# Protokół TCP

## Uwagi ogólne:

TCP jest protokołem transportowym zapewniającym niezawodną wymianę danych między dwiema częściami aplikacji sieciowej. Jest protokołem połączeniowym, tzn. tworzy połączenie wirtualne między dwoma punktami końcowymi. Punkt taki jest parą składającą się z adresu IP oraz numeru portu. TCP działa trójetapowo, przy czym poszczególne etapy to: otwieranie, utrzymywanie i zamykanie połączenia. Efektywnie, połączenie wirtualne jest równoważne niezawodnemu połączeniu fizycznemu.

Segmenty (jednostki danych protokołu TCP) przesyłane między dwiema częściami aplikacji sieciowej mogą ulec zniekształceniu, zagubieniu, powtórzeniu, opóźnieniu, oraz dotrzeć w kolejności innej niż ta, w jakiej zostały wysłane. Protokół TCP zapewnia, że wysłane segmenty dotrą do właściwej aplikacji w komplecie, bez zniekształceń, w prawidłowej kolejności i bez powtórzeń.

Połączenie TCP jest realizowane między parą tzw. punktów końcowych (ang. connection endpoints lub sockets), gdzie punkt końcowy (gniazdo) jest parą złożoną z adresu IP i numeru portu identyfikującego aplikację sieciową, w ramach której połączenie jest nawiązywane.

Porcja informacji przesyłana w ramach sesji TCP jest nazywana segmentem. Segment TCP składa się z nagłówka TCP oraz danych TCP.

Dane TCP przesyłane w jedną stronę tworzą strumień, czyli uporządkowany, spójny ciąg bajtów. Każdy oktet danych należący do strumienia ma swój numer porządkowy zwany numerem sekwencyjnym oktetu. Strumień danych jest identyfikowany przez numer generowany losowo podczas otwierania połączenia TCP. Chodzi przy tym o to, aby uniknąć przemieszania danych należących do różnych sesji TCP. Jeśli strumień identyfikowany jest numerem x, to x+1 jest numerem sekwencyjnym pierwszego oktetu strumienia. Numer sekwencyjny pierwszego oktetu danych segmentu jest zarazem numerem sekwencyjnym tego segmentu.

1 2 3 4 5 <- to jest strumień (bajtów)

1 3 5 4 2 <- to nie jest strumień (zmieniona kolejność)

1 2 4 6 7 <- to też nie jest strumień (są ubytki)

1 2 2 3 3 <- to też nie jest strumień (są powtórzenia)

**Budowa nagłówka TCP**

<table>
<tr><td colspan="8">16-bit source port number</td><td colspan="8">16-bit destination port number</td></tr>
<tr><td colspan="16">32-bit sequence number</td></tr>
<tr><td colspan="16">32-bit acknowledgement number</td></tr>
<tr><td colspan="4">4-bit header length</td><td colspan="6">6-bits reserved</td><td>U</td><td>A</td><td>P</td><td>R</td><td>S</td><td>F</td><td colspan="4">16-bit window size</td></tr>
<tr><td colspan="8">16-bit TCP checksum</td><td colspan="8">16-bit urgent pointer</td></tr>
<tr><td colspan="16">Options (if any) Max 40-bytes</td></tr>
<tr><td colspan="16">Data</td></tr>
</table>

Znaczenie pól:

Numer portu źródłowego (Source port number): 16 bitów
Identyfikuje aplikację źródłową.

Numer portu przeznaczenia (Destination port number): 16 bitów
Identyfikuje aplikację docelową.

Numer sekwencyjny (Sequence number): 32 bity
Numer porządkowy pierwszego oktetu danych przesyłanych w bieżącym segmencie (jeśli nie jest ustawiona flaga SYN). Kiedy flaga SYN jest ustawiona, wtedy numer sekwencyjny jest numerem identyfikującym strumień danych wysyłanych do strony przeciwnej w ramach bieżącej sesji (ISN - Initial Sequence Number), a pierwszy oktet tego strumienia danych ma numer ISN+1. Przykładowo, jeśli ISN=52, to SeqNo(1 segm.) = 53

Numerowanie segmentów w nagłówku TCP za pomocą numerów sekwencyjnych:

ISN (numer identyfikacyjny strumienia) jest generowany losowo z przedziału $[0, 2^{32} – 1]$;

Numer sekwencyjny pierwszego segmentu = seqNo(1 segm.) = ISN + 1

Numer sekwencyjny następnego segmentu = numer sekwencyjny poprzedniego + liczba bajtów w polu danych poprzedniego;

Przykładowo, jeśli SeqNo(1 segm.) = 53, Dług(1 segm.) = 3, to SeqNo(2 segm.) = 56

Numer potwierdzenia (Acknowlegment number): 32 bity

Jeśli jest ustawiona flaga ACK, wówczas pole to zawiera numer sekwencyjny następnego oczekiwanego segmentu, który jest potwierdzeniem odbioru segmentu poprzedniego. Po zestawieniu połączenia flaga ACK jest zawsze ustawiona. Pole to służy do informowania strony przeciwnej (potwierdzania), że otrzymany został od niej segment poprzedni w stosunku do tego, którego numer sekwencyjny jest przesyłany w tym polu.

## Potwierdzanie segmentów za pomocą numerów potwierdzeń w nagłówku TCP:

Numer potwierdzenia segmentu k = Numer sekwencyjny segmentu k+1

Przykładowo, jeśli odbiorca potwierdza odebranie segmentu o numerze sekwencyjnym 56 i pole danych tego segmentu ma długość 7, to w polu „Acknowlegment numer" odbiorca wysyła do nadawcy wartość 63.

TCP header length: 4 bity

Liczba 32-bitowych (4-bajtowych) słów tworzących nagłówek TCP. Wskazuje, gdzie zaczyna się pole danych. Jest konieczny, ponieważ nagłówek TCP może zawierać pole opcji o zmiennej długości. Długość podstawowego nagłówka TCP (bez opcji) wynosi 20 bajtów i w tym przypadku wartość tego pola jest równa 5.

Reserved: 6 bitów

Pole zarezerwowane dla przyszłych zastosowań. Składa się z samych zer.

Bity kontrolne, inaczej flagi (Control bits, TCP flags): 6 bitów

URG: Jeśli ustawiony, to pole „Wskaźnik pilnych danych" jest znaczące
ACK: Jeśli ustawiony, to pole "Numer potwierdzenia" jest znaczące
PSH: Jeśli ustawiony, oznacza żądanie natychmiastowego przekazania danych do właściwej aplikacji. Jest to istotne w przypadku buforowania danych i przekazywania ich do aplikacji dopiero po zapełnieniu bufora.
RST: Jeśli ustawiony, oznacza jednostronne zamknięcie połączenia
SYN: Jeśli ustawiony, oznacza synchronizację numerów początkowych, jest przesyłany na etapie otwierania połączenia
FIN: Jeśli ustawiony, oznacza żądanie zamknięcia połączenia, jest przesyłany na etapie obustronnego zamykania połączenia

Długość okna przesuwnego (Window size): 16 bitów

Liczba oktetów danych, łącznie z oktetem określonym w polu "Numer potwierdzenia", które strona przeciwna może wysłać bez otrzymania potwierdzenia ich odebrania. Jeśli np. strona B otrzymała od strony A segment z numerem potwierdzenia równym x i długością okna przesuwnego równą s, to B może wysłać do A segmenty o łącznej długości pola danych nie przekraczającej s, bez oczekiwania na potwierdzenie otrzymania tych segmentów. Pierwszy z segmentów wysłanych przez stronę B będzie miał numer kolejny równy x. Długość okna przesuwnego równa zeru oznacza żądanie zaprzestania wysyłania danych.

Przykładowo, jeśli nadawca wysyła segmenty z polem danych o długości 100 B i otrzymał od odbiorcy potwierdzenie odbioru piątego segmentu, w którym w polu „długość okna" jest wartość 1024, to przed otrzymaniem kolejnego potwierdzenia może wysłać maksymalnie 10 segmentów (od 6 do 15).

Suma kontrolna (Checksum): 16 bitów

Jest liczona z uwzględnieniem wszystkich bajtów nagłówka TCP (oprócz bajtów pola sumy kontrolnej), pola danych TCP, a także tzw. pseudo-nagłówka TCP, w skład którego wchodzą trzy pola z nagłówka IP, a mianowicie: źródłowy IP, docelowy IP i protokół warstwy 4, oraz całkowita długość segmentu TCP (nagłówek wraz z polem danych).
Zauważmy że
długość segmentu TCP = długość pakietu IP – długość nagłówka IP,
czyli długość segmentu TCP łatwo obliczyć z odpowiednich pól nagłówka IP.
Pseudo-nagłówek jest stosowany w celu wyeliminowania błędnie trasowanych pakietów.

Wskaźnik pilnych danych (Urgent pointer): 16 bitów

Liczba oktetów tzw. „pilnych danych" przenoszonych w bieżącym segmencie. Jeśli numer sekwencyjny bieżącego segmentu jest równy x, a wskaźnik pilnych danych jest równy s, to x+s jest numerem porządkowym pierwszego oktetu danych znajdujących się bezpośrednio za danymi pilnymi. Pole to jest znaczące tylko w przypadku ustawienia flagi URG. Pilne dane są przekazywane do warstwy aplikacji poza kolejnością wynikającą z ich umiejscowienia w strumieniu.

Opcje (Options): zmienna długość, max 40 bajtów

Opcja zajmują przestrzeń za podstawowym nagłówkiem TCP. Jedna opcja może zajmować dowolną liczbę oktetów (nie większą niż 40), nie musi to być wielokrotność czterech. Opcje są uwzględniane przy liczeniu sumy kontrolnej nagłówka TCP. Istnieją dwa rodzaje opcji różniące się liczbą składników:

Rodzaj 1 (1 składnik): Pojedynczy oktet "Typ opcji".

Rodzaj 2 (3 składniki): Oktet "Typ opcji", oktet "Długość opcji", oraz oktety zawierające dane opcji.

Długość opcji drugiego rodzaju jest liczona z uwzględnieniem oktetów „Typ" i „Długość".

Jeśli pole opcji ma długość nie będącą wielokrotnością czterech bajtów, wówczas następuje wypełnienie pola opcji bitami zerowymi do pełnego słowa 32-bitowego.

Obecnie używane są następujące opcje:

| Rodzaj | Typ | Długość | Znaczenie |
|---|---|---|---|
| 1 | 0 | - | Koniec listy opcji. |
| 1 | 1 | - | Brak operacji. |
| 2 | 2 | 4 | Maksymalna długość segmentu danych. |

Pierwsza z wymienionych opcji umieszczana jest za wszystkimi pozostałymi opcjami; powinna być stosowana tylko wówczas, gdy koniec pola opcji nie pokrywa się z końcem nagłówka TCP. W takim przypadku stosowane jest wypełnienie zerami pozostałych bitów nagłówka TCP.

Opcja "Brak operacji" może być stosowana do rozdzielania opcji, na przykład w sytuacji, gdy pożądane jest, aby kolejna opcja zaczynała się od początku 32-bitowego słowa

nagłówka TCP. Ponieważ nie jest wymagane, aby początek opcji pokrywał się z początkiem 32-bitowego słowa nagłówka TCP, oprogramowanie realizujące protokół TCP musi prawidłowo przetwarzać również opcje nie spełniające tego warunku.

Kod binarny kolejnej opcji przedstawia się następująco:

```
-------------------------------------------------------------------------
|00000010|00000100|   maksymalna długość segmentu (4 bajty)   |
-------------------------------------------------------------------------
   Typ=2    Dług.=4
```

Służy ona do informowania strony przeciwnej o maksymalnej długości, jaką może mieć pole danych w odbieranych segmentach. Strona przeciwna nie powinna wysyłać większych segmentów, gdyż zostaną odrzucone. Może wystąpić tylko w segmentach używanych do inicjowania połączenia, tzn. mających ustawioną flagę SYN. Brak tej opcji oznacza zgodę na przyjmowanie dowolnie długich segmentów.

Dane (Data)
Pole danych TCP.

**Przykładowa sesja TCP**

Otwieranie sesji (połączenia TCP)

A: wysłanie segmentu inicjującego połączenie:
SYN=1, SeqNo=x
(x jest identyfikator strumienia danych płynącego od A)
B: wysłanie potwierdzenia odebrania powyższego segmentu od A, a zarazem zgoda na otwarcie połączenia:
SYN=1, ACK=1, SeqNo=y, AckNo=x+1
(y jest identyfikatorem strumienia danych płynącego od B)
A: wysłanie potwierdzenia odebrania powyższego segmentu od B, czyli potwierdzenie odebrania zgody na otwarcie połączenia:
ACK=1, SeqNo=x+1, AckNo=y+1

**Uwaga:** na etapie otwierania połączenia numer potwierdzenia (AckNo) otrzymujemy dodając jedynkę do numeru sekwencyjnego (SeqNo) segmentu potwierdzanego.

Utrzymywanie sesji (transmisja danych)

A: wysłanie pierwszego segmentu z danymi:
ACK=1, SeqNo = x + 1, AckNo = y + 1, $a_1$ bajtów danych
B: wysłanie do A potwierdzenia i $b_1$ bajtów danych w tym samym segmencie:
ACK=1, SeqNo = y + 1, AckNo = x + 1 + $a_1$, $b_1$ bajtów danych
.
.
.

A: wysłanie ostatniego (o numerze m) segmentu z danymi, jednocześnie A potwierdza odebranie (n–1)-ego segmentu od B:
ACK=1, SeqNo = x +1 + ($a_1$+…+$a_{m-1}$), AckNo=y+1+($b_1$+…+$b_{n-1}$), $a_m$ bajtów danych

B: wysłanie do A n-tego segmentu z $b_n$ bajtami danych, oraz potwierdzenia odbioru m-tego segmentu od A:
ACK=1, SeqNo=y+1+($b_1$+…+$b_{n-1}$), AckNo= x+1+($a_1$+…+$a_m$), $b_n$ bajtów danych

**Uwaga:** na etapie utrzymywania połączenia numer potwierdzenia (AckNo) otrzymujemy dodając do numeru sekwencyjnego (SeqNo) segmentu potwierdzanego długość jego pola danych.

Obustronne zamykanie sesji

Zakładamy, że przed wymianą segmentów zamykających sesję A wysłał do B strumień danych o długości $d_A$=$a_1$+…+$a_m$, a B wysłał do A strumień danych o długości $d_B$=$b_1$+…+$b_n$

A: wysłanie segmentu zamykającego połączenie:
FIN=1, ACK=1, SeqNo=x+1+$d_A$, AckNo= y+1+$d_B$

B: wysłanie odpowiedzi na segment zamykający:
FIN=1, ACK=1, SeqNo=y+1+$d_B$, AckNo=x+2+$d_A$

A: wysłanie potwierdzenia odebrania powyższego segmentu od B:
ACK=1, SeqNo=x+2+$d_A$, AckNo=y+2+$d_B$

**Uwaga:** na etapie zamykania połączenia odebranie segmentu z ustawioną flagą FIN potwierdzamy wstawiając w pole AckNo numer sekwencyjny (SeqNo) segmentu potwierdzanego powiększony o 1.

**Ćwiczenie 1**: wykorzystując analizator sieciowy *ethereal* obejrzyj i poddaj analizie pakiety przesyłane między klientem a serwerem usługi *telnet* działającej w oparciu o protokół transportowy TCP. Zastosuj następujący filtr przechwytywania:

**host a and host b and tcp and port telnet**

jest to skrócony (ale mimo to równoważny) zapis następującego filtru

**(ip src host a and ip dst host b) or (ip src host b and ip dst host a) and (ip proto \tcp) and port telnet**

Ze względu na jednakowy priorytet operatorów and i or i obliczanie wartości wyrażeń od lewej do prawej strony, powyższe wyrażenie jest równoważne następującemu:

**((ip src host a and ip dst host b) or (ip src host b and ip dst host a)) and (ip proto \tcp) and port telnet**

Zwróć uwagę, że po otwarciu połączenia TCP następuje negocjacja parametrów właściwych dla protokołu *telnet*, następnie użytkownik jest wzywany do podania nazwy logowania i

hasła. Po wciśnięciu znaku na klawiaturze, jest on przesyłany w osobnym segmencie do serwera, serwer odpowiada odsyłając ten znak do klienta (echo), również w osobnym segmencie, następnie znak jest wypisywany na terminalu użytkownika. Na znaki hasła serwer nie odpowiada echem. Zarówno nazwa logowania jak i hasło są możliwe do odczytania.

**Ćwiczenie 2**: wykorzystując analizator sieciowy *ethereal* obejrzyj i poddaj analizie pakiety przesyłane między klientem a serwerem usługi *ftp* działającej w oparciu o protokół transportowy TCP. Zastosuj następujący filtr przechwytywania:

**host a and host b and tcp**

Zwróć uwagę na to, że podczas sesji otwierane są dwa połączenia – jedno sterujące, a drugie do przesyłania danych. Usługa ftp może działać w tzw. trybie aktywnym albo pasywnym. Do przełączania między trybami służy polecenie „passive". Zaobserwuj różnicę w działaniu obu trybów.

Tryb aktywny ftp:

1. Klient wysyła żądanie „PORT" zgłaszając numer swojego portu dla połączenia do przesyłania danych
2. Po zaakceptowaniu żądania „PORT" przez serwer, klient wysyła żądanie pobrania (RECV) lub wysłania (STOR) pliku
3. Serwer inicjuje połączenie do przesyłania danych. Dla tego połączenia port serwera ma numer 20 (ftp-data).

Tryb pasywny ftp:

4. Klient wysyła żądanie „PASV"
5. Serwer odpowiada zgłaszając numer „wysokiego portu". Będzie to port serwera dla połączenia do przesyłania danych
6. Klient inicjuje połączenie do przesyłania danych. Dla tego połączenia port klienta ma „wysoki numer", tj. większy od 1023.
7. Klient wysyła żądanie pobrania (RECV) lub wysłania (STOR) pliku.

Tryb pasywny stosowany jest zazwyczaj wtedy, kiedy serwer ftp znajduje się poza siecią, do której są przyłączeni klienci, natomiast sieć chroniona jest przez firewall blokujący wszystkie połączenia TCP inicjowane z zewnątrz. W trybie aktywnym połączenie do przesyłania danych inicjuje serwer ftp, a więc zostanie ono zablokowane przez firewall. Możliwym wyjściem z sytuacji jest zainicjowanie tego połączenia przez klienta, czyli przejście w tryb pasywny.