Piotr Kowalski
KAiTI

Sieci komputerowe
- Protokoły warstwy transportowej

Plan i problematyka wykładu

1. Funkcje warstwy transportowej i wspólne cechy typowych protokołów tej warstwy
2. Protokół UDP
   - Ogólna charakterystyka, format wiadomości, co UDP zapewnia, a czego nie gwarantuje (głównie to drugie)
3. Protokół TCP
   - Ogólna charakterystyka
   - Przesyłanie danych, inicjacja i kończenie połączenia
   - Segment TCP – charakterystyka i flagi
   - Algorytm sliding-window w praktyce

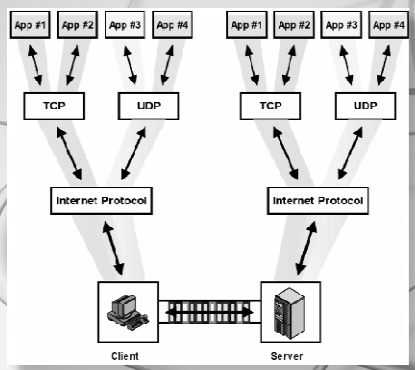
Funkcje warstwy transportowej

Protokół IP:
- bezpołączeniowy (ang. connectionless)
- zawodny (ang. unreliable)
- brak potwierdzeń (ang. unacknowledged)

Wymieniona powyżej lista cech protokołu IP to tak naprawdę jego braki. Braki te uzupełnia w pewnym stopniu wyższa warstwa intersieci – warstwa transportowa.

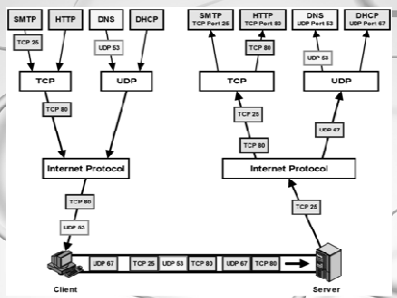
Dwa podstawowe protokoły tej warstwy to TCP (ang. Transmission Control Protocol) i UDP (ang. User Datagram Protocol). Pierwszy zawiera pełen zestaw funkcji (kosztem narzutu komunikacyjnego), drugi definiuje jedynie niezbędne minimum wymaganych funkcji.

Pakiety IP a aplikacje



Porty jako metoda multipleksowania pakietów

Adresowanie w warstwie transportowej jest osiągane przez **porty**. Każdy numer portu identyfikuje proces działający w ramach węzła o danym adresie IP.



Gniazda i przypisanie usług do portów

Połączenie numeru IP komputera i portu na którym odbywa się komunikacja nazywamy **gniazdem** (ang. socket). Dwa gniazda jednoznacznie definiujące w danej chwili transmisję w całym Internecie, można zapisać w następujący sposób: (212.51.219.50.23 : 212.51.219.4.6000)

Numer portu jest liczbą 16 bitową, czyli może przyjmować wartości od 0 do 65535. Wprowadzono następujący podział przestrzeni portów:
- zakres 0 do 1023 - zarezerwowany dla tzw. dobrze znanych portów (ang. well-known/privilleged ports)
- zakres 1024 do 49151 – zarezerwowany dla zarejestrowanych portów/portów użytkownika (ang. registered/user ports)
- zakres 49152 do 65535 – zarezerwowany dla prywatnych/dynamicznie alokowanych portów (ang. private/dynamic ports)

Zbiór dobrze znanych portów - przykłady

Port	TCP/UDP	Usługa
20	TCP	FTP (dane)
21	TCP	FTP (inf. kontrolne)
22	TCP	SSH
25	TCP	SMTP
53	TCP+UDP	DNS
80	TCP	HTTP
110	TCP	POP3
143	TCP	IMAP
161,162	UDP	SNMP
179	UDP	BGP
443	TCP	HTTPS
520	UDP	RIP-2

Więcej:
/etc/services (Linux)
Windows/System32/drivers/etc/services (Windows XP)

UDP

UDP jest bardzo prostym protokołem warstwy transportowej opracowanym na początku lat 80.

UDP implementuje:
- Przesyłanie danych pomiędzy dwoma procesami odległych węzłów
- i… nic więcej!

Czego UDP nie zapewnia:
- połączenia (wiadomości są po prostu „wypychane" przez łącze) i wykrywania zagubionych komunikatów
- potwierdzeń i gwarancji dostarczenia wiadomości
- kontroli przepływności oraz właściwej kolejności dostarczenia wiadomości

Wiadomość UDP

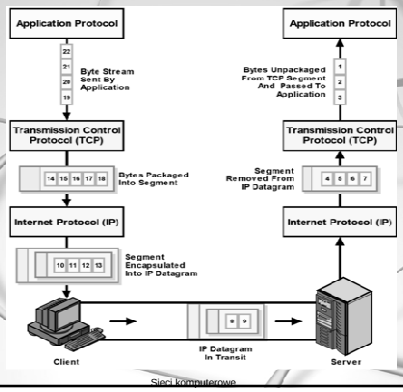
Bity 0-15	Bity 0-15
Port źródłowy	Port przeznaczenia
Długość	Suma kontrolna
DANE	

Protokół TCP

Protokół TCP jest powiązany wspólną historią z protokołem IP, początkowo stanowiły one jeden spójny protokół sieciowy

Cechy:
- Zorientowany na połączenie – obie strony je nawiązują i obie strony mogą przesyłać do siebie dane (dwukierunkowy)
- Obsługa wielu połączeń równocześnie
- Niezawodny – sprawdza integralność danych, zapewnia ich retransmisję w przypadku błędu, definiuje potwierdzenia
- Strumieniowy – pozwala na transmisję strumienia danych, sam dzieląc je na segmenty w zależności od potrzeb, zapewnia także kontrolę przepływności

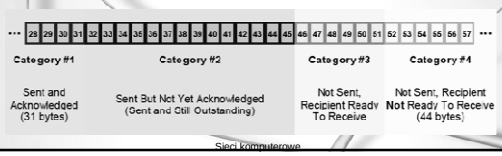
Segmenty TCP



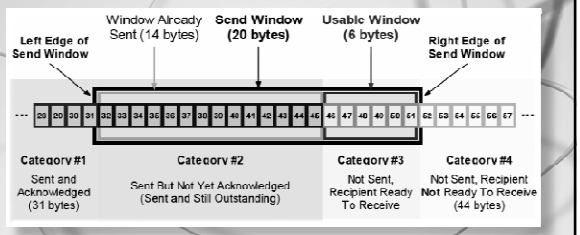
Przesyłanie danych w TCP

Każdy segment zawiera numer sekwencji, TCP dla niezawodności transmisji implementuje schemat **PAR** (ang. positive acknowledgements with retransmission) - dla każdej sekwencji musi być wysłane potwierdzenie o odpowiadającym jej numerze (nadawca uruchamia też zegar w momencie wysyłania pakietu i wysyła ten pakiet ponownie, gdy minie odpowiedni czas, a potwierdzenie nie nadejdzie), a także zasygnalizowany już wcześniej mechanizm ruchomego okna (ang. sliding window) pracującego na przesyłanym strumieniu danych.

Każdy bajt jest przypisany do jednej z 4 kategorii:



Okno transmisji – szkic



Połączenia w TCP – flagi

W TCP w celu efektywnego i niezawodnego przesyłania danych między węzłami nawiązywane są połączenia.

Segment TCP zawiera specjalne pole definiujące flagi, określające typ komunikatu przez ten segment przenoszonego, flagi te będą opisane później, na razie ograniczmy się do scharakteryzowania następujących:

- Flaga SYN (ang. synchronize) – używana do nawiązania połączenia, jedna z funkcji: synchronizacja numerów sekwencji
- Flaga FIN (ang. finish) – używana do zakończenia połączenia
- Flaga ACK (ang. acknowledgement) – używana do potwierdzenia odbioru wiadomości

Diagram stanów – nawiązanie połączenia

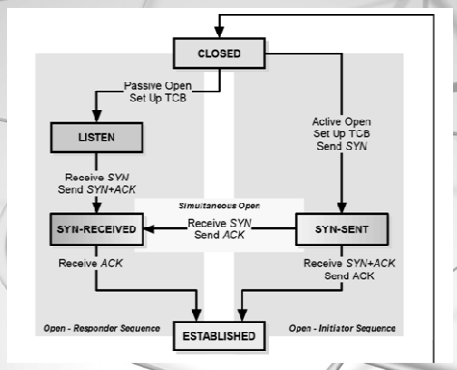
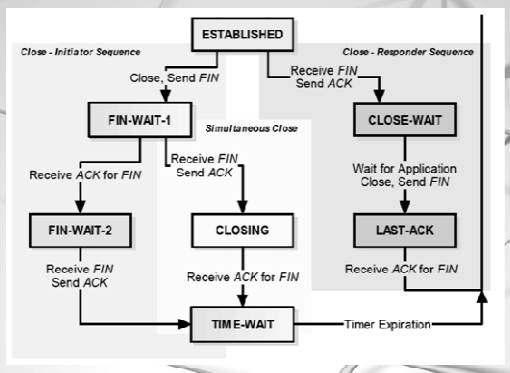


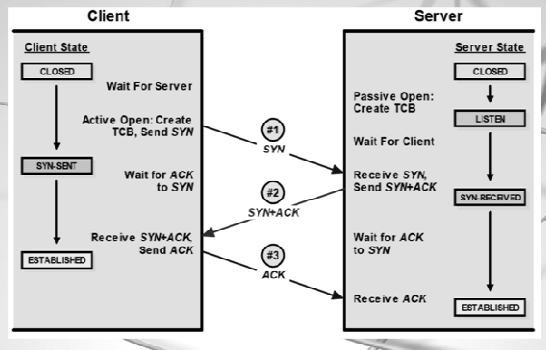
Diagram stanów – zakończenie połączenia



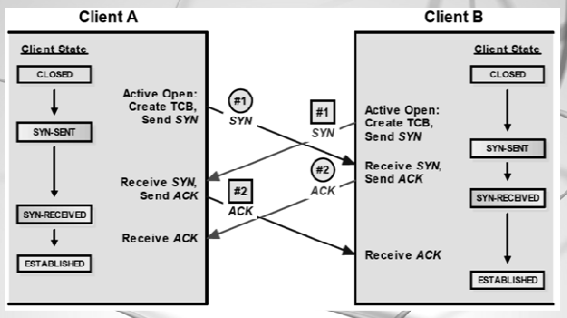
Możliwe stany klienta/serwera TCP (1)

- CLOSED – brak połączenia
- LISTEN – nasłuchiwanie (serwer), gotowość do odbioru segmentu inicjalizującego (SYN), po jego odebraniu wysyła się SYN i ACK, i do stanu SYN-RECEIVED
- SYN-SENT – klient po wysłaniu segmentu inicjalizującego (SYN), czeka na odpowiedź gdy jest to SYN – do stanu SYN-RECEIVED, gdy SYN+ACK – wyślij ACK i do stanu ESTABLISHED
- SYN-RECEIVED – serwer po odebraniu SYN i wysłaniu swojego SYN, czeka na ACK, po odebraniu ESTABLISHED
- ESTABLISHED – nawiązane połączenie, aby je skończyć wysyła się FIN – i skok do FIN-WAIT-1

Three-way handshake (nawiązanie połączenia TCP)



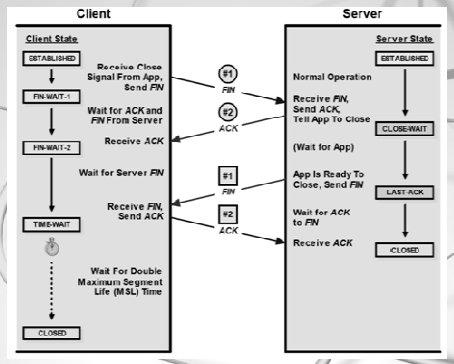
Równoczesne nawiązanie połączenia



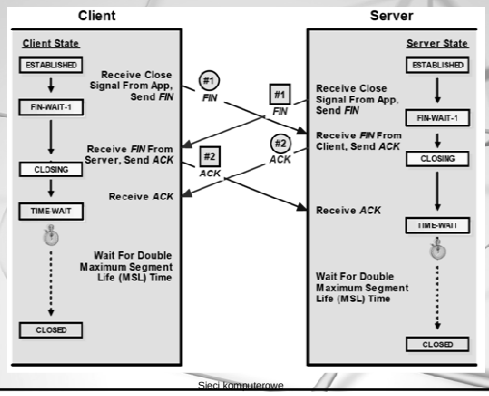
Możliwe stany klienta/serwera TCP (2)

- CLOSE-WAIT – otrzymano FIN (prośbę o zakończenie połączenia), odsyła się FIN i przechodzi do LAST-ACK
- LAST-ACK – czeka na ACK, po jego otrzymaniu przechodzi do CLOSED
- FIN-WAIT-1 – po wysłaniu swojego FIN czeka się na odpowiedź. Gdy jest to też FIN, należy odesłać ACK i do CLOSING, gdy jest to ACK – przejść do FIN-WAIT-2
- FIN-WAIT-2 – oczekiwanie na FIN, po jego odebraniu należy odesłać ACK i przejść do TIME-WAIT
- CLOSING – oczekiwanie na ACK, po odebraniu – TIME-WAIT
- TIME-WAIT – po określonym czasie (2*maksymalny czas życia segmentu) przechodzi się do CLOSED

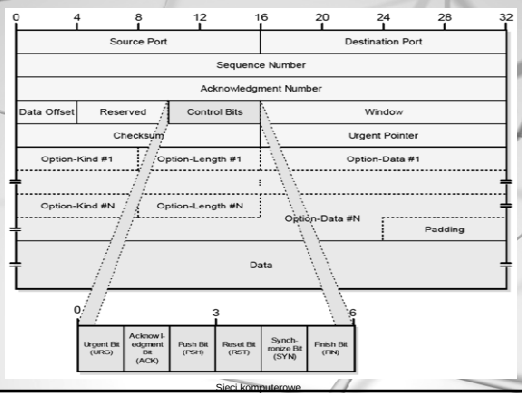
Zakończenie połączenia



Równoczesne zakończenie połączenia



Segment TCP



Pola segmentu TCP

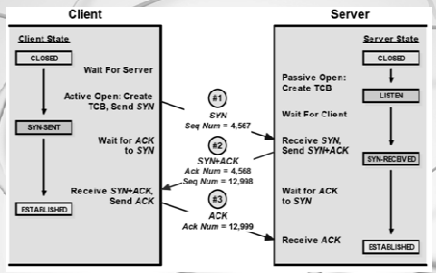
- **Numer sekwencji** – 4 bajtowy numer wskazujący położenie w strumieniu pierwszego bajtu danych segmentu. Przy nawiązywaniu transmisji (flaga SYN) zawiera ISN (ang. initial sequence number) nadawany przez nadawcę. Pierwszy bajt danych otrzymuje numer sekwencji ISN+1.
- **Numer potwierdzenia** – jeśli jest ustawiona flaga ACK, pole to zawiera wartość następnego numeru sekwencji kolejnego segmentu, którą nadawca spodziewa się otrzymać.
- **Przesunięcie danych** – liczba 32-bitowych słów w nagłówku TCP
- **Okno** – określa rozmiar okna jakim wysyłający dysponuje dla odbierania danych (pozwala na sterowanie przepływem danych – 0 oznacza „wstrzymaj transmisję”)

Pole flag (bitów sterujących)

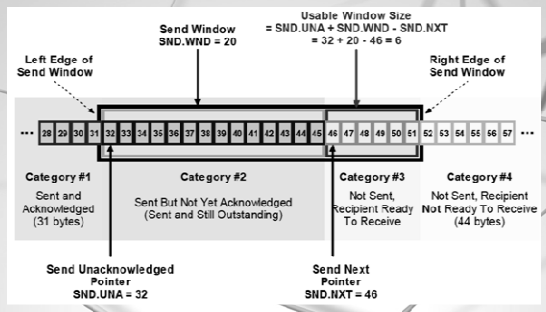
- **URG** – segment priorytetowy (pole Urgent Pointer wskazuje koniec danych o priorytetowej „wartości”)
- **ACK** – segment potwierdzenia
- **PSH** – dane powinny być natychmiast przekazane do wyższej warstwy
- **RST** – funkcja resetująca połączenie („delta mamy problem”)
- **SYN** – segment synchronizacji numerów sekwencji
- **FIN** – segment sygnalizujący chęć zakończenia połączenia

Numer sekwencji i jego synchronizacja

Inicjujący połączenie powinien wybrać na początku numer sekwencji - stosuje w tym celu licznik, zwiększający się co 4 mikrosekundy aż do INT_MAX (zatem dopiero po 4 godzinach numer sekwencji może się powtórzyć).

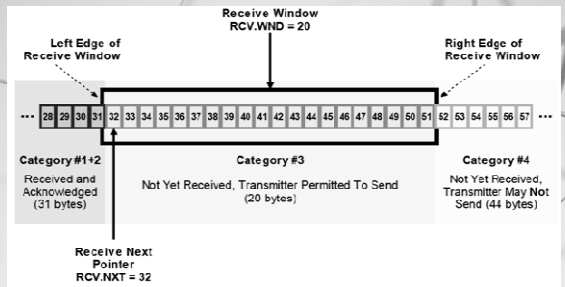


Sliding window w TCP - nadawca



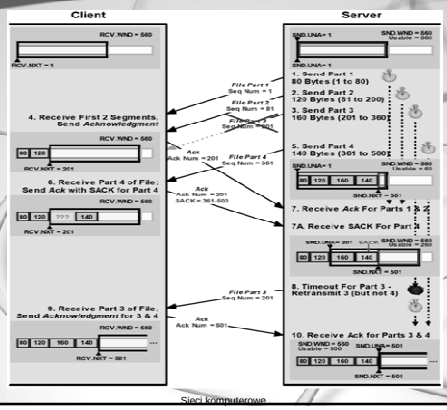
SND.UNA – odpowiada numerowi sekwencji wysyłanych danych

Sliding window w TCP - odbiorca



RCV.NXT – odpowiada numerowi odsyłanemu w ACK

Selective ACK (gdy zgubimy segment)



DZIĘKUJĘ ZA UWAGĘ!

NASTĘPNY WYKŁAD:

IPV6