Sieci komputerowe Wykład 4 - warstwa transportowa, TCP cz. 2

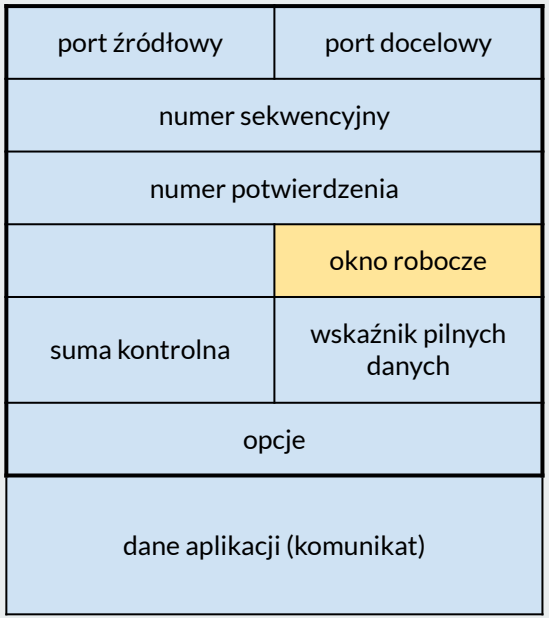
Agata Janowska
2022

Przypomnienie gdzie jesteśmy

- przekazanie komunikatów między procesami tworzącymi aplikację w sposób
  - połączeniowy po TCP
  - bezpołączeniowy po UDP

Spis treści

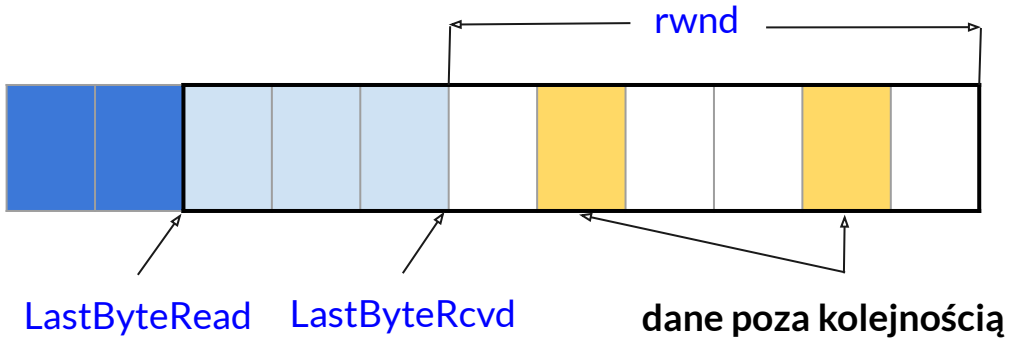
Protokół TCP
Opcja skalowania okna



- dla sieci o dużych przepustowościach i dużych opóźnieniach okno opisane liczbą 16-bitową może być zbyt małe
- stosowana jest opcja TCP skalowania okna
  - opcja zawiera wykładnik ***n***, który może przyjmować wartości od 0 do 14
  - rozmiar okna to $(2^{16}-1) * 2^{n}$
- opcja skalowania może się pojawiać jedynie w segmentach SYN, więc jest stała dla danego połączenia
  - jeśli strona, która nawiązuje połączenie umieści tę opcję w segmencie SYN, ale nie dostanie jej w segmencie SYN ACK przesłanym przez drugą stronę, oznacza to odmowę

Protokół TCP
Potwierdzanie selektywne

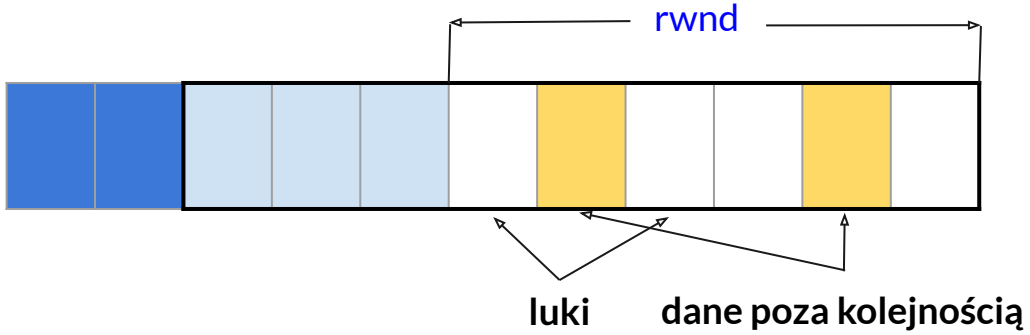
Opcja SACK (*ang. Selective Acknowledgement*)



- klient i serwer mogą ustawić opcję **SACK-Permitted** w czasie ustanawiania połączenia TCP (w segmencie SYN lub SYN-ACK)

Protokół TCP
Potwierdzanie selektywne

Opcja SACK (*ang. Selective Acknowledgement*)

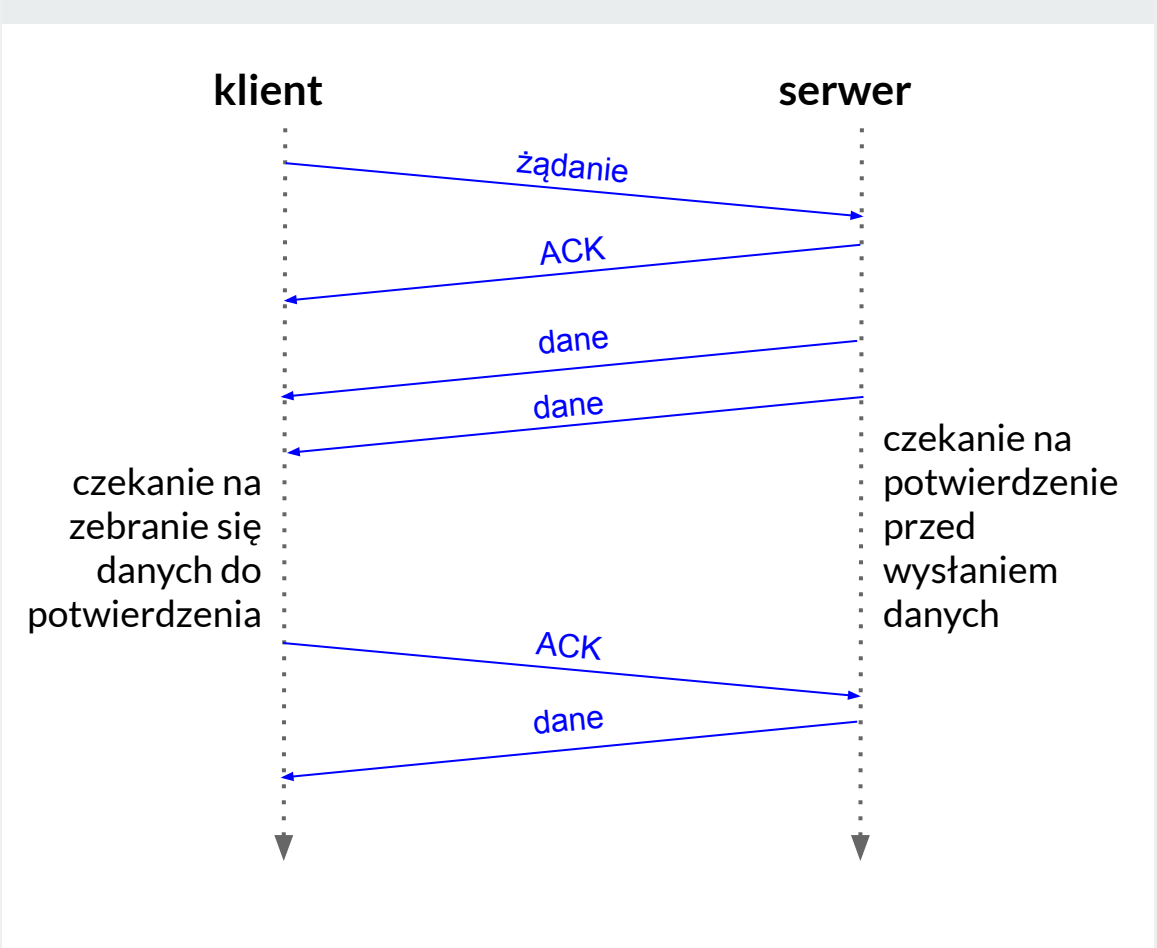


- zadaniem nadawcy jest uzupełnienie luk u odbiorcy
- segment ACK jest wzbogacony o co najwyżej 3 bloki opisujące dane poza kolejnością, blok to numery: pierwszego i (ostatniego + 1) bajtu
- w czasie RTT mogą zostać wypełnione 3 luki

Protokół TCP
Problem małych pakietów i algorytm Nagle'a

- mały pakiet = mniejszy niż MSS
- algorytm Nagle'a polega na łączeniu kilku małych komunikatów i wysyłaniu ich w jednym segmencie (minimalizacji liczby wysyłanych segmentów)
- dopóki nadawca nie otrzyma potwierdzenia ostatnio wysłanego segmentu, dopóty wstrzymuje się z wysłaniem kolejnych danych
- nie współpracuje ze skumulowanym potwierdzaniem
- można wyłączyć algorytm Nagle'a (opcje gniazd), z tej możliwości korzystają np. gry sieciowe, ssh

Protokół TCP
Algorytm Nagle'a i skumulowane potwierdzenia



Protokół TCP
Przeciążenie sieci

- przeciążenia sieci pojawia się kiedy zbyt wiele nadawców wysyła dane z nadmierną szybkością
- utrata pakietów jest zwykle wynikiem przepełnienia buforów routera w przeciążonej sieci
- retransmisja eliminuje efekty przeciążenia, ale nie jego przyczynę
- konieczne jest wprowadzenie mechanizmów ograniczających ilość danych wysyłanych przez nadawców

Protokół TCP
Kontrola przeciążenia

polega na ograniczeniu szybkości nadawcy w przypadku wykrycia przeciążenia sieci

Jak nadawca dostosowuje swoją szybkość do stanu przeciążenia sieci?

- **utrata segmentu** oznacza przeciążenie, po tym zdarzeniu nadawca powinien **zmniejszyć szybkość**
- otrzymanie **potwierdzenia** oznacza, że sieć dostarcza segmenty, można więc **zwiększyć szybkość**
- **testowanie przepustowości** – nadawca zwiększa szybkość do momentu pojawienia się przeciążenia

Protokół TCP
Algorytm kontroli przeciążenia

- **Powolny start**
- **Unikanie przeciążenia**
- **Szybkie przywracanie**

cwnd – rozmiar okna przeciążenia (*ang. congestion window*)

ssthresh – limit powolnego startu (*ang. slow start threshold*)

LastByteSend−LastByteAcked ≤ min {rwnd, cwnd}

- początkowo: cwnd = MSS
- po otrzymaniu pierwszego potwierdzenia dla danego segmentu: cwnd = cwnd + MSS
- jeśli upłynie czas oczekiwania na potwierdzenie: ssthresh = ½ cwnd, cwnd = MSS
- jeśli cwnd osiągnie ssthresh: przejście do fazy unikania przeciążenia
- po otrzymaniu 3 duplikatów potwierdzenia: szybka retransmisja i przejście do fazy szybkiego przywracania

Protokół TCP
Algorytm kontroli przeciążenia

- **Powolny start**
- **Unikanie przeciążenia**
- **Szybkie przywracanie**

cwnd – rozmiar okna przeciążenia (*ang. congestion window*)

ssthresh – limit powolnego startu (*ang. slow start threshold*)

LastByteSend−LastByteAcked ≤ min {cwnd, rwnd}

- po otrzymaniu pierwszego potwierdzenia dla danego segmentu: cwnd = cwnd + MSS/cwnd
- jeśli upłynie czas oczekiwania na potwierdzenie: ssthresh = ½ cwnd, cwnd = MSS przejście do fazy powolnego startu
- po otrzymaniu 3 duplikatów potwierdzenia: szybka retransmisja, ssthresh = ½ cwnd, cwnd = cwnd/2 + 3MSS i przejście do fazy szybkiego przywracania

Protokół TCP
Algorytm kontroli przeciążenia

- **Powolny start**
- **Unikanie przeciążenia**
- **Szybkie przywracanie**

cwnd – rozmiar okna przeciążenia (*ang. congestion window*)

ssthresh – limit powolnego startu (*ang. slow start threshold*)

LastByteSend−LastByteAcked ≤ min {cwnd, rwnd}

- po otrzymaniu potwierdzenia: cwnd = cwnd + MSS
- po otrzymaniu „dobrego” potwierdzenia (potwierdzenia utraconego segmentu): zmniejszenie cwnd do poprzedniej wartości i przejście do fazy unikania przeciążenia
- jeśli upłynie czas oczekiwania na potwierdzenie: ssthresh = ½ cwnd, cwnd = MSS przejście do fazy powolnego startu

  (faza opcjonalna, tylko niektóre implementacje)

Protokół TCP
Bity kontroli przeciążenia

port źródłowy	port docelowy
numer sekwencyjny	
numer potwierdzenia	
	okno robocze
suma kontrolna	wskaźnik pilnych danych
opcje	
dane aplikacji (komunikat)	

długość nagłówka		NS	CWR	ECE	URG	ACK	PSH	RST	SYN	FIN

- **CWR, ECE** – do powiadamiania o przeciążeniu
- ECE – Explicit Congestion Notification Echo
- CRW – Congestion Window Reduced

Mechanizm ECN (*ang*. Explicit Congestion Notification)

Kontrola przeciążenia wspomagana przez warstwę sieciową

- umożliwia warstwie sieci sygnalizowanie przeciążenia
- router, który wykryje długotrwałe przeciążenie, ustawia odpowiednią kombinację bitów w nagłówku datagramu IP skierowanego do odbiorcy
- odbiorca przekazuje informacje do nadawcy ustawiając bit ECE w nagłówku TCP
- nadawca dzieli okno przeciążenia na pół i ustawia bit CWR
- odpowiednia kombinacja bitów w nagłówku IP informuje router, że nadawca i odbiorca obsługują ECN

Protokół TCP
Mechanizm podtrzymania aktywności (*ang. keepalive*)

- sprawdzanie drugiej strony połączenia bez wpływania na zawartość strumienia danych
- nie jest częścią specyfikacji TCP, z drugiej strony wiele aplikacji korzysta z takiej funkcji, kontrowersyjne
- metoda wykrywania nieaktywnych klientów wykorzystywana np. przez serwery WWW, pocztowe, ssh
- pozwala uniknąć usunięcia połączenia na routerach NAT

Protokół TCP
Mechanizm podtrzymania aktywności (*ang. keepalive*)

- może być skonfigurowane dla jednej strony połączenia, obydwu lub żadnej
- jeśli w połączeniu przez okres czasu keepalive time (2h) brakuje aktywności, to strona wysyła **sondę** do drugiej strony
- jeśli nie otrzyma odpowiedzi, to czynność jest powtarzana keepalive probes (9) razy co keepalive interval (75s) jednostek czasu
- **sonda** jest pustym lub 1-bajtowym segmentem z numerem sekwencyjnym o 1 mniejszym od największego numeru ACK dotąd otrzymanego

net.ipv4.tcp_keepalive_time
net.ipv4.tcp_keepalive_intvl
net.ipv4.tcp_keepalive_probes

Protokół TCP
Mechanizm podtrzymania aktywności (*ang. keepalive*)

1. jeśli druga strona pracuje i jest dostępna, to przesyła potwierdzenie odebrania danych, zegar jest zerowany
2. jeśli druga strona nie odpowiada na tę ani kolejne sondy, to połączenie jest uznane za zakończone "Connection timeout"
3. jeśli druga strona przeszła awarię i ponowne uruchomienie, to odpowie segmentem RST, który powoduje zamknięcie połączenia "Connection reset by peer"
4. jeśli druga strona pracuje, ale nie jest dostępna, to połączenie również jest uznane za zakończone jak w pkt. 2

Porównanie TCP i UDP

	UDP	**TCP**
Nagłówek	8B	co najmniej 20B
Typ	datagramowy	strumieniowy
Połączeniowy	nie	tak
Segment z niepoprawną sumą kontrolną	odrzucany	retransmitowany
Zachowanie kolejności	nie	tak
Dane wysyłane natychmiast	tak	nie
Kontrola przepływu i przeciążenia	nie	tak
Dodatkowy narzut	brak	potwierdzenia, retransmisje

Porównanie TCP i UDP

Typowe zastosowania

Zastosowanie	Przykładowy protokół warstwy aplikacji	Protokół transportowy
Poczta elektroniczna	SMTP	TCP
Zdalny dostęp	SSH	TCP
Technologia WWW	HTTP	TCP
Strumieniowa transmisja danych audio i wideo	zwykle zastrzeżony	UDP lub TCP
Wideokonferencje	zwykle zastrzeżony	UDP lub TCP
Translacja nazw	DNS	UDP

Porównanie TCP i UDP
Zalety UDP

- nie korzysta z mechanizmu kontroli przeciążenia, więc może wysyłać dane tak szybko jak chce (istotne np. przy wideokonferencjach)
- nie ma opóźnienia związanego z nawiązywaniem połączenia
- nie utrzymuje danych dotyczących stanu połączenia, jest w stanie obsłużyć znacznie więcej aktywnych klientów
- nowy trend – protokół QUIC (*ang. Quick UDP Internet Connection*) korzysta z UDP i zapewnia niezawodność w warstwie aplikacji, może stać się kolejnym protokołem transportowym

Thank you!

Due to COVID19
all TCP applications
are being converted to UDP
to avoid handshakes