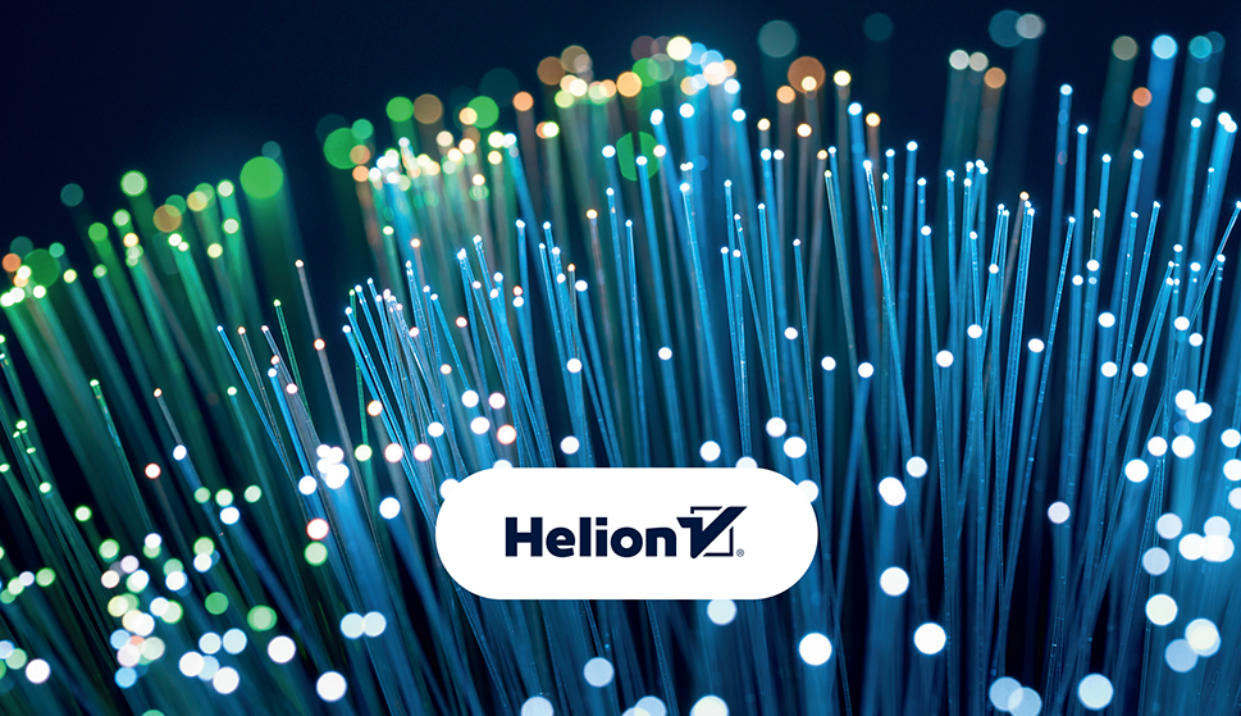
Adam Józefiok

CCNP 350-401 ENCOR

Zaawansowane administrowanie siecią Cisco



Helion

Redaktor prowadzący: Małgorzata Kulik

Recenzja naukowa: Prof. dr hab. inż. Tadeusz Czachórski

Projekt okładki: Studio Gravite / Olsztyn
Obarek, Pokoński, Pazdrijowski, Zaprucki
Grafika na okładce została wykorzystana za zgodą Shutterstock.com

Helion S.A.
ul. Kościuszki 1c, 44-100 Gliwice
tel. 32 231 22 19, 32 230 98 63
e-mail: *helion@helion.pl*
WWW: *https://helion.pl* (księgarnia internetowa, katalog książek)

Drogi Czytelniku!
Jeżeli chcesz ocenić tę książkę, zajrzyj pod adres
https://helion.pl/user/opinie/zaawpr
Możesz tam wpisać swoje uwagi, spostrzeżenia, recenzję.

ISBN: 978-83-283-5237-7

Printed in Poland.

Spis treści

Wstęp 15

Rozdział 1. Wprowadzenie do sieci kampusowych i projektowania 17

Wstęp 17
Pytania wstępne 18
Sieci kampusowe 18
Model hierarchiczny 24
Wprowadzenie do projektowania sieci kampusowych 29
Pytania sprawdzające 35
Odpowiedzi 35

Rozdział 2. Przełączniki — urządzenia warstwy drugiej i trzeciej OSI 36

Wstęp 36
Pytania wstępne 36
Informacje wprowadzające i ogólne działanie przełącznika 37
Ethernet 39
Działanie przełącznika warstwy drugiej 44
Działanie przełączników warstwy trzeciej OSI 47
CEF 48
Sposoby dostępu do urządzeń sieciowych 53
Pytania sprawdzające 56
Odpowiedzi 56

Rozdział 3. Konfiguracja przełączników 57

Wstęp 57
Pytania wstępne 57
Sieci VLAN i połączenia trunk 58
Ogólne informacje o VLAN 58
Konfiguracja sieci VLAN 63

Połączenia trunk ... 68
Protokół DTP ... 78
Protokół VLAN Trunking Protocol ... 82
VTP Pruning ... 92
Tunelowanie 802.1Q (QinQ) ... 94
Redundancja w sieci L2 — protokół STP ... 99
Rodzaje portów w STP ... 106
Koszty tras ... 108
Stany portów ... 111
Rozszerzenie protokołu STP, czyli protokół PVST ... 114
Protokół RSTP ... 119
Protokół MST ... 124
Agregacja portów — EtherChannel ... 133
Pytania sprawdzające ... 138
Odpowiedzi ... 138

Rozdział 4. Routing EIGRP ... 139

Wstęp ... 139
Pytania wstępne ... 139
Podstawy routingu ... 140
Algorytmy używane przez protokoły routingu ... 143
Routing statyczny ... 146
Sumaryzacja tras statycznych ... 149
Default route ... 153
Najdłuższe dopasowanie ... 155
Floating Static Route ... 156
Protokół EIGRPv4 ... 162
Konfiguracja EIGRP ... 164
Konfiguracja routera stub w EIGRP ... 194
Protokół EIGRPv6 ... 200
Pytania sprawdzające ... 205
Odpowiedzi ... 206

Rozdział 5. Protokół routingu OSPF ... 207

Wstęp ... 207
Pytania wstępne ... 207
Protokół OSPFv2 ... 209
Pakiety hello i pola komunikatu ... 210
Baza LSDB (Link State Data Base) ... 213

Konfiguracja protokołu OSPF ... 214
Alternatywna konfiguracja protokołu OSPF ... 219
Równoważenie obciążenia w OSPF ... 221
Zmiana identyfikatora routera ... 223
Stany interfejsów i relacje sąsiedzkie ... 225
Wymiana informacji pomiędzy routerami — obserwacja ... 226
Metryka w OSPF ... 233
Zmiana czasów ... 240
Konfiguracja passive-interface ... 241
Rozgłaszanie tras domyślnych ... 242
OSPF w sieciach wielodostępowych ... 243
Wybór routerów DR i BDR ... 243
Statusy po nawiązaniu relacji sąsiedztwa ... 249
Routery DR i BDR w połączeniu punkt – punkt ... 252
Uwierzytelnianie w OSPF ... 254
Wieloobszarowy OSPF ... 258
Typy przesyłanych pakietów LSA ... 259
Konfiguracja wieloobszarowego OSPF ... 260
Typy obszarów i tras wykorzystywanych przez OSPF ... 270
Wybór odpowiedniego typu obszaru ... 275
Filtrowanie w OSPF na podstawie listy dystrybucyjnej ... 280
Protokół OSPFv3 ... 282
Konfiguracja OSPFv3 ... 283
Funkcja BFD (Bidirectional Forwarding Detection) ... 287
Virtual Routing and Forwarding (VRF) ... 289
Pytania sprawdzające ... 294
Odpowiedzi ... 295

Rozdział 6. Protokół BGP ... 296

Wstęp ... 296
Pytania wstępne ... 296
Wprowadzenie do BGP ... 297
Formowanie relacji w BGP ... 302
Różne topologie BGP ... 306
Podstawowa konfiguracja ... 308
Rozwiązywanie problemów ... 313
Uwierzytelnianie w BGP ... 314
eBGP Multihop ... 314
Łącza redundantne w eBGP ... 318
Rozgłaszanie sieci w BGP ... 319

Redystrybucja w BGP ... 322
Sumaryzacja i autosumaryzacja w BGP ... 325
Internal BGP, czyli iBGP ... 328
Konfiguracja atrybutów BGP ... 335
Atrybut Weight (waga) ... 335
Atrybut Local Preference ... 337
Używanie funkcjonalności route maps (mapy trasy) w BGP ... 339
Atrybut Origin Code ... 351
Atrybut MED (Multi Exit Discriminator) ... 351
Filtrowanie tras BGP za pomocą rozszerzonych ACL ... 354
Filtrowanie sieci w BGP ... 355
Filtrowanie za pomocą list prefiksów ... 359
IPv6 w BGP ... 360
Multiprotocol BGP (MP-BGP) ... 360
Filtrowanie informacji w BGP ... 365
Filtrowanie za pomocą ACL i AS-PATH ... 366
Filtrowanie za pomocą oznaczania ... 368
BGP communities ... 369
Pytania sprawdzające ... 373
Odpowiedzi ... 374

Rozdział 7. Koncepcja transmisji grupowej ... 375

Wstęp ... 375
Pytania wstępne ... 375
Wprowadzenie do transmisji grupowej ... 376
Protokół IGMPv1 ... 378
Adresowanie grupowe ... 383
Protokół IGMPv2 ... 386
Protokół IGMPv3 ... 390
PIM, czyli Protocol Independent Multicast ... 393
Dense mode z bliska ... 395
Tryb sparse ... 401
Tryb sparse-dense ... 414
Pytania sprawdzające ... 424

Rozdział 8. Quality of Service (QoS) ... 426

Wstęp ... 426
Pytania wstępne ... 427
Kolejkowanie w sieciach ... 427
Modele QoS ... 433

Konfiguracja protokołu RSVP ... 437
QoS w przełącznikach ... 450
Pytania sprawdzające ... 468
Odpowiedzi ... 468

Rozdział 9. Redundancja w L3 (FHRP). Protokoły HSRP, VRRP i GLBP ... 469

Wstęp ... 469
Pytania wstępne ... 470
Protokół HSRP i jego konfiguracja ... 473
Konfiguracja VRRP ... 485
Konfiguracja GLBP ... 495
Pytania sprawdzające ... 500
Odpowiedzi ... 501

Rozdział 10. Sieci wi-fi ... 502

Wstęp ... 502
Pytania wstępne ... 502
Wprowadzenie do sieci bezprzewodowych ... 503
Działanie sieci bezprzewodowej ... 505
Standardy sieci wi-fi ... 510
Urządzenia bezprzewodowe ... 511
Format ramki ... 513
Mechanizm CSMA/CA ... 527
Sposób połączenia ... 527
Bezpieczeństwo sieci bezprzewodowych ... 532
Typowe ataki na sieci bezprzewodowe ... 535
Zastosowanie i projektowanie sieci bezprzewodowych ... 536
Pytania sprawdzające ... 538
Odpowiedzi ... 538

Rozdział 11. Bezpieczeństwo sieci — niektóre rozwiązania ... 539

Wstęp ... 539
Pytania wstępne ... 539
Rozwiązanie AAA ... 540
Konfiguracja 802.1X na routerze ... 542
Konfiguracja 802.1X na przełączniku ... 545
Konfiguracja stacji roboczej ... 545
Listy ACL ... 547
Konfiguracja standardowych list ACL ... 547
Konfiguracja rozszerzonych ACL ... 550

VLAN Access-List (VACL) ... 551
Funkcjonalność CoPP (Control Plane Policing) ... 554
Zastosowanie rodziny IPsec (Internet Protocol Security) ... 557
Sieci VPN ... 563
Implementacja IKE do połączenia VPN ... 565
Zapory ogniowe (firewalls) ... 579
Tunelowanie i techniki z tym związane ... 584
Tworzenie tunelu GRE ... 584
Tunel VTI (Virtual Tunnel Interface) ... 591
Funkcjonalność LISP ... 594
Virtual Extensible LAN (VXLAN) ... 604
Pytania sprawdzające ... 611
Odpowiedzi ... 612

Rozdział 12. Wirtualizacja w sieci ... 613

Wstęp ... 613
Pytania wstępne ... 613
Sieci SDN ... 614
Przykład zastosowania sieci SDN ... 619
Wirtualizacja ... 630
Usługi chmury ... 630
Maszyny wirtualne ... 632
Routing w środowisku wirtualnym ... 635
Automatyzacja sieci ... 640
Szablony ... 644
Pytania sprawdzające ... 645
Odpowiedzi ... 646

Rozdział 13. Monitorowanie sieci LAN i technologie wspierające komunikację ... 647

Wstęp ... 647
Pytania wstępne ... 647
Logowanie zdarzeń — syslog ... 648
Powiadomienie o zmianach w konfiguracji ... 652
Protokół NTP i konfiguracja ... 654
Konfiguracja NTP ... 657
Funkcja debugowania ... 662
Polecenie traceroute ... 663
Konfiguracja funkcjonalności SPAN port (switch port analyzer) ... 666
Konfiguracja funkcjonalności RSPAN (Remote SPAN) ... 669
Konfiguracja funkcjonalności ERSPAN (Encapsulated Remote Switched Port Analyser) ... 672

Protokół SNMP ... 674
Wykorzystanie i działanie NetFlow ... 682
Wprowadzenie do IP SLA ... 684
Narzędzia służące do zarządzania siecią WAN ... 687
Rozwiązanie SD-WAN ... 687
Rozwiązanie SD-ACCESS ... 690
Translacja adresów — usługa NAT ... 691
Static NAT (translacja statyczna) ... 692
Dynamic NAT (translacja dynamiczna) ... 697
PAT ... 697
NAT oparty na wirtualnym interfejsie ... 699
Interfejs wirtualny i NAT statyczny ... 699
Interfejs wirtualny i NAT dynamiczny ... 701
Interfejs wirtualny i PAT ... 702
Pytania sprawdzające ... 702
Odpowiedzi ... 703

Rozdział 14. Programowanie sieci i automatyzacja ... 704

Wstęp ... 704
Pytania wstępne ... 704
Programowanie ... 705
Korzystanie z rozwiązania Cisco DNA ... 715
Różne modele danych ... 717
Funkcjonalność Cisco EEM ... 719
Pytania sprawdzające ... 724
Odpowiedzi ... 725

Zakończenie ... 727

Skorowidz ... 729

Rozdział 12.

Wirtualizacja w sieci

Wstęp

W tym rozdziale poznasz informacje dotyczące zagadnień związanych z szeroko rozumianą wirtualizacją. Dowiesz się w skrócie, jak działają sieci SDN, poznasz działania kontrolera oraz nauczysz się jego implementacji. Poznasz rodzaje środowisk chmurowych. Następnie zapoznasz się z pojęciami związanymi z maszynami wirtualnymi i skonfigurujesz przykładowy routing w środowisku wirtualnym.

Pytania wstępne

1. Która warstwa działająca w oparciu o ASIC jest odpowiedzialna za przesyłanie pakietów z przełącznika?
   - **a)** control plane,
   - **b)** forwarding plane,
   - **c)** processor plane,
   - **d)** ASIC plane.
2. W której usłudze chmurowej aplikacja jest przechowywana i uruchamiana na komputerach dostawcy, a użytkownik otrzymuje dostęp do niej przez internet?
   - **a)** PaaS,
   - **b)** IaaS,
   - **c)** SaaS,
   - **d)** żadne z powyższych.
3. Który interfejs odpowiedzialny jest za komunikację z kontrolerem w rozwiązaniach SDN?
   - **a)** SBI,
   - **b)** VINT,

c) NBI,

d) SDN.

Sieci SDN

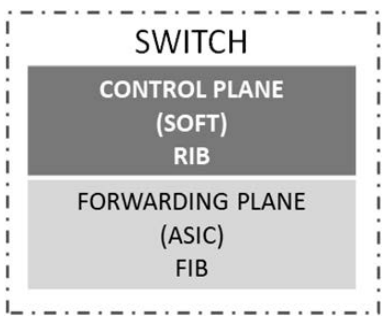
Podczas wykonywania konfiguracji urządzeń w tej książce za każdym razem używałeś konsoli. Jesteśmy też przyzwyczajeni do tego, że przełącznik łączy ze sobą stacje i urządzenia końcowe. Potem router spina wszystko w całość i daje dostęp do sieci WAN lub do internetu. Sieci SDN są rozwiązaniem, które pewne czynności urządzenia odwraca do góry nogami.

SDN (ang. *Software-Defined Network*) to nowe podejście i styl w IT i sieciach komputerowych. Aby zrozumieć tę koncepcję, trzeba się cofnąć w czasie kilkadziesiąt lat. W latach 70. ubiegłego wieku nie stosowało się przełączników. Były to czasy komputerów mainframe, do których były podłączane bezpośrednio inne stacje tego typu. Kolejne lata to era mikrokomputerów, które łączyło się w sieci komputerowe w bardzo popularnej w latach 80. topologii pierścienia. Typowe sieci LAN, które zasadniczo nie zmieniły się do dziś, to lata 90., kiedy to pojawiły się koncentratory, mosty oraz przełączniki.

Obecne przełączniki warstwy 3. pracują, wykorzystując dwie warstwy: *control plane* i *forwarding plane* (*data plane*).

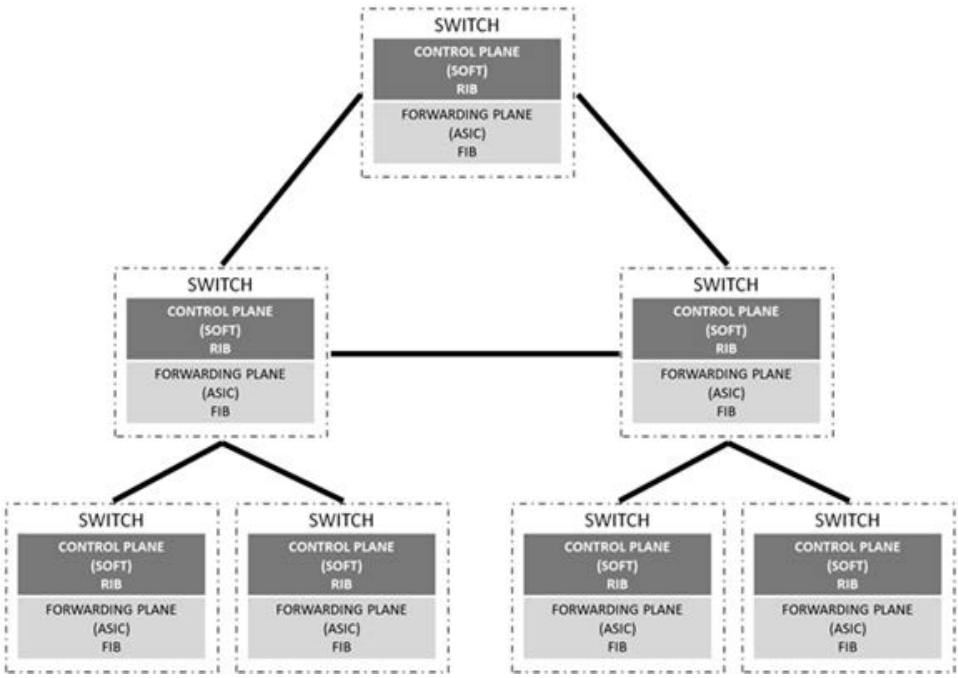
Warstwa *control plane* to swego rodzaju umysł urządzenia, decyduje bowiem o tym, gdzie przesłać pakiety. Jest to rozwiązanie programowe, w którym jako wsparcie funkcjonuje tablica routingu (RIB). To właśnie routery jako typowe rozwiązanie programowe podejmują decyzję o przesłaniu pakietów na podstawie tablicy routingu. Tutaj w przypadku przełącznika odbywa się również nauka adresów MAC koniecznych do zbudowania tablicy MAC. Działają tutaj mechanizmy odpowiedzialne za budowanie tablicy ARP.

Warstwa *forwarding plane* jest odpowiedzialna za „wypychanie" pakietów z przełącznika. Działa w oparciu o układ ASIC; nie jest rozwiązaniem programowym, lecz sprzętowym, a tym samym szybszym. W warstwie tej występuje tablica przełączania (FIB) — na jej podstawie, wykorzystując informacje z *control plane*, przełącznik jest w stanie wysyłać pakiety dalej przez swoje interfejsy (rysunek 12.1).



Rysunek 12.1.
Warstwy działające w przełączniku

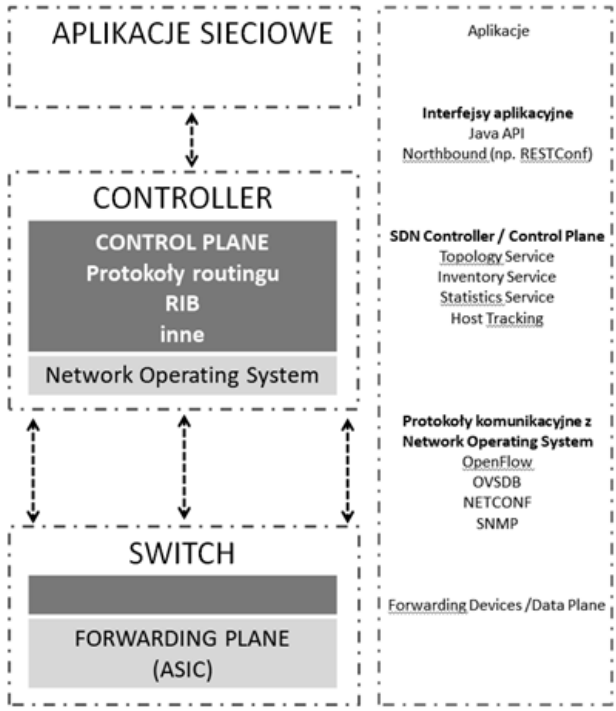
Typowe podejście to zatem tablica RIB i tablica FIB działające na przykład na przełączniku. Przełącznik ma więc warstwę *control plane*, w której znajdują się informacje zasilające tablicę FIB, aby przełącznik wiedział, przez jakie interfejsy wysłać ramki. To rozwiązanie wciąż w sieciach funkcjonuje i dzięki temu, że moce obliczeniowe urządzeń są coraz większe, będzie jeszcze funkcjonowało. Ma jednak kilka wad. Spójrz na rysunek 12.2.



Rysunek 12.2. *Typowy schemat współpracy między przełącznikami*

Na rysunku 12.2 widocznych jest siedem przełączników, które realizują konkretne zadania w postaci przełączania pakietów i ramek. W takim modelu działania znajduje się więc siedem urządzeń, które mają siedem systemów operacyjnych, które trzeba aktualizować. Tyleż samo mamy punktów, które mogą stać się celem ataku. Musimy wykorzystać siedem konsol, aby skonfigurować te urządzenia. Jednak największym minusem tego rozwiązania jest to, że mamy po siedem tablic RIB i FIB. Jeśli więc na jednym urządzeniu pojawia się nowy strumień, musi przejść przez *control plane*. Jeśli jest przekazywany przez kilka przełączników, na każdym z nich dzieje się to samo.

W SDN następuje przejęcie pełnej kontroli nad działaniem sieci poprzez kontrolę sposobu przekazywania strumieni przez urządzenia. Spójrz na rysunek 12.3. Z czego wynika taka potrzeba? Przede wszystkim z potrzeb rynku, który oczekuje pełnego wsparcia w oprogramowaniu i wirtualizowaniu. Skoro mamy maszyny wirtualne w postaci serwerów, to dlaczego przełączniki nie mogłyby być również wirtualne. To samo dotyczy innych urządzeń, takich jak np. routery, urządzenia ASA czy kontrolery bezprzewodowe itd.



Rysunek 12.3. *Działanie urządzeń SDN*

Z przełącznika, który znajduje się na samym dole, „wyjęto" *control plane*. Od tej pory przełączniki to tylko szybkie układy ASIC, które przełączają pakiety; nie robią nic więcej. Można obrazowo powiedzieć, że nie mają funkcji myślenia, lecz tylko wykonują komendy. Dzięki temu pakiety są przełączane bardzo szybko i od razu odsyłane do odpowiedniego interfejsu.

Mózg sieci (*control plane*) znajduje się natomiast w urządzeniu, które nazywane jest kontrolerem SDN (ang. *SDN controller*). Tam zaimplementowany jest system operacyjny wraz z funkcjami zarządzania ruchem. Na kontrolerze następuje proces odnajdywania najlepszej trasy dla pakietów danych i realizowany jest algorytm routingu. Na kontrolerze mogą być implementowane różne protokoły routingu. Jeden kontroler może zarządzać wieloma przełącznikami, przesyłając im odpowiednie rozkazy dotyczące przekazywania strumieni na poszczególne interfejsy.

Jeśli chodzi o kontroler, to administrator może go dowolnie programować z wykorzystaniem różnych aplikacji sieciowych napisanych w różnych językach programowania. Dzięki temu sieci SDN mogą być odpowiednio dopasowane. Jeżeli przedsiębiorstwo potrzebuje określonego rozwiązania i sposobu działania, może opracować je sobie samodzielnie, nie musi korzystać z gotowego oprogramowania do zarządzania kontrolerami.

Kontroler komunikuje się z przełącznikiem za pomocą protokołów komunikacyjnych, takich jak bardzo często wykorzystywany OpenFlow, który umożliwia również zarządzanie kontrolerem, przesyłanie danych i komend. Jednym przełącznikiem może sterować wiele kontrolerów OpenFlow. Sieć SDN wspiera tworzenie wirtualnych sieci komputerowych.

Poniekąd urządzenie, np. przełącznik, będzie w takim przypadku tylko urządzeniem posiadającym płaszczyznę danych i jego zadaniem będzie tylko przesłanie fizyczne danych z jednego interfejsu fizycznego na inny. Natomiast taki przełącznik nie będzie już „samodzielnie myślał", jak to zrobić. Wszystkie rozkazy związanie z przekazywaniem będzie otrzymywał z kontrolera SDN.

Kontroler do zarządzania urządzeniami wykorzystuje interfejsy komunikacyjne z siecią. Są one nazywane interfejsem północnym (ang. *northbound interface — NBI*) oraz interfejsem południowym (ang. *southbound interface — SBI*).

Interfejs NBI daje możliwość komunikacji z kontrolerem administratorowi. Administrator może dzięki niemu zarządzać samym kontrolerem, ale również zarządzać siecią poprzez kontroler. NBI daje możliwość zarządzania bezpośrednio przez GUI, ale również poprzez API i wszelkiego rodzaju związane z tym udogodnienia, takie jak skrypty JavaScriptu lub Pythona oraz inne. Są to rozwiązania, które na dzień dzisiejszy dają wiele możliwości administratorowi. Oczywiście tak jak nigdy wcześniej wymagają poznania technik pisania skryptów lub nauki programowania. Ale staje się to obecnie nieuchronne i konieczne dla dalszego rozwoju. Myślę, że CLI będzie coraz mocniej wypierane na rzecz szeroko rozumianego API.

Interfejs SBI to interfejs odpowiedzialny za komunikację kontrolera z urządzeniami sieciowymi. Obecnie jest to również API. Jak więc widzisz, wsparcie programowe jednak wiedzie prym. I tak jak wspomniałem wcześniej, może to być OpenFlow czy Cisco OpFlex.

API to oprogramowanie, które umożliwia innym aplikacjom dostęp do danych lub usług innego urządzenia. Jest w pewnym sensie pośrednikiem pomiędzy użytkownikiem a danymi na innym urządzeniu, czyli Ty, jako użytkownik, chcesz uzyskać informacje od urządzenia i poprzez API wysyłasz do niego żądanie, aby przesłało Ci dane w odpowiedniej formie (odpowiednio sformatowane), np. JSON.

Jeszcze kilka lat temu, aby zarezerwować miejsce w hotelu, wchodziłeś na stronę WWW danego obiektu i tam mogłeś dokonać rezerwacji. Obecnie istnieją serwisy, które w jednym miejscu kumulują ogłoszenia wielu hoteli. Dzięki temu można porównywać ceny i usługi, by na tej podstawie podjąć decyzję. Jest to możliwe, ponieważ każdy z hoteli daje dostęp do swojej bazy danych i poprzez API zewnętrzny portal może wymieniać informacje z API hotelu. Zauważ, że dostęp w takim przypadku jest dwukierunkowy, a zatem klient po dokonaniu rezerwacji na zewnętrznym portalu otrzymuje od niego informację, że rezerwacja została przyjęta. Zewnętrzny portal musi jednak zaznaczyć w bazie danych hotelu, że pokój jest już zajęty w danym terminie. Wszystko to odbywa się dzięki API i dostępowi do bazy danych.

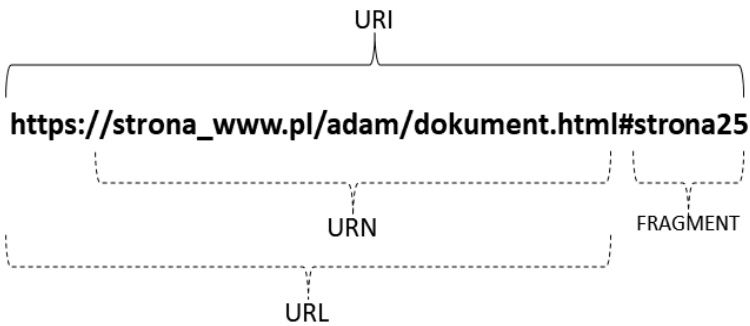
API może być publiczne i wówczas każdy może mieć do niego dostęp, dzięki czemu może prezentować za pomocą własnych aplikacji dane czyjejś bazy. Drugi rodzaj to prywatne API — przykładem może być wspomniana usługa hotelowa. Dzięki rozwiązaniom opartym na API możesz znaleźć lot w interesującym Cię terminie czy dokonać zakupu towarów po atrakcyjnych cenach.

API, które jest szeroko stosowane przez serwisy WWW, to RESTful API (ang. *Representational State Transfer*), bazujące na HTTP. W RESTful API informacje są transportowane przez HTTP, dlatego operacje głównie są wykonywane poprzez URL. W HTTP, jeśli chcesz otrzymać stronę WWW, to właśnie wykorzystując HTTP, wysyłasz żądanie `GET`, a otrzymując stronę, dostajesz odpowiedź w postaci dokumentu HTML. Czasem, gdy uzupełniasz formularz z danymi, wysyłasz żądanie `HTTP POST`. Podczas wykorzystywania API odpowiedzią nie będzie format HTML, lecz np. JSON.

Tak więc REST to model klient – serwer, który umożliwia podłączenie do serwera wielu klientów. Jest też rozwiązaniem, które nie rezerwuje i nie zestawia połączenia na stałe. REST jest w związku z tym bezstanowy.

Wspomniałem o wykorzystaniu URL, czyli ujednoliconego formatu, który znasz. Wykorzystujesz go niemal każdego dnia podczas surfowania w internecie. Ale tak naprawdę URL jest tylko częścią ścieżki dostępu do danych. RESTful wykorzystuje URI (ang. *Uniform Resource Identifier*), czyli ujednolicony identyfikator zasobów (opisany w RFC2396). URI jest standardem, który pozwala na identyfikację zasobów w sieci i składa się z łańcucha znaków prowadzących do tego zasobu (rysunek 12.4).

Rysunek 12.4. *Format URI*



URI składa się z URL (ang. *Uniform Resource Locator*), który jest formatem adresowania zawierającym m.in. protokół, hosta (np. nazwę strony WWW) i ścieżkę do zasobu. Druga część URI to URN (ang. *Uniform Resource Name*) — w tym przypadku sama ścieżka do zasobu. Na samym końcu znajduje się określony fragment poszukiwanych zasobów, najbardziej szczegółowy. URI może być bardziej rozbudowane i może znaleźć się tam również konkretne zapytanie, numer portu. URI, jak wspomniałem, wykorzystywane jest przez API.

Opierając się na tych informacjach, chciałbym Ci uzmysłowić, że komunikacja z urządzeniem w zasadzie odbywa się w bardzo podobny sposób. Tak więc wykorzystując np. skrypt Python, możesz skonstruować żądanie pobrania (HTTP GET) np. stanów interfejsu urządzenia, które będzie wyglądało np. tak: *https://172.16.2.1:8888/sdn/int/status/nodes*. Urządzenie odpowie, np. wykorzystując format JSON.

```
{
    "nodes": [
        {
            "ip": "172.16.2.1",
```

```
            "mac": "abab.1111.baba",
            "vid": 0,
            "dpid": "00:00:00:00:00:00:00:01",
            "port": 1,
        }
    ]
}
```

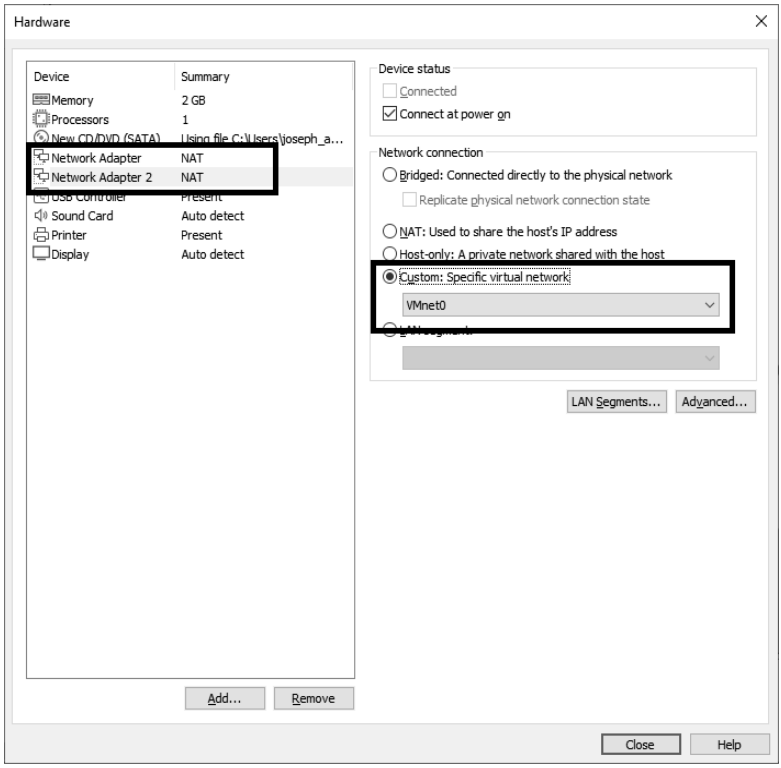
Przykład zastosowania sieci SDN

Chciałbym, abyś wiedział, jak wszystko działa w praktyce. Tak więc w tym podrozdziale chciałbym dać Ci namiastkę własnego kontrolera i wirtualnych przełączników. Do uruchomienia kontrolera będziesz potrzebował maszyny wirtualnej z serwerem Ubuntu. Ja w tym przykładzie użyłem gotowego obrazu *.iso* Ubuntu Server ze strony *ubuntu.com*. Po pobraniu pliku należy utworzyć nową wirtualną maszynę (rysunek 12.5).

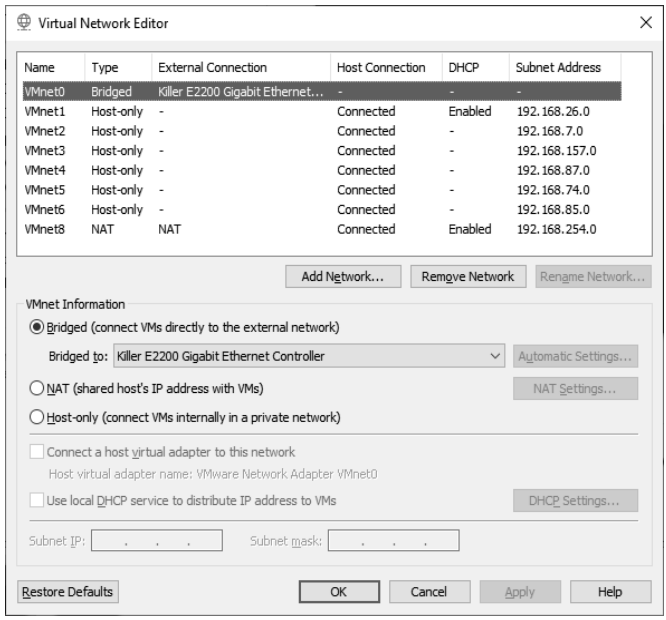
Rysunek 12.5. *Okno kreatora tworzenia nowej maszyny wirtualnej*

Przypisz do maszyny dodatkowy interfejs sieciowy. Pierwszy interfejs posłuży do komunikacji z internetem w celu pobierania dodatków, które za chwilę doinstalujesz, natomiast drugi interfejs posłuży do komunikacji z siecią lokalną lub inną siecią, jeśli takiej będziesz chciał użyć. W oknie *Hardware* dodaj nową kartę sieciową. Na poniższym rysunku (rysunek 12.6) w polu *Custom* wybrano interfejs `VMnet0`.

Jeśli chodzi o interfejs `VMnet0`, to wcześniej należy dostosowac go do komunikacji z odpowiednią siecią. W moim przypadku, jak możesz zauważyć na poniższym rysunku (rysunek 12.7), interfejs wirtualny współpracuje jako interfejs zmostkowany z rzeczywistym interfejsem sieciowym. Dzięki temu maszyna wirtualna otrzymuje dostęp do mojej sieci LAN.

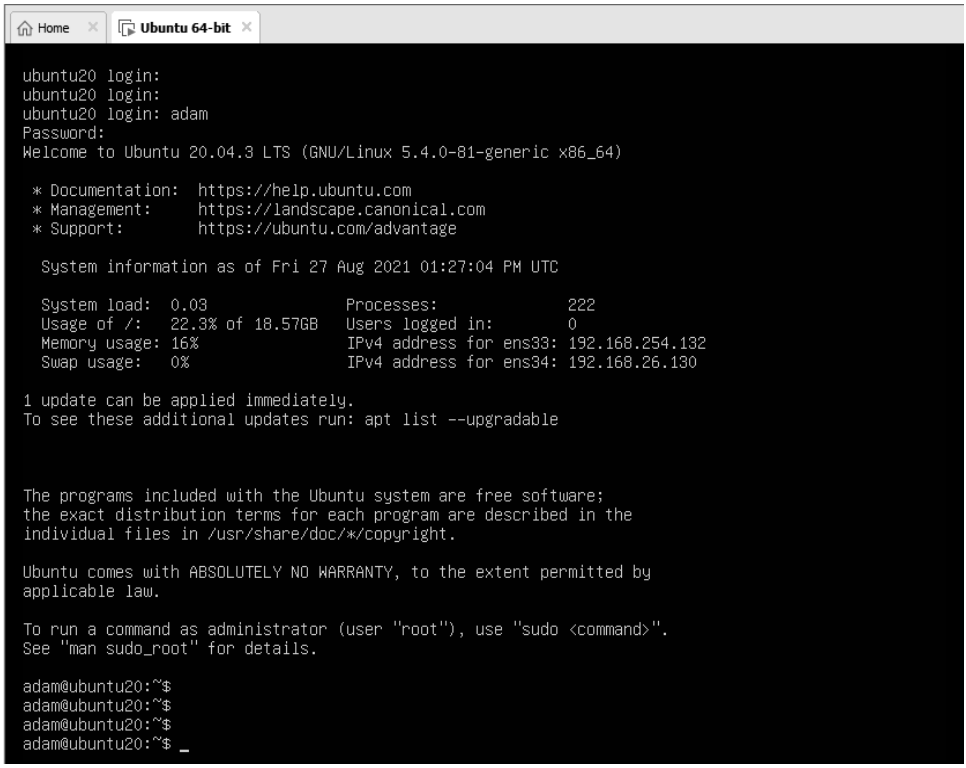


Rysunek 12.6. *UStawienia sieci wirtualnej maszyny*



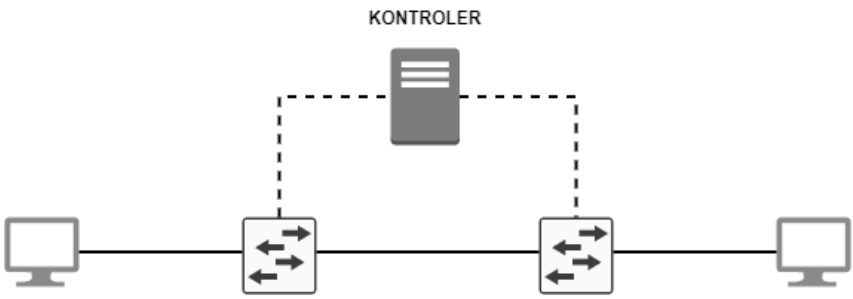
Rysunek 12.7. *Ustawienia sieci hypervisora*

Tak więc po uruchomieniu maszyny wirtualnej rozpocznie sie proces instalacji serwera Ubuntu. Wykorzystaj domyślne ustawienia, aby po instalacji uruchomić widoczną poniżej konsolę (rysunek 12.8).



Rysunek 12.8. *Zainstalowany system operacyjny Ubuntu*

Będziemy emulować sieć widoczną na poniższym rysunku 12.9. Przełączniki, których możesz użyć, mogą być rzeczywistym sprzętem, ale należy pamiętać o tym, aby obsługiwały OpenFlow. Natomiast z racji tego, że wkraczamy w wirtualny świat, wprowadzimy rozwiązanie, które nazywa się Mininet. Jest to niewielka dystrybucja Linuksa, za pomocą której możesz utworzyć wirtualne przełączniki obsługujące OpenFlow. Również stacje robocze zostaną emulowane w Mininet.



Rysunek 12.9. *Schemat sieci z kontrolerem*

Jeśli chodzi o kontroler, którego będziemy używali, to będzie nim zainstalowane na serwerze Ubuntu darmowe oprogramowanie OpenDaylight. Jest to kontroler SDN oparty na Linuksie, którego podstawy poznasz w tym przykładzie.

Zacznijmy od początku, czyli najpierw po zainstalowaniu serwera Ubuntu sprawdź, czy ma połączenie z siecią lokalną oraz internetem. Najlepiej, jeśli w konsoli wydasz komendę `ifconfig`. Jeśli komenda nie działa, doinstaluj odpowiednie narzędzia komendą `sudo apt install net-tools`.

Poniższy listing prezentuje ustawienie interfejsów na maszynie wirtualnej. Serwer Ubuntu jest więc dostępny w tym przypadku w sieci lokalnej przez interfejs `ens34`. Adres został przydzielony automatycznie przez serwer DHCP sieci LAN. Ale nic nie stoi na przeszkodzie, aby adres dodać manualnie.

```
adam@ubuntus:~$ ifconfig
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 192.168.254.134  netmask 255.255.255.0  broadcast 192.168.254.255
        inet6 fe80::20c:29ff:fead:f73f  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:ad:f7:3f  txqueuelen 1000  (Ethernet)
<POMINIĘTO>
ens34: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 172.30.100.96  netmask 255.255.255.0  broadcast 172.30.100.255
        inet6 fe80::20c:29ff:fead:f749  prefixlen 64  scopeid 0x20<link>
        ether 00:0c:29:ad:f7:49  txqueuelen 1000  (Ethernet)
<POMINIĘTO>
lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
<POMINIĘTO>
adam@ubuntus:~$
```

Kolejnym istotnym krokiem jest instalacja Javy. Trzeba to robić uważnie, gdyż w przeciwnym wypadku mogą pojawić się problemy w trakcie instalacji. Zanim cokolwiek zainstalujemy, zaktualizuj system i pakiety komendą `sudo apt-get -y update`, a po chwili wydaj komendę `sudo apt-get -y upgrade`. System jest zaktualizowany, to jeszcze na samym początku zainstaluj komendą `sudo apt-get -y install unzip` program ZIP do rozpakowywania archiwów; za chwilę się przyda.

Teraz mamy już wszystkie niezbędne narzędzia, możesz więc przejść do instalacji Javy. Ponieważ zainstalujemy OpenDaylight w wersji obsługującej Javę 8, dlatego zacznij konfigurację od komendy `sudo apt-get -y install openjdk-8-jre`. Pobieranie i instalacja pakietów może trochę potrwać, więc należy cierpliwie czekać.

Następnie warto sprawdzić komendą `sudo update-alternatives --config java`, czy nie ma już na serwerze innych wersji Javy. To bardzo ważne, aby później w konfiguracji była zachowana pewna spójność. Gdyby były inne wersje, zostaniesz poproszony o wybranie tej, której będziesz używał. Zwykle odbywa się to poprzez podanie właściwego numeru przypisanego dla danej wersji.

Komendą `ls -l /etc/alternatives/java` sprawdź, jaka wersja aktualnie znajduje się w systemie. Otrzymasz ścieżkę dostępu, której za chwilę użyjesz do dalszej konfiguracji.

```
adam@ubuntus:~$ ls -l /etc/alternatives/java
lrwxrwxrwx 1 root root 46 Aug 28 10:53 /etc/alternatives/java -> /usr/lib/jvm/java-8-
openjdk-amd64/jre/bin/java
adam@ubuntus:~$
```

Podczas konfiguracji OpenDaylight wymagane jest ustawienie określonej zmiennej środowiskowej zawierającej ścieżkę do zestawu narzędzi Javy. Tak więc komendą `echo 'export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64' >> ~/.bashrc` należy dokonać tych ustawień. Komenda `source ~/.bashrc` spowoduje zaimplementowanie pliku źródłowego `bashrc`. Ten sam efekt uzyskasz, wylogowując się i ponownie logując. Ale przedstawioną komendą zrobisz to po prostu szybciej.

```
adam@ubuntus:~$ echo 'export JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64' >> ~/.bashrc
adam@ubuntus:~$ source ~/.bashrc
adam@ubuntus:~$
```

Aby sprawdzić ustawienia, wydaj komendę `echo $JAVA_HOME`. Jak widzisz, ścieżka jest taka, jaką podałeś przed chwilą.

```
adam@ubuntus:~$ echo $JAVA_HOME
/usr/lib/jvm/java-8-openjdk-amd64
adam@ubuntus:~$
```

Java gotowa, dlatego teraz zainstalujemy OpenDaylight. Jeśli przejdziesz na stronę tego projektu, możesz wybrać kilka wersji. Mając na celu jedynie demonstrację, nie jest istotne, jaką wersję pobierzesz. Warto jednak wcześniej upewnić się, jaka wersja Javy jest potrzebna. Poniżej znajduje się listing z wersją, która została użyta w tym przykładzie. Po wprowadzeniu odpowiedniego polecenia rozpocznie się pobieranie archiwum ZIP. Jeśli chcesz pobrać inną wersję, po prostu przejdź na stronę projektu i zapisz ścieżkę do odpowiedniego archiwum ZIP, a następnie wklej ścieżkę za komendą `wget`.

```
adam@ubuntus:~$ wget
https://nexus.opendaylight.org/content/groups/public/org/opendaylight/integration/distributi
on-karaf/0.5.0-Boron/distribution-karaf-0.5.0-Boron.zip
--2021-08-28 11:08:13--
https://nexus.opendaylight.org/content/groups/public/org/opendaylight/integration/distributi
on-karaf/0.5.0-Boron/distribution-karaf-0.5.0-Boron.zip
Resolving nexus.opendaylight.org (nexus.opendaylight.org)… 199.204.45.87,
2604:e100:1:0:f816:3eff:fe45:48d6
Connecting to nexus.opendaylight.org (nexus.opendaylight.org)|199.204.45.87|:443… connected.
HTTP request sent, awaiting response… 200 OK
Length: 425577192 (406M) [application/zip]
Saving to: 'distribution-karaf-0.5.0-Boron.zip'
araf-0.5.0-Boron.zip     16%[===>                        ]  68.81M  13.9MB/s    eta 30s
```

Po pobraniu pliku sprawdź, jak nazywa się archiwum. Komendą `ls` sprawdzisz zawartość kontenera.

```
adam@ubuntus:~$ ls
distribution-karaf-0.5.0-Boron.zip
adam@ubuntus:~$
```

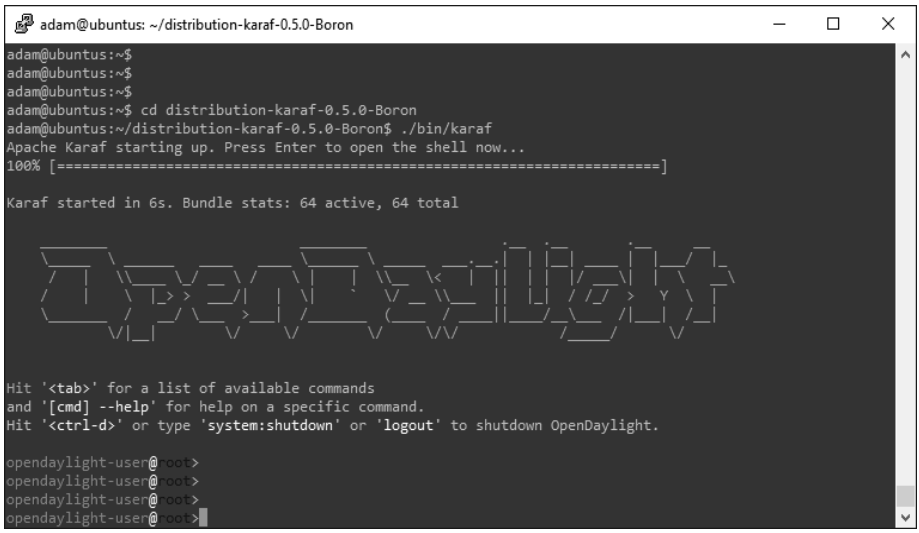
Archiwum ZIP znajduje się na swoim miejscu, dlatego rozpakuj je komendą `unzip`.

```
adam@ubuntus:~$ unzip distribution-karaf-0.5.0-Boron.zip
```

Po przejściu do rozpakowanego folderu komendą `./bin/karaf` uruchom OpenDaylight.

```
adam@ubuntus:~$ cd distribution-karaf-0.5.0-Boron
adam@ubuntus:~/distribution-karaf-0.5.0-Boron$ ./bin/karaf
```

Poniższy rysunek (rysunek 12.10) przedstawia uruchomione środowisko pracy wraz z linią komend.



Rysunek 12.10. *Uruchomione środowisko OpenDaylight*

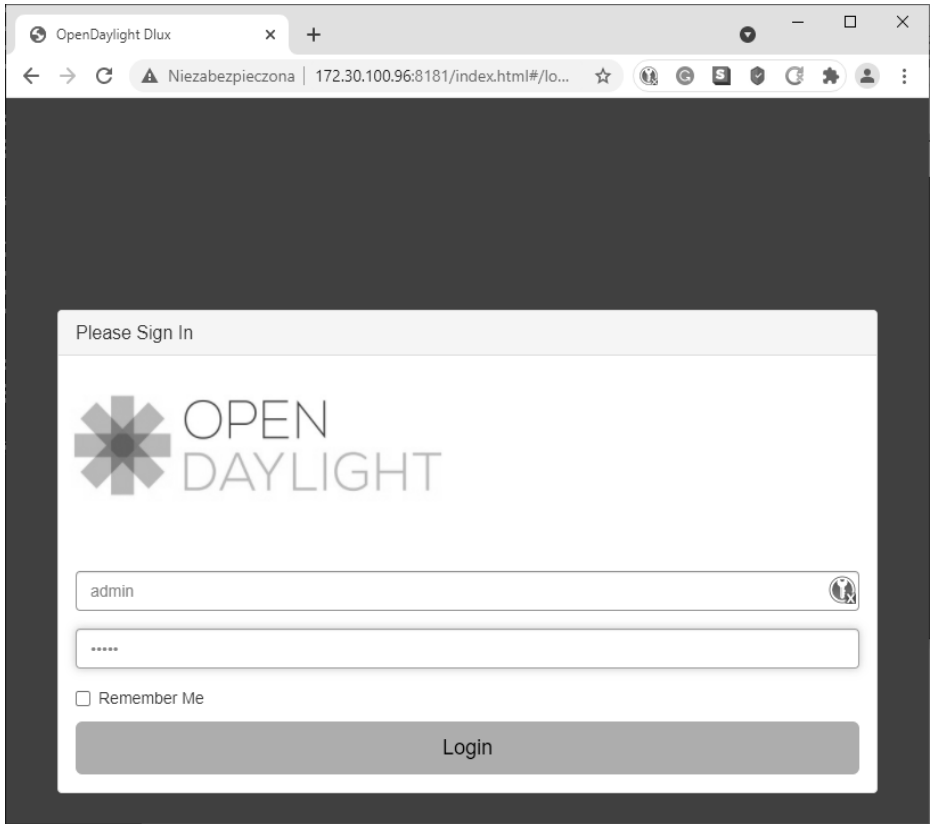
Domyślnie pobrana paczka danych zawiera wszystko, co potrzebne do uruchomienia kontrolera SDN. Jednak funkcjonalności nie są domyślnie zainstalowane, dlatego trzeba to zrobić. Serwer WWW również nie działa domyślnie, dlatego nie można połączyć się z kontrolerem przez stronę WWW. Jeśli chcesz wyświetlić dostępne funkcjonalności, wydaj komendę `feature:list`. Możesz również na stronie projektu zapoznać się ze znaczeniem każdej z funkcji. Na stronie znajduje się wyczerpujący opis.

Do naszych celów będą nam potrzebne najważniejsze funkcje — odpowiedzialne za serwer WWW oraz obsługę przełączników L2. Aby doinstalować odpowiednie funkcje, użyj komendy `feature:install`. Instalacja może potrwać kilka minut, ale na ekranie nie będzie widoczny żaden pasek postępu, dlatego musisz cierpliwie poczekać.

```
opendaylight-user@root>feature:install odl-restconf odl-l2switch-switch odl-mdsal-apidocs
odl-dlux-all
opendaylight-user@root>
```

Po instalacji wymienionych funkcji możesz przejść do przeglądarki, wpisać adres IP, który ma maszyna Ubuntu. W podanym przykładzie będzie to *http://172.30.100.96:8181/index.html*.

Na ekranie pojawi się okno logowania (rysunek 12.11). Domyślnie użytkownik to `admin` z hasłem `admin`.



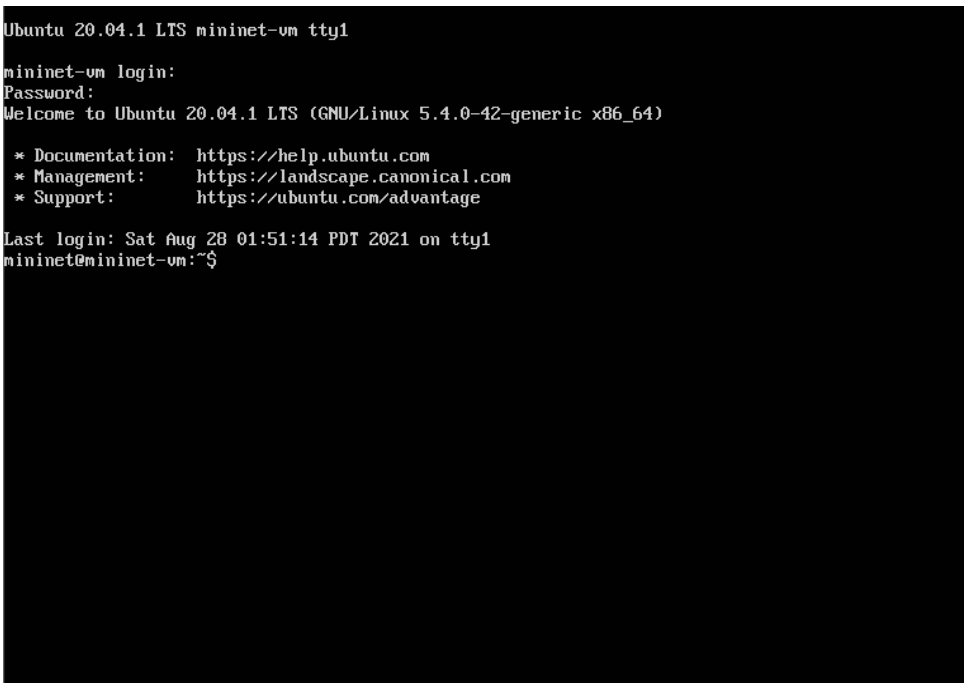
Rysunek 12.11. *Okno logowania środowiska GUI*

Po zalogowaniu na ekranie pojawi się puste białe okno z przyciskiem `Reload`. Na razie wszystko idzie zgodnie z planem i nie musisz się przejmować, że okno jest puste. Dzieje się tak, ponieważ nie ma utworzonej żadnej topologii służącej do zarządzania. Czyli z kontrolerem nie połączyło się żadne urządzenie. Dlatego kontroler nie ma co robić. Za chwilę to zmienimy, bo wspomniałem Ci, że użyjemy do tego celu oprogramowania Mininet.

Aby zaimplementować w środowisku wirtualnym maszynę Mininet, przejdź do strony tego projektu — *mininet.org*. Znajduje się tam gotowe środowisko pracy. Wcześniej, podobnie jak poprzednio, przygotuj odpowiedni interfejs, aby Mininet bez problemu komunikował się z kontrolerem. Aby zalogować się do Mininet, użyj konta użytkownika o nazwie `mininet` z tym samym hasłem (rysunek 12.12).

Sprawdź adres IP komendą `ifconfig` i wykonaj test `ping` do kontrolera.

```
mininet@mininet-vm:~$ ping 172.30.100.96
PING 172.30.100.96 (172.30.100.96) 56(84) bytes of data.
64 bytes from 172.30.100.96: icmp_seq=1 ttl=64 time=0.597 ms
64 bytes from 172.30.100.96: icmp_seq=2 ttl=64 time=0.246 ms
64 bytes from 172.30.100.96: icmp_seq=3 ttl=64 time=0.372 ms
^C
```



Rysunek 12.12. *Konsola Mininet*

```
--- 172.30.100.96 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2019ms
rtt min/avg/max/mdev = 0.246/0.405/0.597/0.145 ms
mininet@mininet-vm:~$
```

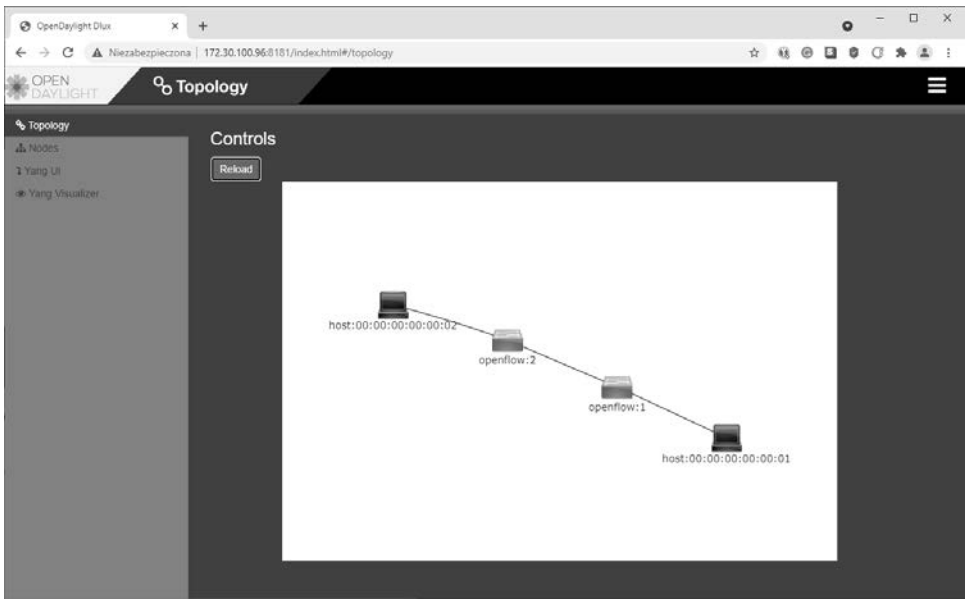
Komunikacja działa, więc możesz przejść do konfigurowania pierwszej topologii. Na początku ustaliliśmy wygląd topologii, dlatego teraz wydaj komendę, która ją utworzy.

```
mininet@mininet-vm:~$ sudo mn --topo linear,2 --mac --
controller=remote,ip=172.30.100.96,port=6633 --switch ovs,protocols=OpenFlow10
*** Creating network
*** Adding controller
Unable to contact the remote controller at 172.30.100.98:6633
*** Adding hosts:
h1 h2
*** Adding switches:
s1 s2
*** Adding links:
(h1, s1) (h2, s2) (s2, s1)
*** Configuring hosts
h1 h2
*** Starting controller
c0
*** Starting 2 switches
s1 s2 …
*** Starting CLI:
mininet>
```

Od razu możesz wykonać test połączenia dla wszystkich, używając komendy `pingall`. Jak widać na poniższym listingu, połączenia działają prawidłowo.

```
mininet>pingall
*** Ping: testing ping reachability
h1 -> h2
h2 -> h1
***Results: 0% dropped (2/2 received)
mininet>
```

Następnie wróć do przeglądarki, w której otwarte jest okno OpenDaylight, i kliknij niebieski przycisk `Reload` (rysunek 12.13). Powinieneś połączyć się z Mininet i utworzoną topologią. Na ekranie możesz zauważyć połączenie dwóch stacji roboczych oraz pomiędzy nimi dwa przełączniki — dokładnie tak miało to wyglądać. Pamiętaj, że teraz kontroler umożliwia konfigurację przełączników.

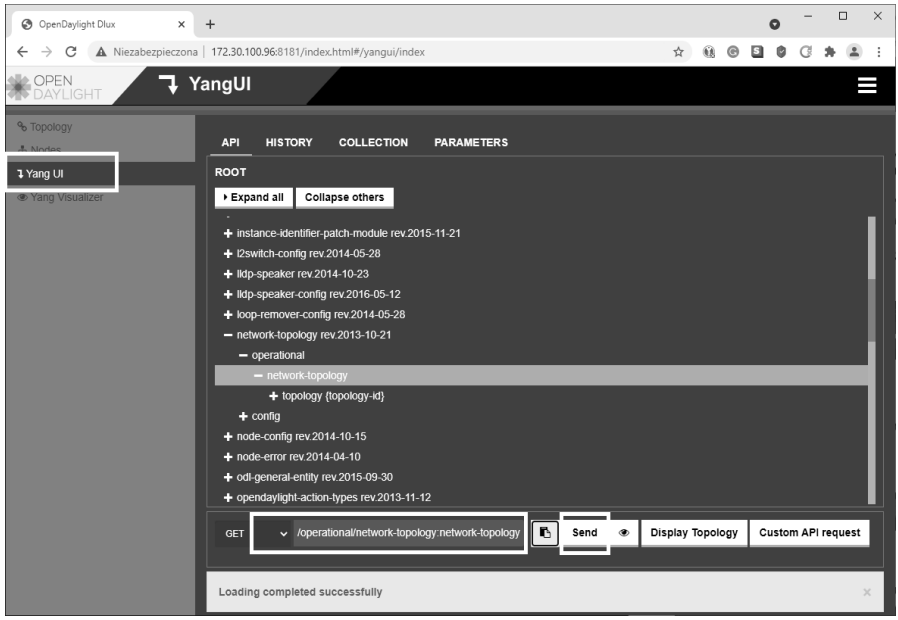


Rysunek 12.13. *Utworzona topologia*

Oczywiście to tylko podstawowy przykład na początek możliwości, jakie daje Mininet z kontrolerem SDN. Topologie, które możesz budować w ten sposób, mogą być naprawdę imponujące.

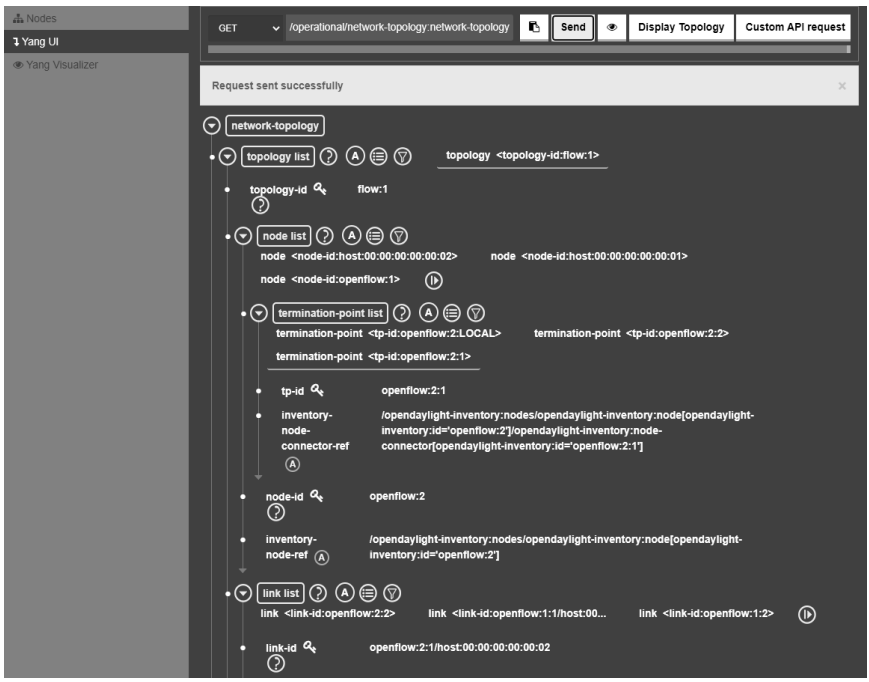
Jeśli chcesz wykorzystać możliwości, jakie daje `NETCONF`, możesz wykorzystać wbudowany w OpenDaylight interfejs o nazwie `Yang`. Jest to coś w rodzaju interfejsu API dostępnego przez HTTP i umożliwiającego dostęp do wykreowanych przez Ciebie danych dotyczących konfiguracji oraz infrastruktury. Nie będę szczegółowo omawiał jego funkcjonalności, a jedynie wskażę, jak dostać się do danych infrastruktury i sprawdzić je w formacie nieco bardziej przyjaznym.

Po lewej stronie okna OpenDaylight kliknij pozycję `Yang UI`. Następnie odszukaj na liście pozycję `network –topology`. Skopiuj adres i kliknij przycisk `Send` (rysunek 12.14).



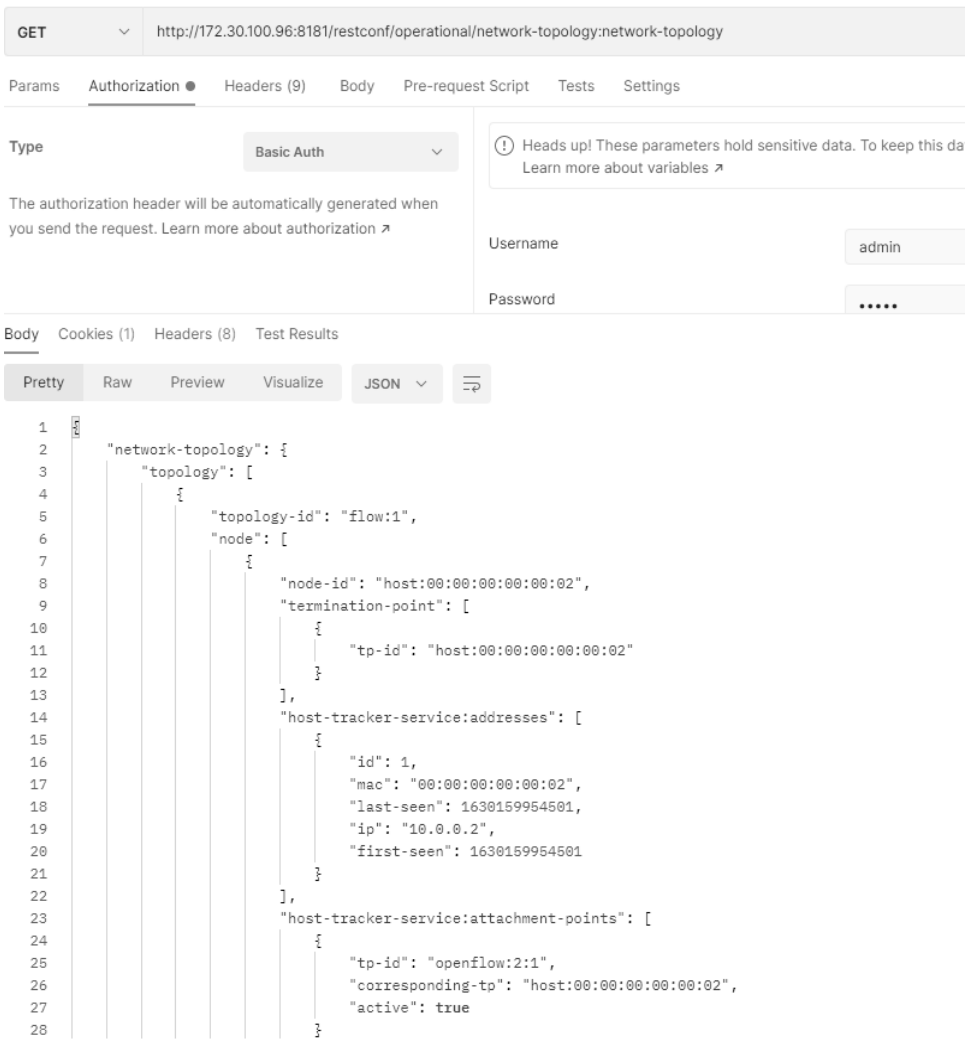
Rysunek 12.14. *Dane dotyczące urządzeń pogrupowane i udostępnione w formie drzewa*

Zauważ, że prezentowane dane zawierają informacje o urządzeniach, które znalazły się w topologii, ich dane adresowe i nazwy (rysunek 12.15).



Rysunek 12.15. *Wynik polecenia wydanego w Yang UI*

Na koniec możesz wykorzystać aplikację Postman jako wsparcie do pracy nad danymi czy komunikację z kontrolerem. Wystarczy, jeśli wcześniej skopiowany adres wkleisz w oknie Postman, otwierając nowy projekt, i się uwierzytelnisz, a będziesz mógł wyświetlić identyczne dane w różnych formatach, np. JSON. Dodatkowo Postman daje możliwość wysyłania żądań do kontrolera przygotowanych za pomocą interfejsu API RESTCONF. Poniżej przedstawiono okno środowiska Postman (rysunek 12.16).



Rysunek 12.16. *Okno środowiska Postman*

Wirtualizacja

Wirtualizacja, automatyzacja sieci i programowalność sieci to pojęcia, które mają w zasadzie jeden cel: uproszczenie. Automatyzacja sieci znacznie ułatwia administratorom konfigurowanie sieci i zarządzanie nimi. Dzięki wdrożeniu odpowiedniego oprogramowania można zautomatyzować wiele codziennych czynności (choć było to już możliwe za sprawą różnego rodzaju skryptów). Poprzez pełną automatyzację sieci można jednak wpływać już nie tylko na konfigurację, ale również na działanie całej sieci, która może automatycznie wykrywać na przykład obciążenie jednych urządzeń i kierować ruch przez inne, mniej w danym momencie obciążone.

Usługi chmury

Obecnie szybkość działania internetu jest bardzo duża, co daje nowe możliwości. W typowym modelu przedsiębiorstwo posiada własną serwerownię z serwerami, oprogramowaniem i urządzeniami aktywnymi. Posiadanie własnej serwerowni wymaga jednak dbania o bezpieczeństwo, regularnego sporządzania kopii zapasowych i zatrudnienia wykwalifikowanego personelu, który wszystko skonfiguruje.

Alternatywą jest wykorzystanie chmury (ang. *cloud*), czyli wirtualnego środowiska pracy, które jest umieszczone poza siedzibą przedsiębiorstwa i nie stanowi jego własności. Polega to na tym, że możesz wynająć sobie dowolną usługę, program czy inne rozwiązanie, które będzie dostępne przez internet. Oczywiście do zarządzania usługą i wykorzystywania jej potrzebny będzie dostęp do sieci.

Odpowiednie wdrożenie usług chmurowych (ang. *cloud computing*) w przedsiębiorstwie może zmniejszyć koszty operacyjne. Daje również wiele możliwości, takich jak dostęp do danych w dowolnym miejscu i czasie. Umożliwia korzystanie tylko z wybranych usług w ramach danej organizacji, ponadto w zależności od rodzaju usług chmurowych zmniejsza zapotrzebowanie firmy na sprzęt fizyczny czy określone usługi wymagające licencji. Jeśli przedsiębiorstwo decyduje się na dzierżawę usług chmurowych, często nie musi ponosić już kosztów utrzymania swojego środowiska chmurowego. Nie musi też szkolić personelu i dbać o bezpieczeństwo systemów i regularne sporządzanie ich kopii. Jeśli pojawią się potrzeby większych możliwości fizycznych czy pojemnościowych, wówczas można je w ciągu kilku minut zwiększyć bez wychodzenia z firmy.

Wyróżniamy trzy podstawowe usługi dostępne w chmurze.

Oprogramowanie jako usługa (ang. *Software as a Service* — SaaS) to usługa polegająca na tym, że aplikacja jest przechowywana i uruchamiana na komputerach dostawcy usługi i jest dostępna dla użytkownika przez internet. Typowym przykładem takiego rozwiązania jest Office 365. Nie musisz w tym przypadku instalować żadnego oprogramowania na komputerze lokalnym, gdyż edytor tekstowy umieszczony jest na zdalnych komputerach dostawcy, który zapewnia działanie usługi i jej bezpieczeństwo, a użytkownik otrzymuje jedynie dane do logowania do usługi.

Platforma jako usługa (ang. *Platform as a Service* — PaaS) to usługa, która polega na tym, że usługodawca udostępnia odbiorcy przez internet dostosowaną do jego potrzeb platformę z określonym zestawem narzędzi służących do różnych celów: projektowania, pisania i testowania nowych aplikacji. W ten sposób odbiorca otrzymuje gotowe środowisko pracy bez ponoszenia dodatkowych kosztów związanych z wdrożeniem i utrzymaniem go w swojej firmie.

Infrastruktura jako usługa (ang. *Infrastructure as a Service* — IaaS) to z kolei usługa polegająca na udostępnieniu odbiorcy przez internet pełnej infrastruktury informatycznej. W tym przypadku możesz wynająć całą wirtualną serwerownię z określonymi systemami operacyjnymi. Za realizację wszystkich zadań związanych z wykonywaniem kopii bezpieczeństwa i zapewnieniem bezpieczeństwa takiej serwerowni odpowiedzialny jest dostawca. Dostawca zarządza również systemem operacyjnym całej chmury i odbiorca nie musi się tym przejmować.

Ze względu na oferowane usługi można wyróżnić różne rodzaje chmur, które różnią się zastosowaniem oraz rodzajem użytkownika końcowego. Pierwszy rodzaj to chmury publiczne (ang. *public cloud*). Jest to usługa, z której może korzystać każdy użytkownik, np. zakładając darmowe konto. Przeważnie taka usługa oferuje kilka gigabajtów przestrzeni dyskowej i zapewnia podstawowe funkcjonalności z tym związane. Oczywiście usługi chmurowe publiczne zawsze wymagają dostępu do internetu. Drugi rodzaj to chmura prywatna (ang. *private coud*). Jest to chmura umożliwiająca świadczenie konkretnych usług dla danego podmiotu i mają do niej dostęp jedynie pracownicy. Niekiedy fizyczne dyski w takiej chmurze znajdują się u zewnętrznego dostawcy, a przedsiębiorstwo zarządza nią na poziomie dostępu. Można wtedy do takiej chmury logować się na przykład z uprawnieniami użytkownika domeny pomimo tego, że fizycznie infrastruktura znajduje się u dostawcy. Tworzy się także rozwiązanie hybrydowe (ang. *hybrid cloud*), w którym pracownik ma dostęp do zasobów chmury służbowej znajdującej się w firmie oraz do zasobów prywatnych udostępnianych przez dostawcę zewnętrznego. Czasem klientom umożliwia się skorzystanie z chmury publicznej na przykład w celu bezpiecznego przesłania danych.

Jeszcze do niedawna, jeśli określony dział w firmie miał potrzebę przetestowania konkretnego rozwiązania, np. programistycznego, zwykle był zmuszony do utworzenia odpowiedniej maszyny wirtualnej samodzielnie lub przekierowania prośby o jej utworzenie do działu odpowiedzialnego za wirtualizację. Oczywiście wszystko trzeba było przygotować i udostępnić. Omawiane wyżej usługi znacznie przyspieszają takie potrzeby. Poprzez odpowiednią usługę stosowny dział jest w stanie utworzyć potrzebne środowisko testowe kilkoma kliknięciami, wybierając wszystko to, co jest na danej maszynie konieczne. Nie ma już potrzeby, aby inny człowiek był za ten proces odpowiedzialny (za sam proces tworzenia maszyny).

Ciekawe jest to, że chmura może być podwójnie niebezpieczna. Dzieje się tak, jeśli musisz zabezpieczyć chmurę i dostęp do niej. Wiem, brzmi dziwnie, ale tak naprawdę chodzi o środki zapewniające bezpieczeństwo aplikacjom i zasadom, jakie panują podczas przetwarzania danych w chmurze.

Podczas korzystania ze środowiska chmurowego czy administrowania nim istnieje szereg zasad i praktyk, które prawidłowo wdrożone znacząco zmniejszają ryzyko pojawienia się niebezpiecznych sytuacji. Standaryzowaniem tych praktyk zajmuje się organizacja CSA (*Cloud Security Alliance*). Jeśli jesteś zainteresowany tym tematem, możesz przejść do strony tej organizacji: *https://cloudsecurityalliance.org*.

Tak naprawdę w zakresie bezpieczeństwa środowiska chmurowe są narażone na identyczne niebezpieczeństwa, na jakie narażona jest typowa serwerownia. Środowisko chmurowe wykorzystuje bowiem normalny sprzęt i oprogramowanie, które nim zarządza. Jedyną różnicą jest odpowiedzialność. W przypadku własnej serwerowni odpowiedzialność, rzecz jasna, spada na administratora, w środowisku chmurowym to dostawca musi o nią zadbać.

Podczas korzystania ze środowiska chmurowego warto również przyjąć zawsze założenie ograniczonego zaufania oraz przypomnieć sobie o maksymie „ufaj i kontroluj". Pomimo że chmura jest ciekawym rozwiązaniem, warto pamiętać o tym, że wdrożenie infrastruktury lokalnej sprawia, że między systemami znajdującymi się blisko siebie występujące opóźnienie jest zawsze znacząco mniejsze w porównaniu z infrastrukturą w chmurze. To czasem może być kluczowa sprawa dla jakości usług, więc warto o tym pamiętać.

Maszyny wirtualne

Kilka lat temu pomieszczenia serwerowni wyglądały niemal identycznie jak obecnie. W specjalnych szafach było wiele komputerów, które miały wiele dysków, mrugały diody i wszystko działało. Urządzenia zużywały dużo prądu i oddawały dużo ciepła, które trzeba było za pomocą klimatyzatorów schłodzić. Bardzo często dla każdej usługi był przeznaczony osobny fizyczny serwer. Tak więc były serwery pocztowe, serwery webowe, serwery plików i serwery bazodanowe oraz np. serwer Active Directory, w ramach którego można było uruchamiać więcej usług, takich jak DHCP czy RADIUS. Często zasoby urządzeń fizycznych nie były w pełni wykorzystywane, a nie zawsze można było uruchomić na serwerze kilka usług. Po wdrożeniu wirtualizacji fizyczny serwer umożliwia włączenie na nim wielu maszyn wirtualnych, które wykorzystują przydzielone im zasoby fizyczne serwera bezpośrednio, bez pośrednictwa systemu operacyjnego. Na fizycznym urządzeniu można zatem uruchomić kilka systemów operacyjnych jednocześnie i zainstalować w tych systemach dowolne usługi.

Dzięki wirtualizacji serwerownia wymaga mniej sprzętu fizycznego. A im mniej fizycznego sprzętu, tym mniej sprzętu sieciowego, który musi to wszystko obsłużyć. Zmniejsza się również wydzielanie ciepła w serwerowni, chłodzenie staje się wydajniejsze. Można zaoszczędzić na samym zużyciu energii elektrycznej, choć tutaj trzeba być ostrożnym, czasem bowiem jeden serwer z kilkoma maszynami wirtualnymi może pochłaniać podobną ilość energii. Zdecydowanym plusem jest to, że obecnie sprzęt nie zajmuje tyle miejsca w serwerowni co kiedyś. Bardzo wydajny serwer zajmuje jedną czwartą szafy teleinformatycznej, a może obsłużyć bardzo dużą liczbę wirtualnych maszyn.

Oczywiście wszystko zaczęło się od odpowiedniego wsparcia dla urządzeń fizycznych. Technologie, które powstały, znacząco zwiększyły możliwości dzisiejszych procesorów. Jak wiesz, dziś nawet w nieco tańszych modelach laptopów dostępne są procesory zaopatrzone w kilka rdzeni. Dzięki temu możliwości komputerów osobistych są znacznie większe. Procesory przeznaczone dla serwerów charakteryzują się jeszcze większymi możliwościami, więc tym bardziej jest w nich potencjał do wykorzystania ich mocy w maszynach wirtualnych.

Wirtualizacja jest ogromnym ułatwieniem, jeśli chodzi o naukę czy testowanie nowych rozwiązań czy aplikacji. Kiedyś, aby przeprowadzić testy, trzeba było instalować odrębny serwer, potem go konfigurować i w razie niepowodzenia czynności powtarzać. Dzięki wirtualizacji możesz utworzyć takich testowych środowisk setki w kilka minut. Jeśli coś pójdzie nie tak, wystarczy wykasować plik i wgrać czysty, już skonfigurowany, aby powrócić do wszystkich ustawień.

Z maszyn wirtualnych bardzo szybko i wygodnie tworzy się kopie bezpieczeństwa i szybkie kopie migawkowe (ang. *snapshot*). Nawet jeśli ktoś nie ma opracowanego systemu kopii bezpieczeństwa, może łatwo utworzyć kopię całego systemu. Zaletą wirtualizowania jest też to, że w razie problemów z urządzeniem fizycznym maszyny wirtualne można dosyć łatwo przenieść na inny serwer. Oczywiście nie zawsze jest to możliwe i bywa, że się nie udaje, ale to naprawdę rzadkie przypadki.

Jak już wspomniałem, maszyna wirtualna wykorzystuje zasoby sprzętowe komputera, na którym jest implementowana. Jak więc działa taka maszyna wirtualna?

Na wirtualnej maszynie, pomimo tego, że jak jej nazwa wskazuje, jest wirtualna, pracuje system operacyjny, który potrzebuje określonych zasobów do działania, tak jakby był zainstalowany na fizycznym urządzeniu. Samo działanie systemu operacyjnego nie zmienia się, dlatego wymaga on dysku twardego, pamięci, procesora, a jeśli potrzebny jest dostęp do sieci — również karty sieciowej.

Maszyna fizyczna ma te zasoby i musi się nimi podzielić. Oznacza to, że im więcej wirtualnych maszyn uruchomisz na serwerze fizycznym, tym więcej będzie „chętnych" na te ograniczone przecież zasoby fizyczne.

Pomiędzy maszyną wirtualną a fizycznym komputerem funkcjonuje pewnego rodzaju pośrednik, tzw. hipernadzorca (ang. *hypervisor*). To za jego pomocą maszynie wirtualnej przydzielane są odpowiednie zasoby, które administrator określa w ustawieniach. Jest to nic innego jak oprogramowanie służące do wykorzystania zasobów i tworzenia nowych wirtualnych maszyn. Generalnie możemy wyróżnić dwa typy hypervisorów. Pierwszy typ to coś w rodzaju dostępu bezpośredniego, w którym program działa od razu na sprzęcie. Czyli jest jakby zainstalowany zamiast systemu operacyjnego i dzięki temu ma bezpośredni dostęp do fizycznego sprzętu. Przykładem jest np. VMware ESXi czy Microsoft Hyper-V. Na przykład VMWare wykorzystuje do tego celu rozwiązanie VMkernel.

Drugi typ działa już w określonym systemie operacyjnym. Czyli np. masz zainstalowany system Windows i na nim instalujesz dopiero np. VMware Workstation lub VirtualBox. W tym rozwiązaniu to system operacyjny Windows ma bezpośredni dostęp do zasobów, a nie hypervisor.

Najmniejszy problem jest z dyskiem twardym. Maszyna wirtualna zapisywana jest na fizycznym dysku jako jeden lub kilka plików. Pliki mogą być zapisywane w dowolnym miejscu. Jest to więc bardzo komfortowa sytuacja, gdyż dodanie nowego fizycznego dysku nie jest kosztowne i jest bardzo łatwe. Dzięki temu maszyny wirtualne mogą mieć wiele miejsca.

Maszyna wirtualna korzysta również z fizycznej pamięci, która jest przydzielana zgodnie ze wskazaniami administratora. Pamięć może być przydzielana dynamicznie, co oznacza, że jeśli jedna maszyna wirtualna aktualnie nie potrzebuje zasobów, może z nich skorzystać inna.

Kwestia procesorów nie jest bynajmniej trudna do zrozumienia. Jeżeli posiadasz czterordzeniowy procesor, to tak, jakbyś miał cztery procesory w jednej kości CPU. Kiedy jest to procesor, który wspiera wielowątkowość, wówczas na każdy rdzeń przypadają na przykład dwa wątki i procesor może w danym momencie wykonać osiem operacji. Oprogramowanie wirtualizacyjne może skorzystać z tego faktu i dzięki temu każda maszyna otrzyma jeden własny wirtualny procesor, zwany vCPU.

Jeśli chodzi o aspekt sieciowy i wirtualne karty sieciowe (vNIC), to tworzone są wirtualne (programowe) interfejsy, które wykorzystują rzeczywistą kartę sieciową do komunikacji z rzeczywistą siecią. W większych rozwiązaniach kilka maszyn wirtualnych można podłączyć do wirtualnego przełącznika, a ten za pomocą połączenia `trunk` podłączyć do rzeczywistego sprzętu. Wirtualny przełącznik ma identyczne funkcje jak rzeczywisty.

W świecie wirtualizacji będziesz miał również do czynienia z pojęciem kontener (ang. *container*). Nie jest to stricte maszyna wirtualna, ale nieco ją przypomina swoim działaniem. Kontener to coś w rodzaju pudełka przechowującego aplikację i jej ustawienia. Czyli „organizm", który zawiera tylko kod i potrzebuje do działania nosiciela. Nosicielem tym jest silnik kontenerów (ang. *container engine*), a ten działa zawsze w oparciu o jakiś system operacyjny. Mając aplikację, można ją zatem uruchomić za pośrednictwem kontenera w dowolnym miejscu, po prostu przesyłając taki kontener do kogoś lub udostępniając go w chmurze.

Obecnie wirtualizować można niemalże każde urządzenie sieciowe. Jeszcze kilka lat temu było nie do pomyślenia, żeby wirtualizować router, a co dopiero przełącznik czy inne urządzenie sieciowe. Teraz jest to już standardem w sieciach, w których istnieje wiele wirtualnych maszyn oraz relacji pomiędzy nimi. Funkcjonalność wirtualizacji urządzenia sieciowego jest częścią ogólnej funkcjonalności znanej jako funkcja sieci wirtualnej (ang. *Virtual Network Function — VNF*). Przykładem może być chociażby przełącznik wirtualny CSR1000V czy ASAv. Tak więc nie kupujesz fizycznego urządzenia, ale po prostu oprogramowanie pełniące jego rolę i oferujące większość z funkcji, jakie są dostępne na zwykłym fizycznym sprzęcie. Zalety wykorzystania takiego sprzętu są znaczące. Samo uruchomienie sprzętu w różnych częściach sieci nie niesie za sobą konieczności dostosowywania infrastruktury sprzętowej. Po prostu gotową maszynę wirtualną można zawsze włączyć lub skopiować, jeśli zachodzi taka konieczność.

Ze względu na mnogość rozwiązań wirtualizacyjnych zaistniała konieczność ustandaryzowania niektórych z nich. Do tej pory dana technologia funkcjonowała w obrębie urządzenia i można powiedzieć, że była zamknięta w „blaszanym pudełku". Obecnie wirtualizacja granice te zamazuje. Gdy ma się wiele rozwiązań różnych producentów, brakuje często rozwiązań, które zepną funkcje zarządzania czy monitorowania w jedną całość. Próbę ustandaryzowania podjął Europejski Instytut Norm Telekomunikacyjnych (ang. *European Telecommunications Standards Institute*).

Obecnie można wyróżnić w zasadzie cztery najważniejsze elementy całej platformy zawierającej różne rodzaje rozwiązań wirtualnych. To wspominane VNF oraz:

- Network Functions Virtualization Infrastructure (NFVI),
- NFV Management and Orchestration (MANO),
- Operations and Billing Support System (OSS/BSS).

Infrastruktura zawierająca sprzęt sieciowy, oprogramowanie, serwery i inne urządzenia to właśnie NFVI. Oczywiście sprzętem trzeba odpowiednio zarządzać. Często zarządzanie to nie tylko konfiguracja, ale również dostosowanie jej do panujących w przedsiębiorstwie procedur lub wytycznych zarządu. Funkcję koordynującą pełni więc narzędzie koordynujące zwane NFVO (*NFV Orchestrator*). Na początku książki poznałeś również pewien ogólny cykl wdrażania określonego projektu, tutaj również taki cykl może wystąpić, dlatego VNFM (*VNF Manager*) zajmuje się zarządzaniem różnego rodzaju cyklami życia rozwiązań VNF. Samym zarządzaniem od strony technicznej, czyli mierzeniem wydajności, ilości błędów itp., zajmuje się VIM (*Virtualized Infrastructure Manager*). Te trzy różne funkcjonalności są ogólnie znane jako MANO.

Ostatnim elementem wspierającym zarządzanie sprzętem fizycznym, różnego rodzaju przeprowadzaniem inwentaryzacji sprzętu czy zasobów oraz np. klientów czy kontrahentów zajmuje się OSS/BSS. Wspominam o tym, ale z punktu widzenia administratora nie są to istotne tematy. Niemniej jednak warto orientować się, co poszczególne pojęcia oznaczają.

Na przykład firma Cisco w swoim rozwiązaniu NFVI proponuje głównie zwirtualizowane produkty opierające się na dystrybucji Red Hat oraz IOS. Są to np. UCS Servers, Cisco NSO, Cisco ECS i wiele innych.

Wspominałem o urządzeniach sieciowych i ich roli w nowoczesnych rozwiązaniach dotyczących wirtualizacji. Jednak ze względu właśnie na te usługi związane z wirtualizacją rola fizycznych urządzeń znacząco się zmieniła. Jeśli kilka lat temu serwer był odpowiedzialny np. tylko za usługi związane z pocztą elektroniczną, a inny serwer za usługi wydruku, to podłączenie serwerów, zapewnienie im bezpieczeństwa oraz kopie zapasowe były na zupełnie innym poziomie niż teraz. Serwery, które teraz pełnią funkcje wirtualizacyjne, również potrzebują miejsca w serwerowni, chłodzenia itd. Mało tego, są one znacznie bardziej obciążone niż kiedyś i pełnią znacznie więcej funkcji. I tak jak wspomniałem, kiedyś serwer zapewniał jedną usługę, a teraz może ich zapewniać kilkaset i więcej. Jeśli jedna usługa, np. wydruku, nie działała w sieci, to jakoś można było sobie poradzić nawet jeden dzień. Teraz kiedy jeden serwer fizyczny nie działa, cała firma i wszystkie usługi stoją. Dlatego zarządzanie takimi serwerami, ich podłączenie do sieci czy tworzenie ich kopii bezpieczeństwa znacząco się różni.

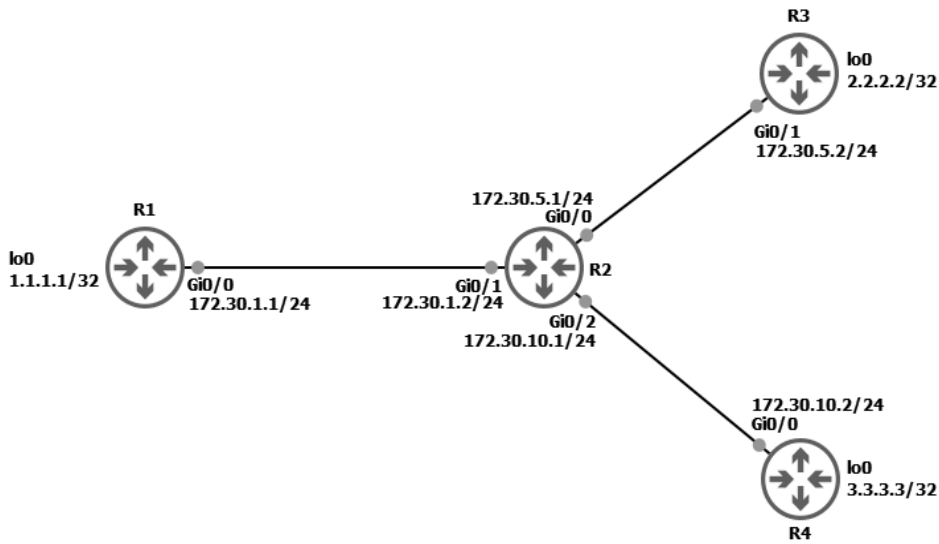
Routing w środowisku wirtualnym

Zacznijmy od tego, dlaczego w tym miejscu o tym piszę. Piszę, ponieważ funkcjonalność ta może się przydać w przypadku, kiedy posiadasz kilka miejsc, w których uruchomiony jest protokół routingu, a nie chcesz stosować innych rozwiązań L2 czy L3. Możesz również być podłączony

do sieci dostawcy i też nie interesuje Cię dodatkowa konfiguracja lub po prostu chciałbyś uruchomić routing dla innych oddziałów, z którymi określone routery nie graniczą bezpośrednio. W środowisku wirtualnym może to być propozycja godna rozważenia.

Poniższa konfiguracja zostanie przedstawiona na podstawie protokołu EIGRP, a dokładniej EIGRP Over The Top (EIGRP OTP). EIGRP do tego celu wykorzystuje coś w rodzaju tunelu opartego na LISP (*Locator Identifier Separation Protocol*). W rozwiązaniu LISP będziesz mógł zauważyć, że urządzenie pomimo tego, że będzie w innej sieci, zachowa swój własny adres IP oraz przez sieć dostawcy wymieni się z pozostałymi routerami pakietami, nawiązując zwyczajne relacje sąsiedzkie dokładnie tak, jakby te routery były bezpośrednio ze sobą połączone.

Spójrz na poniższą topologię, która jak widzisz na rysunku 12.17, obejmuje cztery routery. Router `R2` jest siecią dostawcy lub Twoją siecią, np. WAN, posiadającą odrębną infrastrukturę. Generalnie chodzi o to, aby nie ingerować w router `R2`, a tylko wykorzystać go jako transparentny podczas przekazywania pakietów pomiędzy routerami `R1`, `R3` i `R4`.



Rysunek 12.17. *Sieć WAN z czterema routerami*

Najpierw musisz mieć pewność, że routery są w stanie się ze sobą komunikować. Jest to warunek poprawnej konfiguracji EIGRP OTP. Tak więc na każdym z routerów należy wykonać przynajmniej routing statyczny. Oczywiście jak już wspominałem, omijamy router `R2`. Zauważ również, że nie dodaję całych sieci, a jedynie pojedyncze adresy IP. Rozwiązanie wymaga jedynie docelowego adresu IP, nie musisz wprowadzać całej sieci.

Poniżej znajduje się listing z konfiguracją routingu statycznego, tak aby routery mogły ustanowić pomiędzy sobą połączenie do wymiany pakietów EIGRP.

```
R1(config)#ip route 172.30.5.2 255.255.255.255 172.30.1.2
R1(config)#ip route 172.30.10.2 255.255.255.255 172.30.1.2
```

```
R3(config)#ip route 172.30.1.1 255.255.255.255 172.30.5.1
R3(config)#ip route 172.30.10.2 255.255.255.255 172.30.5.1

R4(config)#ip route 172.30.1.1 255.255.255.255 172.30.10.1
R4(config)#ip route 172.30.5.2 255.255.255.255 172.30.10.1
```

Po ustawieniu routingu statycznego możesz wykonać przykładowy test. Poniższy listing prezentuje próbę komunikacji routera R1 z adresem interfejsu routera R3. Jak widać, komunikacja przebiega prawidłowo i jest to wystarczające do tego, aby rozpocząć konfigurację EIGRP OTP.

```
R1#ping 172.30.5.2
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 172.30.5.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 5/10/15 ms
R1#
```

Jeśli chciałbyś osiągnąć np. adres interfejsu `loopback` routera R3, to zauważ, że ta komunikacja już nie działa.

```
R1#ping 2.2.2.2
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2.2.2.2, timeout is 2 seconds:
…
Success rate is 0 percent (0/5)
R1#
```

W przedstawionym rozwiązaniu router R1 będzie, można powiedzieć, routerem głównym. Chociaż to określenie nie pasuje do końca do funkcji, jaką pełni ten router. Chodzi o to, że w razie jeśli inne routery, w naszym przypadku R3 i R4, będą przesyłały do siebie pakiety, to będzie się to odbywało na tej zasadzie, że pakiety będą przesyłane do routera R1, a ten przekaże je (odbije) do pozostałych routerów. Router R1 w tym rozwiązaniu będzie pełnił funkcję „odbijacza tras" (ang. *route reflector*). Na routerze R1 będzie więc konieczna dodatkowa konfiguracja. Zacznijmy więc od routera R1. W pierwszej kolejności, aby włączyć routing EIGRP z funkcją OTP, wydaj komendę `router eigrp OTP`. Następnie komendą `address-family [ipv4 lub ipv6] autonomous-system [numer_systemu_autonomicznego]` określ, dla jakiego protokołu oraz dla jakiego systemu autonomicznego chcesz uruchomić EIGRP. Podczas konfiguracji standardowego EIGRP informacje na temat systemu autonomicznego podaje się od razu podczas uruchamiania instancji routingu. Następna komenda, `remote-neighbors source [interfejs_nasłuchujący_pakietów_LISP] unicast-listen lisp-encap [system_autonomiczny]`, wskazuje routerowi, jakiego interfejsu ma użyć do nasłuchiwania pakietów LISP unicast.

```
R1(config)#router eigrp OTP
R1(config-router)#address-family ipv4 autonomous-system 100
R1(config-router-af)#remote-neighbors source g0/0 unicast-listen lisp-encap 100
```

Dalej w konfiguracji należy za pomocą komendy `network` rozgłosić te sieci, które są konieczne. W przypadku routera R1 jest to adres interfejsu `loopback` oraz sieć bezpośrednio podłączona `172.30.1.0`.

```
R1(config-router-af)#network 1.1.1.1 0.0.0.0
R1(config-router-af)#network 172.30.1.0 0.0.0.255
```

Na routerze R1 musimy wykonać jeszcze dwie czynności. Pierwszą czynność wykonaj komendą `no next-hop-self`. Ta komenda sprawi, że router R1 nie będzie w stanie ustawić swojego własnego adresu IP jako następnego skoku dla pakietów, które nie są dla niego przeznaczone. Czyli będzie niejako transparentny. Konieczne jest również wydanie komendy `no split-horizon`. W przypadku EIGRP jest to konieczne, ponieważ w omawianej sieci komunikacja pomiędzy routerami R3 i R4 odbywa się przez router R1, a zgodnie z zasadą podzielonego horyzontu router nie może rozgłaszać sieci przez interfejs, na którym otrzymał informację o tej sieci. Dlatego routery R3 i R4 nie byłyby w stanie wymienić się trasami, gdyby ta funkcja była włączona.

```
R1(config-router-af)#af-interface g0/0
R1(config-router-af-interface)#no next-hop-self
R1(config-router-af-interface)#no split-horizon
R1(config-router-af-interface)#
```

Tak więc po konfiguracji routera R1 nadszedł czas na routery R3 i R4. W pierwszej kolejności router R3. Początek konfiguracji jest identyczny jak w przypadku R1. Na routerze R3 zauważysz jedynie rozszerzoną o adres IP zdalnego sąsiada komendę `neighbor [adres_IP_zdalnego_sąsiada] [interfejs_wychodzący] remote lisp-encap [numer_systemu_autonomicznego]`. Podobnie jak poprzednio należy komendą `network` rozgłosić odpowiednie sieci.

```
R3(config)#router eigrp OTP
R3(config-router)#address-family ipv4 autonomous-system 100
R3(config-router-af)#neighbor 172.30.1.1 g0/1 remote lisp-encap 100
R3(config-router-af)#network 2.2.2.2 0.0.0.0
R3(config-router-af)#network 172.30.5.0 0.0.0.255
```

Po przeprowadzeniu konfiguracji zauważysz na konsoli komunikaty informujące o nawiązaniu relacji sąsiedzkiej. Jeśli komunikaty się nie pojawią, należy sprawdzić, czy wszystkie czynności przeprowadziłeś poprawnie oraz czy na pewno trasy statyczne zostały wcześniej poprawnie skonfigurowane.

```
R3(config-router-af)#
*Aug 30 14:19:54.026: %DUAL-5-NBRCHANGE: EIGRP-IPv4 100: Neighbor 172.30.1.1
(GigabitEthernet0/1) is up: new adjacency
*Aug 30 14:19:54.919: %LINEPROTO-5-UPDOWN: Line protocol on Interface LISP100, changed state to up
R3(config-router-af)#
```

Teraz z routera R1 możesz wykonać `ping` na adres interfejsu `loopback` routera R3. Jak widzisz na poniższym listingu, komunikacja działa prawidłowo.

```
R1#ping 2.2.2.2
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 2.2.2.2, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5), round-trip min/avg/max = 7/8/14 ms
R1#
```

Sprawdźmy jeszcze, jak wygląda tablica routingu routera R1. Zauważ, że wpis dotyczący EIGRP OTP w zasadzie nie różni się zbytnio od standardowego EIGRP. Jest tutaj jednak informacja o interfejsie wyjściowym `LISP100`. W standardowym protokole routingu byłby tutaj adres interfejsu fizycznego.

```
R1#show ip route
<POMINIĘTO>
Gateway of last resort is not set
      1.0.0.0/32 is subnetted, 1 subnets
C        1.1.1.1 is directly connected, Loopback0
      2.0.0.0/32 is subnetted, 1 subnets
D        2.2.2.2 [90/93994331] via 172.30.5.2, 00:00:42, LISP100
      172.30.0.0/16 is variably subnetted, 4 subnets, 2 masks
C        172.30.1.0/24 is directly connected, GigabitEthernet0/0
L        172.30.1.1/32 is directly connected, GigabitEthernet0/0
S        172.30.5.2/32 [1/0] via 172.30.1.2
S        172.30.10.2/32 [1/0] via 172.30.1.2
R1#
```

Konfigurację routera R4 przeprowadź analogicznie, zmieniając wymagane dane dotyczące interfejsu oraz rozgłaszanych sieci.

```
R4(config)#router eigrp OTP
R4(config-router)#address-family ipv4 autonomous-system 100
R4(config-router-af)#neighbor 172.30.1.1 g0/0 remote lisp-encap 100
R4(config-router-af)#
*Aug 30 14:25:07.116: %DUAL-5-NBRCHANGE: EIGRP-IPv4 100: Neighbor 172.30.1.1
(GigabitEthernet0/0) is up: new adjacency
*Aug 30 14:25:07.974: %LINEPROTO-5-UPDOWN: Line protocol on Interface LISP100, changed state to up
R4(config-router-af)#network 3.3.3.3
R4(config-router-af)#network 172.30.10.0 0.0.0.255
```

Zauważ, że przeprowadzone ustawienia wprowadzają do konfiguracji routera również dodatkowy interfejs LISP100. Był on widoczny już w tablicy routingu, ale również wydając komendę `show ip int brief`, możesz sprawdzić jego ustawienia adresacji oraz status.

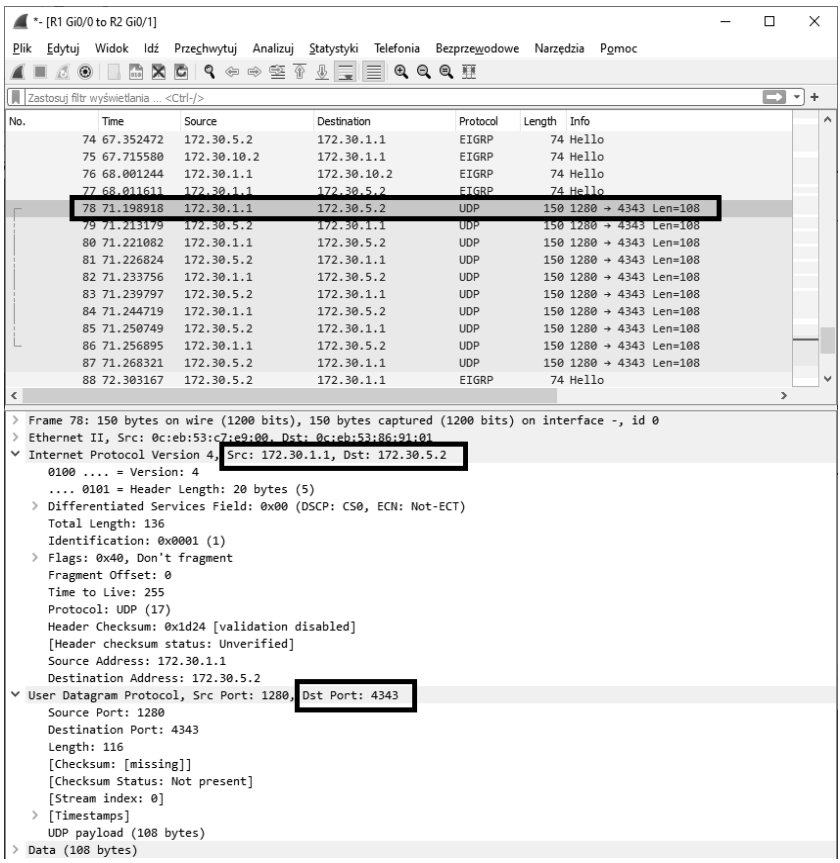
```
R1#
Interface                  IP-Address      OK? Method Status                Protocol
GigabitEthernet0/0         172.30.1.1      YES manual up                    up
GigabitEthernet0/1         unassigned      YES unset  administratively down down
GigabitEthernet0/2         unassigned      YES unset  administratively down down
GigabitEthernet0/3         unassigned      YES unset  administratively down down
LISP100                    1.1.1.1         YES unset  up                    up
Loopback0                  1.1.1.1         YES manual up                    up
R1#
```

Wydaj jeszcze na routerze R1 komendę dotyczącą sieci routera R3. Jest to komenda `show ip cef 2.2.2.2 internal`. W wyniku tej komendy możesz zauważyć, że router R1 używa właśnie interfejsu LISP100 do przesłania danych do adresu 2.2.2.2. Adres 172.30.5.2 jest to adres kolejnego skoku dla pakietów przeznaczonych do tej sieci.

```
R1#show ip cef 2.2.2.2 internal
2.2.2.2/32, epoch 0, RIB[I], refcnt 5, per-destination sharing
  sources: RIB
  feature space:
    IPRM: 0x00028000
  ifnums:
    LISP100(10): 172.30.5.2
  path list 0D922E34, 3 locks, per-destination, flags 0x49 [shble, rif, hwcn]
    path 105860B0, share 1/1, type attached nexthop, for IPv4
      nexthop 172.30.5.2 LISP100, IP midchain out of LISP100, addr 172.30.5.2 0F7D4C48
```

```
  output chain:
    IP midchain out of LISP100, addr 172.30.5.2 0F7D4C48
    IP adj out of GigabitEthernet0/0, addr 172.30.1.2 0F7D4D78
R1#
```

Zaobserwować można to dokładniej, kiedy podczas wydania komendy `ping` na adres `2.2.2.2` z routera R1 przechwycisz ruch na łączu pomiędzy R1 i R2. Zauważ, że na rysunku 12.18 znajdują się przechwycone pakiety. Jak widzisz, są enkapsulowane w pakiety UDP z docelowym numerem portu 4343.



Rysunek 12.18. *Przechwycone dane komunikacyjne pomiędzy routerem R1 i R2*

Automatyzacja sieci

W dzisiejszych czasach automatyzacja jest wszechobecna. Polega na wprowadzeniu do procesu rozwiązań, które sprawiają, że odbywa się on bez udziału człowieka. Jeśli jakaś czynność, np. w procesie produkcyjnym, jest powtarzalna, wówczas można ją zautomatyzować. W procesach produkcyjnych automatyzacja jest nieoceniona — dziś trudno sobie wyobrazić duże fabryki bez automatyzacji.

Automatyzacja pozwala także wyeliminować błędy i ograniczenia związane z czynnikiem ludzkim. Jeśli na przykład dziesięć osób przykręci śruby, każda osoba zrobi to z inną siłą i szybkością, bo nawet stan psychiczny i fizyczny człowieka ma wpływ na to, jak śruba będzie dokręcona.

Z maszynami jest zupełnie inaczej. Przede wszystkim mogą one pracować 24 godziny na dobę praktycznie każdego dnia przez okrągły rok (jeśli oczywiście wykluczymy ewentualne awarie). Maszyna będzie znacznie bardziej dokładna i jest w stanie powtórzyć dokładnie tak samo każdy swój ruch. Ponadto każdy ruch maszyny można monitorować, a uzyskane w ten sposób dane poddać analizie, która pozwoli zoptymalizować jej działanie. Niestety człowiek w wielu aspektach nie będzie tak wydajny jak maszyna.

Obecnie na świecie jest coraz więcej zautomatyzowanych fabryk, w których pracują tylko roboty, a prym w robotyzacji wiodą Chiny. To właśnie w chińskich fabrykach produkujących masowo między innymi części elektroniczne roboty przejmują całość procesów na taśmie. W jednej z takich fabryk w Chinach jeszcze w 2017 roku było zatrudnionych 650 osób, obecnie pracuje ich tam 50. Coraz więcej marketów ma samoobsługowe kasy, a są już nawet takie, w których nie ma kas, ponieważ urządzenia elektroniczne rejestrują, co wkładasz do koszyka. Przed wyjściem ze sklepu potwierdzasz tylko zakupy i płacisz przez telefon komórkowy. Ludzie nie są potrzebni, z wyjątkiem jednego czy dwóch operatorów, którzy nadzorują pracę systemu.

Sieci komputerowe również są coraz bardziej zautomatyzowane i konsola powoli odchodzi do lamusa. Oczywiście jeszcze trochę to potrwa, ale koniec narzędzia, które znasz, jest już coraz bliższy.

Coraz szersze zastosowanie ma sztuczna inteligencja, która sprawia, że urządzenia są w stanie same zmieniać swoje zachowanie i działanie w zależności od zmieniających się warunków zewnętrznych. Czy jest to inteligencja, czy może raczej programistyczna iluzja inteligencji, można by o tym dyskutować. Niemniej jednak urządzenia mogą już w zależności od sytuacji w sieci na przykład przekonfigurować router tak, aby przesyłał dane tym segmentem sieci, który jest mniej zajęty. Jest to automatyzacja sieci (ang. *network automation*), związana oczywiście z jej programowaniem.

Dzisiejszym światem rządzi nie tylko pieniądz, ale i obraz. Ogrom reklam, spotów, plansz, plakatów, banerów, których przekaz odbieramy świadomie bądź podświadomie, zewsząd atakuje nasze zmysły, zachęcając, abyśmy się zainteresowali danym produktem czy usługą i oczywiście dokonali zakupu. Obraz ma wzbudzić pragnienie posiadania na tyle silne, abyś w końcu uległ i kupił. Przestań na chwilę czytać i pomyśl, ile masz w domu rzeczy, które wydawały Ci się bardzo potrzebne w chwili zakupu, a teraz leżą od wielu tygodni nieużywane. Zapewne już wiesz, które to rzeczy, więc przejdźmy dalej.

W świecie cyfrowym przekazywane informacje muszą mieć określone formaty danych, aby były zrozumiałe dla użytkownika. Komunikacja w sieci to tylko 0 i 1, a mimo to widzisz przecież przyciągającą wzrok stronę WWW, a nie ciąg zer i jedynek. Dzieje się tak, ponieważ warstwa aplikacji jest w stanie poskładać bity i poprzez odpowiednie oprogramowanie wyświetlić je tak, aby były dla nas zrozumiałe i atrakcyjne. Najprostszym przykładem tego jest HTML (ang. *Hypertext Markup Language*). Dzięki niemu poprzez wykorzystanie znaczników widzisz strony WWW.

Formatów danych jest wiele. Obecnie najbardziej popularne to:

- JavaScript Object Notation (JSON),
- eXtensible Markup Language (XML),
- YAML Ain't Markup Language (YAML).

W zależności od tego, jakiego formatu danych będziesz używać, wynik końcowy może być inny, ale też sam proces przygotowania i zastosowane przez programistę metody będą inne. To tak jak programowanie w różnych językach programowania, które różnią się przecież składnią, obiektami itd.

Format danych to przede wszystkim coś, dzięki czemu różne aplikacje, nawet te uruchomione na urządzeniach, są w stanie się komunikować. Jeśli masz dwa urządzenia, to bardzo ważne jest, aby format danych był na nich ujednolicony. Wspomniany HTML jest tego przykładem: na serwerze znajduje się zawartość strony napisana w HTML, serwer wysyła ją do klienta, gdzie znajduje się przeglądarka, która jest w stanie wyświetlić te dane, odpowiednio je prezentując, tak jak jest to sformatowane w pliku ze stroną.

Urządzenia wspierające API mogą dzięki temu wymieniać się informacjami. API (ang. *Application Programming Interface* — interfejs programowania aplikacji) to mechanizm, który umożliwia właśnie wymianę informacji pomiędzy oprogramowaniem urządzenia A i urządzenia B, jest więc istotnym narzędziem komunikacji pomiędzy urządzeniami, na których to oprogramowanie jest zaimplementowane i działa. Za chwilę zrozumiesz, po co właściwie są formaty danych, API oraz inne mechanizmy.

Aby przekonać się, jak dane mogą zostać sformatowane, w pierwszej kolejności na przełączniku Nexus, który został zaimplementowany w programie GNS3, wydano komendę `show interface brief`. Są to dane, które na przykład przełącznik mógłby wysłać do innego urządzenia. Nie może ich jednak wysłać ot tak. Jak możesz zauważyć, jest to standard wyświetlania typowy dla systemu IOS, który już doskonale znasz.

```
switch# show interface brief
--------------------------------------------------------------------------------
Port   VRF          Status IP Address                              Speed    MTU
--------------------------------------------------------------------------------
mgmt0  --           down   --                                      --       1500
--------------------------------------------------------------------------------
Ethernet      VLAN    Type Mode   Status  Reason                 Speed     Port
Interface                                                                  Ch
--------------------------------------------------------------------------------
Eth2/1        --      eth  routed down    Administratively down      auto(D) --
Eth2/2        --      eth  routed down    Administratively down      auto(D) --
Eth2/3        --      eth  routed down    Administratively down      auto(D) --
<POMINIĘTO>
switch#
```

Trzeba te otrzymane dane jakoś zaprezentować, aby ewentualnie móc się nimi podzielić. Dlatego te same dane możesz uzyskać, wykorzystując wybrane argumenty dodatkowe.

Jeśli chodzi o format danych JSON, to jest on bardzo czytelny, co możesz zauważyć na poniższym listingu, który prezentuje również polecenie `show interface brief` na przełączniku Nexus, ale tym razem wyświetlone w formacie JSON. Służy do tego komenda `show interface brief | json`.

```
switch# show interface brief | json
{
  "TABLE_interface": {
    "ROW_interface": [
      {
        "interface": "mgmt0",
        "state": "down",
        "mtu": 1500
      },
      {
        "interface": "Ethernet2/1",
        "vlan": "--",
        "type": "eth",
        "portmode": "routed",
        "state": "down",
        "state_rsn_desc": "Administratively down",
        "speed": "auto",
        "ratemode": "D"
      },
      {
        "interface": "Ethernet2/2",
        "vlan": "--",
        "type": "eth",
        "portmode": "routed",
        "state": "down",
        "state_rsn_desc": "Administratively down",
        "speed": "auto",
        "ratemode": "D"
      },
```

Każdy interfejs w tym formacie jest ujęty w odrębne nawiasy. Zwiększa to czytelność całości.

Jeśli na przełączniku wydasz komendę `show interface brief | xml`, te same dane będziesz mógł wyświetlić w formacie XML. Spójrz na poniższy listing.

```
switch# show interface brief | xml
<?xml version="1.0" encoding="ISO-8859-1"?>
<nf:rpc-reply xmlns:nf="urn:ietf:params:xml:ns:netconf:base:1.0" xmlns="http://w
ww.cisco.com/nxos:1.0:if_manager">
 <nf:data>
  <show>
   <interface>
    <__XML__OPT_Cmd_show_interface_brief___readonly__>
     <__readonly__>
      <TABLE_interface>
       <ROW_interface>
        <interface>mgmt0</interface>
        <state>down</state>
        <mtu>1500</mtu>
       </ROW_interface>
       <ROW_interface>
        <interface>Ethernet2/1</interface>
```

```
        <vlan>--</vlan>
        <type>eth</type>
        <portmode>routed</portmode>
        <state>down</state>
        <state_rsn_desc>Administratively down</state_rsn_desc>
        <speed>auto</speed>
        <ratemode>D</ratemode>
       </ROW_interface>
       <ROW_interface>
<POMINIĘTO>
```

Format XML jest również czytelny nawet pomimo tego, że zawiera znacznie więcej znaczników niż JSON. Najważniejsze w tych formatach danych jest to, że jeśli taka konfiguracja w postaci XML zostanie przesłana do innego urządzenia, to inne urządzenie wspierające ten format zrozumie te dane i będzie mogło je odpowiednio zinterpretować. Będziesz mógł napisać program, który na przykład po wgraniu danych z pliku XML skonfiguruje urządzenie sieciowe automatycznie.

Nie będę tutaj omawiał konstrukcji poszczególnych formatów. Chodzi tylko o to, abyś zrozumiał, do czego są potrzebne.

Szablony

Automatyzacja sieci jest także pomocna podczas mozolnej konfiguracji urządzeń. Różnego rodzaju szablony sprawiają, że konfigurowanie wielu urządzeń staje się szybsze, ale przede wszystkim nie jest narażone na błędy. Jako administrator zapewne możesz „pochwalić się", ile razy w trakcie konfigurowania kolejnych urządzeń popełniłeś błędy, zwykle polegające na zamianie cyfr w adresach IP.

Szablony to gotowce, które działają w taki sposób, że administrator wpisuje do nich dane i na podstawie tych danych urządzenie jest na przykład konfigurowane. Oczywiście dane mogą również zostać do szablonu wprowadzone automatycznie przez inne oprogramowanie. Wprowadzane dane przeważnie przyjmują ustandaryzowany format, aby mogły zostać prawidłowo zinterpretowane przez dany schemat.

Bardzo popularnym językiem wykorzystywanym w tworzeniu szablonów jest Jinja, który świetnie współpracuje z językiem programowania Python. To właśnie Pythona możesz użyć do wypełniania szablonu danymi.

Jednym z najważniejszych narzędzi służących do automatyzacji sieci jest Ansible (inne to np. Puppet, Salt czy Chef). Ansible wykorzystuje do transportu informacji protokół SSH. Ponieważ został napisany w języku Python, świetnie z nim współpracuje, podobnie jak z oprogramowaniem Jinja.

Pytania sprawdzające

4. Jaki format danych prezentuje poniższy listing?

```
{
  "TABLE_interface": {
    "ROW_interface": [
      {
        "interface": "mgmt0",
        "state": "down",
        "mtu": 1500
      },
      {
        "interface": "Ethernet2/1",
        "vlan": "--",
        "type": "eth",
        "portmode": "routed",
        "state": "down",
        "state_rsn_desc": "Administratively down",
        "speed": "auto",
        "ratemode": "D"
      },
      {
        "interface": "Ethernet2/2",
        "vlan": "--",
        "type": "eth",
        "portmode": "routed",
        "state": "down",
        "state_rsn_desc": "Administratively down",
        "speed": "auto",
        "ratemode": "D"
      },
```

a) YAML,

b) XML,

c) Java,

d) JSON.

5. Który z interfejsów służy do komunikacji kontrolera z urządzeniami sieciowymi?

a) SBI,

b) VINT,

c) NBI,

d) SDN.

6. Jaka usługa chmurowa udostępnia użytkownikowi pełną infrastrukturę informatyczną?

a) PaaS,

b) IaaS,

c) SaaS,

d) żadna z powyższych.

Odpowiedzi

1b, 2c, 3c, 4d, 5a, 6b.

Skorowidz

3DES, Triple Data Encryption Standard, 560
802.1X, 540

A

Administrative Distance, dystans administracyjny, 156
adres
  ARP, 53
  MAC
    docelowy, destination MAC address, 37, 40
    wirtualny, 476
    źródłowy, source MAC address, 37
  rozgłoszeniowy sieci, 60
  VIP, 472
adresowanie grupowe, 383
AES, Advanced Encryption Standard, 533, 560
agregacja portów, link-aggregation, 133
AH, Authentication Header, 558
akcja, actions, 720
algorytm
  3DES, 560
  AES, 560
  DES, 560
  Dijkstry, 145
  DKE, 534
  DUAL, 144
  IDEA, 560
  MD5, 255
  Path Vector Algorithm, 145
  RC, 560
  RSA, 564
  Single Token Bucket Algorithm, 459
  SPF, 209
  STA, 101
  stanu łącza, 145
  TKIP, 533
  wektora odległości, 143
algorytmy haszujące, 562
alternate port, port alternatywny, 107, 120
amplituda, 507
analiza ramek, 76
Ansible, 724
antena
  dookólna, omnidirectional antenna, 512
  kierunkowa, directional antenna, 512
  MIMO, 512
API, Application Programming Interface, 642, 706
architektura Spine and Leaf, 34
AS, Autonomous System, 298
ASBR, Autonomous System Boundary Router, 259
AS-PATH, 366
ataki na sieci bezprzewodowe, 535
atrybut
  AS-Path, 301, 350
  atomic aggregate, 301
  local preference, 301
  Local Preference, 337, 350
  MED, 351
  Next-hop, 301
  Origin, 301
  Origin Code, 351
  Weight, 335, 348
atrybuty ścieżki, Path Attributes, 300
authenticator, 534
automatyzacja, 719
automatyzacja
  konfiguracji, 704, 724
  sieci, network automation, 640, 641
autosumaryzacja, 325
  w BGP, 325

B

backup port, port zapasowy, 120
bandwidth, przepustowość, 141, 169
baza danych stanu łącza, 209
baza LSDB, 213
BDR, Backup Designated Router, 212, 243
bezpieczeństwo, 21
  DHCP snooping, 26

bezpieczeństwo
Dynamic ARP Inspection, 26
funkcja passive-interface, 257
funkcjonalność CoPP, 554
LISP, 594
IPsec, 557
konfiguracja 802.1X, 542, 545
MD5, 481
listy ACL, 547
Port Security, 26
router dostępowy, 553
RSTP, 123
sieci bezprzewodowych, 532
VPN, 563
SSH, 55
STP, 129
TELNET, 55
tunelowanie, 584
VTP, 89
zapory ogniowe, firewalls, 579
Zone-Based Firewall, 581
BFD, Bidirectional Forwarding Detection, 287, 689
BGP, 296
aktywacja MP-BGP, 360, 361
atrybut Local Preference, 337
MED, 351
Origin Code, 351
Weight, waga, 335
atrybuty, 301
communities, 369
Community No Advertise, 370
Community No-Export, 371
eBGP Multihop, 314
filtrowanie informacji, 365
lista prefiksów, 359
oznaczanie, 368
rozszerzona lista dostępu, 355
znaki specjalne, 367
iBGP, 328
IPv6, 360
konfiguracja, 308
maszyna stanowa, 304
MP-BGP, 360
prefiksy IPv6, 361
procesy, 303
redystrybucja, 322
relacje sąsiedzkie, 302
route maps, mapy trasy, 339
routery, 300
rozgłaszanie sieci, 319
sumaryzacja, 325, 364
topologie, 306
uwierzytelnianie, 314
BGP, Border Gateway Protocol, 298
blok przełączania, switch block, 29
błąd enkapsulacji, encap_fail, 52
fragmentacji pakietów, 53
sumy kontrolnej, chksum_Err, 52
BPDU, Bridge Protocol Data Units, 102
BPDU guard, 123, 131
brak sąsiedztwa, no_adj, 52
właściwej trasy, no_route, 52
broadcast multiaccess, 243
storm, burza rozgłoszeniowa, 101
BSA, Basic Service Area, 504
BSR, 416
konfiguracja, 416
BSS, Basic Service Set, 504
BSSID, BSS Identifier, 505

C

CAC, Call Admission Control, 432
CAM, Content-Addressable Memory, 44
CAPWAP, 532
CBWFQ, 431
CCMP, 533
CDP, 98
CEF, Cisco Express Forwarding, 47–51
CFI, Canonical Format Indicator, 42
Chef, 724
chmura, 630
hybrydowa, hybrid cloud, 631
prywatna, private cloud, 631
publiczna, public cloud, 631
Cisco DNA, 715
EEM, 719
CLI, 706
Community No Advertise, 370
No-Export, 371
control plane, 614
controller-based WLAN, 63
CoPP, Control Plane Policing, 554
CRC, Cyclic Redundancy Check, 41
CSMA/CA, 43, 527
CSMA/CD, 43
C-VLAN, 94

D

database description, opis bazy danych, 209
debugowanie, 722
pakietów hello, 663
warunkowe, conditional debug, 662
default route, trasa domyślna, 153

delay, opóźnienie, 141, 169
dense mode, 393, 395
  konfiguracja, 396
DES, Data Encryption Standard, 560
designated port, port desygnowany, 106, 120
DFS, Dynamic Frequency Selection, 509
DHCP, 650
distance vector, 143, 144, 163
DKE, Dragonfly Key Exchange, 534
długość fali, wavelength, 506
DNA, 715
domena
  kolizyjna, collision domain, 23, 39, 46
  rozgłoszeniowa, broadcast domain, 22, 46, 59, 60
  VTP, 81
dostęp do urządzeń sieciowych, 53
DR, Designated Router, 212, 243, 422
drothery, 212, 246
DTP, Discovery Trunk Protocol, 65, 78
DUAL, Diffusing Update Algorithm, 144
DVMRP, Distance Vector Multicast Routing Protocol, 394
Dynamic NAT, 697
dystans administracyjny, 156, 209
  w EIGRP, 199

E

EAP, Extensible Authentication Protocol, 534, 541
EAP-FAST, 534
eBGP, External BGP, 300
  Multihop, 314
  połączenia redundantne, 318
ECMP, Equal-Cost Multipath Routing, 145
edge port, port brzegowy, 123
EEM, Embedded Event Manager, 720
EGP, Exterior Gateway Protocols, 298
EIGRP, Enhanced Interior Gateway Routing Protocol, 163
  algorytm DUAL, 144
  autosumaryzacja, 176
  dystans administracyjny, 199
  interfejs pasywny, 184
  konfiguracja, 164
    route map, 341
    routera stub, 194
  load balancing, 188
  mechanizmy zapobiegające powstawaniu pętli, 163
  metryka, 169
  modyfikacja czasów, 186
  nawiązywanie relacji sąsiedztwa, 180
  OTP, 636
    konfiguracja, 637
  prefix-list, 202
  rodzajów pakietów, 163
  rozgłaszanie tras, 185
  równoważenie obciążenia, 190
  tablica topologii, 195
  tablice, 166
  uwierzytelnianie tras, 187
  zużycie pasma, 187
EIGRPv4, 162
EIGRPv6, 200
enkapsulacja, 43
ERSPAN, 672
ESP, Encapsulating Security Payload, 558
ESS, Extended Service Set, 504
EtherChannel, 101, 133
Ethernet, 39
Ethernet II, 40

F

fala radiowa, 506
fale elektromagnetyczne, 507
FD, Feasible Distance, 168, 173
feasibility condition, warunek dopuszczalności, 174
Feasible Distance, 168
FHRP, First-Hop Redundancy Protocols, 27, 472
FIB, Forwarding Information Base, 49
filtrowanie
  bezstanowe, stateless filtering, 579
  obszarów, 281
  sieci w BGP, 355
  stanowe, statefull filtering, 579
  tras BGP, 354
  w OSPF, 280
firewall, 579
Floating Static Route, 156
format
  JSON, 718
  URI, 618
  XML, 644
forwarding plane, 614
FS, feasible successor, 174
funkcja
  BFD, 287
  passive-interface, 257
funkcjonalność
  autoRP, 404
  Cisco EEM, 719
  CoPP, 554
  ERSPAN, 672
  IGMP snooping, 423
  LISP, 594
  load balancing, 473
  passive-interface, 241
  Preemption, 477
  QoS, 430

funkcjonalność
Remote SPAN, 669
route map, 340
SPAN port, 666
track, 483
wirtualizacji, 634

G

GLBP, Gateway Load Balancing Protocol, 473, 495
konfiguracja, 495
granice zaufania, trust boundary, 451
GRE, Generic Routing Encapsulation, 585
szyfrowanie tunelu, 589
tworzenie tunelu, 584

H

hash value, 255
hasło
przechwytywanie, 255
hipernadzorca, hypervisor, 633
hold down timer, czas wstrzymania, 163
hop, 163
HSRP, Hot Standby Router Protocol, 473
funkcjonalność track, 483
konfiguracja, 473
czasów, 479
parametr Preempt, 478
priorytety, 477
status interfejsu, 475
uwierzytelnianie, 480
weryfikacja konfiguracji, 476
wirtualny adres MAC, 476

I

IaaS, Infrastructure as a Service, 631
iBGP, Internal BGP, 300, 328
IBSS, Independent Basic Service Set, 504
IDEA, International Data Encryption Algorithm, 560
identyfikator
BSSID, 505
routera, 223
SSID, 511
VLAN, 72
IGMP, Internet Group Management Protocol, 376
ramka komunikatu, 390
snooping, 423
IGMPv1, 378
IGMPv2, 386
IGMPv3, 390
IKE, Internet Key Exchange, 561
infrastruktura jako usługa, IaaS, 631
initial exchange, 142
instalacja Javy, 622
integralność danych, data integrity, 558
interfejs
NBI, 617
OSPF
statusy, 225
wyświetlanie właściwości, 230
pasywny, 184
SBI, 617
SVI, 49
trunk, 72
interfejsy
tryby pracy, 79
IP SLA, 684
IPsec, Internet Protocol Security, 557
IPv6 w BGP, 360
ISL, Inter-Switch Link, 71
IST, Internal Spanning Tree), 124

J

Java, 622
język
Jinja, 644
Python, 705
JSON, Java Script Object Notation, 718

K

klasy adresowe, 692
klasyfikator, policer, 464
klucz
tajny, 564
WEP, 534
kolejkowanie, 427, 430
CBWFQ, 431
konfiguracja LLQ, 446
LLQ, 432
WFQ, 431
kolektor NetFlow, 682
kolizje, 23, 39
komunikacja
grupowa, multicast, 40
HTTP, 711
jednostkowa, unicast, 40, 61
rozgłoszeniowa, broadcast, 40
urządzeń redundantnych, 472
z punktami dostępowymi, 521
komunikat
CDP, 77
DSCP, 455
Membership Query, 388
Membership report, 392
Membership Report, 381
Register-stop, 407
RP announcement, 411
komunikaty
LISP, 600
LISP UDP, 600
Notice, 653

konfiguracja
802.1X na przełączniku, 545
802.1X na routerze, 542
alternatywna OSPF, 219
atrybutów BGP, 335
BGP, 308
BSR, 416
dense mode, 396
EIGRP, 164
EIGRP OTP, 637
funkcjonalności ERSPAN, 672
GLBP, 495
HSRP, 473
interfejsu trunk, 72
LISP, 601
list ACL, 547
LLQ, 446
MD5
jeden klucz, 481
łańcuch kluczy, 482
MST, 124
NetFlow, 682
NTP, 657
OpenDaylight, 623
OSPF, 214
wieloobszarowego, 260
OSPFv3, 283
passive-interface, 241, 261
przełączników, 57
PVST, 117
Remote SPAN, 669
route map, 341
routera stub, 194
rozszerzonych ACL, 550
RSTP, 120
RSVP, 437
SNMPv2, 676
SNMPv3, 679
SPAN port, 666
sparse mode, 403, 409
szyfrowania, 286
track, 491
tunelu, 96
VLAN, 92
VRRP, 485
VRRPv3, 494
VTP, 84
VXLAN, 607
ZBF, 581
konsola Mininet, 626
kontener, container, 634
kontroler
LAP, 512
OpenDaylight, 622
SDN, 616
WLC, 512
kopie migawkowe, snapshot, 633
korzeń drzewa, root of the tree, 396
koszt trasy, 108, 237

L

LACP, Link Aggregation Control Protocol, 133
LAG, 133
LAP, Lightweight Access Point, 512
LEAP, Lightweight EAP, 534
liczba hopów, 141
Link State Database, 209
Link-State Algorithms, 145
LISP, Locator Identifier Separation Protocol, 594, 636
konfiguracja, 601
lista
ACL, 37, 47, 342, 343, 441, 547
konfiguracja, 547
rozszerzona, extended ACL, 354, 547, 550
w sieci VLAN, 552
warunek deny, 345
warunek out, 346
dystrybucyjna, 280
prefiksów, 202
QoS, 37
LLC, Logical Link Control, 39, 42
LLC1, connectionless, 42
LLC2, connection-oriented, 42
LLQ, Low Latency Queuing, 432
konfiguracja, 446
load balancing, 146, 188, 194
loading, 169
logi systemowe, 649
Loop Guard, 131
LSA, Link State Advertisements, 145
LSAck, Link State Acknowledgment, 210
LSDB, Link State Database, 213
LSP, Link State Packet, 209
LSR, Link State Request, 209
LSU, Link State Update, 209
LWAP, Lightweight Access Point Protocol, 512

Ł

łącze trunk, 26

M

MAC, Media Access Control, 42
MANO, 635
mapa polityk, class-map, 441, 447
maska odwrotna, wildcard mask, 215
maszyna
stanowa BGP, 304
wirtualna, 619, 632
ustawienie interfejsów, 622
max distance, maksymalny dystans, 163
MD5
konfiguracja, 481, 482

mechanizm
BPDU guard, 131
CSMA/CA, 43
CSMA/CD, 43
przełączania, switch engine, 48
root guard, 129
TCN, 112
metryka, 141
w EIGRP, 169
w OSPF, 233
zmiana wartości, 171
MIB, Management Information Base, 675
MIC, Message Integrity Check, 533
Microsoft Network Monitor, 525
model
dwuwarstwowy, two-tier, 25
hierarchiczny, 24
bez warstwy rdzenia, 31
dodatkowe rozwiązania, 30
warstwa dostępu, 25
warstwa dystrybucji, 27
warstwa rdzenia, 28
trójwarstwowy, three-tier, 25
modele danych, 717
MOSPF, Multicast OSPF, 394
MP-BGP, 360
MRP, Multicast Routing Protocol, 393
dense mode, 395
tryb dense mode, 393
tryb sparse mode, 394
MST, Multiple Spanning Tree, 124
konfiguracja, 124
multicast, 375
Multicast PIM Bootstrap, 415

N

najdłuższe dopasowanie, 155
NAT, Network Address Translation, 580, 691
dynamiczny, 697
interfejs wirtualny, 701
statyczny, 692
interfejs wirtualny, 699
z przeciążeniem, 697
interfejs wirtualny, 702
NBI, northbound interface, 617
neighbor table, tablica sąsiadów, 166
NetFlow, 682
konfiguracja, 682
NFVI, 635
nieobsługiwany protokół, unsupported, 52
nierozpoznany adres, unresolved, 52
non-designated port, port niedesygnowany, 106
NTP
konfiguracja, 657
wymiana danych, 661
NTP peer, 660
NTP, Network Time Protocol, 655

O

obszar
NSSA, 271, 278
stub, 275
stub area, 270
totally stub, 271, 278
totally stuby NSSA, 280
TSNSSA, 271
obszary w OSPF, 270
odległość, distance, 143
ogłoszenie VTP
advertisement request, 90
okno
Network Interface Configuration, 524
WiFi Scanning Options, 525
OpenDaylight, 622–625
okno logowania, 625
OpenFlow, 616
operacje SLA, SLA operations, 684
opóźnienie, delay, 430
propagacji, propagation delay, 428
przetwarzania, processing delay, 428
serializacji, serialization delay, 428
oprogramowanie jako usługa, SaaS, 630
OSPF, Open Shortest Path First, 207, 209
baza LSDB, 213
częstotliwość wysyłania pakietów, 240
filtrowanie
obszarów, 281
tras, 280
identyfikator routera, 223
konfiguracja
alternatywna, 219
protokołu, 214
koszt przesłania pakietów, 237
metryka, 233
obliczanie kosztu, 239
pakiety hello, 210
passive-interface, 241
polecenia weryfikujące, 267
przepustowość, 233
relacje sąsiedztwa, 225
równoważenie obciążenia, 221
sieci wielodostępowe, 243
sieć wieloobszarowa, 258

stany interfejsów, 225
trasy
domyślne, 242
statyczne, 274
typy
obszarów, 270
tras, 270
uwierzytelnianie, 254
wieloobszarowy
konfiguracja sieci, 260
trasy domyślne, 265
trasy statyczne, 266
wybór typu obszaru, 275
wymiana informacji, 226
OSPFv2, 209
OSPFv3, 282
konfiguracja, 283
szyfrowania, 286
uwierzytelniania, 286
OSS/BSS, 635
oznaczanie
ramek, 71
ruchu w QoS, 443

P

PaaS, Platform as a Service, 631
PAC, Protected Access Credential, 534
PAGP, Port Aggregation Protocol, 133
pakiet
Bootstrap, 416
Candidate-RP--Advertisement, 418
database description, 209
ESP, 559
hello, 163, 166, 180, 209, 397
Backup Designated Router, 212
Designated Router, 212
ID obszaru, 211
identyfikator routera, 210
intervals, 211
neighbors, 213
network mask, 212
router priority, 212
typ komunikatu, 210
HSRP, 481
ICMP, 61
Identity Protection, 576
ISAKMP, 575, 578
Join/Prune, 401, 408
LISP, 598, 599
LSA, 145, 213, 248
typy, 259
LSAck, 210
LSP, 209
LSR, 209
LSU, 209
NTP, 662
PIMv2, 404
query, 164
Register, 407
RP mapping, 412
RSVP, 440
unicast, 60
update, 164, 182, 183
pamięć
CAM, 44
NVRAM, 88
TCAM, 37, 47
parametr Preempt, 478
passive-interface, 241, 257, 261
PAT, Port Address Translation, 697
interfejs wirtualny, 702
Path Vector Algorithm, 145
PCP, Priority Code Point, 41
pętla routingu, 144
pętle, 395
piaskownica Cisco, 716
PIM, Protocol Independent Multicast, 376, 393
Auto-RP listener, 409
Sparse-Dense mode, 409
platforma jako usługa, PaaS, 631
płaszczyzna danych, data plane, 49
PMF, Protected Management Frames, 534
podpis elektroniczny, 563
podwarstwa LLC, 39
PoE, Power over Ethernet, 26
pola
pakietu LISP, 598
ramki wi-fi, 514, 530
pole
CFI, 42, 72
CRC, 41
danych, data, 41
LLC, 42
PCP, 41
preambuły, preambule, 40
Priority, 72
SFD, 40
TAG, 71
TCI, 41
TPID, 41
TTL, 99
typu, type, 40
VID, 42
policing, 463
połączenia trunk, 68, 78
port
alternatywny, alternate port, 107, 120
brzegowy, edge port, 123
chroniony przełącznika, 66
desygnowany, designated port, 106
desygnowany, designated port, 120
główny, root port, 106
główny, root port, 120
hosta, 66
niedesygnowany, non-designated port, 106
zapasowy, backup port, 120
PortFast, 112

porty
agregacja, 133
RSTP, 120
w STP, 106
Postman, 629, 706
eksportowanie, 707
importowanie, 708
konfigurowanie
zmiennych, 711
tworzenie interfejsu, 712
URL dostępny, 710
uwierzytelnienie, 709
wysyłanie poleceń, 714
potwierdzenie LSAck, 250
poufność, confidentiality, 558
powiadomienie o zmianie konfiguracji, 652
PPDIOO, 32
priorytet, Priority, 104
priorytetyzacja ramek, *Patrz* QoS
procesor routingu, routing procesor, 48
program
Ansible, 724
Cisco DNA, 715
GNS3, 666
Kiwi Syslog, 677
Microsoft Network Monitor, 524, 526
Mininet, 625
NetFlow, 682
Puppet, 724
SyslogViewer, 651
Wireshark, 226, 255, 401
tryb monitorowania, 523
protokoły
bezklasowe, 209
distance vector, 143
do tunelowania, 98
haszujące, 558
link state, 145
redundancji, 473
routingu
algorytmy, 143
dynamicznego, 142
symetrycznego szyfrowania, 558
wewnętrznej bramy, 142
zewnętrznej bramy, 142
protokół
AutoRP, 415
Auto-RP, 409
BFD, 689
BGP, 296
BSR, 416
CCMP, 533
CDP, 62
Diffiego-Hellmana, 558
DTP, 65, 78
DVMRP, 394
EAP, 541
eBGP, 314
EIGRP, 341
EIGRPv4, 162
EIGRPv6, 200
FHRP, 27
GLBP, 473, 495
HSRP, 473
iBGP, 328
IGMP, 376
IGMPv1, 378
IGMPv2, 386
IGMPv3, 390
IKE, 561
IPsec, 557
ISL, 71
LACP, 133
LISP, 636
LWAP, 512
MOSPF, 394
MST, 124
Multicast PIM Bootstrap, 415
NTP, 654
OpenFlow, 616
OSPF, 207
OSPFv2, 209
OSPFv3, 282
PAGP, 133
PIM, 393
PVST, 114
RSTP, 119
RSVP, 437
SNMP, 674
STP, 26, 99
TCP, 300
typu link state, 145, 209
UDLD, 131
UDP, 649
VRRP, 473, 485
VTP, 82
przechwycenie
hasła, 255
pakietu
Bootstrap, 417
ESP, 559
hello, 257, 497
HSRP, 481
IGMPv2, 387
IGMPv3, 391
ISAKMP, 575
LISP, 598
NTP, 662
RP mapping, 412
RSVP, 440
STP, 670
TCP, 332
komunikatu
DSCP, 455
Membership Query, 388
Membership Report, 381, 392
RP announcement, 411
test ping, 672
transmisji IGMPv1, 381
przeciążenie, congestion, 430
przełącznik
konfiguracja, 57
802.1X, 545
o budowie modułowej, modular switches, 37
o stałej konfiguracji, fixed configuration, 37
przekazywanie ramek, 37
tryb
client, 88
server, 88
w ruchu grupowym, 423
warstwy, 614
warstwy drugiej, L2, 23, 37
działanie, 44

warstwy trzeciej OSI, L3, 37
działanie, 47
wielowarstwowy, 47, 49
przełączniki
Catalyst, 37
Leaf, 34
Nexus, 37
Spine, 34
PSK, Pre-Shared Key, 564
punkt dostępowy, 511
punkty udostępniania usług, Service Access Point, 42
Puppet, 724
PVRST, Per-VLAN Rapid Spanning Tree, 114
PVST, Per-VLAN Spanning Tree, 103, 114
konfiguracja, 117
zmiana mostu głównego, 117
Python, 705

Q

QinQ, 94
protokoły do tunelowania, 98
QoS, Quality of Service, 21, 47, 426
klasyfikatory ruchu sieciowego, 439, 456
model
best-effort, 433
kategorie narzędzi, 434
zintegrowanych usług, 433
znakowanie pakietów, 434
zróżnicowanych usług, 433
oznaczanie ruchu, 443
w przełącznikach, 450
QUERY, 199
Quiet Mode, 572

R

ramka
beacon, 529
BPDU, 102
DTP, 81
ethernetowa Ethernet II, 40
ethernetowa 802.1Q, 41
komunikatu IGMP, 390
STP, 670
typu Prune, 401
uwierzytelniająca, authentication frame, 531
VRRP, 491
VTP, 86
wi-fi, 513, 514, 529
raport członkowski, membership report, 378
RC, Rives Cipher, 560
RD, Reported Distance, 173
redundancja
przełączników, 99
w L3, 471
redystrybucja
tras statycznych, 274
w BGP, 322
relacja sąsiedztwa, 225
statusy, 249
reliability, 169
reported distance, 192
REST API, 706
RESTful API, 618
revision number, 86, 88, 90
RFP, Reverse Path Forwarding, 395
RIB, Routing Information Base, 49
root
bridge, most główny, 102, 104, 117
guard, 129
port, port główny, 106, 120
route
maps, mapy trasy, 339–349
poisoning, zatruwanie trasy, 163
router
ABR
sumaryzacja tras, 263
ASBR, 259
BDR, 243, 246, 252
BGP, 300
komunikaty, 300
desygnowany, DR, 422
distance vector, 144
DR, 243, 246, 252
ID, 210
konfiguracja 802.1X, 542
na patyku, router on a stick, 48, 94
stub, router szczątkowy, 194
routing
dynamiczny, 140
statyczny, 146
w środowisku wirtualnym, 635
rozgłaszanie
sieci w BGP, 319
tras, 185
tras domyślnych, 242
równoważenie obciążenia, load balancing, 222
w OSPF, 221
RSPAN, Remote SPAN, 669
RSTP, Rapid Spanning Tree Protocol, 119
knfiguracja, 120
porty, 120
zabezpieczenie BPDU guard, 123
RSVP, 437
RTP, Reliable Transport Protocol, 163, 180

S

SaaS, Software as a Service, 630
SBI, southbound interface, 617
SD-ACCESS, 690
komponenty, 690

SDN, Software-Defined Network, 614
  zastosowanie, 619
SD-WAN, 688
  Schemat działania, 689
  vAnalytics, 690
  vBond, 690
  vEdge, 689
  vManage, 688
  vSmart, 689
serwer
  syslog, 650
  TELNET, 54
  VMPS, 78
  VPN, 564
  VTP, 88
SFD, start frame delimiter, 40
sieci bezprzewodowe, 503
  ataki, 535
  bezpieczeństwo, 532
  CSMA/CA, 527
  częstotliwości, 509
  działanie, 505
  etapy połączenia, 527
  format ramki, 513
  kanały, 509
  mechanizm
    DFS, 509
    TPC, 509
  projektowanie, 536
  przechwytywanie ramek, 522
  punkt dostępowy, 511
  standardy, 510
  tryb
    ad hoc, 504
    infrastrukturalny, 504, 527
  urządzenia bezprzewodowe, 511
  uwierzytelnianie, 533, 534
  WLAN, 504
  WPAN, 503
  WWAN, 504
  wymiana informacji, 528
sieć
  domowa, 22
  kampusowa, 18
  point-to-point, 243, 252
  SDN, 614
  SD-WAN, 688
  Token Ring, 42
  VLAN dynamiczna, 78
  VLAN, 24, 42, 58, 59
  VPN, 563
  WAN, 687
  z redundancją, 471
skrypt main.py, 420
SNMP, 675
  read-only community string, 675
  read-write community string, 675
SNMPv2
  konfiguracja, 676
SNMPv3
  konfiguracja, 679
SPAN port, 666
sparse mode, 394, 401
  konfiguracja, 403, 409
sparse-dense, 414
SPB, Shortest Path Bridging, 34
SPF, Shortest Path First, 209
split horizon, podzielony horyzont, 163
SPT, Shortest Path Tree, 396
SSH, 55
SSID, Service Set Identifier, 505
STA, Spanning-Tree Algorithm, 101
standalone WLAN, 63
standard
  IEEE 802.1D, 101
  IEEE 802.1Q, 71
  IEEE 802.3ec, 39
stany portów, 111
Static NAT, 692
status
  2-WAY/DROTHER, 250
  FULL/BDR, 250
  FULL/DR, 250
  FULL/DROTHER, 250, 251
status interfejsu OSPF, 225
STP, Spanning Tree Protocol, 26, 100
  działanie protokołu, 102
  koszty tras, 108
  Loop Guard, 131
  mechanizm
    BPDU guard, 131
    root guard, 129
  PortFast, 112
  rodzaje portów, 106
  stany portów, 111
  wybór mostu głównego, 104
stub, 199
sukcesor, 173
sumaryzacja, 259
  adresów, 177, 179
  tras, 263
  tras statycznych, 149
  w BGP, 325, 364
SVI, ang. Switched Virtual Interface, 49
S-VLAN, 94
syslog, 648
system autonomiczny, 143
szablony, 644
szyfrowanie
  asymetryczne, 561
  w OSPFv3, 286

Ś

śledzenie obiektów, object tracking, 472

T

tablica
  adresów MAC, 38
  ARP, 476

CAM, 47
FIB, 49, 615
MAC, 37, 44
MAC, MAC address table, 37
przełączania, forwarding table, 49
przełączania CAM, 37
RIB, 49, 615
routingu, 51, 142
EIGRP, 166, 168
minimalizowanie, 151
trasa domyślna, 155
sąsiadów, 52
PIM, 422
TCAM, 47
topologii, topology table, 167
EIGRP, 195
TAG, 71
TAS, Trunk Administrative Status, 80
TAT, Trunk Admnistrative Trunk-Encapsulation, 80
TCAM, Ternary Content Addressable Memory, 47
TCI, Tag Control Information, 41, 72
TCN, Topology Change Notification, 112
TCP, 300
technologia
agregacji portów, 133
CEF, 48
EtherChannel, 101
EtherChannel, 133
TELNET, 53
TKIP, Temporal Key Integrity Protocol, 533
TNS, Trunk Negotiated Status, 80
TNT, Trunk Negotiated Trunk-Encapsulation, 80
token, 717
topologia
dual homed, 307
fizyczna, physical topology, 18
logiczna, logical topology, 20
single homed, 306
single multihomed, 308
topologie BGP, 306
TOS, Trunk Operational Status, 80
TOT, Trunk Operational Trunk-Encapsulation, 80
TPC, Transmit Power Control, 509
TPID, Tag Protocol IDentifier, 41, 72
traceroute, 663
traffic
policer, 456, 457
sharper, 460
translacja adresów, 691
dynamiczna, 697
statyczna, 692
z przeciążeniem, 697
transmisja grupowa, multicast, 375
adresowanie, 377
protokoły, 376, 378
trasy
domyślne, 153, 242, 265
domyślne IPv6, 161
redundantne, 191
statyczne, 274
statyczne IPv6, 159
w OSPF, 270
zapasowe, 174
triggered updates, aktualizacje wyzwalane, 163
TRILL, Transparent Interconnection of Lots of Links, 34
tryby pracy interfejsów, 79
TTL, time-to-live, 50, 99
tunel
CAPWAP, 531
GRE, 584
IPsec, 561
VPN, 565
VTI, 591
VXLAN, 604, 606
tunelowanie 802.1Q, 94
konfiguracja, 96
QinQ, 98
sprawdzanie działania, 98
tworzenie maszyny wirtualnej, 619

U

UDLD, Unidirectional Link Detection, 131
UDP, 649
Unequal-Cost Load Balancing, 146
URI, Uniform Resource Identifier, 618
URL, Uniform Resource Locator, 618
URN, Uniform Resource Name, 618
usługa NAT, 691
usługi chmury, 630
uwierzytelnianie, authentication, 558, 563
HSRP, 480
MD5, 257
w BGP, 314
w EIGRP, 187
w OSPF, 254
w OSPFv3, 286
w VRRP, 490
Quiet Mode, 572

V

vAnalytics, 690
vBond, 690
vEdge, 689
VID, VLAN ID, 42
VIP, Virtual IP, 472
VLAN, 23, 24
identyfikator, 72
konfiguracja sieci, 63
sieci wirtualne LAN, 58
usuwanie konfiguracji, 92
VLAN Access-List, 552

VLAN operatorski, 94
vManage, 688
VMPS, VLAN Membership Policy Server, 78
VMR, Value Mask Result, 48
VNF, Virtual Network Function, 634
VOIP, 62
VPN, 563
  implementacja IKE, 565
  konfiguracja site-to-site, 566
  remote access, 564
  tunel site-to-site, 563, 573
  wymiana kluczy DH, 577
VRF, Virtual Routing and Forwarding, 289
VRRP
  konfiguracja, 485
    czasów, 490
    track, 491
  przeglądanie rozgłoszeń, 489
  uwierzytelnianie, 490
VRRP, Virtual Router Redundancy Protocol, 473
VRRPv3
  konfiguracja, 494
vSmart, 689
VTI, Virtual Tunnel Interface, 591
  konfiguracja, 591
VTP, VLAN Trunking Protocol, 82
  aktualizacje, 90
  konfiguracja, 84
  ogłoszenia, 85
  Pruning, 92
  zabezpieczenie hasłem, 89
VXLAN, Virtual Extensible LAN, 604
  konfiguracja, 607

W

warstwa
  dostępu, access layer, 25
  dystrybucji, 27
  LLC, 42
  łącza danych, data link layer, 42
  MAC, 42
  oparta na routingu, 26
  oparta na STP, 26
  rdzenia, core, 28
warunek
  deny, 345
  match, 344
  out, 346
  permit, 345
wąskie gardło, bottleneck, 430
wektor, vector, 143
  odległości, distance vector, 143
WEP, dynamic WEP keys, 534
WEP, Wired Equivalent Privacy, 532
WFQ, Weighted Fair Queuing, 431
wiadomości przycinające, prune messages, 394
wi-fi, *Patrz* sieci bezprzewodowe
wirtualizacja, 613, 630
wirtualna maszyna
  ustawienia sieci, 620
wirtualne karty sieciowe, vNIC, 634
WLAN, Wireless LAN, 504
WLC, WLAN Controller, 512
WPA, WiFi Protected Access, 533
WPA2, 533
WPAN, Wireless Personal Area Network, 503
WWAN, Wireless Wide Area Networks, 504
wyzwalacz, trigger, 720

Z

zalewanie, 44
  pakietami LSA, 259
zapytanie o członkostwo, membership query, 379
ZBF, Zone-Based Firewall, 581
  konfiguracja, 581
zbieżność, 287
znacznik, tag, 94
  802.1Q, 41
znak
  pytajnik, 351
  gwiazdka, 396
znaki specjalne BGP, 367

Ż

żądanie ARP, 53

ZOSTAŃ ADMINISTRATOREM SIECI!

- **Poznaj technologie sieciowe**
- **Naucz się je wykorzystywać**
- **Zdobądź certyfikat CCNP**

Sieci komputerowe oplatają świat dosłownie i w przenośni. Stanowią krwiobieg współczesnych systemów informatycznych, zapewniając dostęp do internetu nawet w najdalszych zakątkach globu i dostarczając miliardom ludzi najrozmaitszych informacji i usług cyfrowych. Oczywiście sieci nie utrzymują się same. Odpowiadają za to wykwalifikowani specjaliści — administratorzy sieci — którzy od lat mogą przebierać w atrakcyjnych ofertach zatrudnienia i bez wątpienia także w przyszłości będą przez pracodawców równie intensywnie poszukiwani.

Jeśli interesujesz się sieciami i chcesz się rozwijać w tym zakresie, sięgnij po książkę *CCNP 350-401 ENCOR. Zaawansowane administrowanie siecią Cisco*. Pomoże Ci ona przygotować się do egzaminu umożliwiającego uzyskanie certyfikatu Cisco Certified Network Professional, lecz również bliżej poznać technologie przydatne w pracy administratora. Krok po kroku będziesz przyswajać odpowiednie wiadomości, ugruntujesz swoje kompetencje i potwierdzisz kwalifikacje bez konieczności sięgania po inne materiały. Jeśli szukasz dobrego kompendium wiedzy o sieciach komputerowych, nie mogłeś lepiej trafić!

- **Projektowanie sieci kampusowych**
- **Konfiguracja przełączników**
- **Konfiguracja routingu**
- **Protokół BGP**
- **Transmisja grupowa**
- **Zapewnianie jakości usług**
- **Sieci bezprzewodowe**
- **Bezpieczeństwo sieci**
- **Wirtualizacja i monitorowanie sieci**
- **Programowanie sieci i automatyzacja**

PROJEKTUJ, ADMINISTRUJ I ROZWIJAJ SIECI CISCO!

ADAM JÓZEFIOK ukończył studia doktoranckie na Politechnice Śląskiej w Gliwicach, na Wydziale Automatyki, Elektroniki i Informatyki. Specjalizuje się w tematyce sieci komputerowych (przełączanie, routing, bezpieczeństwo i projektowanie). Jest autorem publikacji polskich i zagranicznych z tej dziedziny. Brał udział w konferencjach naukowych (krajowych i międzynarodowych) dotyczących sieci komputerowych. Jest pracownikiem naukowym Politechniki Śląskiej w Gliwicach. Na co dzień administruje również bezpieczeństwem urządzeń sieciowych, konfiguruje między innymi urządzenia sieciowe (Cisco, HP), utrzymuje sieci WAN. Przez siedem lat kierował komórką realizacji wsparcia usług IT. Posiada wieloletnie doświadczenie w zakresie pracy jako administrator sieci i projektant sieci komputerowych. Przez lata projektował serwerownie i tworzył projekty okablowania. Przygotowywał procedury i dokumentacje projektowe. Na koncie ma wiele zrealizowanych projektów zakupu sprzętu IT wraz z prowadzeniem procedur przetargowych, wdrożeniem, wykonaniem dokumentacji i testami. Posiada certyfikaty: CCNA Security, CCNP Routing and Switching, Cisco CCDP, CCNAV, a także certyfikat instruktorski Cisco CCAI, jak również certyfikaty ITIL i PRINCE2. Jego pasją jest pisanie książek, praca ze studentami i szeroko rozumiana dydaktyka.

Helion

helion.pl

HELION SA
ul. Kościuszki 1c
44-100 Gliwice
tel.: 32 230 98 63
helion@helion.pl

ISBN 978-83-283-5237-7

9 788328 352377

INFORMATYKA W NAJLEPSZYM WYDANIU

Cena: 199,00 zł