

Linux

i obsługa sieci dla profesjonalistów

Konfiguracja i stosowanie
bezpiecznych usług sieciowych



Rob VandenBrink

Helion 



Tytuł oryginału: Linux for Networking Professionals: Securely configure and operate Linux network services for the enterprise

Tłumaczenie: Grzegorz Werner

ISBN: 978-83-283-9710-1

Copyright © Packt Publishing 2021. First published in the English language under the title 'Linux for Networking Professionals – (9781800202399)'.

Polish edition copyright © 2023 by Helion S.A.

All rights reserved. No part of this book may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording or by any information storage retrieval system, without permission from the Publisher.

Wszelkie prawa zastrzeżone. Nieautoryzowane rozpowszechnianie całości lub fragmentu niniejszej publikacji w jakiegokolwiek postaci jest zabronione. Wykonywanie kopii metodą kserograficzną, fotograficzną, a także kopiowanie książki na nośniku filmowym, magnetycznym lub innym powoduje naruszenie praw autorskich niniejszej publikacji.

Wszystkie znaki występujące w tekście są zastrzeżonymi znakami firmowymi bądź towarowymi ich właścicieli.

Autor oraz wydawca dołożyli wszelkich starań, by zawarte w tej książce informacje były kompletne i rzetelne. Nie biorą jednak żadnej odpowiedzialności ani za ich wykorzystanie, ani za związane z tym ewentualne naruszenie praw patentowych lub autorskich. Autor oraz wydawca nie ponoszą również żadnej odpowiedzialności za ewentualne szkody wynikłe z wykorzystania informacji zawartych w książce.

Drogi Czytelniku!

Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres

<https://helion.pl/user/opinie/linobs>

Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

Helion S.A.

ul. Kościuszki 1c, 44-100 Gliwice

tel. 32 230 98 63

e-mail: helion@helion.pl

WWW: <https://helion.pl> (księgarnia internetowa, katalog książek)

Printed in Poland.

- [Kup książkę](#)
- [Poleć książkę](#)
- [Oceń książkę](#)

- [Księgarnia internetowa](#)
- [Lubię to! » Nasza społeczność](#)

Spis treści

O autorze	13
O recenzencie	14
Przedmowa	15
Część I. Podstawy Linuksa	23
Rozdział 1. Witamy w rodzinie Linuksa	25
Dlaczego Linux jest dobrym wyborem dla zespołu ds. sieci?	26
Dlaczego Linux jest ważny?	27
Historia Linuksa	28
Popularne odmiany Linuksa dla centrów danych	29
Red Hat	30
Oracle/Scientific Linux	30
SUSE	30
Ubuntu	31
BSD/FreeBSD/OpenBSD	31
Wyspecjalizowane dystrybucje Linuksa	32
Zapory open source	32
Kali Linux	32
SIFT	33
Security Onion	33
Wirtualizacja	33
Linux i przetwarzanie danych w chmurze	33
Wybieranie dystrybucji Linuksa dla swojej organizacji	34
Podsumowanie	35
Dalsza lektura	36

Rozdział 2. Podstawowa konfiguracja i obsługa sieci w Linuksie	
— praca z interfejsami lokalnymi	37
Wymagania techniczne	38
Praca z ustawieniami sieci — dwa zbiory poleceń	38
Wyświetlanie informacji o interfejsie IP	40
Wyświetlanie informacji o trasach	43
Adresy IPv4 i maski podsieci	44
Adresy specjalnego przeznaczenia	45
Adresy prywatne — RFC 1918	48
Przypisywanie adresu IP do interfejsu	48
Dodawanie trasy	50
Dodawanie trasy tradycyjnym sposobem	51
Wyłączanie i włączanie interfejsu	52
Ustawianie jednostki MTU interfejsu	52
Więcej o poleceniu nmcli	53
Podsumowanie	55
Pytania	55
Dalsza lektura	56

Część II. Linux jako węzeł sieciowy i platforma diagnostyczna **57**

Rozdział 3. Używanie Linuksa i linuksowych narzędzi do diagnostyki sieci	59
Wymagania techniczne	60
Podstawy sieci — model OSI	61
Warstwa 2. — kojarzenie adresów IP i MAC za pomocą protokołu ARP	63
Wartości OUI adresów MAC	68
Warstwa 4. — jak działają porty TCP i UDP?	69
Warstwa 4. — TCP i potrójne uzgodnienie	70
Wylizanie portów lokalnych — do czego jestem podłączony? Czego nasłuchuję?	72
Wylizanie portów zdalnych za pomocą natywnych narzędzi	80
Wylizanie zdalnych portów i usług — Nmap	86
Skrypty Nmap	91
Czy Nmap ma jakieś ograniczenia?	97
Diagnozowanie łączności bezprzewodowej	98
Podsumowanie	103
Pytania	104
Dalsza lektura	104
Rozdział 4. Zapora Linuksa	105
Wymagania techniczne	106
Konfigurowanie zapory iptables	106
Ogólny opis zapory iptables	107
Tabela NAT	111
Tabela mangle	113
Kolejność operacji w iptables	114

Konfigurowanie zapory nftables	115
Podstawowa konfiguracja nftables	117
Używanie plików include	118
Usuwanie konfiguracji zapory	119
Podsumowanie	119
Pytania	120
Dalsza lektura	120
Rozdział 5. Standardy bezpieczeństwa Linuksa na praktycznych przykładach	121
Wymagania techniczne	122
Dlaczego trzeba zabezpieczać hosty linuxowe?	122
Kwestie bezpieczeństwa specyficzne dla chmury	123
Często spotykane branżowe standardy bezpieczeństwa	124
Krytyczne środki kontroli Center for Internet Security	125
Krytyczne środki kontroli CIS nr 1 i 2 — pierwsze kroki	130
OSQuery — krytyczne środki kontroli nr 1 i 2 uzupełnione o nr 10 i 17	136
Wzorce Center for Internet Security	140
Stosowanie wzorca CIS — zabezpieczanie SSH w Linuksie	141
SELinux i AppArmor	147
Podsumowanie	149
Pytania	149
Dalsza lektura	150
Część III. Usługi sieciowe w Linuksie	151
Rozdział 6. Usługi DNS w Linuksie	153
Wymagania techniczne	154
Co to jest DNS?	154
Dwa główne zastosowania serwera DNS	154
„Wewnętrzny” serwer DNS organizacji (i przegląd systemu DNS)	155
Internetowy serwer DNS	158
Często używane implementacje DNS	160
Podstawowa instalacja: BIND do użytku wewnętrznego	160
BIND: implementacja do użytku internetowego	163
Diagnostowanie i rekonesans DNS	165
DoH	166
DoT	168
knot-dnsutils	170
Implementacja DoT w Nmap	172
DNSSEC	172
Podsumowanie	174
Pytania	174
Dalsza lektura	175

Rozdział 7. Usługi DHCP w Linuksie	177
Jak działa DHCP?	177
Podstawowe działanie DHCP	177
Żądania DHCP z innych podsieci (przełączniki DHCP)	179
Opcje DHCP	181
Zabezpieczanie usług DHCP	182
Nieautoryzowany serwer DHCP	183
Nieautoryzowany klient DHCP	186
Instalowanie i konfigurowanie serwera DHCP	186
Podstawowa konfiguracja	187
Rezerwacje statyczne	189
Proste rejestrowanie zdarzeń i diagnozowanie DHCP	190
Podsumowanie	191
Pytania	191
Dalsza lektura	192
Rozdział 8. Usługi certyfikatów w Linuksie	193
Wymagania techniczne	194
Czym są certyfikaty?	194
Pozyskiwanie certyfikatu	195
Używanie certyfikatu na przykładzie serwera WWW	197
Budowanie prywatnego urzędu certyfikacji	201
Budowanie urzędu CA z wykorzystaniem OpenSSL	202
Tworzenie i podpisywanie wniosku CSR	204
Zabezpieczanie infrastruktury urzędu certyfikacji	206
Tradycyjna, wypróbowana rada	206
Współczesna rada	207
Zagrożenia specyficzne dla urzędów CA działających w nowoczesnych infrastrukturach	208
Transparentność certyfikatów	208
Używanie usług CT do inwentaryzacji lub rekonesansu	209
Automatyzacja zarządzania certyfikatami i protokół ACME	211
Ściągawka z OpenSSL	212
Podsumowanie	214
Pytania	215
Dalsza lektura	215
Rozdział 9. Usługi RADIUS w Linuksie	216
Wymagania techniczne	216
Podstawy protokołu RADIUS — czym jest i jak działa?	217
Wdrażanie usług RADIUS z uwierzytelnianiem lokalnym	221
Usługi RADIUS z zapleczem LDAP/LDAPS	223
Uwierzytelnianie NTLM (AD) — wprowadzenie do CHAP	227
Unlang — niejęzyk	232
Zastosowania usług RADIUS	234
Uwierzytelnianie VPN za pomocą identyfikatora użytkownika i hasła	234
Dostęp administracyjny do urządzeń sieciowych	235

Dostęp administracyjny do routerów i przełączników	236
Konfigurowanie serwera RADIUS pod kątem uwierzytelniania EAP-TLS	237
Uwierzytelnianie w sieci bezprzewodowej za pomocą 802.1x/EAP-TLS	239
Uwierzytelnianie w sieci przewodowej za pomocą 802.1x/EAP-TLS	241
Używanie usługi Google Authenticator do uwierzytelniania MFA z wykorzystaniem serwera RADIUS	244
Podsumowanie	245
Pytania	245
Dalsza lektura	246
Rozdział 10. Usługi równoważenia obciążenia w Linuksie	248
Wymagania techniczne	249
Wprowadzenie do równoważenia obciążenia	249
Round Robin DNS (RRDNS)	249
Serwer proxy dla ruchu przychodzącego	
— równoważenie obciążenia w warstwie 7.	251
Translacja NAT połączeń przychodzących	
— równoważenie obciążenia w warstwie 4.	253
Równoważenie obciążenia z użyciem DSR	255
Algorytmy równoważenia obciążenia	257
Sprawdzanie kondycji serwerów i usług	258
Usługi równoważenia obciążenia w centrum danych — kwestie projektowe	259
Sieć w centrum danych a kwestie zarządzania	262
Budowanie usług równoważenia obciążenia typu NAT/proxy za pomocą HAProxy	266
Przed przystąpieniem do konfiguracji — karty sieciowe, adresowanie i routing	266
Przed przystąpieniem do konfiguracji — dostrajanie wydajności	267
Równoważenie obciążenia usług TCP — usługi WWW	269
Konfigurowanie trwałych połączeń	272
Nota wdrożeniowa	273
Przetwarzanie HTTPS na froncie	273
Końcowe uwagi dotyczące bezpieczeństwa usług równoważenia obciążenia	275
Podsumowanie	277
Pytania	277
Dalsza lektura	278
Rozdział 11. Przechwytywanie i analiza pakietów w Linuksie	279
Wymagania techniczne	280
Wprowadzenie do przechwytywania pakietów — miejsca, w których należy szukać	280
Przechwytywanie po jednej ze stron	280
Użycie portu monitorowania	281
Pośredni host na ścieżce ruchu	282
Podstuch sieciowy	283
Złośliwe sposoby przechwytywania pakietów	284
Kwestie wydajnościowe podczas przechwytywania	287
Narzędzia do przechwytywania	289
tcpdump	289
Wireshark	289

TShark	289
Inne narzędzia PCAP	289
Filtrowanie przechwytywanego ruchu	290
Filtry przechwytywania w programie Wireshark (przechwytywanie ruchu w sieci domowej)	290
Filtry przechwytywania tcpdump — telefonyVoIP i DHCP	292
Więcej filtrów przechwytywania — LLDP i CDP	297
Pozyskiwanie plików z przechwyconych pakietów	300
Diagnozowanie aplikacji — przechwytywanie połączenia telefonicznego VoIP	302
Filtry wyświetlania w programie Wireshark — wyodrębnianie konkretnych danych	306
Podsumowanie	309
Pytania	309
Dalsza lektura	310
Rozdział 12. Monitorowanie sieci z wykorzystaniem Linuksa	311
Wymagania techniczne	312
Rejestrowanie zdarzeń z wykorzystaniem usługi syslog	312
Rozmiar dziennika, rotacja i bazy danych	313
Analiza dzienników — znajdowanie „tego czegoś”	314
Alarmy dotyczące konkretnych zdarzeń	316
Przykładowy serwer syslog	319
Projekt Dshield	324
Zarządzanie urządzeniami sieciowymi za pomocą SNMP	327
Podstawowe zapytania SNMP	327
Przykład wdrożenia systemu SNMP NMS — LibreNMS	331
SNMPv3	335
Gromadzenie danych NetFlow w Linuksie	346
Co to jest NetFlow i jego „kuzyni” — SFLOW, J-Flow i IPFIX?	346
Koncepcje wdrożeniowe związane z gromadzeniem informacji o przepływach	347
Konfigurowanie routera lub przełącznika do gromadzenia informacji o przepływach	349
Przykładowy serwer NetFlow wykorzystujący NFDump i NFSen	351
Podsumowanie	361
Pytania	362
Dalsza lektura	362
Często używane identyfikatory SNMP OID	364
Rozdział 13. Systemy zapobiegania włamaniom w Linuksie	365
Wymagania techniczne	366
Co to jest IPS?	366
Opcje architektoniczne — umiejscowienie systemu IPS w centrum danych	367
Techniki unikania systemów IPS	372
Wykrywanie rozwiązań WAF	372
Fragmentacja i inne techniki unikania systemów IPS	373
Klasyczne/sieciowe rozwiązania IPS — Snort i Suricata	375
Przykładowy system IPS Suricata	376
Konstruowanie reguły IPS	385

Pasywne monitorowanie ruchu	389
Monitorowanie pasywne za pomocą POF — przykład	389
Przykład użycia monitora Zeek — gromadzenie metadanych dotyczących sieci	391
Podsumowanie	400
Pytania	401
Dalsza lektura	401
 Rozdział 14. Usługi honeypot w Linuksie	 402
Wymagania techniczne	402
Przegląd usług honeypot — co to jest honeypot i do czego może się przydać?	403
Architektura i scenariusze wdrożeniowe — gdzie umieścić honeypota?	405
Zagrożenia związane z wdrażaniem honeypotów	408
Przykładowe honeypoty	409
Podstawowe honeypoty alarmujące o próbach połączenia z portem — iptables, netcat i portspooof	410
Inne często używane honeypoty	413
Honeypot rozproszony/społecznościowy — projekt DShield prowadzony przez Internet Storm Center	414
Podsumowanie	425
Pytania	426
Dalsza lektura	426
 Sprawdziany wiadomości	 427
Rozdział 2., „Podstawowa konfiguracja i obsługa sieci w Linuksie — praca z interfejsami lokalnymi”	427
Rozdział 3., „Używanie Linuksa i linuksowych narzędzi do diagnostyki sieci”	428
Rozdział 4., „Zapora Linuksa”	430
Rozdział 5., „Standardy bezpieczeństwa Linuksa na praktycznych przykładach”	431
Rozdział 6., „Usługi DNS w Linuksie”	431
Rozdział 7., „Usługi DHCP w Linuksie”	432
Rozdział 8., „Usługi certyfikatów w Linuksie”	435
Rozdział 9., „Usługi RADIUS w Linuksie”	436
Rozdział 10., „Usługi równoważenia obciążenia w Linuksie”	438
Rozdział 11., „Przechwytywanie i analiza pakietów w Linuksie”	438
Rozdział 12., „Monitorowanie sieci z wykorzystaniem Linuksa”	439
Rozdział 13., „Systemy zapobiegania włamaniom w Linuksie”	441
Rozdział 14., „Usługi honeypot w Linuksie”	442
 Skorowidz	 443

Podstawowa konfiguracja i obsługa sieci w Linuksie — praca z interfejsami lokalnymi

W tym rozdziale wyjaśnimy, jak wyświetlać oraz konfigurować lokalne interfejsy i trasy w linuksowym hoście. Omówimy zarówno nowe, jak i tradycyjne polecenia służące do wykonywania takich operacji jak wyświetlanie i modyfikowanie danych adresowych IP, lokalnych tras oraz innych parametrów interfejsu. Opiszemy też, jak konstruować adresy IP i adresy podsieci w formacie binarnym.

Niniejszy rozdział powinien dać Ci solidne podstawy do zrozumienia zagadnień omawianych dalej w tej książce, takich jak diagnozowanie problemów z siecią, „utwardzanie” hosta oraz instalowanie zabezpieczonych usług.

W tym rozdziale omówimy następujące tematy:

- praca z ustawieniami sieci — dwa zbiory poleceń,
- wyświetlanie informacji o interfejsie IP,
- adresy IPv4 i maski podsieci,
- przypisywanie adresu IP do interfejsu.

Wymagania techniczne

W tym i wszystkich pozostałych rozdziałach, kiedy będziemy omawiać różne polecenia, możesz wypróbować je na własnym komputerze. Polecenia w tej książce zilustrowano w systemie Ubuntu Linux 20 (wersja ze wsparciem długoterminowym — LTS), ale w większości powinny działać identycznie lub bardzo podobnie w dowolnej dystrybucji Linuksa.

Praca z ustawieniami sieci — dwa zbiory poleceń

Przez wiele lat `ifconfig` (konfiguracja interfejsu) i powiązane polecenia były jednym z filarów Linuksa, do tego stopnia, że choć obecnie są w większości dystrybucji uznawane za przestarzałe, to nadal pozostają zakodowane w pamięci mięśniowej wielu administratorów systemu i sieci.

Dlaczego zastąpiono te stare polecenia sieciowe? Z paru powodów. Nie obsługują dobrze części nowego sprzętu (zwłaszcza kart sieciowych InfiniBand), a ponadto z biegiem lat, w miarę jak jądro Linuksa się zmieniało, ich działanie było coraz bardziej niespójne. Jednak presja na zachowanie wstecznej kompatybilności utrudniała rozwiązanie tych problemów.

Stare polecenia znajdują się w pakiecie oprogramowania `net-tools`, a nowe w pakiecie `iproute2`. Nowi administratorzy powinni skupić się na nowych poleceniach, ale znajomość starszych nadal bywa przydatna. Często spotyka się wiekowe komputery działające pod kontrolą Linuksa, maszyny, które prawdopodobnie nigdy nie zostaną zaktualizowane, a nadal używają starych poleceń. Dlatego opiszemy oba zestawy narzędzi.

Lekcja, jaką można z tego wyciągnąć, jest taka, że w świecie Linuksa jedyną stałą rzeczą jest zmiana. Stare polecenia pozostają dostępne, ale nie są domyślnie instalowane.

Aby zainstalować tradycyjne polecenia, wydaj następującą instrukcję:

```
robv@ubuntu:~$ sudo apt install net-tools
[sudo] password for robv:
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following package was automatically installed and is no longer required:
  libfprint-2-tod1
Use 'sudo apt autoremove' to remove it.
The following NEW packages will be installed:
  net-tools
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 0 B/196 kB of archives.
After this operation, 864 kB of additional disk space will be used.
Selecting previously unselected package net-tools.
(Reading database ... 183312 files and directories currently installed.)
```

```
Preparing to unpack .../net-tools_1.60+git20180626.aebd88e-lubuntu1_amd64.deb ...
Unpacking net-tools (1.60+git20180626.aebd88e-lubuntu1) ...
Setting up net-tools (1.60+git20180626.aebd88e-lubuntu1) ...
Processing triggers for man-db (2.9.1-1) ...
```

Warto w tym poleceniu instalacyjnym i jego wynikach zwrócić uwagę na kilka rzeczy:

- **sudo**: użyto polecenia **sudo** — **sudo** zasadniczo oznacza „zrób jako superużytkownik” — więc powyższa instrukcja została wykonana z przywilejami użytkownika *root* (administratora). Wymaga to podania hasła użytkownika, który wydaje polecenie. Ponadto użytkownik musi być poprawnie wprowadzony do pliku konfiguracyjnego */etc/sudoers*. W większości dystrybucji identyfikator użytkownika określony podczas instalacji systemu operacyjnego jest automatycznie dołączany do tego pliku. Innych użytkowników lub grupy można dodać poleceniem *visudo*.

Dlaczego użyto **sudo**? Instalowanie oprogramowania, zmienianie parametrów sieci i inne operacje systemowe wymagają podniesionych uprawnień — byłoby niedobrze, gdyby w systemie korporacyjnym z wieloma użytkownikami osoby niebędące administratorami mogły dokonywać takich zmian.

Skoro więc **sudo** jest takie świetne, dlaczego nie uruchamiać wszystkich programów jako *root*? Oczywiście, wszystko będzie działać poprawnie, jeśli masz przywileje superużytkownika, ale ewentualne pomyłki lub literówki mogą mieć katastrofalne konsekwencje. Ponadto, jeśli pracujesz z pewnymi przywilejami i zdarzy Ci się uruchomić jakieś złośliwe oprogramowanie, będzie ono działać z tymi samymi przywilejami, co z pewnością nie jest pożądane! Jeśli ktoś zapyta, owszem, złośliwe oprogramowanie do Linuksa definitywnie istnieje i niestety towarzyszy systemowi niemal od pierwszych dni.

- **apt**: użyto polecenia **apt** — **apt** to skrót od **Advanced Package Tool**. Narzędzie to instaluje nie tylko wybrany pakiet, ale także inne pakiety, biblioteki lub zależności wymagane do działania tego pakietu. W dodatku domyślnie pobiera wszystkie te komponenty z internetowych repozytoriów. Jest to rozwiązanie zdecydowanie szybkie i wygodne w porównaniu ze starym procesem, który wymagał zgromadzenia wszystkich zależności (w odpowiednich wersjach), a następnie zainstalowania ich we właściwej kolejności w celu uruchomienia jakiegokolwiek nowej funkcji.

Program **apt** jest domyślnym instalatorem w Ubuntu, Debianie i powiązanych dystrybucjach, ale aplikacje do zarządzania pakietami różnią się w zależności od dystrybucji. Oprócz **apt** i odpowiedników obsługiwane jest również instalowanie z pobranych plików. Debian, Ubuntu i powiązane dystrybucje używają plików *deb*, podczas gdy wiele innych dystrybucji używa plików *rpm*. Podsumowano to w poniższej tabeli:

System operacyjny	Format pliku	Narzędzia instalacyjne
Debian	<i>.deb</i>	apt, apt-cache, apt-get, dpkg
Ubuntu	<i>.deb</i>	apt, apt-cache, apt-get, dpkg
Red Hat/CentOS	<i>.rpm</i>	yum, rpm
SUSE	<i>.rpm</i>	zypper, rpm

Zatem teraz, kiedy mamy mnóstwo nowych poleceń, którym chcielibyśmy przyjrzeć się bliżej, jak możemy uzyskać więcej informacji na ich temat? Polecenie `man` (od *manual* — podręcznik) wyświetla dokumentację większości programów i operacji w Linuksie. Stronę `man` programu `apt` można na przykład wyświetlić poleceniem `man apt`; wynik wygląda tak jak na rysunku 2.1.

```
APT(8)                                APT                                APT(8)

NAME
    apt - command-line interface

SYNOPSIS
    apt [-h] [-o=config_string] [-c=config_file] [-t=target_release]
        [-a=architecture] {list | search | show | update |
        install pkg [{=pkg version number | /target_release}]... | remove pkg... |
        upgrade | full-upgrade | edit-sources | {-v | --version} | {-h | --help}}
```

DESCRIPTION

apt provides a high-level commandline interface for the package management system. It is intended as an end user interface and enables some options better suited for interactive usage by default compared to more specialized APT tools like `apt-get(8)` and `apt-cache(8)`.

Much like `apt` itself, its manpage is intended as an end user interface and as such only mentions the most used commands and options partly to not duplicate information in multiple places and partly to avoid overwhelming readers with a cornucopia of options and details.

update (apt-get(8))

Manual page apt(8) line 1 (press h for help or q to quit)

Rysunek 2.1. Strona `man` programu `apt`

Kiedy będziemy przedstawiać nowe polecenia, poświęć kilka chwil na sprawdzenie ich za pomocą polecenia `man` — książka ta ma służyć Ci jako przewodnik, nie jako zamiennik rzeczywistej dokumentacji systemu operacyjnego.

Skoro wspomnieliśmy już o współczesnych i tradycyjnych narzędziach, a następnie zainstalowaliśmy starszy pakiet `net-tools`, wyjaśnijmy, czym są te polecenia i do czego służą.

Wyświetlanie informacji o interfejsie IP

Wyświetlanie informacji o interfejsie to zadanie, które często wykonuje się w linuksowej stacji roboczej, zwłaszcza jeśli jej karta sieciowa jest konfigurowana automatycznie, na przykład z wykorzystaniem protokołu **DHCP** (ang. *Dynamic Host Configuration Protocol*) lub autokonfiguracji IPv6.

Jak już wspomniano, istnieją dwa przeznaczone do tego zbiory poleceń. Polecenie `ip` pozwala wyświetlić lub skonfigurować parametry sieci w nowych systemach operacyjnych. W starszych wersjach służy do tego polecenie `ifconfig`.

Polecenie `ip` umożliwia wyświetlanie lub aktualizowanie adresów IP, tras oraz innych informacji sieciowych. Na przykład w celu wyświetlenia informacji o bieżącym adresie IP należy wydać polecenie:

```
ip address
```

Program `ip` obsługuje **uzupełnianie argumentów**, więc te same wyniki da polecenie `ip addr`, a nawet `ip a`:

```
robv@ubuntu:~$ ip ad
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default
↳ qlen 1000
   link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
   inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
   inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group
↳ default qlen 1000
   link/ether 00:0c:29:33:2d:05 brd ff:ff:ff:ff:ff:ff
   inet 192.168.122.182/24 brd 192.168.122.255 scope global dynamic noprefixroute
       ↳ ens33
       valid_lft 6594sec preferred_lft 6594sec
   inet6 fe80::led6:5b7f:5106:1509/64 scope link noprefixroute
       valid_lft forever preferred_lft forever
```

Jak widać, nawet najprostsze polecenie czasem zwraca znacznie więcej informacji, niż byś chciał. Na przykład wyświetlane są informacje zarówno o protokole **IP w wersji 4. (IPv4)**, jak i IPv6 — możemy ograniczyć je tylko do wersji 4. lub 6. przez dodanie opcji `-4` albo `-6`:

```
robv@ubuntu:~$ ip -4 ad
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default
↳ qlen 1000
   inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
2: ens33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group
↳ default qlen 1000
   inet 192.168.122.182/24 brd 192.168.122.255 scope global dynamic noprefixroute
       ↳ ens33
       valid_lft 6386sec preferred_lft 6386sec
```

W wynikach tych widać, że interfejs `lo` (logiczny, wewnętrzny interfejs pętli zwrotnej) ma adres IP `127.0.0.1`, a interfejs ethernetowy `ens33` ma adres IP `192.168.122.182`.

Teraz jest doskonały moment, żeby wpisać `man ip` i przejrzeć różne operacje, które można wykonać za pomocą tego polecenia (patrz rysunek 2.2).

robv@ubuntu:~/bin

IP (8)

Linux

- □ ×

IP (8)

NAME

ip - show / manipulate routing, network devices, interfaces and tunnels

SYNOPSIS

ip [OPTIONS] OBJECT { COMMAND | help }ip [-force] -batch filename

OBJECT := { link | address | addrlabel | route | rule | neigh | ntable
 | tunnel | tuntap | maddress | mroute | mrule | monitor | xfrm
 | netns | l2tp | tcp_metrics | token | macsec }

OPTIONS := { -V[ersion] | -h[uman-readable] | -s[tatistics] |
 -d[etails] | -r[esolve] | -iec | -f[amily] { inet | inet6 |
 link } | -4 | -6 | -I | -D | -B | -0 | -l[oops] { maximum-addr-
 flush-attempts } | -o[neline] | -rc[vbuf] [size] | -t[imestamp]
 | -ts[hort] | -n[etns] name | -N[umeric] | -a[ll] | -c[olor] |
 -br[ief] | -j[son] | -p[retty] }

OPTIONS

-V, -Version

Rysunek 2.2. Strona man programu ip

Polecenie `ifconfig` ma funkcje bardzo podobne do polecenia `ip`, ale, jak wspomniano, zwykle spotyka się je w starszych wersjach Linuksa. Tradycyjne polecenia rosły organicznie, a nowe funkcje „doczepiano” do nich w miarę potrzeb. Doprowadziło to do sytuacji, w której im bardziej skomplikowane rzeczy wyświetla się lub konfiguruje, tym mniej spójna staje się składnia. Nowocześniejsze polecenia zaprojektowano od podstaw pod kątem spójności.

Powtórzmy naszą operację, korzystając z tradycyjnego polecenia; aby wyświetlić adres IP interfejsu, po prostu wpisz `ifconfig`:

```
robv@ubuntu:~$ ifconfig
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1400
    inet 192.168.122.22 netmask 255.255.255.0 broadcast 192.168.122.255
    inet6 fe80::1ed6:5b7f:5106:1509 prefixlen 64 scopeid 0x20<link>
    ether 00:0c:29:33:2d:05 txqueuelen 1000 (Ethernet)
    RX packets 161665 bytes 30697457 (30.6 MB)
    RX errors 0 dropped 910 overruns 0 frame 0
    TX packets 5807 bytes 596427 (596.4 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0
    collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 1030 bytes 91657 (91.6 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 1030 bytes 91657 (91.6 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0
    collisions 0
```

Jak widać, wyświetlane są mniej więcej te same informacje w nieco innym formacie. Jeśli przejrzysz strony man obu poleceń, zauważysz, że opcje `ip` są bardziej spójne, a `ifconfig` w ograniczonym stopniu obsługuje IPv6 — na przykład nie pozwala wyświetlić tylko informacji o IPv4 lub IPv6.

Wyświetlanie informacji o trasach

W zbiorze współczesnych poleceń sieciowych do wyświetlania informacji o trasach służy ten sam program `ip`. Jak można się było spodziewać, odpowiednie polecenie to `ip route`, które można skrócić nawet do `ip r`:

```
robv@ubuntu:~$ ip route
default via 192.168.122.1 dev ens33 proto dhcp metric 100
169.254.0.0/16 dev ens33 scope link metric 1000
192.168.122.0/24 dev ens33 proto kernel scope link src 192.168.122.156 metric 100

robv@ubuntu:~$ ip r
default via 192.168.122.1 dev ens33 proto dhcp metric 100
169.254.0.0/16 dev ens33 scope link metric 1000
192.168.122.0/24 dev ens33 proto kernel scope link src 192.168.122.156 metric 100
```

Wyniki te pokazują, że mamy *trasę domyślną* wskazującą adres 192.168.122.1. Trasa domyślna działa dokładnie tak, jak sugeruje jej nazwa — jeśli pakiet jest zaadresowany do miejsca przeznaczenia, którego nie ma w tabeli tras, host wyśle go do swojej bramy domyślnej. Tabela tras zawsze preferuje trasę „najbardziej konkretną” — taką która najlepiej pasuje do docelowego adresu IP. Jeśli nie da się dopasować adresu, najbardziej konkretna jest trasa 0.0.0.0 0.0.0.0, prowadząca do bramy domyślnej (innymi słowy, jest to trasa oznaczająca „jeśli pakiet nie pasuje do niczego innego”). Host zakłada, że adres IP bramy domyślnej należy do routera, który będzie wiedział (oby!), gdzie dalej przesłać ten pakiet.

Widzimy też trasę do adresu 169.254.0.0/16. Według definicji w dokumencie RFC 3927 jest to tak zwany **adres lokalny dla łącza** (LLA, ang. *Link-Local Address*). RFC (skrót od ang. *Request for Comment* — prośba o komentarze) to dokumenty używane w nieformalnym procesie recenzowania rozwijanych standardów internetowych. Lista opublikowanych dokumentów RFC jest dostępna w witrynie IETF (ang. *Internet Engineering Task Force*) pod adresem <https://www.ietf.org/standards/rfcs/>.

Adresy LLA działają tylko w bieżącej podsieci — jeśli host nie ma statycznie skonfigurowanego adresu IP, a protokół DHCP nie przydzieli mu adresu, host użyje dwóch pierwszych oktetów zdefiniowanych w dokumencie RFC (169.254), a następnie w sposób pseudolosowy wybierze dwa ostatnie oktety. Po teście ping/ARP (protokół ARP omówimy w rozdziale 3., „Używanie Linuksa i linuksowych narzędzi do diagnostyki sieci”), który sprawdza, czy ten obliczony adres jest rzeczywiście dostępny, host jest gotowy do komunikacji. Adres ten jest przeznaczony tylko do komunikacji z innymi adresami LLA w tym samym segmencie sieci, zwykle z wykorzystaniem protokołów rozgłoszeniowych i multemisyjnych, takich jak ARP i Alljoyn, w celu „odnalezienia” siebie nawzajem. Dla jasności: z adresów tych niemal nigdy nie korzysta się w rzeczywistych sieciach; są one używane tylko wtedy, gdy nie ma absolutnie

żadnej alternatywy. A żeby jeszcze bardziej namieszać nam w głowach, Microsoft określa je innym terminem — **APIPA** (ang. *Automatic Private Internet Protocol Addressing*).

Wreszcie widzimy trasę do lokalnej podsieci, w tym przypadku 192.168.122.0/24. Jest ona nazywana **trasą połączoną** (ponieważ jest połączona z tym interfejsem). Z perspektywy hosta oznacza to, że nie potrzebuje on żadnego routingu do komunikowania się z innymi hostami w jego własnej podsieci.

Taki zestaw tras jest bardzo często spotykany w prostych sieciach — brama domyślna, segment lokalny, i to wszystko. W wielu systemach operacyjnych nie zobaczysz podsieci 169.254.0.0, chyba że host rzeczywiście używa adresu LLA.

Jeśli chodzi o polecenia tradycyjne, jest wiele sposobów wyświetlania bieżącego zestawu tras. Najczęściej używa się polecenia `netstat -rn` (*network status* — stan sieci; `-r` — pokaż trasy, `-n` — wyniki w postaci liczbowej). Istnieje jednak również oddzielne polecenie `route` (dlaczego — dowiesz się dalej w tym rozdziale):

```
robv@ubuntu:~$ netstat -rn
```

```
Kernel IP routing table
```

Destination	Gateway	Genmask	Flags	MSS	Window	irrt	Iface
0.0.0.0	192.168.122.1	0.0.0.0	UG	0	0	0	ens33
169.254.0.0	0.0.0.0	255.255.0.0	U	0	0	0	ens33
192.168.122.0	0.0.0.0	255.255.255.0	U	0	0	0	ens33

```
robv@ubuntu:~$ route -n
```

Destination	Gateway	Genmask	Flags	Metric	Ref	Use	Iface
0.0.0.0	192.168.122.0	0.0.0.0	UG	100	0	0	ens33
169.254.0.0	0.0.0.0	255.255.0.0	U	1000	0	0	ens33
192.168.122.0	0.0.0.0	255.255.255.0	U	100	0	0	ens33

Wyświetlane są te same informacje, ale teraz mamy dwa dodatkowe polecenia — `netstat` i `route`. Zbiór tradycyjnych poleceń zwykle ma oddzielne, unikatowe programy do każdego celu, a w tym przypadku widzimy dwa, których funkcje w dużej mierze się pokrywają. Nauczenie się tych wszystkich poleceń i zapamiętanie ich zróżnicowanej składni może być trudne dla początkujących użytkowników Linuksa. Zbiór poleceń `ip` bardzo ułatwia sprawę!

Bez względu na to, którego zbioru narzędzi ostatecznie będziesz używał, wiesz teraz, jak sprawdzić informacje o adresach IP i trasach, co powinno Ci pomóc w zapewnieniu podstawowej łączności dla swojego hosta.

Adresy IPv4 i maski podsieci

W poprzednim rozdziale krótko wspomnieliśmy o adresach IP, ale porozmawiajmy o nich nieco bardziej szczegółowo. Protokół IPv4 pozwala unikatowo zaadresować każde urządzenie w *podsieci* przez przypisanie mu adresu oraz maski podsieci. W naszym przykładzie adres IPv4 to 192.168.122.182. Każdy *oktet* adresu IPv4 może mieć wartość od 0 do 255, a maska

podsieci to /24, co często zapisuje się również jako 255.255.255.0. Wydaje się to skomplikowane, dopóki nie przekształcimy liczb w reprezentację binarną. 255 w formacie binarnym to 11111111 (8 bitów), a w trzech takich grupach są 24 bity. Zatem adres i maska mówią nam, że po zamaskowaniu część reprezentująca sieć to 192.168.122.0, a część reprezentująca hosta (która może przybierać wartości od 1 do 254) to 182.

Rozpiszmy to:

Adres	192	168	122	182
Adres w formacie binarnym	11000000	10101000	01111010	10110110
Maska w formacie binarnym	11111111	11111111	11111111	00000000

A gdybyśmy potrzebowali większej podsieci? Możemy po prostu przesunąć maskę o kilka bitów w lewo. Na przykład w przypadku maski 20-bitowej otrzymujemy:

Adres	192	168	122	182
Adres w formacie binarnym	11000000	10101000	01111010	10110110
Maska w formacie binarnym	11111111	11111111	11110000	00000000

Trzeci oktet maski jest teraz równy 0b11110000 (zauważ prefiks 0b, oznaczający „liczbę binarną”), co jest równoważne liczbie 240 w zapisie dziesiętnym. Zwiększa to zakres adresów dla naszych hostów do 0 - 15 (0 - 0b1111) w trzecim okciecie oraz 0 - 255 (0 - 0b11111111) w czwartym okciecie, co razem daje $15 \cdot 255 - 1 = 3824$ adresy (w następnym podrozdziale wyjaśnimy, skąd to -1).

Jak widzisz, aplikacja kalkulatora, która konwertuje liczby binarne na dziesiętne i odwrotnie, może być bardzo przydatna dla specjalisty ds. sieci! Upewnij się, że obsługuje również liczby szesnastkowe (o podstawie 16); zajmiemy się nimi za kilka minut.

Teraz, kiedy już wiesz, jak pracować z adresami i maskami podsieci w formacie dziesiętnym, a zwłaszcza binarnym, rozwińmy te informacje i zbadajmy, jak wykorzystać je do zilustrowania innych koncepcji adresowania.

Adresy specjalnego przeznaczenia

Istnieje kilka adresów *specjalnego przeznaczenia*, które musimy omówić, abyś zrozumiał, jak działają adresy IP w lokalnej podsieci. Po pierwsze, jeśli wszystkie bity hosta w adresie są ustawione na 1, taki adres nazywamy **adresem rozgłoszeniowym** (ang. *broadcast address*). Jeśli wyślesz dane na adres rozgłoszeniowy, zostaną one odczytane przez wszystkie interfejsy sieciowe w podsieci.

Zatem adres rozgłoszeniowy dla przykładowej sieci /24 wyglądałby tak:

Adres	192	168	122	182
Adres w formacie binarnym	11000000	10101000	01111010	10110110
Maska w formacie binarnym	11111111	11111111	11111111	00000000
Adres rozgłoszeniowy	11000000	10101000	01111010	11111111

Innymi słowy, nasz adres rozgłoszeniowy to 192.168.122.255.

Adres rozgłoszeniowy dla sieci /20 wygląda następująco:

Adres	192	168	122	182
Adres w formacie binarnym	11000000	10101000	01111010	10110110
Maska w formacie binarnym	11111111	11111111	11110000	00000000
Adres rozgłoszeniowy	11000000	10101000	01111111	11111111

Możemy przekształcić go z powrotem w format dziesiętny i otrzymać 192.168.127.255.

Przesuwanie granicy między bitami sieci a bitami hosta w adresie IPv4 przychodzi na myśl koncepcję **klas adresów**. Po przekształceniu w postać binarną kilka pierwszych bajtów definiuje tak zwaną **klasową** maskę podsieci dla adresu. W większości systemów operacyjnych, kiedy ustawiasz adres IP w graficznym interfejsie użytkownika, ta klasowa maska podsieci jest często wprowadzana automatycznie. Relacja między klasami a maskami podsieci przedstawia się następująco:

Klasa	Początkowe bity adresu	Maska podsieci (bity)	Maska podsieci (dziesiętna)	Pierwszy adres w zakresie	Ostatni adres w zakresie
Klasa A	0	/8	255.0.0.0	0.0.0.0	127.255.255.255
Klasa B	10	/16	255.255.0.0	128.0.0.0	191.255.255.255
Klasa C	110	/24	255.255.255.0	192.0.0.0	223.255.255.255
Klasa D (adresy multimedialne)	1110	nd.	nd.	224.0.0.0	239.255.255.255
Klasa E (zarezerwowana, nieużywana)	1111	nd.	nd.	240.0.0.0	255.255.255.255

Powyższa tabela definiuje domyślne klasowe maski podsieci. Przyjrzymy się temu bliżej w następnych dwóch podrozdziałach.

Z tego wszystkiego łatwo wywnioskować, dlaczego administratorzy zwykle używają **klasowych granic** podsieci w swoich organizacjach. Zdecydowana większość wewnętrznych podsieci ma maskę 255.255.255.0 lub 255.255.0.0. Każdy inny wybór prowadzi do zamieszania i grozi błędami w konfiguracji serwerów lub stacji roboczych, kiedy do zespołu dołącza nowa osoba. Poza tym większość osób nie ma ochoty wykonywać obliczeń za każdym razem, kiedy trzeba ustawić lub zinterpretować adres sieciowy.

Drugi typ specjalnego adresu, o którym przed chwilą napomknęliśmy, to adres **multiemisyjny** (ang. *multicast address*). Adres ten umożliwia komunikację między kilkoma urządzeniami jednocześnie. Możesz na przykład użyć adresu multiemisyjnego, aby wysłać identyczny strumień wideo do kilku ekranów podłączonych do sieci albo poprowadzić połączenie konferencyjne w aplikacji głosowej lub wideo. Lokalne adresy multiemisyjne mają następującą postać:

Możliwe wartości binarne	11100000	00000000	00000xxx	xxxxxxxx
Możliwe wartości dziesiętne	224	0	Dowolna liczba od 1 do 8	Dowolna liczba od 1 do 255

Ostatnie 11 bitów (3 + 8) zwykle tworzy „dobrze znane adresy” różnych protokołów multiemisji. Niektóre często używane adresy multiemisyjne to:

224.0.0.1	Wszystkie hosty w podsieci
224.0.0.2	Wszystkie routery w podsieci
224.0.0.12	Urządzenia uczestniczące w protokole VRRP
224.0.0.18	Serwery DHCP i przełączniki DHCP
224.0.0.102	Urządzenia uczestniczące w protokole HSRP (ang. <i>Hot Standby Router Protocol</i>)
224.0.1.1	Wszystkie serwery NTP (ang. <i>Network Time Protocol</i>)
224.0.0.113	Hosty Alljoyn (używane przez Windows do odkrywania sąsiednich urządzeń)

Pełna lista dobrze znanych, zarejestrowanych adresów multiemisyjnych jest dostępna w witrynie organizacji **IANA** (ang. *Internet Assigned Numbers Authority*) pod adresem <https://www.iana.org/assignments/multicast-addresses/multicast-addresses.xhtml>. Choć wydaje się ona kompletna, producenci często definiują własne adresy multiemisyjne w tej przestrzeni adresowej.

To tylko krótkie wprowadzenie do adresowania multiemisyjnego — zagadnienie to jest znacznie bardziej skomplikowane, do tego stopnia, że projektowaniu, wdrażaniu i teorii sieci multiemisyjnych poświęcono całe książki. To, co omówiliśmy, powinno jednak dać Ci ogólne pojęcie i wystarczyć na początek.

Po omówieniu adresów rozgłoszeniowych i multimiisyjnych zajmijmy się „rodzinami” adresów IP, które najprawdopodobniej są używane w Twoim środowisku.

Adresy prywatne — RFC 1918

Innym zbiorem adresów specjalnych jest przestrzeń adresowa RFC 1918. Dokument RFC 1918 zawiera listę podsieci IP przydzielonych do wewnętrznego użytku organizacji. Adresy te są ignorowane w publicznym internecie, więc jeśli mają być używane do komunikacji przez internet, konieczne jest „tłumaczenie” ich z wykorzystaniem **translacji adresów sieciowych** (NAT, ang. *Network Address Translation*).

Adresy RFC1918 są następujące:

- 10.0.0.0/8 (klasa A),
- od 172.16.0.0 do 172.31.0.0/16 (klasa B) — można je podsumować jako 172.16.0.0/12,
- 192.168.0.0/16 (klasa C).

Adresy te zapewniają organizacjom dużą przestrzeń adresową IP do użytku wewnętrznego, która nie koliduje z żadnymi adresami w publicznym internecie.

Interesującym ćwiczeniem będzie użycie tych podsieci RFC 1918 do zweryfikowania domyślnej klasy adresów przez przetłumaczenie pierwszego oktetu na postać binarną i porównanie go z tabelą w poprzednim podrozdziale.

Pełną specyfikację RFC 1918 można znaleźć pod adresem <https://tools.ietf.org/html/rfc1918>.

Teraz, kiedy omówiliśmy binarne aspekty adresowania IP i maski podsieci, a także różne specjalne grupy adresów IP, zapewne masz już dość teorii oraz matematyki i chcesz jak najszybciej wrócić do eksperymentowania w wierszu poleceń swojego komputera. Zła wiadomość jest taka, że musimy jeszcze omówić szczegóły adresowania IPv6 (wersji 6. protokołu IP). Dobra zaś taka, że zaczekamy z tym do dodatku, więc będziesz mógł szybciej wrócić do klawiatury!

Skoro umiesz już wyświetlać parametry i rozumiesz, jak działa adresowanie IP, skonfigurujmy interfejs IP do użytku.

Przypisywanie adresu IP do interfejsu

Przypisywanie stałego adresu IPv4 do interfejsu to coś, co będziesz robił w niemal każdym samodzielnie zbudowanym serwerze. Na szczęście jest to dość proste. W nowym zbiorze poleceń użyjemy do tego programu `nmcli` (ang. *Network Manager Command Line* — **wiersz poleceń menedżera sieci**). Ustawimy adres IP, bramę domyślną i serwer DNS. Wreszcie

ustawimy tryb adresowania na manual (ręczny). Zaczniemy od wyświetlenia połączeń sieciowych w formacie nmcli:

```
robv@ubuntu:~$ sudo nmcli connection show
NAME                                UUID                                TYPE    DEVICE
Wired connection 1                 02ea4abd-49c9-3291-b028-7dae78b9c968 ethernet ens33
```

Nasza nazwa połączenia to Wired connection 1. Nie musimy jednak wpisywać jej w całości za każdym razem; wystarczy, że napiszemy Wi i naciśniemy klawisz *Tab*, aby uzupełnić nazwę. Warto też wiedzieć, że nmcli obsługuje skrócone nazwy argumentów, więc możemy pisać mod zamiast modify, con zamiast connection itd. Kontynuujemy naszą sekwencję poleceń (zauważ skrócone argumenty w ostatnim poleceniu):

```
$ sudo nmcli connection modify "Wired connection 1" ipv4.addresses 192.168.122.22/24
$
$ sudo nmcli connection modify "Wired connection 1" ipv4.gateway 192.168.122.1
$
$ sudo nmcli connection modify "Wired connection 1" ipv4.dns "8.8.8.8"
$
$ sudo nmcli con mod "Wired connection 1" ipv4.method manual
$
```

Teraz zapiszmy zmiany i uaktywnijmy je:

```
$ sudo nmcli connection up "Wired connection 1"
Connection successfully activated (D-Bus active path:
/org/freedesktop/NetworkManager/ActiveConnection/5)
$
```

W tradycyjnym podejściu wszystkie zmiany wprowadza się przez edycję plików. A żeby było ciekawiej, nazwy plików i ich lokalizacje bywają różne w różnych dystrybucjach. Tutaj przedstawimy najbardziej typowe modyfikacje i pliki.

Aby zmienić serwer DNS, otwórz w edytorze plik */etc/resolv.conf* i zmodyfikuj wiersz nameserver tak, aby odzwierciedlał adres IP wybranego serwera:

```
nameserver 8.8.8.8
```

Aby zmienić adres IP, maskę podsieci itd., otwórz w edytorze plik */etc/sysconfig/network-scripts/ifcfg-eth0* i zmodyfikuj wartości w następujący sposób:

```
DEVICE=eth0
BOOTPROTO=none
ONBOOT=yes
NETMASK=255.255.255.0
IPADDR=10.0.1.27
```

Jeśli ten interfejs jest połączony z bramą domyślną, możesz dodać wiersz:

```
GATEWAY=192.168.122.1
```

Warto jeszcze raz przypomnieć, że w różnych dystrybucjach pliki do edycji mogą być inne, a **podejście to nie jest kompatybilne wstecz**. W większości nowoczesnych dystrybucji zmienianie ustawień sieci przez edycję plików nie działa.

Teraz, kiedy już wiesz, jak przypisać adres IP do interfejsu, nauczysz się modyfikować trasy w swoim hoście.

Dodawanie trasy

Aby dodać tymczasową trasę statyczną, ponownie sięgniemy po polecenie `ip`. W tym przykładzie informujemy hosta, że ma kierować pakiety na adres `192.168.122.10`, aby dostać się do sieci `10.10.10.0/24`:

```
robv@ubuntu:~$ sudo ip route add 10.10.10.0/24 via 192.168.122.10
[sudo] password for robv:
robv@ubuntu:~$ ip route
default via 192.168.122.1 dev ens33 proto dhcp metric 100
10.10.10.0/24 via 192.168.122.10 dev ens33
169.254.0.0/16 dev ens33 scope link metric 1000
192.168.122.0/24 dev ens33 proto kernel scope link src 192.168.122.156 metric 100
```

Możesz też dodać przeznaczony do tego wyjściowy interfejs sieciowy przez dołączenie `dev <nazwa_urzqdzienia>` na końcu tego polecenia `ip route add`.

Jest to jednak tylko trasa tymczasowa, która zniknie, jeśli host albo procesy sieciowe zostaną uruchomione ponownie. Trwałą trasę statyczną można dodać za pomocą polecenia `nmcli`. Najpierw wyświetlimy połączenia sieciowe w formacie `nmcli`:

```
robv@ubuntu:~$ sudo nmcli connection show
NAME                UUID                                  TYPE      DEVICE
Wired connection 1  02ea4abd-49c9-3291-b028-7dae78b9c968  ethernet  ens33
```

Teraz za pomocą `nmcli` dodajmy do połączenia `Wired connection 1` trasę, która prowadzi do sieci `10.10.11.0/24` przez adres `192.168.122.11`:

```
robv@ubuntu:~$ sudo nmcli connection modify "Wired connection 1" +ipv4.routes
"10.10.11.0/24 192.168.122.11"
```

Zapiszmy dokonane zmiany poleceniem `nmcli`:

```
$ sudo nmcli connection up "Wired connection 1"
Connection successfully activated (D-Bus active path:
/org/freedesktop/NetworkManager/ActiveConnection/5)
$
```

Jeśli teraz przyjrzymy się tabeli tras, zobaczymy obie nasze trasy statyczne:

```
robv@ubuntu:~$ ip route
default via 192.168.122.1 dev ens33 proto dhcp metric 100 10.10.10.0/24 via
192.168.122.10 dev ens33
10.10.11.0/24 via 192.168.122.11 dev ens33 proto static metric 100
```

```
169.254.0.0/16 dev ens33 scope link metric 1000
192.168.122.0/24 dev ens33 proto kernel scope link src 192.168.122.156 metric 100
```

Jeśli jednak ponownie uruchomimy komputer, przekonamy się, że trasa tymczasowa zniknęła, a trasa trwała nadal jest obecna:

```
robv@ubuntu:~$ ip route
default via 192.168.122.1 dev ens33 proto dhcp metric 100
10.10.11.0/24 via 192.168.122.11 dev ens33 proto static metric 100
169.254.0.0/16 dev ens33 scope link metric 1000
192.168.122.0/24 dev ens33 proto kernel scope link src 192.168.122.156 metric 100
```

Znasz już podstawy dodawania tras. Przyjrzyjmy się teraz, jak wykonać to samo zadanie w starszym hoście linuksowym za pomocą tradycyjnych poleceń `route`.

Dodawanie trasy tradycyjnym sposobem

Aby tymczasowo dodać trasę, użyj polecenia:

```
$ sudo route add -net 10.10.12.0 netmask 255.255.255.0 gw 192.168.122.12
```

Trwale dodawanie trasy jest jednak bardziej skomplikowane — trwale trasy są przechowywane w plikach, których nazwy i lokalizacje zależą od dystrybucji. Właśnie dlatego spójność poleceń `iproute2/nmcli` tak bardzo ułatwia pracę we współczesnych systemach.

W starszych dystrybucjach Debian/Ubuntu zwykle edytuje się plik `/etc/network/interfaces` i dodaje następujący wiersz:

```
up route add -net 10.10.12.0 netmask 255.255.255.0 gw 192.168.122.12
```

W starszych dystrybucjach z rodziny Red Hat edytuje się plik `/etc/sysconfig/network-scripts/route-<nazwa_urzadzenia>` i dodaje następujący wiersz:

```
10.10.12.0/24 via 192.168.122.12
```

Można też po prostu dodać trasy za pomocą poleceń `route` umieszczonych w pliku `/etc/rc.local` — ten sposób zadziała właściwie w każdym systemie linuksowym, ale jest uważany za mało elegancki, ponieważ plik ten jest ostatnim miejscem, gdzie następny administrator będzie szukał odpowiedniego ustawienia (bo nie jest to plik, w którym typowo zapisuje się ustawienia sieci). Plik `rc.local` jest uruchamiany podczas rozruchu systemu i wykonuje dowolne polecenia, które są w nim zapisane. W tym przypadku może to być nasze polecenie `route add`:

```
/sbin/route add -net 10.10.12.0 netmask 255.255.255.0 gw 192.168.122.12
```

W tym momencie jesteśmy już bliscy chwili, gdy w pełni skonfigurujemy łączność sieciową w naszym hoście. Ustawiliśmy adres IP, maskę podsieci i trasy. Jednak często, zwłaszcza podczas rozwiązywania problemów albo wstępnej konfiguracji, trzeba włączyć lub wyłączyć interfejs; zajmiemy się tym w następnym punkcie.

Wyłączanie i włączanie interfejsu

W świecie nowych poleceń do włączania i wyłączania interfejsu służy — zgodnie — program `ip`. Poniżej „przeładowujemy” interfejs przez jego wyłączenie i ponowne włączenie:

```
robv@ubuntu:~$ sudo ip link set ens33 down
robv@ubuntu:~$ sudo ip link set ens33 up
```

W starym zbiorze poleceń interfejs włącza się lub wyłącza poleceniem `ipconfig`:

```
robv@ubuntu:~$ sudo ifconfig ens33 down
robv@ubuntu:~$ sudo ifconfig ens33 up
```

Podczas wykonywania poleceń związanych z interfejsami zawsze pamiętaj, żeby nie *podcinać gałęzi, na której siedzisz*. Jeśli jesteś zalogowany zdalnie (na przykład za pomocą `ssh`) i zmienisz adresowanie IP lub trasy albo wyłączysz interfejs, możesz utracić połączenie z hostem.

Omówiliśmy już większość zadań, które będziesz musiał wykonywać, aby skonfigurować swój system do pracy w nowoczesnej sieci. Ważną częścią administrowania siecią jest jednak również diagnozowanie jej działania i dobieranie ustawień do specjalnych przypadków, na przykład regulowanie ustawień w celu zoptymalizowania ruchu, kiedy potrzebne są mniejsze lub większe rozmiary pakietów.

Ustawianie jednostki MTU interfejsu

Jedną z operacji, którą coraz częściej wykonuje się w nowoczesnych systemach, jest ustawianie **maksymalnej jednostki transmisji** (MTU, ang. *Maximum Transmission Unit*). Jest to rozmiar największej **jednostki danych protokołu** (PDU, ang. *Protocol Data Unit* — w większości sieci nazywanej **ramką**), którą interfejs będzie wysyłać lub odbierać. W Ethernetie domyślną jednostką MTU jest 1500 bajtów, co przekłada się na maksymalny rozmiar pakietu równy 1500 bajtom. Maksymalny rozmiar pakietu w danym nośniku jest zwykle nazywany **maksymalnym rozmiarem segmentu** (MSS, ang. *Maximum Segment Size*). W przypadku Ethernetu te trzy parametry mają wartości przedstawione w tabeli 2.1.

Tabela 2.1. Relacja między rozmiarem ramki, MTU, rozmiarem pakietu i MSS dla Ethernetu

Maksymalny rozmiar ramki	1518 bajtów
MTU — maksymalny rozmiar danych w ramce	1500 bajtów
Maksymalny rozmiar pakietu (taki sam jak MTU, ponieważ pakiet jest danymi użytkowymi ramki)	1500 bajtów
MSS — maksymalny rozmiar danych użytkowych jednego pakietu	1460 bajtów

Dlaczego miałbyś je zmieniać? 1500 to dobry kompromis, ponieważ pakiet jest na tyle mały, że w razie błędu można szybko wykryć problem, a ilość danych do ponownego przesłania jest względnie mała. Istnieje jednak kilka wyjątków, zwłaszcza w centrach danych.

W przypadku ruchu związanego z pamięcią masową, w szczególności iSCSI, pożądane są większe rozmiary ramki, żeby pakiet mógł pomieścić więcej danych. Wówczas MTU zwykle ustawia się na mniej więcej 9000 (co często określa się mianem **pakietu jumbo**). Sieci te zazwyczaj działają z prędkością 1 Gb/s, 10 Gb/s lub wyższą. Większe pakiety pojawiają się też w ruchu związanym z tworzeniem kopii zapasowych lub migracją maszyn wirtualnych (na przykład VMotion w VMware lub Live Migration w Hyper-V).

Na drugim końcu spektrum często zdarzają się sytuacje, w których potrzebny jest mniejszy rozmiar pakietu. Jest to szczególnie ważne, ponieważ nie wszystkie hosty potrafią to poprawnie wykrywać, a wiele aplikacji ustawia w swoim ruchu bit **DF** (ang. *Don't Fragment* — **nie fragmentuj**). Pakiet o rozmiarze 1500 bajtów z ustawionym bitem DF może wówczas pojawić się w nośniku, który obsługuje tylko pakiety o rozmiarze na przykład 1380 bajtów — w takim przypadku aplikacja po prostu zawiedzie, a komunikaty o błędach nie pomogą rozwiązać problemu. W jakich okolicznościach może tak się zdarzyć? Przykładowo na dowolnym łączu z kapsułkowaniem pakietów, takim jak tunel lub sieć VPN. Rozmiar ramki (i w konsekwencji rozmiar pakietu) jest tu ograniczony przez dodatkowe dane związane z kapsułkowaniem, których rozmiar zwykle dość łatwo obliczyć. Innym typowym przykładem są łącza satelitarne. Mają one często ramki o domyślnym rozmiarze 512 bajtów — w takich przypadkach rozmiary są publikowane przez dostawców usług.

Ustawianie MTU jest dokładnie tak proste, jak można się było spodziewać — ponownie użyjemy do tego polecenia `nmcli`. Zauważ, że w tym przykładzie skracamy argumenty wiersza polecenia `nmcli`, a na koniec zapisujemy zmiany w konfiguracji — jednostka MTU jest zmieniana natychmiast po ostatnim poleceniu. Ustawmy MTU na 9000, aby zoptymalizować ruch iSCSI:

```
$ sudo nmcli con mod "Wired connection 1" 802-3-ethernet.mtu 9000
$ sudo nmcli connection up "Wired connection 1"
Connection successfully activated (D-Bus active path:
/org/freedesktop/NetworkManager/ActiveConnection/5)
$
```

Co jeszcze można zrobić za pomocą `nmcli` po ustawieniu MTU?

Więcej o poleceniu `nmcli`

Program `nmcli` można również uruchomić w trybie interaktywnym, aby dokonywać zmian w interpreterze czasu rzeczywistego, czyli powłoce. Aby uruchomić tę powłokę dla interfejsu ethernetowego, wydaj polecenie `nmcli connection edit type ethernet`. Po uruchomieniu powłoki możesz wydać polecenie `print`, aby wyświetlić listę wszystkich parametrów, które można zmienić dla tego typu interfejsu. Zauważ, że parametry są podzielone na logiczne grupy — zredagowaliśmy tę (bardzo długą) listę, aby pokazać ustawienia, które prawdopodobnie będziesz musiał regulować, modyfikować lub diagnozować w różnych sytuacjach:

```
nmcli> print
=====
Connection profile details (ethernet)
```

```

=====
connection.id:                ethernet
connection.uuid:              e0b59700-8dcb-4801-9557-9dee5ab7164f
connection.stable-id:        --
connection.type:              802-3-ethernet
connection.interface-name:    --
...
connection.lldp:              -1 (default)
connection.llmnr:             -1 (default)
=====

```

Oto często używane opcje połączenia ethernetowego:

```

802-3-ethernet.port:          --
802-3-ethernet.speed:         0
802-3-ethernet.duplex:        --
802-3-ethernet.auto-negotiate: no
802-3-ethernet.mac-address:   --
802-3-ethernet.mtu:           auto
...
802-3-ethernet.wake-on-lan:    default
802-3-ethernet.wake-on-lan-password: --
=====

```

Oto często używane opcje IPv4:

```

ipv4.method:                  auto
ipv4.dns:                     --
ipv4.dns-search:              --
ipv4.dns-options:             --
ipv4.dns-priority:            0
ipv4.addresses:               --
ipv4.gateway:                 --
ipv4.routes:                  --
ipv4.route-metric:            -1
ipv4.route-table:             0 (unspec)
ipv4.routing-rules:           --
ipv4.ignore-auto-routes:      no
ipv4.ignore-auto-dns:         no
ipv4.dhcp-client-id:          --
ipv4.dhcp-iaid:               --
ipv4.dhcp-timeout:            0 (default)
ipv4.dhcp-send-hostname:      yes
ipv4.dhcp-hostname:           --
ipv4.dhcp-fqdn:               --
ipv4.dhcp-hostname-flags:     0x0 (none)
ipv4.never-default:           no
ipv4.may-fail:                yes
ipv4.dad-timeout:             -1 (default)
=====

```

(Tutaj następowałyby opcje IPv6, ale je usunęliśmy, aby zwiększyć czytelność listingu).
Oto ustawienia proxy:

```
-----
proxy.method:                none
proxy.browser-only:          no
proxy.pac-url:                --
proxy.pac-script:            --
-----
```

Jak już wspomniano, powyższy listing jest nieco skrócony. Pokazaliśmy ustawienia, które najczęściej sprawdza się lub zmienia podczas konfigurowania i diagnozowania połączeń sieciowych. Aby zobaczyć pełny listing, uruchom polecenie we własnym komputerze.

Jak pokazaliśmy, polecenie `nmcli` pozwala zmienić kilka parametrów interfejsu sieciowego albo interaktywnie, albo z poziomu wiersza poleceń. Interfejs wiersza poleceń umożliwia określanie ustawień sieciowych w skryptach, co jest bardziej skalowalnym sposobem, pozwalającym na modyfikowanie parametrów dziesiątek, setek, a nawet tysięcy stacji jednocześnie.

Podsumowanie

Po przeczytaniu tego rozdziału powinieneś rozumieć adresowanie IP z perspektywy binarnej. Wiesz, jak działają podsieci i maski, a także adresowanie rozgłoszeniowe i multemisyjne. Poznałeś też różne klasy adresów IP. Dzięki tej wiedzy będziesz umiał wyświetlić lub ustawić adresy IP i trasy w hoście linuxowym za pomocą wielu różnych poleceń. Łatwo zmienisz też inne parametry interfejsu sieciowego, takie jak jednostka MTU.

Po opanowaniu tych umiejętności jesteś dobrze przygotowany na nasz następny temat: używanie Linuksa i narzędzi linuxowych do diagnozowania sieci.

Pytania

Oto lista pytań, które pozwolą Ci sprawdzić swoją wiedzę na temat materiału omawianego w tym rozdziale. Odpowiedzi znajdziesz w dodatku A, „Sprawdziany wiadomości”:

1. Do czego służy brama domyślna?
2. Jaka jest maska podsieci i adres rozgłoszeniowy sieci 192.168.25.0/24?
3. W jaki sposób w tej samej sieci używany jest adres rozgłoszeniowy?
4. Jakie są w tej samej sieci możliwe adresy hostów?
5. Gdybyś musiał statycznie ustawić szybkość i tryb dwukierunkowy interfejsu ethernetowego, jakiego polecenia byś użył?

Dalsza lektura

- RFC 1918 — Address Allocation for Private Internets:
<https://tools.ietf.org/html/rfc1918>
- RFC 791 — Internet Protocol: *<https://tools.ietf.org/html/rfc791>*

Skorowidz

A

- ACL, Access Control List, 162
- ACME, 211
- AD, Active Directory, 212, 224
- adres
 - IPv4, 41, 44, 218
 - IPv6, 43
 - lokalny dla łącza, LLA, 43
 - MAC, 63, 65, 233
 - multimisyjny, multicast address, 47
 - rozgłoszeniowy, broadcast address, 45
- adresy
 - prawdziwe, 252
 - prywatne, 48
 - przypisywane do interfejsu, 48
 - specjalnego przeznaczenia, 45
 - statyczne, 189
 - wartości OUI, 68
 - wirtualne, 252
 - zmienianie, 67
- Advanced Package Tool, 39
- aktualizacje, 122
- alarmy, 316, 339, 340, 384
 - Suricata, 378
- algorytm
 - Least Connections, 258
 - roundrobin, 269
 - równoważenia obciążenia, 257
- analiza dzienników, 314
- API, Application Programming Interface, 34, 166
- APIPA, Automatic Private Internet Protocol Addressing, 44

- aplikacje NetFlow, 346, 360
- AppArmor, 18, 147
 - funkcje, 148
- apt
 - apt-cache, 39
 - apt-get, 39
 - strona man, 40
- ARP, Address Resolution Protocol, 60, 63, 285
- ASN, Autonomous System Numbers, 69
- atak
 - DoS, 270
 - typu brute-force, 224
 - typu MITM, 94, 184, 288
 - WPAD, 95, 185
- automatyczna konfiguracja serwera proxy, 184
- Azure AD, Azure Active Directory, 211

B

- backend, 252, 269
- Balance URI, 258
- bazy danych, 313
- Bettercap, 19
- bezpieczeństwo, 121
 - branżowe standardy, 124
 - hostów, 122
 - infrastruktury urzędu certyfikacji, 206
- moduł
 - AppArmor, 147
 - SELinux, 147
- usług
 - chmurowych, 123
 - DHCP, 182
 - równoważenia obciążenia, 275

bezprzewodowe punkty dostępowe, WAP, 160, 222

BIND, Berkeley Internet Name Domain, 18, 158, 160
implementacja, 163
instalacja serwera, 160

bit DF, 53

błędy SSL, 392

Boulder, 201

BPF, Berkely Packet Filter, 289

BSD, 31

C

CA, Certificate Authority, 171, 193

cat, 78

CDN, Content Delivery Networks, 371

CDP, 297

centralna konfiguracja, 177

Certbot, 211

Certificate Manager, 201

certyfikat SSL/TLS, 394

certyfikaty, 96, 193

automatyzacja zarządzania, 211

bezpłatne, 211

budowanie urzędu certyfikacji, 201

pozyskiwanie, 195

SSL wygasłe, 396

testowe, 209

transparentność, 208

używanie, 197

weryfikacja, 198

wieloznaczne, 209

wycofane, 199

CHAP, Challenge-Handshake Authentication Protocol, 227

chmura, 33

CIS, Center for Internet Security, 125

środki kontroli, 125

wzorce, 140

Cisco, 236

CN, Common Name, 171

Cowrie, 19

CRL, Certificate Revocation List, 199

CSR, Certificate Signing Request, 195

tworzenie wniosku, 204

CT, Certificate Transparency, 193, 209

cut, 77

D

DAI, Dynamic ARP Inspection, 286

dane przepływu, 356

DDNS, Dynamic DNS, 159, 189

DF, Don't Fragment, 53

DHCP, Dynamic Host Configuration Protocol, 40, 95, 155, 177, 293

analiza pakietu, 296

diagnozowanie, 190

działanie przełącznika, 180

działanie, 177

instalowanie serwera, 186

konfigurowanie serwera, 186

lista dzierżaw, 190

nieautoryzowany klient, 186

nieautoryzowany serwer, 183

opcje, 181, 296

rejestrowanie zdarzeń, 190

sekwencja DORA, 178

zabezpieczanie usług, 182

żądania z innych podsioci, 179

diagnostyka sieci, 59

diagnozowanie

DHCP, 190

łączności bezprzewodowej, 98

DNS, Domain Name System, 49, 95, 153

diagnozowanie, 165

funkcje serwera, 157, 159

implementacje, 160

internetowy serwer, 158

konfiguracja internetowego serwera, 164

rekonesans, 165

dnsmasq, 160

DNSSEC, DNS Security Extension, 154, 172

dodawanie

daty i godziny, 85

tekstu, 85

trasy, 50, 51

urządzeń, 342

DoH, 153, 166

domeny najwyższego poziomu, 156

DORA, Discover, Offer, Request,

Acknowledgement, 178

dostawca usług internetowych, ISP, 157

dostęp

do graficznego interfejsu użytkownika, 235

do interfejsu SSH, 235

do routerów i przełączników, 236

do urządzeń sieciowych, 235

do zapory, 237

dostrajanie wydajności, 267
 DoT, 153, 168
 implementacja w Nmap, 172
 dpkg, 39
 DSCP, Differentiated Services Code Point, 69, 114, 119
 DShield, 19, 324, 414
 DSL, Digital Subscriber Line, 113
 dSniff, 19
 DSR, 255
 ustawienia serwera, 257
 wady, 256
 dyrektywy SSL, 271
 dystrybucje, 32, 34
 dzienniki, 313
 analiza, 314
 priorytety zdarzeń, 319
 zapory, 315

E

EAP, Extensible Authentication Protocol, 237
 Easy-RSA, 201
 EF, Expedited Forwarding, 303
 egzekwowanie typów, 147
 Ettercap, 19, 287
 EveBox
 wyświetlanie informacji o zdarzeniu, 378
 wyświetlanie zdarzeń, 382

F

filtrowanie danych przepływu, 357
 filtry przechwytywania, 291
 CDP, 297
 LLDP, 297
 w programie tcpdump, 292
 w programie Wireshark, 290, 295
 filtry wyświetlania
 w programie Wireshark, 306
 format PEM, 224
 fragmentacja, 373
 FreeBSD, 31
 FreeRADIUS, 18, 216, 232
 frontend, 252, 269
 przetwarzanie HTTPS, 273

G

Google Authenticator, 244
 graficzny interfejs użytkownika, GUI, 26, 235
 grep, 78, 82
 gromadzenie informacji o przepływach, 347, 349
 GUI, Graphical User Interface, 26, 235

H

HAProxy, 18, 266–271
 hasła do kluczy certyfikatów, 204
 hiperwizor, 332
 honeypot, 20, 402
 alarmujący o próbach połączenia z portem, 410
 iptables, 410
 ISC, 422
 netcat, 410
 portspooft, 410
 raporty online, 422–424
 rozproszony/społecznościowy, 414
 scenariusze wdrożeniowe, 405
 SSH, 423
 Thinkst Canary, 414
 w publicznym internecie, 408
 w sieci wewnętrznej, 407
 w strefie DMZ, 407
 WebLabyrinth, 414
 zagrożenia, 408
 HSM, Hardware Security Module, 206
 HTTPS, HyperText Transfer Protocol Secure, 153, 271, 273

I

IANA, Internet Assigned Numbers Authority, 47
 ICMP, 83, 259
 ICS, Industrial Control Systems, 86
 identyfikator
 obiektu, OID, 328
 punktu dostępowego, BSSID, 98
 sieci bezprzewodowej, SSID, 98, 219, 240
 użytkownika, ID, 217
 iDNS, Internal DNS Service, 157
 IDS, Intrusion Detection System, 290
 IETF, Internet Engineering Task Force, 43
 ifconfig, 38, 60
 implementacja DoT, 172

informacje
 o przepływach, 347, 349
 o trasach, 43
 o zdarzeniu, 378, 379, 380
 Infrastructure as Code, 34
 infrastruktura kluczy publicznych, PKI, 201
 instalacja
 programu bind, 160
 serwera DHCP, 186
 interfejs
 IP, 40
 programowania aplikacji, API, 34, 166
 ustawianie jednostki MTU, 52
 wyłączanie i włączanie, 52
 internet rzeczy, IoT, 224, 290
 Internet Storm Center, 414
 inwentaryzacja, 209
 oprogramowania, 134
 sprzętu, 131
 zasobów programowych, 127
 zasobów sprzętowych, 126
 IOS, Internetwork Operating System, 236
 IoT, Internet of Things, 224, 290
 IP, Internet Protocol, 60, 218
 strona man, 42
 IPFIX, 346
 iproute, 43
 iproute2, 38
 IPS, Intrusion Prevention Systems, 290, 365
 konstruowanie reguły, 385
 rozwiązanie
 Snort, 375
 Suricata, 375
 WAF, 372
 umiejscowienie systemu, 367
 unikanie detekcji, 372
 iptables, 18, 410
 kolejność operacji, 114, 116
 konfigurowanie zapory, 106
 opis zapory, 107
 pliki include, 118
 reguły, 109
 usuwanie konfiguracji, 119
 wykrywanie żądania połączenia, 410
 IPv4, 41, 44, 218
 IPv6, 43
 IR, Incident Response, 312
 ISC DHCP, 18
 ISP, Internet Service Provider, 157

J

jednostka
 danych protokołu, PDU, 52
 MTU, 52
 transferu komunikatów, MTU, 94
 J-Flow, 346
 JSON, JavaScript Object Notation, 211
 JWK, JSON Web Key, 211
 JWT, JSON Web Token, 211

K

Kali Linux, 32
 kapsułkowanie danych, 62
 karta sieciowa, NIC, 63
 Kismet, 18, 60, 98
 ekran programu, 100
 klasy adresów, 46
 klient DHCP, 186
 klucze certyfikatów, 204
 knot-dnutils, 170
 komunikacja TLS, 198
 konfiguracja
 interfejsu przełącznika, 242, 243
 przełącznika Cisco, 281
 przełącznika, 241
 routera Cisco, 236
 serwera RADIUS, 237
 serwera DHCP, 186
 SSID, 240
 urządzenia SNMPv2, 333
 konsole SSL, 395
 kontrola
 CIS, 125
 dostępu oparta na rolach, 147, 208
 jakości, QA, 210
 kończenie sesji, 81
 krotka, 69

L

LDAP, 223
 LDAPS, 224
 LibreNMS, 19, 331
 dodawanie
 alarmu, 340
 urządzenia, 334, 338
 informacje o interfejsie, 336
 kolekcja alarmów, 339

- monitorowanie
 - podstawowej usługi, 343
 - usługi HTTPS, 344
- reguły alarmu, 340, 341
- statystyki urządzenia, 335
- widok
 - Syslog, 342
 - usług, 344
- LinSSID, 60, 101, 102
 - ekran główny, 101
- Linux, 28
- lista
 - bieżących dzierżaw, 190
 - kontroli dostępu, ACL, 162
 - nasłuchujących portów i połączeń, 72
 - portów, 70
 - urzędów certyfikacji, 171
 - wycofanych certyfikatów, 199
- LLA, Link-Local Address, 43
- LLDP, Link Layer Discovery Protocol, 181, 297
 - analiza pakietu, 299
 - ramka, 298
- LLMNR, Local Link Multicast Name Resolution, 96
- logowanie superużytkownika, 143
- LSM, Linux Security Module, 147
- lsnf, 80

M

- MAC, Media Access Control, 233
- maksymalna jednostka transmisji, MTU, 52
- maksymalny rozmiar segmentu, MSS, 52, 114
- man, 40, 42
- mangle, 113
- maski podsieci, 44
- MDM, Mobile Device Management, 211
- metadane sieci, 391
- MFA, Multi-Factor Authentication, 164, 216
- MITM, Machine in the Middle, 94, 184
- MITM, Man in the Middle, 288
- model
 - Cap-Ex, 34
 - Op-Ex, 34
 - OSI, 61
 - warstwa 2., 63
 - warstwa 4., 69
- moduły bezpieczeństwa, LSM, 147
- monitor Zeek, 391
- monitorowanie, 129
 - pasywne POF, 389
 - ruchu

- sieci, 311
- usług, 342
- MS-CHAP, Microsoft CHAP, 227
- MSS, Maximum Segment Size, 52, 114
- MTU, Maximum Transmission Unit, 52
- MTU, Message Transfer Unit, 94

N

- named, 160
- narzędzia PCAP, 289
- NAS, Network-Attached Storage, 92
- NAS, network access server, 218
- NAT, Network Address Translation, 48, 111, 253, 262
- nc, 60, 81, 84
- NDES, Network Device Enrollment Service, 212
- Netcat, 18, 60, 81, 83
- NetFlow, 346, 351
- netplan, 60
- netstat, 60, 72, 76
 - strona man, 73
- net-tools, 38
- Network Mapper, 167
- NetworkMiner, 19
- nfcapd, 19
- nfdump, 19
- NFDump, 351
- NFSen, 19, 351
 - agregacja danych przepływu, 357
 - analiza natężenia ruchu, 358
 - dane przepływu, 356
 - filtrowanie adresów IP, 359
 - filtrowanie danych przepływu, 357
- nftables, 18
 - konfigurowanie zapory, 115
- NIC, Network Interface Card, 63
- niepożądane hosty VPN, 96
- Nmap, 18, 60, 81, 86, 167
 - implementacja DoT, 172
 - ograniczenia, 97
 - skrypty, 91
 - zastosowania programu, 96
- nmbd, 230
- nmcli, 48, 53
- NMS, 331
- nota wdrożeniowa, 273
- NPS, Network Policy Server, 227
- NTLM, NT LAN Manager, 226

O

OAuth, Open Authorization, 211
 obowiązkowa kontrola
 dostępu, 147
 integralności, 147
 obrona
 przed złośliwym oprogramowaniem, 128
 sieci, 129
 obsługa HTTPS, 271
 ochrona
 danych, 127
 poczty elektronicznej, 128
 OSCP, Online Certificate Status Protocol, 199
 OIDC, OpenID Connect, 211
 opcje DHCP, 296
 OpenBSD, 31
 OpenSSL, 18, 212
 budowanie urzędu certyfikacji, 202
 operator &&, 123
 oprogramowanie aplikacyjne, 130
 Oracle/Scientific Linux, 30
 OSQuery, 136
 OUI, Organizationally Unique Identifier, 68

P

P0F, 389
 pakiet jumbo, 53
 PAM, Pluggable Authentication Module, 244
 PAP, Password Authentication Protocol, 225
 pasywne monitorowanie ruchu, 389
 pasywny skaner podatności, PVS, 389
 PBR, Policy-Based Routing, 263
 PDU, Protocol Data Unit, 52
 PEAP, Protected Extensible Authentication Protocol, 228
 PEM, Privacy Enhanced Mail, 224
 PKI, Public Key Infrastructure, 201
 plik OVA, 266, 331
 pliki
 .tsv, 78
 include, 118
 przechwytywanie, 300
 podatność na exploity, 93
 podsłuch sieciowy, 283
 port, 70
 22, 80
 389, 225
 443, 80, 271, 393
 4444, 412

514, 319
 80, 80
 efemeryczny, 69
 monitorowania, 281
 Portspooft, 19, 412
 porty
 lokalne
 nasłuchiwanie, 72
 wyliczanie, 72
 podsłuchu, 283
 TCP, 72
 UDP, 72
 zdalne
 wyliczanie, 80, 86
 potrójne uzgodnienie, 70
 priorytety zdarzeń, 319
 projekt DShield, 19, 324, 414
 protokoły
 802.1x, 241
 ACME, 211
 ARP, 60, 63, 285
 CDP, 297
 CHAP, 227
 DHCP, 40, 95, 155, 177, 293
 DNSSEC, 154, 172
 DoH, 153, 166
 DoT, 153, 168
 EAP-TLS, 237
 EAP-TLS/802.1x, 239
 LDAPS, 224
 LLDP, 181, 297
 LLMNR, 96
 PAP, 225
 PEAP, 228
 RADIUS, 216, 217
 RTP, 303
 SCEP, 211
 SIP, 303
 TCP, 69, 70
 UDP, 69, 70, 159, 219
 UPnP, 96
 proxy, 251
 przechwytywanie pakietów, 280
 filtrowanie, 290
 narzędzia, 289
 podsłuch sieciowy, 283
 połączenia telefonicznego VoIP, 302
 pozyskiwanie plików, 300
 wydajność, 287
 złośliwe sposoby, 284
 przekaznik DHC, 179P, 180

przetwarzanie danych w chmurze, 33
 przywracanie danych, 128
 PVS, Passive Vulnerability Scanner, 389
 pwnplug, 186

Q

QA, Quality Assurance, 210
 QOS, Quality of Service, 69, 119

R

RADIUS, 216, 217
 dostęp do urządzeń sieciowych, 235
 klient, 223
 kody, 218
 konfiguracja serwera, 237, 240
 odpowiedź, 220
 serwer, 223
 uwierzytelnianie
 EAP-TLS, 237
 lokalne, 221
 NTLM, 227
 w sieci bezprzewodowej, 239
 w sieci przewodowej, 241
 używanie
 LDAP, 223, 225
 LDAPS, 224
 wdrażanie usług, 221
 zastosowania usług, 234
 żądanie, 219
 RASP, Runtime Application Self Protection, 371
 reagowanie na incydenty, IR, 312
 Red Hat, 30
 reguły IPS, 385
 rejestrowanie zdarzeń, 190, 312, 319
 rekonesans, 209
 RFC, Request for Comments, 43, 158, 209
 RIP, Real IP, 252, 267
 rotacja, 313
 routing, 263
 PBR, 263, 264
 równoważenie obciążenia, 248, 249
 algorytm, 257, 269
 architektura, 261
 bezpieczeństwo, 275
 dostrajanie wydajności, 267
 konfigurowanie trwałych połączeń, 272
 metoda
 Balance URI, 258
 RRDNS, 250, 258

odwrotny serwer proxy, 252
 przetwarzanie HTTPS, 273
 serwer proxy, 251
 typu NAT/proxy, 266
 usług TCP, 269
 użycie

 DSR, 255
 HAProxy, 266
 translacji NAT, 253

 w warstwie 4., 253
 w warstwie 7., 251
 w centrum danych, 259
 wdrażanie systemu, 259

rpm, 39

RRDNS, Round Robin DNS, 249
 konfigurowanie, 250

RRL, Response Rate Limiting, 159

RSPAN, Remote Switched Port Analyzer, 281

rsyslog, 18, 319

RTP, Real-Time Protocol, 303
 bity DSCP, 304
 dane aplikacyjne, 304

S

SAN, Subject Alternative Name, 171

SCEP, Simple Certificate Enrollment Protocol, 211

Security Onion, 33

sekwencja DORA, 178

SELinux, 18, 147

 funkcje, 147

SELKS, 366

serwer

 BIND, 160

 DHCP, 95

 instalowanie i konfigurowanie, 186

 nieautoryzowany, 183

 DNS, 49, 95, 154

 funkcje, 157, 159

 internetowy, 158

 ograniczenia, 158

 DoH, 166

 dostępu do sieci, NAS, 218

 NetFlow, 351

 OCSP, 199

 proxy, 251

 RADIUS, 244

 syslog, 319

 RDP, 96

 SSH, 96

sesja TCP, 74, 75
 SFLOW, 346
 sieci
 bezprzewodowe, 98
 dystrybucji treści, CDN, 371
 rozległe, WAN, 94
 wirtualne, VLAN, 265
 SIFT, 33
 SIP
 analiza pakietu, 303
 skanery portów, 81
 skanowanie
 pod kątem podatności na exploity, 93
 portów, 86
 TCP, 87
 UDP, 88
 skrót MD5, 219
 skrypty Nmap, 91
 Smallstep, 201, 211
 smbd, 230
 SMS, Short Message Service, 244
 SNI, Server Name Indication, 171
 SNMP, 95
 drzewo identyfikatorów OID, 328, 329
 wdrożenie systemu NMS, 331
 zapytania, 327
 zarządzanie urządzeniami sieciowymi, 327
 snmpget, 19
 SNMPv2
 dodawanie urządzenia, 334
 konfigurowanie, 333
 SNMPv3
 dodawanie urządzenia, 338
 konfigurowanie, 335
 snmpwalk, 19, 337
 snooping DHCP, 183
 Snort, 19, 375
 SPAN, Switched Port Analyzer, 281
 ss, 60, 77
 SSH, Secure Shell, 141, 235
 logowanie superużytkownika, 143
 SSID, 98, 219, 240
 SSL, Secure Sockets Layer, 209
 testowanie błędów, 392
 wyświetlanie danych, 393
 SSO, Single Sign-On, 211
 sudo, 39
 Suricata, 19, 375, 376
 alarmy, 378
 SUSE, 30
 syslog
 rejestrowanie zdarzeń, 312

system
 IPS Suricata, 376
 nazw domenowych, DNS, 153
 zapobiegania włamaniom, IPS, 365
 szyfry, 144

Ś

środki kontroli, 126
 CIS, 130

T

tabela
 mangle, 113
 NAT, 111
 TCP, Transmission Control Protocol, 69, 70
 potrójne uzgodnienie, 71
 równoważenie obciążenia, 269
 stany sesji, 74, 75
 tcpdump, 19, 289
 filtry przechwytywania, 292
 telnet, 60, 80
 testy penetracyjne, 130
 Thinkst Canary, 20
 Throwing Star, 283
 TLD, Top-Level Domain, 156
 TLS, Transport Layer Security, 153
 TOS, Type of Service, 69, 114
 translacja adresów sieciowych, NAT, 48, 111, 253, 262
 transparentność certyfikatów, 193, 208
 trasy, 43
 dodawanie, 50, 51
 statyczne, 262
 TShark, 19, 289
 TTL, Time to Live, 155
 tworzenie urzędu certyfikacji, 201

U

Ubuntu, 31
 UDP, User Datagram Protocol, 69, 70, 159, 219
 unikanie systemów IPS, 373
 Unlang, 232
 UPnP, Universal Plug and Play, 96
 urząd certyfikacji, CA, 193
 tworzenie, 201
 zabezpieczanie infrastruktury, 206
 zagrożenia, 208

urządzenia sieciowe, 327

usługa

 certyfikatów, 193

 CT, 209

 DHCP, 177

 DNS, 153

 DSL, 113

 FreeRADIUS, 216

 Google Authenticator, 244

 honeypot, 402

 nmbd, 230

 RADIUS, 216

 równoważenia obciążenia, 248

 smbd, 230

 SNMP, 95

 SSL, 96

 syslog, 312

 TCP, 269

 TLS, 96

uwierzytelnianie

 802.1x, 241

 CHAP, 227

 dwuetapowe, 227

 EAP-TLS, 237–239

 lokalne, 221

 MFA, 223, 244

 NTLM, 227

 PEAP, 232

 VPN, 234

 w sieci bezprzewodowej, 239

 w sieci przewodowej, 241

 wielopoziomowe, MFA, 164, 216

uzupełnianie argumentów, 41

V

VIP, Virtual IP, 252, 266

visudo, 39

VLAN, 241, 265

VoIP, Voice over IP, 70, 181, 242, 302

VPN, Virtual Private Network, 94, 218

 identyfikator użytkownika i hasło, 234

W

WAF, Web Application Firewall, 254, 372

WAN, Wide Area Network, 94

WAP, Wireless Access Point, 160, 222

warstwa

 łącza danych, 63

 transportu, 69

Wavemon, 18, 60, 100

WebLabyrinth, 20

wewnętrzne żądania usług DNS, 157

widok

 EveBox, 382

 Hunting, 382

 Management, 383

wiersz poleceń menedżera sieci, 48

Wireshark, 19, 289

 badanie pakietu DHCP, 296

 filtry przechwytywania, 290, 295

 filtry wyświetlania, 306

 odtworzenie konwersacji VoIP, 305

 przechwytywanie połączenia

 telefonicznego, 302

 zapisywanie konwersacji VoIP, 306

wirtualizacja, 33

wirtualne

 sieci lokalne, VLAN, 241

 sieci prywatne, VPN, 94

wirtualny adres IP, VIP, 266

wniosek CSR, 204

WPAD, Windows Proxy Auto Discovery, 185

wybieranie dystrybucji, 34

wygaśnięcie certyfikatu, 96

wykrywanie rozwiązań WAF, 372

wrażenia regularne, 233

wzorce CIS, 140

Y

YaST, Yet another Setup Tool, 201

yum, 39

Z

zabezpieczanie

 konfiguracji zasobów, 127

 SSH, 141

zaplecze, backend, 252, 269

zapobieganie włamaniom, 365

zapor, 105

 aplikacji webowych, WAF, 254

 iptables, 106

 nftables, 115

zapory open source, 32

zapytania SNMP, 327

zarządzanie

 certyfikatami, 211

 dostawcami usług, 130

 dziennikami audytu, 128

zarządzanie
 infrastrukturą sieciową, 129
 kontami, 127
 kontrolą dostępu, 127
 lukami w zabezpieczeniach, 127
 reakcją na incydenty, 130
 urządzeniami sieciowymi, 327
 zatruwanie pamięci, 285
 zdarzenia
 informacje szczegółowe, 378–380
 rejestrowanie, 319
 Zeek, 19, 391
 filtrowanie portu 443, 393
 informacje
 geolokalizacyjne, 398
 o podejrzanym adresie IP, 399
 konsola Security Onion, 395
 metadane
 certyfikatu SSL/TLS, 394
 dotyczące sieci, 391
 testowanie błędów SSL, 392
 wyświetlanie
 danych SSL, 393
 informacji SSL/TLS, 397

złośliwe
 routery, 94
 serwery, 95
 serwery DHCP, 185
 sposoby przechwytywania pakietów, 284
 usługi, 95
 zmienianie adresu MAC, 67
 zypper, 39

Ż

źle
 skonfigurowana infrastruktura sieciowa, 94
 skonfigurowane usługi SNMP, 95

Ż

żądania
 ARP, 63
 DHCP, 17966
 DNS, 156
 GET, 80, 84
 RADIUS, 219

PROGRAM PARTNERSKI

— GRUPY HELION —



1. ZAREJESTRUJ SIĘ
2. PREZENTUJ KSIĄŻKI
3. ZBIERAJ PROWIZJĘ

Zmień swoją stronę WWW w działający bankomat!

Dowiedz się więcej i dołącz już dzisiaj!

<http://program-partnerski.helion.pl>

GRUPA
Helion 

Linux: korzystaj z najwyższych standardów bezpieczeństwa!

Linux zdobył popularność zarówno wśród użytkowników, jak i administratorów sieci i zaskarbił sobie ich uznanie. Stało się tak nie bez powodu, ponieważ pozwala on na uzyskanie imponującej elastyczności usług sieciowych przy relatywnie niewielkich kosztach. Usługi sieciowe Linuksa mogą zapewnić funkcjonalność niedostępną w przypadku innych systemów. Dzięki nim można stworzyć solidnie zabezpieczone, efektywne i doskonale dopasowane do szczególnych potrzeb organizacji środowisko sieciowe. Wystarczy dobrze poznać i zrozumieć działanie poszczególnych usług sieciowych Linuksa.

Ta książka jest przeznaczona dla inżynierów zarządzających infrastrukturą sieciową dowolnego rodzaju. Znajdziesz tu niezbędne informacje, których potrzebujesz do uruchomienia i skonfigurowania różnych użytecznych usług sieciowych. Najpierw poznasz najważniejsze dystrybucje i podstawy konfiguracji sieci w Linuksie. Następnie przejdziesz do diagnozowania sieci, konfigurowania zapory oraz używania Linuksa jako hosta usług sieciowych. W dalszej kolejności uzyskasz informacje o przydatnych usługach i o ich wdrażaniu w środowisku korporacyjnym. Sporo miejsca w książce poświęcono zagadnieniom ochrony przed nieuprawnionym dostępem: omówiono typowe sposoby przeprowadzania ataków oraz techniki skutecznego zabezpieczania usług sieciowych. Ta publikacja dostarczy Ci przydatnych wskazówek, które pozwolą nie tylko skonfigurować potrzebne usługi sieciowe, ale także zbudować centrum danych oparte wyłącznie na Linuksie.

Najciekawsze zagadnienia:

- Linux jako platforma do diagnozowania sieci i rozwiązywania problemów
- konfiguracja zapory Linuksa
- konfiguracja usług sieciowych, w tym DNS i DHCP
- rejestrowanie zdarzeń w celu monitorowania sieci
- wdrażanie i konfiguracja systemów zapobiegania włamaniom (IPS)
- konfiguracja usługi honeypot w celu wykrywania i odpierania ataków

Rob VandenBrink jest uznanym ekspertem w dziedzinie bezpieczeństwa informacji, infrastruktury sieciowej oraz projektowania sieci i centrów danych, a także automatyzacji IT i wirtualizacji. Napisał oprogramowanie ułatwiające bezpieczne korzystanie z VPN i stworzył różnorodne narzędzia sieciowe dla Cisco IOS. Uzyskał liczne certyfikaty SANS/GIAC, VMware i Cisco.

Helion 	KOD KORZYŚCI Ściągnij po więcej!	
 helion.pl	ISBN 978-83-283-9710-1	
 HELION SA ul. Kościuszki 1c 44-100 Gliwice tel.: 32 230 98 63 helion@helion.pl	 9 788328 397101	
Cena: 119,00 zł		

Packt