Piotr Fulmański

Computer networks

Lecture notes

Łódź, 2013

(revision: 11 listopada 2013)

Spis treści

ROZDZIAŁ 1

Przedmowa

- Tekst pisany na czerwono oznacza tekst, który musi zostać sprawdzony pod względem merytorycznym.
- Tekst pisany na niebiesko oznacza tekst poprawny pod względem merytorycznym, który jednakże musi zostać przeedytowany.

ROZDZIAŁ 2

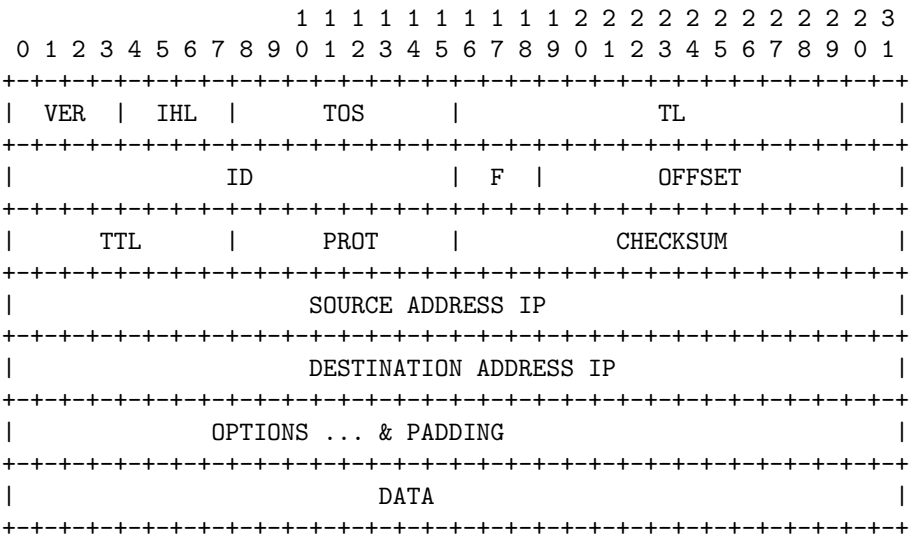
Protokół IP wersja 4

Protokół IP ([?]) (ang. *internet protocol*) jest protokołem warstwy 3 modelu ISO/OSI. Mówiąc najogólniej, dzięki przenoszonym przez niego informacjom adresowym dane mogą zostać dostarczone od nadawcy do odbiorcy.

Aby umożliwić intersieci udostępnianie jednolitego systemu komunikacyjnego, oprogramowanie ukrywa szczegóły sieci fizycznych, oferując udogodnienia dużej sieci wirtualnej. Wirtualna intersieć działa podobnie do każdej innej sieci, umożliwiając komputerom wysyłanie i odbieranie pakietów z informacjami. Główna różnica polega na tym, że intersieć jest jedynie modelem działającym dzięki odpowiedniemu oprogramowaniu.

Krytycznym elementem modelu intersieci jest adresowanie. Aby dawać obraz pojedyńczego, jednolitego systemu, wszystkie komputery muszą wykorzystywać jednolity schemat adresowania, a każdy adres musi być jednoznaczny. Fizyczne adresy sieciowe są w tym przypadku nieodpowiednie, gdyż intersieć może obejmować wiele technik sieciowych, z których każda może definiować własny format adresu.

Aby zagwarantować jednolite adresowanie we wszystkich węzłach, oprogramowanie protokołów określa schemat adresowania, który jest **nie zależny** od bazowych adresów fizycznych. Chociaż schemat adresowania w intersieci jest realizowany przez oprogramowanie, adresy protokołowe są wykorzystywane jako punkty docelowe w intersieci, tak jak adresy sprzętowe są wykorzystywane jako punkty docelowe w sieciach fizycznych. Aby wysłać pakiet przez intersieć, nadawca umieszcza protokołowy adres odbiorcy w pakiecie i przekazuje pakiet do oprogramowania protokołu w celu wysłania. Oprogramowanie i sprzęt wykorzystują protokołowy adres docelowy przy przekazywaniu pakietu poprzez intersieć do komputera odbiorcy wiążąc go z jego adresem fizycznym – szczegóły tego procesu zo-



Rysunek 2.1: Nagłówek protokołu IP

aczymy w dalszej części. Dwa programy komunikują się, nie znając swoich adresów fizycznych. W stosie protokołów TCP/IP adresowanie zdefiniowane jest w protokole intersieci – Internet Protocol – IP.

2.1 Opis pakietu IP

Opis protokołu rozpoczniemy od przedstawienia datagramu (pakietu) IP. Zaznaczmy przy tym, że mówiac protokół IP mamy zawsze na myśli wersję 4 tego protokołu (czyli IPv4). Datagram taki składa się z nagłówka IP oraz następujących po nim danych.

Nagłówek IP składa się z części stałej (20 bajtowej) oraz części opcjonalnej o zmiennej długości. Format nagłówka przedstawiono na rysunku 2.1

VER (ang. *Version*, 4 bity) Wersja nagłówka IP; określa format nagłówka IP. Wartość równa 4 oznacza standardowy nagłowek wersji 4 protokołu IP.

IHL (*ang.Internet Header Length*, 4 bity) Określa długość pakietu IP w 32 bitowych słowach*. Minimalna wartość dla poprawnego nagłówka wynosi 5.

TOS (ang. *Type Of Service*, 8 bitów) Rodzaj usługi. Parametry te mogą wpływać na sposób traktowania pakietu przez urządzenia sieci podczas jego przesyłania. Na przykład bardzo ważne

*To znaczy wielokrotnościach 32 bitów; 32 bity to 4 bajty.

pakiety można oznaczyć etykietą „wysokiego priorytetu". Bardziej szczegółowo pole to dzieli się na następujące fragmenty

```
 012.3.4.5.6.7
|P  |D|T|R|M|0|
```

Znaczenie ich jest następujące

P (ang. *Precedence*, 3 bity) Określa „ważność" pakietu – od 0 (zwykły) do 7 (sterujący siecią a więc najważniejszy).

```
0 - Routine
1 - Priority
2 - Immediate
3 - Flash
4 - Flash override
5 - CRITIC/ECP
6 - Internetwork control
7 - Network control
```

D (ang. *Delay*, 1 bit) Określa opóźnienie.

```
0 - Normal delay
1 - Low delay
```

T (ang. *Throughput*, 1 bit) Określa przepustowość.

```
0 - Normal throughput
1 - High throughput
```

R (ang. *Reliability*, 1 bit) Określa niezawodność.

```
0 - Normal reliability
1 - High reliability
```

M (ang. *Monetary*, 1 bit) Określa koszt przesłania.

```
0 - Normal monetary cost
1 - Minimize monetary cost
```

0 (1 bit) Puste

TL (ang. *Total Length*, 16 bitów) Długość datagramu a więc długość nagłówka razem z danymi. Największą liczbą jaką na 16 bitach można zapisać jest 65536 (2^{16}).

ID (ang. *Identification*, 16 bitów) Pozwala jednoznacznie odróżnić wysyłane datagramy. Mówiąc precyzyjniej: w przypadku podziału datagramu na mniejsze części, pole to pozwala na jego ponowne „złożenie". Pole to musi mieć unikalną wartość dla pary nadawca-odbiorca.

F (ang. *Flags*, 3 bity) Flagi

```
 0. 1. 2
|R|DF|MF|
```

Znaczenie ich jest następujące

R (ang. *Reserved*, 1 bit) Zarezerwowany; jego wartość powinna być równa 0.

DF (ang. *Don't Fragment*, 1 bit) Oznacza sposób przesyłania datagramu.

```
0 - Fragment if necessary
1 - Do not fragment
```

MF (ang. *More Fragments*, 1 bit) Określa czy dany fragment jest ostatnim fragmentem datagramu.

```
0 - This is the last fragment
1 - More fragments follow this fragment
```

OFFSET (ang. *Fragment Offset*, 13 bitów) Używany do określenia kolejności fragmentów wiadomości wchodzących w skład większego datagramu. Wszystkie fragmenty jednego datagramu mają to samo ID. Wszystkie fragmenty datagramu z wyjątkiem ostatniego muszą mieć długość równą wielokrotności 8 bajtów. Ponieważ pole to ma 13 bitów, więc datagram może mieć maksymalnie 8192 fragmenty (2^{13}) czyli 65536 bajtów ($8192 \cdot 8$) a więc o jeden więcej niż pozwala na to pole TL.

(TTL) (ang. *Time To Live*, 8 bitów) Określa czas życia datagramu. Gdy liczba zapisana w tym polu osiągnie wartość 0, datagram jest usuwany.

PROT (ang. *Protocol*, 8 bitów) Określa jaki protokół zawarty jest w datagramie IP. Możliwe wartości to między innymi[†]

```
 1 - ICMP - Internet Control Message Protocol
 2 - IGAP - IGMP for user Authentication Protocol
     IGMP - Internet Group Management Protocol
     RGMP - Router-port Group Management Protocol
 6 - TCP  - Transmission Control Protocol
17 - UDP  - User Datagram Protocol
```

CHECKSUM (ang. *Header Checsum*, 16 bitów) Suma kontrolna nagłówka IP. Według algorytmu, dodawane są, przy użyciu arytmetyki uzupełnień jedynkowych, 16-bitowe liczby; jako sumę kontrolną bierzemy uzupełnienie jedynkowe liczby otrzymanej w wyniku tego dodawania.DOPISAC jak (z czego) liczona jest ta suma? naglowka? naglowka + dane? jaka czesc naglowka skoro czescia naglowka jest suma kontrolna

SOURCE IP ADDRESS (16 bitów)Adres IP nadawcy.

DESTINATION IP ADDRESS (16 bitów) Adres IP odbiorcy.

OPTIONS (różna długość) Opcje. Pole to dzieli się na następujące fragmenty

```
 0.12.34567
|C|CL| OPT |
```

Znaczenie ich jest następujące

C (ang. *Copy*, 1 bit)

```
0 - Do not copy
1 - Copy
```

CL (ang. *Class*, 2 bity)

```
0 - Control
1 - Reserved
```

[†]Kiedyś wartości te były określone przez RFC 1700; aktualny obecnie spis dostępny jest pod adresem `www.iana.org`

`2 - Debugging and measurement`

`3 - Reserved`

OPT (ang. *Option*, 5 bitów) Jedną z dostępnych opcji jest na przykład możliwość śledzenia trasy jaką podąża datagram.

PADDING Używane do wypełnienia pustego miejsca aby zapewnić, że dane zaczną się na granicy 32 bitowego słowa.

DATA Dane

2.2 Cechy protokołu IP

DOPISAC!!! niepewnosc, bezpolaczeniowosc

2.3 Podstawy adresowania IP

2.3.1 Główne reguły

Rozważania na temat adresów IP rozpoczniemy od, historycznie rzecz biorąc, pierwszego ujęcia tego tematu mianowicie **klasowego adresowania IP**. Dobre zrozumienie tego tematu ułatwi przejście do **bezklasowego adresowania IP** o którym mówić będziemy w dalszej części. Co jest ważne, zasady poznane tutaj przenoszą się także na późniejsze rozważania.

Standard IPv4 określa, że każdy węzeł ma przypisany 32-bitowy numer, zwany adresem węzła w protokole intersieci, lub po prostu adresem IP. Każdy pakiet wysyłany przez intersieć zawiera zarówno adres IP odbiorcy jak i nadawcy. Ponieważ tak określona przestrzeń adreswowa zawiera 4 294 967 296 różnych adesów więc wskazane jest aby przydzielanie tych adresów odbywało się zgodnie z pewnymi regułami. Reguły te określają

- jakie adresy mogą być użyte w różnych sieciach (zgrubny podział globalny dostępnej puli adresowej);
- jakie adresy mogą być użyte w jakiej sieci (podział lokalny przyznanej puli adresowej).

Główne reguły przydzielania adresów są następujące (porównaj rysunek 2.2).

- Adresy dzielone są na grupy.

rysunek 9.13 s 414

Rysunek 2.2: Przydział adresów w sieci LAN

- Numery dostępne w grupie odpowiadają kolejno po sobie następującym liczbom (z dopuszczalengo zakresu wyznaczonego przez wielkość liczby binarnej).
- Urządzenia znajdujące się w różnych sieciach LAN (odseparowanych przez co najmniej jeden router) powinny używać adresów z różnych grup.
- Urządzenia w tej samej sieci LAN powinny używać adresów z tej samej grupy (w dalszej części grupę taką nazywać będziemy siecią IP).
- W obrębie jednej sieci LAN różnym urządzeniom (mówiąc bardziej precyzyjnie: różnym interfejsom, gdyż jedno urządzenie może mieć wiele interfejsów) przypisywane są różne numery.

Rysunek (2.2) obrazuje przydział adresów zgodnie z powyższymi regułami. Jak widać jednej sieci przyporządkowana została grupa adresów rozpoczynających się od 64. drugiej od 65 i trzeciej od 66.

2.3.2 Notacja dziesiętna z kropką

Chociaż adresy IP są 32-bitowymi liczbami binarnymi, użytkownicy rzadko wpisują lub czytają ich wartość w tej postaci, stosując zamiast niej notację dziesiętną z kropką. W tym sposobie zapisu każda 8-bitowa część 32-bitowej liczby jest wyrażona jako wartość dziesiętna, zaś kropki są wykorzystywane jako separatory części.

Przykład 2.1. *Notacja "zwykła"*

```
10000001 00110100 00000110 00000000
```

Ten sam adres zapisany z wykorzystaniem notacji dziesiętnej z kropką

```
129.52.6.0
```

W ten sposób adresy dziesiętne z kropką sięgają od $0.0.0.0$ do $255.255.255.255$.

2.3.3 Klasy adresowe

Sformułowanie „*grupa adresów rozpoczynających się od*" czytelne dla człowieka jest mało użyteczne dla komputera. Tym bardziej, iż w podanym przykładzie zamiast *grupa adresów rozpoczynających*

się od 64 można by, teoretycznie, powiedzieć *grupa adresów rozpoczynających się od 64.10*. Skąd więc wiadomo co jest częścią wspólną adresów?

Każdy adres IP podzielony jest na dwie części: część „wspólną", czyli **prefiks** oraz **sufiks**. Prefiks identyfikuje sieć fizyczną, do której jest podłączony fizycznie komputer. Sufiks wskazuje konkretny komputer (lub jego konkretny interfejs jeśli posiada ich więcej) w danej sieci. Zgodnie z zasadami podanymi wcześniej, żadne dwie sieci nie mogą mieć przyznanego tego samego numeru jak i żadne dwa komputery w ustalonej sieci nie mogą posiadać identycznego numeru. Inaczej: jeśli dwa komputery są przyłączone do różnych sieci fizycznych, to mają różne prefiksy; jeśli dwa komputery są podłączone do tej samej sieci fizycznej, to ich adresy mają różne sufiksy. Powstała w ten sposób hierarchia adresów IP gwarantuje dwie ważne własności:

- każdy komputer ma przyznany jednoznaczny adres,
- chociaż przypisanie numerów sieci muszą być koordynowane globalnie, sufiksy mogą być przyznawane lokalnie bez globalnego uzgadniania.

Zauważmy, że takie rozwiązanie pozwala na rozsądne powiązanie urządzeń ze sobą - na podstawie analizy adresu można odgadnąć np. gdzie znajduje się dana maszyna. Dużo łatwiejsze jest także zarządzanie adresami bo adres globalnie przydziela się sieci a nie każdemu hostowi z osobna a taka sytuacja miałaby miejsce gdyby nie opisany podział.

Po określeniu rozmiaru pojedyńczego adresu należało zdecydować ile bitów przeznaczyć na każdą z dwóch części. Żaden prosty wybór nie był tu możliwy, ponieważ dodanie bitów do jednej z nich powodowało zabranie ich z drugiej. Obranie długiego prefiksu jest odpowiednie w przypadku istnienia wielu sieci, ale powoduje ograniczenia rozmiaru każdej z nich. Obranie długiego sufiksu oznacza, że każda sieć fizyczna może zawierać wiele komputerów, ale całkowita liczba sieci jest wówczas ograniczona.

Ze względu na to, że intersieć może obejmować dowolne techniki sieciowe zbudowane z „mieszaniny" dużych i małych sieci, podzielono przestrzeń adresową na trzy podstawowe klasy[‡] (A, B, C) o różnych rozmiarach prefiksu i sufiksu (patrz rysunek 2.3) oraz dwie klasy specjalne.

Dla (klasowych) adresów IP szczególnie użyteczna jest notacja z kropkami, gdyż w adresach tych podział na prefiks i sufiks jest na granicy oktetów. W przypadku adresów klasy A ostatnie trzy oktety odpowiadają sufiksowi komputera, klasy B – ostatnie dwa, a w adresach klasy C – jeden oktet.

[‡]Obecnie adresy IPv4 uważa się za adresy bezklasowe. Jak jednak zostało to napisane na początku rozdziału, podaję te informacje mając nadzieję, że pomogą one zrozumieć inne zagadnienia czy idee.

Klasa A: max. liczba sieci – 128, max. liczba komputerów w sieci 16777216, łącznie 2 147 483 648 adresów co stanowi 50% całej przestrzeni adresowej.

```
0|1 2 3 4 5 6 7 | 8 9 10 11 12 13 14 ... 25 26 27 28 29 30 31
0| Prefiks - 7 b|    Sufiks - 24 bity
```

Klasa B: max. liczba sieci – 16384, max. liczba komputerów w sieci 65536, łącznie 1 073 741 824 adresów co stanowi 25% całej przestrzeni adresowej.

```
0 1|2 3 4 5 6 ... 12 13 14 15|16 17 18 19  ...  28 29 30 31
1 0| Prefiks - 14 bitów      |   Sufiks - 16 bitów
```

Klasa C: max. liczba sieci – 2097152, max. liczba komputerów w sieci 256, łącznie 536 870 912 adresów co stanowi 12,5% całej przestrzeni adresowej.

```
0 1 2|3 4 5 6 ... 18 19 20 21 22 23|24 25 26 27 28 29 30 31
1 1 0| Prefiks - 21 bitów          | Sufiks - 8 bitów
```

Klasa D – adres rozgłaszania grupowego (RFC 1112) (6,25% całej przestrzeni adresowej).

```
0 1 2 3|4 5 6 ... 18 19 20 21 22 23 24 25 26 27 28 29 30 31
1 1 1 0|  28 bitów
```

Klasa E – zarezerwowane na przyszłość (6,25% całej przestrzeni adresowej).

```
0 1 2 3|4 5 6 ... 18 19 20 21 22 23 24 25 26 27 28 29 30 31
1 1 1 1|  28 bitów
```

Rysunek 2.3: Klasy adresowe

Niestety w tej postaci nie widać poszczególnych bitów adresu, przez co klasę musimy rozpoznwać na podstawie wartości dziesiętnej pierwszego oktetu.

Klasa	Zakres wartości
A	1 – 126
B	128 – 191
C	192 – 223
D	224 – 239
E	240 – 255

Zauważmy, iż w kolumnie „Zakres wartości" nie występują wartości 0 i 127, które zostały trwale zarezerwowane do celów specjalnych.

2.3.4 Adresy IP specjalnego przeznaczenia

IP określa zestaw adresów o szczególnej postaci, które są zarezerwowane

Adresy sieciowe. (NET.0) IP rezerwuje adres zerowy węzła w danej sieci i wykorzystuje go przy odwoływaniu się do sieci. Odnosi się on do samej, a nie do komputerów podłączonych do niej. Adres ten oznacza prefiks przyznany sieci. Na przykład adres $128.211.0.0$ oznacza sieć, której przyznano prefiks klasy B równy 128.211.

Adres rozgłaszania ukierunkowanego. (NET.255) Używany jest w celu przesłania pakietu do wszystkich węzłów w danej sieci fizycznej. Gdy jest wysyłany pakiet pod adres rozgłaszania ukierunkowanego danej sieci, przez intersieć podróżuje tylko jedna jego kopia, aż dotrze do sieci. Następnie pakiet ten jest dostarczany do wszystkich węzłów tej sieci.

Adres rozgłaszania ukierunkowanego danej sieci jest tworzony przez dodanie do jej prefiksu sufiksu, któty składa się z samych jedynek. IP rezerwuje adres węzła, którego wszystkie bity to jedynki. Zatem sufiks składa się z liczb 255.

Adres rozgłaszania ograniczonego. (255) Termin rozgłaszanie ograniczone odnosi się do rozgłaszania w lokalnej sieci fizycznej. Jest ono używane na przykład przy starcie systemu przez komputey, które nie znają w tym momencie numeru sieci. IP na rozgłaszanie ograniczone rezerwuje adres składający się z samych jedynek.

Adres bieżącego komputera. (0) Każdy pakiet musi zawierać adres odbiorcy i nadawcy; komputer by wysłać i odebrać pakiety, musi znać swój adres IP. Zestaw protokołów TCP/IP obejmuje protokoły, których można użyć przy automatycznym uzyskiwaniu adresu IP przy starcie. Co ciekawe, protokoły startowe do komunikacji używają IP. Komputer korzystając z takich protokołów uruchomieniowych, nie może podać prawidłowego adresu IP nadawcy. Aby radzić sobie w takich sytuacjach, w IP zarezerwowano adres, który składa się z samych zer, na oznaczenie bieżącego komputera.

Adres pętli zwrotnej. (127.X) Protokół IP rezerwuje prefiks sieciowy klasy A równy 127 na adres pętli zwrotnej. Adres węzła (sufiks) używany przytym jest bez znaczenia. Najpopularniejszym adresem pętli zwrotnej jest $127.0.0.1$. Adres ten przydaje się na przykład przy testowaniu programów sieciowych. Używając pętli zwrotnej, żadne pakiety nigdy nie opuszczą komputera – oprogramowanie IP przekazuje pakiety z jednego programu użytkownika do drugiego.

Adres ??? (0.X) ??? from RFC 1122 section 3.2.1.3, see rfc 5735, section 3

0, 0

```
jak rys:ip_example_01 ale z uwzględnieniem informajc z rys. 9.17 s. 420
[to nie podobnie do 9.18 s. 425]
```

Rysunek 2.4: Przydział adresów w sieci LAN uwzględniający adresy specjalnego przeznaczenia

```
jak rys:ip_example_02 ale z uwzględnieniem informajc z rys. 9.18 s. 425
```

Rysunek 2.5: Przykład podziału na podsieci

> This host on this network. MUST NOT be sent, except as a source address as part of an initialization procedure by which the host learns its own IP address.
>
> See also Section 3.3.6 for a non-standard use of 0,0.
>
> (b) 0, ¡Host-number¿
>
> Specified host on this network. It MUST NOT be sent, except as a source address as part of an initialization procedure by which the host learns its full IP address.

Przykład wykorzystania tych adresów zawiera rysunek 2.4.

2.4 Idea podziału na podsieci

W części 2.3 powiedziane zostało, iż „*Urządzenia znajdujące się w różnych sieciach LAN (odseparowanych przez co najmniej jeden router) powinny używać adresów z różnych grup.*" Gdyby jako grupę uważać jedną sieć klasową IP (to znaczy sieć o numerze należącym do jednej klasy) mogło by to doprowadzić do znacznego marnowania adresów. Jako przykład wystarczy rozważyć firmę z adresem sieci klasy A – prawdopodobieństwo, że działa w niej ponad 16 milionów urządzeń i to w jednej sieci LAN nie jest za duże. Dlatego umożliwiono podział jednej klasowej sieci IP na mniejsze części nazywane **podsieciami**. Utworzone podsieci podlegają takim samym zasadom jak sieci. Na rysunku 2.5 zamieszczono przykład obrazujący ideę podsieci. Jest on tylko podobny do sieci z rysunku 2.4. Zasadnicza różnica polega na tym, iż zamiast trzech sieci IP odpowiadających trzem sieciom LAN podzielono jedną sieć IP na trzy podsieci i każdą z nich przypisano do jednej sieci LAN. W konsekwencji obserwujemy zastosowanie zasad dotyczących sieci w kontekście podsieci

- każda podsieć, odpowiadająca jednej fizycznie sieci LAN, ma swój jednoznaczny numer (używa adresów z różnych grup), analogiczny do uprzednio wykorzystywanych numerów sieci IP, ale składający się trzech oktetów zamiast jednego;

- adresy IP hostów w jednej sieci LAN mają identyczną wartość początkową (tym razem pierwsze trzy oktety zamiast jednego jak poprzednio) – należą do tej samej grupy;
- w obrębie jednej podsieci różnym urządzeniom przypisane są różne numery.

W przykładzie jakim posługiwaliśmy się do tej pory używana była sieć klasy A z uwzględnieniem reguł podziału na podsieci. Oznacza to, iż pierwsze 3 oktety adresów hostów w tej samej podsieci musiały być jednakowe. Tak więc mieliśmy, niejako z definicji klasy adresowej A, następujace pola

- 1-oktetowe pole sieci,
- 2-oktetowe pole podsieci,
- 1-oktetowe pole hosta.

W procesie podziału na podsieci można jednak wybrać rozmiar pola podsieci adresów. Zasada podziału jest bardzo prosta. Oto **nie ruszając** numeru sieci, na potrzeby wskazania numeru podsieci „zabieramy" część bitów przeznaczonych na numer hosta. W tym celu standard IP definiuje liczbę nazywaną **maską podsieci** informującą, które bity są adresem sieci i podsieci.

Spójrzmy na następujący przykład. Załóżmy, że przyznano nam następujący adres sieci klasy A

```
65.0.0.0
```

czyli dwójkowo

```
01000001.00000000.00000000.00000000
```

Zapiszmy teraz pod dwójkowym ciągiem będącym adresem sieci, pewien ciąg złożony z zer i jedynek według następującej zasady: 1 piszemy pod bitami wchodzącymi w skład adresu sieci, 0 pod bitmi wchodzącymi w ewentualne numery hostów w ramach tejże sieci.

```
01000001.00000000.00000000.00000000
11111111.00000000.00000000.00000000
```

Otrzymaliśmy w ten sposób maskę sieci a więc „coś" informującego nas o tym jaka część adresu jest adresem sieci a jaka jest adresem hosta. Zwykle zarówno adres jak i związaną z nim maskę będziemy zapisywać z użyciem notacji kropkowej; a więc w tym przypadku otrzymujemy

```
65.0.0.0 - adres sieci,
255.0.0.0 - maska sieci
```

lub równoważnie

65.0.0.0/8.

Spróbujmy teraz podzielić naszą sieć na kawałki. Załóżmy, że chcemy mieć 254 podsieci o maksymalnej liczbie hostów w każdej z nich. Tak więc, nie ruszając adresu sieci już przyznanego, możemy pożyczyć kilka bitów (w tym przypadku 8) z puli bitów przeznaczonych na adres hosta i dołączyć je do adresów sieci

01000001.xxxxxxxx.00000000.00000000

x - bity, które staną się adresem podsieci w ramach sieci 65.0.0.0

Wówczas maska sieci przyjmuje postać

01000001.xxxxxxxx.00000000.00000000
11111111.11111111.00000000.00000000 - maska

czyli

255.255.0.0

lub też

65.0.0.0/16

Tak więc pierwsza podsieć będzie używać adresów zaczynających się od 65.0.0.1, druga 65.1.0.1, trzecia 65.2.0.1 itd. Wyjaśnienie tak przyznanych adresów jest następujące.

Pierwszych 8 bitów, jako stanowiących odgórnie przyznany nam adres, nie możemy zmienić, stąd też pierwsza liczba pozostanie niezmieniona, a więc mamy 65. Następnie w ramach dostępnej przestrzeni adresowej 8 najmłodszych bitów postanawiamy przeznaczyć na adresy podsieci. Tak więc niech pierwsza podsiec ma numer 00000000, druga 00000001, trzecia 00000010 itd. Pozostaje nam do dyspozycji jeszcze 16 bitów. Najmniejszą możliwą do zapisania liczbą 16-bitową będącą poprawnym adresem hosta jest 00000000 00000001. Składając to wszystko razem, otrzymujemy, że ostatnie 24 bity będące adresem pierwszego hosta w każdej z podsieci będzą postaci

dla podsieci ,,pierwszej'', tj. 00000000: niepoprawna
00000000 00000000 00000001 czyli 0.0.1

```
dla podsieci ,,drugiej'', tj. 00000001:
00000001 00000000 00000001 czyli 1.0.1

dla podsieci ,,trzeciej'', tj. 00000010:
00000010 00000000 00000001 czyli 2.0.1
```

Ostatecznie otrzymujemy

```
Podsieć     Najmniejszy adres IP  Największy adres IP   Adres rozgłoszeniowy
65.0.0.0    65.0.0.1              65.0.255.254          65.0.255.255             niepopra
65.1.0.0    65.1.0.1              65.1.255.254          65.1.255.255
...
65.254.0.0  65.254.0.1            65.254.255.254        65.254.255.255
65.255.0.0  65.255.0.1            65.255.255.254        65.255.255.255           niepopra
```

Przedstawione tutaj rozumowanie uzasadnia wartości adresów i masek z rysunku 2.5. Zauważmy, iż w powyższym zestawieniu przy dwóch sieciach dopisano „niepoprawna": przy sieci o adresie 65.0.0.0 oraz podsiec rozgloszeniową. Pierwsza z nich jest tak zwaną **podsiecią zerową**. Podsieć zerowa ma najmniejszy liczbowo numer podsieci (bez względu na przyjęty schemat podziału na podsieci), charakteryzujący się tym, że w polu podsieci zawiera same zera. W przypadku klasowego adresowania IP podsieć ta jest jedną z dwóch zarezerwowanych podsieci, których nie wolno używać do innych celów. Drugą taką siecią jest **podsiecią rozgłoszeniowa**. Podsieć rozgłoszeniowa ma największy liczbowo numer podsieci (bez względu na przyjęty schemat podziału na podsieci), charakteryzujący się tym, że w polu podsieci zawiera same jedynki. Przez to adres rozgłoszeniowy takiej podsieci jest taki sam jak adres rozgłoszeniowy całej sieci.

Przyjrzyjmy się teraz jeszcze jednemu przykładowi. Załóżmy, że przyznano nam następujący adres sieci klasy B

```
130.12.0.0
```

czyli dwójkowo

```
10000010.00001100.00000000.00000000
```

Zapiszmy teraz pod dwójkowym ciągiem będącym adresem sieci, pewien ciąg złożony z zer i jedynek według poznanej zasady: 1 piszemy pod bitami wchodzącymi w skład adresu sieci, 0 pod bitmi wchodzącymi w ewentualne numery hostów w ramach tejże sieci.

```
10000010.00001100.00000000.00000000
11111111.11111111.00000000.00000000
```

Otrzymaliśmy w ten sposób maskę sieci

```
130.12.0.0 - adres sieci,
255.255.0.0 - maska sieci

lub równoważnie

130.12.0.0/16.
```

Spróbujmy teraz podzielić naszą sieć na kawałki. Załóżmy, że chcemy mieć 6 podsieci. Tak więc, nie ruszając adresu sieci już przyznanego, możemy pożyczyć jeszcze 3 bity z puli bitów przeznaczonych na adres hosta i dołączyć je do adresów sieci.

```
10000010.00001100.xxx00000.00000000

x - bity, które staną się adresem podsieci w ramach sieci 130.12.0.0

maska:

10000010.00001100.xxx00000.00000000
11111111.11111111.11100000.00000000 - maska

czyli

255.255.224.0

lub też
```

130.12.0.0/19

Tak więc pierwsza podsieć będzie używać adresów zaczynających się od 130.12.0.1, druga 130.12.32.1, trzecia 130.12.64.1 itd. Wyjaśnienie tak przyznanych adresów jest następujące.

Pierwszych 16 bitów, jako stanowiących odgórnie przyznany nam adres, nie możemy zmienić, stąd też dwie pierwsze liczby pozostaną niezmienione, a więc mamy 130.12. Następnie w ramach dostępnej przestrzeni adresowej 3 pierwsze bity postanawiamy przeznaczyć na adresy podsieci. Tak więc niech pierwsza podsiec ma numer 000, druga 001, trzecia 010 itd. Pozostaje nam do dyspozycji jeszcze 13 bitów. Najmniejszą możliwą do zapisania liczbą 13-bitową będącą poprawnym adresem hosta jest 00000 00000001. Składając to wszystko razem, otrzymujemy, że ostatnie 16 bitów będące adresem pierwszego hosta w każdej z podsieci będzi postaci

```
dla podsieci ,,pierwszej'', tj. 000: niepoprawna
00000000 00000001 czyli 0.1

dla podsieci ,,drugiej'', tj. 001:
00100000 00000001 czyli 32.1

dla podsieci ,,trzeciej'', tj. 010:
01000000 00000001 czyli 64.1
```

Ostatecznie otrzymujemy[§]

Podsieć	Najmniejszy adres IP	Największy adres IP	Adres rozgłoszeniowy	
130.12.0.0	130.12.0.1	130.12.31.254	130.12.31.255	niepopra
130.12.32.0	130.12.32.1	130.12.63.254	130.12.63.255	
...				
130.12.192.0	130.12.192.1	130.12.223.254	130.12.223.255	
130.12.224.0	130.12.224.1	130.12.224.254	130.12.224.255	niepopra

2.5 Sieci prywatne

Generaly speaking hosts[¶] within enterprises that use IP can be partitioned into three categories:

[§]W zestawieniu, podobnie jak miało to miejsce wcześniej, pominięto sieć zerową i rozgłoszeniową.

[¶]Znaczna część tego rozdziału stanowi przedruk dokumentu RFC 1918 [**?**].

Category 1: hosts that do not require access to hosts in other enterprises or the Internet at large; hosts within this category may use IP addresses that are unambiguous within an enterprise, but may be ambiguous between enterprises.

Category 2: hosts that need access to a limited set of outside services (e.g., E-mail, FTP, netnews, remote login) which can be handled by mediating gateways (e.g., application layer gateways). For many hosts in this category an unrestricted external access (provided via IP connectivity) may be unnecessary and even undesirable for privacy/security reasons. Just like hosts within the first category, such hosts may use IP addresses that are unambiguous within an enterprise, but may be ambiguous between enterprises.

Category 3: hosts that need network layer access outside the enterprise (provided via IP connectivity); hosts in the last category require IP addresses that are globally unambiguous.

We will refer to the hosts in the first and second categories as *private* and to the hosts in the third category as *public*.

Many applications require connectivity only within one enterprise and do not need external (outside the enterprise) connectivity for the majority of internal hosts. In larger enterprises it is often easy to identify a substantial number of hosts using TCP/IP that do not need network layer connectivity outside the enterprise.

Some examples, where external connectivity might not be required, are:

- A large airport which has its arrival/departure displays individually addressable via TCP/IP. It is very unlikely that these displays need to be directly accessible from other networks.
- Large organizations like banks and retail chains are switching to TCP/IP for their internal communication. Large numbers of local workstations like cash registers, money machines, and equipment at clerical positions rarely need to have such connectivity.
- For security reasons, many enterprises use application layer gateways to connect their internal network to the Internet. The internal network usually does not have direct access to the Internet, thus only one or more gateways are visible from the Internet. In this case, the internal network can use non-unique IP network numbers.
- Interfaces of routers on an internal network usually do not need to be directly accessible from outside the enterprise.

jak rys. 9.21 s. 431

Rysunek 2.6: Koncepcja użycia NAT i prywatnych sieci IP

The Internet Assigned Numbers Authority (IANA) has reserved the following three blocks of the IP address space for private internets:

```
10.0.0.0        -   10.255.255.255  (10/8 prefix)
172.16.0.0      -   172.31.255.255  (172.16/12 prefix)
192.168.0.0     -   192.168.255.255 (192.168/16 prefix)
```

We will refer to the first block as "24-bit block", the second as "20-bit block", and to the third as "16-bit" block. Note that (in pre-CIDR notation) the first block is nothing but a single class A network number, while the second block is a set of 16 contiguous class B network numbers, and third block is a set of 256 contiguous class C network numbers.

An enterprise that decides to use IP addresses out of the address space defined in this document can do so without any coordination with IANA or an Internet registry. The address space can thus be used by many enterprises. Addresses within this private address space will only be unique within the enterprise, or the set of enterprises which choose to cooperate over this space so they may communicate with each other in their own private internet.

Zdefiniowanie sieci prywatnych pozwala także, krótkoterminowo, rozwiązać problem wyczerpywania się adresów IPv4. Otóż w większości sytuacji mamy do czynienia z jeszcze jedną kategorią użytkowników niewymienioną wcześniej

Category 4: hosts które wymagają swobodnego dostępu do wszelkich usług dostępnych w sieci (Internecie), ale same nigdy takich usług nie będą oferowały.

W takim przypadku często najlepszym rozwiązaniem jest zastosowanie translacji NAT[‖] (ang. *Network Address Translation*). Pozwala ona zamiast całej sieci klasy A, B lub C używać jedynie kilku lub wręcz jednego adresu IP. Ogólną koncepcję tego rozwiązania przedstawiono na rysunku 2.6; więcej szczegółów poznamy w dalszej części. Stosując translację NAT można było przydzielić hostom w firmie dowolne adresy IP, choć do dobrych praktyk należy wykorzystanie w tym celu właśnie adresów prywatnych. Co jest istotne, z punktu widzenia hostów podłączonych do Internetu, używają one prawidłowych, globalnie unikalnych adresów IP. Jednak w Internecie wszystkie połączenia tych

[‖]Czasem nazywane PAT (ang. *Port Address Translation*).

jak rys. 9.25 s. 436

Rysunek 2.7: Stan przykładowej sieci przed przydzieleniem adresów

jak rys. 9.26 s. 437

Rysunek 2.8: Proces przydzielania adresów dla przykładowej sieci z rysunku 2.7

hostów są widziane tak, jakby pochodziły z jednego hosta o zarejestrowanym globalnym adresie IP przydzielonym przez IANA (ang. *Internet Assigned Numbers Authority*).

2.6 Przypisywanie i odwzorowywanie adresów IP

Omówimy teraz dwie metody przypisywania adresów IP

- konfiguracja statyczna,
- konfiguracja dynamiczna z użyciem protokołu DHCP (ang. *Dynamic Host Configuration Protocol*).

2.6.1 Statyczna konfiguracja adresów IP

DOPISAC

2.6.2 Dynamiczna konfiguracja adresów IP

Na rysunku 2.7 przedstawiona została przykładowa sieć przed przydzieleniem adresów do poszczególnych hostów. Rysunek 2.8 obrazuje proces przydzielania adresów dla sieci z rysunku 2.7. Proces ten przebiega w następujących krokach

1. DOPISAC!!!

Dla większości hostów pracujących w sieci najlepszym rozwiązaniem jest stosowanie dynamicznego przydzielania adresów z wykorzystaniem protokołu DHCP. W niektórych sytuacjach jednak zdecydowanie lepszym rozwiązaniem jest statyczne przydzielenie adresu, np.

- w przypadku serwerów;
- w przypadku routerów i przełączników;

jak rys. 9.27 s. 439 część górna

Rysunek 2.9: Przekazywanie pakietu z punktu widzenia adresów IP

jak rys. 9.27 s. 439 część dolna

Rysunek 2.10: Przekazywanie pakietu z punktu widzenia adresów IP i warstwy łącza danych

- ze względów bezpieczeństwa.

2.7 Protokół ARP

Zanim opiszemy protokół ARP przyjrzyjmy się procesowi przekazywania pakietu w sieci z kilku różnych punktów widzenia.

Widok ukierunkowany na adresy IP 2.9 DOPISAC!!!

Widok ukierunkowany na adresy IP i warstwę łącza danych 2.10 DOPISAC!!!

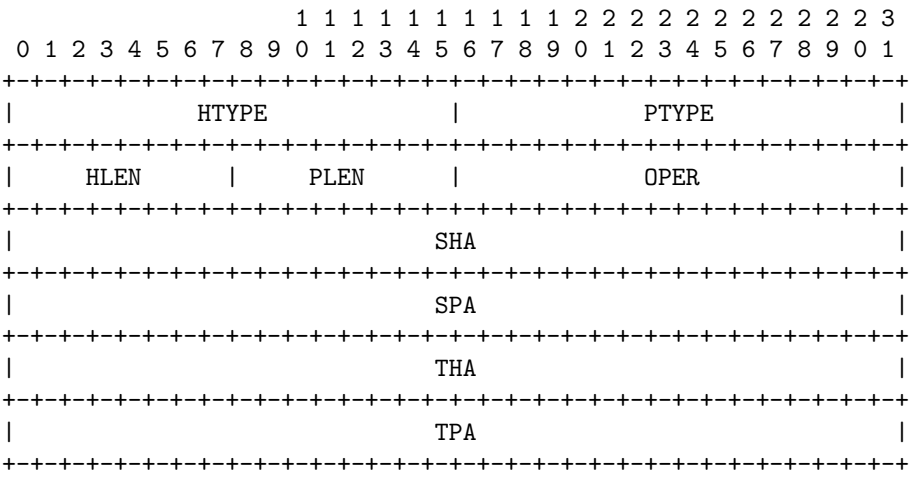
Opisane powyżej procesy pokazują, że o ile z punktu widzenia przeciętnego użytkownika istotne są adresy IP – wszak nimi posługujemy się powszechnie w komunikacji pomiędzy urządzeniami – to, aby faktycznie mogło dojść od połączenia, niezbędne jest posługiwanie się adresami warstwy łącza adnych (adresami MAC). Stąd ptrzeba istnienia systemu pozwalającego przekształcać adrsy IP na adresy sprzętowe.

2.7.1 Opis protokołu

Address Resolution Protocol (ARP), defined by RFC 826 in 1982, is a telecommunications protocol used for resolution of network layer addresses into link layer addresses, a critical function in multiple-access networks. ARP is used to convert an IP address to a physical address such as an Ethernet address**. ARP has been implemented with many combinations of network and data link layer technologies.

ARP is a request and reply protocol that runs encapsulated by the line protocol. It is communicated within the boundaries of a single network, never routed across internetwork nodes. This property places ARP into the Link Layer of the Internet Protocol Suite, while in the Open Systems Interconnection

**In Internet Protocol Version 6 (IPv6) networks, the functionality of ARP is provided by the Neighbor Discovery Protocol (NDP).



Rysunek 2.11: ARP packet structure

(OSI) model, it is often described as residing between Layers 2 and 3, being encapsulated by Layer 2 protocols.

2.7.2 Packet structure

The Address Resolution Protocol uses a simple message format that contains one address resolution request or response. The size of the ARP message depends on the upper layer and lower layer address sizes, which are given by the type of networking protocol (usually IPv4) in use and the type of hardware or virtual link layer that the upper layer protocol is running on. The message header specifies these types, as well as the size of addresses of each. The message header is completed with the operation code for request (1) and reply (2). The payload of the packet consists of four addresses, the hardware and protocol address of the sender and receiver hosts.

The principal packet structure of ARP packets is shown in the figure 2.11. Following fields means

HTYPE (ang. *Hardware Type*, 16 bits) This field specifies the network protocol type. For Ethernet this has the value 1.

PTYPE (ang. *Protocol type*, 16 bits) This field specifies the internetwork protocol for which the ARP request is intended. For IPv4, this has the value 0x0800.

HLEN (ang. *Hardware Address Length*, 8 bits) Length (in octets) of a hardware address. Ethernet addresses size is 6.

```
jak rys. 9.28 s. 440 ale chyba ważniejszy od samego
rysunku jest tekst opisujący co się na nim dzieje
```

Rysunek 2.12: Działanie ARP w najprostszym przypadku (w tej samej podsieci)

```
jak rys. 9.31 s. 444
```

Rysunek 2.13: Działanie ARP w dla dwóch podsieci

PLEN (ang. *Protocol Address Length*, 8 bits) Length (in octets) of addresses used in the upper layer protocol. (The upper layer protocol specified in PTYPE.) IPv4 address size is 4.

OPER (ang. *Operation*, 16 bits) Specifies the operation that the sender is performing: 1 for request, 2 for reply.

SHA (ang. *Sender Hardware Address*, x bits) Media address of the sender.

SPA (ang. *Sender Protocol Address*, y bits) Internetwork address of the sender.

THA (ang. *Target Hardware Address*, x bits) Media address of the intended receiver. This field is ignored in requests.

TPA (ang. *Target Protocol Address*, y bits) Internetwork address of the intended receiver.

Przykład 2.2 (Przykładowy komunikat ARP)**.** `Do zrobienia na zajęciach`

Działanie protokołu ARP ilustrują rysunki 2.12 oraz 2.13. DOPISAC OPIS poszczegolnych krokow Proces ten przebiega w następujących krokach

1. DOPISAC!!!

2.8 Routig – an introduction

Podrozdział na temat routingu w rozdziale poświęconym protokołowi IP nie jest przypadkiem. Wszak większość komputerów wymienia dane z komputerami, które nie są podłączone od tego samego przełącznika. Przełączniki ograniczają zasięg działania do lokalnego obszaru. Dopiero routery, dzięki swojej konstrukcji umożliwiają połączenie różnych typów sieci fizycznych i zmianę punktu widzenia z lokalnego (sieci LAN) na globalny (WAN). Istotne jednak jest to, że bez względu na różnice pomiędzy sieciami, wciąż posługujemy się pakietami IP i adresami IP.

jak rys. 9.3 s. 401

Rysunek 2.14: Idea routingu

jak rys. 10.1 s. 455

Rysunek 2.15: Routing w prostej międzysieci – wersja uproszczona

jak rys. 10.2 s. 458

Rysunek 2.16: Routing z punktu widzenia protokołu IP

Protokół IP definiuje proces określany routingiem IP lub krócej routingiem. Routing określa, jak należy przekazywać dane w formie pakietów IP od jednego hosta do drugiego. Proces routingu umożliwia wysyłanie pakietów przez różnorodne sieci fizyczne (patrz rysunek 2.14).

Routery odbierają pakiety IP pochodzące od hosta z jednej sieci i wysyłają je, być może za pośrednictwem zupełnie innej sieci, aby we współpracy z innymi routerami dostarczyć pakiety do zadanego miejsca docelowego. Tak więc routery muszą przekazywać pakiety z jednej sieci fizycznej do drugiej, wspólnie przekazując dany pakiet od hosta źródłowego do hosta docelowego. Proces ten nazywany jest routingiem lub przekazywaniem. Aby możliwa stała się jego realizacja, router sprawdza nadchodzące pakiety i posługując się odczytanym docelowym adresem IP i podejmuje decyzję, przez który interfejs należy dany pakiet wysłać. Podjęcie decyzji staje się możliwe dzięki znajomości tras zapisanych w tablicach routingu poszczególnych routerów. Routery używają protokołów routingu do dynamicznej nauki wymaganych tras.

Najpierw omówimy sam proces routingu, zakładając, iż routery posiadają już wszystkie niezbędne do tego informacje. W dalszej części natomiast (podrozdział 2.8.2) podamy przykład najprostszego protokołu routingu. Bardziej szczegółowe informacje podane zostaną w rozdziale **??**.

2.8.1 Routing – study case

Routing w prostej międzysieci

Rysunek 2.15.

Routing z punktu widzenia protokołu IP

Rysunek 2.16.

jak rys. 10.3 s. 459

Rysunek 2.17: Routing z punktu widzenia warstwy dostępu do sieci

jak rys. 10.5 s. 461

Rysunek 2.18: Routing z punktu widzenia hosta

jak rys. 10.6 s. 464

Rysunek 2.19: Routing w prostej międzysieci – szczegóły procesu dla sieci z rysunku 2.15

Routing z punktu widzenia warstwy dostępu do sieci

Rysunek 2.17.

Routing z punktu widzenia hosta

Rysunek 2.18.

Podsumowując, proces routingu przebiega według następujących kroków.

1. DOPISAC na wzor krokow ze strony 462

Szczegóły procesu routingu dla prostej międzysieci

Powracamy do międzysieci z podrozdziału 2.8.1 aby zaprezentować więcej szczegółów tego procesu. Rysunek 2.19.

Wykorzystanie pamięci ARP w procesie routingu

Rysunek 2.20. Rysunek 2.21.

2.8.2 Routing protocols

A routing protocol specifies how routers communicate with each other, propagating information that enables them to select routes (routers on a path) between any two nodes on a computer network.

jak rys. 10.8 s. 467

Rysunek 2.20: Wykorzystanie pamięci ARP w procesie routingu

jak rys. 10.9 s. 468

Rysunek 2.21: Wykorzystanie pamięci ARP w procesie routingu – przykład drugi

Routing algorithms determine the specific choice of route. Each router has a priori knowledge only of networks attached to it directly. A routing protocol shares this information first among immediate neighbors, and then throughout the network. This way, routers gain knowledge of the topology of the network.

Although there are many types of routing protocols, three major classes are in widespread use on IP networks:

- Interior gateway routing via distance vector routing protocols, such as RIPv2, IGRP and EIGRP;
- Interior gateway routing via link state routing protocols, such as OSPF and IS-IS;
- Exterior gateway routing. DOPISAC!!! WYJASNIC!!!

Criteria used to compare routing protocols includes

- Time to convergence — how quickly all routers share consistent information.
- Scalability — can the network continue to grow?
- Resource usage — memory, CPU, bandwidth.
- Implementation and maintenance — intelligence level required of network admin.

2.8.3 Distance vector routing protocols

Description

Distance-vector routing protocols is a class of routing algorithms, w której router zna jedynie odległość wszystkich swoich sąsiadów do każdego węzła docelowego w sieci (przy czym pojęcie odległość może być zdefiniowane w różny sposób, niekoniecznie jako fizyczna odległość do pokonania - często jest to po prostu liczba węzłów pośrednich). The term distance vector refers to the fact that the protocol manipulates vectors (arrays) of distances to other nodes in the network. Bazując na informacjach o swoich sąsiadach router może wyznaczyć drogę, która jest najkrótsza do każdego celu i przez którego z sąsiadów przebiega. Router nie ma jednak pełnej informacji o kolejnych routerach. W związku z tym, może się okazać, że droga najkrótsza nie będzie najlepsza. A distance-vector routing

protocol requires that a router informs its neighbours of topology changes periodically. Compared to link-state protocols, which require a router to inform all the nodes in a network of topology changes, distance-vector routing protocols have less computational complexity and message overhead. Algorytmy trasowania wektora odległości są podatne na pętle trasowania, ale też są łatwiejsze do realizacji niż algorytmy trasowania stanu łącza.

Consider simple example to explain how it works. The starting assumption for distance-vector routing is that each node knows the cost of the link to each of its directly connected neighbors. Each node constructs a one-dimensional array containing the costs (in our case simply distances) to all other nodes and distributes that vector to its immediate neighbors.

We can represent each node's knowledge about the distances to all other nodes as a table like the one given in **??**. Note that each node only knows the information in one row of the table. Rules for distance vector algorithm are as follow

1. Every node sends a message to its directly connected neighbors containing its personal list of distance. (for example, A sends its information to its neighbors B,C,E, and F.)

2. If any of the recipients of the information from A find that A is advertising a path shorter than the one they currently know about, they update their list to give the new path length and note that they should send packets for that destination through A. (node B learns from A that node E can be reached at a cost of 1; B also knows it can reach A at a cost of 1, so it adds these to get the cost of reaching E by means of A. B records that it can reach E at a cost of 2 by going through A.)

3. After every node has exchanged a few updates with its directly connected neighbors, all nodes will know the least-cost path to all the other nodes.

4. In addition to updating their list of distances when they receive updates, the nodes need to keep track of which node told them about the path that they used to calculate the cost, so that they can create their forwarding table. (for example, B knows that it was A who said „I can reach E in one hop" and so B puts an entry in its table that says" To reach E, use the link to A.)

Taking as an inital state the table **??** and apply above rules we have DOPISAC PRZYKLAD DZIALANIA

ROZDZIAŁ 3

TCP

Protokół TCP ([?],[?],[?]) (ang. *Transmission Control Protocol*) jest protokołem warstwy 4 modelu ISO/OSI. Mówiąc najogólniej jego zadaniem jest zapewnienie niezawodnej transmisji strumienia bajtów (jakim są dane otrzymywane z warstw wyższych) na bazie (zawodnych z natury) usług warstw niższych.

TCP daje nam połączenie:

- zawsze dwupunktowe (brak np. obsługi multicastingu);
- pełnodupleksowe – jednocześnie przesyłane są dane w obu kierunkach;
- traktujace dane przesyłane w ramach połączenia TCP jak strumień bajtów.

Usługa TCP realizowana jest w oparciu o tak zwane gniazda (ang. *sockets*). Gniazdo tworzone jest zarówno przez nadawcę jak i odbiorcę. Jednoznacznie identyfikowane jest ono prze numer IP urządzenia i loklanego (przyznawanego w ramach urządzenia) numeru nazywanego portem (ang. *port*). Ważne aby pamiętać, że określone gniazdo może być wykorzystywane w tym samym czasie w kilku różnych połączeniach; identyfikację połączenia stanowi para złożona z dwóch gniazd reprezentujących dwie strony kanału komunikacyjnego*.

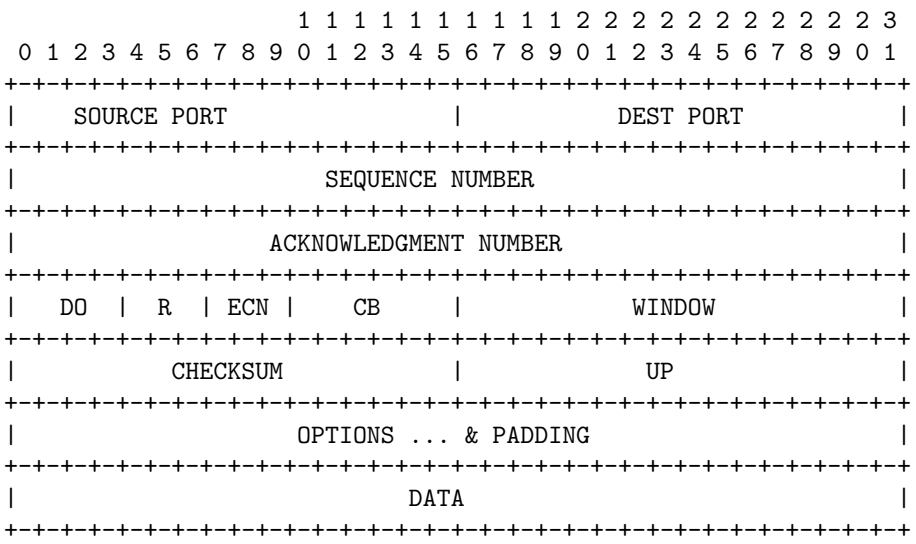
Porty o numerach mniejszych od 1024 określane są mianem dobrze znanych portów (ang. *well-known ports*) i ich wykorzystanie zarezerwowane jest na potrzeby usług standardowych†. Niektóre z częściej używanych portów przedstawia tabela 3.1.

*Celowo unikam w tym miejscu użycia terminów nadawca i odbiorac, gdyż utworzony kanał komunikacyjny nie wskazuje jednoznacznie żadngo z nich.

†Oczywiście nikt nie zabroni nam napisania programu wykorzystującego jeden z portów poniżej 1024 tylko, że wówczas może to rodzić spore komplikacje.

Wpisy w nagłówku IP (adres nadawcy i odbiorcy)	Operacja

Tabela 3.1: Niktóre z częściej używanych portów



Rysunek 3.1: Nagłówek protokołu TCP

3.1 Nagłówek TCP

A TCP segment[‡] consists of a segment header and a data section. Nagłówek TCP składa się z części stałej (20 bajtowej) oraz części opcjonalnej o zmiennej długości. Format nagłówka przedstawiono na rysunku 3.1.

SOURCE PORT (16 bitów) Port źródłowy (numery portów przypisanych do różnych usług – patrz dalej).

DEST PORT (16 bitów) Port docelowy.

SEQUENCE NUMBER (32 bity) Numer sekwencyjny pierwszego oktetu danych. Jeśli obecny jest znacznik SYN to numer sekwencyjny jest początkowym numerem sekwenycjnym a pierwszy oktet ma numer o jeden większy od tego numeru.

ACKNOWLEDGMENT NUMBER (32 bity) Jeśli bit ACK jest ustawiony (ma wartość 1) pole to zawiera liczbę określającą kolejny numer danych oczekiwanych przez odbiorcę.

[‡]The term TCP packet, though sometimes informally used, is not in line with current terminology, where segment refers to the TCP Protocol Data Unit (PDU), datagram to the IP PDU and frame to the data link layer PDU.

DO (*ang. Data Offset*, 4 bity) Określa długość nagłówka TCP w 32 bitowych słowach.

R (*ang. Reserved*, 3 bity) Zarezerwowane. Wartość tego pola powinna wynosić 0.

ECN (*ang. Explicit Congestion Notifiction*, 3 bity)

```
 0.1.2
|N|C|E|
```

N (*ang. Nonce Sum*, 1 bit)

C (*ang. CWR*, 1 bit)

E (*ang. ECE, ECN-Echo*, 1 bit)

CB (*ang. Control Bits*, 6 bitów)

```
 0.1.2.3.4.5
|U|A|P|R|S|F|
```

U, URG (*ang. Urgent*, 1 bit) – pole wskaźnika do pilnych danych ma znaczenie;

A, ACK (*ang. Acknowledgment*, 1 bit) – pole potwierdzenia ma znaczenie;

P, PSH (*ang. Push*, 1 bit) – wymuszenie transmisji danych;

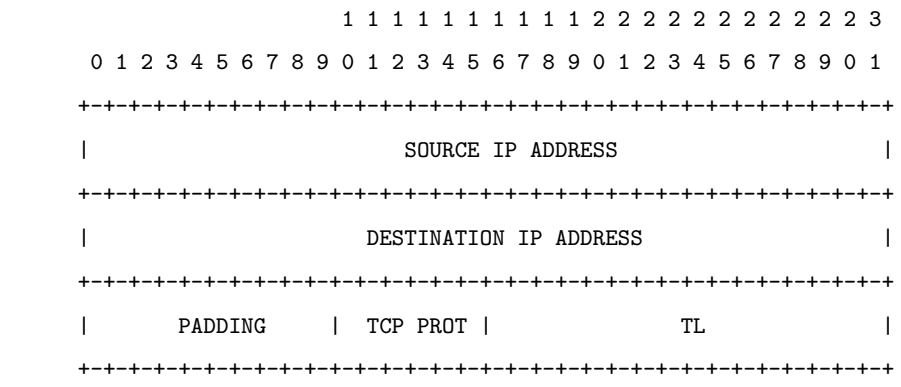
R, RST (*ang. Reset*, 1 bit) – zerowanie połączenia;

S, SYN (*ang. Synchronize*, 1 bit) – synchronizacja numerów sekwencyjnych;

F, FIN (1 bit) – koniec danych od nadawcy;

WINDOW (16 bitów) Ilość danych jaką może przyjąć odbiorca. W RFC 1323 określono tzw. okno skalowalne, czyli możliwość interpretowania zawartości tego pola w jednostkach większych niż bajt. Jako maksymalną jednostkę przyjęto 2^{14} bajtów co w połączeniu z samą wielkością pola, pozwala definiować pola o rozmiarze 1GB ($2^30 = 1.073.741.824$).

CHECKSUM (16 bitów) Suma kontrola nagłówka TCP (wraz z danymi) uzupełnionego o pseudonagłowek zawierający informacje z nagłówka IP oraz TCP. Format tego pseudonagłówka jest następujący



gdzie PADDING powinno być wypełnione zerami, TCP PROT jest numerem wersji protokołu TCP, natomiast TL jest rozmiarem pakietu TCP.

Najpierw wpisujemy w pole sumy kontrolnej wartość zero. Następne dodajemy do siebie wszystkie 16-bitowe słowa tworzące pseudonagłówek i nagłówek właściwy TCP (wraz z danymi). Przy dodawaniau wykorzystujemy tylko 16 młodszych bitów wyniku (zatem ignorujemy przeniesienia na pozycje dalsze niż na 16 bit). Sumę kontrolną stanowi zanegowany wynik tych dodawań.

UP (*ang. Urgent Pointer*, 16 bitów) Jeśli flaga URG jest ustawiona, pole to wskazuje na kolejność ważnych danych.

OPTIONS (różna długość; od 0 do 44 bajtów)

PADDING Używane do wypełnienia pustego miejsca aby zapewnić, że dane zaczną się na granicy 32 bitowego słowa.

DATA Dane

Oto fragment listy dostępnej pod adresem
`http://www.iana.org/assignments/port-numbers`
opisującej porty przypisane do różnych usług:

```
PORT NUMBERS

(last updated 17 November 2005)

The port numbers are divided into three ranges: the Well Known Ports,
the Registered Ports, and the Dynamic and/or Private Ports.

The Well Known Ports are those from 0 through 1023.
```

The Registered Ports are those from 1024 through 49151

The Dynamic and/or Private Ports are those from 49152 through 65535

```
**************************************************************************
* PLEASE NOTE THE FOLLOWING:                                             *
*                                                                        *
* 1. UNASSIGNED PORT NUMBERS SHOULD NOT BE USED.  THE IANA WILL ASSIGN *
* THE NUMBER FOR THE PORT AFTER YOUR APPLICATION HAS BEEN APPROVED.    *
*                                                                        *
* 2. ASSIGNMENT OF A PORT NUMBER DOES NOT IN ANY WAY IMPLY AN          *
* ENDORSEMENT OF AN APPLICATION OR PRODUCT, AND THE FACT THAT NETWORK  *
* TRAFFIC IS FLOWING TO OR FROM A REGISTERED PORT DOES NOT MEAN THAT   *
* IT IS "GOOD" TRAFFIC. FIREWALL AND SYSTEM ADMINISTRATORS SHOULD      *
* CHOOSE HOW TO CONFIGURE THEIR SYSTEMS BASED ON THEIR KNOWLEDGE OF    *
* THE TRAFFIC IN QUESTION, NOT WHETHER THERE IS A PORT NUMBER          *
* REGISTERED OR NOT.                                                     *
**************************************************************************
```

WELL KNOWN PORT NUMBERS

The Well Known Ports are assigned by the IANA and on most systems can only be used by system (or root) processes or by programs executed by privileged users.

Ports are used in the TCP [RFC793] to name the ends of logical connections which carry long term conversations. For the purpose of providing services to unknown callers, a service contact port is defined. This list specifies the port used by the server process as its contact port. The contact port is sometimes called the "well-known port".

To the extent possible, these same port assignments are used with the UDP [RFC768].

The range for assigned ports managed by the IANA is 0-1023.

```
Port Assignments:

Keyword         Decimal    Description                      References
-------         -------    -----------                      ----------
echo              7/tcp    Echo
echo              7/udp    Echo
daytime          13/tcp    Daytime (RFC 867)
daytime          13/udp    Daytime (RFC 867)
qotd             17/tcp    Quote of the Day (RFC 865)
qotd             17/udp    Quote of the Day (RFC 865)
ftp-data         20/tcp    File Transfer [Default Data]
ftp-data         20/udp    File Transfer [Default Data]
ftp              21/tcp    File Transfer [Control]
ftp              21/udp    File Transfer [Control]
ssh              22/tcp    SSH Remote Login Protocol
ssh              22/udp    SSH Remote Login Protocol
telnet           23/tcp    Telnet
telnet           23/udp    Telnet
smtp             25/tcp    Simple Mail Transfer
smtp             25/udp    Simple Mail Transfer
time             37/tcp    Time (RFC 1305)
time             37/udp    Time (RFC 1305)
domain           53/tcp    Domain Name Server
domain           53/udp    Domain Name Server
http             80/tcp    World Wide Web HTTP
http             80/udp    World Wide Web HTTP
www              80/tcp    World Wide Web HTTP
www              80/udp    World Wide Web HTTP
www-http         80/tcp    World Wide Web HTTP
www-http         80/udp    World Wide Web HTTP


REGISTERED PORT NUMBERS

The Registered Ports are listed by the IANA and on most systems can be
used by ordinary user processes or programs executed by ordinary
users.
```

```
Ports are used in the TCP [RFC793] to name the ends of logical
connections which carry long term conversations.  For the purpose of
providing services to unknown callers, a service contact port is
defined.  This list specifies the port used by the server process as
its contact port.

The IANA registers uses of these ports as a convenience to the
community.

To the extent possible, these same port assignments are used with the
UDP [RFC768].

The Registered Ports are in the range 1024-49151.

Port Assignments:

Keyword         Decimal    Description                    References
-------         -------    -----------                    ----------
interwise       7778/tcp    Interwise
interwise       7778/udp    Interwise
quake           26000/tcp  quake
quake           26000/udp  quake

DYNAMIC AND/OR PRIVATE PORTS

The Dynamic and/or Private Ports are those from 49152 through 65535
```

3.2 Connection establishment

3.2.1 Problems

Tanenbaum, 6.2.2, p. 428

3.2.2 Solution: three-way handshake

To establish a connection, TCP uses a three-way handshake. Before a client attempts to connect with a server, the server must first bind to and listen at a port to open it up for connections: this is called a passive open. Once the passive open is established, a client may initiate an active open. To

establish a connection, the three-way (or 3-step) handshake occurs:

SYN : The active open is performed by the client sending a SYN to the server. The client sets the segment's sequence number to a random value A.

SYN-ACK : In response, the server replies with a SYN-ACK. The acknowledgment number is set to one more than the received sequence number i.e. A+1, and the sequence number that the server chooses for the packet is another random number, B.

ACK : Finally, the client sends an ACK back to the server. The sequence number is set to the received acknowledgement value i.e. A+1, and the acknowledgement number is set to one more than the received sequence number i.e. B+1.

At this point, both the client and server have received an acknowledgment of the connection. The steps 1, 2 establish the connection parameter (sequence number) for one direction and it is acknowledged. The steps 2, 3 establish the connection parameter (sequence number) for the other direction and it is acknowledged. With these, a full-duplex communication is established.

3.2.3 Scenarios

3.3 Connection termination

3.3.1 Problems: Byzantine fault tolerance

In computing, the Two Generals' Problem[§] is a thought experiment meant to illustrate the pitfalls and design challenges of attempting to coordinate an action by communicating over an unreliable link.

Definition of the problem

Two armies, each led by a general, are preparing to attack a fortified city. The armies are encamped near the city, each on its own hill. A valley separates the two hills, and the only way for the two generals to communicate is by sending messengers through the valley. Unfortunately, the valley is occupied by the city's defenders and there's a chance that any given messenger sent through the valley will be captured (this scenario assumes that while the two generals have agreed that they will

[§]Related to the more general Byzantine Generals' Problem. Some authors also refer to the Two Generals' Problem as the Two Armies Problem or the Coordinated Attack Problem.

attack, they haven't agreed upon a time for attack before taking up their positions on their respective hills).

The two generals must have their armies attack the city at the same time in order to succeed. They must thus communicate with each other to decide on a time to attack and to agree to attack at that time, and each general must know that the other general knows that they have agreed to the attack plan. Because acknowledgement of message receipt can be lost as easily as the original message, a potentially infinite series of messages are required to come to consensus.

The thought experiment involves considering how they might go about coming to consensus. In its simplest form one general (referred to as the "first general" below) is known to be the leader, decides on the time of attack, and must communicate this time to the other general. The requirement that causes the "problem" is that both generals must attack at the agreed-upon time to succeed. Having a solitary general attack is considered a disastrous failure. The problem is to come up with algorithms that the generals can use, including sending messages and processing received messages, that can allow them to correctly conclude: *Yes, we will both attack at the agreed-upon time.* Allowing that it is quite simple for the generals to come to an agreement on the time to attack (i.e. one successful message with a successful acknowledgement), the subtlety of the Two Generals' Problem is in the impossibility of designing algorithms for the generals to use to safely agree to the above statement.

Illustrating the problem

The first general may start by sending a message "Let us attack at 0900 on August 4." However, once dispatched, the first general has no idea whether or not the messenger got through. Any amount of uncertainty may lead the first general to hesitate to attack, since if the second general does not also attack at that time, the city's garrison will repel the advance, leading to the destruction of that attacking general's forces.

Knowing this, the second general may send a confirmation back to the first: "I received your message and will attack at 0900 on August 4." However, if the confirmation messenger were captured, the second general (knowing that the first will hesitate without the confirmation) may himself hesitate. A solution might seem to be to have the first general send a second confirmation: "I received your confirmation of the planned attack at 0900 on August 4." However, if that messenger were captured, it quickly becomes evident that no matter how many rounds of confirmation are made, there is no way to guarantee the second requirement that both generals agree the message was delivered and that the enemy did not alter any of the messages. Proof[edit]

3.3.2 Acceptable solution: three (four)-way handshake

The connection termination phase uses a four-way handshake, with each side of the connection terminating independently. When an endpoint wishes to stop its half of the connection, it transmits a FIN packet, which the other end acknowledges with an ACK. Therefore, a typical tear-down requires a pair of FIN and ACK segments from each TCP endpoint. After both FIN/ACK exchanges are concluded, the side which sent the first FIN before receiving one waits for a timeout before finally closing the connection, during which time the local port is unavailable for new connections; this prevents confusion due to delayed packets being delivered during subsequent connections. A connection can be "half-open", in which case one side has terminated its end, but the other has not. The side that has terminated can no longer send any data into the connection, but the other side can. The terminating side should continue reading the data until the other side terminates as well. It is also possible to terminate the connection by a 3-way handshake, when host A sends a FIN and host B replies with a FIN and ACK (merely combines 2 steps into one) and host A replies with an ACK.[13] This is perhaps the most common method.

3.3.3 Scenarios

3.4 Flow control

CCNA, p. 523 Tanenbaum, img. 6.26 (p. 475)

3.5 Retransmission of lost packets

CCNA, p. 527

3.6 Ordered data transfer

CCNA, p. 529

Bibliografia

Spis rysunków

Spis tabel