Adresy i sieci

W każdej sieci każde miejsce, do którego inne komputery wysyłają informacje, musi mieć niepowtarzalny identyfikator. Identyfikator taki nazywany jest zwykle *adresem.* W niektórych technologiach sieciowych adres wskazuje konkretną maszynę, podczas gdy w innych, takich jak IP, adres wskazuje punkt przyłączenia do sieci, który jest powszechnie nazywany *interfejsem.* W rezultacie pojedyncza maszyna pracująca w sieci, która jest wyposażona w kilka interfejsów, może mieć kilka adresów IP - po jednym dla każdego z tych interfejsów. Interfejsy to zwykle fizycznie rozróżnialne przyłącza (tzn. gniazda, do których dołączany jest kabel sieciowy), ale mogą być nimi również logiczne przyłącza, które mają jedno wspólne przyłącze fizyczne.
Możesz się spotkać również z innym rozwiązaniem określanym jako *multipleksacja interfejsu,* które stosuje się w przyłączach do sieci ATM. Logiczny podział hostów w sieci ATM na kilka grup pozwala na traktowanie każdej z nich jako oddzielnej sieci logicznej, mimo że wszystkie hosty przyłączone są do jednej sieci fizycznej. Urządzenie przyłączone do tego typu sieci fizycznej może jednocześnie należeć do kilku sieci logicznych dzięki nawiązaniu kilku logicznych połączeń, z których każde ma własny adres IP.
Maszyny, które mają kilka adresów, określa się jako *multi-homed.* Wszystkie rutery są z definicji maszynami multi-homed, ponieważ zajmują się przesyłaniem pakietów pomiędzy kilkoma sieciami. Jednakże nie wszystkie maszyny określane mianem multi-homed są ruterami, np.: jedna maszyna może mieć kilka przyłączeń do sieci i pełnić funkcję serwera plików współdzielonego przez kilka różnych sieci, bez rutowania informacji pomiędzy tymi sieciami.

Struktura adresu IP

Adresy IP mają długość 32 bitów. Rozpatruje się je jako sekwencję czterech bajtów lub inaczej, czterech *oktetów* (bajtów 8-bitowych). Aby zapisać adres IP, należy dokonać konwersji każdego z oktetów do postaci zapisu dziesiętnego i oddzielić cztery powstałe w ten sposób liczby dziesiętne kropkami. A zatem 32-bitowy adres IP:

```
10101100   00011101   00100000   01000010
```

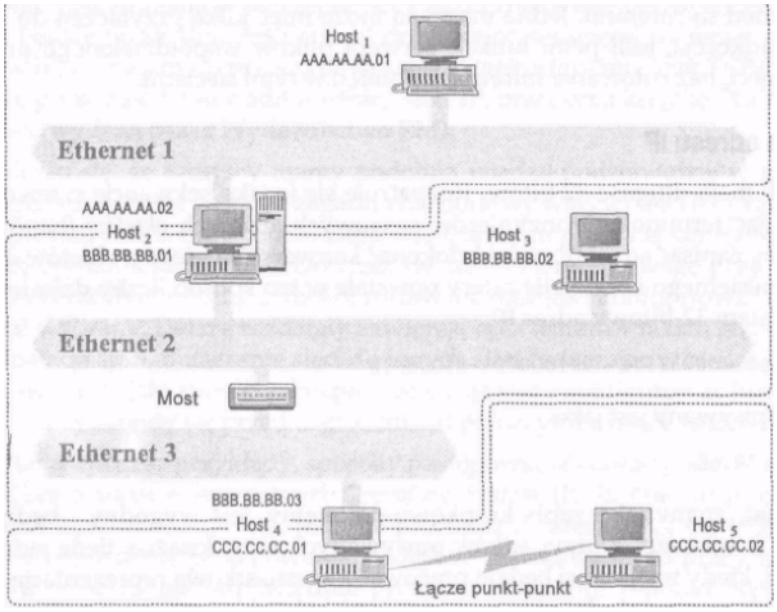
zwykle zapisywany jest jako:

```
172.29.32.66
```

Taki format, znany jako zapis kropkowo-dziesiętny, jest wygodny i będziemy go stosowali w większości przypadków opisywanych w tej książce. Będą jednak takie przypadki, kiedy wygodniej będzie pracować z szesnastkową reprezentacją adresów 32-bitowych, ponieważ ułatwi to wykonanie niektórych operacji lub pozwoli je lepiej zrozumieć. W zapisie szesnastkowym adres IP, przedstawiony wyżej, będzie reprezentowany w następujący sposób:

```
0xAC1D2042
```

Mimo że adres IP jest pojedynczą liczbą 32-bitową to zbiór adresów IP nie jest płaski. Zamiast tego adresy zbudowane są w oparciu o dwupoziomową hierarchię: sieci i hostów wchodzących w skład tych sieci.
Każda z tych dwóch przestrzeni adresowych identyfikowana jest przez określoną część adresu IP, w wyniku czego każdy adres IP możemy podzielić na numer sieci i numer hosta.
W protokole IP numer sieci reprezentuje zbiór maszyn, które zdolne są do bezpośredniej komunikacji w warstwie drugiej sieciowego modelu odniesienia ISO-OSI. Warstwa ta to warstwa łącza danych, która odzwierciedla działanie takich rozwiązań jak Ethernet, Token Ring, FDDI *(Fiber Distńbuted Data Interconnect),* a także łącza typu punkt-punkt. Każda z tych technologii sieciowych traktowana jest przez IP jako jedna sieć, niezależnie od tego, czy jest to rzeczywiście jeden kabel sieciowy, czy też składa się ona z kilku segmentów połączonych ze sobą przez wzmacniaki, mosty lub przełączniki.
Nie powinieneś być zaskoczony, dowiadując się, że numer hosta określa konkretną maszynę, która należy do danej sieci. Na rysunku 1-1 pokazano przykład opisanego wyżej sposobu adresowania.



Rysunek 1-1: Ethernet 2 i 3 to jedna sieć.

Na rysunku 1-1 sieci Ethernet 2 i 3 tworzą jedną sieć IP, mimo że rozdziela je most, co wynika z faktu, że urządzenie takie jak most jest niewidoczne z poziomu protokołów warstwy sieci, jaką jest IP. Host2, Host3 i Host4 mają adresy IP, w których znajduje się taki sam numer sieci przydzielony dla tego podwójnego układu sieci Ethernet połączonych mostem.
Łącze szeregowe pomiędzy Host4 a HostS tworzy drugą sieć IP i hosty te będą miały adresy składające się z numeru sieci tworzonej przez to połączenie szeregowe.
Sieć Ethernet l jest trzecią siecią IP, a Host1 i Host2 będą miały adresy IP zawierające jej adres.
Hosty o nazwach Host2 i Host4 mają po dwa adresy IP: są to hosty typu multi-homed i mogą pełnić funkcje ruterów.

Dwupoziomowa struktura adresów IP będzie ważna w dalszej części książki, gdy będzie mowa o rutowaniu. Na razie wystarczy, jeśli wskażemy, która część adresu IP to numer sieci, a która - numer hosta.
Umieszczenie numeru sieci w adresie IP powoduje, że adres hosta zależy od sieci, w której ten host się znajduje. Oznacza to, że jeśli host zostanie przeniesiony do innej sieci, to konieczna jest zmiana jego adresu.
W przeciwieństwie do innych technologii sieciowych, takich jak IPX Novella, gdzie adres ustalany jest w oparciu o adres sprzętowy karty sieciowej lub AppleTalk firmy Apple Computer, gdzie adres wybierany jest automatycznie, adres IP jest nadawany i wyznaczany ręcznie. Mimo że dostępne są protokoły takie jak *Boot Strap Protocd (BOOTP)* i *Dynamie Host Configuration Protocd (DHCP),* które wspomagają wyznaczanie adresu IP dla maszyny w sieci, to serwery obsługujące te protokoły wymagają ręcznej konfiguracji i nie wszystkie urządzenia w sieci są w stanie wykorzystać zalety tych usług. Konieczność zmiany numeru hosta po zmianie jego miejsca pracy oznacza zawsze dodatkowe zadania dla personelu odpowiedzialnego za utrzymanie sieci.

Numery sieci i maski

Jak napisałem wcześniej, wszystkie adresy IP składają się z numeru sieci i numeru hosta w tej sieci. Jednakże granica pomiędzy numerem sieci i numerem hosta może przebiegać różnie. Aby oprogramowanie ruterów i hostów mogło w łatwy sposób określić, w którym miejscu znajduje się wspomniany podział adresu, każdy z nich ma dołączoną informację w postaci *maski sieci.* Maska ta to liczba 32-bitowa, podobnie jak w adresie IP, w której wszystkie bity określające sieciową część adresu są równe l, a bity określające część adresu będącą numerem hosta ustawione są na 0.
Na przykład:
11111111 11111111 00000000 00000000
oznacza, że pierwszych 16 bitów adresu IP, z którym skojarzona jest ta maska, reprezentuje numer sieci, a ostatnich 16 bitów reprezentuje numer hosta w tej sieci. Komputer może w prosty sposób wyliczyć numer sieci z adresu IP stosując bitowe działanie AND pomiędzy adresem IP i jego maską.
Początkowo maski sieci mogły zawierać nie przylegające bity 1. Praktyka ta została jednak zmieniona, częściowo z powodu trudności, jakie sprawiała, a częściowo po to, by uprościć wymianę informacji o rutowaniu. Obecnie wszystkie maski muszą mieć wszystkie bity l przylegające. Oznacza to, że następująca maska:
11111111 11111111 00000011 00000000

jest niedozwolona, ponieważ ostatnie dwa bity l nie przylegają do innych. Ograniczenie to nie spowodowało większych kłopotów, ponieważ do chwili jego wprowadzenia używano niewielu masek, w których bity l nie były przylegające.

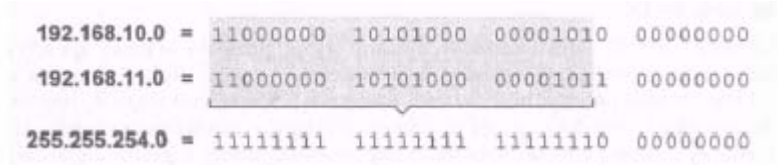
Podobnie jak adres IP, maska sieciowa jest tradycyjnie reprezentowana przy użyciu zapisu kropkowo-dziesiętnego lub szesnastkowego. A zatem maska może być zapisana jako 255.255.254.0 lub jako OxFFFFFE00 - ten sposób jest częściej używany podczas tworzenia oprogramowania.

Ponieważ jednak maski zawsze są związane z adresem IP i bez niego nie mają większego znaczenia, coraz popularniejszy staje się nowy format zapisu maski. W związku z tym, że wymagany jest obecnie zapis w postaci nieprzerwanego ciągu bitów l, możliwe jest posługiwanie się pojęciem maski np.: 23-bitowej. Takie określenie jednoznacznie mówi, że mamy na myśli maskę złożoną z 23 bitów l, po których następuje 9 bitów 0 lub w zapisie szesnastkowym OxFFFFFE00. Pozwala to na uproszczenie stwierdzenia, że „sieć rozpoczyna się adresem 192.168.2.0 z maską 255.255.254.0" i zapisanie go w postaci 192.168.2.0/23. Ten nowy zapis adresów i masek nazywany jest zapisem *adres/maska.*

Format wyświetlania
192.168.2.0/23
192.168.2.0 255.255.254.0
192.168.2.0 0xFFFFFE00

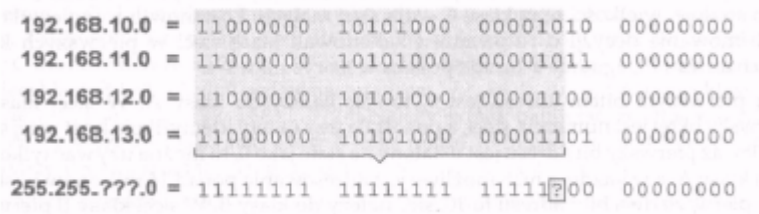
Tabela 1-1. Formaty wyświetlania informacji o maskach

Podstawowy zapis adres/maska pozwala na łatwe opisywanie adresów IP z sieci o dowolnym rozmiarze, poczynając od prostego łącza punkt-punkt, w którym pracują dwa hosty w sieci, kończąc na sieciach, w których znajduje się wiele milionów hostów. Rozważmy na przykład dwa adresy pokazane na rysunku 1-2. Ponieważ mają one jednakowy 23-bitowy przedrostek i są kolejnymi numerami, to możliwe jest zapisanie przestrzeni adresowej obu wymienionych adresów przy użyciu wspomnianego zapisu, w wyniku czego powstaje adres w postaci 192.168.10.0/23.



Rysunek 1-2. Dwa adresy ze wspólnym 23-bitowym przedrostkiem.

Nie wszystkie kombinacje adresów i masek sieci mogą być poprawnie zapisane przy użyciu takiego zapisu. Na rysunku 1-3 pokazano cztery adresy, które nie mogą być reprezentowane przez jeden zapis typu adres/maska. Dzieje się tak dlatego, że adresy, mimo swej ciągłości, nie mają jednakowego 22-bitowego przedrostka. Dlatego nie jest możliwe podanie maski o długości 22 bitów, która objęłaby wszystkie te adresy. Jeśli będziesz chciał zapisać te adresy podając 192.168.10.0/22, to zapis ten obejmie tylko dwa z podanych czterech adresów, a dwa pozostałe zostaną pominięte.



Rysunek 1-3. Cztery adresy bez wspólnego 22-bitowego przedrostka.

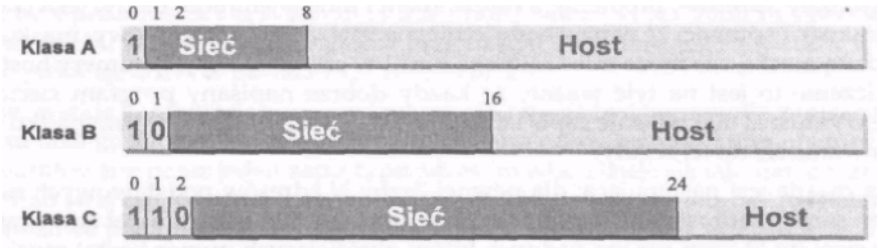
Zamiast takiego zapisu należy użyć dwóch oddzielnych specyfikacji: 192.168.10.0/23 i 192. 168.12.0/23, co oznacza dwa oddzielne zapisy w tablicy rutowania. o czym powiemy w dalszej części tego rozdziału.
Czy zapis 192.168.10.0/22 określa jakąś poprawną przestrzeń adresową? Tak i nie. Jeśli użyjesz maski z tym adresem, okaże się, że powstała w ten sposób przestrzeń adresowa jest taka sama jak dla adresu 192.168.8.0/22. Czy w tego rodzaju zapisie ważny jest adres podstawowy?
Tak! Nawet doświadczeni administratorzy błędnie sądzą, że opisana w ten sposób przestrzeń adresowa to numery od 192.168.10.0 do 192.168.13.255, choć komputer na podstawie zapisu 192 .168.1 0.0/22 wyznaczy przestrzeń adresową od 192.168.8.0 do 192.168.11.255. Są to oczywiście dwie zupełnie inne przestrzenie adresów.
Takie błędne zapisy mogą powodować podwójne przydziały adresów, problemy z rutowaniem i inne tajemnicze błędy.

Jeśli chcesz tego uniknąć i sprawić, że zapisy będą jednoznaczne, adres podstawowy, maskowany podaną maską, nie może mieć żadnego bitu l w części opisującej numery hostów. Ograniczenie to jest na tyle ważne, że każdy dobrze napisany program sieciowy będzie wymuszał taki właśnie zapis i informował o błędzie adresu w przypadku niezastosowania się do tej reguły.
Ogólna zasada jest następująca: dla pewnej liczby N adresów podstawowych mających ten sam przedrostek N musi być podstawą potęgi 2, a ostatni oktet zawierający numer sieci (w którym nie ma żadnych bitów określających numer hosta) musi być bez reszty podzielny przez N.

Klasy adresów IP

Podstawowy sposób zapisu adresów, opisany wyżej, pozwala w łatwy sposób rozróżnić rozmiar części będącej adresem sieci oraz części określającej liczbę hostów w tej sieci. Łatwo można policzyć hosty w sieci, następnie liczbę tę zaokrąglić do najbliższej wartości potęgi liczby dwa i na tej podstawie wystąpić o numer sieci i maskę dla tej sieci. Należy jeszcze pamiętać o dodaniu odpowiedniej liczby adresów zapasowych, które pozwolą na rozbudowę sieci w przyszłości.
Nie zawsze jednak przydzielanie adresów sieci odbywało się w taki sposób. W początkowym okresie rozwoju sieci IP maski miały ustalone wielkości, przez co po dodaniu ich do numerów sieci powstawały klasy sieci. Choć zastąpiono je bardziej elastyczną architekturą opisaną wyżej, to w literaturze i w języku potocznym często występują odwołania do nich. Niektóre protokoły rutowania, takie jak RIP, nadal posługują się tym pojęciem, dlatego zajmijmy się podstawowymi klasami sieci oraz ich ewolucją w kierunku używanej obecnie nowoczesnej architektury klas sieci.

Twórcy IP nie przewidywali, że protokół ten będzie musiał obsługiwać sieć o wielkości dzisiejszego Internetu. Zakładali, że będzie istniała potrzeba obsługi tylko kilku dużych sieci (działających w dużych firmach komputerowych i głównych uniwersytetach), średniej liczby sieci o średniej wielkości oraz wielu małych sieci. Dlatego też stworzyli trzy klasy sieci: klasę A przeznaczoną dla największych sieci, klasę B - dla sieci średniej wielkości oraz klasę C - dla sieci małych. Postanowili również ułatwić podejmowanie decyzji o rutowaniu i zakodowali klasę sieci w pierwszych kilku bitach adresu IP, zgodnie z zasadą pokazaną na rysunku 1-4.



Rysunek 1 -4: Klasa adresu jest zakodowana w pierwszych kilku bitach

Jeśli pierwszym bitem adresu jest 0, to sieć należy do klasy A. W sieci klasy A pierwszy oktet jest numerem sieci, a pozostałe trzy oktety identyfikują host w tej sieci. Ponieważ pierwszy bit adresu jest ustalony na stale jako 0, to można używać tylko 127 sieci klasy A, a w każdej z nich możliwe jest adresowania ponad 16 milionów hostów.
Jeśli pierwsze dwa bity adresu to 10, sieć należy do klasy B. W sieci klasy B pierwsze dwa oktety oznaczają numer sieci, a kolejne dwa - numer hosta w sieci. Pozwala to na utworzenie 16 384 sieci klasy B (zwróć uwagę, że podobnie jak w poprzedniej klasie, pierwsze dwa bity są stałe), a w każdej z nich może być 65 000 hostów.
Wreszcie jeśli pierwsze trzy bity to 110, sieć należy do klasy C. W sieci klasy C pierwsze trzy oktety są numerem sieci, a ostami oktet określa numer hosta w sieci. Pozwala to na utworzenie około 2 milionów sieci, z których każda może składać się z 256 hostów.

Zwróć uwagę, jak łatwo jest na podstawie pierwszych kilku bitów określić klasę sieci, a następnie znaleźć część adresu opisującą numer sieci i część z numerem hosta. Taka prostota była konieczna, ponieważ komputery w tamtych czasach miały znacznie mniejsze moce przetwarzania niż obecnie.

Zgodnie z oryginalną definicją adresy, w których pierwsze trzy bity to 111, należą do klasy D i zostały przeznaczone do wykorzystania w przyszłości. Od tego czasu definicja sieci tej klasy zmieniła się i klasa D definiowana jest obecnie jako adresy, w których pierwsze cztery bity to 1110. Adresy te nie oznaczają pojedynczego urządzenia, lecz zestaw urządzeń, które wchodzą w skład grupy IP, określanej jako *multicast.*
Adresy rozpoczynające się od 1111 nazywane są obecnie adresami klasy E i są zarezerwowane do wykorzystania w przyszłości. Prawdopodobnie jeśli dla kolejnej klasy adresów zostanie przydzielony jakiś sposób ich wykorzystania, to

definicja klas zostanie zmodyfikowana tak, że klasa E będzie się zaczynała od 11110, a nowa zdefiniowana klasa F (jako rezerwa na przyszłość) wyróżniana będzie początkowymi bitami w postaci 11111.

Jak się więc mają opisane wyżej klasy sieci do swych najnowszych odpowiedników?
Zwróć uwagę, że sieć klasy A ma 8-bitową maskę sieci. Oznacza to, że taka sieć o numerze 10.0.0.0 może być opisana jako 10.0.0.0/8 przy użyciu zapisu bezklasowego. Także *naturalna* maska sieci dla sieci klasy B ma długość 16 bitów, a dla sieci klasy C długość ta wynosi 24 bity.
W wyniku tak ustalonych długości masek oznaczenie sieci klasy B 172.16.0.0 będzie następujące: 172.16.0.0/16, a dla sieci klasy C o adresie 192.168.1.0 - 192.168.1.0/24.
Należy jednak pamiętać, że choć wszystkie sieci znane wcześniej jako sieci klasy B mają maski 16-bitowe, to nie jest prawdą, iż wszystkie sieci mające maski o długości 16 bitów są sieciami klasy B. Rozważmy przykład sieci 10.0.0.0/16. Wykorzystuje ona maskę 16-bitową, ale nadal pozostaje siecią klasy A (a raczej częścią takiej sieci), ponieważ jej binarna reprezentacja nadal zaczyna się od bitu 0. Na podobnej zasadzie skonstruowana jest sieć opisana przez 192.168.0.0/16, która nie jest siecią klasy B, lecz zbiorem 256 sieci klasy C. Różnice te mają duże znaczenie, gdy masz do czynienia z hostami i protokołami, które są świadome istnienia klas sieci. W takich przypadkach poprawne konfigurowanie maski jest sprawą niezmiernie istotną dla pracy systemu. W przypadku stosowania adresacji bezklasowej maska 16-bitowa to po prostu maska 16-bitowa.

Podsieci i super sieci (nadsieci)

W miarę jak twórcy protokołów IP nabierali doświadczenia w pracy z siecią, odkryli, że ustanowione początkowo klasy sieci pozwalały na przydzielanie sieci o wielkościach, które nie pasowały do potrzeb pojawiających się technologii LAN. Na przykład nie ma potrzeby przydzielać sieci klasy B, dającej możliwość adresowania ponad 65 000 hostów, sieci Ethernet, w której będzie pracowało maksymalnie l 200 urządzeń. Opracowano rozwiązanie nazywane podziałem na *podsieć,* w którym po raz pierwszy rzeczywiście wykorzystane zostały maski sieciowe.

W podsieciach IP bity należące do adresu IP hosta wykorzystywane są w charakterze bitów rozszerzających numer sieci. Na przykład w sieci klasy A 10.0.0.0 numer sieci opisany jest przez 8 pierwszych bitów, a pozostałe 24 bity tworzą numer hosta. Twórcy sieci IP zdali sobie sprawę, że możliwy jest podział tej sieci na podsieci dzięki wykorzystaniu kolejnych 8 bitów adresu, które z adresu hosta zostaną przypisane do adresu sieci, jak pokazano na rysunku 1-5.
Takie rozwiązanie pozwala stworzyć 256 podsieci, a w każdej z nich zaadresować 65 000 hostów. Możliwe jest również wykorzystanie 16 bitów z numeru hosta dla określenia adresów podsieci, co zwiększa liczbę podsieci do 65 000, a liczbę hostów w każdej z nich do 256.

Maski sieciowe nie muszą przebiegać zgodnie z kolejnymi granicami wyznaczonymi przez 8-bitowe porcje adresu IP. W wielu miejscach używa się takiego rozwiązania, ponieważ sposób podziału adresu na część sieciową i numer hosta jest łatwy do zapamiętania. Jeśli sieci klasy A 10.0.0.0 nie będziemy dzielili na podsieci, podział pomiędzy adresem sieci i adresem hosta przebiega w miejscu pierwszej kropki w zapisanym dziesiętnie adresie. Jeśli użyjemy 8-bitowej podsieci (tzn. 16-bitowej maski sieci), to granica podziału pomiędzy podsiecią a adresem hosta będzie przebiegała w miejscu występowania drugiej kropki. Jeśli z kolei użyjemy podsieci o wielkości 16 bitów (24-bitowej maski sieci), to linia podziału przebiegała będzie w miejscu trzeciej kropki.

Sieć	Podsieć	Host
10		27.9.4
10	27	9.4
10	27.9	4

Rysunek 1-5: Różne interpretacje adresu 10.27.9.4

Choć dla komputerów takie ułatwienia nie mają żadnego znaczenia, to dla ludzi są one bardzo wygodne i pozwalają w bardziej naturalny sposób dzielić adres na poszczególne części. Na przykład, jeśli w naszej przykładowej sieci 10.0.0.0 zdecydujemy się użyć maski o długości 10 bitów, to otrzymamy 1024 podsieci, a w każdej z nich po 4 miliony hostów. W takim przypadku granica podziału pomiędzy numerem podsieci a numerem hosta przebiega wewnątrz trzeciego oktetu i nie jest wyraźnie widoczna w zapisie kropkowo-dziesiętnym. Zastanów się nad adresami 10.1.190.0 oraz 10.1.191.1. Czy należą one do tej samej podsieci? Tak, lecz adres

10.1.192.1 już nie będzie do niej należał. Nawet szesnastkowy zapis adresu nie pokazuje wyraźnie tego rozdziału. Tylko zapis binarny pozwala na wyraźne rozróżnienie podsieci.

Maska podsieci ma zawsze przynajmniej tyle bitów 1, ile jest ich w naturalnej masce dla danej klasy sieci. Oznacza to, że podsieć jest zawsze mniejsza od sieci, bez względu na to, z jakiej klasy ta sieć pochodzi.
Kilka lat temu, gdy zaczęły się problemy związane z wyczerpywaniem się przestrzeni adresowej, zwrócono uwagę na fakt, że nie ma technicznego uzasadnienia dla tak sztywnego traktowania masek. Dlaczego nie przydzielać adresów sieci z maskami większymi od naturalnej maski dla sieci klasy C i nie stworzyć bloków kilku sieci C traktowanych jako jedna sieć lub *supersieć?* Właściwie, dlaczego ograniczać takie podejście do sieci klasy C? Dlaczego nie połączyć kolejnych sieci klasy B w jedną super sieć?
Takie rozwiązania są podstawą *bezklasowego rutowania pomiędzy domenami (Classless Interdomain Ruting - CIDR),* które tworzy stosowaną obecnie w sieci architekturę bezklasową.
Dzięki zastosowaniu maski sieciowej do wyznaczania zarówno podsieci, jak i supersieci, powstała nowa grupa *bezklasowych* protokołów rutowania, pozwalająca na rozszerzenie funkcji rutowania, które wcześniej możliwe było tylko pomiędzy sieciami z klas. Protokoły rutowania pracujące z klasami i protokoły bezklasowe nie mogą być ze sobą mieszane, ponieważ te drugie wymagają znajomości maski adresu, podczas gdy protokół klasowy sam określa maskę dla klasy sieci na podstawie pierwszych bitów adresu. Możliwe jest jednak kontrolowane połączenie obu typów protokołów na obrzeżach domeny rutowania. Rozwiązanie takie powinno być jednak stosowane w ostateczności i z pełną świadomością jego konsekwencji.

Adresy broadcast i multicast

Zdarzają się sytuacje, w których host pracujący w sieci IP musi komunikować się ze wszystkimi innymi hostami pracującymi w tej sieci. Ponieważ nie ma łatwego sposobu na stwierdzenie, jakie inne adresy w sieci są przypisane do hostów, a nawet trudno jest stwierdzić, które hosty w danym momencie są uruchomione, to host może wysłać kopię komunikatu na każdy adres w danej sieci po kolei. Jest to marnotrawstwo pasma sieci i mocy pracujących w niej komputerów. Aby poradzić sobie z tym problemem, IP definiuje adres 255.255.255.255 jako adres *broadcast* w sieci lokalnej. Każdy host pracujący w sieci IP odbiera komunikaty przychodzące na jego własny adres IP oraz na adres typu *broadcast.*

Broadcast w sieci lokalnej działa dobrze, jeśli host chce tylko przesłać komunikat do innych hostów połączonych bezpośrednio do tej samej sieci. Zdarzają, się jednak sytuacje, kiedy host chce wysłać pakiet do wszystkich hostów z danej sieci IP, które jednak nie są bezpośrednio połączone z jego siecią fizyczną. IP definiuje taki pakiet jako *skierowany broadcast.* Jego adres zawiera numer sieci, do której jest on kierowany, oraz wszystkie bity numeru hosta ustawione na 1. A zatem broadcast skierowany do sieci 10.0.0.0/8 będzie miał adres 10.255.255.255, a w przypadku sieci 172.29.0.0/16 będzie to adres 172.29.255.255.
W związku z potencjalnym zagrożeniem ze strony nieuczciwych użytkowników sieci lub ignorantów wiele ruterów może być skonfigurowanych tak, aby odrzucały skierowane pakiety broadcast, nie przepuszczając ich do wnętrza sieci, którą chronią.
Niektóre wersje starszego oprogramowania stosowały bity 0 zamiast l dla oznaczania adresów *broadcast.* Pomimo że systemy takie zanikają możesz się na nie natknąć, zwłaszcza jeśli w Twojej sieci pracują starsze systemy. Najnowsze oprogramowanie powinno akceptować oba sposoby adresowania pakietów broadcast i mieć możliwość konfigurowania sposobu adresowania przez bity 1 lub 0 przy wysyłanych przez siebie pakietach broadcast. Domyślnym ustawieniem adresu broadcast w nowych systemach jest 1.

Podobnie jak adres broadcast, adres multicast jest pojedynczym adresem reprezentującym grupę urządzeń w sieci. W przeciwieństwie do adresu broadcast, maszyny korzystające z adresu multicast muszą wcześniej wyrazić życzenie otrzymania pakietów kierowanych na ten adres. Komunikat wysyłany na adres broadcast jest odbierany przez wszystkie maszyny obsługujące protokół IP, niezależnie od tego, czy są one zainteresowane jego zawartością czy też nie.
Niektóre protokoły rutowania wykorzystują adresy multicast jako adres przeznaczenia dla wysyłanych okresowo informacji o rutowaniu. Pozwala to na łatwe ignorowanie takich komunikatów przez maszyny, które nie są zainteresowane uaktualnianiem informacji o rutowaniu. Z kolei broadcast musi być odebrany i przeanalizowany przez wszystkie maszyny, włączając w to hosty, które nie obsługują protokołu IP. Dopiero po odebraniu takiego pakietu maszyna może stwierdzić, czy jest zainteresowana jego zawartością. Wynika to z faktu, że obsługa pakietów broadcast realizowana jest na poziomie sprzętowym. Powoduje to, że pakiet tego typu wysyłany jest do wszystkich kart sieciowych niezależnie od tego, czy obsługuje je protokół IP, czy też inny protokół sieciowy, nie rozumiejący komunikatów

broadcast. Hosty pracujące z innym protokołem powinny odrzucić pakiety broadcast, ale takie działanie wymaga od hosta przetworzenia pakietu w celu potwierdzenia, że nie jest on nim zainteresowany.

Inne adresy specjalne

Należy jeszcze wspomnieć o dwóch specjalnych adresach IP. Pierwszym z nich jest adres *loopback,* 127.0.0.1. Adres ten zdefiniowany jest jako adres programowego interfejsu pętli zwrotnej działającego na danej maszynie. Adres ten nie jest przypisany do żadnego interfejsu sprzętowego i nie łączy się z siecią. Jest używany głównie w celu testowania oprogramowania IP na maszynie, która nie jest przyłączona do sieci, i bez względu na to, czy interfejs sieciowy lub jego sterowniki działają poprawnie.
Może on być również używany na maszynie lokalnej jako adres interfejsu, który jest zawsze aktywny i osiągalny przez oprogramowanie, niezależnie od aktualnego stanu interfejsów sprzętowych. Adres ten może być na przykład używany do adresowania odwołań oprogramowania klienta z serwerem uruchomionym na tej samej maszynie, bez konieczności używania zewnętrznego adresu l P hosta.
Specyfikacja protokołu IP wymaga aby adres ten, jak i cala sieć 127.0.0.0/8, nigdy nie był przypisywany do zewnętrznego interfejsu maszyny. Jeśli tak się zdarzy, adresy te będą odrzucane przez każdy host lub ruter, który będzie otrzymywał w taki sposób zaadresowane pakiety.
Zwróć uwagę, iż adres ten narusza zasadę, że adres IP jednoznacznie identyfikuje host, ponieważ wszystkie hosty pracujące w sieci IP wykorzystują ten sam adres dla obsługi interfejsu loopback.
Drugim specjalnym adresem IP jest 0.0.0.0. Oprócz wykorzystania go w starszym oprogramowaniu jako adresu broadcast w sieci lokalnej, niektóre protokoły rutowania traktują go jako adres przechwytywania lub *domyślną* trasę.

Adresy nadające się do użytku przy danej masce sieci

Do tej pory przyjmowaliśmy, że w każdej sieci z maską 24-bitową można umieścić do 256 hostów. Nie jest to do końca prawda.
Przypomnij sobie, że adres zawierający bity 1, w części określającej numer hosta, to adres broadcast.
Przypomnij sobie również, że w niektórych starszych implementacjach dla określenia adresu broadcast stosowane są bity 0.
W związku z tym adresy zawierające bity l i bity 0 w części określającej numer hosta nie mogą być stosowane do adresowania hosta w sieci. Daje to rzeczywistą liczbę dostępnych adresów hostów w takiej sieci, która wynosi 254.
Takie same restrykcje dotyczą wszystkich sieci i podsieci, niezależnie od długości maski.
Na przykład maska o długości 31 bitów w zapisie szesnastkowym 0xFFFFFFFE powinna dać możliwość wydzielenia podsieci, w której będą pracowały dwa hosty, idealnej dla konfiguracji łącza punkt-punkt. Ponieważ jednak nie możemy nadawać hostom numerów złożonych z samych bitów l ani samych bitów 0, to sieć utworzona taką maską jest bezużyteczna. Poprawną maską dla sieci, w której będą dostępne dwa adresy hostów, jest maska 30-bitowa – 0xFFFFFFFC. Pierwszy host w sieci będzie miał numer l, a drugi 2. Numer 0 nie jest dostępny dla hostów, a numer 3 będzie adresem broadcast.
Wyżej opisana niejednoznaczność występuje także w przypadku podsieci, dla których numer podsieci składa się z samych bitów 0 lub 1. Niektóre wersje oprogramowania sieciowego nie potrafią poprawnie obsługiwać tego typu podsieci. Inne wersje wymagają wyraźnego skonfigurowania funkcji programu, tak by te dwie sieci były obsługiwane poprawnie.

W tabeli 1-2 pokazano liczbę podsieci i hostów dla wszystkich masek podsieci w trzech blokach sieci o różnej wielkości. Na przykład jeśli wykorzystywany przez Ciebie blok sieci ma długość 16 bitów, to możesz użyć 25-bitowej maski podsieci w celu uzyskania 510 podsieci i 126 hostów w każdej z nich. Jeśli jednak długość bloku sieci wynosi 20 bitów, to taka sama 25-bitowa maska pozwoli na zaadresowanie 30 podsieci i 126 hostów w każdej z nich. Zwróć uwagę na to, że niektóre maski nie tworzą użytecznej liczby podsieci. Takie przypadki oznaczono za pomocą kreski poziomej. Podobne numery sieci można łatwo podzielić na bloki sieci o innej długości. Gdy będziesz się zastanawiał nad wyborem maski dla Twoich podsieci, pamiętaj o przykładach z poniższej tabeli.

Tabela 1-2. Liczba podsieci i hostów w zależności od długości maski i sieci

Liczba bitów	*Maska podsieci*	*Liczba podsieci w bloku sieci*			*Efektywna liczba hostów*
		16 bitów	*20 bitów*	*24 bity*	
16	255.255.0.0	1	-	-	65534
17	255.255.128.0	-	-	-	32766
18	255.255.192.0	2	-	-	16382
19	255.255.224.0	6	-	-	8190
20	255.255.240.0	14	1	-	4094
21	255.255.248.0	30	-	-	2046
22	255.255.252.0	62	2	-	1022
23	255.255.254.0	126	6	-	510
24	255.255.255.0	254	14	1	254
25	255.255.255.128	510	30	-	126
26	255.255.255.192	1022	62	2	62
27	255.255.255.224	2046	126	6	30
28	255.255.255.240	4094	254	14	14
29	255.255.255.248	8190	510	30	6
30	255.255.255.252	16382	1022	62	2
31	255.255.255.254	32766	2046	126	-
32	255.255.255.255	65534	4094	254	-

Algorytm rutowania IP

W sieci IP każde urządzenie podejmuje samodzielnie decyzje o rutowaniu. Wykorzystywany przy podejmowaniu tych decyzji algorytm jest taki sam, niezależnie od tego, czy jest to host, czy też ruter. Komputer wysyłający informacje nie musi definiować całej drogi prowadzącej przez sieć do punktu przeznaczenia. Musi jedynie wskazać kolejne urządzenie lub *przeskok (next-hop),* wchodzący w skład pełnej trasy. Następnie pakiet wysyłany jest do wskazanego urządzenia, które jest odpowiedzialne za wskazanie kierunku następnego przeskoku prowadzącego do punktu przeznaczenia. Proces ten jest powtarzany dotąd, aż pakiet będzie ostatecznie dostarczony do urządzenia, do którego był adresowany. Informacje o kolejnych przeskokach w kierunku adresu przeznaczenia przechowywane są w *tablicy rutowania.* Każdy wiersz w tej tablicy opisuje jedną sieć IP, podsieć lub hosta oraz adres kolejnego przeskoku, który tam prowadzi.

Klasyczne rutowanie IP

Mimo że większość ruterów potrafi rutować pakiety w bezklasowych sieciach IP, niektóre rutery nadal używają algorytmu rutowania powiązanego z klasą sieci, w której znajduje się adres przeznaczenia. Ten klasowy algorytm rutowania jest następujący:

```
Dla docelowego adresu IP:
jeśli (dysponujemy bezpośrednią trasą do hosta)
        odczytaj adres następnego skoku ze znalezionego wpisu,
        wyślij pakiet pod znaleziony adres następnego skoku.
jeśli (nie dysponujemy bezpośrednią trasą do hosta)
        jeśli (posiadamy interfejs należący do tej sieci)
                określamy maskę podsieci na podstawie informacji ze swojego interfejsu
        jeśli (nie posiadamy interfejsu należącego do tej sieci)
                określamy maskę podsieci na podstawie klasy adresu
        nakładamy uzyskaną maskę na adres aby otrzymać adres podsieci
```

```
jeśli (mamy interfejs w tej podsieci)
        wysyłamy pakiet do adresata.
jeśli (nie mamy interfejsu w tej podsieci)
        przeszukujemy tablicę routingu w poszukiwaniu wpisu dotyczącego tej podsieci
                jeśli (znajdziemy wpis)
                        wysyłamy pakiet pod znaleziony adres następnego skoku
                jeśli (nie znajdziemy wpisu)
                        szukamy trasy domyślnej w tablicy routingu
                        jeśli (mamy trasę domyślną)
                                wysyłamy pakiet pod adres następnego skoku trasy domyślnej
                        jeśli (nie mamy trasy domyślnej)
                                odrzucamy pakiet z komunikatem „destination unreachable”
```

Algorytm najpierw dokonuje sprawdzenia trasy prowadzącej bezpośrednio do hosta. Trasa bezpośrednia to umieszczony w tablicy rutowania zapis, który dokładnie opisuje trasę do adresu IP, gdzie kierowany jest pakiet. Taki zapis może być używany dla wskazania urządzenia pracującego po drugiej stronie szeregowego łącza punkt-punkt.
Jeśli trasa bezpośrednia nie zostanie znaleziona w tablicy rutowania, algorytm próbuje określić maskę podsieci dla sieci przeznaczenia. W przypadku sieci odległych (takich, do których wysyłający pakiety komputer nie jest bezpośrednio dołączony) w tablicy rutowania nie ma informacji o używanej masce podsieci, używana jest więc naturalna maska z klasy sieci. Jeśli mamy do czynienia z połączeniem bezpośrednim do sieci, maska określana jest na podstawie konfiguracji interfejsu sieciowego hosta. Interfejs ten może, lecz nie musi, być dołączony do podsieci, w której znajduje się adres przeznaczenia, ale algorytm zakłada, że maska sieci jest taka sama. W rezultacie rutowanie klasowe nie będzie poprawnie działało w sieci, w której stosowane są różne maski podsieci w różnych obszarach, chyba że sieć taka będzie bardzo starannie skonfigurowana przez administratora, tak by uniknąć niejednoznaczności.
Kiedy algorytm określi maskę podsieci dla sieci, do której wysyłane są pakiety, adres przeznaczenia maskowany jest tą maską w celu uzyskania numeru podsieci, który zostanie użyty jako klucz dla przeszukania tablicy rutowania. Jeśli algorytm stwierdzi, że host jest dołączony bezpośrednio do tej sieci, to pakiet wysyłany jest wprost do adresata. W przeciwnym wypadku tablica rutowania przeszukiwana jest w celu znalezienia rekordu z informacjami o trasie do danej podsieci, a po znalezieniu takiego rekordu określany jest adres kolejnego przeskoku.
Jeśli i to się nie powiedzie, jako ostatnia deska ratunku traktowane jest wyszukanie przez algorytm rekordu z informacją o rutowaniu domyślnym. Rutowanie domyślne wskazuje zwykle inteligentniejszy ruter (taki, który ma pełniejszą tablicę rutowania), ale może również wskazywać ruter, który jest bliżej głównej sieci IP (rdzenia) niż nadawca.
Jeśli algorytm nie jest w stanie określić kolejnego przejścia, zwraca komunikat o tym, że adres przeznaczenia nie jest osiągalny. Informacja ta wysyłana jest bezpośrednio do programu użytkownika (jeśli to komputer wysyłający pakiet nie może znaleźć kolejnego przejścia) lub przy użyciu protokołu *Internet Control Mcssage Protocol (ICMP).*

Bezklasowe rutowanie IP

Wraz z wprowadzeniem supersieci algorytm rutowania musi być uaktualniony tak, by mógł pracować z arbitralnie określoną częścią przestrzeni adresów IP. W każdym wpisie w tablicy rutowania konieczne jest umieszczenie adresu przeznaczenia i adresu kolejnego przeskoku, a także maski, która pozwoli określić wielkość przestrzeni adresowej opisywanej przez ten zapis. Dodanie tej maski do rekordu umieszczanego w tablicy rutowania pozwala na uogólnienie algorytmu rutowania klasowego do postaci algorytmu bezklasowego. Implementacja części wyszukującej w takim algorytmie jest znacznie bardziej skomplikowana niż w przypadku algorytmu klasowego, ale za to sam algorytm jest znacznie prostszy:

```
Dla docelowego adresu IP:
przeszukaj tablicę routingu w poszukiwaniu najdłuższego prefiksu pasującego do danego adresu,
        jeśli (znaleziono pasujący wpis):
                odczytaj adres następnego skoku ze znalezionego wpisu,
                wyślij pakiet pod znaleziony adres następnego skoku.
        jeśli (nie znaleziono pasującego wpisu):
                odrzuć pakiet z komunikatem „destination unreachable”
```

Prefiks oznacza tu część pozostałą z adresu IP po zamaskowaniu, np.:
192.168.44.1/8 daje prefiks 192.
192.168.44.1/16 daje prefiks 192.168.
Przy wyborze trasy porównywany jest tylko prefiks adresu docelowego z prefiksem wpisu w tablicy routingu. Patrz przykład pod koniec rozdziału.

Pierwszą widoczną różnicą jest fakt, że algorytm ten jest znacznie prostszy i mniej szczegółowy od algorytmu działającego w oparciu o sieci z klas. Umieszczenie masek sieci w tablicy rutowania pozwala redukować większość z

działań nietypowych, koniecznych do wykonania w algorytmie klasowym. Na przykład trasy do hosta są w tym algorytmie zapisami z maską 255.255.255.255. Ponieważ takie 32-bitowe maski zawsze odpowiadają adresom przeznaczenia o przedrostku dłuższym niż jakakolwiek podsieć, sieć lub supersieć, są one zawsze preferowane przed mniej jednoznacznymi trasami, podobnie jak to miało miejsce w przypadku algorytmu klasowego.
Także trasa domyślna, jeśli istnieje, zapisana jest w postaci rekordu z adresem przeznaczenia 0.0.0.0 i maską 0.0.0.0. Jeśli maska ta zostanie użyta w stosunku do dowolnego adresu przeznaczenia, wynikiem będzie zawsze 0.0.0.0, co zawsze odpowiada adresowi przeznaczenia umieszczonemu w tym rekordzie. Powstały w ten sposób przedrostek będzie jednak zawsze krótszy niż jakakolwiek inna określona trasa, która może prowadzić do danej sieci, podsieci lub supersieci, co powoduje, że trasa ta pozostaje nadal trasą wybieraną na samym końcu.

Przydatną konsekwencją wymagania dotyczącego „najdłuższego dopasowania" jest możliwość umieszczenia w tablicy rutowania mniej określonej trasy, prowadzącej na przykład do supersieci oraz lepiej określonej trasy prowadzącej do podsieci. Obie te trasy prowadzą do adresu przeznaczenia pakietów, ale mają inny adres kolejnego przeskoku. Pozwala to na użycie jednego zapisu trasy prowadzącej do większości supersieci i dodanie zapisów tras, które zapełnią dziury w rutowaniu wynikające z tego ogólnego zapisu.
Jest to wprawdzie przydatne, lecz należy unikać tworzenia zbyt dużej liczby dziur w bloku sieci lub bloku adresów, ponieważ nie pozwalają one na stworzenie małych, wydajniej pracujących tablic rutowania. Pamiętaj o tym, że jeśli masz dziury w bloku sieci lub w bloku adresów, to poza zapisami w tablicy rutowania, definiującymi trasę do super sieci lub sieci, musisz dopisać trasy odnoszące się do każdej z tych dziur.

Przykład tablicy rutingu systemu z obsługą rutingu bezklasowego:

```
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
192.168.1.0     192.168.44.1    255.255.255.0   UG    0      0        0 eth0
192.168.44.0    *               255.255.255.0   U     0      0        0 eth0
127.0.0.0       *               255.0.0.0       U     0      0        0 lo
default         *               0.0.0.0         U     0      0        0 ppp0
```

co oznacza:

1. Jeśli masz pakiet do wysłania pod adres zaczynający się od 192.168.1. to prześlij go do routera o adresie 192.168.44.1 który jest podłączony do sieci na twoim interfejsie eth0.
2. Jeśli masz pakiet do wysłania pod adres zaczynający się od 192.168.44. to wyślij go po prostu na interfejs eth0 – adresat jest też podłączony bezpośrednio do tej sieci, więc go odbierze.
3. Jeśli masz pakiet do wysłania pod adres zaczynający się od 127. to wyślij go na interfejs lo (loopback).
4. Jeśli masz pakiet do wysłania pod dowolny adres (maska 0 bitów) wyślij go przez interfejs ppp0.

Należy pamiętać, że jeśli adres pasuje do kilku tras, to używana jest ta z najdłuższą maską (np.: 192.168.44.0/24 jest ważniejsza niż 192.168.0.0/16).

Utrzymywanie tablicy rutowania

Ponieważ każde urządzenie w sieci IP przesyła pakiet IP do punktu kolejnego przejścia (next-hop - bez zapamiętywania całej trasy tego pakietu), aż do punktu przeznaczenia, wszystkie urządzenia, a zwłaszcza wszystkie rutery, muszą na bieżąco tworzyć sobie obraz tras prowadzących w każdym z kierunków. Innymi słowy, najważniejsza jest synchronizacja tablic rutowania pomiędzy współpracującymi ze sobą ruterami.
Aby zrozumieć, dlaczego jest ona niezbędna, rozważmy przypadek, w którym ruter A i ruter B wierzą, że ten drugi jest poprawną trasą kolejnego przeskoku do adresu przeznaczenia 10.0.0.1. Kiedy ruter A odbierze pakiet przeznaczony dla 10.0.0.1, prześle go do rutera B. Ruter B z kolei przejrzy swoją tablicę rutowania i stwierdzi, że ruterem kolejnego przeskoku dla tego adresu jest ruter A, po czym odeśle pakiet do tego rutera. W rezultacie otrzymamy *pętlę rutowania,* którą mogą tworzyć więcej niż dwa rutery.

Synchronizacja tablic rutowania może być wykonywana kilkoma metodami. Najprostszą do opanowania i wdrożenia jest rutowanie *statyczne.* W rutowaniu statycznym każdy z ruterów jest ręcznie konfigurowany, a do jego tablicy wpisywana jest lista adresów przeznaczenia i informacja o adresie kolejnego przejścia dla tych adresów.
W takim przypadku tablica rutowania jest przechowywana w pliku konfiguracyjnym, umieszczonym na trwałym nośniku. Zadaniem administratora sieci jest upewnienie się, czy wszystkie tablice rutowania współpracujących ze sobą ruterów są spójne. To administrator musi sprawdzić, czy nie powstały jakieś pętle rutowania, a także czy wszystkie kierunki są osiągalne ze współpracujących ruterów.

Prostota konfiguracji rutowania statycznego odnosi się do sieci, z których pakiety wychodzą do niewielu punktów lub do sieci końcowych, które mają tylko jedno lub dwa połączenia z resztą sieci. Jednak i ta konfiguracja nie jest pozbawiona wad. Najważniejszą z nich jest to, że rutowanie statyczne nie potrafi adaptować konfiguracji sieci do uszkodzeń, które w niej występują ani też wykorzystywać zalet istnienia trasy alternatywnej prowadzącej do punktu docelowego. Ponadto kiedy liczba kierunków wysyłania pakietów, a także liczba ruterów, wzrośnie, uaktualnianie tablic rutowania przy zmianie topologii sieci staje się trudne i czasochłonne.

Elastyczniejsze rozwiązania stosują protokoły rutowania pozwalające ruterom na dynamiczne tworzenie tablic rutowania w oparciu o informacje przesyłane z innych ruterów pracujących w sieci. Opracowano i wdrożono wiele takich protokołów.
Mówiąc ogólnie, rutery rozmawiają ze sobą stosując protokół, który potrafi dynamicznie ustalać bieżącą topologię sieci. Na podstawie tych informacji każdy z ruterów ustala rutery (jeden lub więcej) kolejnego przejścia do danego punktu przeznaczenia próbując określić najlepszą trasę. Jeśli nic nie będzie zakłócało komunikacji pomiędzy ruterami i jeśli wszystkie z nich będą poprawnie stosowały wspomniany protokół, to obliczą pasujące do siebie tablice rutowania.

Pomiędzy krańcowo różnymi rozwiązaniami, jakimi są rutowanie statyczne oraz rutowanie dynamiczne, istnieje wiele rozwiązań, które są połączeniem zalet funkcji dynamicznych i funkcji statycznych. Takie hybrydowe sposoby rutowania pozwalają znaleźć rozwiązanie mające zalety elastyczności rutowania dynamicznego i prostoty rutowania statycznego. Na przykład rutery pracujące w sieci mogą używać rutowania dynamicznego, a hosty przyłączone do pojedynczych sieci mogą mieć skonfigurowaną trasę domyślną. Możliwe jest również takie skonfigurowanie rutera, aby miał on w tablicy kilka tras statycznych, prowadzących na przykład do obszarów sieci znajdujących się poza kontrolowaną przez administratora domeną oraz *rozgłaszał* trasy do innych ruterów, wykorzystując dynamiczny protokół rutowania.