appaiono e ignora i dispositivi già configurati con altri mezzi, quindi non c'è bisogno di aggiungere questi file.

Il suffisso più importante è `.network`. I file che usano questo suffisso possono essere usati per impostare indirizzi e percorsi di rete. Come per gli altri tipi di file di configurazione, il nome del file definisce l'ordine in cui il file sarà processato. L'interfaccia di rete a cui il file di configurazione si riferisce è definita nella sezione `[Match]` all'interno del file.

Per esempio, l'interfaccia di rete ethernet `enp3s5` può essere selezionata nel file `/etc/systemd/network/30-lan.network` usando la voce `Name=enp3s5` nella sezione `[Match]`:

```
[Match]
Name=enp3s5
```

Viene accettata anche una lista di nomi separati da spazi vuoti per abbinare in una sola volta molte interfacce di rete con questo stesso file. I nomi possono contenere raggruppamenti (*globs*) in stile shell, come `en*`. Altre voci forniscono varie regole di corrispondenza, come il selezionare un dispositivo di rete in base al suo indirizzo MAC:

```
[Match]
MACAddress=00:16:3e:8d:2b:5b
```

Le impostazioni per il dispositivo sono nella sezione `[Network]` del file. Una semplice configurazione di rete statica richiede solo le voci `Address` e `Gateway`:

```
[Match]
MACAddress=00:16:3e:8d:2b:5b

[Network]
Address=192.168.0.100/24
Gateway=192.168.0.1
```

Per usare il protocollo DHCP invece di indirizzi IP statici, si dovrebbe usare la voce `DHCP`:

```
[Match]
MACAddress=00:16:3e:8d:2b:5b

[Network]
DHCP=yes
```

Il servizio systemd-networkd cercherà di recuperare entrambi gli indirizzi IPv4 e IPv6 per l'interfaccia di rete. Per usare solo IPv4 dovrebbe essere usato `DHCP=ipv4`. Allo stesso modo, `DHCP=ipv6` ignorerà le impostazioni IPv4 e userà solo l'indirizzo IPv6 fornito.

Le reti wireless protette da password possono anche essere configurate da systemd-networkd, ma l'adattatore di rete deve essere già autenticato nella rete *prima* che systemd-networkd possa configurarlo. L'autenticazione è eseguita da *WPA supplicant*, un programma dedicato alla

configurazione di adattatori di rete per reti protette da password.

Il primo passo è creare il file delle credenziali con il comando `wpa_passphrase`:

```
# wpa_passphrase MyWifi > /etc/wpa_supplicant/wpa_supplicant-wlo1.conf
```

Questo comando prenderà la *passphrase* per la rete wireless `MyWifi` dallo standard input e memorizzerà il suo *hash* nel file `/etc/wpa_supplicant/wpa_supplicant-wlo1.conf`. Si noti che il nome del file dovrebbe contenere il nome appropriato dell'interfaccia wireless, da cui il `wlo1` nel nome del file.

Il gestore systemd legge i file di passphrase WPA in `/etc/wpa_supplicant/` e crea il servizio corrispondente per eseguire *WPA supplicant* e attivare l'interfaccia. Il file passphrase creato nell'esempio avrà quindi un servizio corrispondente chiamato `wpa_supplicant@wlo1.service`. Il comando `systemctl start wpa_supplicant@wlo1.service` assocerà l'adattatore wireless con il punto di accesso remoto. Il comando `systemctl enable wpa_supplicant@wlo1.service` rende l'associazione automatica durante l'avvio.

Infine, un file `.network` corrispondente all'interfaccia `wlo1` deve essere presente in `/etc/systemd/network/`, poiché systemd-networkd lo userà per configurare l'interfaccia non appena il WPA supplicant termina l'associazione con l'access point.

Esercizi Guidati

1. Qual è il significato della parola `Portal` nella colonna `CONNECTIVITY` nell'output del comando `nmcli general status`?

2. In un terminale di console, come può un utente normale usare il comando `nmcli` per connettersi alla rete wireless `MyWifi` protetta dalla password `MyPassword`?

3. Quale comando può accendere l'adattatore wireless se è stato precedentemente disabilitato dal sistema operativo?

4. In quale directory dovrebbero essere messi i file di configurazione personalizzati quando systemd-networkd sta gestendo le interfacce di rete?

 Questa opera è concessa in licenza CC BY-NC-ND 4.0. |

Esercizi Esplorativi

1. Come può un utente eseguire il comando `nmcli` per cancellare una connessione inutilizzata chiamata `Hotel Internet`?

2. NetworkManager scansiona le reti wi-fi periodicamente e il comando `nmcli device wifi list` elenca solo gli access point trovati nell'ultima scansione. Come dovrebbe essere usato il comando `nmcli` per chiedere a NetworkManager di eseguire immediatamente una nuova scansione di tutti gli access point disponibili?

3. Quale voce `name` dovrebbe essere usata nella sezione `[Match]` di un file di configurazione systemd-networkd per far corrispondere tutte le interfacce ethernet?

4. Come dovrebbe essere eseguito il comando `wpa_passphrase` per usare la passphrase data come argomento e non dallo standard input?

Sommario

Questa lezione mostra gli strumenti comuni usati in Linux per gestire connessioni di rete eterogenee e dinamiche. Anche se la maggior parte dei metodi di configurazione non richiede l'intervento dell'utente, a volte può essere necessario farlo e strumenti come *NetworkManager* e *systemd-networkd* possono ridurre il fastidio al minimo. La lezione tratta i seguenti argomenti:

- Come NetworkManager e systemd-networkd si integrano con il sistema.
- Come l'utente può interagire con NetworkManager e systemd-networkd.
- Configurazione di base dell'interfaccia sia con NetworkManager che con systemd-networkd.

I concetti, i comandi e le procedure affrontati sono stati:

- I comandi client di NetworkManager: `nmtui` e `nmcli`.
- Scansione e connessione alle reti wireless usando i comandi appropriati di `nmcli`.
- Connessioni di rete wi-fi persistenti usando systemd-networkd.

Risposte agli Esercizi Guidati

1. Qual è il significato della parola `Portal` nella colonna `CONNECTIVITY` nell'output del comando `nmcli general status`?

   Significa che sono necessari dei passi di autenticazione extra (di solito attraverso il browser web) per completare il processo di connessione.

2. In un terminale di console, come può un utente normale usare il comando `nmcli` per connettersi alla rete wireless `MyWifi` protetta dalla password `MyPassword`?

   In un terminale di solo testo, il comando sarebbe:

   ```
   $ nmcli device wifi connect MyWifi password MyPassword
   ```

3. Quale comando può accendere l'adattatore wireless se è stato precedentemente disabilitato dal sistema operativo?

   ```
   $ nmcli radio wifi on
   ```

4. In quale directory dovrebbero essere messi i file di configurazione personalizzati quando systemd-networkd sta gestendo le interfacce di rete?

   Nella directory dell'amministrazione locale di rete: `/etc/systemd/network`.

Risposte agli Esercizi Esplorativi

1. Come può un utente eseguire il comando `nmcli` per cancellare una connessione inutilizzata chiamata `Hotel Internet`?

   ```
   $ nmcli connection delete "Hotel Internet"
   ```

2. NetworkManager scansiona le reti wi-fi periodicamente e il comando `nmcli device wifi list` elenca solo gli access point trovati nell'ultima scansione. Come dovrebbe essere usato il comando `nmcli` per chiedere a NetworkManager di eseguire immediatamente una nuova scansione di tutti gli access point disponibili?

   L'utente root può eseguire `nmcli device wifi rescan` per far sì che NetworkManager riesamini i punti di accesso disponibili.

3. Quale voce `name` dovrebbe essere usata nella sezione `[Match]` di un file di configurazione systemd-networkd per far corrispondere tutte le interfacce ethernet?

   La voce `name=en*`, poiché `en` è il prefisso per le interfacce ethernet in Linux e systemd-networkd accetta i raggruppamenti di tipo shell.

4. Come dovrebbe essere eseguito il comando `wpa_passphrase` per usare la passphrase data come argomento e non dallo standard input?

   La password dovrebbe essere data subito dopo l'SSID, come in `wpa_passphrase MyWifi MyPassword`.



109.3 Risoluzione dei problemi di base di una rete

Obiettivi LPI di riferimento

LPIC-1 version 5.0, Exam 102, Objective 109.3

Peso

4

Aree di Conoscenza Chiave

- Configurare manualmente le interfacce di rete, inclusa la visualizzazione e la modifica della configurazione delle interfacce di rete utilizzando iproute2.
- Configurare manualmente il routing, inclusa la visualizzazione e la modifica delle tabelle di routing e l'impostazione della rotta predefinita utilizzando iproute2.
- Debug dei problemi associati alla configurazione di rete.
- Conoscenza dei comandi legacy net-tools.

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

- `ip`
- `hostname`
- `ss`
- `ping`
- `ping6`
- `traceroute`
- `traceroute6`
- `tracepath`
- `tracepath6`

- `netcat`
- `ifconfig`
- `netstat`
- `route`



109.3 Lezione 1

Certificazione:	LPIC-1
Versione:	5.0
Argomento:	109 Fondamenti di Networking
Obiettivo:	109.3 Risoluzione dei problemi di base di una rete
Lezione:	1 di 2

Introduzione

Linux ha capacità di rete molto flessibili e potenti. Infatti i sistemi operativi basati su Linux sono spesso usati sui comuni dispositivi di rete, comprese costose apparecchiature commerciali. Questa lezione tratterà solo alcuni strumenti di base per la configurazione e la risoluzione delle problematiche di rete.

Assicurati di aver letto le lezioni sui protocolli Internet e sulla configurazione persistente della rete *prima* di affrontare questa lezione. In questa lezione scopriremo gli strumenti per risolvere i problemi delle reti IPv4 e IPv6.

Anche se non è un Obiettivo d'Esame ufficiale, i *packet sniffers* come `tcpdump` sono strumenti utili per la risoluzione dei problemi. I packet sniffers permettono di visualizzare e registrare i pacchetti che entrano ed escono da un'interfaccia di rete. Strumenti come *hex viewer* e *protocol analyzer* possono essere usati per vedere questi pacchetti in modo più dettagliato di quanto uno sniffer di pacchetti permetta. Non sarebbe male avere una conoscenza almeno di base di tali programmi.

Il Comando `ip`

Il comando `ip` è un'utilità abbastanza recente usata per visualizzare e lavorare su praticamente tutto ciò che riguarda le configurazioni di rete. Questa lezione copre alcuni dei sottocomandi più usati di `ip`, ma senza scendere troppo nel dettaglio. Imparare a leggere la documentazione ti aiuterà ad essere molto più efficiente.

Ogni sottocomando di `ip` ha la propria pagina man. La sezione `SEE ALSO` della pagina `man` di `ip` ne ha una lista:

```
$ man ip
...
SEE ALSO
       ip-address(8), ip-addrlabel(8), ip-l2tp(8), ip-link(8), ip-maddress(8),
       ip-monitor(8), ip-mroute(8), ip-neighbour(8), ip-netns(8), ip-
       ntable(8), ip-route(8), ip-rule(8), ip-tcp_metrics(8), ip-token(8), ip-
       tunnel(8), ip-xfrm(8)
       IP Command reference ip-cref.ps
...
```

Invece di visualizzarlo ogni volta che hai bisogno della pagina man, aggiungi semplicemente `-` e il nome del sottocomando a `ip`, per esempio `man ip-route`.

Un'altra fonte di informazioni è la funzione di *aiuto*. Per visualizzare l'aiuto integrato, aggiungi `help` dopo il sottocomando:

```
$ ip address help
Usage: ip address {add|change|replace} IFADDR dev IFNAME [ LIFETIME ]
                                                      [ CONFFLAG-LIST ]
       ip address del IFADDR dev IFNAME [mngtmpaddr]
       ip address {save|flush} [ dev IFNAME ] [ scope SCOPE-ID ]
                            [ to PREFIX ] [ FLAG-LIST ] [ label LABEL ] [up]
       ip address [ show [ dev IFNAME ] [ scope SCOPE-ID ] [ master DEVICE ]
                         [ type TYPE ] [ to PREFIX ] [ FLAG-LIST ]
                         [ label LABEL ] [up] [ vrf NAME ] ]
       ip address {showdump|restore}
IFADDR := PREFIX | ADDR peer PREFIX
...
```

Controllo della Maschera di Rete e dell'Instradamento

IPv4 e IPv6 sono noti come *protocolli instradati o instradabili*. Questo significa che sono progettati in modo da rendere possibile ai progettisti di rete di controllare il flusso del traffico. Ethernet non è un protocollo instradabile. Questo significa che se si dovesse collegare un gruppo di dispositivi insieme usando nient'altro che Ethernet, c'è molto poco che si possa fare per controllare il flusso del traffico di rete. Qualsiasi misura di controllo finirebbe per essere simile agli attuali protocolli di routing e di instradamento.

I protocolli instradabili permettono ai progettisti di rete di segmentare le reti per ridurre i requisiti di elaborazione dei dispositivi di connettività, fornire ridondanza e gestire il traffico.

Gli indirizzi IPv4 e IPv6 hanno due sezioni. La prima serie di bit costituisce la sezione di rete, mentre la seconda serie costituisce la parte host. Il numero di bit che compongono la parte di rete sono determinati dalla *netmask* (chiamata anche *subnet mask*). A volte ci si riferisce a essa anche come alla *lunghezza del prefisso*. Indipendentemente da come viene chiamata, è il numero di bit che la macchina tratta come porzione di rete dell'indirizzo. Con IPv4, a volte è specificato in notazione decimale punteggiata.

Qui si seguito c'è un esempio che utilizza IPv4. Nota come le cifre binarie mantengono il loro valore di posto negli ottetti anche quando viene diviso per la netmask.

```
192.168.130.5/20

          192      168      130      5
          11000000 10101000 10000010 00000101

20 bits = 11111111 11111111 11110000 00000000

Network = 192.168.128.0
Host    = 2.5
```

La porzione di rete di un indirizzo è usata da una macchina IPv4 o IPv6 per cercare su quale interfaccia un pacchetto debba essere spedito rispetto alla sua tabella di routing. Quando un host IPv4 o IPv6 con routing abilitato riceve un pacchetto che non è per l'host stesso, cerca di far corrispondere la porzione di rete della destinazione ad una rete nella tabella di routing. Se viene trovata una voce corrispondente, invia il pacchetto alla destinazione specificata nella tabella di routing. Se non vengono trovate voci e viene configurata una rotta predefinita, il pacchetto viene inviato alla rotta predefinita. Se non viene trovata alcuna voce e non è configurata alcuna rotta predefinita, il pacchetto viene scartato.

Configurare un'Interfaccia

Tratteremo due strumenti che puoi usare per configurare un'interfaccia di rete: `ifconfig` e `ip`. Il programma `ifconfig`, sebbene ancora ampiamente utilizzato, è considerato uno strumento *legacy* e potrebbe non essere disponibile sui sistemi più recenti.

> Sulle distribuzioni Linux più recenti, l'installazione del pacchetto `net-tools` ti fornirà i comandi di rete legacy.

Prima di configurare un'interfaccia devi sapere quali siano quelle disponibili. Ci sono alcuni modi per farlo. Uno di questi è usare l'opzione `-a` di `ifconfig`:

```
$ ifconfig -a
```

Un altro modo è con `ip`. A volte vedrai esempi con `ip addr`, `ip a`, e alcuni con `ip address`. Sono sinonimi. Ufficialmente, il sottocomando è `ip address`. Questo significa che se vuoi vedere la pagina man, devi usare `man ip-address` e non `man ip-addr`.

Il sottocomando `link` per `ip` elencherà i collegamenti di interfaccia disponibili per la configurazione:

```
$ ip link
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN mode DEFAULT group
default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP mode DEFAULT
group default qlen 1000
    link/ether 08:00:27:54:18:57 brd ff:ff:ff:ff:ff:ff
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP mode DEFAULT
group default qlen 1000
    link/ether 08:00:27:ab:11:3e brd ff:ff:ff:ff:ff:ff
```

Assumendo che il filesystem `sys` sia montato, puoi anche elencare il contenuto di `/sys/class/net`:

```
$ ls /sys/class/net
enp0s3  enp0s8  lo
```

Per configurare un'interfaccia con `ifconfig`, devi essere loggato come root o usare un'utilità

come `sudo` per eseguire il comando con i privilegi di root. Segui l'esempio qui sotto:

```
# ifconfig enp1s0 192.168.50.50/24
```

La versione Linux di `ifconfig` è abbastanza flessibile sul come indicare la subnet mask:

```
# ifconfig eth2 192.168.50.50 netmask 255.255.255.0
# ifconfig eth2 192.168.50.50 netmask 0xffffff00
# ifconfig enp0s8 add 2001:db8::10/64
```

Nota come con IPv6 è stata usata la parola chiave `add`. Se non fai precedere un indirizzo IPv6 da `add`, otterrai un messaggio di errore.

Il seguente comando configura un'interfaccia con `ip`:

```
# ip addr add 192.168.5.5/24 dev enp0s8
# ip addr add 2001:db8::10/64 dev enp0s8
```

Con `ip`, lo stesso comando è usato sia per IPv4 sia per IPv6.

Configurare Opzioni di Basso Livello

Il comando `ip link` è usato per configurare l'interfaccia o le impostazioni del protocollo come VLAN, ARP o MTU a basso livello o per disabilitare un'interfaccia.

Un compito frequente per `ip link` è disabilitare o abilitare un'interfaccia. Questo può essere fatto anche con `ifconfig`:

```
# ip link set dev enp0s8 down
# ip link show dev enp0s8
3: enp0s8: <BROADCAST,MULTICAST> mtu 1500 qdisc pfifo_fast state DOWN mode DEFAULT group
default qlen 1000
    link/ether 08:00:27:ab:11:3e brd ff:ff:ff:ff:ff:ff
# ifconfig enp0s8 up
# ip link show dev enp0s8
3: enp0s8: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP mode DEFAULT
group default qlen 1000
    link/ether 08:00:27:ab:11:3e brd ff:ff:ff:ff:ff:ff
```

A volte potrebbe essere necessario regolare l'MTU di un'interfaccia. Come per

abilitare/disabilitare le interfacce, questo può essere fatto sia con `ifconfig` che con `ip link`:

```
# ip link set enp0s8 mtu 2000
# ip link show dev enp0s3
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 2000 qdisc pfifo_fast state UP mode DEFAULT
group default qlen 1000
    link/ether 08:00:27:54:53:59 brd ff:ff:ff:ff:ff:ff
# ifconfig enp0s3 mtu 1500
# ip link show dev enp0s3
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP mode DEFAULT
group default qlen 1000
    link/ether 08:00:27:54:53:59 brd ff:ff:ff:ff:ff:ff
```

La Tabella di Routing

I comandi `route`, `netstat -r`, e `ip route` possono tutti essere usati per visualizzare la tabella di routing. Se vuoi modificare le tue rotte, devi usare `route` o `ip route`. Di seguito sono riportati esempi di visualizzazione di una tabella di routing:

```
$ netstat -r
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
default         10.0.2.2        0.0.0.0         UG        0 0          0 enp0s3
10.0.2.0        0.0.0.0         255.255.255.0   U         0 0          0 enp0s3
192.168.150.0   0.0.0.0         255.255.255.0   U         0 0          0 enp0s8
$ ip route
default via 10.0.2.2 dev enp0s3 proto dhcp metric 100
10.0.2.0/24 dev enp0s3 proto kernel scope link src 10.0.2.15 metric 100
192.168.150.0/24 dev enp0s8 proto kernel scope link src 192.168.150.200
$ route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
default         10.0.2.2        0.0.0.0         UG    100    0        0 enp0s3
10.0.2.0        0.0.0.0         255.255.255.0   U     100    0        0 enp0s3
192.168.150.0   0.0.0.0         255.255.255.0   U     0      0        0 enp0s8
```

Nota che non c'è alcun output riguardante IPv6. Se vuoi vedere la tua tabella di routing per IPv6, devi usare `route -6`, `netstat -6r` o `ip -6 route`.

```
$ route -6
Kernel IPv6 routing table
```

```
Destination                    Next Hop                   Flag Met Ref Use If
2001:db8::/64                  [::]                       U    256 0      0 enp0s8
fe80::/64                      [::]                       U    100 0      0 enp0s3
2002:a00::/24                  [::]                       !n   1024 0      0 lo
[::]/0                         2001:db8::1                UG   1   0      0 enp0s8
localhost/128                  [::]                       Un   0   2     84 lo
2001:db8::10/128               [::]                       Un   0   1      0 lo
fe80::a00:27ff:fe54:5359/128   [::]                       Un   0   1      0 lo
ff00::/8                       [::]                       U    256 1      3 enp0s3
ff00::/8                       [::]                       U    256 1      6 enp0s8
```

Un esempio di `netstat -r6` è stato omesso perché il suo output è identico a quello di `route -6`. Alcuni degli output del precedente comando `route` sono autoesplicativi. La colonna `Flag` fornisce alcune informazioni sulla rotta. La flag `U` indica che una rotta è attiva. Un `!` significa rifiutare la rotta, cioè una rotta con un `!` non sarà usata. La flag `n` significa che la rotta non è stata messa in cache (il kernel mantiene una cache delle rotte per una ricerca più veloce separatamente da tutte le rotte conosciute). La flag `G` indica un gateway. La colonna `Metric` o `Met` non è usata dal kernel: si riferisce alla distanza amministrativa dalla destinazione. Questa distanza amministrativa è usata dai protocolli di routing per determinare le rotte dinamiche. La colonna `Ref` è il conteggio dei riferimenti, o il numero di usi di una rotta. Come la `Metric`, non è usata dal kernel Linux. La colonna `Use` mostra il numero di ricerche per una rotta.

Nell'output di `netstat -r`, `MSS` indica la dimensione massima del segmento per le connessioni TCP su quel percorso. La colonna `Window` mostra la dimensione della finestra TCP default. `irtt` mostra il tempo di andata e ritorno per i pacchetti su questa rotta.

L'output di `ip route` e `ip -6 route` è il seguente:

1. Destinazione.
2. Indirizzo opzionale seguito dall'interfaccia.
3. Il protocollo di routing utilizzato per aggiungere il percorso.
4. L'ambito della rotta. Se questo è omesso, è l'ambito globale, o un gateway.
5. La metrica della rotta. Questo è usato dai protocolli di routing dinamico per determinare il costo della rotta. Non è usato dalla maggior parte dei sistemi.
6. Se è una rotta IPv6, la preferenza di rotta RFC4191.

Lavorare su alcuni esempi dovrebbe chiarire questo punto:

Esempio IPv4

```
default via 10.0.2.2 dev enp0s3 proto dhcp metric 100
```

1. La destinazione è la rotta predefinita.
2. L'indirizzo del gateway è `10.0.2.2` raggiungibile attraverso l'interfaccia `enp0s3`.
3. La destinazione è stata aggiunta alla tabella di routing dal DHCP.
4. L'ambito è stato omesso, quindi è globale.
5. La rotta ha un valore di costo di `100`.
6. Nessuna preferenza di rotta IPv6.

Esempio IPv6

```
fc0::/64 dev enp0s8 proto kernel metric 256 pref medium
```

1. La destinazione è `fc0::/64`.
2. È raggiungibile attraverso l'interfaccia `enp0s8`.
3. La destinazione è stata aggiunta alla tabella di routing dal kernel.
4. Lo scopo è stato omesso, quindi è globale.
5. La rotta ha un valore di costo di `256`.
6. Ha una preferenza IPv6 di `medium`.

Gestire le Rotte

Le rotte possono essere gestite usando `route` o `ip route`. Di seguito è riportato un esempio di aggiunta e rimozione di una rotta usando il comando `route`. Con `route`, devi usare l'opzione `-6` per IPv6:

```
# ping6 -c 2 2001:db8:1::20
connect: Network is unreachable
# route -6 add 2001:db8:1::/64 gw 2001:db8::3
# ping6 -c 2 2001:db8:1::20
PING 2001:db8:1::20(2001:db8:1::20) 56 data bytes
64 bytes from 2001:db8:1::20: icmp_seq=1 ttl=64 time=0.451 ms
64 bytes from 2001:db8:1::20: icmp_seq=2 ttl=64 time=0.438 ms
# route -6 del 2001:db8:1::/64 gw 2001:db8::3
# ping6 -c 2 2001:db8:1::20
connect: Network is unreachable
```

Lo stesso esempio usando il comando `ip route`:

```
# ping6 -c 2 2001:db8:1:20
connect: Network is unreachable
# ip route add 2001:db8:1::/64 via 2001:db8::3
# ping6 -c 2 2001:db8:1:20
PING 2001:db8:1::20(2001:db8:1::20) 56 data bytes
64 bytes from 2001:db8:1::20: icmp_seq=2 ttl=64 time=0.529 ms
64 bytes from 2001:db8:1::20: icmp_seq=2 ttl=64 time=0.438 ms
# ip route del 2001:db8:1::/64 via 2001:db8::3
# ping6 -c 2 2001:db8:1::20
connect: Network is unreachable
```

Esercizi Guidati

1. Quali comandi possono essere usati per elencare le interfacce di rete?

2. Come si fa a disabilitare temporaneamente un'interfaccia? Come la riabiliteresti?

3. Quale delle seguenti è una subnet mask possibile per IPv4?

`0.0.0.255`	
`255.0.255.0`	
`255.252.0.0`	
`/24`	

4. Quali comandi puoi usare per verificare la tua rotta predefinita?

5. Come si può aggiungere un secondo indirizzo IP a un'interfaccia?

Esercizi Esplorativi

1. Quale sottocomando di `ip` può essere usato per configurare il *vlan tagging*?

2. Come si configura una rotta predefinita?

3. Come si possono ottenere informazioni dettagliate sul comando `ip neighbour`? Che cosa succede se lo esegui da solo?

4. Come eseguiresti il backup della tua tabella di routing? Come la ripristineresti?

5. Quale sottocomando `ip` può essere usato per configurare le opzioni dello *spanning tree*?

Sommario

Il networking è solitamente configurato dagli script di avvio di un sistema o da un assistente come NetworkManager. La maggior parte delle distribuzioni ha strumenti che modificano i file di configurazione degli script d'avvio. Consulta la documentazione della distribuzione per maggiori dettagli.

Essere in grado di configurare manualmente la rete permette di risolvere i problemi in modo più efficace. È utile in ambienti minimali, utilizzati per operazioni quali il ripristino da backup o la migrazione a un nuovo hardware. Le utilità trattate in questa sezione hanno più funzionalità di quelle trattate. Sarebbe utile dare un'occhiata alla pagina `man` di ciascuna per familiarizzare con le opzioni disponibili. I comandi `ss` e `ip` rappresentano il modo moderno di fare le cose, mentre gli altri che sono trattati, sebbene ancora di uso comune, sono considerati strumenti *legacy*.

Il modo migliore per familiarizzare con gli strumenti trattati è la pratica. Usando un computer con una modesta quantità di RAM è possibile impostare un laboratorio di rete virtuale usando macchine virtuali con cui fare pratica. Tre *virtual machine* sono sufficienti per prendere confidenza con gli strumenti elencati.

I comandi usati in questa lezione includono:

`ifconfig`
: utilità *legacy* usata per configurare le interfacce di rete e rivedere i loro stati.

`ip`
: Utilità moderna e versatile usata per configurare le interfacce di rete e rivedere i loro stati.

`netstat`
: Comando *legacy* usato per visualizzare le connessioni di rete correnti e le informazioni sulle rotte.

`route`
: Comando legacy usato per visualizzare o modificare la tabella di routing di un sistema.

Risposte agli Esercizi Guidati

1. Quali comandi possono essere usati per elencare le interfacce di rete?

   Uno dei comandi sottostanti:

   `ip link`, `ifconfig -a` o `ls /sys/class/net`

2. Come si fa a disabilitare temporaneamente un'interfaccia? Come la riabiliteresti?

   Puoi utilizzare `ifconfig` o `ip link`:

   Usando `ifconfig`:

   ```
   $ ifconfig wlan1 down
   $ ifconfig wlan1 up
   ```

   Usando `ip link`:

   ```
   $ ip link set wlan1 down
   $ ip link set wlan1 up
   ```

3. Quale delle seguenti è una subnet mask possibile per IPv4?
   - `255.252.0.0`
   - `/24`

     Le altre maschere elencate non sono valide perché non separano l'indirizzo in modo chiaro in due sezioni, la prima parte che definisce la rete e la seconda l'host. I bit più a sinistra di una maschera saranno sempre `1` e i bit a destra saranno sempre `0`.

4. Quali comandi puoi usare per verificare la tua rotta predefinita?

   Puoi utlizzare `route`, `netstat -r`, o `ip route`:

   ```
   $ route
   Kernel IP routing table
   Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
   default         server          0.0.0.0         UG    600    0        0 wlan1
   192.168.1.0     0.0.0.0         255.255.255.0   U     600    0        0 wlan1
   $ netstat -r
   ```

```
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
default         server          0.0.0.0         UG        0 0          0 wlan1
192.168.1.0     0.0.0.0         255.255.255.0   U         0 0          0 wlan1
$ ip route
default via 192.168.1.20 dev wlan1 proto static metric 600
192.168.1.0/24 dev wlan1 proto kernel scope link src 192.168.1.24 metric 600
```

5. Come si può aggiungere un secondo indirizzo IP a un'interfaccia?

   Puoi usare `ip address` o `ifconfig`. Tieni presente che `ifconfig` è uno strumento *legacy*:

   ```
   $ ip addr add 172.16.15.16/16 dev enp0s9 label enp0s9:sub1
   ```

   La parte del comando `label enp0s9:sub1` aggiunge un alias a `enp0s9`. Se non usi il *legacy* `ifconfig` puoi ometterlo. Se lo fai, il comando funzionerà ancora, ma l'indirizzo che hai appena aggiunto non apparirà nell'output di `ifconfig`.

   Puoi usare anche `ifconfig`:

   ```
   $ ifconfig enp0s9:sub1 172.16.15.16/16
   ```

Risposte agli Esercizi Esplorativi

1. Quale sottocomando di `ip` può essere usato per configurare il *vlan tagging*?

   `ip link` ha un'opzione `vlan` che può essere usata. Di seguito è riportato un esempio di *tagging* di una sottointerfaccia con vlan 20.

   ```
   # ip link add link enp0s9 name enp0s9.20 type vlan id 20
   ```

2. Come si configura una rotta predefinita?

   Utilizzando `route` o `ip route`:

   ```
   # route add default gw 192.168.1.1
   # ip route add default via 192.168.1.1
   ```

3. Come si possono ottenere informazioni dettagliate sul comando `ip neighbour`? Cosa succede se lo esegui da solo?

   Leggendo la pagina `man`:

   ```
   $ man ip-neigbour
   ```

   Visualizza la cache ARP:

   ```
   $ ip neighbour
   10.0.2.2 dev enp0s3 lladdr 52:54:00:12:35:02 REACHABLE
   ```

4. Come faresti il backup della tua tabella di routing? Come la ripristineresti?

   L'esempio seguente mostra il backup e il ripristino di una tabella di routing:

   ```
   # ip route save > /root/routes/route_backup
   # ip route restore < /root/routes/route_backup
   ```

5. Quale sottocomando `ip` può essere usato per configurare le opzioni dello *spanning tree*?

   Similmente alla gestione delle impostazioni *vlan*, `ip link` può configurare lo spanning tree usando il tipo `bridge`. L'esempio mostra l'aggiunta di un'interfaccia virtuale con una priorità

STP di 50:

```
# ip link add link enp0s9 name enp0s9.50 type bridge priority 50
```

109.3 Lezione 2

Certificazione:	LPIC-1
Versione:	5.0
Argomento:	109 Fondamenti di Networking
Obiettivo:	109.3 Risoluzione dei problemi di base di una rete
Lezione:	2 di 2

Introduzione

I sistemi operativi basati su Linux hanno una varietà di strumenti con cui risolvere tutta una serie di problematiche di rete. Questa lezione tratterà alcuni dei più comuni. A questo punto dovresti avere una comprensione del modello OSI o di altri modelli di rete a strati, dell'indirizzamento IPv4 o IPv6, e una conoscenza di base di routing e switching.

Il modo migliore per testare una connessione di rete è provare a usare un'applicazione. Quando questo metodo non funziona, ci sono molti strumenti disponibili per aiutarti a diagnosticare il problema.

Fare Test di Connessione con `ping`

I comandi `ping` e `ping6` possono essere usati per inviare una richiesta *ICMP echo* a un indirizzo IPv4 o IPv6. Un messaggio ICMP *echo request* invia una piccola quantità di dati all'indirizzo di destinazione. Se l'indirizzo di destinazione è raggiungibile, invierà un messaggio di ICMP *echo reply* al mittente con gli stessi dati che gli sono stati inviati:

```
$ ping -c 3 192.168.50.2
PING 192.168.50.2 (192.168.50.2) 56(84) bytes of data.
64 bytes from 192.168.50.2: icmp_seq=1 ttl=64 time=0.525 ms
64 bytes from 192.168.50.2: icmp_seq=2 ttl=64 time=0.419 ms
64 bytes from 192.168.50.2: icmp_seq=3 ttl=64 time=0.449 ms

--- 192.168.50.2 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2006ms
rtt min/avg/max/mdev = 0.419/0.464/0.525/0.047 ms
```

```
$ ping6 -c 3 2001:db8::10
PING 2001:db8::10(2001:db8::10) 56 data bytes
64 bytes from 2001:db8::10: icmp_seq=1 ttl=64 time=0.425 ms
64 bytes from 2001:db8::10: icmp_seq=2 ttl=64 time=0.480 ms
64 bytes from 2001:db8::10: icmp_seq=3 ttl=64 time=0.725 ms

--- 2001:db8::10 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2000ms
rtt min/avg/max/mdev = 0.425/0.543/0.725/0.131 ms
```

L'opzione `-c` è usata per specificare il numero di pacchetti da inviare. Se ometti questa opzione, `ping` e `ping6` continueranno a inviare pacchetti fino a quando non li fermerai, tipicamente con la combinazione di tastiera `Ctrl` + `C`.

Solo perché non si può eseguire il ping di un host, questo non significa che non ci si possa connettere. Molte organizzazioni hanno *firewall* o liste di controllo degli accessi ai router che bloccano tutto tranne il minimo necessario per il funzionamento dei loro sistemi. Questo include le ICMP *echo request* e *echo reply*. Poiché questi pacchetti possono includere dati arbitrari, un hacker intelligente potrebbe usarli per esfiltrare dati.

Tracciare le Rotte

I programmi `traceroute` e `traceroute6` possono essere usati per mostrare il percorso che un pacchetto fa per arrivare a destinazione. Lo fanno inviando più pacchetti a destinazione, incrementando il campo *Time-To-Live* (TTL) dell'*IP header* con ogni pacchetto successivo. Ogni router lungo il percorso risponderà con un messaggio ICMP *TTL exceeded*:

```
$ traceroute 192.168.1.20
traceroute to 192.168.1.20 (192.168.1.20), 30 hops max, 60 byte packets
 1  10.0.2.2 (10.0.2.2)  0.396 ms  0.171 ms  0.132 ms
```

```
 2  192.168.1.20 (192.168.1.20)  2.665 ms  2.573 ms  2.573 ms
$ traceroute 192.168.50.2
traceroute to 192.168.50.2 (192.168.50.2), 30 hops max, 60 byte packets
 1  192.168.50.2 (192.168.50.2)  0.433 ms  0.273 ms  0.171 ms
$ traceroute6 2001:db8::11
traceroute to 2001:db8::11 (2001:db8::11), 30 hops max, 80 byte packets
 1  2001:db8::11 (2001:db8::11)  0.716 ms  0.550 ms  0.641 ms
$ traceroute 2001:db8::11
traceroute to 2001:db8::11 (2001:db8::11), 30 hops max, 80 byte packets
 1  2001:db8::10 (2001:db8::11)  0.617 ms  0.461 ms  0.387 ms
$ traceroute net2.example.net
traceroute to net2.example.net (192.168.50.2), 30 hops max, 60 byte packets
 1  net2.example.net (192.168.50.2)  0.533 ms  0.529 ms  0.504 ms
$ traceroute6 net2.example.net
traceroute to net2.example.net (2001:db8::11), 30 hops max, 80 byte packets
 1  net2.example.net (2001:db8::11)  0.738 ms  0.607 ms  0.304 ms
```

Per impostazione predefinita `traceroute` invia 3 pacchetti UDP con dati "spazzatura" alla porta 33434, incrementandola di un valore ad ogni invio successivo. Ogni linea nell'output del comando è un'interfaccia del router che il pacchetto attraversa. Il tempo mostrato in ogni riga dell'output è il tempo di andata e ritorno per ogni pacchetto. L'indirizzo IP è l'indirizzo dell'interfaccia del router in questione. Se `traceroute` è in grado di farlo, usa il nome DNS dell'interfaccia del router. A volte vedrai `*` al posto del tempo. Quando questo accade, significa che `traceroute` non ha mai ricevuto il messaggio *TTL exceeded* per questo pacchetto. Questo comportamento indica spesso che l'ultima risposta è l'ultimo *hop* del percorso.

Se hai accesso a `root`, l'opzione `-I` imposterà `traceroute` nell'usare le ICMP *echo request* invece dei pacchetti UDP. Questo metodo è spesso più efficace di UDP perché è più probabile che l'host di destinazione risponda a una ICMP *echo request* invece che a un pacchetto UDP:

```
# traceroute -I learning.lpi.org
traceroute to learning.lpi.org (208.94.166.201), 30 hops max, 60 byte packets
 1  047-132-144-001.res.spectrum.com (47.132.144.1)  9.764 ms  9.702 ms  9.693 ms
 2  096-034-094-106.biz.spectrum.com (96.34.94.106)  8.389 ms  8.481 ms  8.480 ms
 3  dtr01hlrgnc-gbe-4-15.hlrg.nc.charter.com (96.34.64.172)  8.763 ms  8.775 ms  8.770 ms
 4  acr01mgtnnc-vln-492.mgtn.nc.charter.com (96.34.67.202)  27.080 ms  27.154 ms  27.151 ms
 5  bbr01gnvlsc-bue-3.gnvl.sc.charter.com (96.34.2.112)  31.339 ms  31.398 ms  31.395 ms
 6  bbr01aldlmi-tge-0-0-0-13.aldl.mi.charter.com (96.34.0.161)  39.092 ms  38.794 ms  38.821
ms
 7  prr01ashbva-bue-3.ashb.va.charter.com (96.34.3.51)  34.208 ms  36.474 ms  36.544 ms
 8  bx2-ashburn.bell.ca (206.126.236.203)  53.973 ms  35.975 ms  38.250 ms
 9  tcore4-ashburnbk_0-12-0-0.net.bell.ca (64.230.125.190)  66.315 ms  65.319 ms  65.345 ms
```

```
10  tcore4-toronto47_2-8-0-3.net.bell.ca (64.230.51.22)  67.427 ms  67.502 ms  67.498 ms
11  agg1-toronto47_xe-7-0-0_core.net.bell.ca (64.230.161.114)  61.270 ms  61.299 ms  61.291
ms
12  dis4-clarkson16_5-0.net.bell.ca (64.230.131.98)  61.101 ms  61.177 ms  61.168 ms
13  207.35.12.142 (207.35.12.142)  70.009 ms  70.069 ms  59.893 ms
14  unassigned-117.001.centrilogic.com (66.135.117.1)  61.778 ms  61.950 ms  63.041 ms
15  unassigned-116.122.akn.ca (66.135.116.122)  62.702 ms  62.759 ms  62.755 ms
16  208.94.166.201 (208.94.166.201)  62.936 ms  62.932 ms  62.921 ms
```

Alcune organizzazioni bloccano le ICMP *echo request* e *echo reply*. Per aggirare questo problema, si può usare il TCP. Usando una porta TCP conosciuta e aperta, puoi garantire che l'host di destinazione risponda. Per usare TCP, usa l'opzione `-T` insieme a `-p` per specificare la porta. Come per le ICMP *echo request*, devi avere accesso a `root` per farlo:

```
# traceroute -m 60 -T -p 80  learning.lpi.org
traceroute to learning.lpi.org (208.94.166.201), 60 hops max, 60 byte packets
 1  * * *
 2  096-034-094-106.biz.spectrum.com (96.34.94.106)  12.178 ms  12.229 ms  12.175 ms
 3  dtr01hlrgnc-gbe-4-15.hlrg.nc.charter.com (96.34.64.172)  12.134 ms  12.093 ms  12.062 ms
 4  acr01mgtnnc-vln-492.mgtn.nc.charter.com (96.34.67.202)  31.146 ms  31.192 ms  31.828 ms
 5  bbr01gnvlsc-bue-3.gnvl.sc.charter.com (96.34.2.112)  39.057 ms  46.706 ms  39.745 ms
 6  bbr01aldlmi-tge-0-0-0-13.aldl.mi.charter.com (96.34.0.161)  50.590 ms  58.852 ms  58.841
ms
 7  prr01ashbva-bue-3.ashb.va.charter.com (96.34.3.51)  34.556 ms  37.892 ms  38.274 ms
 8  bx2-ashburn.bell.ca (206.126.236.203)  38.249 ms  36.991 ms  36.270 ms
 9  tcore4-ashburnbk_0-12-0-0.net.bell.ca (64.230.125.190)  66.779 ms  63.218 ms tcore3-
ashburnbk_100ge0-12-0-0.net.bell.ca (64.230.125.188)  60.441 ms
10  tcore4-toronto47_2-8-0-3.net.bell.ca (64.230.51.22)  63.932 ms  63.733 ms  68.847 ms
11  agg2-toronto47_xe-7-0-0_core.net.bell.ca (64.230.161.118)  60.144 ms  60.443 ms agg1-
toronto47_xe-7-0-0_core.net.bell.ca (64.230.161.114)  60.851 ms
12  dis4-clarkson16_5-0.net.bell.ca (64.230.131.98)  67.246 ms dis4-clarkson16_7-
0.net.bell.ca (64.230.131.102)  68.404 ms dis4-clarkson16_5-0.net.bell.ca (64.230.131.98)
67.403 ms
13  207.35.12.142 (207.35.12.142)  66.138 ms  60.608 ms  64.656 ms
14  unassigned-117.001.centrilogic.com (66.135.117.1)  70.690 ms  62.190 ms  61.787 ms
15  unassigned-116.122.akn.ca (66.135.116.122)  62.692 ms  69.470 ms  68.815 ms
16  208.94.166.201 (208.94.166.201)  61.433 ms  65.421 ms  65.247 ms
17  208.94.166.201 (208.94.166.201)  64.023 ms  62.181 ms  61.899 ms
```

Come `ping`, anche `traceroute` ha i suoi limiti. È possibile che *firewall* e *router* blocchino i pacchetti inviati o restituiti da `traceroute`. Se hai accesso come "root", ci sono opzioni che possono aiutarti a ottenere risultati accurati.

Trovare le MTU con `tracepath`

Il comando `tracepath` è simile a `traceroute`. La differenza è che tiene traccia delle dimensioni della *Maximum Transmission Unit* (MTU) lungo il percorso. L'MTU è un'impostazione configurata su un'interfaccia di rete o una limitazione hardware dell'unità di dati di protocollo più grande che può trasmettere o ricevere. Il programma `tracepath` funziona allo stesso modo di `traceroute` nel senso che incrementa il TTL con ogni pacchetto. Si differenzia inviando un *datagramma* UDP molto grande. È quasi inevitabile che il datagramma sia più grande del dispositivo con il più piccolo MTU lungo il percorso. Quando il pacchetto raggiunge questo dispositivo, il dispositivo tipicamente risponde con un pacchetto di destinazione non raggiungibile. Il pacchetto ICMP *destination unreachable* ha un campo per l'MTU del link su cui invierebbe il pacchetto se fosse in grado di farlo. `tracepath` invia quindi tutti i pacchetti successivi con questa dimensione:

```
$ tracepath 192.168.1.20
 1?: [LOCALHOST]                                         pmtu 1500
 1:  10.0.2.2                                              0.321ms
 1:  10.0.2.2                                              0.110ms
 2:  192.168.1.20                                          2.714ms reached
     Resume: pmtu 1500 hops 2 back 64
```

A differenza di `traceroute`, devi usare esplicitamente `tracepath6` per IPv6:

```
$ tracepath 2001:db8::11
tracepath: 2001:db8::11: Address family for hostname not supported
$ tracepath6 2001:db8::11
 1?: [LOCALHOST]                      0.027ms pmtu 1500
 1:  net2.example.net                                      0.917ms reached
 1:  net2.example.net                                      0.527ms reached
     Resume: pmtu 1500 hops 1 back 1
```

L'output è simile a quello di `traceroute`. Il vantaggio di `tracepath` è che nell'ultima riga mostra il più piccolo MTU sull'intero collegamento. Questo può essere utile per la risoluzione dei problemi delle connessioni che non possono gestire la frammentazione dei pacchetti.

Come per i precedenti strumenti di risoluzione dei problemi, c'è comunque la possibilità che qualche apparecchiatura blocchi i pacchetti.

Creare Connessioni Arbitrarie

Il programma `nc`, conosciuto come *netcat*, può inviare o ricevere dati arbitrari su una connessione

di rete TCP o UDP. I seguenti esempi dovrebbero rendere chiare le sue funzionalità.

Ecco un esempio per attivare una porta in ascolto sulla porta `1234`:

```
$ nc -l 1234
LPI Example
```

L'output di `LPI Example` appare dopo l'esempio seguente, che creerà un *mittente netcat* per inviare pacchetti a `net2.example.net` sulla porta `1234`. L'opzione `-l` è usata per specificare che desideri che `nc` riceva i dati invece di inviarli:

```
$ nc net2.example.net 1234
LPI Example
```

Premi `Ctrl` + `C` su entrambi i sistemi per interrompere la connessione.

Netcat funziona con indirizzi IPv4 e IPv6. Funziona sia con TCP che con UDP. Può anche essere usato per impostare una shell remota minimale.

WARNING

Nota che non tutte le installazioni di `nc` supportano lo switch `-e`. Assicurati di rivedere le pagine `man` della tua installazione per avere informazioni sulla sicurezza di questa opzione e sui metodi alternativi per eseguire comandi su un sistema remoto.

```
$ hostname
net2
$ nc -u -e /bin/bash -l 1234
```

L'opzione `-u` è per UDP. L'opzione `-e` ordina a netcat di inviare tutto ciò che riceve allo standard input dell'eseguibile che lo segue. In questo esempio, `/bin/bash`.

```
$ hostname
net1
$ nc -u net2.example.net 1234
hostname
net2
pwd
/home/emma
```

Nota come l'output del comando `hostname` corrisponde a quello dell'host in ascolto e l'output del

comando `pwd` a una directory.

Visualizzazione delle Connessioni Attive e/o in Ascolto

I programmi `netstat` e `ss` possono essere usati per visualizzare lo stato delle connessioni attive e/o in ascolto. Come per `ifconfig`, `netstat` è uno strumento *legacy*. Sia `netstat` sia `ss` hanno output e opzioni simili. Qui alcune opzioni disponibili per entrambi i programmi:

`-a`
: Mostra tutti i *socket*.

`-l`
: Mostra solo i *socket* in ascolto.

`-p`
: Mostra il processo associato alla connessione.

`-n`
: Evita la risoluzione del nome sia per le porte sia per gli indirizzi host.

`-t`
: Mostra le connessioni TCP.

`-u`
: Mostra le connessioni UDP.

Gli esempi qui sotto mostrano l'output di un set di opzioni comunemente usato per entrambi i programmi:

```
# netstat -tulnp
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program
name
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      892/sshd
tcp        0      0 127.0.0.1:25            0.0.0.0:*               LISTEN      1141/master
tcp6       0      0 :::22                   :::*                    LISTEN      892/sshd
tcp6       0      0 ::1:25                  :::*                    LISTEN      1141/master
udp        0      0 0.0.0.0:68              0.0.0.0:*                           692/dhclient
# ss -tulnp
# ss -tulnp
Netid  State      Recv-Q Send-Q      Local Address:Port                     Peer
```

```
Address:Port
udp    UNCONN     0      0                        :68                                    *:
users:(("dhclient",pid=693,fd=6))
tcp    LISTEN     0      128                      :22                                    *:
users:(("sshd",pid=892,fd=3))
tcp    LISTEN     0      100             127.0.0.1:25                                     :
users:(("master",pid=1099,fd=13))
tcp    LISTEN     0      128                  [::]:22                                 [::]:*
users:(("sshd",pid=892,fd=4))
tcp    LISTEN     0      100                 [::1]:25                                 [::]:*
users:(("master",pid=1099,fd=14))
```

La colonna `Recv-Q` è il numero di pacchetti che un socket ha ricevuto ma non passato al suo programma. La colonna `Send-Q` è il numero di pacchetti che un socket ha inviato e che non sono stati riconosciuti dal ricevitore. Il resto delle colonne sono auto esplicative.

Esercizi Guidati

1. Quale comando(i) useresti per inviare un ICMP *echo request* a `learning.lpi.org`?

2. Come puoi determinare il percorso verso `8.8.8.8`?

3. Quale comando ti mostrerebbe se un processo è in ascolto sulla porta TCP 80?

4. Come puoi trovare quale processo è in ascolto su una porta?

5. Come si può determinare la MTU massima di un percorso di rete?

Esercizi Esplorativi

1. Come potresti usare netcat per inviare una richiesta HTTP a un server web?

2. Quali possono essere i motivi per cui il ping verso un host può fallire?

3. Nomina uno strumento che potresti usare per vedere i pacchetti di rete che raggiungono o lasciano un host Linux.

4. Come si può forzare `traceroute` a usare un'interfaccia diversa?

5. È possibile che `traceroute` riporti gli MTU?

Sommario

Il networking è solitamente configurato dagli script di avvio di un sistema o da un assistente come NetworkManager. La maggior parte delle distribuzioni ha strumenti che modificano i file di configurazione degli script d'avvio. Consulta la documentazione della distribuzione per maggiori dettagli.

Essere in grado di configurare manualmente la rete permette di risolvere i problemi in modo più efficace. È utile in ambienti minimali, utilizzati per operazioni quali il ripristino da backup o la migrazione a un nuovo hardware. Le utilità trattate in questa sezione hanno più funzionalità di quelle trattate. Sarebbe utile dare un'occhiata alla pagina `man` di ciascuna per familiarizzare con le opzioni disponibili. I comandi `ss` e `ip` rappresentano il modo moderno di fare le cose, mentre gli altri che sono trattati, sebbene ancora di uso comune, sono considerati strumenti *legacy*.

Il modo migliore per familiarizzare con gli strumenti trattati è la pratica. Usando un computer con una modesta quantità di RAM, è possibile impostare un laboratorio di rete virtuale con macchine virtuali con cui fare pratica. Tre *virtual machine* sono sufficienti per prendere confidenza con gli strumenti elencati.

I comandi usati in questa lezione includono:

`ping` e **`ping6`**

Utilizzati per trasmettere pacchetti ICMP a un host remoto per testare la disponibilità di una connessione di rete.

`traceroute` e **`traceroute6`**

Utilizzati per tracciare un percorso attraverso una rete per determinarne la connettività.

`tracepath` e **`tracepath6`**

Utilizzato per tracciare un percorso attraverso una rete e per determinare le dimensioni MTU lungo un percorso.

`nc`

Usato per impostare connessioni arbitrarie su una rete per testare la connettività, così come per interrogare una rete per servizi e dispositivi disponibili.

`netstat`

Comando *legacy* usato per determinare le connessioni di rete aperte di un sistema e le statistiche.

`ss`

Comando moderno usato per determinare le connessioni di rete aperte e le statistiche di un sistema.

Risposte agli Esercizi Guidati

1. Quale comando(i) useresti per inviare un ICMP *echo request* a `learning.lpi.org`?

   Useresti `ping` o `ping6`:

   ```
   $ ping learning.lpi.org
   ```

   o

   ```
   $ ping6 learning.lpi.org
   ```

2. Come puoi determinare il percorso verso `8.8.8.8`?

   Usando i comandi `tracepath` o `traceroute`.

   ```
   $ tracepath 8.8.8.8
   ```

   o

   ```
   $ traceroute 8.8.8.8
   ```

3. Quale comando ti mostrerebbe se c'è un processo in ascolto sulla porta TCP 80?

   Con `ss`:

   ```
   $ ss -ln | grep ":80"
   ```

   Con `netstat`:

   ```
   $ netstat -ln | grep ":80"
   ```

   Anche se non è elencato come requisito per l'esame, puoi anche usare `lsof`:

   ```
   # lsof -Pi:80
   ```

4. Come puoi trovare quale processo è in ascolto su una porta?

Di nuovo, ci sono diversi modi per farlo. Potresti usare `lsof` come nella risposta precedente, sostituendo il numero di porta. Potresti anche usare `netstat` o `ss` con l'opzione `-p`. Ricorda che `netstat` è considerato uno strumento *legacy*.

```
# netstat -lnp | grep ":22"
```

Le stesse opzioni che funzionano con `netstat` funzionano anche con `ss`:

```
# ss -lnp | grep ":22"
```

5. Come si può determinare la MTU massima di un percorso di rete?

   Utilizzando il comando `tracepath`:

   ```
   $ tracepath somehost.example.com
   ```

Risposte agli Esercizi Esplorativi

1. Come potresti usare netcat per inviare una richiesta HTTP a un server web?

   Inserendo la linea di richiesta HTTP, qualsiasi intestazione e una linea vuota nel terminale:

```
$ nc learning.lpi.org 80
GET /index.html HTTP/1.1
HOST: learning.lpi.org

HTTP/1.1 302 Found
Location: https://learning.lpi.org:443/index.html
Date: Wed, 27 May 2020 22:54:46 GMT
Content-Length: 5
Content-Type: text/plain; charset=utf-8

Found
```

2. Quali possono essere i motivi per cui il ping verso un host può fallire?

   Ci sono diverse ragioni possibili. Eccone alcune:

   - L'host remoto è fuori uso.
   - Una *ACL* su un router sta bloccando il ping.
   - Il firewall dell'host remoto sta bloccando il ping.
   - State usando un nome o un indirizzo host errato.
   - La risoluzione del nome sta restituendo un indirizzo errato.
   - La configurazione di rete della macchina non è corretta.
   - Il firewall della macchina lo sta bloccando.
   - La configurazione di rete dell'host remoto non è corretta.
   - Le interfacce della macchina sono scollegate.
   - Le interfacce della macchina remota sono scollegate.
   - Un componente di rete come uno switch, un cavo o un router tra la macchina locale e quella remota non funziona più.

3. Nomina uno strumento che potresti usare per vedere i pacchetti di rete che raggiungono o lasciano un host Linux.

Possono essere usati sia `tcpdump` sia `wireshark`.

4. Come si può forzare `traceroute` a usare un'interfaccia diversa?

   Attraverso l'opzione `-i`:

```
$ traceroute -i eth2 learning.lpi.org
traceroute -i eth2 learning.lpi.org
traceroute to learning.lpi.org (208.94.166.201), 30 hops max, 60 byte packets
...
```

5. È possibile che `traceroute` riporti gli MTU?

   Si, attraverso l'opzione `--mtu`:

```
# traceroute -I --mtu  learning.lpi.org
traceroute to learning.lpi.org (208.94.166.201), 30 hops max, 65000 byte packets
 1  047-132-144-001.res.spectrum.com (47.132.144.1)  9.974 ms F=1500  10.476 ms  4.743 ms
 2  096-034-094-106.biz.spectrum.com (96.34.94.106)  8.697 ms  9.963 ms  10.321 ms
...
```



109.4 Configurare un client DNS

Obiettivi LPI di riferimento

LPIC-1 version 5.0, Exam 102, Objective 109.4

Peso

2

Aree di Conoscenza Chiave

- Interrogare server DNS remoti.
- Configurare la risoluzione dei nomi locali e utilizzare server DNS remoti.
- Modificare l'ordine in cui viene eseguita la risoluzione dei nomi.
- Debug degli errori relativi alla risoluzione dei nomi.
- Conoscenza di systemd-resolved.

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

- `/etc/hosts`
- `/etc/resolv.conf`
- `/etc/nsswitch.conf`
- `host`
- `dig`
- `getent`



109.4 Lezione 1

Certificazione:	LPIC-1
Versione:	5.0
Argomento:	109 Fondamenti di Networking
Obiettivo:	109.4 Configurare un client DNS
Lezione:	1 di 1

Introduzione

Questa lezione si concentra sulla configurazione della risoluzione dei nomi lato client e sull'uso di alcuni strumenti CLI per la risoluzione dei nomi.

Ricordarsi e mantenere a memoria indirizzi IP, UID e GID e altri numeri è una strada decisamente impraticabile. I servizi di risoluzione dei nomi traducono nomi facili da ricordare in numeri e viceversa. Questa lezione si concentra sulla risoluzione dei nomi degli host, ma un processo simile avviene per i nomi degli utenti, i nomi dei gruppi, i numeri delle porte e molto altro ancora.

Il Processo di Risoluzione dei Nomi

I programmi che risolvono i nomi in numeri usano quasi sempre funzioni fornite da una specifica libreria C standard, che sui sistemi Linux è la `glibc` del progetto *GNU*. La prima cosa che queste funzioni fanno è leggere il file `/etc/nsswitch.conf` per avere istruzioni su come risolvere un certo tipo di nome. Questa lezione è focalizzata sulla risoluzione dei nomi host, ma lo stesso processo si applica anche ad altri tipi di risoluzione dei nomi. Una volta che il processo legge `/etc/nsswitch.conf`, cerca il nome nel modo specificato. Poiché `/etc/nsswitch.conf` supporta i plugin, ciò che segue potrebbe essere particolarmente specifico. Dopo che la funzione

ha finito di cercare il nome o il numero, restituisce il risultato al processo chiamante.

Le Classi DNS

DNS ha tre classi di record: IN, HS e CH. In questa lezione, tutte le query DNS saranno di tipo IN. La classe IN è per gli indirizzi Internet che usano lo stack TCP/IP. CH sta per *ChaosNet*, che è una tecnologia di rete che ha avuto vita breve e non è più in uso. La classe HS corrisponde a *Hesiod*. Hesiod è un modo per memorizzare elementi quali voci di *passwd* e di *group* nel DNS. Hesiod va oltre lo scopo di questa lezione.

Comprendere `/etc/nsswitch.conf`

Il modo migliore per conoscere questo file è leggere la pagina `man` che fa parte del progetto Linux *man-pages* e che è disponibile sulla maggior parte delle distribuzioni. Vi si può accedere con il comando `man nsswitch.conf`. In alternativa, può essere trovato su https://man7.org/linux/man-pages/dir_section_5.html

Ecco di seguito un semplice esempio di `/etc/nsswitch.conf` dalla sua pagina man:

```
passwd:         compat
group:          compat
shadow:         compat

hosts:          dns [!UNAVAIL=return] files
networks:       nis [NOTFOUND=return] files
ethers:         nis [NOTFOUND=return] files
protocols:      nis [NOTFOUND=return] files
rpc:            nis [NOTFOUND=return] files
services:       nis [NOTFOUND=return] files
# This is a comment. It is ignored by the resolution functions.
```

Il file è organizzato in colonne. La colonna all'estrema sinistra è il tipo di database dei nomi. Il resto delle colonne riporta i metodi che le funzioni di risoluzione dovrebbero usare per cercare un nome. I metodi sono seguiti dalle funzioni, da sinistra a destra. Le colonne con `[]` sono usate per fornire una logica condizionale limitata alla colonna immediatamente a sinistra.

Supponiamo che un processo stia cercando di risolvere il nome host `learning.lpi.org` con una chiamata appropriata alla libreria C (molto probabilmente `gethostbyname`). Questa funzione leggerà poi `/etc/nsswitch.conf`. Poiché il processo sta cercando un nome di host, troverà la linea che inizia con `hosts`. Quindi tenterà di usare il DNS per risolvere il nome. La colonna successiva, `[!UNAVAIL=return]` significa che se il servizio *non è* indisponibile, allora non proverà

metodi ulteriori. Se il DNS non è disponibile, allora continua con la fonte successiva. In questo caso, la fonte successiva è `files`.

Quando vedi una colonna nel formato `[result=action]`, significa che quando una ricerca del resolver della colonna a sinistra di essa è `result`, allora viene eseguita `action`. Se `result` è preceduto da un `!`, significa che se il risultato non è `result`, allora viene eseguita `action`. Per le descrizioni dei possibili risultati e azioni, vedi la pagina `man`.

Supponiamo ora che un processo stia cercando di risolvere un numero di porta in un nome di servizio: leggerà la linea `services`. La prima fonte elencata è NIS. NIS sta per *Network Information Service* (a volte ci si riferisce al NIS come alle *yellow pages*, le pagine gialle). È un vecchio servizio che permetteva la gestione centrale di oggetti come per esempio gli utenti. Viene usato raramente a causa della sua scarsa sicurezza. La prossima colonna `[NOTFOUND=return]` indica se la ricerca ha avuto successo ma il servizio non è stato trovato, si deve smettere di cercare. Se la condizione di cui sopra non si applica, usa i file locali.

Qualsiasi cosa a destra di `#` è un commento e viene ignorato.

Il File `/etc/resolv.conf`

Il file `/etc/resolv.conf` è usato per configurare la risoluzione degli host tramite DNS. Alcune distribuzioni hanno script di avvio, demoni e altri strumenti che scrivono su questo file. Tienilo a mente quando lo modifichi manualmente. Controlla la documentazione della distribuzione e di qualsiasi strumento di configurazione di rete. Alcuni strumenti, come Network Manager, lasceranno un commento nel file per farti sapere che le modifiche manuali saranno sovrascritte.

Come per `/etc/nsswitch.conf`, c'è una pagina `man` associata al file. Vi si può accedere con il comando `man resolv.conf` o su https://man7.org/linux/man-pages/man5/resolv.conf.5.html.

Il formato del file è piuttosto semplice. Nella colonna all'estrema sinistra c'è l'opzione `name`. Il resto delle colonne sulla stessa linea è il valore dell'opzione.

L'opzione più comune è l'opzione `nameserver`. È usata per specificare l'indirizzo IPv4 o IPv6 di un server DNS. A oggi è possibile specificare fino a tre server di nomi. Se il tuo `/etc/resolv.conf` non ha un'opzione `nameserver`, il tuo sistema userà di default il name server della macchina locale.

Qui sotto c'è un semplice file di esempio che è rappresentativo delle configurazioni più comuni:

```
search lpi.org
nameserver 10.0.0.53
```

```
nameserver fd00:ffff::2:53
```

L'opzione `search` è usata per permettere ricerche in forma breve. Nell'esempio, è configurato un singolo dominio di ricerca di `lpi.org`. Questo significa che ogni tentativo di risolvere un nome host senza una porzione di dominio avrà `.lpi.org` aggiunto prima della ricerca. Per esempio, se dovessi provare a cercare un host chiamato `learning`, il resolver cercherà `learning.lpi.org`. Puoi avere fino a sei domini di ricerca configurati.

Un'altra opzione comune è l'opzione `domain`. Questa è usata per impostare il tuo nome di dominio locale. Se questa opzione non è presente, il default è tutto ciò che segue dopo il primo `.` nel nome host della macchina. Se il nome dell'host non contiene un `.`, si presume che la macchina faccia parte del dominio principale. Come `search`, `domain` può essere usato per ricerche di nomi brevi.

Tieni presente che `domain` e `search` si escludono a vicenda. Se entrambi sono presenti, viene usata l'ultima opzione dichiarata nel file.

Ci sono diverse opzioni che possono essere impostate per influenzare il comportamento del resolver. Per impostarle, usa la parola chiave `options`, seguita dal nome dell'opzione da impostare e, se applicabile, da un `:` seguito dal valore. Qui sotto c'è un esempio di impostazione dell'opzione *timeout*, che è la quantità di tempo in secondi che il resolver aspetterà dopo aver contattato un server di nomi prima di rinunciare:

```
option timeout:3
```

Ci sono altre opzioni in `resolv.conf`, ma queste sono le più comuni.

Il FIle `/etc/hosts`

Il file `/etc/hosts` è usato per risolvere i nomi in indirizzi IP e viceversa. Sono supportati sia IPv4 che IPv6. La colonna di sinistra è l'indirizzo IP, il resto sono nomi associati a quell'indirizzo. L'uso più comune di `/etc/hosts` è per gli host e gli indirizzi dove il DNS non è possibile, come gli indirizzi di loopback. Nell'esempio qui sotto, sono definiti gli indirizzi IP di componenti di infrastrutture critiche.

Ecco un esempio realistico di un file `/etc/hosts`:

```
127.0.0.1       localhost
127.0.1.1       proxy
::1             localhost ip6-localhost ip6-loopback
ff02::1         ip6-allnodes
ff02::2         ip6-allrouters
```

```
10.0.0.1        gateway.lpi.org gateway gw
fd00:ffff::1    gateway.lpi.org gateway gw

10.0.1.53       dns1.lpi.org
fd00:ffff::1:53 dns1.lpi.org
10.0.2.53       dns2.lpi.org
fd00:ffff::2:53 dns2.lpi.org
```

systemd-resolved

Systemd fornisce un servizio chiamato `systemd-resolved`. Fornisce mDNS, DNS e LLMNR. Quando è in esecuzione, ascolta le richieste DNS su `127.0.0.53`. Non fornisce un server DNS completo. Qualsiasi richiesta DNS che riceve viene cercata interrogando i server configurati in `/etc/systemd/resolv.conf` o `/etc/resolv.conf`. Se lo vuoi utilizzare, usa `resolve` per `hosts` in `/etc/nsswitch.conf`. Tieni presente che il pacchetto del sistema operativo che ha la libreria `systemd-resolved` potrebbe *non* essere installato di default.

Strumenti per la Risoluzione dei Nomi

Ci sono molti strumenti disponibili agli utenti Linux per la risoluzione dei nomi. Questa lezione ne tratta tre. Uno, `getent`, è utile per vedere come le richieste del mondo reale vengono risolte. Un altro è il comando `host`, che è utile per semplici query DNS. Il terzo strumento è un programma chiamato `dig`, utile per operazioni DNS complesse che possono aiutare nella risoluzione delle problematiche relative ai server DNS.

Il Comando `getent`

L'utilità `getent` è usata per visualizzare le voci dai database del servizio dei nomi. Può recuperare record da qualsiasi fonte configurabile in `/etc/nsswitch.conf`.

Per usare `getent`, segui il comando con il tipo di nome che vuoi risolvere; come opzione puoi aggiungere una voce specifica da cercare. Se si specifica solo il tipo di nome, `getent` tenterà di visualizzare *tutte* le voci di quel tipo di dati:

```
$ getent hosts
127.0.0.1       localhost
127.0.1.1       proxy
10.0.1.53       dns1.lpi.org
10.0.2.53       dns2.lpi.org
127.0.0.1       localhost ip6-localhost ip6-loopback
```

```
$ getent hosts dns1.lpi.org
fd00:ffff::1:53 dns1.lpi.org
```

A partire dalla versione 2.2.5 di *glibc*, puoi forzare `getent` a usare una specifica sorgente di dati tramite l'opzione `-s`. L'esempio qui sotto lo dimostra:

```
$ getent -s files hosts learning.lpi.org
::1             learning.lpi.org
$ getent -s dns hosts learning.lpi.org
208.94.166.198  learning.lpi.org
```

Il Comando `host`

`host` è un semplice programma per cercare voci DNS. Senza opzioni, se viene dato un nome a `host`, restituisce i record A, AAAA e MX. Se viene dato un indirizzo IPv4 o IPv6, restituisce il record PTR se disponibile:

```
$ host wikipedia.org
wikipedia.org has address 208.80.154.224
wikipedia.org has IPv6 address 2620:0:861:ed1a::1
wikipedia.org mail is handled by 10 mx1001.wikimedia.org.
wikipedia.org mail is handled by 50 mx2001.wikimedia.org.
$ host 208.80.154.224
224.154.80.208.in-addr.arpa domain name pointer text-lb.eqiad.wikimedia.org.
```

Se stai cercando un tipo di record specifico, puoi usare `host -t`:

```
$ host -t NS lpi.org
lpi.org name server dns1.easydns.com.
lpi.org name server dns3.easydns.ca.
lpi.org name server dns2.easydns.net.
$ host -t SOA lpi.org
lpi.org has SOA record dns1.easydns.com. zone.easydns.com. 1593109612 3600 600 1209600 300
```

`host` può anche essere usato per interrogare uno specifico *name server* se non si desidera usare quelli in `/etc/resolv.conf`. Aggiungi semplicemente l'indirizzo IP o il nome dell'host del server che vuoi usare come ultimo argomento:

```
$ host -t MX lpi.org dns1.easydns.com
```

```
Using domain server:
Name: dns1.easydns.com
Address: 64.68.192.10#53
Aliases:

lpi.org mail is handled by 10 aspmx4.googlemail.com.
lpi.org mail is handled by 10 aspmx2.googlemail.com.
lpi.org mail is handled by 5 alt1.aspmx.l.google.com.
lpi.org mail is handled by 0 aspmx.l.google.com.
lpi.org mail is handled by 10 aspmx5.googlemail.com.
lpi.org mail is handled by 10 aspmx3.googlemail.com.
lpi.org mail is handled by 5 alt2.aspmx.l.google.com.
```

Il Comando `dig`

Un altro strumento per interrogare i server DNS è `dig`. Questo comando è molto più verboso di `host`. Per impostazione predefinita, `dig` interroga i record A. Probabilmente è anche troppo verboso per cercare semplicemente un indirizzo IP o un nome host. `dig` funzionerà per semplici ricerche, ma è più adatto per la risoluzione dei problemi di configurazione dei server DNS:

```
$ dig learning.lpi.org

; <<>> DiG 9.11.5-P4-5.1+deb10u1-Debian <<>> learning.lpi.org
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 63004
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 3, ADDITIONAL: 5

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: ca7a415be1cec45592b082665ef87f3483b81ddd61063c30 (good)
;; QUESTION SECTION:
;learning.lpi.org.      IN  A

;; ANSWER SECTION:
learning.lpi.org.   600 IN  A   208.94.166.198

;; AUTHORITY SECTION:
lpi.org.        86400   IN  NS  dns2.easydns.net.
lpi.org.        86400   IN  NS  dns1.easydns.com.
lpi.org.        86400   IN  NS  dns3.easydns.ca.

;; ADDITIONAL SECTION:
```

```
dns1.easydns.com.   172682  IN  A   64.68.192.10
dns2.easydns.net.   170226  IN  A   198.41.222.254
dns1.easydns.com.   172682  IN  AAAA    2400:cb00:2049:1::a29f:1835
dns2.easydns.net.   170226  IN  AAAA    2400:cb00:2049:1::c629:defe

;; Query time: 135 msec
;; SERVER: 192.168.1.20#53(192.168.1.20)
;; WHEN: Sun Jun 28 07:29:56 EDT 2020
;; MSG SIZE  rcvd: 266
```

Come puoi vedere, `dig` fornisce molte informazioni. L'output è diviso in sezioni. La prima sezione mostra informazioni sulla versione di `dig` installata e sulla query inviata, insieme a qualsiasi opzione usata per il comando. Poi mostra informazioni sulla query e sulla risposta.

La prossima sezione mostra le informazioni sulle estensioni EDNS utilizzate e la query. Nell'esempio, viene usata l'estensione cookie. `dig` sta cercando un record A per `learning.lpi.org`.

La sezione successiva mostra il risultato della query. Il numero nella seconda colonna è il TTL della risorsa in secondi.

Il resto dell'output fornisce informazioni sui name server del dominio, compresi i record NS per il server insieme ai record A e AAAA dei server nel record NS del dominio.

Come `host` puoi specificare un tipo di record con l'opzione `-t`:

```
$ dig -t SOA lpi.org

; <<>> DiG 9.11.5-P4-5.1+deb10u1-Debian <<>> -t SOA lpi.org
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 16695
;; flags: qr rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 3, ADDITIONAL: 6

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 185c67140a63baf46c4493215ef8906f7bfbe15bdca3b01a (good)
;; QUESTION SECTION:
;lpi.org.           IN  SOA

;; ANSWER SECTION:
lpi.org.        600 IN  SOA dns1.easydns.com. zone.easydns.com. 1593109612 3600 600 1209600
300
```

```
;; AUTHORITY SECTION:
lpi.org.        81989   IN  NS  dns1.easydns.com.
lpi.org.        81989   IN  NS  dns2.easydns.net.
lpi.org.        81989   IN  NS  dns3.easydns.ca.

;; ADDITIONAL SECTION:
dns1.easydns.com.   168271  IN  A   64.68.192.10
dns2.easydns.net.   165815  IN  A   198.41.222.254
dns3.easydns.ca.    107 IN  A   64.68.196.10
dns1.easydns.com.   168271  IN  AAAA    2400:cb00:2049:1::a29f:1835
dns2.easydns.net.   165815  IN  AAAA    2400:cb00:2049:1::c629:defe

;; Query time: 94 msec
;; SERVER: 192.168.1.20#53(192.168.1.20)
;; WHEN: Sun Jun 28 08:43:27 EDT 2020
;; MSG SIZE  rcvd: 298
```

`dig` ha molte opzioni per mettere a punto sia l'output sia la query inviata al server. Queste opzioni iniziano con `+`. Una tra queste è l'opzione `short`, che sopprime tutti gli output tranne il risultato:

```
$ dig +short lpi.org
65.39.134.165
$ dig +short -t SOA lpi.org
dns1.easydns.com. zone.easydns.com. 1593109612 3600 600 1209600 300
```

Ecco un esempio di disattivazione dell'estensione cookie EDNS:

```
$ dig +nocookie -t MX lpi.org

; <<>> DiG 9.11.5-P4-5.1+deb10u1-Debian <<>> +nocookie -t MX lpi.org
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 47774
;; flags: qr rd ra; QUERY: 1, ANSWER: 7, AUTHORITY: 3, ADDITIONAL: 5

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;lpi.org.           IN  MX

;; ANSWER SECTION:
```

```
lpi.org.        468 IN  MX  0 aspmx.l.google.com.
lpi.org.        468 IN  MX  10 aspmx4.googlemail.com.
lpi.org.        468 IN  MX  10 aspmx5.googlemail.com.
lpi.org.        468 IN  MX  10 aspmx2.googlemail.com.
lpi.org.        468 IN  MX  10 aspmx3.googlemail.com.
lpi.org.        468 IN  MX  5 alt2.aspmx.l.google.com.
lpi.org.        468 IN  MX  5 alt1.aspmx.l.google.com.

;; AUTHORITY SECTION:
lpi.org.        77130   IN  NS  dns2.easydns.net.
lpi.org.        77130   IN  NS  dns3.easydns.ca.
lpi.org.        77130   IN  NS  dns1.easydns.com.

;; ADDITIONAL SECTION:
dns1.easydns.com.   76140   IN  A   64.68.192.10
dns2.easydns.net.   73684   IN  A   198.41.222.254
dns1.easydns.com.   76140   IN  AAAA    2400:cb00:2049:1::a29f:1835
dns2.easydns.net.   73684   IN  AAAA    2400:cb00:2049:1::c629:defe

;; Query time: 2 msec
;; SERVER: 192.168.1.20#53(192.168.1.20)
;; WHEN: Mon Jun 29 10:18:58 EDT 2020
;; MSG SIZE  rcvd: 389
```

Esercizi Guidati

1. Qual è l'esito del comando seguente?

```
getent group openldap
```

2. Qual è la più grande differenza tra `getent` e gli altri strumenti trattati, `host` e `dig`?

3. Quale opzione di `dig` e `host` è usata per specificare il tipo di record che vuoi recuperare?

4. Quale delle seguenti è una voce corretta di `/etc/hosts`?

`::1 localhost`	
`localhost 127.0.0.1`	

5. Quale opzione di `getent` è usata per specificare quale fonte di dati deve essere usata per eseguire una ricerca?

Esercizi Esplorativi

1. Come opereresti per modificare il seguente `/etc/resolv.conf` con un editor di testo?

```
# Generated by NetworkManager
nameserver 192.168.1.20
```

Le modifiche saranno sovrascritte da NetworkManager.	
NetworkManager aggiornerà la sua configurazione con le tue modifiche.	
Le tue modifiche non influenzeranno il sistema.	
NetworkManager sarà disabilitato.	

2. Che cosa significa la seguente linea in `/etc/nsswitch.conf`?

```
hosts: files [SUCCESS=continue] dns
```

3. Considerando il seguente `/etc/resolv.conf`, perché il sistema non risolve i nomi tramite DNS?

```
search lpi.org
#nameserver fd00:ffff::1:53
#nameserver 10.0.1.53
```

4. Qual è l'esito del comando `dig +noall +answer +question lpi.org`?

5. Come si possono sovrascrivere i valori predefiniti di `dig` senza specificarli sulla linea di comando?

Sommario

Il comando `getent` è un ottimo strumento per vedere i risultati delle chiamate al resolver. Per semplici query DNS, `host` è facile da usare e produce un output diretto. Se hai bisogno di informazioni dettagliate o di mettere a punto una query DNS, `dig` è molto probabilmente la scelta migliore.

Grazie all'abilità di aggiungere plugin di librerie condivise e configurare il comportamento del resolver, Linux ha un eccellente supporto per la risoluzione di nomi e numeri di vari tipi. Il programma `getent` può essere usato per risolvere i nomi usando le librerie resolver. `host` e `dig` possono essere usati per interrogare i server DNS.

Il file `/etc/nsswitch.conf` è usato per configurare il comportamento del resolver. Sei in grado di cambiare le fonti di dati e aggiungere una semplice logica condizionale per i tipi di nome con fonti multiple.

Il DNS è configurato modificando `/etc/resolv.conf`. Molte distribuzioni hanno strumenti che gestiscono questo file per te, quindi assicurati di controllare la documentazione del tuo sistema se le modifiche manuali non persistono.

Il file `/etc/hosts` è usato per risolvere i nomi degli host in IP e viceversa. È tipicamente usato per definire nomi, come `localhost`, che non sono disponibili tramite DNS.

È possibile lasciare commenti nei file di configurazione trattati in questa lezione. Qualsiasi testo a destra di `#` viene ignorato dal sistema.

Risposte agli Esercizi Guidati

1. Qual è l'esito del comando seguente?

   ```
   getent group openldap
   ```

   Leggerà `/etc/nsswitch.conf`, cercherà il gruppo `openldap` dalle fonti elencate e mostrerà informazioni su di esso se viene trovato.

2. Qual è la più grande differenza tra `getent` e gli altri strumenti trattati, `host` e `dig`?

   `getent` cerca i nomi usando le librerie resolver, gli altri si limitano a interrogare il DNS. `getent` può essere usato per risolvere i problemi del tuo `/etc/nsswitch.conf` e la configurazione delle librerie di risoluzione dei nomi che il tuo sistema è configurato per usare. `host` e `dig` sono usati per cercare i record DNS.

3. Quale opzione di `dig` e `host` è usata per specificare il tipo di record che vuoi recuperare?

   Entrambi i programmi usano `-t` per specificare il tipo di record che vuoi cercare.

4. Quale delle seguenti è una voce corretta di `/etc/hosts`?

`::1 localhost`	X
`localhost 127.0.0.1`	

   `::1 localhost` è la linea corretta. La colonna di sinistra è sempre un indirizzo IPv4 o IPv6.

5. Quale opzione di `getent` è usata per specificare quale fonte di dati deve essere usata per eseguire una ricerca?

   L'opzione `-s` è usata per specificare l'origine dei dati. Per esempio:

   ```
   $ getent -s files hosts learning.lpi.org
   192.168.10.25   learning.lpi.org
   $ getent -s dns hosts learning.lpi.org
   208.94.166.198  learning.lpi.org
   ```

Risposte agli Esercizi Esplorativi

1. Come opereresti per modificare il seguente `/etc/resolv.conf` con un editor di testo?

```
# Generated by NetworkManager
nameserver 192.168.1.20
```

Le modifiche saranno sovrascritte da NetworkManager.	X
NetworkManager aggiornerà la sua configurazione con le tue modifiche.	
Le tue modifiche non influenzeranno il sistema.	
NetworkManager sarà disabilitato.	

2. Che cosa significa la seguente linea in `/etc/nsswitch.conf`?

```
hosts: files [SUCCESS=continue] dns
```

   La ricerca dei nomi degli host controllerà prima il file `/etc/hosts` e poi i DNS. Se una voce trovata si trova sia nei file che nel DNS, verrà usata la voce nel DNS.

3. Considerando il seguente `/etc/resolv.conf` perché il sistema non risolve i nomi tramite DNS?

```
search lpi.org
#nameserver fd00:ffff::1:53
#nameserver 10.0.1.53
```

   Entrambi i server DNS sono commentati e non c'è alcun server DNS in esecuzione sull'host locale.

4. Qual è l'esito del comando `dig +noall +answer +question lpi.org`?

   Cerca il record A per `lpi.org` e mostra solo la query e la risposta.

5. Come si possono sovrascrivere i valori predefiniti di `dig` senza specificarli sulla linea di comando?

Crei un file `.digrc` nella tua home directory.



Linux Professional Institute

Argomento 110: Sicurezza



110.1 Eseguire attività di amministrazione della sicurezza

Obiettivi LPI di riferimento

LPIC-1 version 5.0, Exam 102, Objective 110.1

Peso

3

Aree di Conoscenza Chiave

- Controllare un sistema per trovare file con suid/sgid impostato.
- Impostare o modificare le password utente e le informazioni sulla durata della password.
- Essere in grado di utilizzare nmap e netstat per scoprire le porte aperte su un sistema.
- Impostare limiti su accessi utente, processi e utilizzo della memoria.
- Determinare quali utenti hanno effettuato l'accesso al sistema o sono attualmente connessi.
- Configurazione e utilizzo di base del comando sudo.

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

- `find`
- `passwd`
- `fuser`
- `lsof`
- `nmap`
- `chage`
- `netstat`
- `sudo`

- `/etc/sudoers`
- `su`
- `usermod`
- `ulimit`
- `who`, `w`, `last`

110.1 Lezione 1

Certificazione:	LPIC-1
Versione:	5.0
Argomento:	110 Sicurezza
Obiettivo:	110.1 Eseguire attività di amministrazione della sicurezza
Lezione:	1 di 1

Introduzione

La sicurezza è un *must* nell'amministrazione di sistema. Come buon *sysadmin* di Linux devi tenere d'occhio un certo numero di elementi come i permessi speciali sui file, la scadenza delle password degli utenti, porte e socket aperti, limitare l'uso delle risorse di sistema, trattare con gli utenti loggati, e l'escalation dei privilegi attraverso `su` e `sudo`. In questa lezione esamineremo ognuno di questi argomenti.

Controllo dei File con SUID e SGID Attivo

Oltre al tradizionale set di permessi di *read*, *write* e *execute*, i file in un sistema Linux possono anche avere alcuni permessi speciali come i bit *SUID* o *SGID*.

Il bit SUID permetterà al file di essere eseguito con i privilegi del proprietario del file. È rappresentato numericamente da `4000` e simbolicamente da `s` o `S` sul bit di autorizzazione *execute* del proprietario. Un esempio classico di un file eseguibile con il permesso SUID impostato è `passwd`:

```
carol@debian:~$ ls -l /usr/bin/passwd
-rwsr-xr-x 1 root root 63736 jul 27  2018 /usr/bin/passwd
```

La `s` minuscola in `rws` indica la presenza del SUID sul file insieme al permesso *execute*. Una `S` maiuscola invece (`rwS`) significherebbe che il sottostante permesso *execute* non è impostato rendendo il SUID non operativo.

NOTE

Imparerai a conoscere `passwd` nella prossima sezione. L'utilità è usata principalmente da `root` per impostare/cambiare le password degli utenti (per esempio: `passwd carol`). Tuttavia, gli utenti regolari possono usarla anche per cambiare le proprie password. Questo è il motivo per il quale viene fornito con il set SUID.

D'altra parte, il bit SGID può essere impostato sia sui file sia sulle directory. Con i file, il suo comportamento è equivalente a quello di SUID ma i privilegi sono quelli del proprietario del gruppo. Quando è impostato su una directory, tuttavia, permetterà a tutti i file creati in essa di ereditare la proprietà del gruppo della directory. Come SUID, SGID è rappresentato simbolicamente da `s` o `S` sul bit di autorizzazione *execute* del gruppo. Numericamente, è rappresentato da `2000`. Puoi impostare l'SGID su una directory usando `chmod`. Devi aggiungere `2` (SGID) ai permessi tradizionali (`755` nel nostro caso):

```
carol@debian:~$ ls -ld shared_directory
drwxr-xr-x 2 carol carol 4096 may 30 23:55 shared_directory
carol@debian:~$ sudo chmod 2755 shared_directory/
carol@debian:~$ ls -ld shared_directory
drwxr-sr-x 2 carol carol 4096 may 30 23:55 shared_directory
```

Per trovare file con uno o entrambi i set SUID e SGID puoi usare il comando `find` e fare uso dell'opzione `-perm`. Puoi usare sia valori numerici sia simbolici. I valori — a loro volta — possono essere passati da soli o preceduti da un trattino (`-`) o da uno *slash* (`/`). Il significato è il seguente:

`-perm numeric-value` o **`-perm symbolic-value`**
: trova i file che hanno *esclusivamente* il permesso speciale.

`-perm -numeric-value` o **`-perm -symbolic-value`**
: trova i file che hanno il permesso speciale e altri permessi.

`-perm /numeric-value` o **`-perm /symbolic-value`**
: trova i file che hanno uno dei permessi speciali (e altri permessi).

Per esempio, per trovare i file con *solo* SUID impostato nell'attuale directory di lavoro, usa il seguente comando:

```
carol@debian:~$ find . -perm 4000
carol@debian:~$ touch file
carol@debian:~$ chmod 4000 file
carol@debian:~$ find . -perm 4000
./file
```

Nota come, dato che non c'erano file che avessero esclusivamente il SUID, ne abbiamo creato uno per mostrare qualche output. Puoi eseguire lo stesso comando in notazione simbolica:

```
carol@debian:~$ find . -perm u+s
./file
```

Per trovare i file che corrispondono a SUID (indipendentemente da qualsiasi altro permesso) nella directory `/usr/bin/`, puoi usare uno dei seguenti comandi:

```
carol@debian:~$ sudo find /usr/bin -perm -4000
/usr/bin/umount
/usr/bin/newgrp
/usr/bin/gpasswd
/usr/bin/chfn
/usr/bin/mount
/usr/bin/passwd
/usr/bin/chsh
/usr/bin/sudo
/usr/bin/su
carol@debian:~$ sudo find /usr/bin -perm -u+s
/usr/bin/umount
/usr/bin/newgrp
/usr/bin/gpasswd
/usr/bin/chfn
/usr/bin/mount
/usr/bin/passwd
/usr/bin/chsh
/usr/bin/sudo
/usr/bin/su
```

Se stai cercando file nella stessa directory con il bit SGID impostato, puoi eseguire `find /usr/bin/ -perm -2000` o `find /usr/bin/ -perm -g+s`.

Infine, per trovare i file con uno dei due permessi speciali impostati, aggiungi `4` a `2` e usa `/`:

```
carol@debian:~$ sudo find /usr/bin -perm /6000
/usr/bin/dotlock.mailutils
/usr/bin/umount
/usr/bin/newgrp
/usr/bin/wall
/usr/bin/ssh-agent
/usr/bin/chage
/usr/bin/dotlockfile
/usr/bin/gpasswd
/usr/bin/chfn
/usr/bin/mount
/usr/bin/passwd
/usr/bin/chsh
/usr/bin/expiry
/usr/bin/sudo
/usr/bin/bsd-write
/usr/bin/crontab
/usr/bin/su
```

Gestione e Scadenza delle Password

Come già detto, puoi usare l'utilità `passwd` per cambiare la tua password come utente normale. Inoltre, puoi utilizzare l'opzione `-S` o `--status` per ottenere informazioni sullo stato del tuo account:

```
carol@debian:~$ passwd -S
carol P 12/07/2019 0 99999 7 -1
```

Ecco la spiegazione dei sette campi che si ottengono nell'output:

`carol`
: nome di login dell'utente.

`P`
: Indica che l'utente ha una password valida (`P`); altri valori possibili sono `L` per una password bloccata e `NP` per nessuna password.

`12/07/2019`

Data dell'ultimo cambio password.

`0`

Età minima in giorni (il numero minimo di giorni tra i cambi di password). Un valore di `0` significa che la password può essere cambiata in qualsiasi momento.

`99999`

Età massima in giorni (il numero massimo di giorni di validità della password). Un valore di `99999` disabilita la scadenza della password.

`7`

Periodo di avvertimento in giorni (il numero di giorni prima della scadenza della password nei quali un utente verrà avvisato).

`-1`

Periodo di inattività della password in giorni (il numero di giorni inattivi dopo la scadenza della password prima che l'account sia bloccato). Un valore di `-1` rimuoverà l'inattività dell'account.

Oltre a riportare lo stato dell'account, userai il comando `passwd` come root per eseguire alcune operazioni di base di manutenzione degli account. Puoi bloccare e sbloccare gli account, forzare un utente a cambiare la sua password al prossimo login e cancellare la password di un utente rispettivamente con le opzioni `-l`, `-u`, `-e` e `-d`.

Per testare queste opzioni è conveniente introdurre a questo punto il comando `su`. Attraverso `su` si può cambiare utente durante una sessione di login. Quindi, per esempio, usiamo `passwd` come root per bloccare la password di `carol`. Poi passeremo a `carol` e controlleremo lo stato del nostro account per verificare che la password sia stata effettivamente bloccata (`L`) e non possa essere cambiata. Infine, tornando all'utente root, sbloccheremo la password di `carol`:

```
root@debian:~# passwd -l carol
passwd: password expiry information changed.
root@debian:~# su - carol
carol@debian:~$ passwd -S
carol L 05/31/2020 0 99999 7 -1
carol@debian:~$ passwd
Changing password for carol.
Current password:
passwd: Authentication token manipulation error
passwd: password unchanged
```

```
carol@debian:~$ exit
logout
root@debian:~# passwd -u carol
passwd: password expiry information changed.
```

In alternativa, puoi anche bloccare e sbloccare la password di un utente con il comando `usermod`:

Blocca la password dell'utente `carol`

`usermod -L carol` o `usermod --lock carol`.

Sblocca la password dell'utente `carol`

`usermod -U carol` o `usermod --unlock carol`.

NOTE Con le opzioni `-f` o `--inactive`, `usermod` può anche essere usato per impostare il numero di giorni prima che un account con una password scaduta sia disabilitato (per esempio: `usermod -f 3 carol`).

Oltre a `passwd` e `usermod`, il comando più diretto per gestire la durata di password e account è `chage` ("change age"). Come root, puoi passare a `chage` l'opzione `-l` (o `--list`) seguito da un nome utente per avere le sue informazioni sulla scadenza della sua password e dell'account stesso visualizzate sullo schermo; come utente normale, puoi visualizzare le tue informazioni:

```
carol@debian:~$ chage -l carol
Last password change                    : Aug 06, 2019
Password expires                        : never
Password inactive                       : never
Account expires                         : never
Minimum number of days between password change      : 0
Maximum number of days between password change      : 99999
Number of days of warning before password expires   : 7
```

Eseguito senza opzioni e seguito solo da un nome utente, `chage` opererà in modalità interattiva:

```
root@debian:~# chage carol
Changing the aging information for carol
Enter the new value, or press ENTER for the default

    Minimum Password Age [0]:
    Maximum Password Age [99999]:
    Last Password Change (YYYY-MM-DD) [2020-06-01]:
    Password Expiration Warning [7]:
```

```
    Password Inactive [-1]:
    Account Expiration Date (YYYY-MM-DD) [-1]:
```

Le opzioni per modificare le diverse impostazioni di `chage` sono le seguenti:

`-m days username` o **`--mindays days username`**

Specifica il numero minimo di giorni tra i cambi di password (per esempio: `chage -m 5 carol`). Un valore di `0` permetterà all'utente di cambiare la sua password in qualsiasi momento.

`-M days username` o **`--maxdays days username`**

Specifica il numero massimo di giorni di validità della password (per esempio: `chage -M 30 carol`). Per disabilitare la scadenza della password, si usa dare a questa opzione un valore di `99999`.

`-d days username` o **`--lastday days username`**

Specifica il numero di giorni dall'ultima modifica della password (per esempio: `chage -d 10 carol`). Un valore di `0` forzerà l'utente a cambiare la sua password al prossimo login.

`-W days username` o **`--warndays days username`**

Specifica il numero di giorni in cui all'utente verrà ricordato che la sua password è scaduta.

`-I days username` o **`--inactive days username`**

Specifica il numero di giorni inattivi dopo la scadenza della password (per esempio: `chage -I 10 carol`) — lo stesso di `usermod -f` o `usermod --inactive`. Una volta trascorso questo numero di giorni, l'account sarà bloccato. Con un valore di `0`, l'account non sarà bloccato.

`-E date username` o **`--expiredate date username`**

Specifica la data (o il numero di giorni da *epoch* — 1 gennaio 1970) in cui l'account sarà bloccato. È normalmente espressa nel formato `YYYY-MM-DD` (per esempio: `chage -E 2050-12-13 carol`).

NOTE Puoi imparare di più su `passwd`, `usermod` e `chage` — e le loro opzioni — consultando le rispettive pagine di manuale.

Scoprire le Porte Aperte

Quando si tratta di tenere d'occhio le porte aperte presenti su un sistema, quattro potenti utility sono presenti sulla maggior parte dei sistemi Linux: `lsof`, `fuser`, `netstat` e `nmap`.

`lsof` sta per "list open files", che non è una cosa da poco considerando che — per Linux — tutto è

sostanzialmente un file. Infatti, se si digita `lsof` nel terminale, si otterrà un grande elenco di file normali, file di dispositivi, socket, ecc. Tuttavia, per lo scopo di questa lezione, ci concentreremo principalmente sulle porte. Per visualizzare l'elenco di tutti i file di rete "Internet", esegui `lsof` con l'opzione `-i`:

```
root@debian:~# lsof -i
COMMAND  PID     USER   FD   TYPE DEVICE SIZE/OFF NODE NAME
dhclient 357     root    7u  IPv4  13493      0t0  UDP *:bootpc
sshd     389     root    3u  IPv4  13689      0t0  TCP *:ssh (LISTEN)
sshd     389     root    4u  IPv6  13700      0t0  TCP *:ssh (LISTEN)
apache2  399     root    3u  IPv6  13826      0t0  TCP *:http (LISTEN)
apache2  401 www-data    3u  IPv6  13826      0t0  TCP *:http (LISTEN)
apache2  402 www-data    3u  IPv6  13826      0t0  TCP *:http (LISTEN)
sshd     557     root    3u  IPv4  14701      0t0  TCP 192.168.1.7:ssh->192.168.1.4:60510
(ESTABLISHED)
sshd     569    carol    3u  IPv4  14701      0t0  TCP 192.168.1.7:ssh->192.168.1.4:60510
(ESTABLISHED)
```

A parte il servizio `bootpc` - che è usato da DHCP - l'ouput mostra due servizi in ascolto per le connessioni - `ssh` e il server web Apache (`http`) - così come due connessioni SSH stabilite. Puoi specificare un particolare host con la notazione `@ip-address` per controllare le sue connessioni:

```
root@debian:~# lsof -i@192.168.1.7
COMMAND PID  USER   FD   TYPE DEVICE SIZE/OFF NODE NAME
sshd    557  root    3u  IPv4  14701      0t0  TCP 192.168.1.7:ssh->192.168.1.4:60510
(ESTABLISHED)
sshd    569 carol    3u  IPv4  14701      0t0  TCP 192.168.1.7:ssh->192.168.1.4:60510
(ESTABLISHED)
```

NOTE Per visualizzare solo i file di rete IPv4 e IPv6, usate rispettivamente le opzioni `-i4` e `-i6`.

Allo stesso modo, puoi filtrare per porta passando all'opzione `-i` (o `-i@ip-address`) l'argomento `:port`:

```
root@debian:~# lsof -i :22
COMMAND PID  USER   FD   TYPE DEVICE SIZE/OFF NODE NAME
sshd    389  root    3u  IPv4  13689      0t0  TCP *:ssh (LISTEN)
sshd    389  root    4u  IPv6  13700      0t0  TCP *:ssh (LISTEN)
sshd    557  root    3u  IPv4  14701      0t0  TCP 192.168.1.7:ssh->192.168.1.4:60510
(ESTABLISHED)
```

```
sshd    569 carol    3u  IPv4  14701      0t0  TCP 192.168.1.7:ssh->192.168.1.4:60510
(ESTABLISHED)
```

Le porte multiple sono separate da virgole (e gli intervalli sono specificati usando un trattino):

```
root@debian:~# lsof -i@192.168.1.7:22,80
COMMAND PID  USER   FD   TYPE DEVICE SIZE/OFF NODE NAME
sshd    705  root    3u  IPv4  13960      0t0  TCP 192.168.1.7:ssh->192.168.1.4:44766
(ESTABLISHED)
sshd    718 carol    3u  IPv4  13960      0t0  TCP 192.168.1.7:ssh->192.168.1.4:44766
(ESTABLISHED)
```

NOTE La quantità di opzioni disponibili per `lsof` è piuttosto ampia. Per saperne di più, consulta la relativa pagina di manuale.

Il prossimo nella lista dei comandi di rete è `fuser`. Il suo scopo principale è quello di trovare un "utente del file" — il che implica sapere quali processi stiano accedendo a quali file; ti dà anche alcune altre informazioni come il tipo di accesso. Per esempio, per controllare la directory di lavoro corrente, è sufficiente eseguire `fuser .` Tuttavia, per ottenere qualche informazione in più, è conveniente usare l'opzione *verbose* (`-v` o `--verbose`):

```
root@debian:~# fuser .
/root:                580c
root@debian:~# fuser -v .
                     USER        PID ACCESS COMMAND
/root:               root        580 ..c.. bash
```

Scomponiamo l'output:

File

Il file su cui stiamo ottenendo informazioni (`/root`).

Colonna `USER`

Il proprietario del file (`root`).

Colonna `PID`

L'identificatore del processo (`580`).

Colonna `ACCESS`

Tipo di accesso (`..c..`). Uno tra:

c

Directory corrente.

e

Eseguibile in fase di esecuzione.

f

File aperto (omesso nella modalità di visualizzazione predefinita).

F

File aperto per la scrittura (omesso nella modalità di visualizzazione predefinita).

r

directory principale.

m

file mappato in memoria (*mmap*) o libreria condivisa.

.

Segnaposto (omesso nella modalità di visualizzazione predefinita).

Colonna `COMMAND`

Il comando associato al file (`bash`).

Con l'opzione `-n` (o `--namespace`), puoi trovare informazioni sulle porte/socket di rete. Devi anche fornire il protocollo di rete e il numero di porta. Così, per ottenere informazioni sul server web Apache eseguirai il seguente comando:

```
root@debian:~# fuser -vn tcp 80
                     USER        PID ACCESS COMMAND
80/tcp:              root        402 F.... apache2
                     www-data    404 F.... apache2
                     www-data    405 F.... apache2
```

NOTE

`fuser` può anche essere usato per terminare i processi che accedono al file con gli switch `-k` o `--kill` (per esempio: `fuser -k 80/tcp`). Fai riferimento alla pagina del manuale per informazioni più dettagliate.

Passiamo ora a `netstat`. `netstat` è uno strumento di rete molto versatile che viene usato principalmente per visualizzare "statistiche di rete".

Eseguito senza opzioni, `netstat` mostrerà sia le connessioni Internet attive sia i *socket* Unix. A causa della dimensione dell'elenco, potresti voler convogliare il suo output verso `less`:

```
carol@debian:~$ netstat |less
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 192.168.1.7:ssh         192.168.1.4:55444       ESTABLISHED
Active UNIX domain sockets (w/o servers)
Proto RefCnt Flags       Type       State         I-Node   Path
unix  2      [ ]         DGRAM                    10509    /run/systemd/journal/syslog
unix  3      [ ]         DGRAM                    10123    /run/systemd/notify
(...)
```

Per elencare solo le porte e i socket "in ascolto", si useranno le opzioni `-l` o `--listening`. Le opzioni `-t`/`--tcp` e `-u`/`--udp` possono essere aggiunte per filtrare rispettivamente per protocollo TCP e UDP (possono anche essere combinate nello stesso comando). Allo stesso modo, `-e`/`--extend` mostrerà informazioni aggiuntive:

```
carol@debian:~$ netstat -lu
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
udp        0      0 0.0.0.0:bootpc          0.0.0.0:*
carol@debian:~$ netstat -lt
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 0.0.0.0:ssh             0.0.0.0:*               LISTEN
tcp        0      0 localhost:smtp          0.0.0.0:*               LISTEN
tcp6       0      0 [::]:http               [::]:*                  LISTEN
tcp6       0      0 [::]:ssh                [::]:*                  LISTEN
tcp6       0      0 localhost:smtp          [::]:*                  LISTEN
carol@debian:~$ netstat -lute
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       User
Inode
tcp        0      0 0.0.0.0:ssh             0.0.0.0:*               LISTEN      root
13729
tcp        0      0 localhost:smtp          0.0.0.0:*               LISTEN      root
14372
tcp6       0      0 [::]:http               [::]:*                  LISTEN      root
14159
tcp6       0      0 [::]:ssh                [::]:*                  LISTEN      root
13740
```

```
tcp6       0      0 localhost:smtp          [::]:*                  LISTEN      root
14374
udp        0      0 0.0.0.0:bootpc          0.0.0.0:*                           root
13604
```

Se ometti l'opzione `-l`, saranno mostrate *solo* le connessioni stabilite:

```
carol@debian:~$ netstat -ute
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       User
Inode
tcp        0      0 192.168.1.7:ssh         192.168.1.4:39144       ESTABLISHED root
15103
```

Se sei interessato solo alle informazioni numeriche riguardanti porte e host, puoi usare l'opzione `-n` o `--numeric` per stampare solo i numeri di porta e gli indirizzi IP. Nota come `ssh` si trasforma in `22` quando aggiungi `-n` al comando precedente:

```
carol@debian:~$ netstat -uten
Active Internet connections (w/o servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       User
Inode
tcp        0      0 192.168.1.7:22          192.168.1.4:39144       ESTABLISHED 0
15103
```

Come puoi vedere, è possibile creare comandi `netstat` molto utili e produttivi combinando alcune delle sue opzioni. Sfoglia le pagine `man` per saperne di più e trova le combinazioni che meglio si adattano alle tue esigenze.

Infine `nmap` — o `network mapper`. Questo *port scanner* viene eseguito specificando un indirizzo IP o un hostname:

```
root@debian:~# nmap localhost
Starting Nmap 7.70 ( https://nmap.org ) at 2020-06-04 19:29 CEST
Nmap scan report for localhost (127.0.0.1)
Host is up (0.0000040s latency).
Other addresses for localhost (not scanned): ::1
Not shown: 998 closed ports
PORT   STATE SERVICE
22/tcp open  ssh
80/tcp open  http
```

```
Nmap done: 1 IP address (1 host up) scanned in 1.58 seconds
```

A parte un singolo host, `nmap` permette di fare la scansione di:

host multipli
: separandoli con degli spazi (per esempio: `nmap localhost 192.168.1.7`).

intervalli di host
: usando un trattino (per esempio: `nmap 192.168.1.3-20`).

sottoreti
: usando un carattere jolly o la notazione CIDR (per esempio: `nmap 192.168.1.*` o `nmap 192.168.1.0/24`). Puoi escludere particolari host (per esempio: `nmap 192.168.1.0/24 --exclude 192.168.1.7`).

Per scansionare una particolare porta, usa l'opzione `-p` seguita dal numero della porta o dal nome del servizio (`nmap -p 22` e `nmap -p ssh` ti daranno lo stesso risultato):

```
root@debian:~# nmap -p 22 localhost
Starting Nmap 7.70 ( https://nmap.org ) at 2020-06-04 19:54 CEST
Nmap scan report for localhost (127.0.0.1)
Host is up (0.000024s latency).
Other addresses for localhost (not scanned): ::1

PORT   STATE SERVICE
22/tcp open  ssh

Nmap done: 1 IP address (1 host up) scanned in 0.22 seconds
```

Puoi anche scansionare più porte o intervalli di porte usando rispettivamente virgole e trattini:

```
root@debian:~# nmap -p ssh,80 localhost
Starting Nmap 7.70 ( https://nmap.org ) at 2020-06-04 19:58 CEST
Nmap scan report for localhost (127.0.0.1)
Host is up (0.000051s latency).
Other addresses for localhost (not scanned): ::1

PORT   STATE SERVICE
22/tcp open  ssh
80/tcp open  http
```

```
Nmap done: 1 IP address (1 host up) scanned in 0.22 seconds
```

```
root@debian:~# nmap -p 22-80 localhost
Starting Nmap 7.70 ( https://nmap.org ) at 2020-06-04 19:58 CEST
Nmap scan report for localhost (127.0.0.1)
Host is up (0.000011s latency).
Other addresses for localhost (not scanned): ::1
Not shown: 57 closed ports
PORT   STATE SERVICE
22/tcp open  ssh
80/tcp open  http

Nmap done: 1 IP address (1 host up) scanned in 1.47 seconds
```

Altre due importanti e pratiche opzioni `nmap` sono:

`-F`
: Esegui una scansione veloce sulle 100 porte più comuni.

`-v`
: Ottieni un output verboso (`-vv` stamperà un output ancora più verboso).

NOTE `nmap` può eseguire comandi abbastanza complessi facendo uso di tipi di scansione. Tuttavia, questo argomento è al di fuori dallo scopo di questa lezione.

Limiti ai Login degli Utenti, ai Processi e all'Uso della Memoria

Le risorse su un sistema Linux non sono illimitate quindi — come sysadmin — dovresti assicurare un buon equilibrio tra i limiti dell'utente sulle risorse e il corretto funzionamento del sistema operativo. `ulimit` può aiutarti in questo senso.

`ulimit` si occupa dei limiti *soft* e *hard*, specificati rispettivamente dalle opzioni `-S` e `-H`. Eseguito senza opzioni o argomenti, `ulimit` mostrerà il limite *soft* dei file a blocchi dell'utente corrente:

```
carol@debian:~$ ulimit
unlimited
```

Con l'opzione `-a`, `ulimit` mostrerà tutti i limiti soft attuali (lo stesso di `-Sa`); per visualizzare tutti i limiti hard attuali, usa `-Ha`:

```
carol@debian:~$ ulimit -a
core file size          (blocks, -c) 0
data seg size           (kbytes, -d) unlimited
scheduling priority             (-e) 0
(...)
carol@debian:~$ ulimit -Ha
core file size          (blocks, -c) unlimited
data seg size           (kbytes, -d) unlimited
scheduling priority             (-e) 0
(...)
```

Le risorse della shell disponibili sono specificate da opzioni come:

`-b`
: dimensione massima del *buffer* del socket.

`-f`
: dimensione massima dei file scritti dalla shell e dai suoi figli.

`-l`
: dimensione massima che può essere bloccata in memoria.

`-m`
: dimensione massima dell'insieme residente (RSS) — la porzione corrente di memoria tenuta da un processo nella memoria principale (RAM).

`-v`
: massima quantità di memoria virtuale.

`-u`
: numero massimo di processi disponibili per un singolo utente.

Per visualizzare i limiti userai `ulimit` seguito da `-S` (soft) o `-H` (hard) e dall'opzione *resource*; se non viene fornito né `-S` né `-H`, verranno mostrati i limiti soft:

```
carol@debian:~$ ulimit -u
10000
carol@debian:~$ ulimit -Su
10000
carol@debian:~$ ulimit -Hu
```

```
15672
```

Allo stesso modo, per impostare nuovi limiti su una particolare risorsa si specificherà o `-S` o `-H`, seguito dall'opzione della risorsa corrispondente e dal nuovo valore. Questo valore può essere un numero o le parole speciali `soft` (limite soft corrente), `hard` (limite hard corrente) o `unlimited` (nessun limite). Se non viene specificato né `-S` né `-H`, verranno impostati entrambi i limiti. Per esempio, leggiamo prima il valore della dimensione massima corrente per i file scritti dalla shell e dai suoi figli:

```
root@debian:~# ulimit -Sf
unlimited
root@debian:~# ulimit -Hf
unlimited
```

Ora, cambiamo il valore da `unlimited` a `500` blocchi senza specificare né `-S` né `-H`. Nota come entrambi i limiti soft e hard vengono cambiati:

```
root@debian:~# ulimit -f 500
root@debian:~# ulimit -Sf
500
root@debian:~# ulimit -Hf
500
```

Infine, diminuiremo solo il limite soft a `200` blocchi:

```
root@debian:~# ulimit -Sf 200
root@debian:~# ulimit -Sf
200
root@debian:~# ulimit -Hf
500
```

I limiti *hard* possono essere aumentati solo dall'utente root. D'altra parte, gli utenti regolari possono diminuire i limiti hard e aumentare i limiti soft fino al valore dei limiti hard. Per rendere i nuovi valori dei limiti persistenti attraverso i riavvii, è necessario scriverli nel file `/etc/security/limits.conf`. Questo è anche il file usato dall'amministratore per applicare restrizioni a particolari utenti.

NOTE Non c'è una pagina `man` di `ulimit`. È un *builtin* di `bash` quindi devi fare riferimento alla pagina `man` di `bash` per utilizzarlo.

Trattare con gli Utenti in Sessione

Un altro dei tuoi compiti come sysadmin consiste nel tenere traccia degli utenti connessi. Ci sono tre utility che possono aiutarti in questi compiti: `last`, `who` e `w`.

`last` mostra un elenco degli ultimi utenti connessi, a partire dalle informazioni più recenti:

```
root@debian:~# last
carol    pts/0        192.168.1.4      Sat Jun  6 14:25   still logged in
reboot   system boot  4.19.0-9-amd64   Sat Jun  6 14:24   still running
mimi     pts/0        192.168.1.4      Sat Jun  6 12:07 - 14:24  (02:16)
reboot   system boot  4.19.0-9-amd64   Sat Jun  6 12:07 - 14:24  (02:17)
(...)
wtmp begins Sun May 31 14:14:58 2020
```

Considerando l'elenco troncato, otteniamo informazioni sugli ultimi due utenti del sistema. Le prime due linee ci parlano dell'utente `carol`; le due linee successive, dell'utente `mimi`. Le informazioni sono le seguenti:

1. L'utente `carol` sul terminale `pts/0` dall'host `192.168.1.4` ha iniziato la sua sessione `sab giu 6` alle `14:25` ed è ancora `logged in`. Il sistema — che sta utilizzando il kernel `4.19.0-9-amd64` — è stato avviato (`reboot system boot`) `sab giu 6` alle `14:24` ed è `still running`.
2. L'utente `mimi` sul terminale `pts/0` dall'host `192.168.1.4` ha iniziato la sua sessione `sab giu 6` alle `12:07` e si è disconnesso alle `14:24` (la sessione è durata un totale di `(02:16)` ore). Il sistema — che sta utilizzando il kernel `4.19.0-9-amd64` — è stato avviato (`reboot system boot`) `sab giu 6` alle `12:07` ed è stato spento alle `14:24` (è stato in funzione per `(02:17)` ore).

NOTE La linea `wtmp begins Sun May 31 14:14:58 2020` si riferisce a `/var/log/wtmp`, che è il file di log speciale da cui `last` prende le informazioni.

Puoi passare a `last` un nome utente per far visualizzare solo le voci che lo riguardano:

```
root@debian:~# last carol
carol    pts/0        192.168.1.4      Sat Jun  6 14:25   still logged in
carol    pts/0        192.168.1.4      Sat Jun  6 12:07 - 14:24  (02:16)
carol    pts/0        192.168.1.4      Fri Jun  5 00:48 - 01:28  (00:39)
(...)
```

Per quanto riguarda la seconda colonna (terminal), `pts` sta per *Pseudo Terminal Slave* — al contrario di un vero e proprio terminale *TeleTYpewriter* o `tty`; `0` si riferisce al primo (il conteggio

inizia da zero).

NOTE Per controllare i tentativi falliti di login, esegui `lastb` invece di `last`.

Le utilità `who` e `w` si concentrano sugli utenti attualmente connessi e sono abbastanza simili. La prima mostra chi è connesso, mentre la seconda mostra anche informazioni su che cosa stanno facendo.

Quando viene eseguito senza opzioni, `who` mostrerà quattro colonne corrispondenti all'utente connesso, al terminale, alla data e all'ora e al nome dell'host:

```
root@debian:~# who
carol    pts/0        2020-06-06 17:16 (192.168.1.4)
mimi     pts/1        2020-06-06 17:28 (192.168.1.4)
```

`who` accetta una serie di opzioni, tra le quali possiamo evidenziare le seguenti:

`-b`,`--boot`
Visualizza l'ora dell'ultimo avvio del sistema.

`-r`,`--runlevel`
Visualizza il runlevel corrente.

`-H`,`--heading`
Mostra le intestazioni delle colonne.

Rispetto a `who`, `w` dà un output un po' più dettagliato:

```
root@debian:~# w
 17:56:12 up 40 min,  2 users,  load average: 0.04, 0.12, 0.09
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
carol    pts/0    192.168.1.4    17:16    1.00s  0.15s  0.05s sshd: carol [priv]
mimi     pts/1    192.168.1.4    17:28   15:08   0.05s  0.05s -bash
```

La linea superiore fornisce informazioni sull'ora corrente (`17:56:12`), da quanto tempo il sistema è attivo e funzionante (`up 40 min`), il numero di utenti attualmente loggati (`2 users`) e i numeri di carico medio (`load average: 0.04, 0.12, 0.09`). Questi valori si riferiscono al numero di lavori nella coda di esecuzione in media rispettivamente negli ultimi 1, 5 e 15 minuti.

Seguono otto colonne:

`USER`

Login name dello user.

`TTY`

Nome del terminale su cui si trova l'utente.

`FROM`

Host remoto da cui l'utente ha effettuato l'accesso.

`LOGIN@`

Orario d'accesso.

`IDLE`

Tempo di inattività.

`JCPU`

Tempo utilizzato da tutti i processi collegati alla tty (compresi i lavori in *background* attualmente in esecuzione).

`PCPU`

Tempo utilizzato dal processo corrente (quello che appare sotto `WHAT`).

`WHAT`

Linea di comando del processo corrente.

Proprio come con `who`, si possono passare nomi utente `w`:

```
root@debian:~# w mimi
 18:23:15 up  1:07,  2 users,  load average: 0.00, 0.02, 0.05
USER     TTY      FROM             LOGIN@   IDLE   JCPU   PCPU WHAT
mimi     pts/1    192.168.1.4      17:28    9:23   0.06s  0.06s -bash
```

Configurazione e Utilizzo di Base di `sudo`

Come già evidenziato in questa lezione, `su` ti permette di passare a qualsiasi altro utente del sistema purché tu fornisca la password dell'utente di destinazione. Nel caso dell'utente root, avere la sua password distribuita o conosciuta da (molti) altri utenti mette a rischio il sistema ed è una pessima pratica di sicurezza. L'uso base di `su` è `su - target-username`. Quando si passa a root — però — il nome utente di destinazione è opzionale:

```
carol@debian:~$ su - root
Password:
root@debian:~# exit
logout
carol@debian:~$ su -
Password:
root@debian:~#
```

L'uso del trattino (`-`) assicura che l'ambiente dell'utente di destinazione sia caricato. Senza di esso, verrà mantenuto l'ambiente del vecchio utente:

```
carol@debian:~$ su
Password:
root@debian:/home/carol#
```

Con il comando `sudo` si può eseguire un comando come utente root o qualsiasi altro utente. Dal punto di vista della sicurezza, `sudo` è un'opzione di gran lunga migliore di `su` in quanto presenta due vantaggi principali:

1. per eseguire un comando come root, non è necessaria la password dell'utente root, ma solo quella dell'utente che lo invoca in conformità con una politica di sicurezza. La politica di sicurezza predefinita è `sudoers` come specificato in `/etc/sudoers` e `/etc/sudoers.d/*`.
2. `sudo` ti permette di eseguire singoli comandi con privilegi elevati invece di lanciare un'intera nuova *subshell* per root come fa `su`.

L'uso di base di `sudo` è `sudo -u target-username command`. Tuttavia, per eseguire un comando come utente root, l'opzione `-u target-username` non è necessaria:

```
carol@debian:~$ sudo -u mimi whoami
mimi
carol@debian:~$ sudo whoami
root
```

NOTE

`sudoers` userà un riferimento temporale per utente (e per terminale) per la cache delle credenziali, in modo che tu possa usare `sudo` senza password per un periodo predefinito di quindici minuti. Questo valore predefinito può essere modificato aggiungendo l'opzione `timestamp_timeout` come impostazione `Defaults` in `/etc/sudoers` (per esempio: `Defaults timestamp_timeout=1` imposterà il timeout della cache delle credenziali a un minuto).

Il File `/etc/sudoers`

Il file di configurazione principale di `sudo` è `/etc/sudoers` (c'è anche la directory `/etc/sudoers.d`), dove vengono determinati i privilegi di `sudo` degli utenti. In altre parole, è qui che si specifica chi può eseguire quali comandi, come quale utente su quali macchine così come anche altre impostazioni. La sintassi usata è la seguente:

```
carol@debian:~$ sudo less /etc/sudoers
(...)
# User privilege specification
root    ALL=(ALL:ALL) ALL

# Allow members of group sudo to execute any command
%sudo   ALL=(ALL:ALL) ALL
(...)
```

La specifica dei privilegi per l'utente root è `ALL=(ALL:ALL) ALL`. Questo si traduce come: l'utente root (`root`) può accedere da tutti gli host (`ALL`), come tutti gli utenti e tutti i gruppi (`(ALL:ALL)`), ed eseguire tutti i comandi (`ALL`). Lo stesso vale per i membri del gruppo `sudo`: nota come i nomi dei gruppi siano identificati da un segno di percentuale precedente (%).

Così, per avere l'utente `carol` in grado di controllare lo stato di `apache2` da qualsiasi host come qualsiasi utente o gruppo, si aggiungerà la seguente linea nel file `sudoers`:

```
carol   ALL=(ALL:ALL) /usr/bin/systemctl status apache2
```

Potresti voler risparmiare a `carol` l'inconveniente di dover fornire la sua password per eseguire il comando `systemctl status apache2`. Per questo, modificherai la linea in questo modo:

```
carol   ALL=(ALL:ALL) NOPASSWD: /usr/bin/systemctl status apache2
```

Diciamo che ora vuoi limitare i tuoi host su 192.168.1.7 e permettere a `carol` di eseguire `systemctl status apache2` come utente `mimi`. Dovresti modificare la linea nel seguente modo:

```
carol   192.168.1.7=(mimi) /usr/bin/systemctl status apache2
```

Ora puoi controllare lo stato del server web Apache come utente `mimi`:

```
carol@debian:~$ sudo -u mimi systemctl status apache2
```

```
● apache2.service - The Apache HTTP Server
   Loaded: loaded (/lib/systemd/system/apache2.service; enabled; vendor preset: enabled)
   Active: active (running) since Tue 2020-06-09 13:12:19 CEST; 29min ago
(...)
```

Se `carol` dovesse essere promossa a sysadmin e tu volessi darle tutti i privilegi, l'approccio più semplice sarebbe quello di includerla nel gruppo speciale `sudo` con `usermod` e l'opzione `-G` (potresti anche usare l'opzione `-a`, che assicura che l'utente non sia rimosso da qualsiasi altro gruppo a cui potrebbe appartenere):

```
root@debian:~# sudo useradd -aG sudo carol
```

NOTE Nella famiglia delle distribuzioni Red Hat il gruppo `wheel` è la controparte dello speciale gruppo amministrativo `sudo` dei sistemi Debian.

Invece di modificare `/etc/sudoers` direttamente, dovresti semplicemente usare il comando `visudo` come root: aprirà `/etc/sudoers` usando il tuo editor di testo predefinito. Per cambiare l'editor di testo predefinito, puoi aggiungere l'opzione `editor` come impostazione `Defaults` in `/etc/sudoers`. Per esempio, per cambiare l'editor in `nano`, aggiungerai la seguente linea:

```
Defaults    editor=/usr/bin/nano
```

NOTE In alternativa, puoi specificare un editor di testo tramite la variabile d'ambiente `EDITOR` quando usi `visudo` (per esempio: `EDITOR=/usr/bin/nano visudo`)

Oltre agli utenti e ai gruppi, puoi anche fare uso di alias in `/etc/sudoers`. Ci sono tre categorie principali di alias che puoi definire: *host alias* (`Host_Alias`), *user alias* (`User_Alias`) e *command alias* (`Cmnd_Alias`). Ecco un esempio:

```
# Host alias specification

Host_Alias SERVERS = 192.168.1.7, server1, server2

# User alias specification

User_Alias REGULAR_USERS = john, mary, alex

User_Alias PRIVILEGED_USERS = mimi

User_Alias ADMINS = carol, %sudo, PRIVILEGED_USERS, !REGULAR_USERS
```

```
# Cmnd alias specification

Cmnd_Alias SERVICES = /usr/bin/systemctl *

# User privilege specification
root    ALL=(ALL:ALL) ALL
ADMINS  SERVERS=SERVICES

# Allow members of group sudo to execute any command
%sudo   ALL=(ALL:ALL) ALL
```

Considerando questo file di esempio `sudoers`, spieghiamo i tre tipi di alias un po' più nel dettaglio:

Host aliases

Includono una lista separata da virgole di nomi di host, indirizzi IP, così come reti e *netgroup* (preceduti da `+`). Possono essere specificate anche le *netmask*. L'alias di host `SERVERS` include un indirizzo IP e due nomi di host:

```
Host_Alias SERVERS = 192.168.1.7, server1, server2
```

User aliases

Includono una lista separata da virgole di utenti specificati come nomi utente, gruppi (preceduti da `%`) e *netgroup* (preceduti da `+`). Puoi escludere particolari utenti con `!`. L'alias utente `ADMINS`, per esempio, include l'utente `carol`, i membri del gruppo `sudo` e quei membri dell'alias utente `PRIVILEGE_USERS` che non appartengono all'alias utente `REGULAR_USERS`:

```
User_Alias ADMINS = carol, %sudo, PRIVILEGED_USERS, !REGULAR_USERS
```

Command aliases

Includono una lista di comandi e directory separata da virgole. Se viene specificata una directory, qualsiasi file in quella directory sarà incluso, ma le sottodirectory saranno ignorate. L'alias di comando `SERVICES` include un singolo comando con tutti i suoi sottocomandi, come specificato dall'asterisco (`*`):

```
Cmnd_Alias SERVICES = /usr/bin/systemctl *
```

Come risultato delle specifiche degli alias, la linea `ADMINS SERVERS=SERVICES` sotto la sezione

`User privilege specification` si traduce come: tutti gli utenti appartenenti a `ADMINS` possono usare `sudo` per eseguire qualsiasi comando in `SERVICES` su qualsiasi server in `SERVERS`.

NOTE

C'è un quarto tipo di alias che puoi includere in `/etc/sudoers`: *runas alias* (`Runas_Alias`). Sono molto simili agli alias utente, ma ti permettono di specificare gli utenti per il loro *user ID* (UID). Questa caratteristica potrebbe essere conveniente in alcuni scenari.

Esercizi Guidati

1. Completa la seguente tabella relativa alle autorizzazioni speciali:

Permesso speciale	**Rappresentazione numerica**	**Rappresentazione simbolica**	**Trova i file con *solo* quel permesso impostato**
`SUID`			
`SGID`			

2. Visualizzare i file con *solo* il bit `SUID` o `SGID` impostato non è normalmente molto pratico. Esegui i seguenti compiti per dimostrare che le tue ricerche possono essere più produttive:
   - Trova tutti i file con `SUID` (e altri permessi) impostato in `/usr/bin`:
   - Trova tutti i file con `SGID` (e altri permessi) impostato in `/usr/bin`:
   - Trova tutti i file con il `SUID` o il `SGID` impostato in `/usr/bin`:

3. `chage` permette di cambiare le informazioni sulla scadenza della password di un utente. Come root, completa la seguente tabella fornendo i comandi corretti sull'utente `mary`:

Azione	**Comandi** `chage`
Rendi la password valida per 365 giorni.	
Fai in modo che l'utente cambi la password al prossimo login.	
Imposta a 1 il numero minimo di giorni tra i cambi di password.	
Disabilita la scadenza della password.	
Abilita l'utente a cambiare la sua password in qualsiasi momento.	
Imposta il periodo di avviso a 7 giorni e la data di scadenza dell'account al 20 agosto 2050.	

Azione	**Comandi** `chage`
Stampa le informazioni sulla scadenza attuale della password dell'utente.	

4. Completa la seguente tabella con l'utilità di rete appropriata:

Azione	**Comando(i)**
Mostra i file di rete per l'host `192.168.1.55` sulla porta `22` usando `lsof`.	
Mostra i processi che accedono alla porta predefinita del server web Apache sulla tua macchina con `fuser`.	
Elenca tutti i socket *udp* in ascolto sulla tua macchina usando `netstat`.	
Scansiona le porte da `80` a `443` sull'host `192.168.1.55` usando `nmap`.	

5. Esegui i seguenti compiti relativi a *resident set size* (*RSS*) e `ulimit` come un utente normale:
   - Visualizza i limiti *soft* sul *maximum RSS*:
   - Visualizza limiti *hard* sul *maximum RSS*:
   - Imposta i limiti *soft* del *maximum RSS* a 5.000 kilobyte:
   - Imposta i limiti *hard* del *maximum RSS* a 10.000 kilobyte:
   - Infine, prova ad aumentare il limite *hard* del *maximum RSS* fino a 15.000 kilobyte. Puoi farlo? Perché?

6. Considera la seguente linea di output del comando `last` e rispondi alle domande:

```
carol    pts/0        192.168.1.4      Sun May 31 14:16 - 14:22  (00:06)
```

- `carol` si è collegata da un host remoto? Perché?

- Quanto è durata la sessione di `carol`?

- Il collegamento di `carol` è avvenuto attraverso un vero terminale classico basato su testo? Perché?

7. Considera il seguente estratto da `/etc/sudoers` e rispondi alla domanda che segue.

```
# Host alias specification

Host_Alias SERVERS = 192.168.1.7, server1, server2

# User alias specification

User_Alias REGULAR_USERS = john, mary, alex

User_Alias PRIVILEGED_USERS = mimi

User_Alias ADMINS = carol, %sudo, PRIVILEGED_USERS, !REGULAR_USERS

# Cmnd alias specification

Cmnd_Alias WEB_SERVER_STATUS = /usr/bin/systemctl status apache2

# User privilege specification
root    ALL=(ALL:ALL) ALL
ADMINS  SERVERS=WEB_SERVER_STATUS

# Allow members of group sudo to execute any command
%sudo   ALL=(ALL:ALL) ALL
```

Può `alex` controllare lo stato del server web Apache su qualsiasi host? Perché?

Esercizi Esplorativi

1. Oltre a `SUID` e `SGID`, c'è un terzo permesso speciale: lo *sticky bit*. Attualmente è usato per lo più su directory come `/tmp` per impedire agli utenti regolari di cancellare o spostare file diversi dai propri. Esegui le seguenti operazioni:
   - Imposta lo *sticky bit* su `~/temporal`:
   - Trova le directory con lo *sticky bit* (e qualsiasi altro permesso) impostato sulla tua home directory:
   - Rimuovi lo *sticky bit* su `~/temporal`:
2. Quando la password di un utente è bloccata tramite `passwd -l username` o `usermod -L username`, come puoi capirlo guardando in `/etc/shadow`?
3. Qual è la controparte del comando `usermod` di `chage -E date username` o `chage --expiredate date username`?
4. Fornisci due diversi comandi `nmap` per analizzare tutte le 65535 porte su localhost:

Sommario

In questa lezione hai imparato come eseguire una serie di compiti di amministrazione della sicurezza. Sono stati trattati i seguenti argomenti:

- Trovare i file con i permessi speciali `SUID` e `SGID`.
- Impostare e cambiare le password degli utenti e gestire le informazioni sulla scadenza delle password.
- Usare un certo numero di utility di rete per scoprire le porte aperte su host/reti.
- Impostare dei limiti delle risorse di sistema.
- Controllare gli utenti che hanno effettuato l'accesso al sistema o che sono attualmente connessi.
- Utilizzare e configurare `sudo` (attraverso il file `/etc/sudoers`).

Comandi e file discussi in questa lezione:

`find`
: cerca i file in una gerarchia di directory.

`passwd`
: Cambia la password dell'utente.

`chmod`
: Cambia i bit della modalità del file.

`chage`
: Cambia le informazioni sulla scadenza della password dell'utente.

`lsof`
: Elenca i file aperti.

`fuser`
: Identifica i processi che utilizzano file o socket.

`netstat`
: Visualizza le connessioni di rete.

`nmap`
: Esplorazione rete e scansione porte.

`ulimit`

Ottiene e imposta i limiti utente.

`/etc/security/limits.conf`

Configurazione delle restrizioni agli utenti.

`last`

Visualizzazione elenco degli ultimi utenti loggati.

`lastb`

Visualizzazione elenco di tentativi di login falliti.

`/var/log/wtmp`

Database dei login degli utenti.

`who`

Mostra chi è loggato.

`w`

Mostra chi è collegato e cosa sta facendo.

`su`

Cambia utente o diventa superutente.

`sudo`

Esegui un comando come un altro utente (incluso il superutente).

Risposte agli Esercizi Guidati

1. Completa la seguente tabella relativa alle autorizzazioni speciali:

Permesso speciale	**Rappresentazione numerica**	**Rappresentazione simbolica**	**Trova i file con *solo* quel permesso impostato**
`SUID`	`4000`	`s`,`S`	`find -perm 4000`, `find -perm u+s`
`SGID`	`2000`	`s`,`S`	`find -perm 2000`, `find -perm g+s`

2. Visualizzare i file con *solo* il bit `SUID` o `SGID` impostato non è normalmente molto pratico. Esegui i seguenti compiti per dimostrare che le tue ricerche possono essere più produttive:
   - Trova tutti i file con `SUID` (e altri permessi) impostato in `/usr/bin`:

     `find /usr/bin -perm -4000` or `find /usr/bin -perm -u+s`
   - Trova tutti i file con `SGID` (e altri permessi) impostato in `/usr/bin`:

     `find /usr/bin -perm -2000` or `find /usr/bin -perm -g+s`
   - Trova tutti i file con il `SUID` o il `SGID` impostato in `/usr/bin`:

     `find /usr/bin -perm /6000`

3. `chage` permette di cambiare le informazioni sulla scadenza della password di un utente. Come root, completa la seguente tabella fornendo i comandi corretti sull'utente `mary`:

Azione	**Comandi** `chage`
Rendi la password valida per 365 giorni.	`chage -M 365 mary`, `chage --maxdays 365 mary`
Fai in modo che l'utente cambi la password al prossimo login.	`chage -d 0 mary`, `chage --lastday 0 mary`
Imposta a 1 il numero minimo di giorni tra i cambi di password.	`chage -m 1 mary`, `chage --mindays 1 mary`
Disabilita la scadenza della password.	`chage -M 99999 mary`, `chage --maxdays 99999 mary`

Azione	**Comandi** `chage`
Abilita l'utente a cambiare la sua password in qualsiasi momento.	`chage -m 0 mary`, `chage --mindays 0 mary`
Imposta il periodo di avviso a 7 giorni e la data di scadenza dell'account al 20 agosto 2050.	`chage -W 7 -E 2050-08-20 mary`, `chage --warndays 7 --expiredate 2050-08-20 mary`
Stampa le informazioni sulla scadenza attuale della password dell'utente.	`chage -l mary`, `chage --list mary`

4. Completa la seguente tabella con l'utilità di rete appropriata:

Azione	**Comando(i)**
Mostra i file di rete per l'host `192.168.1.55` sulla porta `22` usando `lsof`.	`lsof -i@192.168.1.55:22`
Mostra i processi che accedono alla porta predefinita del server web Apache sulla tua macchina con `fuser`.	`fuser -vn tcp 80`, `fuser --verbose --namespace tcp 80`
Elenca tutti i socket *udp* in ascolto sulla tua macchina usando `netstat`.	`netstat -lu`,`netstat --listening --udp`
Scansiona le porte da `80` a `443` sull'host `192.168.1.55` usando `nmap`.	`nmap -p 80-443 192.168.1.55`

5. Esegui i seguenti compiti relativi a *resident set size* (*RSS*) e `ulimit` come un utente normale:
   - Visualizza i limiti *soft* sul *maximum RSS*:

     `ulimit -m`, `ulimit -Sm`
   - Visualizza limiti *hard* sul *maximum RSS*:

     `ulimit -Hm`
   - Imposta i limiti *soft* del *maximum RSS* a 5.000 kilobyte:

     `ulimit -Sm 5000`
   - Imposta i limiti *hard* del *maximum RSS* a 10.000 kilobyte:

     `ulimit -Hm 10000`

- Infine, prova ad aumentare il limite *hard* del *maximum RSS* fino a 15.000 kilobyte. Puoi farlo? Perché?

  No. Una volta impostato, gli utenti regolari non possono aumentare i limiti *hard*.

6. Considera la seguente linea di output del comando `last` e rispondi alle domande:

```
carol    pts/0        192.168.1.4      Sun May 31 14:16 - 14:22  (00:06)
```

   - `carol` si è collegata da un host remoto? Perché?

     Sì, l'indirizzo IP dell'host remoto è nella terza colonna.

   - Quanto è durata la sessione di `carol`?

     Sei minuti (come mostrato nell'ultima colonna).

   - Il collegamento di `carol` è avvenuto attraverso un vero terminale classico basato su testo? Perché?

     No, `pts/0` nella seconda colonna indica che la connessione è stata fatta attraverso un emulatore di terminale grafico (conosciuto come: *Pseudo Terminal Slave*).

7. Considera il seguente estratto da `/etc/sudoers` e rispondi alla domanda che segue.

```
# Host alias specification

Host_Alias SERVERS = 192.168.1.7, server1, server2

# User alias specification

User_Alias REGULAR_USERS = john, mary, alex

User_Alias PRIVILEGED_USERS = mimi

User_Alias ADMINS = carol, %sudo, PRIVILEGED_USERS, !REGULAR_USERS

# Cmnd alias specification

Cmnd_Alias WEB_SERVER_STATUS = /usr/bin/systemctl status apache2

# User privilege specification
root    ALL=(ALL:ALL) ALL
```

```
ADMINS  SERVERS=WEB_SERVER_STATUS

# Allow members of group sudo to execute any command
%sudo   ALL=(ALL:ALL) ALL
```

Può `alex` controllare lo stato del server web Apache su qualsiasi host? Perché?

No, perché è un membro di `REGULAR_USERS` e quel gruppo di utenti è escluso da `ADMINS`; gli unici utenti (a parte `carol`, membri del gruppo `sudo` e `root`) che possono eseguire `systemctl status apache2` sui `SERVERS`.

Risposte agli Esercizi Esplorativi

1. Oltre a `SUID` e `SGID`, c'è un terzo permesso speciale: lo *sticky bit*. Attualmente è usato per lo più su directory come `/tmp` per impedire agli utenti regolari di cancellare o spostare file diversi dai propri. Esegui le seguenti operazioni:
   - Imposta lo *sticky bit* su `~/temporal`:

     `chmod +t temporal`, `chmod 1755 temporal`

   - Trova le directory con lo *sticky bit* (e qualsiasi altro permesso) impostato sulla tua home directory:

     `find ~ -perm -1000`, `find ~ -perm /1000`

   - Rimuovi lo *sticky bit* su `~/temporal`:

     `chmod -t temporal`, `chmod 0755 temporal`

2. Quando la password di un utente è bloccata tramite `passwd -l username` o `usermod -L username`, come puoi capirlo guardando in `/etc/shadow`?

   Un punto esclamativo apparirà nel secondo campo, subito dopo il nome di login dell'utente interessato (per esempio: `mary:!$6$gOg9xJgv...`).

3. Qual è la controparte del comando `usermod` di `chage -E date username` o `chage --expiredate date username`?

   `usermod -e date username`, `usermod --expiredate date username`

4. Fornisci due diversi comandi `nmap` per analizzare tutte le 65535 porte su localhost:

   `nmap -p 1-65535 localhost` e `nmap -p- localhost`



110.2 Configurare la sicurezza dell'host

Obiettivi LPI di riferimento

LPIC-1 version 5.0, Exam 102, Objective 110.2

Peso

3

Aree di Conoscenza Chiave

- Conoscenza delle shadow password e del loro funzionamento.
- Disattivare i servizi di rete non in uso.
- Comprendere il ruolo dei wrapper TCP.

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

- `/etc/nologin`
- `/etc/passwd`
- `/etc/shadow`
- `/etc/xinetd.d/`
- `/etc/xinetd.conf`
- `systemd.socket`
- `/etc/inittab`
- `/etc/init.d/`
- `/etc/hosts.allow`
- `/etc/hosts.deny`



110.2 Lezione 1

Certificazione:	LPIC-1
Versione:	5.0
Argomento:	110 Sicurezza
Obiettivo:	110.2 Configurare la sicurezza dell'host
Lezione:	1 di 1

Introduzione

Questo capitolo spiega quattro modi fondamentali per migliorare la sicurezza degli host:

1. Alcuni comandi di base e impostazioni di configurazione per migliorare la sicurezza dell'autenticazione con le *shadow password* .
2. Come usare i *superdaemons* per metterli in ascolto sulle connessioni di rete in entrata.
3. Controllare i servizi di rete rispetto a demoni non necessari.
4. Utilizzo di *TCP wrapper* come una sorta di semplice firewall.

Migliorare la Sicurezza dell'Autenticazione con le Shadow Password

I componenti di base dei dati dell'account di un utente sono memorizzati nel file `/etc/passwd`. Questo file contiene sette campi: nome di login, userid, groupid, password, commento (*GECOS*), posizione della home directory e infine la shell di default. Un modo semplice per ricordare l'ordine di questi campi è pensare al processo di login di un utente: prima si inserisce un nome di

login, in secondo luogo il sistema lo mappa in un userid (uid) e in terzo luogo in un groupid (gid). Il quarto passo chiede una password, il quinto cerca il commento, il sesto entra nella home directory dell'utente e il settimo passo imposta la shell di default.

Nei sistemi moderni la password non è più memorizzata nel file `/etc/passwd`. Ciò che rimane al suo interno è una `x` minuscola nel campo password. Il file `/etc/passwd` deve essere leggibile da tutti gli utenti. La `x` indica che la password criptata (*hash*) è invece memorizzata nel file `/etc/shadow`. Questo file non deve essere leggibile da tutti gli utenti.

Le impostazioni delle password sono configurate con i comandi `passwd` e `chage`. Entrambi i comandi cambieranno la voce per l'utente `emma` nel file `/etc/shadow`. Come superutente puoi impostare la password per l'utente `emma` con il seguente comando:

```
$ sudo passwd emma
New password:
Retype new password:
passwd: password updated successfully
```

Ti verrà quindi richiesto *due volte* di confermare la nuova password.

Per elencare la data di scadenza della password e altre impostazioni di scadenza della password per l'utente `emma` usa:

```
$ sudo chage -l emma
Last password change                    : Apr 27, 2020
Password expires                        : never
Password inactive                       : never
Account expires                         : never
Minimum number of days between password change      : 0
Maximum number of days between password change      : 99999
Number of days of warning before password expires   : 7
```

Per impedire all'utente `emma` di accedere al sistema il superutente può impostare una data di scadenza della password che precede la data corrente. Per esempio, se la data di oggi fosse 2020-03-27, si potrebbe far scadere la password dell'utente usando una data antecedente:

```
$ sudo chage -E 2020-03-26 emma
```

In alternativa, il superutente può usare:

```
$ sudo passwd -l emma
```

per bloccare temporaneamente l'account tramite l'opzione `-l` per `passwd`. Per testare gli effetti di queste modifiche, prova a fare il login come `emma`:

```
$ sudo login emma
Password:
Your account has expired; please contact your system administrator

Authentication failure
```

Per impedire a tutti gli utenti tranne l'utente root di accedere al sistema temporaneamente, il superutente può creare un file chiamato `/etc/nologin`. Questo file può contenere un messaggio agli utenti che notifica loro il motivo per cui non possono accedere (per esempio con notifiche di manutenzione del sistema). Per dettagli vedere `man 5 nologin`. Nota che c'è anche un comando `nologin` che può essere usato per prevenire un login quando è impostato come shell di default per un utente. Per esempio:

```
$ sudo usermod -s /sbin/nologin emma
```

Vedi `man 8 nologin` per maggiori dettagli.

Come Mettere in Ascolto un Superdaemon sulle Connessioni di Rete in Entrata.

I servizi di rete come i server web, i server di posta elettronica e i server di stampa di solito funzionano come un servizio autonomo in ascolto su una porta di rete dedicata. Tutti questi servizi *standalone* sono in esecuzione fianco a fianco. Su un sistema classico basato su *Sys-V-init* ognuno di questi servizi può essere controllato dal comando `service`. Sugli attuali sistemi basati su systemd si dovrebbe usare `systemctl` per gestire il servizio.

In passato la disponibilità di risorse informatiche era molto minore. Eseguire molti servizi in modalità *standalone* non era una buona opzione. Invece veniva usato un cosiddetto "Superdemone" per ascoltare le connessioni di rete in arrivo e avviare il servizio appropriato su richiesta. Ben noti superdemoni sono `inetd` e `xinetd`. Sui sistemi attuali basati su systemd l'unità `systemd.socket` può essere usata in modo simile. In questa sezione useremo `xinetd` per intercettare le connessioni al demone `sshd` e avviare questo demone su richiesta per dimostrare come è stato usato il superdemone.

Prima di configurare il servizio `xinetd` è necessaria qualche attività preparatoria. Non importa se si usa un sistema basato su Debian o Red Hat: sebbene la lezione sia basata su Debian/GNU Linux 9.9, dovrebbero poter funzionare su qualsiasi sistema Linux attuale con systemd, senza alcun cambiamento significativo. Per prima cosa assicurati che i pacchetti `openssh-server` e `xinetd` siano installati. Quindi verifica che il servizio SSH funzioni con:

```
$ systemctl status sshd
● ssh.service - OpenBSD Secure Shell server
   Loaded: loaded (/lib/systemd/system/ssh.service; enabled; vendor preset: enabled)
   Active: active (running) since Mon 2020-04-27 09:33:48 EDT; 3h 11min ago
     Docs: man:sshd(8)
           man:sshd_config(5)
  Process: 430 ExecStartPre=/usr/sbin/sshd -t (code=exited, status=0/SUCCESS)
 Main PID: 460 (sshd)
    Tasks: 1 (limit: 1119)
   Memory: 5.3M
   CGroup: /system.slice/ssh.service
           └─460 /usr/sbin/sshd -D
```

Controllate anche che il servizio SSH sia in ascolto sulla sua porta di rete standard 22:

```
# lsof -i :22
COMMAND  PID USER   FD   TYPE   DEVICE SIZE/OFF NODE NAME
sshd    1194 root    3u  IPv4 16053268      0t0  TCP *:ssh (LISTEN)
sshd    1194 root    4u  IPv6 16053270      0t0  TCP *:ssh (LISTEN)
```

Infine fermate il servizio SSH con:

```
$ sudo systemctl stop sshd.service
```

Nel caso in cui tu voglia rendere questo cambiamento permanente, usa `systemctl disable sshd.service`.

Ora puoi creare il file di configurazione di xinetd `/etc/xinetd.d/ssh` con alcune impostazioni di base:

```
service ssh
{
    disable     = no
    socket_type = stream
```

```
    protocol    = tcp
    wait        = no
    user        = root
    server      = /usr/sbin/sshd
    server_args     = -i
    flags       = IPv4
    interface   = 192.168.178.1
}
```

Riavviate il servizio xinetd con:

```
$ sudo systemctl restart xinetd.service
```

Controlla quale servizio sia ora in ascolto ora per le connessioni SSH in entrata.

```
$ sudo lsof -i :22
COMMAND   PID USER   FD   TYPE  DEVICE SIZE/OFF NODE NAME
xinetd  24098 root    5u  IPv4 7345141      0t0  TCP 192.168.178.1:ssh (LISTEN)
```

Possiamo vedere che il servizio xinetd ha preso il controllo dell'accesso alla porta 22.

Qui di seguito ci sono alcuni dettagli in più sulla configurazione di `xinetd`. Il file di configurazione principale è `/etc/xinetd.conf`:

```
# Simple configuration file for xinetd
#
# Some defaults, and include /etc/xinetd.d/

defaults
{

# Please note that you need a log_type line to be able to use log_on_success
# and log_on_failure. The default is the following :
# log_type = SYSLOG daemon info

}

includedir /etc/xinetd.d
```

Oltre alle impostazioni predefinite, c'è solo una direttiva per impostare una *include directory*. In

questa directory puoi impostare un singolo file di configurazione per ogni servizio che vuoi far gestire da `xinetd`. Lo abbiamo fatto sopra per il servizio SSH e abbiamo chiamato il file `/etc/xinetd.d/ssh`. I nomi dei file di configurazione possono essere scelti arbitrariamente, tranne i nomi di file che contengono un punto (`.`) o che terminano con una tilde (`~`). Ma è pratica diffusa nominare il file con il nome del servizio che si vuole configurare.

Alcuni file di configurazione nella directory `/etc/xinet.d/` sono già forniti dalla distribuzione:

```
$ ls -l /etc/xinetd.d
total 52
-rw-r--r-- 1 root root 640 Feb  5  2018 chargen
-rw-r--r-- 1 root root 313 Feb  5  2018 chargen-udp
-rw-r--r-- 1 root root 502 Apr 11 10:18 daytime
-rw-r--r-- 1 root root 313 Feb  5  2018 daytime-udp
-rw-r--r-- 1 root root 391 Feb  5  2018 discard
-rw-r--r-- 1 root root 312 Feb  5  2018 discard-udp
-rw-r--r-- 1 root root 422 Feb  5  2018 echo
-rw-r--r-- 1 root root 304 Feb  5  2018 echo-udp
-rw-r--r-- 1 root root 312 Feb  5  2018 servers
-rw-r--r-- 1 root root 314 Feb  5  2018 services
-rw-r--r-- 1 root root 569 Feb  5  2018 time
-rw-r--r-- 1 root root 313 Feb  5  2018 time-udp
```

Questi file possono essere usati come modelli nel raro caso in cui si debbano usare alcuni servizi *legacy* come `daytime`, una primissima implementazione di un time server. Tutti questi file *template* contengono la direttiva `disable = yes`.

Qui ci sono alcuni dettagli in più sulle direttive usate nel file di esempio `/etc/xinetd.d/ssh` per ssh.

```
service ssh
{
    disable     = no
    socket_type = stream
    protocol    = tcp
    wait        = no
    user        = root
    server      = /usr/sbin/sshd
    server_args     = -i
    flags       = IPv4
    interface   = 192.168.178.1
```

```
}
```

`service`

Elenca il servizio che xinetd deve controllare. Si può usare un numero di porta, come 22, o il nome mappato al numero di porta in `/etc/services`, per esempio `ssh`.

`{`

Le impostazioni dettagliate iniziano con una parentesi graffa di apertura.

`disable`

Per attivare queste impostazioni, impostalo su `no`. Se vuoi disabilitare temporaneamente l'impostazione, puoi impostarla su `yes`.

`socket_type`

Puoi scegliere `stream` per le porte TCP o `dgram` per quelle UDP e altro.

`protocol`

Scegliere TCP o UDP.

`wait`

Per le connessioni TCP questo è impostato di solito su `no`.

`user`

Il servizio avviato sarà di proprietà di questo utente.

`server`

Percorso completo al servizio che dovrebbe essere avviato da `xinetd`.

`server_args`

Qui si possono aggiungere opzioni per il servizio. Se avviato da un *super-server* molti servizi richiedono un'opzione speciale. Per SSH questa opzione è `-i`.

`flags`

Si può scegliere tra IPv4, IPv6 e altri.

`interface`

L'interfaccia di rete che `xinetd` deve controllare. Nota: si può anche scegliere la direttiva `bind`, che è solo un sinonimo di `interface`.

```
}
```

Le impostazioni dettagliate terminano con una parentesi graffa di chiusura.

I successori dei servizi avviati dal super-server `xinetd` sono le unità *socket* di systemd. Impostare una *socket unit* di systemd è molto semplice e diretto, perché c'è una socket unit di systemd predefinita per SSH già disponibile. Assicurarsi che i servizi `xinetd` e SSH non siano in esecuzione.

Ora devi solo avviare l'unità socket SSH:

```
$ sudo systemctl start ssh.socket
```

Per controllare quale servizio è ora in ascolto sulla porta 22 usiamo di nuovo `lsof`. Nota qui che l'opzione `-P` è stata usata per mostrare il numero della porta invece del nome del servizio nell'output:

```
$ sudo lsof -i :22 -P
COMMAND PID USER   FD   TYPE   DEVICE SIZE/OFF NODE NAME
systemd   1 root   57u  IPv6 14730112      0t0  TCP *:22 (LISTEN)
```

Per completare il test, prova ad accedere al server con un client SSH di tua scelta.

TIP Nel caso in cui `systemctl start ssh.socket` non funzioni con la tua distribuzione, prova `systemctl start sshd.socket`.

Controllare i Servizi per i Demoni non Necessari

Per ragioni di sicurezza, così come per controllare le risorse di sistema, è importante avere una panoramica di quali servizi siano in esecuzione. I servizi non necessari e inutilizzati dovrebbero essere disabilitati. Per esempio, se non avete bisogno di distribuire pagine web, non c'è bisogno di eseguire un server web come Apache o Nginx.

Sui sistemi basati su SyS-V-init potete controllare lo stato di tutti i servizi nel seguente modo:

```
$ sudo service --status-all
```

Verifica se ognuno dei servizi elencati nell'output del comando è necessario e disabilita tutti i servizi non necessari con (per i sistemi basati su Debian):

```
$ sudo update-rc.d SERVICE-NAME remove
```

O, su sistemi basati su Red Hat:

```
$ sudo chkconfig SERVICE-NAME off
```

Sui moderni sistemi basati su systemd possiamo usare il seguente modo per elencare tutti i servizi in esecuzione:

```
$ systemctl list-units --state active --type service
```

Dovreste quindi disattivare ogni unità di servizio non necessaria con:

```
$ sudo systemctl disable UNIT --now
```

Questo comando fermerà il servizio e lo rimuoverà dalla lista dei servizi, in modo da evitare che si avvii al prossimo avvio del sistema.

Inoltre, per ottenere una panoramica dei servizi di rete in ascolto, puoi usare `netstat` sui vecchi sistemi (a condizione che tu abbia installato il pacchetto `net-tools`):

```
$ netstat -ltu
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
tcp        0      0 0.0.0.0:ssh             0.0.0.0:*               LISTEN
tcp        0      0 localhost:mysql         0.0.0.0:*               LISTEN
tcp6       0      0 [::]:http               [::]:*                  LISTEN
tcp6       0      0 [::]:ssh                [::]:*                  LISTEN
udp        0      0 0.0.0.0:bootpc          0.0.0.0:*
```

Oppure, sui sistemi moderni, si può usare il comando equivalente `ss` (per "socket services"):

```
$ ss -ltu
Netid      State         Recv-Q        Send-Q      Local Address:Port          Peer
Address:Port
udp        UNCONN        0             0                 0.0.0.0:bootpc
0.0.0.0:*
tcp        LISTEN        0             128               0.0.0.0:ssh
```

```
0.0.0.0:*
tcp        LISTEN       0            80             127.0.0.1:mysql
0.0.0.0:*
tcp        LISTEN       0            128                *:http
*:*
tcp        LISTEN       0            128              [::]:ssh
[::]:*
```

TCP Wrapper come una Sorta di Semplice Firewall

In tempi in cui non erano disponibili firewall per Linux, i wrapper TCP erano usati per proteggere le connessioni di rete in un host. Oggi molti programmi non obbediscono più ai *wrapper* TCP. Nelle recenti distribuzioni basate su Red Hat (per esempio Fedora 29) il supporto ai TCP wrapper è stato rimosso completamente. Al fine di supportare i sistemi Linux *legacy* che usano ancora i TCP wrapper, è utile avere alcune conoscenze di base su questa particolare tecnologia.

Useremo ancora una volta il servizio SSH come esempio di base. Il servizio sul nostro host di esempio dovrebbe essere raggiungibile solo dalla rete locale. Per prima cosa controlliamo se il demone SSH usa la libreria `libwrap` che offre supporto ai wrapper TCP:

```
$ ldd /usr/sbin/sshd | grep "libwrap"
        libwrap.so.0 => /lib/x86_64-linux-gnu/libwrap.so.0 (0x00007f91dbec0000)
```

Ora aggiungiamo la seguente linea nel file `/etc/hosts.deny`:

```
sshd: ALL
```

Infine configuriamo un'eccezione nel file `/etc/hosts.allow` per le connessioni SSH dalla rete locale:

```
sshd: LOCAL
```

I cambiamenti hanno effetto immediato, non c'è bisogno di riavviare alcun servizio. Puoi verificarlo con il client `ssh`.

Esercizi Guidati

1. Come si può sbloccare l'account `emma` precedentemente bloccato?

2. In precedenza l'account `emma` aveva una data di scadenza impostata. Come può la data di scadenza essere impostata su `mai`?

3. Immagina che il servizio di stampa CUPS che gestisce i lavori di stampa non sia necessario sul server. Come puoi disabilitare il servizio in modo permanente? Come puoi controllare che la porta appropriata non sia più attiva?

4. Hai installato il server web Nginx. Come puoi controllare se quest'ultimo supporta i *wrapper* TCP?

Esercizi Esplorativi

1. Verifica se l'esistenza del file `/etc/nologin` impedisce il login dell'utente `root`?

2. L'esistenza del file `/etc/nologin` impedisce i login senza password con chiavi SSH?

3. Cosa succede al login, quando il file `/etc/nologin` contiene solo questa linea di testo `login currently is not possible`?

4. Un utente ordinario `emma` può ottenere informazioni sull'utente `root` contenute nel file `/etc/passwd`, per esempio con il comando `grep root /etc/passwd`?

5. Un normale utente `emma` può recuperare informazioni sul suo *password hash* contenuto nel file `/etc/shadow`, per esempio con il comando `grep emma /etc/shadow`?

6. Quali passi devono essere fatti per abilitare e controllare il servizio *legacy* `daytime` attraverso l'utilizzo di xinetd? Nota che questo è solo un esercizio esplorativo, non farlo in un ambiente di produzione.

Sommario

In questa lezione abbiamo imparato:

1. In quale file sono memorizzate le password e alcune impostazioni di sicurezza delle password, per esempio la data di scadenza.
2. Lo scopo del superdemone `xinetd` e come farlo funzionare e avviare il servizio `sshd` a richiesta.
3. A controllare quali servizi di rete sono in esecuzione e come disabilitare i servizi non necessari.
4. A usare i *wrapper* TCP come una sorta di semplice firewall.

Comandi usati nel laboratorio e negli esercizi:

`chage`
: Comando che cambia la scadenza della password di un utente.

`chkconfig`
: Un classico comando usato inizialmente sui sistemi basati su Red Hat per impostare se un servizio viene eseguito all'avvio o meno.

`netstat`
: Una classica utility (ora nel pacchetto `net-tools`) che visualizza le statistiche e le informazioni di rete.

`nologin`
: Un comando che può essere usato al posto della shell di un utente per impedirgli di fare il login.

`passwd`
: Un comando usato per creare o cambiare la password di un utente.

`service`
: Vecchio metodo per controllare lo stato di un demone, come fermare o avviare un servizio.

`ss`
: L'equivalente moderno di `netstat`, ma mostra anche più informazioni sui vari *socket* in uso nel sistema.

`systemctl`
: Il comando di controllo del sistema usato per gestire vari aspetti dei servizi e dei socket su un

Linux che implementa systemd.

`update-rc.d`

Un classico comando simile a `chkconfig` che abilita o disabilita l'avvio di un servizio all'avvio nelle distribuzioni basate su Debian.

`xinetd`

Un superdemone che può controllare l'accesso a un servizio di rete su richiesta, lasciando così un servizio inattivo finché non venga effettivamente richiamato per eseguire un qualche compito.

Risposte agli Esercizi Guidati

1. Come si può sbloccare l'account `emma` precedentemente bloccato?

   Il superutente può eseguire `passwd -u emma` per sbloccare l'account.

2. In precedenza l'account `emma` aveva una data di scadenza impostata. Come può la data di scadenza essere impostata su `mai`?

   Il superutente può usare `chage -E -1 emma` per impostare la data di scadenza a `mai`. Questa impostazione può essere controllata con `chage -l emma`.

3. Immagina che il servizio di stampa CUPS che gestisce i lavori di stampa non sia necessario sul server. Come puoi disabilitare il servizio in modo permanente? Come puoi controllare che la porta appropriata non sia più attiva?

   Come superutente digita:

   ```
   systemctl disable cups.service --now
   ```

   Ora controlla con:

   ```
   netstat -l | grep ":ipp "` or `ss -l | grep ":ipp "
   ```

4. Hai installato il server web Nginx. Come puoi controllare se quest'ultimo supporta i *wrapper* TCP?

   Il comando

   ```
   ldd /usr/sbin/nginx | grep "libwrap"
   ```

   mostrerà una voce nel caso in cui Nginx supporti i wrapper TCP.

Risposte agli Esercizi Esplorativi

1. Verifica se l'esistenza del file `/etc/nologin` impedisce il login dell'utente `root`?

   L'utente `root` è ancora in grado di accedere.

2. L'esistenza del file `/etc/nologin` impedisce i login senza password con chiavi SSH?

   Sì, anche i login senza password sono impediti.

3. Cosa succede al login, quando il file `/etc/nologin` contiene solo questa linea di testo `login currently is not possible`?

   Verrà mostrato il messaggio `login currently is not possible`, e il login sarà impedito.

4. Un utente ordinario `emma` può ottenere informazioni sull'utente `root` contenute nel file `/etc/passwd`, per esempio con il comando `grep root /etc/passwd`?

   Sì, perché tutti gli utenti hanno il permesso di lettura a questo file.

5. Un normale utente `emma` può recuperare informazioni sul suo *password hash* contenuto nel file `/etc/shadow`, per esempio con il comando `grep emma /etc/shadow`?

   No, perché gli utenti ordinari non hanno il permesso di lettura a questo file.

6. Quali passi devono essere fatti per abilitare e controllare il servizio *legacy* `daytime` attraverso l'utilizzo di xinetd? Nota che questo è solo un esercizio esplorativo, non farlo in un ambiente di produzione.

   Per prima cosa cambia il file `/etc/xinetd.d/daytime` e imposta la direttiva `disable = no`. In secondo luogo riavvia il servizio xinetd `systemctl restart xinetd.service` (o `service xinetd restart` sui sistemi con SyS-V-Init). Ora controlla se funziona con `nc localhost daytime`. Invece di `nc` si può usare anche `netcat`.



110.3 Proteggere i dati con la crittografia

Obiettivi LPI di riferimento

LPIC-1 version 5.0, Exam 102, Objective 110.3

Peso

4

Aree di Conoscenza Chiave

- Effettuare la configurazione e l'utilizzo di base del client OpenSSH 2.
- Comprendere il ruolo delle chiavi host del server OpenSSH 2.
- Eseguire la configurazione di base, l'utilizzo e la revoca delle chiavi di GnuPG.
- Usare GPG per crittografare, decrittografare, firmare e verificare i file.
- Comprendere i tunnel delle porte attraverso il protocollo SSH (inclusi i tunnel X11).

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

- `ssh`
- `ssh-keygen`
- `ssh-agent`
- `ssh-add`
- `~/.ssh/id_rsa` and `id_rsa.pub`
- `~/.ssh/id_dsa` and `id_dsa.pub`
- `~/.ssh/id_ecdsa` and `id_ecdsa.pub`
- `~/.ssh/id_ed25519` and `id_ed25519.pub`
- `/etc/ssh/ssh_host_rsa_key` and `ssh_host_rsa_key.pub`

- `/etc/ssh/ssh_host_dsa_key` and `ssh_host_dsa_key.pub`
- `/etc/ssh/ssh_host_ecdsa_key` and `ssh_host_ecdsa_key.pub`
- `/etc/ssh/ssh_host_ed25519_key` and `ssh_host_ed25519_key.pub`
- `~/.ssh/authorized_keys`
- `ssh_known_hosts`
- `gpg`
- `gpg-agent`
- `~/.gnupg/`

110.3 Lezione 1

Certificazione:	LPIC-1
Versione:	5.0
Argomento:	110 Sicurezza
Obiettivo:	110.3 Proteggere i dati con la crittografia
Lezione:	1 di 2

Introduzione

Proteggere i dati con la crittografia è di fondamentale importanza in molti aspetti dell'amministrazione di sistema di oggi: ancora di più quando si tratta di accedere ai sistemi in remoto. Al contrario di soluzioni insicure come *telnet*, *rlogin* o *FTP*, il protocollo *SSH* (*Secure Shell*) è stato progettato con in mente la sicurezza. Utilizzando la crittografia a chiave pubblica autentica sia gli host sia gli utenti e cripta tutti i successivi scambi di informazioni. Inoltre, SSH può essere usato per stabilire *port tunnel*, che — tra le altre cose — permette a un protocollo non criptato di trasmettere dati su una connessione SSH criptata. L'attuale versione raccomandata del protocollo SSH è la 2.0. *OpenSSH*, è un'implementazione libera e open source del protocollo SSH.

Questa lezione tratterà la configurazione di base del client *OpenSSH* così come il ruolo delle chiavi host del server *OpenSSH*. Verrà anche discusso il concetto di tunnel di porte SSH. Useremo due macchine con la seguente configurazione:

Ruolo Macchina	OS	Indirizzo IP	Nome Host	Utente
Client	Debian GNU/Linux 10 (buster)	`192.168.1.55`	`debian`	`carol`
Server	openSUSE Leap 15.1	`192.168.1.77`	`halof`	`ina`

Configurazione e Uso di Base del Client OpenSSH

Sebbene il server e il client OpenSSH siano in pacchetti separati, puoi normalmente installare un metapacchetto che li installerà entrambi in una volta sola. Per stabilire una sessione remota con il server SSH si usa il comando `ssh`, specificando l'utente con cui ci si vuole connettere sulla macchina remota e l'indirizzo IP o l'hostname della macchina remota. La prima volta che ti connetti a un host remoto riceverai un messaggio come questo:

```
carol@debian:~$ ssh ina@192.168.1.77
The authenticity of host '192.168.1.77 (192.168.1.77)' can't be established.
ECDSA key fingerprint is SHA256:5JF7anupYipByCQm2BPvDHRVFJJixeslmppi2NwATYI.
Are you sure you want to continue connecting (yes/no)?
```

Dopo aver digitato `sì` e premuto Invio, ti verrà chiesta la password dell'utente remoto. Se è stata inserita con successo, ti verrà mostrato un messaggio di avvertimento e poi sarai connesso all'host remoto:

```
Warning: Permanently added '192.168.1.77' (ECDSA) to the list of known hosts.
Password:
Last login: Sat Jun 20 10:52:45 2020 from 192.168.1.4
Have a lot of fun...
ina@halof:~>
```

I messaggi sono abbastanza autoesplicativi: poiché era la prima volta che si stabiliva una connessione al server remoto `192.168.1.77`, la sua autenticità non poteva essere verificata con nessun database. Così, il server remoto ha fornito un `ECDSA key fingerprint` della sua chiave pubblica (usando la funzione hash `SHA256`). Una volta accettata la connessione, la chiave pubblica del server remoto veniva aggiunta al database degli *ospiti conosciuti*, permettendo così l'autenticazione del server per le connessioni future. Questa lista di chiavi pubbliche di *known hosts* è conservata nel file `known_hosts` che si trova in `~/.ssh`:

```
ina@halof:~> exit
logout
Connection to 192.168.1.77 closed.
carol@debian:~$ ls .ssh/
known_hosts
```

Sia `.ssh` che `known_hosts` sono stati creati dopo aver stabilito la prima connessione remota. `~/.ssh` è la directory predefinita per la configurazione specifica dell'utente e le informazioni di autenticazione.

NOTE Puoi anche usare `ssh` per eseguire solo un singolo comando sull'host remoto e poi tornare al tuo terminale locale (per esempio: eseguire `ssh ina@halof ls`).

Se stai usando lo stesso utente sia sull'host locale sia su quello remoto, non c'è bisogno di specificare il nome utente quando si stabilisce la connessione SSH. Per esempio: se sei loggato come utente `carol` su `debian` e vuoi connetterti a `halof` sempre come utente `carol`, devi semplicemente digitare `ssh 192.168.1.77` o `ssh halof` (se il nome può essere risolto):

```
carol@debian:~$ ssh halof
Password:
Last login: Wed Jul  1 23:45:02 2020 from 192.168.1.55
Have a lot of fun...
carol@halof:~>
```

Ora supponi di stabilire una nuova connessione remota con un host che per caso ha lo stesso indirizzo IP di `halof` (una cosa comune se usate il DHCP nella vostra LAN). Sarai avvertito della possibilità di un attacco *man-in-the-middle*:

```
carol@debian:~$ ssh john@192.168.1.77
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@    WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED!     @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!
Someone could be eavesdropping on you right now (man-in-the-middle attack)!
It is also possible that a host key has just been changed.
The fingerprint for the ECDSA key sent by the remote host is
SHA256:KH4q3vP6C7e0SEjyG8Wlz9fVlf+jmWJ5139RBxBh3TY.
Please contact your system administrator.
Add correct host key in /home/carol/.ssh/known_hosts to get rid of this message.
Offending ECDSA key in /home/carol/.ssh/known_hosts:1
  remove with:
```

```
  ssh-keygen -f "/home/carol/.ssh/known_hosts" -R "192.168.1.77"
ECDSA host key for 192.168.1.77 has changed and you have requested strict checking.
Host key verification failed.
```

Poiché non hai a che fare con un attacco *man-in-the-middle*, puoi tranquillamente aggiungere l'impronta della chiave pubblica del nuovo host a `.ssh/known_hosts`. Come indica il messaggio, puoi prima usare il comando `ssh-keygen -f "/home/carol/.ssh/known_hosts" -R "192.168.1.77"` per rimuovere la chiave *offendente* (in alternativa, potete andare per `ssh-keygen -R 192.168.1.77` per cancellare tutte le chiavi appartenenti a `192.168.1.77` da `~/.ssh/known_hosts`). Poi, sai in grado di stabilire una connessione al nuovo host.

Login Basato su Chiavi

Puoi impostare il tuo client SSH in modo che *non* fornisca alcuna password al momento del login, ma usi invece le chiavi pubbliche. Questo è il metodo preferito per connettersi a un server remoto via SSH, perché è molto più sicuro. La prima cosa da fare è creare una coppia di chiavi sulla macchina client. Per farlo, userai `ssh-keygen` con l'opzione `-t` specificando il tipo di crittografia che vuoi (*Elliptic Curve Digital Signature Algorithm* nel nostro caso). Poi, ti verrà chiesto il percorso in cui salvare la coppia di chiavi (`~/.ssh/` è conveniente così come la posizione predefinita) e una passphrase. Mentre una *passphrase* è opzionale, è altamente raccomandato usarne sempre una.

```
carol@debian:~/.ssh$ ssh-keygen -t ecdsa
Generating public/private ecdsa key pair.
Enter file in which to save the key (/home/carol/.ssh/id_ecdsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/carol/.ssh/id_ecdsa.
Your public key has been saved in /home/carol/.ssh/id_ecdsa.pub.
The key fingerprint is:
SHA256:tlamD0SaTquPZYdNepwj8XN4xvqmHCbe8g5FKKUfMo8 carol@debian
The key's randomart image is:
+---[ECDSA 256]---+
|      .          |
|     o .         |
|    = o o        |
|     B *         |
|    E B S o      |
|     o & O       |
|      @ ^ =      |
|     *.@ @.      |
|    o.o+B+o      |
```

```
+----[SHA256]-----+
```

NOTE Quando si crea la coppia di chiavi, si può passare a `ssh-keygen` l'opzione `-b` per specificare la dimensione della chiave in bit (per esempio: `ssh-keygen -t ecdsa -b 521`).

Il comando precedente ha prodotto altri due file nella tua directory `~/.ssh`:

```
carol@debian:~/.ssh$ ls
id_ecdsa  id_ecdsa.pub  known_hosts
```

`id_ecdsa`
Questa è la tua chiave privata.

`id_ecdsa.pub`
Questa è la tua chiave pubblica.

NOTE Nella crittografia asimmetrica (o anche crittografia a chiave pubblica), la chiave pubblica e quella privata sono matematicamente legate l'una all'altra in modo tale che qualsiasi cosa sia criptata da una può essere decriptata solo dall'altra.

La prossima cosa che devi fare è aggiungere la tua chiave pubblica al file `~/.ssh/authorized_keys` dell'utente con cui vuoi loggarti sull'host remoto (se la directory `~/.ssh` non esiste già, dovrai prima crearla). Puoi copiare la tua chiave pubblica nel server remoto in diversi modi: usando una chiavetta USB, attraverso il comando `scp` - che trasferirà il file usando SSH - o prelevando con `cat` il contenuto della tua chiave pubblica e inserendolo in `ssh` come segue:

```
carol@debian:~/.ssh$ cat id_ecdsa.pub |ssh ina@192.168.1.77 'cat >> .ssh/authorized_keys'
Password:
```

Una volta che la chiave pubblica è stata aggiunta al file `authorized_keys` sull'host remoto, puoi affrontare due scenari quando cerchi di stabilire una nuova connessione:

- Se non hai fornito una passphrase quando hai creato la coppia di chiavi, sarai loggato automaticamente. Anche se comodo, questo metodo può essere insicuro a seconda della situazione:

```
carol@debian:~$ ssh ina@192.168.1.77
Last login: Thu Jun 25 20:31:03 2020 from 192.168.1.55
```

```
Have a lot of fun...
ina@halof:~>
```

- Se hai fornito una passphrase quando hai creato la coppia di chiavi, dovrai inserirla a ogni connessione come se fosse una password. Oltre alla chiave pubblica, questo metodo aggiunge un ulteriore livello di sicurezza sotto forma di passphrase e può — quindi — essere considerato più sicuro. Per quanto riguarda la comodità, tuttavia, è esattamente la stessa cosa che dover inserire una password ogni volta che si stabilisce una connessione. Se non usa una passphrase e qualcuno riesce a ottenere il tuo file chiave SSH privato, avrebbe accesso a ogni server su cui è installata la tua chiave pubblica.

```
carol@debian:~/.ssh$ ssh ina@192.168.1.77
Enter passphrase for key '/home/carol/.ssh/id_ecdsa':
Last login: Thu Jun 25 20:39:30 2020 from 192.168.1.55
Have a lot of fun...
ina@halof:~>
```

C'è però un modo che combina sicurezza e convenienza: usare l'agente di autenticazione *SSH* (`ssh-agent`). L'agente di autenticazione ha bisogno di creare una propria shell e terrà le tue chiavi private per l'autenticazione a chiave pubblica in memoria per il resto della sessione. Vediamo un po' più in dettaglio come funziona:

1. Usa `ssh-agent` per avviare una nuova Shell Bash:

```
carol@debian:~/.ssh$ ssh-agent /bin/bash
carol@debian:~/.ssh$
```

2. Usa il comando `ssh-add` per aggiungere la tua chiave privata in un'area sicura della memoria. Se hai fornito una passphrase quando hai generato la coppia di chiavi, il che è raccomandato per una maggiore sicurezza, ti verrà richiesta:

```
carol@debian:~/.ssh$ ssh-add
Enter passphrase for /home/carol/.ssh/id_ecdsa:
Identity added: /home/carol/.ssh/id_ecdsa (carol@debian)
```

   Una volta che la tua identità è stata aggiunta, potrai accedere a qualsiasi server remoto su cui sia presente la chiave pubblica senza dover digitare nuovamente la tua *passphrase*. È pratica comune sui desktop moderni eseguire questo comando all'avvio del computer, in quanto rimarrà in memoria fino allo spegnimento del computer (fino a quando la chiave non venga

scaricata manualmente).

Completiamo questa sezione elencando i quattro tipi di algoritmi a chiave pubblica che possono essere specificati con `ssh-keygen`:

`RSA`

Prende il nome dai suoi creatori Ron Rivest, Adi Shamir e Leonard Adleman e fu pubblicato nel 1977. È considerata sicura e ancora oggi ampiamente utilizzata. La sua dimensione minima della chiave è di 1024 bit (l'impostazione predefinita è 2048).

`DSA`

Il *Digital Signature Algorithm* si è dimostrato insicuro ed è stato deprecato a partire da OpenSSH 7.0. Le chiavi DSA devono essere lunghe *esattamente* 1024 bit.

`ecdsa`

L'*Elliptic Curve Digital Signature Algorithm* è un miglioramento del DSA e quindi considerato più sicuro. Utilizza la crittografia a curva ellittica. La lunghezza della chiave ECDSA è determinata da una delle tre possibili dimensioni della curva ellittica in bit: 256, 384 o 512.

`ed25519`

È un'implementazione di `EdDSA` — *Edwards-curve Digital Signature Algorithm* — che usa la più forte curva 25519. È considerata la più sicura di tutte. Tutte le chiavi Ed25519 hanno una lunghezza fissa di 256 bit.

NOTE Se invocato senza alcuna specificazione `-t`, `ssh-keygen` genererà per default una coppia di chiavi `RSA`.

Il Ruolo delle Chiavi Host del Server OpenSSH

La directory di configurazione globale per OpenSSH si trova nella directory `/etc`:

```
halof:~ # tree /etc/ssh
/etc/ssh
├── moduli
├── ssh_config
├── ssh_host_dsa_key
├── ssh_host_dsa_key.pub
├── ssh_host_ecdsa_key
├── ssh_host_ecdsa_key.pub
├── ssh_host_ed25519_key
├── ssh_host_ed25519_key.pub
```

```
├── ssh_host_rsa_key
├── ssh_host_rsa_key.pub
└── sshd_config

0 directories, 11 files
```

Oltre a `moduli` e ai file di configurazione per il client (`ssh_config`) e il server (`sshd_config`), troverai quattro coppie di chiavi — una coppia di chiavi per ogni algoritmo supportato — che vengono create quando viene installato il server *OpenSSH*. Come già notato, il server usa queste *host key* per identificarsi verso i client. Il modello di nome è il seguente:

Chiavi private

`ssh_host_` prefix + *algorithm* + `key` suffix (es.: `ssh_host_rsa_key`)

Chiavi pubbliche (o impronte digitali di chiavi pubbliche)

`ssh_host_` prefix + *algorithm* + `key.pub` suffix (es.: `ssh_host_rsa_key.pub`)

NOTE

Un'impronta digitale (*fingerprint*) viene creata applicando una funzione di *hash* crittografica a una chiave pubblica. Poiché le impronte digitali sono più corte delle chiavi a cui si riferiscono, sono utili per semplificare alcuni compiti di gestione delle chiavi.

I permessi sui file contenenti le chiavi private sono `0600` o `-rw-------`: leggibili e scrivibili solo dal proprietario (`root`). D'altra parte, tutti i file delle chiavi pubbliche sono leggibili anche dai membri del gruppo del proprietario e da tutti gli altri (`0644` o `-rw-r—r--`):

```
halof:~ # ls -l /etc/ssh/ssh_host_*
-rw------- 1 root root 1381 Dec 21 20:35 /etc/ssh/ssh_host_dsa_key
-rw-r--r-- 1 root root  605 Dec 21 20:35 /etc/ssh/ssh_host_dsa_key.pub
-rw------- 1 root root  505 Dec 21 20:35 /etc/ssh/ssh_host_ecdsa_key
-rw-r--r-- 1 root root  177 Dec 21 20:35 /etc/ssh/ssh_host_ecdsa_key.pub
-rw------- 1 root root  411 Dec 21 20:35 /etc/ssh/ssh_host_ed25519_key
-rw-r--r-- 1 root root   97 Dec 21 20:35 /etc/ssh/ssh_host_ed25519_key.pub
-rw------- 1 root root 1823 Dec 21 20:35 /etc/ssh/ssh_host_rsa_key
-rw-r--r-- 1 root root  397 Dec 21 20:35 /etc/ssh/ssh_host_rsa_key.pub
```

È possibile visualizzare le impronte digitali delle chiavi passando a `ssh-keygen` l'opzione `-l`. Devi anche fornire il `-f` per specificare il percorso del file delle chiavi:

```
halof:~ # ssh-keygen -l -f /etc/ssh/ssh_host_ed25519_key
256 SHA256:8cnPrinC49ZHc+/9Ai5pV+1JfZ4WBRZhd3rDOsc2zlA root@halof (ED25519)
```

```
halof:~ # ssh-keygen -l -f /etc/ssh/ssh_host_ed25519_key.pub
256 SHA256:8cnPrinC49ZHc+/9Ai5pV+1JfZ4WBRZhd3rDOsc2zlA root@halof (ED25519)
```

Per visualizzare l'impronta della chiave e la sua *random art*, basta aggiungere `-v` nel seguente modo:

```
halof:~ # ssh-keygen -lv -f /etc/ssh/ssh_host_ed25519_key.pub
256 SHA256:8cnPrinC49ZHc+/9Ai5pV+1JfZ4WBRZhd3rDOsc2zlA root@halof (ED25519)
+--[ED25519 256]--+
|              +oo|
|             .+o.|
|        .    ..E.|
|         + .  +.o|
|        S +  + *o|
|          ooo Oo=|
|     . . . =o+.==|
|      = o =oo o=o|
|     o.o +o+..o.+|
+----[SHA256]-----+
```

Tunnel SSH

OpenSSH dispone di una funzione di inoltro molto potente per cui il traffico su una porta sorgente viene fatto transitare - e criptato - attraverso un processo SSH che poi lo reindirizza a una porta su un host di destinazione. Questo meccanismo è noto come *port tunnelling* o *port forwarding* e ha importanti vantaggi tra i quali:

- Permette di bypassare i firewall per accedere alle porte di host remoti.
- Permette l'accesso dall'esterno a un host della rete privata.
- Fornisce la crittografia per tutti gli scambi di dati.

In parole povere, possiamo distinguere tra tunnelling di porte locali e remote.

Tunnel Locale di Porta

Si definisce localmente una porta per inoltrare il traffico all'host di destinazione attraverso il processo SSH che si trova nel mezzo. Il processo SSH può essere eseguito sull'host locale o su un server remoto. Per esempio, se per qualche ragione volessi creare un tunnel per una connessione a `www.gnu.org` attraverso SSH usando la porta `8585` sulla tua macchina locale, faresti qualcosa del genere:

```
carol@debian:~$ ssh -L 8585:www.gnu.org:80 debian
carol@debian's password:
Linux debian 4.19.0-9-amd64 #1 SMP Debian 4.19.118-2 (2020-04-29) x86_64

The programs included with the Debian GNU/Linux system are free software;
(...)
Last login: Sun Jun 28 13:47:27 2020 from 127.0.0.1
```

La spiegazione è la seguente: con l'opzione `-L`, specifichiamo la porta locale `8585` per connetterci alla porta *http* `80` su `www.gnu.org` usando il processo SSH in esecuzione su `debian` — il nostro *localhost*. Avremmo potuto scrivere `ssh -L 8585:www.gnu.org:80 localhost` con lo stesso effetto. Se ora usi un browser web per andare su `http://localhost:8585`, sarai inoltrato a `www.gnu.org`. Per scopi dimostrativi, useremo `lynx` (il "classico" browser web in modalità testo):

```
carol@debian:~$ lynx http://localhost:8585
(...)
  * Back to Savannah Homepage
     * Not Logged in
     * Login
     * New User
     * This Page
     * Language
     * Clean Reload
     * Printer Version
     * Search
     * _
(...)
```

Se avessi voluto fare la stessa identica cosa, ma connettendoti tramite un processo SSH in esecuzione su `halof`, avresti invece proceduto in questo modo:

```
carol@debian:~$ ssh -L 8585:www.gnu.org:80 -Nf ina@192.168.1.77
Enter passphrase for key '/home/carol/.ssh/id_ecdsa':
carol@debian:~$
carol@debian:~$ lynx http://localhost:8585
(...)
  * Back to Savannah Homepage
     * Not Logged in
     * Login
     * New User
     * This Page
```

```
     * Language
     * Clean Reload
     * Printer Version
     * Search
     * _
(...)
```

È importante notare tre dettagli nel comando:

- Grazie all'opzione `-N` non abbiamo fatto il login su `halof` ma abbiamo invece fatto il port forwarding.
- L'opzione `-f` ha comunicato a SSH di funzionare in background.
- Abbiamo specificato l'utente `ina` per fare l'inoltro: `ina@192.168.1.77`.

Tunnel Remoto di Porta

Nel tunneling remoto delle porte (o reverse port forwarding) il traffico che arriva su una porta del server remoto viene inoltrato al processo SSH in esecuzione sul tuo host locale, e da lì alla porta specificata sul server di destinazione (che potrebbe anche essere la tua macchina locale). Per esempio, diciamo che vuoi permettere a qualcuno al di fuori della tua rete di accedere al server web Apache in esecuzione sul tuo host locale attraverso la porta `8585` del server SSH in esecuzione su `halof` (`192.168.1.77`). Procederesti con il seguente comando:

```
carol@debian:~$ ssh -R 8585:localhost:80 -Nf ina@192.168.1.77
Enter passphrase for key '/home/carol/.ssh/id_ecdsa':
carol@debian:~$
```

Ora chiunque stabilisca una connessione con `halof` sulla porta `8585` vedrà la homepage predefinita di Apache2 di `Debian`:

```
carol@debian:~$ lynx 192.168.1.77:8585
(...)
                                                                Apache2 Debian Default

Page: It works (p1 of 3)
   Debian Logo Apache2 Debian Default Page
   It works!

   This is the default welcome page used to test the correct operation of the Apache2 server
after
   installation on Debian systems. If you can read this page, it means that the Apache HTTP
```

```
server
   installed at this site is working properly. You should replace this file (located at
   /var/www/html/index.html) before continuing to operate your HTTP server.
(...)
```

NOTE C'è un terzo tipo, più complesso, di inoltro delle porte che esula dallo scopo di questa lezione: l'*inoltro dinamico delle porte*. Invece di interagire con una singola porta, questo tipo di inoltro utilizza varie comunicazioni TCP su una serie di porte.

Tunnel X11

Ora che hai capito i tunnel di porta, completiamo questa lezione discutendo il tunnelling di X11 (conosciuto anche come *X11forwarding*). Attraverso un tunnel X11, l'*X Window System* sull'host remoto viene inoltrato alla macchina locale. Per questo, devi solo passare a `ssh` l'opzione `-X`:

```
carol@debian:~$ ssh -X ina@halof
...
```

Ora puoi lanciare un'applicazione grafica come il browser web `firefox` con il seguente risultato: l'applicazione verrà eseguita sul server remoto, ma la sua visualizzazione verrà inoltrata al tuo host locale.

Se invece avvii una nuova sessione SSH con l'opzione `-x`, *X11forwarding* sarà disabilitato. Prova ad avviare `firefox` ora e otterrai un errore come il seguente:

```
carol@debian:~$ ssh -x ina@halof
carol@192.168.0.106's password:
(...)
ina@halof:~$ firefox

(firefox-esr:1779): Gtk-WARNING **: 18:45:45.603: Locale not supported by C library.
    Using the fallback 'C' locale.
Error: no DISPLAY environment variable specified
```

NOTE Le tre direttive di configurazione relative al port forwarding locale, al port forwarding remoto e al forwarding X11 sono rispettivamente `AllowTcpForwarding`, `GatewayPorts` e `X11Forwarding`. Per maggiori informazioni, digitare `man ssh_config` e/o `man sshd_config`.

Esercizi Guidati

1. Dopo aver eseguito un accesso come utente `sonya` sulla macchina client, esegui i seguenti compiti SSH sul server remoto `halof`:
   - Esegui il comando per elencare il contenuto di `~/.ssh` come utente `serena` sull'host remoto; poi torna al tuo terminale locale.
   - Accedi come utente `serena` sull'host remoto.
   - Accedi come utente `sonya` sull'host remoto.
   - Cancella tutte le chiavi appartenenti a `halof` dal tuo file locale `~/.ssh/known_hosts`.
   - Sulla tua macchina client, crea una coppia di chiavi `ecdsa` da 256 bit.
   - Sulla tua macchina client, crea una coppia di chiavi `ed25519` da 256 bit.
2. Metti i seguenti passi nel giusto ordine per stabilire una connessione SSH usando l'agente di autenticazione *SSH*:
   - Sul client, avvia una nuova shell Bash per l'*authentication agent* con `ssh-agent /bin/bash`.
   - Sul client, crea una coppia di chiavi usando `ssh-keygen`.
   - Sul client, aggiungi la tua chiave privata in un'area sicura della memoria con `ssh-add`.
   - Aggiungi la chiave pubblica del client al file `~/.ssh/authorized_keys` dell'utente con cui vuoi effettuare il login sull'host remoto.
   - Se non esiste già, crea il file `~/.ssh` per l'utente con cui volete fare il login sul server.
   - Connettiti al server remoto.

   L'ordine corretto è:

Passo 1:	

Passo 2:	
Passo 3:	
Passo 4:	
Passo 5:	
Passo 6:	

3. Per quanto riguarda il *port forwarding*, quale opzione e direttiva si usa per i seguenti tipi di tunnel:

Tipo di Tunnel	Opzioni	Direttiva
Local		
Remote o Reverse		
X		

4. Supponiamo che tu digiti il comando `ssh -L 8888:localhost:80 -Nf ina@halof` nel terminale della tua macchina client. Sempre sulla macchina client, punta un browser a `http://localhost:8888`. Che cosa ottieni?

Esercizi Esplorativi

1. Per quanto riguarda le direttive di sicurezza SSH:
   - Quale direttiva è usata in `/etc/ssh/sshd_config` per abilitare i login di `root`:
   - Quale direttiva useresti in `/etc/ssh/sshd_config` per accettare connessioni SSH solo da un account locale :
2. Quando si usa lo stesso utente sia sul client sia sul server, quale comando `ssh` si può usare per trasferire la chiave pubblica del client sul server in modo da poter effettuare il login tramite autenticazione a chiave pubblica?
3. Crea due tunnel di porte locali in un unico comando che inoltra le porte locali non privilegiate 8080 e 8585 attraverso il server remoto `halof` a rispettivamente i siti web `www.gnu.org` e `www.melpa.org`. Usa l'utente `ina` sul server remoto e non dimenticare di usare le opzioni `-Nf`:

Sommario

In questa lezione abbiamo discusso di *OpenSSH 2*, che usa il protocollo *Secure Shell* per criptare le comunicazioni tra server e client. Hai imparato:

- come accedere a un server remoto.
- come eseguire comandi a distanza.
- come creare coppie di chiavi.
- come stabilire login basati su chiavi.
- come usare l'*authentication agent* per una maggiore sicurezza e comodità.
- gli algoritmi a chiave pubblica supportati da *OpenSSH*: `RSA`, `DSA`, `ecdsa`, `ed25519`.
- il ruolo delle chiavi host *OpenSSH*.
- come creare tunnel di porte: locale, remoto e X.

I seguenti comandi sono stati discussi in questa lezione:

`ssh`
: Accede o esegue comandi su una macchina remota.

`ssh-keygen`
: Genera, gestisce e converte le chiavi di autenticazione.

`ssh-agent`
: Agente di autenticazione OpenSSH.

`ssh-add`
: Aggiunge identità a chiave privata all'agente di autenticazione.

Risposte agli Esercizi Guidati

1. Dopo aver eseguito un accesso come utente `sonya` sulla macchina client, esegui i seguenti compiti SSH sul server remoto `halof`:
   - Esegui il comando per elencare il contenuto di `~/.ssh` come utente `serena` sull'host remoto; poi torna al tuo terminale locale.

     ```
     ssh serena@halof ls .ssh
     ```

   - Accedi come utente `serena` sull'host remoto.

     ```
     ssh serena@halof
     ```

   - Accedi come utente `sonya` sull'host remoto.

     ```
     ssh halof
     ```

   - Cancella tutte le chiavi appartenenti a `halof` dal tuo file locale `~/.ssh/known_hosts`.

     ```
     ssh-keygen -R halof
     ```

   - Sulla tua macchina client, crea una coppia di chiavi `ecdsa` da 256 bit.

     ```
     ssh-keygen -t ecdsa -b 256
     ```

   - Sulla tua macchina client, crea una coppia di chiavi `ed25519` da 256 bit.

     ```
     ssh-keygen -t ed25519
     ```

2. Metti i seguenti passi nel giusto ordine per stabilire una connessione SSH usando l'agente di autenticazione *SSH*:
   - Sul client, avvia una nuova shell Bash per l'*authentication agent* con `ssh-agent /bin/bash`.
   - Sul client, crea una coppia di chiavi usando `ssh-keygen`.
   - Sul client, aggiungi la tua chiave privata in un'area sicura della memoria con `ssh-add`.

- Aggiungi la chiave pubblica del client al file `~/.ssh/authorized_keys` dell'utente con cui vuoi effettuare il login sull'host remoto.
- Se non esiste già, crea il file `~/.ssh` per l'utente con cui vuoi fare il login sul server.
- Connettiti al server remoto.

L'ordine corretto è:

Passo 1:	Sul client, crea una coppia di chiavi usando `ssh-keygen`.
Passo 2:	Se non esiste già, crea il file `~/.ssh` per l'utente con cui volete fare il login sul server.
Passo 3:	Aggiungi la chiave pubblica del client al file `~/.ssh/authorized_keys` dell'utente con cui vuoi effettuare il login sull'host remoto.
Passo 4:	Sul client, avvia una nuova shell Bash per l'*authentication agent* con `ssh-agent /bin/bash`.
Passo 5:	Sul client, aggiungi la tua chiave privata in un'area sicura della memoria con `ssh-add`.
Passo 6:	Connettiti al server remoto.

3. Per quanto riguarda il *port forwarding*, quale opzione e direttiva si usa per i seguenti tipi di tunnel:

Tipo di Tunnel	**Opzioni**	**Direttiva**
Local	`-L`	`AllowTcpForwarding`
Remote o Reverse	`-R`	`GatewayPorts`
X	`-X`	`X11Forwarding`

4. Supponiamo che tu digiti il comando `ssh -L 8888:localhost:80 -Nf ina@halof` nel terminale della tua macchina client. Sempre sulla macchina client, punta un browser a `http://localhost:8888`. Che cosa ottieni?

   La homepage del webserver di `halof`.

Risposte agli Esercizi Esplorativi

1. Per quanto riguarda le direttive di sicurezza SSH:
   - Quale direttiva è usata in `/etc/ssh/sshd_config` per abilitare i login di `root`:

     `PermitRootLogin`

   - Quale direttiva useresti in `/etc/ssh/sshd_config` per accettare connessioni SSH solo da un account locale :

     `AllowUsers`

2. Quando si usa lo stesso utente sia sul client sia sul server, quale comando `ssh` si può usare per trasferire la chiave pubblica del client sul server in modo da poter effettuare il login tramite autenticazione a chiave pubblica?

   `ssh-copy-id`

3. Crea due tunnel di porte locali in un unico comando che inoltra le porte locali non privilegiate 8080 e 8585 attraverso il server remoto `halof` a rispettivamente i siti web `www.gnu.org` e `www.melpa.org`. Usa l'utente `ina` sul server remoto e non dimenticare di usare le opzioni `-Nf`:

   ```
   ssh -L 8080:www.gnu.org:80 -L 8585:www.melpa.org:80 -Nf ina@halof
   ```



110.3 Lezione 2

Certificazione:	LPIC-1
Versione:	5.0
Argomento:	110 Sicurezza
Obiettivo:	110.3 Proteggere i dati con la crittografia
Lezione:	2 di 2

Introduzione

Nella lezione precedente abbiamo imparato come usare *OpenSSH* per crittografare le sessioni di login remoto così come ogni altro successivo scambio di informazioni. Ci possono essere altri scenari in cui si può desiderare di criptare file o email in modo che raggiungano il loro destinatario in modo sicuro e al riparo da occhi indiscreti. Potresti anche aver bisogno di firmare digitalmente quei file o messaggi per evitare che vengano manomessi.

Un ottimo strumento per questo tipo di usi è il *GNU Privacy Guard* (anche noto come *GnuPG* o semplicemente *GPG*); implementazione libera e open source del proprietario *Pretty Good Privacy* (*PGP*). *GPG* usa lo standard *OpenPGP* come definito dall'*OpenPGP Working Group* dell'*Internet Engineering Task Force* (*IETF*) presente nell' RFC 4880. In questa lezione scopriremo i fondamenti del *GNU Privacy Guard*.

Effettuare la Configurazione di Base, Utilizzare ed Eseguire Attività di Revoca con GnuPG

Proprio come per SSH, il meccanismo sottostante a GPG è quello della *crittografia asimmetrica* o

crittografia a chiave pubblica. Un utente genera una coppia di chiavi che è composta da una *chiave privata* e una *chiave pubblica*. Le chiavi sono legate matematicamente in modo tale che ciò che è criptato da una può essere decriptato *solo* dall'altra. Affinché la comunicazione avvenga con successo, l'utente deve inviare la sua chiave pubblica al destinatario.

Configurazione e Uso di GnuPG

Il comando per lavorare con GPG è `gpg`. Puoi passargli una serie di opzioni per eseguire diversi compiti. Cominciamo generando una coppia di chiavi come utente `carol`. Userai il comando `gpg --gen-key`:

```
carol@debian:~$ gpg --gen-key
gpg (GnuPG) 2.2.12; Copyright (C) 2018 Free Software Foundation, Inc.
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.

gpg: directory '/home/carol/.gnupg' created
gpg: keybox '/home/carol/.gnupg/pubring.kbx' created
Note: Use "gpg --full-generate-key" for a full featured key generation dialog.

GnuPG needs to construct a user ID to identify your key.

Real name:

(...)
```

Dopo averti informato, tra le altre cose, che la directory di configurazione `~/.gnupg` e il tuo portachiavi pubblico `~/.gnugpg/pubring.kbx` sono stati creati, `gpg` procede con il chiederti di fornire il tuo vero nome e l'indirizzo email:

```
(...)
Real name: carol
Email address: carol@debian
You selected this USER-ID:
    "carol <carol@debian>"

Change (N)ame, (E)mail, or (O)kay/(Q)uit?
```

Se sei d'accordo con la `USER-ID` risultante e premi `O`, ti verrà chiesta una *passphrase* (si raccomanda di metterne una abbastanza complessa):

```
┌──────────────────────────────────────────────────────┐
│ Please enter the passphrase to                       │
│ protect your new key                                 │
│                                                      │
│ Passphrase:  │

(...)
```

Verranno visualizzati alcuni ulteriori messaggi che informeranno della creazione di altri file e delle chiavi stesse, quindi avrai finito con il processo di generazione delle chiavi:

```
(...)
We need to generate a lot of random bytes. It is a good idea to perform
some other action (type on the keyboard, move the mouse, utilize the
disks) during the prime generation; this gives the random number
generator a better chance to gain enough entropy.
gpg: /home/carol/.gnupg/trustdb.gpg: trustdb created
gpg: key 19BBEFD16813034E marked as ultimately trusted
gpg: directory '/home/carol/.gnupg/openpgp-revocs.d' created
gpg: revocation certificate stored as '/home/carol/.gnupg/openpgp-
revocs.d/D18FA0021F644CDAF57FD0F919BBEFD16813034E.rev'
public and secret key created and signed.

pub   rsa3072 2020-07-03 [SC] [expires: 2022-07-03]
      D18FA0021F644CDAF57FD0F919BBEFD16813034E
uid                      carol <carol@debian>
sub   rsa3072 2020-07-03 [E] [expires: 2022-07-03]
```

Ora puoi vedere che cosa c'è dentro la directory `~/.gnupg` (la directory di configurazione di GPG):

```
carol@debian:~/.gnupg$ ls -l
total 16
drwx------ 2 carol carol 4096 Jul  3 23:34 openpgp-revocs.d
drwx------ 2 carol carol 4096 Jul  3 23:34 private-keys-v1.d
-rw-r--r-- 1 carol carol 1962 Jul  3 23:34 pubring.kbx
-rw------- 1 carol carol 1240 Jul  3 23:34 trustdb.gpg
```

Spieghiamo l'uso di ogni file:

`opengp-revocs.d`

Il certificato di revoca che è stato creato insieme alla coppia di chiavi è conservato qui. I

permessi su questa directory sono abbastanza restrittivi poiché chiunque abbia accesso al certificato potrebbe revocare la chiave (maggiori informazioni sulla revoca della chiave nella prossima sottosezione).

`private-keys-v1.d`

Questa è la directory che mantiene le vostre chiavi private, quindi i permessi sono restrittivi.

`pubring.kbx`

Questo è il tuo *portachiavi pubblico*. Memorizza le tue e qualsiasi altra chiave pubblica importata.

`trustdb.gpg`

"Il database della fiducia". Riguarda il concetto di *Web of Trust* (che esula dallo scopo di questa lezione).

NOTE

L'arrivo di *GnuPG 2.1* ha portato alcuni cambiamenti significativi, come la scomparsa dei file `secring.gpg` e `pubring.gpg` in favore rispettivamente di `private-keys-v1.d` e `pubring.kbx`.

Una volta che la tua coppia di chiavi è stata creata, puoi vedere le tue chiavi pubbliche con `gpg --list-keys`; mostrerà il contenuto del tuo portachiavi pubblico:

```
carol@debian:~/.gnupg$ gpg --list-keys
/home/carol/.gnupg/pubring.kbx
------------------------------
pub   rsa3072 2020-07-03 [SC] [expires: 2022-07-03]
      D18FA0021F644CDAF57FD0F919BBEFD16813034E
uid           [ultimate] carol <carol@debian>
sub   rsa3072 2020-07-03 [E] [expires: 2022-07-03]
```

La stringa esadecimale `D18FA0021F644CDAF57FD0F919BBEFD16813034E` è la tua *impronta della chiave pubblica.*

NOTE

Oltre alla `USER-ID` (`carol` nell'esempio), c'è anche la `KEY-ID`. Il `KEY-ID` consiste nelle ultime 8 cifre esadecimali dell'impronta della tua chiave pubblica (`6813034E`). Puoi controllare il *tuo fingerprint* della chiave con il comando `gpg --fingerprint USER-ID`.

Distribuzione e Revoca della Chiave

Ora che hai la tua chiave pubblica, dovrai salvarla (cioè *esportarla*) in un file per renderla

disponibile ai futuri destinatari. Essi potranno allora usarla per criptare i file o i messaggi destinati a te (poiché sei l'unico in possesso della chiave privata, sarei anche il solo a poterli decriptare e leggere). Allo stesso modo, anche i tuoi destinatari la useranno per decifrare e verificare i tuoi messaggi/file cifrati o firmati. Il comando da usare è `gpg --export` seguito dal `USER-ID` e da un reindirizzamento al nome del file di output di tua scelta:

```
carol@debian:~/.gnupg$ gpg --export carol > carol.pub.key
carol@debian:~/.gnupg$ ls
carol.pub.key  openpgp-revocs.d  private-keys-v1.d  pubring.kbx  trustdb.gpg
```

NOTE

Passando l'opzione `-a` o `--armor` a `gpg --export` (per esempio: `gpg --export --armor carol > carol.pub.key`) si creerà un output ASCII "blindato" (invece del formato binario OpenPGP predefinito) che può essere tranquillamente spedito via email.

Come già detto, devi ora inviare il file di chiave pubblica (`carol.pub.key`) al destinatario con cui vuoi scambiare informazioni. Per esempio, mandiamo il file della chiave pubblica a `ina` sul server remoto `halof` usando `scp` (*secure copy*):

```
carol@debian:~/.gnupg$ scp carol.pub.key ina@halof:/home/ina/
Enter passphrase for key '/home/carol/.ssh/id_ecdsa':
carol.pub.key
100% 1740   775.8KB/s   00:00
carol@debian:~/.gnupg$
```

`ina` è ora in possesso di `carol.pub.key`. La userà per criptare un file e inviarlo a `carol` nella prossima sezione.

NOTE

Un altro mezzo di distribuzione delle chiavi pubbliche è attraverso l'uso di un *key server*: si carica la propria chiave pubblica sul server con il comando `gpg --keyserver keyserver-name --send-keys KEY-ID` e gli altri utenti la ottengono (cioè la *importano*) con `gpg --keyserver keyserver-name --recv-keys KEY-ID`.

Chiudiamo questa sezione discutendo la revoca delle chiavi. La revoca delle chiavi dovrebbe essere usata quando le chiavi private sono state compromesse o ritirate. Il primo passo è creare un certificato di revoca passando a `gpg` l'opzione `--gen-revoke` seguita da `USER-ID`. Puoi far precedere a `--gen-revoke` l'opzione `--output` seguita da un nome di file di destinazione per salvare il certificato risultante in un file (invece di farlo stampare sullo schermo del terminale). I messaggi di output durante il processo di revoca sono abbastanza autoesplicativi:

```
sonya@debian:~/.gnupg$ gpg --output revocation_file.asc --gen-revoke sonya

sec  rsa3072/0989EB7E7F9F2066 2020-07-03 sonya <sonya@debian>

Create a revocation certificate for this key? (y/N) y
Please select the reason for the revocation:
  0 = No reason specified
  1 = Key has been compromised
  2 = Key is superseded
  3 = Key is no longer used
  Q = Cancel
(Probably you want to select 1 here)
Your decision? 1
Enter an optional description; end it with an empty line:
> My laptop was stolen.
>
Reason for revocation: Key has been compromised
My laptop was stolen.
Is this okay? (y/N) y
ASCII armored output forced.
Revocation certificate created.

Please move it to a medium which you can hide away; if Mallory gets
access to this certificate he can use it to make your key unusable.
It is smart to print this certificate and store it away, just in case
your media become unreadable.  But have some caution:  The print system of
your machine might store the data and make it available to others!
```

Il certificato di revoca è stato salvato nel file `revocation_file.asc` (`asc` per il formato ASCII):

```
sonya@debian:~/.gnupg$ ls
openpgp-revocs.d  private-keys-v1.d  pubring.kbx  revocation_file.asc  trustdb.gpg
sonya@debian:~/.gnupg$ cat revocation_file.asc
-----BEGIN PGP PUBLIC KEY BLOCK-----
Comment: This is a revocation certificate

iQHDBCABCgAtFiEEiIVjfDnnpieFi0wvnlcN6yLCeHEFAl8ASx4PHQJzdG9sZW4g
bGFwdG9wAAoJEJ5XDesiwnhxT9YMAKkjQiMpo9Uyiy9hyvukPPSrLcmtAGLk4pKS
pLZfzA5kxa+HPQwBglAEvfNRR6VMxqXUgUGYC/IAyQQM62oNAcY2PCPrxyJNgVF7
8l4mMZKvW++5ikjZwyg6WWV0+w6oroeo9qruJFjcu752p4T+9gsHVa2r+KRqcPQe
aZ65sAvsBJlcsUDZqfWUXg2kQp9mNPCdQuqvDaKRgNCHA1zbzNFzXWVd2X5RgFo5
nY+tUP8ZQA9DTQPBLPcggICmfLopMPZYB2bft5geb2mMi2oNpf9CNPdQkdccimNV
aRjqdUP9C89PwTafBQkQiONlsR/dWTFcqprG5KOWQPA7xjeMV8wretdEgsyTxqHp
```

```
v1iRzwjshiJCKBXXvz7wSmQrJ4OfiMDHeS4ipR0AYdO8QCzmOzmcFQKikGSHGMy1
z/YRlttd6NZIKjf1TD0nTrFnRvPdsZOlKYSArbfqNrHRBQkgirOD4JPI1tYKTffq
iOeZFx25K+fj2+0AJjvrbe4HDo5m+Q==
=umI8
-----END PGP PUBLIC KEY BLOCK-----
```

Per revocare effettivamente la tua chiave privata, hai ora bisogno di unire il certificato con la chiave, cosa che si fa importando il file del certificato di revoca nel tuo portachiavi:

```
sonya@debian:~/.gnupg$ gpg --import revocation_file.asc
gpg: key 9E570DEB22C27871: "sonya <sonya@debian>" revocation certificate imported
gpg: Total number processed: 1
gpg:    new key revocations: 1
gpg: marginals needed: 3  completes needed: 1  trust model: pgp
gpg: depth: 0  valid:   1  signed:   0  trust: 0-, 0q, 0n, 0m, 0f, 1u
gpg: next trustdb check due at 2022-07-04
```

Elenca ora le tue chiavi e sarai informato della tua chiave revocata:

```
sonya@debian:~/.gnupg$ gpg --list-keys
/home/sonya/.gnupg/pubring.kbx
pub   rsa3072 2020-07-04 [SC] [revoked: 2020-07-04]
      8885637C39E7A627858B4C2F9E570DEB22C27871
uid           [ revoked] sonya <sonya@debian>
```

Infine, ma non meno importante, assicurati di rendere la chiave revocata disponibile a qualsiasi parte che abbia chiavi pubbliche associate a essa (inclusi i keyserver).

Usare GPG per Criptare, Cecriptare, Firmare e Verificare i File

Nella sezione precedente, `carol` ha inviato la sua chiave pubblica a `ina`. La useremo ora per discutere come GPG può criptare, decriptare, firmare e verificare i file.

Criptare e Decriptare i File

Per prima cosa, `ina` deve importare la chiave pubblica di `carol` (`carol.pub.key`) nel suo portachiavi in modo che possa iniziare a lavorare con essa:

```
ina@halof:~> gpg --import carol.pub.key
gpg: /home/ina/.gnupg/trustdb.gpg: trustdb created
```

```
gpg: key 19BBEFD16813034E: public key "carol <carol@debian>" imported
gpg: Total number processed: 1
gpg:               imported: 1
ina@halof:~> gpg --list-keys
/home/ina/.gnupg/pubring.kbx
----------------------------
pub   rsa3072 2020-07-03 [SC] [expires: 2022-07-03]
      D18FA0021F644CDAF57FD0F919BBEFD16813034E
uid           [ unknown] carol <carol@debian>
sub   rsa3072 2020-07-03 [E] [expires: 2022-07-03]
```

Poi creerai un file scrivendoci dentro del testo e poi lo cripterai usando `gpg` (poiché non hai firmato la chiave di `carol`, ti verrà chiesto esplicitamente se vuoi usare quella chiave):

```
ina@halof:~> echo "This is the message ..." > unencrypted-message
ina@halof:~> gpg --output encrypted-message --recipient carol --armor --encrypt unencrypted-
message
gpg: 0227347CC92A5CB1: There is no assurance this key belongs to the named user
sub  rsa3072/0227347CC92A5CB1 2020-07-03 carol <carol@debian>
 Primary key fingerprint: D18F A002 1F64 4CDA F57F  D0F9 19BB EFD1 6813 034E
      Subkey fingerprint: 9D89 1BF9 39A4 C130 E44B  1135 0227 347C C92A 5CB1

NON è certo che la chiave appartenga alla persona nominata
nell'ID utente.  Se sapete veramente cosa state facendo,
potete rispondere alla prossima domanda con sì.

Usare comunque questa chiave? (y/N) y
```

Analizziamo il comando `gpg`:

`--output encrypted-message`

Specificazione del nome del file per la versione criptata del file originale (`encrypted-message` nell'esempio).

`--recipient carol`

Specificazione dello `USER-ID` del destinatario (`carol` nel nostro esempio). Se non viene fornito, GnuPG lo richiederà (a meno che non sia specificato `--default-recipient`).

`--armor`

Questa opzione produce un output ASCII "blindato", che può essere copiato in una email.

`--encrypt unencrypted-message`

Specificazione del nome del file originale da criptare.

Ora puoi inviare il messaggio criptato a Carol su Debian usando Scp:

```
ina@halof:~> scp encrypted-message carol@debian:/home/carol/
carol@debian's password:
encrypted-message                                                    100%  736
1.8MB/s   00:00
```

Se ora ti logghi come `carol` e provi a leggere il `messaggio criptato`, troverai che è effettivamente criptato e quindi illeggibile:

```
carol@debian:~$ cat encrypted-message
-----BEGIN PGP MESSAGE-----

hQGMAwInNHzJKlyxAQv/brJ8Ubs/xya35sbv6kdRKm1C7ONLxL3OueWA4mCs0Y/P
GBna6ZEUCrMEgl/rCyByj3Yq74kuiTmzxAIRUDdvHfj0TtrOWjVAqIn/fPSfMkjk
dTxKo1i55tLJ+sj17dGMZDcNBinBTP4U1atuN71A5w7vH+XpcesRcFQLKiSOmYTt
F7SN3/5x5J6io4ISn+b0KbJgiJNNx+Ne/ub4Uzk4NlK7tmBklyC1VRualtxcG7R9
1klBPYSld6fTdDwT1Y4MofpyILAiGMZvUR1RXauEKf7OIzwC5gWU+UQPSgeCdKQu
X7QL0ZIBS0Ug2XKrO1k93lmDjf8PWsRIml6n/hNelaOBA3HMP0b6Ozv1gFeEsFvC
IxhUYPb+rfuNFTMEB7xIO94AAmWB9N4qknMxdDqNE8WhA728Plw6y8L2ngsplY15
MR4lIFDpljA/CcVh4BXVe9j0TdFWDUkrFMfaIfcPQwKLXEYJp19XYIaaEazkOs5D
W4pENN0YOcX0KWyAYX6r0l8BF0rq/HMenQwqAVXMG3s8ATuUOeqjBbR1x1qCvRQP
CR/3V73aQwc2j5ioQmhWYpqxiro0yKX2Ar/E6rZyJtJYrq+CUk8O3JoBaudknNFj
pwuRwF1amwnSZ/MZ/9kMKQ==
=g1jw
-----END PGP MESSAGE-----
```

Tuttavia, poiché sei in possesso della chiave privata, puoi facilmente decifrare il messaggio passando a `gpg` l'opzione `--decrypt` seguita dal percorso del file cifrato (sarà richiesta la passphrase della chiave privata):

```
carol@debian:~$ gpg --decrypt encrypted-message
gpg: encrypted with 3072-bit RSA key, ID 0227347CC92A5CB1, created 2020-07-03
      "carol <carol@debian>"
This is the message ...
```

Puoi anche specificare l'opzione `--output` per salvare il messaggio in un nuovo file non criptato:

```
carol@debian:~$ gpg --output unencrypted-message --decrypt encrypted-message
gpg: encrypted with 3072-bit RSA key, ID 0227347CC92A5CB1, created 2020-07-03
      "carol <carol@debian>"
carol@debian:~$ cat unencrypted-message
This is the message ...
```

Firmare e Verificare i File

Oltre a criptare, GPG può anche essere usato per firmare i file. L'opzione `--sign` è rilevante in questo caso. Iniziamo creando un nuovo messaggio (`message`) e firmiamolo con l'opzione `--sign` (sarà richiesta la passphrase della tua chiave privata):

```
carol@debian:~$ echo "This is the message to sign ..." > message
carol@debian:~$ gpg --output message.sig --sign message
(...)
```

Spiegazione del comando gpg:

`--output message`
: nome del file della versione firmata (`message.sig` nel nostro esempio).

`--sign message`
: Percorso del file originale.

NOTE Usando `--sign` il documento viene compresso e poi firmato. L'output è in formato binario.

Poi trasferiremo il file a `ina` su `halof` usando `scp message.sig ina@halof:/home/ina` . Tornati come `ina` su `halof`, puoi ora verificarlo usando l'opzione `--verify`:

```
ina@halof:~> gpg --verify message.sig
gpg: Signature made Sat 04 jul 2020 14:34:41 CEST
gpg:                using RSA key D18FA0021F644CDAF57FD0F919BBEFD16813034E
gpg: Good signature from "carol <carol@debian>" [unknown]
(...)
```

Se vuoi anche leggere il file, devi decriptarlo in un nuovo file (`message` nel nostro caso) usando l'opzione `--output`:

```
ina@halof:~> gpg --output message --decrypt message.sig
gpg: Signature made Sat 04 jul 2020 14:34:41 CEST
gpg:                using RSA key D18FA0021F644CDAF57FD0F919BBEFD16813034E
gpg: Good signature from "carol <carol@debian>" [unknown]
gpg: WARNING: This key is not certified with a trusted signature!
gpg:          There is no indication that the signature belongs to the owner.
Primary key fingerprint: D18F A002 1F64 4CDA F57F  D0F9 19BB EFD1 6813 034E
ina@halof:~> cat message
This is the message to sign ...
```

GPG-Agent

Completeremo questa lezione trattando brevemente `gpg-agent`. `gpg-agent` è il demone che gestisce le chiavi private per GPG (è avviato su richiesta da `gpg`). Per vedere un riassunto delle opzioni più utili, esegui `gpg-agent --help` o `gpg-agent -h`:

```
carol@debian:~$ gpg-agent --help
gpg-agent (GnuPG) 2.2.4
libgcrypt 1.8.1
Copyright (C) 2017 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <https://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.

Syntax: gpg-agent [options] [command [args]]
Secret key management for GnuPG

Options:

     --daemon                      run in daemon mode (background)
     --server                      run in server mode (foreground)
     --supervised                  run in supervised mode
 -v, --verbose                     verbose
 -q, --quiet                       be somewhat more quiet
 -s, --sh                          sh-style command output
 -c, --csh                         csh-style command output
(...)
```

NOTE Per maggiori informazioni, consultare la pagina man di `gpg-agent`.

Esercizi Guidati

1. Completa la tabella fornendo il nome corretto del file:

Descrizione	**Nome del file**
Database di fiducia	
Directory per i certificati di revoca	
Directory per le chiavi private	
Portachiavi chiavi pubbliche	

2. Rispondi alle seguenti domande:
   - Che tipo di crittografia è usata da *GnuPG*?
   - Quali sono i due componenti principali della crittografia a chiave pubblica?
   - Qual è il `KEY-ID` dell'impronta della chiave pubblica `07A6 5898 2D3A F3DD 43E3 DA95 1F3F 3147 FA7F 54C7`?
   - Quale metodo si usa per distribuire le chiavi pubbliche a livello globale?

3. Metti i seguenti passi nel giusto ordine per quanto riguarda la revoca della chiave privata:
   - Rendi la chiave revocata disponibile ai tuoi corrispondenti.
   - Crea un certificato di revoca.
   - Importa il certificato di revoca nel tuo portachiavi.

   L'ordine corretto è:

Passo 1:	
Passo 2:	
Passo 3:	

4. Per quanto riguarda la crittografia dei file, cosa implica l'opzione `--armor` nel comando `gpg --output encrypted-message --recipient carol --armor --encrypt unencrypted-`

`message`?

Esercizi Esplorativi

1. La maggior parte delle opzioni `gpg` hanno sia una versione lunga si una corta. Completa la tabella con la corrispondente versione breve:

Versione lunga	**Versione breve**
`--armor`	
`--output`	
`--recipient`	
`--decrypt`	
`--encrypt`	
`--sign`	

2. Rispondi alle seguenti domande riguardanti l'esportazione di una chiave:
   - Quale comando useresti per esportare tutte le tue chiavi pubbliche in un file chiamato `all.key`?
   - Quale comando useresti per esportare tutte le tue chiavi private in un file chiamato `all_private.key`?

3. Quale opzione `gpg` permette di eseguire la maggior parte dei compiti relativi alla gestione delle chiavi presentandovi un menu?

4. Quale opzione `gpg` ti permette di fare una firma in chiaro?

Sommario

Questa lezione ha trattato il *GNU Privacy Guard*, una scelta eccellente per criptare/decriptare e firmare/verificare digitalmente i file. Hai imparato:

- come generare una coppia di chiavi.
- come elencare le chiavi nel portachiavi.
- il contenuto della directory `~/.gnupg`.
- cosa sono `USER-ID` e `KEY-ID`.
- come distribuire le chiavi pubbliche ai vostri corrispondenti.
- come distribuire globalmente le chiavi pubbliche attraverso i keyserver.
- come revocare le chiavi private.
- come criptare e decriptare i file.
- come firmare e verificare i file.
- le basi del *GPG-Agent*.

I seguenti comandi sono stati discussi in questa lezione:

`gpg`
: Lo strumento di crittografia e firma *OpenPGP*.

Risposte agli Esercizi Guidati

1. Completa la tabella fornendo il nome corretto del file:

Descrizione	**Nome del file**
Database di fiducia	`trustdb.gpg`
Directory per i certificati di revoca	`opengp-revocs.d`
Directory per le chiavi private	`private-keys-v1.d`
Portachiavi chiavi pubbliche	`pubring.kbx`

2. Rispondi alle seguenti domande:
   - Che tipo di crittografia è usata da *GnuPG*?

     Crittografia a chiave pubblica o crittografia asimmetrica.

   - Quali sono i due componenti principali della crittografia a chiave pubblica?

     Le chiavi pubbliche e private.

   - Qual è il `KEY-ID` dell'impronta della chiave pubblica `07A6 5898 2D3A F3DD 43E3 DA95 1F3F 3147 FA7F 54C7`?

     `FA7F 54C7`

   - Quale metodo si usa per distribuire le chiavi pubbliche a livello globale?

     I key server.

3. Metti i seguenti passi nel giusto ordine per quanto riguarda la revoca della chiave privata:
   - Rendi la chiave revocata disponibile ai tuoi corrispondenti.
   - Crea un certificato di revoca.
   - Importa il certificato di revoca nel tuo portachiavi.

     L'ordine corretto è:

Passo 1:	Crea un certificato di revoca.
Passo 2:	Importa il certificato di revoca nel tuo portachiavi.

Passo 3:	Rendi la chiave revocata disponibile ai tuoi corrispondenti.

4. Per quanto riguarda la crittografia dei file, cosa implica l'opzione `--armor` nel comando `gpg --output encrypted-message --recipient carol --armor --encrypt unencrypted-message`?

   Produce un output ASCII "blindato ", che permette di copiare il file crittografato risultante in una e-mail.

Risposte agli Esercizi Esplorativi

1. La maggior parte delle opzioni `gpg` hanno sia una versione lunga sia una corta. Completa la tabella con la corrispondente versione breve:

Versione lunga	Versione breve
`--armor`	`-a`
`--output`	`-o`
`--recipient`	`-r`
`--decrypt`	`-d`
`--encrypt`	`-e`
`--sign`	`-s`

2. Rispondi alle seguenti domande riguardanti l'esportazione di una chiave:
   - Quale comando useresti per esportare tutte le tue chiavi pubbliche in un file chiamato `all.key`?

     `gpg --export --output all.key` o `gpg --export -o all.key`

   - Quale comando useresti per esportare tutte le tue chiavi private in un file chiamato `all_private.key`?

     `gpg --export-secret-keys --output all_private.key` o `gpg --export-secret-keys -o all_private.key` (`--export-secret-keys` può essere sostituito da `--export-secret-subkeys` con un risultato leggermente diverso — controlla `man pgp` per maggiori informazioni).

3. Quale opzione `gpg` permette di eseguire la maggior parte dei compiti relativi alla gestione delle chiavi presentandovi un menu?

   `--edit-key`

4. Quale opzione `gpg` ti permette di fare una firma in chiaro?

   `--clearsign`

Imprint

© 2023 by Linux Professional Institute: Learning Materials, “LPIC-1 (102) (Versione 5.0)”.

PDF generato: 2023-07-13

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 Istitute (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.