Python w Elektronicznej Sieci #8: Sieci komputerowe – podstawy

Projekt „Matematyka dla Ciekawych Świata”,
Robert Ryszard Paciorek
<rrp@opcode.eu.org>

2020-04-21

1 Podstawy TCP/IP

Sieci komputerowe działają na zasadzie przesyłania informacji w postaci porcji, z których każda posiada co najmniej informację o adresie odbiorcy (zwykle też nadawcy), nazywanych ramkami lub pakietami. Kierowanie pakietów w odpowiednie miejsce odbywa się na podstawie adresu pakietu i nie jest związane z fizycznym zestawianiem łącza pomiędzy nadawcą a odbiorcą - każdy pakiet jest kierowany niezależnie, a w ramach pojedynczego łącza (kanału transmisji) mogą być przekazywane pakiety adresowane do różnych odbiorców. Nazywane jest to komutacją pakietów, w odróżnieniu od komutacji łącza (która występowała np. w klasycznej, analogowej telefonii, gdzie przekaźniki w centralach dokonywały zestawienia połączeń elektrycznych między dwoma aparatami telefonicznymi).

1.1 Struktura warstwowa

Komunikacja sieciowa typowo posiada strukturę warstwową. W modelu OSI wyróżnia się 7 warstw:

1. fizyczną (pierwszą) definiującą aspekty związane z fizycznym przesyłem sygnału takie jak częstotliwości radiowe, poziomy napięć, etc.; określa sposób transmisji kolejnych bajtów
2. łącza danych (drugą) definiującą aspekty związane z formatem ramki, protokoły ustalania zasad dostępu do medium transmisyjnego, itd.; określa sposób transmisji porcji danych pomiędzy hostami w jednej sieci
3. sieciową (trzecią) definiującą aspekty związane z formatem pakietu, adresacją i zasady routingu umożliwiające zapewnienie łączności pomiędzy różnymi sieciami; określa sposoby transmisji porcji danych pomiędzy sieciami
4. transportową (czwartą) odpowiedzialną za podział strumienia na porcje informacji, kontrolę nad poprawnością transmisji, adresację usług w ramach hosta
5. sesji (piątą)
6. prezentacji (szóstą)
7. aplikacji (siódmą)

W modelu TCP/IP wyróżnia się 4 warstwy:

1. Dostępu do sieci - obejmującą warstwy 1 i 2 modelu OSI
2. Internetu - obejmującą warstwę 3 modelu OSI
3. Transportową - obejmującą warstwę 4 modelu OSI
4. Aplikacji - obejmującą warstwy 5, 6 i 7 modelu OSI

Z punktu widzenia modelu TCP/IP można powiedzieć o enkapsulacji danych kolejnych warstw w ramach warstwy niższej, czyli ”surowe” dane (np. strona HTML) obudowywane są strukturą opisywaną przez warstwę aplikacji (np. nagłówkami HTTP), następnie całość ta umieszczana jest w polu danych pakietu warstwy transportowej (np. TCP), ten z kolei w polu danych pakietu IP (warstwy sieciowej), na koniec pakiet IP jest umieszczany w polu danych ramki warstwy dostępu do sieci (np. ramki ethernetowej). W ramach podróży przez kolejne sieci pakiet IP jest wyjmowany i wkładany w kolejne ramki warstwy

dostępu do sieci, na ogół tylko z niewielkimi ingerencjami w zawartość tego pakietu (prawie zawsze nie dochodzącymi do pola danych pakietu TCP lub datagramu UDP, czyli nie wykraczającymi poza warstwę 4 OSI).

1.2 Protokół IP

Protokół IP (Internet Protocol) odpowiedzialny jest przede wszystkim za sposób adresacji hostów oraz reguły komutacji pakietów (routing). Jest on wspomagany przez kolejny protokół z tej rodziny - ICMP (Internet Control Message Protocol), którego zadaniem jest przekazywanie informacji kontrolnych np. o nieosiągalności hosta docelowego, odrzuceniu przetwarzania pakietu ze względu na zbyt dużą liczbę skoków (gdy wartość pola TTL z nagłówka IP wyniesie zero) a także pingi (zarówno żądanie jak i odpowiedź).

1.3 Adresacja IP

Adresy hostów (nazywane adresami IP) są to 32-bitowe (w IPv4) lub 128-bitowe (w IPv6) liczby. Adresy IPv4 zapisywane są najczęściej w notacji kropkowo-dziesiętnej, gdzie każdy bajt (ciąg 8 bitów) zapisywany jest jako liczba dziesiętna rozdzielana kropką od pozostałych. Adresy IPv6 zapisywane są zazwyczaj w notacji dwukropokowej, polegającej na zapisywaniu 16 bitowych części adresu liczbami szesnastkowymi oddzielanymi dwukropkiem, dodatkowo jeden ciąg zer (o długości będącej wielokrotnością 16 bitów) może być skompresowany (pominięty) co daje w zapisie dwa dwukropki `::`.

1.3.1 Długość prefixu i maska

Adresy hostów grupuje się w adresy sieci, bazując na jednakowym (bitowo) początku takiego adresu (zwanym adresem sieci lub prefixem). Ilość bitów stanowiących adres sieci w danym adresie IP nazywana jest długością prefixu i zapisywana jest zazwyczaj po ukośniku. Np. zapis `2001:db8::a17/48` oznacza że pierwsze 48 bity stanowią adres sieci a kolejne $128 - 48 = 80$ bitów stanowi adres hosta w tej sieci.

Długość prefixu jednoznacznie określa maskę danej podsieci, czyli liczbę odpowiadającą długości adresu (32 bity lub 128 bitów), złożoną z ciągu jedynek o długości prefixu oraz ciągu zer (o długości adresu hosta). W przypadku IPv4 spotykane jest także podawanie maski sieci w notacji kropkowo-dziesiętnej zamiast długości prefixu.

Sieć może zostać podzielona na mniejsze sieci (z większą wartością prefixu), jak też grupa sieci może zostać zagregowana w jedną większą (2^n raza) sieć (z prefixem mniejszym o n). Agregacja hostów i sieci w większe całości jest wykorzystywana w mechanizmach routingu, co pozwala na redukcję wielkości tablic routingowych.

1.3.2 Przynależność do sieci

Adres sieci zapisuje się typowo z wyzerowanymi bitami stanowiącymi adres hosta (czyli po dokonaniu bitowego *and* z maską danej sieci) oraz podaną informacją o długości prefixu, dla powyższego przykładu będzie to `2001:db8::/48`. Informacja taka jest wystarczająca do sprawdzenia czy dowolny inny adres IP należy do tej sieci czy nie.

```
from ipaddress import *

a1  = IPv6Address("2001:0db8::17:15")
aa1 = int(a1)
print("adress IPv6 jest 128 bitową liczbą całkowitą np.: " + str(a1) + " == " + hex(aa1))

n0  = IPv6Network("::/112");
m1  = n0.netmask
mm1 = int(m1)
p1  = n0.prefixlen

print("Maska podsieci IPv6 jest 128 bitową liczbą całkowitą np.: " + str(m1) + " == " + hex(mm1))
print("Jako że maska jest liczbą, która zapisana binarnie, zawsze zawiera ciągły ciąg bitów")
```

```
print("o wartości 1, a po nim ciągły ciąg bitów o wartości 0 (mogą być zerowej długości), to")
print("często stosowany jest zapis polegający na podawaniu długości prefiksu: /" + str(p1))
print("jest to ilość bitów o wartości 1 w masce, czyli im większy prefix tym mniejsza sieć.")

n1  = IPv6Network("2001:0db8::17:15/112", strict=False);
nn1 = int(n1.network_address)

print("Aby obliczyć adres sieci (czyli wspólną dla wszystkich hostów w danej sieci część")
print("adresu IP) należy wykonać binarny AND pomiędzy adresem IP hosta a maską podsieci.")
print("Dla powyższego przykładu:")
print(hex(mm1 & aa1) + " == " + str(n1) + " == " + hex(nn1))

# aby sprawdzić czy adres IP należy do danej sieci trzeba obliczyć adres sieci tego hosta
# w oparciu o maskę sieci którą sprawdzamy
def sprawdzSiec(n, a):
        nn = int(a) & int(n.netmask)
        if nn == int(n.network_address):
                print(str(a) + " należy do sieci " + str(n))
        else:
                print(str(a) + " NIE należy do sieci " + str(n))

sprawdzSiec(n1, IPv6Address("2001:0db8::17:ab13"))
sprawdzSiec(n1, IPv6Address("2001:0db8::13:a"))
```

1.4 Routing

Router kieruje każdy z pakietów do kolejnego routera lub bezpośrednio do hosta docelowego na podstawie jego adresu docelowego i tablicy routingu. Tablica taka zawiera adresy sieci wraz z adresami następnych routerów do nich prowadzących bądź wskazaniem lokalnego interfejsu sieciowego poprzez który powinny być osiągalne hosty z danej sieci. W tym celu korzysta z sprawdzania przynależności adresu do sieci, w celu ustalenia adresu następnego routera i/lub interfejsu sieciowego na który ma zostać przekazany pakiet.

Tablica przeglądana jest od wpisów najbardziej precyzyjnych, czyli z największym prefixem do wpisów najbardziej ogólnych (ostatnim wpisem jest na ogół trasa domyślna czyli sieć ::/0 dla IPv6 lub 0.0.0.0/0 dla IPv4). Dzięki czemu jeżeli kilka wpisów (sieci) z tablicy routingu pasuje do adresu docelowego z nagłówka pakietu, wybierany jest wpis najbardziej precyzyjny (o najdłuższym prefixie), a pasująca do każdego adresu trasa domyślna wybierana jest tylko gdy nie ma żadnej lepszej. Może się zdarzyć że kilka wpisów (nawet z tą samą maską) pasuje do adresu docelowego hosta, w takiej sytuacji do wyboru ścieżki używane są inne dane z tablicy routingu (takie jak metryka).

Tablice routingu mogą zawierać wpisy dodawane statycznie (wpisane do konfiguracji danego urządzenia), jak też wpisy dodawane dynamicznie w oparciu o protokołu wymiany informacji routingowych (protokoły routingu) takie jak: IGRP, OSPF, BGP. Protokoły routingu dynamicznego mogą być wykorzystywane m.in. do rozkładania obciążenia na różne łącza, zapewnienia redundancji łącz, blokowania ataków (D)DoS.

Także każdy z hostów ma tablice routingu, typowo składa się z dwóch pozycji – trasy do sieci lokalnej (tej sieci z której adres posiada dany host) wskazującej bezpośrednio na urządzenie sieciowe oraz trasy domyślnej wskazującej na router zapewniający dostęp do innych sieci, nazywany bramką (gateway). Jeżeli router nie posiada adresu w tej samej sieci co host konieczna jest dodatkowa trasa wskazująca poprzez jakie urządzenie dostępny jest router domyślny.

Oprócz opisanego powyżej routingu unicastowego (kierowania do jednego odbiorcy) realizowane są także transmisje:

- *anycast* – do dowolnego / najbliższego hosta o danym adresie; zasadniczo jest to transmisja unicast, tyle że adres docelowy nie jest unikalny w skali globalnej a różne routery kieruje te pakiety do różnych hostów docelowych (typowo wybierając najbliższy taki host)
- *multicast* – do grupy hostów, w tym wypadku (multicastowy) adres IP identyfikuje ”kanał nadawczy” a nie unikalny host docelowy

- *broadcast* – do wszystkich hostów (w ramach danej sieci – nie są routowne), transmisje rozgłoszeniowe można traktować jako szczególny przypadek transmisji multicastowych w których grupa multicastowa obejmuje wszystkie hosty (można je zastąpić takimi transmisjami multicastowymi)

2 Komunikacja TCP/IP

W oparciu o protokół IP działają protokoły warstwy transportowej takie jak UDP, TCP, czy też (mniej znane protokoły czasu rzeczywistego, transmisji strumieniowych): RTP, RTCP i SCTP. Najprostszym protokołem warstwy transmisji wydaje się być UDP, protokół ten umożliwia przesłanie informacji pomiędzy dwoma hostami IP i nie kontroluje on tego czy została ona przesłana poprawnie. Natomiast TCP kontroluje to czy przesłana informacja dotarła do adresata i nie została uszkodzona, a w przypadku problemów informacja wysyłana jest ponownie. TCP w związku z tym w przeciwieństwie do UDP musi otworzyć połączenie i wykorzystywać je do kontroli poprawności przesłania informacji, wymaga zatem przesłania większej liczby pakietów (co może prowadzić do pewnych opóźnień itp). W związku z tym TCP używany jest tam gdzie konieczna jest kontrola poprawności transmisji (oraz ponowne wysłanie zgubionego pakietu), UDP tam gdzie nie jest to potrzebne (a liczy się czas).

Dodatkowo zarówno UDP jak i TCP na każdym z hostów wyróżniają numeryczny identyfikator dla aplikacji/procesu/usługi będącego odbiorcą czy też nadawcą informacji zwany numerem portu.

2.1 Polularne usługi

W ramach sieci mogą być realizowane różne usługi w oparciu o różne protokoły warstwy aplikacyjnej. Standardowe usługi posiadają zdefiniowane domyślne adresy portów dla swoich protokołów. Wśród usług i protokołów sieciowych należy wymienić przynajmniej:

- DNS (Domain Name System) - odpowiedzialny za system mapujący nazwy alfanumeryczne hostów na adresy IP. Domeny posiadają budowę hierarchiczną / drzewiastą (precyzja rośnie od prawej do lewej, a kolejne poziomy oddzielane są kropkami). Realizacja odpowiedzi na zapytanie DNS wygląda następująco:
  1. host kieruje zapytanie do określonego w jego konfiguracji serwera "rozwijającego" DNS (DNS resolver),
  2. serwer taki sprawdza w swojej pamięci podręcznej czy zna odpowiedź na to zapytanie (i nie jest ona przeterminowana - nie upłynął czas TTL od odnalezienia), jeżeli nie ma jej w swojej pamięci to
  3. serwer taki zna adresy głównych serwerów DNS (root serwerów) zawierających informacje na temat serwerów obsługujących domeny najwyższego rzędu i kieruje do jednego z nich zapytanie o serwer obsługujący skrajnie prawą część adresu (np. *.org*),
  4. do otrzymanego serwera kierowane jest zapytanie o większą część adresu (np. *eu.org*),
  5. itd. aż do uzyskania odpowiedzi o pytany adres
- mechanizmy auto konfiguracji hostów - DHCP, rozgłaszanie informacji routingowej poprzez ICMPv6 (protokół warstwy 3)
- WWW - udostępnianie treści z użyciem protokołu HTTP
- pocztę elektroniczną - przesyłanie wiadomości (protokoły SMTP, IMAP, POP)
- komunikację natychmiastową i telefonię IP (protokoły SIP, XMPP, IAX)
- SSH - zdalny, szyfrowany dostęp do systemów IT, przesył plików oraz tunelowanie innych usług

3 Diagnostyka sieci

Istnieje wiele poleceń służących do diagnozowania ewentualnych problemów sieciowych lub mogąccyh być w tym przydatnymi. Poniżej znajduje się zestawienie najbardziej popularnych / użytecznych narzędzi z

podziałem wg zastosowań.

3.1 Adresy

- `ipcalc` oraz `sipcalc` – kalkulator IP (pozwalający na obliczanie adresów sieci rozgłoszeniowych, zmianę notacji itd)
- `whois` – informacje z bazy whois (o domenie lub adresie IP)

3.2 Dostępność i trasy do hostów

- `ping [opcje] host` lub `ping6 [opcje] host` – sprawdzanie dostępności hosta z użyciem protokołu ICMP (obecnie komenda `ping6` najczęściej jest równoważna poleceniu ping z opcją `-6` wymuszającą używanie jedynie IPv6, na starszych systemach komenda ping może nie obsługiwać adresów IPv6 i wtedy konieczne jest stosowanie do nich polecenia `ping6`), ważniejsze opcje:
  `-c n` wykonaj n zapytań (domyślnie pyta do momentu przerwania przy pomocy np. Ctrl-C, lub sygnału wysłanego z uzyciem komendy `kill`)
  `-n` nie zamieniaj adresu IP hosta który odpowiedział na nazwę domenową
- `traceroute`, `traceroute6`, `tracepath`, `tracepath6`, `tcptraceroute` lub `tcptraceroute6` – sprawdzanie ścieżki do hosta (wypisanie listy routerów przez które przechodzi pakiet w drodze do wskazanego hosta)
  Istnieją różne warianty tych poleceń (nawet pod tą samą nazwą), różnią się one stosowanymi mechanizmami i domyślnymi opcjami. Generalnie wszystkie uruchamia się na zasadzie `polecenie [opcje] host`. Warianty z `6` na końcu nazwy będą używały jedynie adresów IPv6, natomiast polecenia bez `6` na końcu nazwy mogą potrafić ich używać lub nie. Wszystkie popularne warianty pozwalają na podanie opcji `-n` wyłączającej zamienianie adresu IP hosta który odpowiedział na jego nazwę domenową.
  Może zdarzyć się że śledzenie urwie się na jakimś hoście (np. z powodu jego konfiguracji lub błędów w jego oprogramowaniu sieciowym), może się zdarzyć że przy użyciu innej komendy z tej grupy (lub zmianie opcji) uda się prześledzić dalszą trasę pakietu.
- `mtr [opcje] host` – sprawdzanie ścieżki do hosta (czyli podobnie jak traceroute i tracepath) w trybie ciągłym (z ciągłym odświeżaniem) wraz z wypisywaniem informacji o stratach pakietów i opóźnieniach na poszczególnych odcinkach, ważniejsze opcje:
  `-n` nie zamieniaj adresu IP hosta który odpowiedział na nazwę domenową
- `nmap` – skaner sieciowy - sprawdzanie dostępnych hostów w sieci, otwartych portów, itd
- `arping` – narzędzie do pingowania z wykorzystaniem zapytań ARP zamiast ICMP
  istnieją dwie zasadnicze odmiany: z iputils oraz z synscan; ta druga zawarta w debianowym pakiecie "arping" umożliwia także pingowanie po adresie MAC (ale nie przez RARP, bo on nie do tego służy), aby to jednak działało host docelowy nie może ignorować pingów rozgłoszeniowych, metoda obejścia opisana jest w README arping'a
- `arp-scan` – wyszukiwanie hostów w oparciu o zapytania ARP (można powiedzieć że jest to równoważne uruchamianiu komendy arping w pętli)

3.3 DNS

- `dig [opcje] nazwa [typRekordu]` – narzędzia do uzyskania informacji z DNS, pozwala na określenie poprzez `@adres` serwera który chcemy odpytać oraz na określenie (poprzez drugi argument) typu rekordu który chcemy uzyskać np.:
  `A` - rekord typu A czyli mapowanie nazwy na adres IPv4
  `AAAA` - rekord typu AAAA czyli mapowanie nazwy na adres IPv4
  `MX` - rekord typu MX czyli informacja o serwerach obsługujących pocztę danej domeny
  `NS` - rekord typu NS czyli informacja o serwerach obsługujących DNS danej domeny
  `SRV` - rekord typu SRV czyli informacje o hoście świadczącym usługę (usługa określana jest w nazwie

domeny o którą pytamy)
`TXT` - rekord typu TXT czyli informacje dodatkowe
`ANY` - powoduje odpytanie o wszystkie rekordy
`AXFR` - powoduje wysłanie prośby o transfer całej strefy (działa jezeli dany host ma prawo transferu całej strefy z danego serwera)

- `host [opcje] nazwa|ip [server]` – narzędzia do zamiany adresów domenowych na IP i odwrotnie oraz wyciągania innych informacji z DNS (np. rekordy MX)
- `nslookup [opcje] nazwa|ip [server]` – narzędzia do zamiany adresów domenowych na IP i odwrotnie oraz wyciągania innych informacji z DNS (np. rekordy MX)
- `dnstracer` – śledzenie trasy zapytań DNS
- `dnswalk` – debuger DNS

3.4 IPv6

- `ndisc6` – testowanie ICMPv6 Neighbor Discovery
- `rdisc6` – testowanie ICMPv6 Router Discovery
- `rltraceroute6` – trasa pakietów do danego hosta IPv6 z użyciem UDP/ICMP
- `tcpspray6` – pomiar prędkości łącza z użyciem TCP/IP Discard/Echo
- `na6` / `ns6` – wysyłanie pakietów Neighbor Advertisement / Solicitation
- `ra6` / `rs6` – wysyłanie pakietów Router Advertisement / Solicitation
- `ni6` / `rd6` – wysyłanie pakietów ICMPv6 Node Information / Redirect
- `scan6` – skanowanie sieci IPv6

3.5 debugowanie łączności sieciowej

- `netcat` lub `nc` lub `netcat6` – program pozwalający na wysyłanie pakietów TCP i UDP z definiowaną przez nas zawartością, oraz odbiór pakietów TCP i UDP (słuchanie na wskazanym porcie), umożliwia m.in. testowanie usług sieciowych (takich jak smtp, www, jabber, ...); uwaga: występuje w kilku wersjach różniących się opcjami
- `telnet` – program umożliwiający zdalny (nieszyfrowany, łącznie z hasłem!) dostęp do powłoki, a także (podobnie jak netcat) m.in. testowanie usług sieciowych
- `swaks` – narzędzie do testowania SMTP
- `tcpdump` – przechwytuje komunikację sieciową celem analizy nagłówków lub pełnej zawartości pakietów (wsparcie dla niektórych z protokołów warst wyższych wymaga doinstalowania - np. obsługę DHCP zapewnia dhcpdump)
- `wireshark` lub `tshark` – przechwytuje komunikację sieciową celem analizy nagłówków lub pełnej zawartości pakietów, wspiera dekodowanie wielu protokołów warstwy aplikacyjnej, wireshark posiada graficzny interfejs użytkownika, tshark jest wersją konsolową

3.6 informacje o wykorzystaniu i prędkości sieci

- `netstat` – informacje o sieci
- `iptraf` – monitor IP LAN
- `nload` – graficzne (ncurses) pokazywanie wykorzystania (prędkości) interfejsów sieciowych
- `ttcp` – testuje prędkość połączenia sieciowego (strona domowa, najnowsza wersja oraz mutacja)
- `iperf` – pomiar prędkości połączenia sieciowego

4 Ethernet

Od strony sprzętowej sieć składa z:

- hostów stanowiących nadawców i odbiorców informacji
- urządzeń sieciowych pośredniczących w ich przekazywaniu, takich jak nadajniki, switche, mediakonwertery
- okablowania miedzianego bądź światłowodowego (jeżeli nie jest siecią bezprzewodową)

W przypadku sieci w standardzie Ethernet stosowane są 48 bitowe adresy MAC (pierwsza część identyfikuje producenta karty) oraz wspólny dla wszystkich odmian (przewodowych i bezprzewodowych) format ramki (określający położenie w ramce adresów, informacji dodatkowych oraz danych). Pakiety protokołu warstwy wyższej (np. pakiety IP wraz ich strukturą zawierającą adresy itd) z punktu widzenia ramki ethernetowej są danymi, w które ta warstwa nie wnika. Do mapowania adresów IP na adresy MAC wykorzystywany jest protokół ARP (dla IPv4) lub Neighbor Discovery (dla IPv6) - odbywa się to poprzez wysłanie ramki ethernetowej na adres rozgłoszeniowy (odbierany przez wszystkie hosty) z pytaniem o to jaki MAC adres ma host o podanym numerze IP.

Sieć ethernetowa typowo posiada strukturę wielokrotnej gwiazdy (drzewa), w węzłach której stosowane są switche. Kierują one ramki do odpowiednich gałęzi na podstawie adresu docelowego i wpisów w tablicy adresów MAC, utworzonej w oparciu o źródłowe nadawców przechodzących przez dany switch ramek. W przypadku gdy adresu docelowego nie ma w tablicy ramka kierowana jest na wszystkie porty switcha z wyjątkiem tego na którym została odebrana. W taki sposób zawsze są też przesyłane ramki wysyłane na adres rozgłoszeniowy (bradcast).

Ethernet pozwala na wirtualne podzielenie pojedynczej sieci lokalnej na wiele niezależnych (nie komunikujących się ze sobą w warstwie ethernetu) sieci, nazywanych VLAN. Działanie tego mechanizmu opiera się na zastosowaniu zarządzalnych switchy, które programowo mogą być dzielone na części zapewniające separację ruchu poszczególnych VLANów. Ponadto wybrane porty takiego switcha mogą być przypisane do różnych części (celem udostępnienia do innego switcha lub hosta kilku sieci wirtualnych), w takim przypadku do ramek ethernetowych wysyłanych tym portem dodawana jest informacja do którego VLANu należą (dwu bajtowy numer), a w przypadku ramek otrzymywanych na podstawie tego numeru odbywa się ich kierowanie do odpowiedniej "części" przełącznika (mówimy o VLANach tagowanych). Możliwe jest aby jeden wybrany VLAN na takim porcie był nie tagowany (do jego ramek nie będzie dodawany numer, a otrzymywane pakiety bez numeru będą kierowane do niego.

Ethernet pozwala również na grupowanie kilku portów w jeden port wirtualny (tzw port trunking / bonding) celem zwiększenia przepustowości lub niezawodności łącza. A dzięki zastosowaniu w różnych typach sieci ethernet tego samego formatu ramki możliwe jest też stosunkowo proste zmienianie medium transmisyjnego (np. z kabla miedzianego na światłowód) z użyciem media-konwerterów.

W przewodowych sieciach Ethernet wykrywaniem zajętości medium transmisyjnego oraz wykrywaniem kolizji zajmuje się protokół CSMA/CD (Carrier Sense Multiple Access with Collision Detection - wielodostęp z rozpoznawaniem stanu kanału oraz wykrywaniem kolizji). Przed rozpoczęciem nadawania stacja musi sprawdzić czy medium jest wolne, jeżeli tak może zacząć nadawać, jeżeli dwie stacje zaczną nadawać równocześnie zostaje to wykryte, obie przerywają nadawanie i wznawiają po losowym czasie. Jednak ze względu na stosowanie głównie połączeń punkt-punkt full-duplex (osobne przewody do nadawania i osobne do odbioru), co ogranicza tzw. domenę kolizji do pojedynczego hosta, protokół ten nie odgrywa obecnie szczególnie istotnej roli.

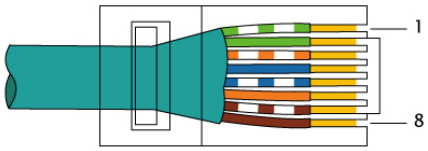
4.1 Kable

Sieć ethernetowa wykorzystuje 8 żyłowe kable złożone z 4 par. Najpopularniejszym przewodem jest kabel UTP kategorii 5e, czyli nieekranowana skrętka pozwalająca na pracę z częstotliwością 100 MHz. W przypadku instalacji okablowania strukturalnego często stosowane są wyższe kategorie okablowania a także kable dodatkowo ekranowane. Ekran może obejmować osobno każdą parę, jak też może być wspólny dla całego przewodu, może być wykonany z folii lub siatki. Np. SF/FTP oznacza kabel z ekranem z siatki i folii (SF/), gdzie dodatkowo każda para jest ekranowana folią (FTP).

W ramach poszczególnych par realizowana jest transmisja różnicowa, czyli istotne jest napięcie pomiędzy przewodami w parze, a nie napięcie na danym przewodzie (w odniesieniu do jakiegoś zewnętrznego poziomu odniesienia). Standard 100Mb/s (dokładniej 100BASE-TX) wykorzystuje jedynie dwie pary przewodów, standard 1Gb/s (1000BASE-T) wykorzystuje wszystkie 4 pary przewodów. Długość kabla pomiędzy dwoma urządzeniami nie powinna przekraczać 100 m. Wykorzystywanie skręconych par przewodów (jeden skręt na 6-10 cm kabla) ma na celu eliminację zakłóceń transmisji - (w uproszczeniu) zakłócenia wchodzą tak samo na oba przewody i różnica miedzy nimi nie zmienia się.

Kable zakańczane są gniazdami bądź wtykami typu RJ-45 montowanymi według jednego z dwóch schematów kolorystycznych: EIA/TIA 568A lub 568B. Pierwotnie (dla sieci 100Mb/s lub starszych) użycie różnych standardów na obu końcach kabla służyło stworzeniu kabla skrosowanego[1].

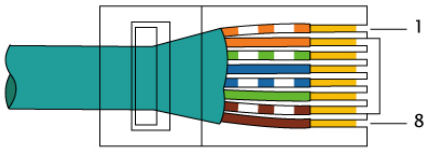
EIA/TIA 568A



Kolejność przewodów we wtyczce/gnieździe:

1. biało-zielony
2. zielony
3. biało-pomarańczowy
4. niebieski
5. biało-niebieski
6. pomarańczowy
7. biało-brązowy
8. brązowy

EIA/TIA 568B



Kolejność przewodów we wtyczce/gnieździe:

1. biało-pomarańczowy
2. pomarańczowy
3. biało-zielony
4. niebieski
5. biało-niebieski
6. zielony
7. biało-brązowy
8. brązowy

Wtyczki RJ-45 są wtyczkami zaciskanymi na przewodzie (bez konieczności odizolowywania żył). Gniazda RJ-45 najczęściej wykonywane są ze złączem typu IDC (Insulation Displacement Connector, KRONE/LSA) służącym do podłączenia przewodu również bez konieczności odizolowywania poszczególnych żył.

5 Zadania

Zadanie 5.0.1

Przeanalizuj kod programu z rozdziału 1.3.2. Zapoznaj się z dokumentacją modułu ipaddress[a] i pamiętając, że adres IPv4 jest 32 bitową liczbą, zmodyfikuj ten program aby zamiast na adresach IPv6 działał na IPv4.

a. Można ją wyświetlić uruchamiając po jego zaimportowaniu: `help('ipaddress')`

Zadanie 5.0.2

Korzystając z narzędzi służących do diagnozowania sieci sprawdź czy host `ciekawi.icm.edu.pl` jest dostępny. Jakiego polecenia użyłeś(aś) w tym celu? Co jeszcze mówi wynik tego polecenia?

1. Połączenie takie przy jednakowych urządzeniach, gdzie nadajnik i odbiornik trafia zawsze na te same piny, zamieniało na kablu nadajnik z odbiornikiem, umożliwiając transmisje między nimi. Aktualnie zdecydowana większość urządzeń obsługuje protokół *Auto MDI-X*, który umożliwia automatyczne ustalenie na których pinach odbywa się nadawanie, a na których odbiór. W rzadkich przypadkach konieczne może być jednak zastosowanie kabla skrosowanego

Zadanie 5.0.3

Korzystając z narzędzi służących do diagnozowania sieci ustal jaką trasą podróżują pakiety z Twojego komputera do `www.opcode.eu.org` oraz do `www.example.org`. Jakiego lub jakich poleceń użyłeś(aś) w tym celu? Co jeszcze mówi ich wynik? Co możesz powiedzieć porównując uzyskane trasy?

Zadanie 5.0.4

Ustal (wszystkie) adresy IPv4 i IPv6 serwera `www.bitbucket.org`. Zastanów się czemu może służyć to że niektóre nazwy domenowe rozwijają się na wiele różnych adresów IP.

Zadanie 5.0.5

Korzystając bezpośrednio z poleceń protokołu HTTP i programu `nc` (`netcat`) lub `telnet`, pobierz i wyświetl kod strony `http://www.opcode.eu.org/`.

Zadanie 5.0.6

Zadanie 5.0.5 można rozwiązać przy pomocy netcat'a bez dodatkowych opcji, jednak jeżeli stroną do pobrania byłoby np. `http://www.icm.edu.pl` to należałoby skorzystać z opcji `-C` netcat'a (w przeciwnym razie serwer zwraca błąd 400 "Bad Request"). Sprawdź co robi ta opcja i zastanów się dlaczego w przypadku niektórych serwerów jest konieczna a w przypadku innych nie? Co na ten temat mówi standard HTTP?

Zadanie 5.0.7

Korzystając bezpośrednio z poleceń protokołu SMTP, programu `nc` (`netcat`) lub `telnet` i serwera `mail.opcode.eu.org` wyślij mail do `rrp@opcode.eu.org` w taki sposób aby:

- nadawca kopertowy był sfałszowany (domena w której znajduje się jego adres powinna istnieć, np. użyj `ciekawi.icm.edu.pl`)
- nadawca i odbiorca nagłówkowy był sfałszowany

Możesz poszaleć z adresami (zwłaszcza nagłówkowymi). Jeżeli nie popełnisz błędu mail trafi do prowadzących zajęcia, zatem jeżeli oczekujesz weryfikacji wykonania zadania to podpisz się w nim. Nie pisz też niczego czego nie chiałbyś aby Twój prowadzący przeczytał ☺.

6 Zadania dodatkowe

Zadanie 6.0.1

Korzystając z standardowych narzędzi shellowych napisz skrypt który sprawdzi dostępność hosta `ciekawi.icm.edu.pl` i w przypadku jego niedostępności wypisze `niedostępny`, a w przypadku dostępności nie wypisze niczego.
Informacja: skrypty tego typu, uruchamiane automatycznie co pewien czas np. przy pomocy cron'a często są wykorzystywane do wysłania powiadomień o niedostępności danej maszyny lub usługi lub podjęcia automatycznie jakiś działań.

Zadanie 6.0.2

Zobacz czy rozwiązanie zadania 5.0.2 zadziała gdy użyjesz nazwy serwera zawierającej polskie znaki: `licealiści.icm.edu.pl`. Jak myślisz, dlaczego polskie znaki są tak rzadko używane w nazwach domenowych?

7 Praca domowa

7.1 Instrukcja wysyłania rozwiązań

Rozwiązania zadań domowych należy przesłać na adres ciekawi.pracownia@icm.edu.pl wpisując jako temat wiadomości `g2.x PD8`, gdzie `x` to numer grupy, np. `g2.1 PD8` dla grupy nr. 1, itd. Zadania domowe są nie obowiązkowe, jednak zachęcamy do ich robienia i wysyłania rozwiązań (nawet niekompletnych).

Termin nadsyłania zdań domowych to 2020-04-28 godzina 16^{00}. Jeżeli wysłałeś rozwiązania w terminie, ale nie były one w 100% poprawne i dostałeś od sprawdzającego możliwość wysłania poprawki masz na to dodatkowe 4 dni.

Na ten adres można także nadsyłać ewentualne pytania do zadań (zarówno domowych jak i innych zamieszczonych w skrypcie), w tym wypadku także prosimy o umieszczenie w temacie wiadomości `g2.x`, gdzie `x` to numer grupy.

7.2 Zadania domowe

Zadanie 7.2.1 — 1 pkt

Ustal czy host o adresie IPv4 192.168.65.20 należy do sieci 192.168.33.15/19.

Zadanie 7.2.2 — 1 pkt

Ustal czy host o adresie IPv6 2001:6a0:0:21::60:2 należy do sieci 2001:6a0:0:10::/58.

Zadanie 7.2.3 — 1 pkt

Ustal adresy serwerów DNS posiadających informację o domenie *gov*. Podaj polecenie którego użyłeś.

Zadanie 7.2.4 — 3 pkt

Polecenie `ip r` pokazało następującą tablicę routingu:

```
default via 192.168.29.2 dev eth0.2
192.168.29.192/27 dev eth0.2  proto kernel  scope link  src 192.168.29.193
172.16.16.0/27 via 172.16.18.2 dev tun5
172.16.16.48/28 dev wlan0  proto kernel  scope link  src 172.16.16.49
172.16.18.0/30 dev tun5  proto kernel  scope link  src 172.16.18.1
192.168.29.0/24 dev eth0  proto kernel  scope link  src 192.168.29.1
```

Ustal trasę (urządzenie którym zostanie wysłany pakiet oraz jeżeli jest potrzebny to adres routera do którego będzie przesyłany) dla następujacych adresów IP:

- 8.8.8.8
- 192.168.29.202
- 172.16.16.15