# Sieci komputerowe

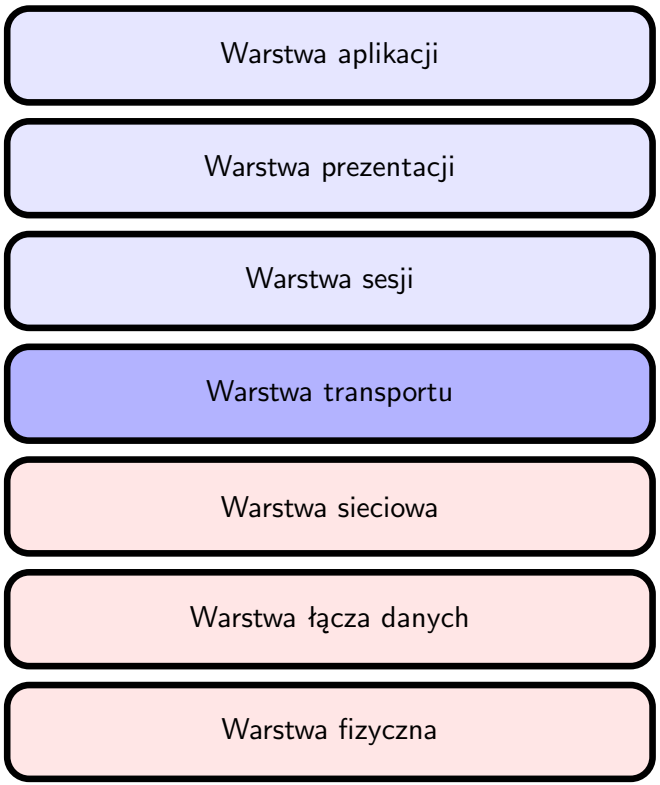
## Warstwa transportu

R. Wojciechowski [1] Ł. Sturgulewski [1] M. Bąkała [1]
A. Sierszeń [1] G. Nowak [1]

[1] Instytut Informatyki Stosowanej Politechniki Łódzkiej
http://www.iis.p.lodz.pl

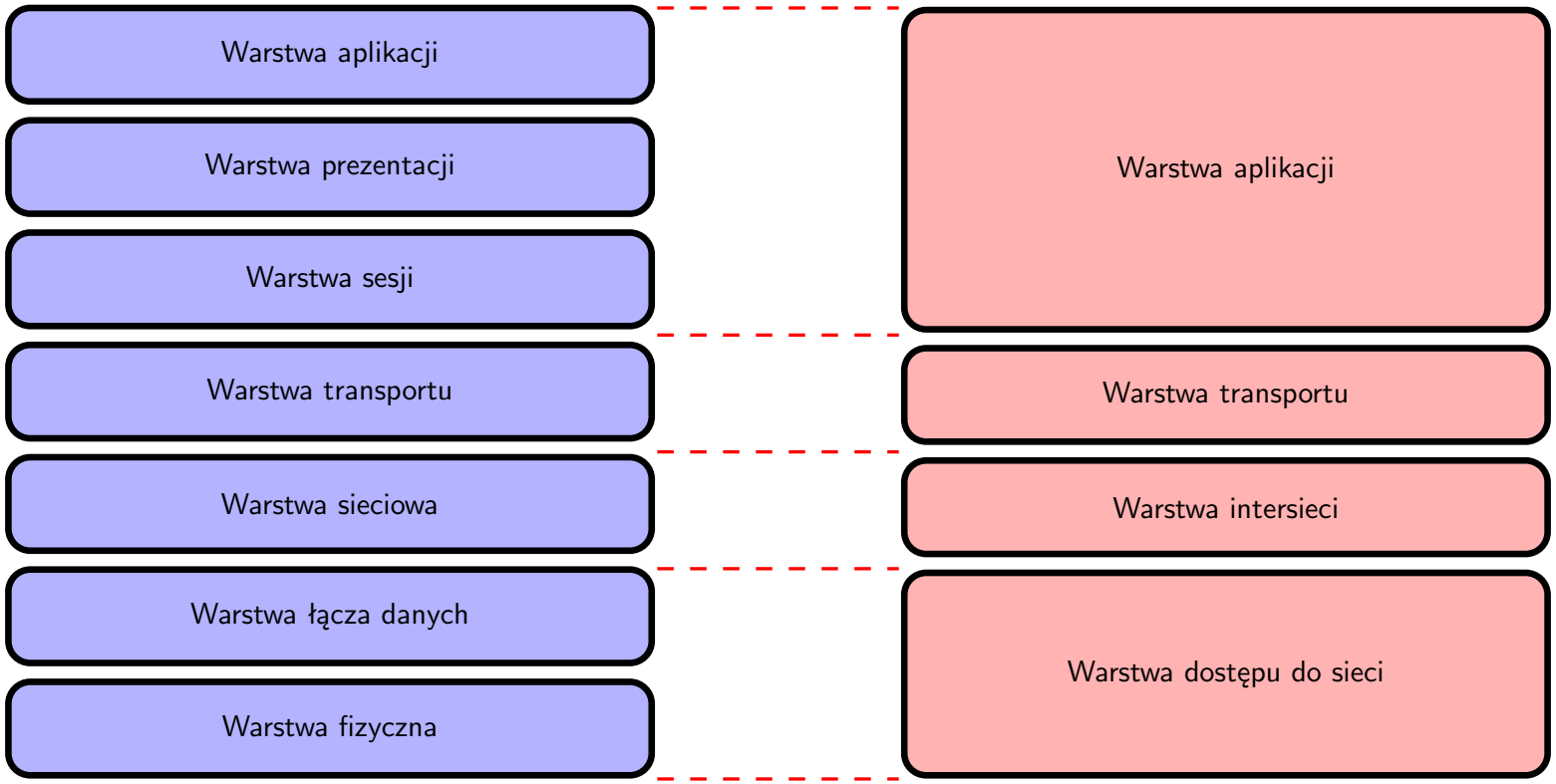
©2018, v2.0

# Warstwa transportu

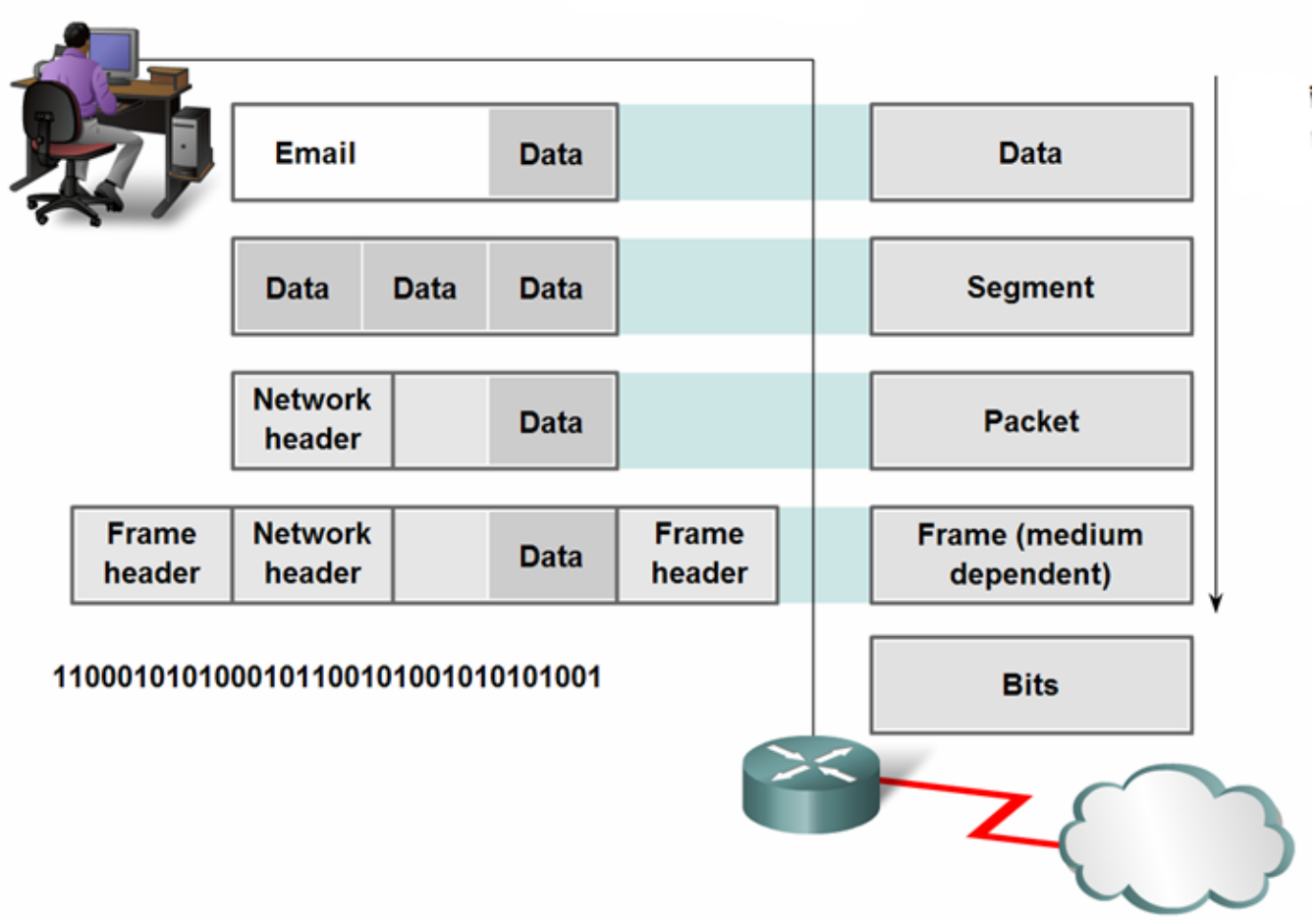



- Oddzielenie warstw fizycznych od aplikacyjnych
- Dostarczanie niezawodnych usług komunikacyjnych – ustanawianie, utrzymywanie i rozłączanie wirtualnych połączeń
- Podział informacji na mniejsze segmenty i składanie segmentów w całą, spójną informację
- Połączenie między stacjami

# Model TCP/IP vs OSI



# Enkapsulacja danych




Materiały szkoleniowe Cisco CCNA: `http://www.netacad.com`

# Zadania warstwy transportowej

- Warstwa transportu wprowadza przezroczysty transfer danych pomiędzy użytkownikami docelowymi wprowadzając mechanizmy zapewniające niezawodność transmisji
- Strumień danych warstwy transportowej jest logicznym połączeniem pomiędzy punktami w sieci
- Kontrola nad strumieniem jest realizowana przy udziale warstw niższych i obejmuje szereg mechanizmów segmentacji, potwierdzeń, przesuwnego okna, ...

# Zadania warstwy transportowej

- Realizacja funkcji transportu danych
- Niezawodne i precyzyjne regulowanie przepływu informacji ze źródła do celu
- *Quality of Service*
- Kontrola przepływu wspomagana przez niższe warstwy modelu OSI

# Identyfikator portu

- Zamysłem wprowadzenia identyfikatorów portów było umożliwienie transmisji współbieżnej informacji pochodzących z różnych strumieni (źródeł danych) poprzez mechanizmy znakowania usług i odpowiedniej synchronizacji dialogu komunikacyjnego

# Znakowanie usług

- Strumienie danych pochodzące od aplikacji klienckich i przekazywane do określonych usług zdalnych wykorzystują mechanizm portów do identyfikacji określonych sesji warstw wyższych
- Numeracja portów
  - 0 – 1023 $\rightarrow$ porty określone przydzielane przez IANA (*Internet Assigned Numbers Authority*)
  - 1024 – 65535 $\rightarrow$ porty dynamiczne
  - porty zarejestrowane dla określonych producentów (najczęściej również w zakresie 1024 – 65535)
- Identyfikacja toru nadawca $\longleftrightarrow$ odbiorca: *[Adres IP]* + *[Numer portu]*

# Numery portów

zły.kis.p.lodz.pl - PuTTY

```
#ident  "@(#)services   1.32    01/11/21 SMI"
#
#
# Copyright (c) 1999-2001 by Sun Microsystems, Inc.
# All rights reserved.
#
# Network services, Internet style
#
tcpmux          1/tcp
echo            7/tcp
echo            7/udp
discard         9/tcp           sink null
discard         9/udp           sink null
systat          11/tcp          users
daytime         13/tcp
daytime         13/udp
netstat         15/tcp
chargen         19/tcp          ttytst source
chargen         19/udp          ttytst source
ftp-data        20/tcp
ftp             21/tcp
ssh             22/tcp                          # Secure Shell
telnet          23/tcp
/etc/services
```

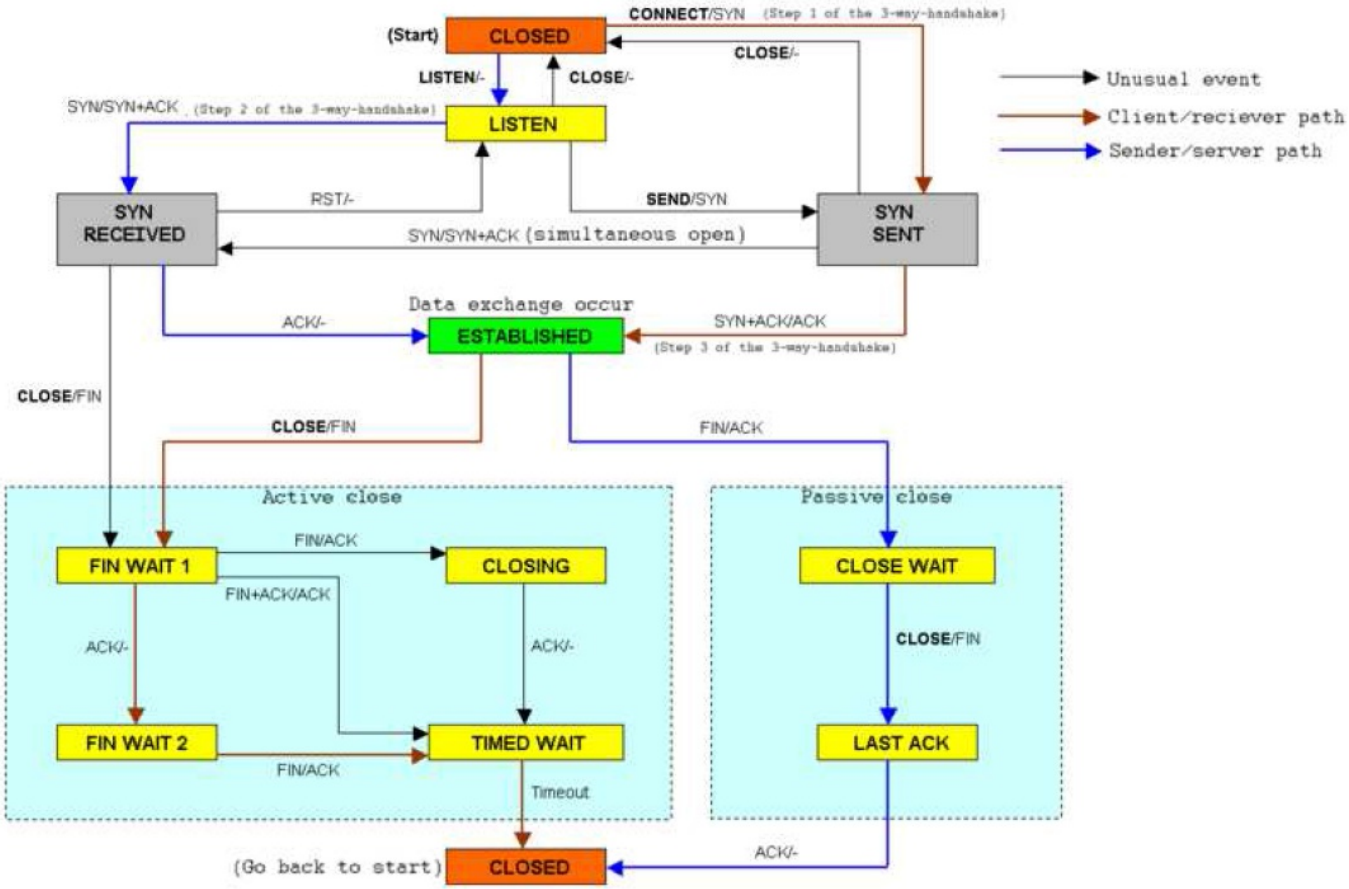
# Stany portów

- Podczas komunikacji pomiędzy usługami na konkretnych portach, wyszczególnia się określone stany portów w zależności od sytuacji w sesji
- Możliwe stany portów: LISTEN, SYN-SENT, SYN-RECEIVED, ESTABLISHED, FIN-WAIT-1, FIN-WAIT-2, CLOSE-WAIT, CLOSING, LAST-ACK, TIME-WAIT, CLOSED

# Stany portów

- LISTEN - stan nasłuchiwania, oczekiwanie na połączenie ze zdalnego hosta ($\rightarrow$ typowy stan w aplikacjach serwerowych)
- SYN-SENT - stan oczekiwania na odpowiedź zdalnej strony z ustawionymi flagami SYN oraz ACK (wysłano SYN, *3-way handshake*)
- SYN-RECEIVED - stan oczekiwania na odpowiedź zdalnej strony z potwierdzeniem ACK (wysłano SYN i ACK, *3-way handshake*)
- ESTABLISHED - stan gotowości do wymiany danych (sesja zestawiona)

# Stany portów



# Protokół zorientowany połączeniowo

## Protokół zorientowany połączeniowo

**Protokół zorientowany połączeniowo** - mechanizm tranmisji danych zakładający ustanowienie połączenia (negocjację ustawień połączenia) przed właściwą wymianą danych. Protokoły zorientowane połączeniowo zalicza się do grupy protokołów niezawodnych, gdyż gwarantują otrzymanie danych przez stronę zdalną we właściwej sekwencji

Materiały szkoleniowe Cisco CCNA: `http://www.netacad.com`

# Protokół bezpołączeniowy

## Protokół bezpołączeniowy

**Protokół bezpołączeniowy** - mechanizm tranmisji danych umożliwiający wysłanie danych bez wcześniejszego ustanowienia połączenia (negocjacji ustawień). Protokoły bezpołączeniowe zalicza się do grupy protokołów zawodnych, gdyż nie gwarantują otrzymania danych przez stronę zdalną (brak mechanizmów potwierdzeń)

Materiały szkoleniowe Cisco CCNA: `http://www.netacad.com`

# Protokoły warstwy transportu

- TCP (*Transmission Control Protocol*)
- UDP (*User Datagram Protocol*)
- SCTP (*Stream Control Transmission Protocol*)

# Protokół TCP, *Transmission Control Protocol*

- Protokół zorientowany połączeniowo (wirtualne obwody) $\rightarrow$ niezawodność
- Segementyzacja danych / konsolidacja segmentów
- Retransmisja wszystkiego, co nie zostało odebrane

# Nagłówek prokołu TCP

| + | Bits 0–3 | 4–9 | 10–15 | 16–31 |
|---|---|---|---|---|
| 0 | Source Port | | | Destination Port |
| 32 | Sequence Number | | | |
| 64 | Acknowledgment Number | | | |
| 96 | Data Offset | Reserved | Flags | Window |
| 128 | Checksum | | | Urgent Pointer |
| 160 | Options (optional) | | | |
| 160/192+ | Data | | | |

# Nagłówek prokołu IP oraz TCP

| + | Bits 0–3 | 4–7 | 8–9 | 10–15 | 16–31 |
|---|---|---|---|---|---|
| 0 | Source address | | | | |
| 32 | Destination address | | | | |
| 64 | Zeros | | Protocol | | TCP length |
| 96 | Source Port | | | | Destination Port |
| 128 | Sequence Number | | | | |
| 160 | Acknowledgement Number | | | | |
| 192 | Data Offset | Reserved | | Flags | Window |
| 224 | Checksum | | | | Urgent Pointer |
| 256 | Options (optional) | | | | |
| 256/288+ | Data | | | | |

# Nagłówek prokołu TCP, flagi

- URG – informuje o istotności pola priorytetu
- ACK – informuje o istotności pola numeru potwierdzenia
- PSH – wymusza przesłanie pakietu
- RST – resetuje połączenie (wymagane ponowne uzgodnienie sekwencji)
- SYN – synchronizuje kolejne numery sekwencyjne
- FIN – oznacza zakończenie przekazu danych
- NS, CWR, ECE – obsługa przeciążeń

# Kontrola przepływu

- Dostosowanie tempa transmisji poprzez wprowadzenie mechanizmów potwierdzeń umożliwiających spowolnienie tempa nadawania
- Transmisja określonych sekwencji danych poprzez wprowadzenie numerów sekwencyjnych i uwzględnienie ich w numerach potwierdzeń
- Wprowadzenie metod korekcji błędów - definicje sposobów wykrywania błędów transmisji (*Error Detection* $\rightarrow$ CRC16, CRC32, ...) oraz reakcji na nie (*Error Control* $\rightarrow$ ARQ)

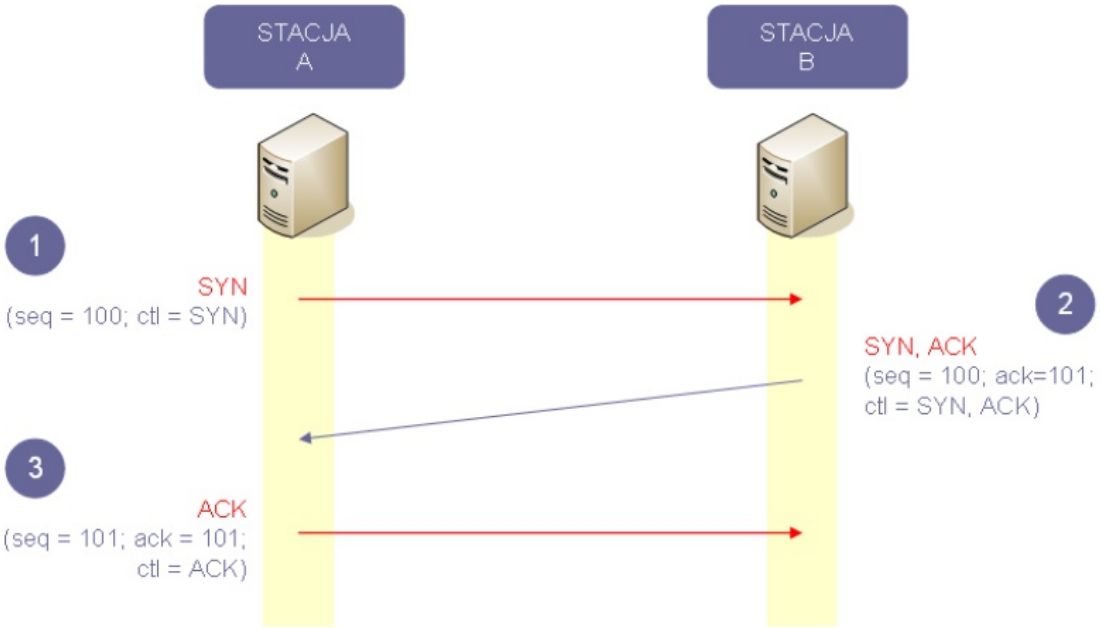
# Metody kontroli przepływu TCP

- Potrójny uścisk ręki / Nawiązanie połączenia
- Wprowadzenie numerów sekwencyjnych transmitowanych danych
- Mechanizm potwierdzeń TCP
- Mechanizm przesuwnego okna

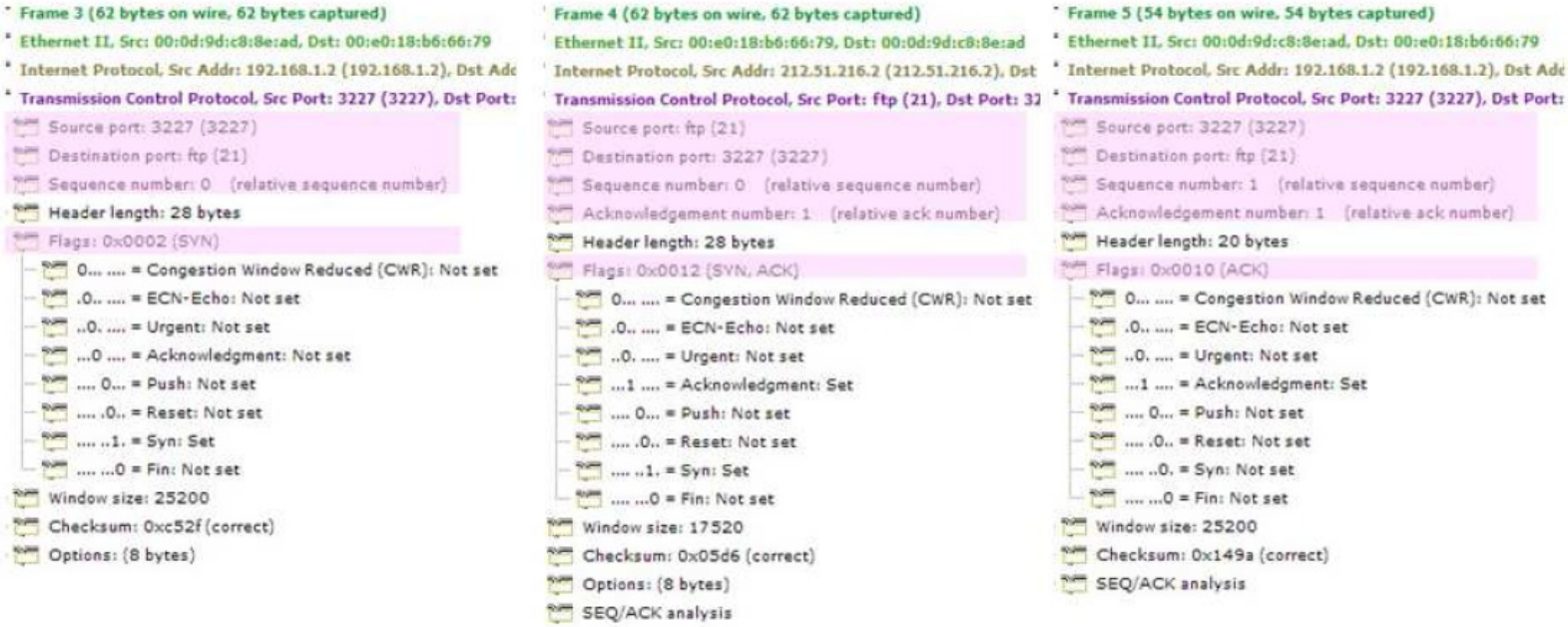
# Potrójny uścisk ręki

- Procedura inicjalizacyjna przeprowadzana w celu synchronizacji początkowych numerów sekwencyjnych ramek nadawcy i odbiorcy
- Kontrola przepływu realizowana na podstawie numerów sekwencyjnych ramek
- Konieczność synchronizacji - brak powiązania między numerami sekwencyjnymi a innymi czynnikami mogącymi mieć wpływ na dobór numeru początkowego

# Potrójny uścisk ręki



# Potrójny uścisk ręki



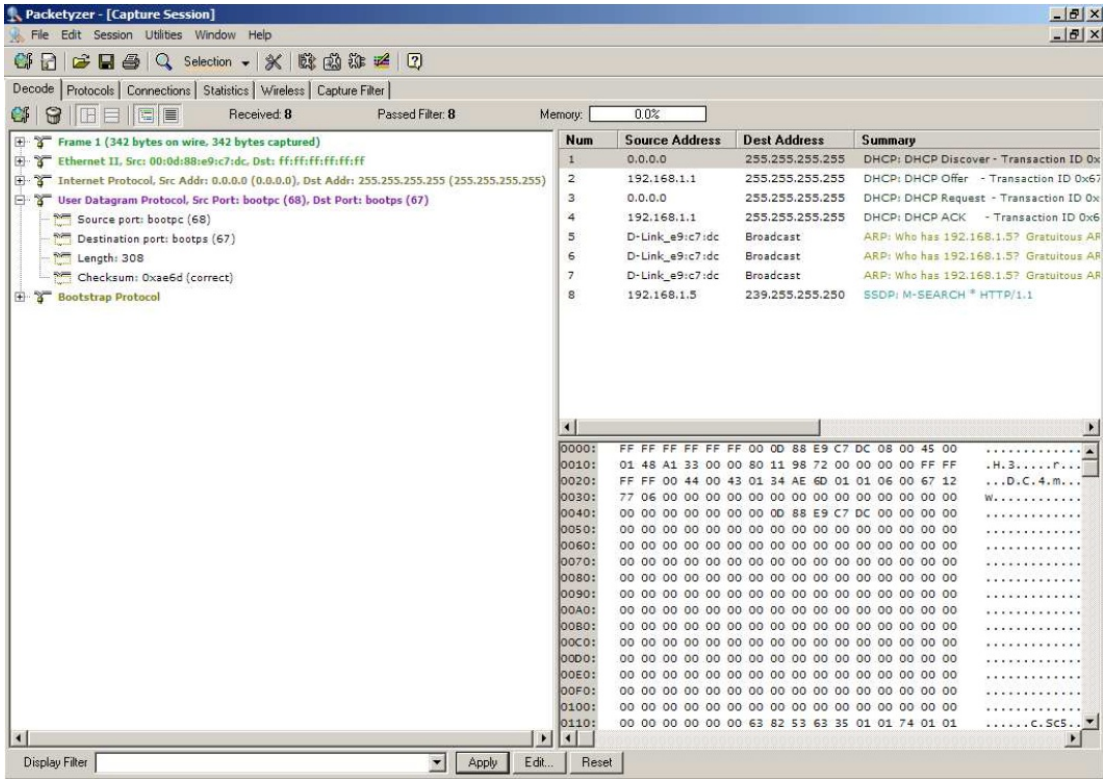
# TCP, wymiana danych

- Bezbłędny transfer danych
- Transfer oczekiwanych danych (z zachowaniem kolejności)
- Retransmisja zagubionych pakietów
- Ignorowanie zduplikowanych pakietów
- Zapobieganie przeciążeniom

# Protokół UDP, *User Datagram Protocol*

- Protokół bezpołączeniowy $\rightarrow$ zawodność
- Transmisja datagramów bez podziału ani konsolidacji przepływających informacji
- Brak mechanizmów sprawdzenia poprawności otrzymania informacji i kontroli przepływu ($\rightarrow$ niezawodność może być realizowana przez protokoły warstw wyższych)

# Nagłówek prokołu UDP



# Porównanie protokołów warstwy transportu

| | **TCP** | **UDP** |
|---|---|---|
| Rozmiar nagłówka | 20 bajtów | 8 bajtów |
| Treść pakietu | Segment | Datagram |
| Numer portu | Tak | Tak |
| Zorientowany połączeniowo | Tak | Nie |
| ARQ | Tak | Nie |
| Numeracja sekw. segmentu | Tak | Nie |
| Kontrola przepływu | Tak | Nie |

# TCP i UDP

Zastanów się, jakie następstwa wynikają z korzystania z TCP, a jakie z UDP.

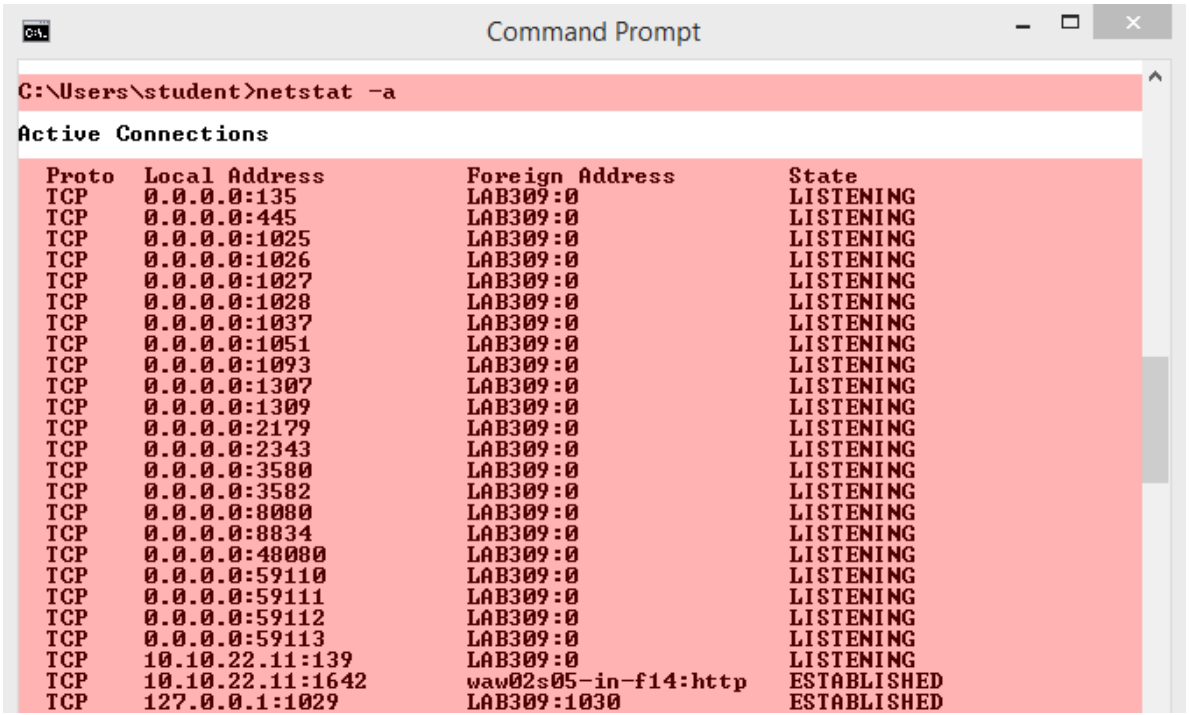
# Warstwa aplikacji i transportu

Zastanów się, które z poznanych protokołów warstwy aplikacji korzystają z TCP, które z UDP.

# Weryfikacja portów komunikacyjnych

- Weryfikacja portów komunikacyjnych

  `Windows -> cmd -> netstat -a`



# Metody skanowania portów

- TCP – connect
- TCP SYN
- TCP FIN
- TCP ACK
- ...

# TCP - connect

- Test polega na przeprowadzeniu połączenia TCP: intruz wysyła na określony port pakiet SYN. Jeśli skanowany host odpowie pakietem SYN/ACK oznacza to, że port jest otwarty, jeśli RST/ACK – port jest zamknięty. Jeśli połączenie zostało nawiązane intruz zamyka je pakietem ACK
- Skanowanie TCP – connect jest łatwe do wykrycia

# TCP SYN

- Modyfikacja TCP – connecta
- Polega na przeprowadzeniu połączenia TCP: intruz wysyła na określony port pakiet SYN. Jeśli skanowany host odpowie pakietem SYN/ACK oznacza to, że port jest otwarty, jeśli RST/ACK – port jest zamknięty. Jesli połączenie zostało nawiązane intruz go NIE zamyka - połączenie nie trafia do logów systemowych
- Skanowanie TCP SYN jest trudniejsze do wykrycia niż TCP - connect

# TCP FIN

- Technika ukrytego skanowania (*stealth scanning*)
- Celem jest ominięcie reguł filtrowania i ukrycie faktu skanowania
- Wykorzystanie własności stosu TCP polegającej na tym, że zamknięty port powinien na każdy błędny pakiet odpowiedzieć pakietem RST, otwarty zignoruje pakiet FIN
- Aby wykryć skanowanie FIN należy zastosować narzędzia IDS

# Skanowanie sieci z wykorzystaniem nmap

- Różne typy skanowania (TCP SYN, Connect, ACK)

```
nmap [-sS | -sT | -sA] (...)
```

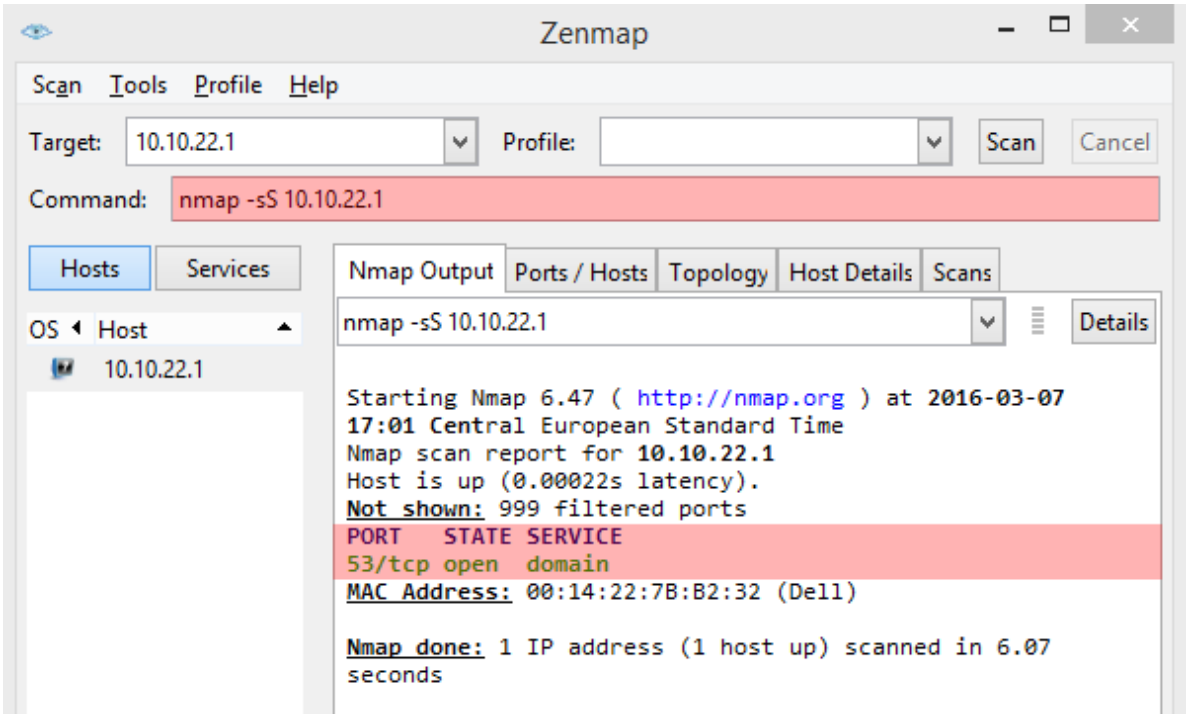
- Ograniczenie zakresu portów poddanych skanowaniu

```
nmap -p [Zakres portów] (...)
nmap -p 22, -p 1-10, -p U:53,T:100-200 (...)
```

# Skanowanie sieci z wykorzystaniem nmap

- Skanowanie portów określonego hosta

  `nmap -sS 10.10.22.1`



# Koniec

Dziękuję za uwagę . . .