



Sieci komputerowe

Wykład 4
28.10.2019

dr inż. Łukasz Graczykowski
lukasz.graczykowski@pw.edu.pl

Semestr zimowy 2020/2021

Jak zbudowany jest Internet?

- Internet jest siecią rozproszoną wykorzystującą protokół IP (oraz TCP i UDP – o nich za chwilę) oraz DNS do przydzielania nazw
- Jest niezależna od fizycznego sposobu realizacji łącza (użytych kabli, sieci radiowych, etc.)
- Najważniejszym urządzeniem w Internecie (obok komputerów-klientów :)) jest router
 - jak już wiemy – router to urządzenie w warstwie Internetu, działające na protokole IP, przekazujące ruch pomiędzy dwoma sieciami
 - routery dzielimy na
 - routery na brzegach (**edge routers**) – bezpośrednio połączone z sieciami (klientami) docelowymi oraz jednym routerem rdzenia
 - routery rdzenia (**core routers**) – połączone ze sobą oraz z core routers, tworzą szkielet sieci (**backbone**)

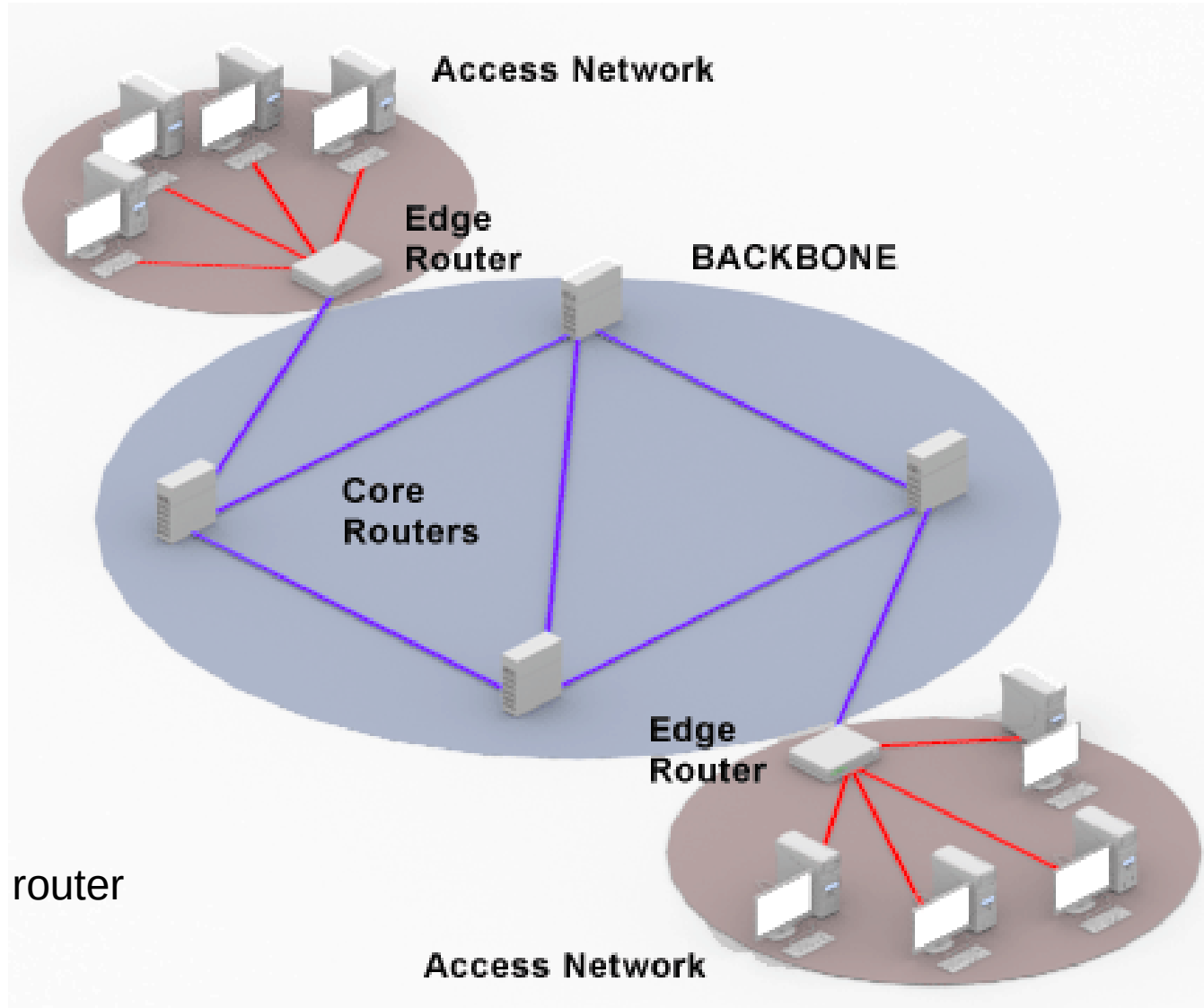
Jak zbudowany jest Internet?

- Core routers to urządzenia bardzo drogie, działające z maksymalną możliwą prędkością przesyłania pakietów
- Kilku producentów np. CISCO, HUAWEI

core router



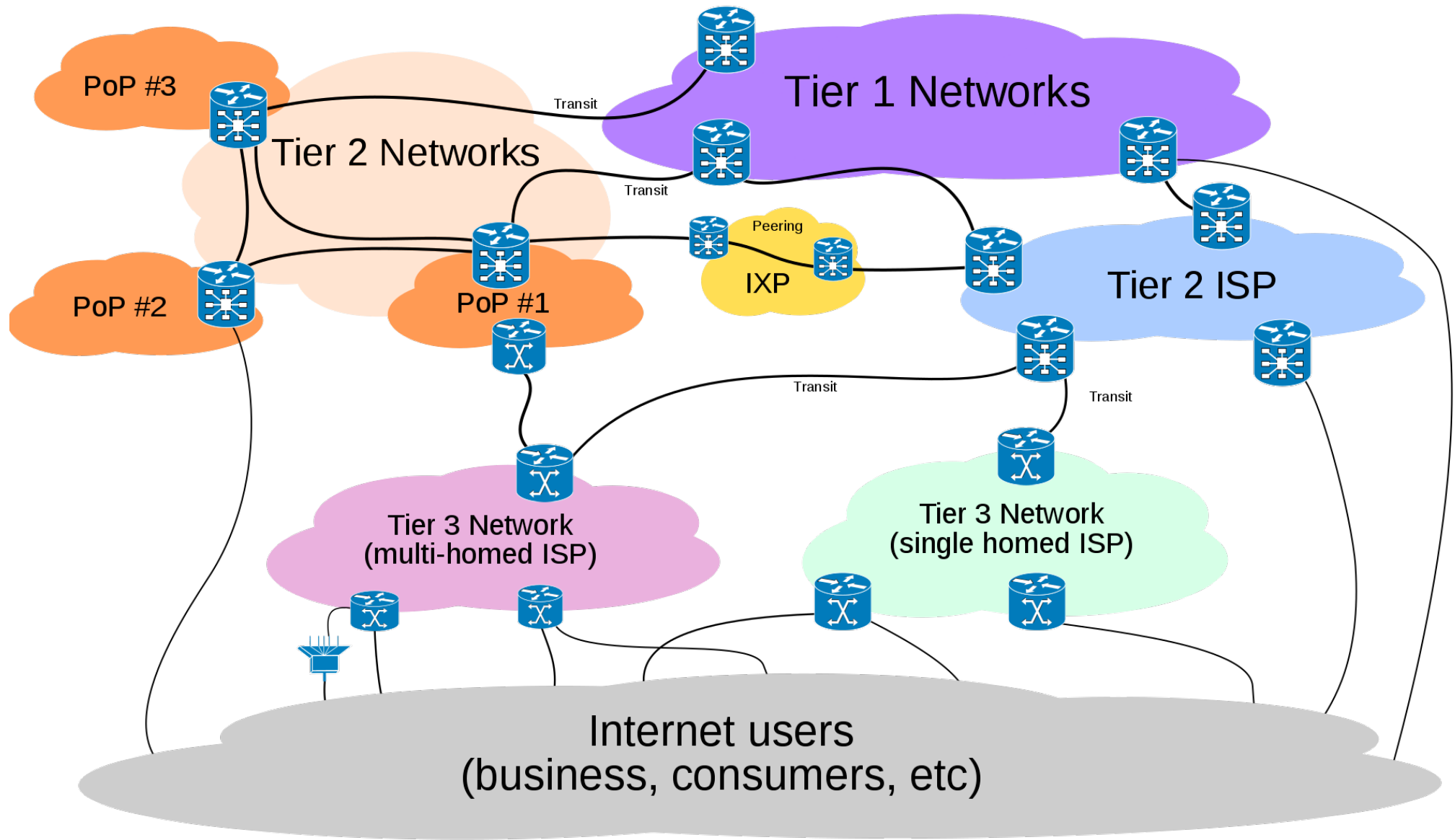
edge router



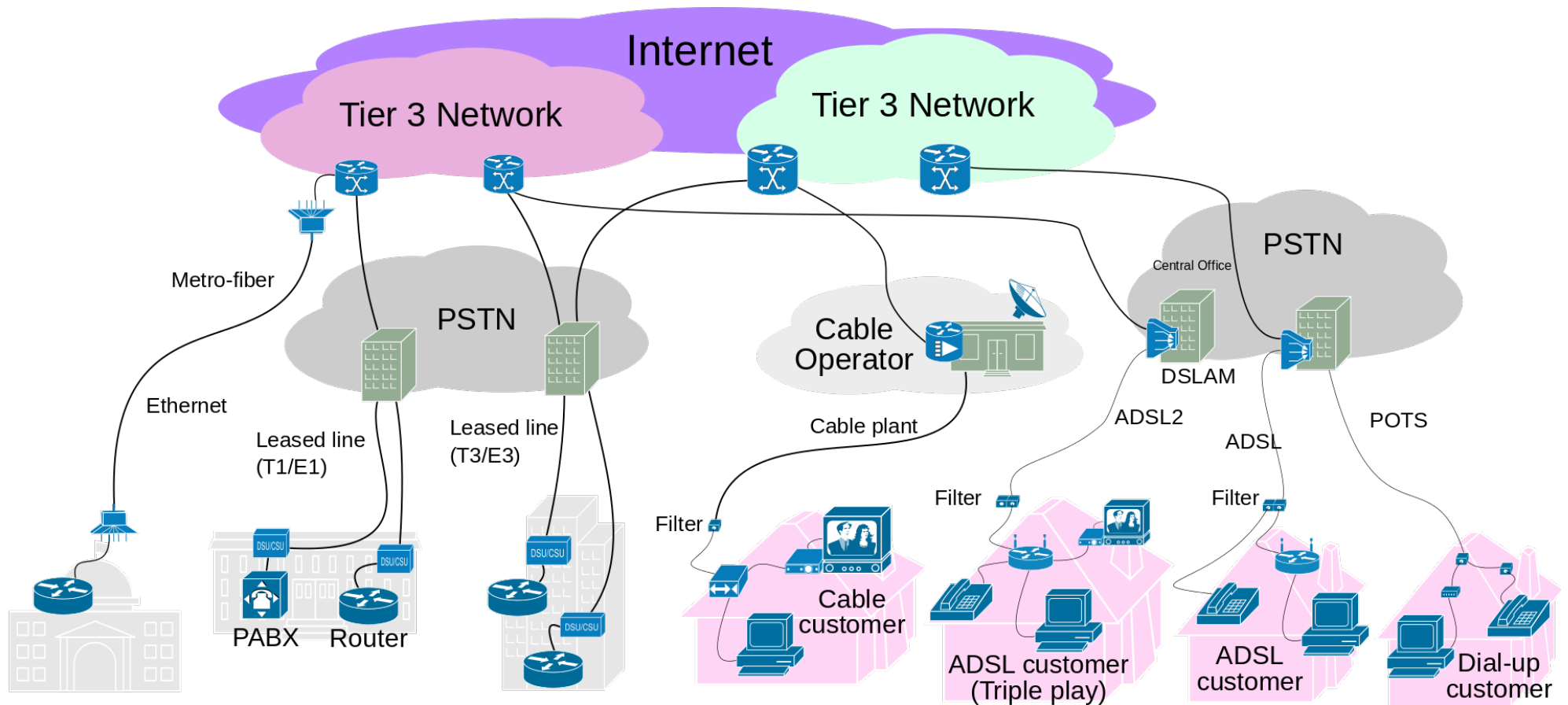
Jak zbudowany jest Internet?

- Internet tworzą tzw. **Systemy Autonomiczne (AS)**
 - zwykle zarządzane przez jedną organizację
 - mającą własną sieć szkieletową (wiele routerów) i systemy klienckie
- Dostawca Internetu (**ISP – *Internet Service Provider***) to taki AS, który jest podłączony do innych sieci i może pełnić również rolę tranzytową (połączony jest z wieloma sieciami klientów oraz z innymi ISP, umożliwia pełną komunikację w Internecie)
- ISP grupuje się w kategorie (**tiers**):
 - np. Tier 1 – połączone każda ze sobą (za pomocą odpowiednich umów), główny szkielet Internetu
 - Tier 2 – musi wykupywać dostęp do części (np. aby dostać się do innego Tier 2 przez kilka Tier 1)
 - Tier 3 – z reguły to z czym komunikujemy się z domu (nasz dostawca Internetu)
 - najlepiej to zobrazować na obrazkach (następne dwa slajdy)

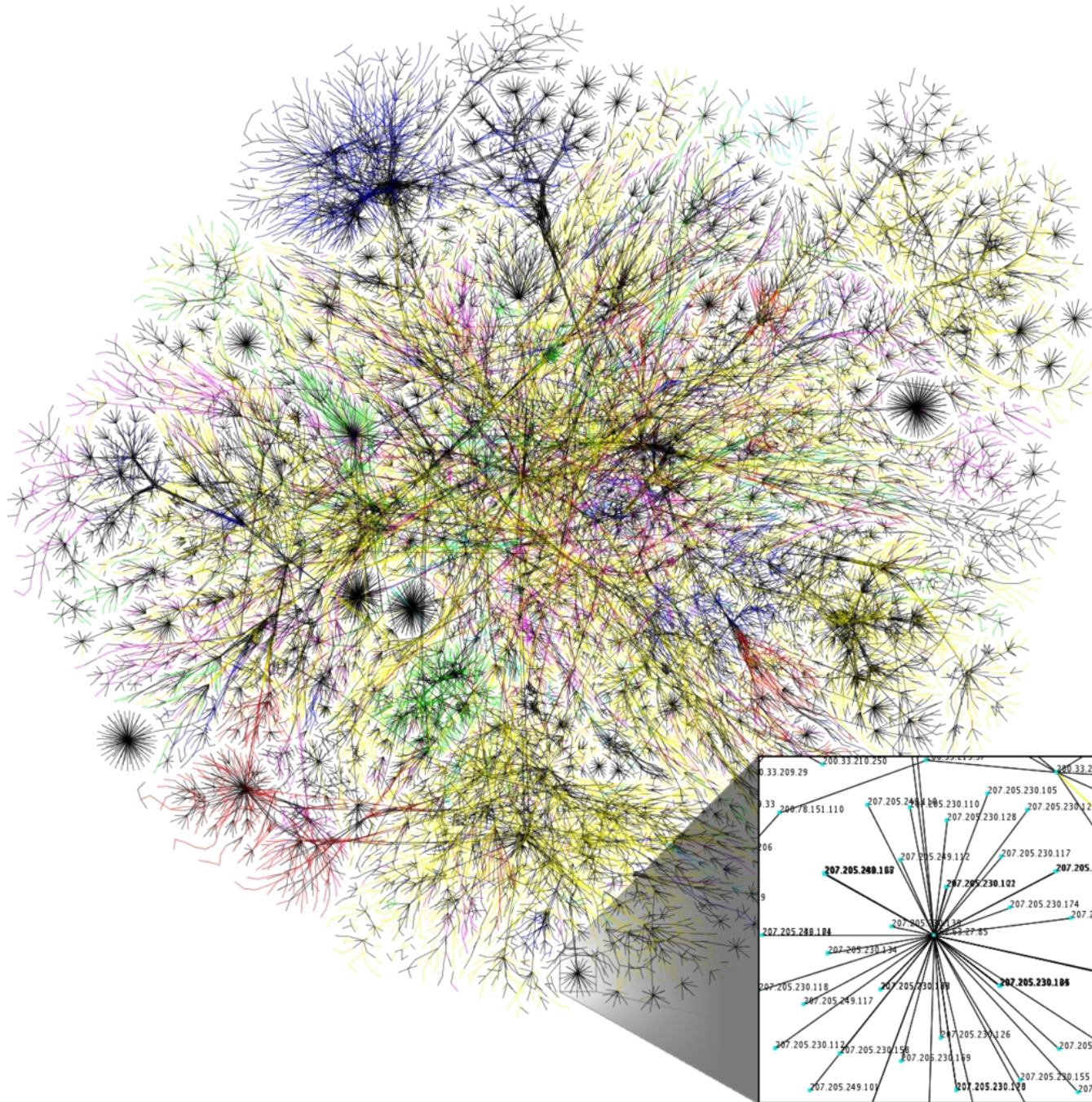
Jak zbudowany jest Internet?



Jak zbudowany jest Internet?



Jak zbudowany jest Internet?



Jak zbudowany jest Internet?

- Tier 1 ISP (oraz czasami Tier 2) komunikują się za pomocą **Internet exchange points (IX, albo IXP)**
- Są to dedykowane urządzenia (switchy), które umożliwiają połączenia pomiędzy różnymi ISP
- Pierwszym w Europie IXP był **CERN Internet Exchange Point** (zlokalizowany w CERN w Genewie) – działający do dzisiaj

<https://cixp.net/>



Members Of CIXP

Internet Service Providers

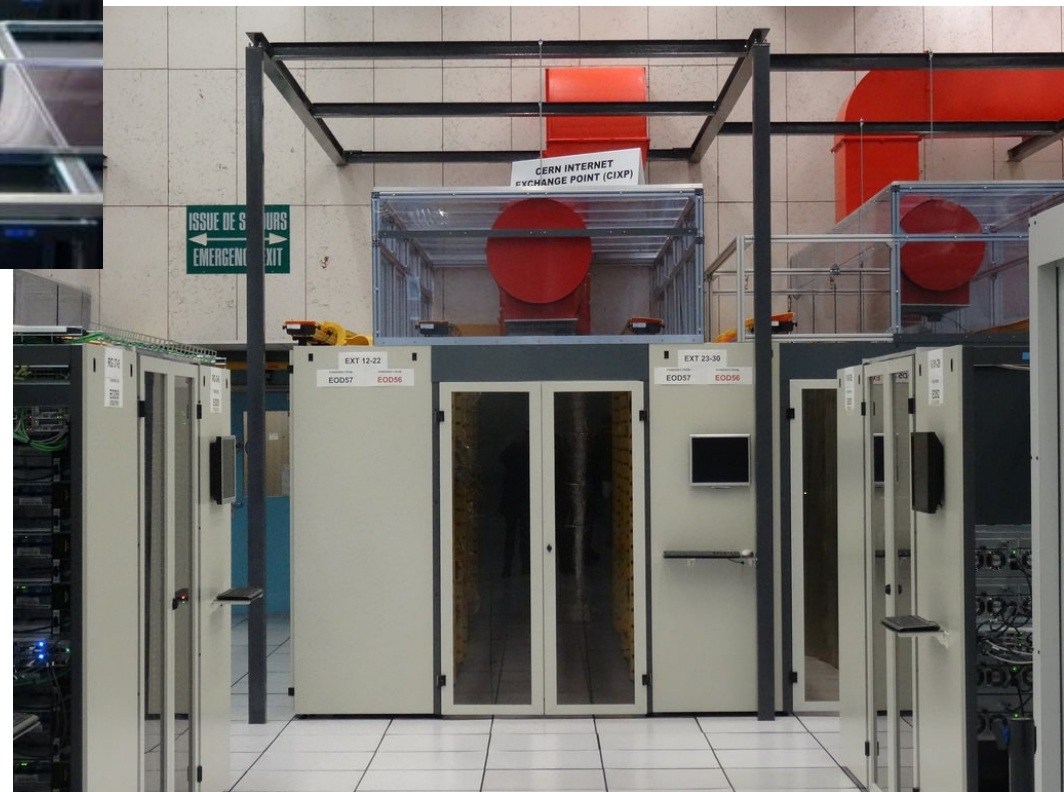
Participant	Located at
Adeli	CERN Geneva
Akenes	Equinix Geneva (GV2)
BIT (Swiss Confederation)	CERN Geneva
Briod SA	Equinix Geneva (GV1)
CERN	CERN Geneva
Cogent	Equinix Geneva (GV1)
COLT	CERN Geneva
CTI	CERN Geneva
DFI Services	Equinix Geneva (GV1)
Equinix	Equinix Geneva (GV1)
euNetworks (was Fibrelac)	CERN Geneva
Forcepoint	Equinix Geneva (GV1)
Hurricane Electric	Equinix Geneva (GV1)
I-root	Equinix Geneva (GV1)
Infomaniak Network SA	CERN Geneva
Infomaniak Network SA	Equinix Geneva (GV1)
Init7	CERN Geneva
IP-MAN (SIG)	Equinix Geneva (GV1)
IP-Max	CERN Geneva
IP-Max	Equinix Geneva (GV1)
IP-Max (route collector)	Equinix Geneva (GV1)
IXreach	Equinix Geneva (GV1)
Jaguar Network	Equinix Geneva (GV1)
K-Net / K-Sys	CERN Geneva
K-root (RIPE)	CERN Geneva
KPN Eurorings BV	CERN Geneva
Media-Lite SA	CERN Geneva
Netplus	Equinix Geneva (GV1)
Orange Business Services - GIBN	CERN Geneva
RIPE-RIS	CERN Geneva
SIMA Lausanne	Equinix Geneva (GV1)
Sunrise (TDC Switzerland AG)	CERN Geneva
Swisscom / IP-Plus	CERN Geneva
T-Systems Schweiz AG (Deutsche Telekom)	CERN Geneva
The Net	Equinix Geneva (GV1)
UPC Cablecom	CERN Geneva
VTX Services SA	CERN Geneva
Zscaler	Equinix Geneva (GV1)

CERN Internet Exchange Point

- Poziom 0 (parter) CERN Data Center
- Szafy z czerwonymi kablami - CIXP

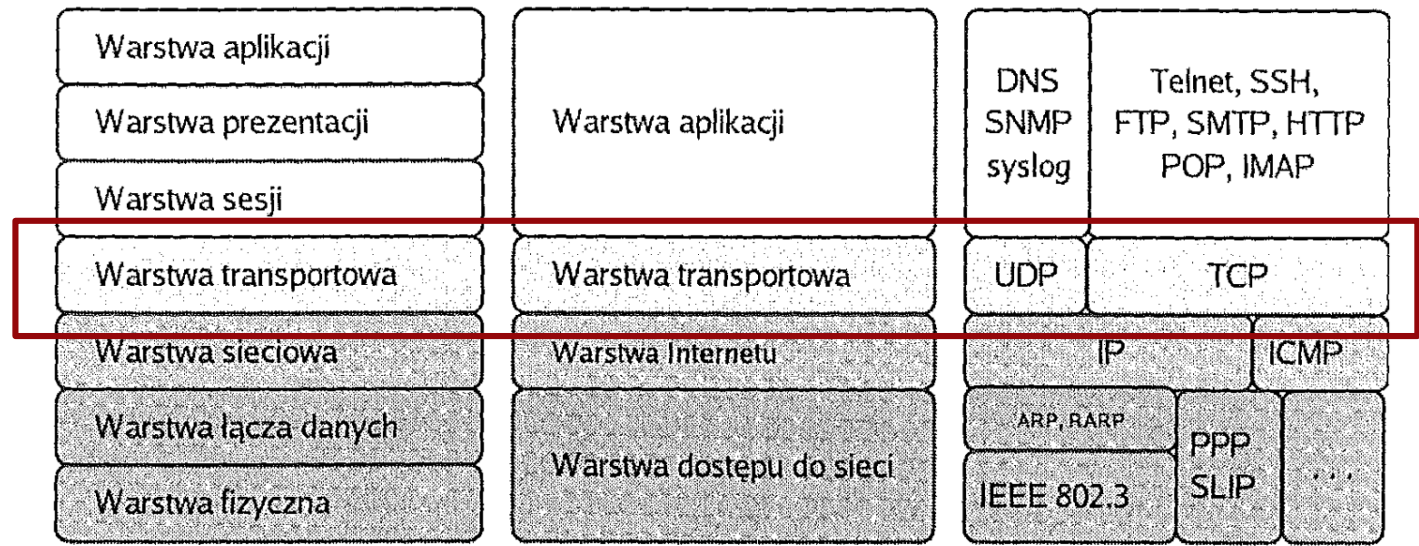


CERN Internet Exchange Point



Warstwa transportowa

Protokoły TCP i UDP



Model ISO/OSI

Model TCP/IP

Przykładowe protokoły

Warstwa transportowa

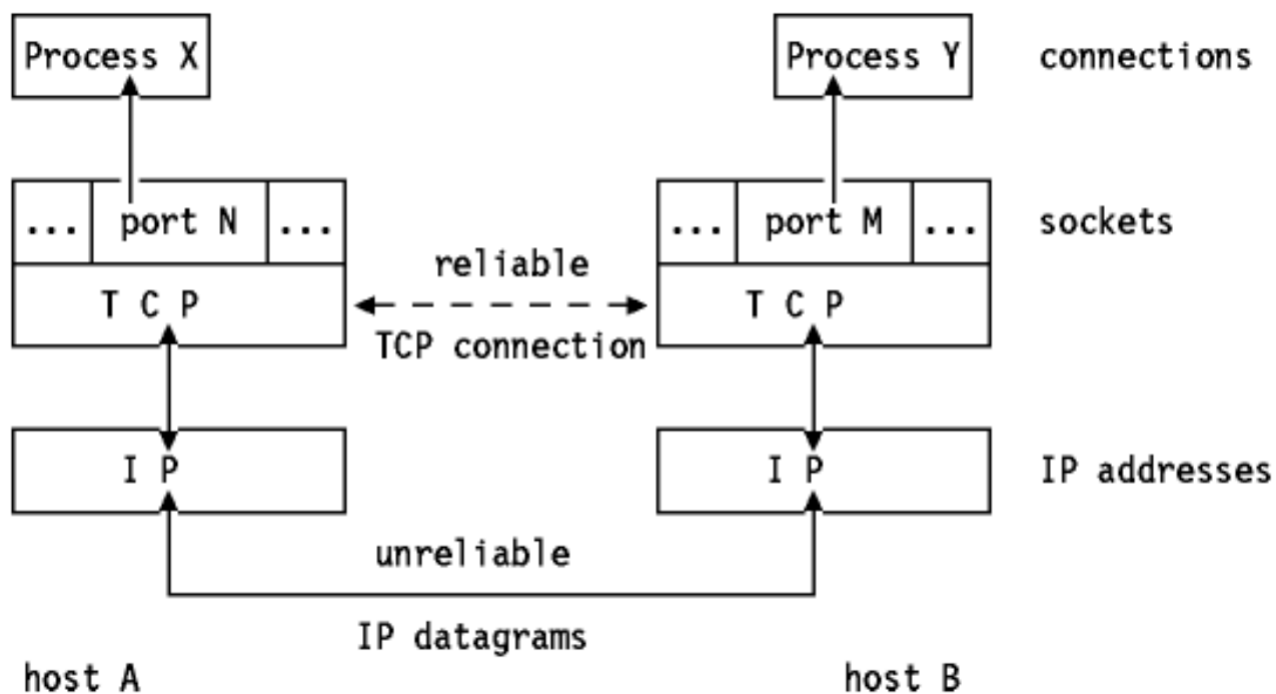
- Zadaniem warstwy transportowej jest **niezawodne** przesyłanie danych między urządzeniami
- Protokoły warstwy transportowej otrzymują dane z warstwy Internetu i rozdzielają je na poszczególne procesy w warstwie aplikacji (“ten fragment danych idzie do komunikatora, ten do e-mail’a a ten do przeglądarki”)
- Zawiera mechanizmy:
 - inicjalizowania, utrzymania, zamykania połączenia
 - sterowania przepływem danych
 - wykrywania błędów transmisji
- Istnieje wiele protokołów warstwy transportowej, ale najważniejsze są dwa:
 - TCP
 - UDP

Warstwa transportowa

- Typy komunikacji możemy klasyfikować następująco:
 - **połączeniowa** (*connection-oriented*) – nawiązanie połączenia przed wysłaniem właściwych danych
 - **bezpoleczeniowa** (*connectionless*) – bez sprawdzania, czy dane dotarły do odbiorcy (od razu wysyłamy właściwe dane)
 - **niezawodna** (*reliable*) – kontrola procesu przesyłania, retransmisja w przypadku niedostarczenia pakietu danych
 - **zawodna** (*unreliable*) – brak kontroli i retransmisji
 - **stanowa** (*stateful*) – sesja pomiędzy serwerem i klientem monitorowana przez serwer
 - **bezstanowa** (*stateless*) – brak monitorowania stanu sesji

Gniazdo sieciowe

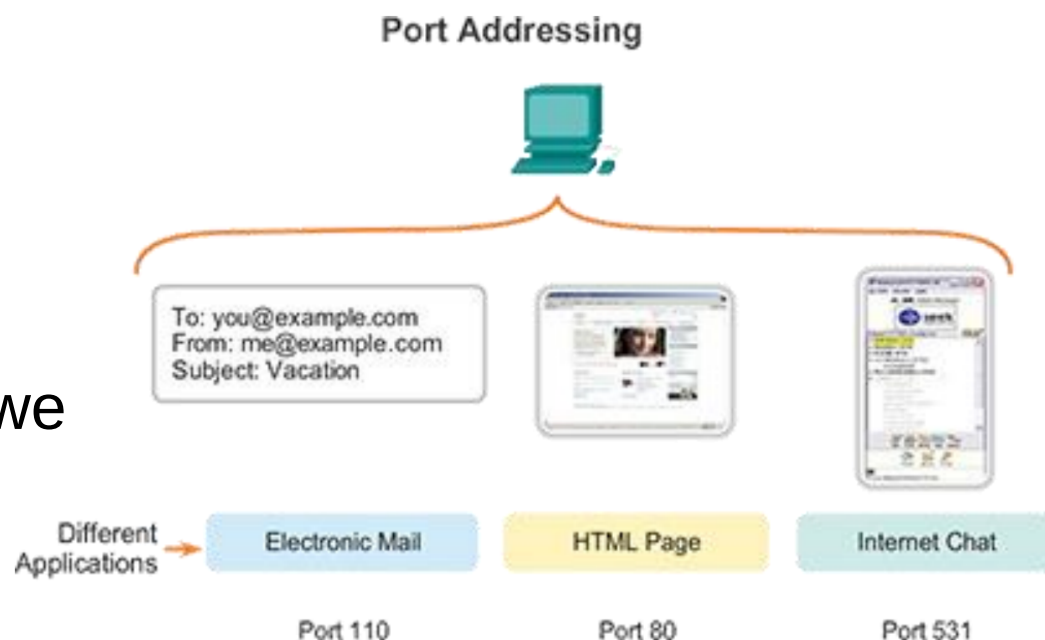
- Identyfikacja procesu, który ma odebrać daną porcję danych, odbywa się na podstawie numeru portu
 - **numer portu** jest 16-bitową liczbą związaną z danym typem komunikacji w sieci – przykładowo, serwer WWW odbierając zapytanie i następnie przesyłając stronę odbiorcy działa na porcie 80



Flug Flight	nach to	über via	planmäßig scheduled	voraus estimated	Gate	Check-In
LH 4916	Birmingham		15:20		639	Lufthansa
LH 714	Tokio		15:25		H28	Lufthansa
LH 1156	Münster/Osnabr.		15:25		609	Lufthansa
LH 1276	Köln/Bonn		15:25		601	Lufthansa
LH 474	Montreal/YUL		15:30		H38	Lufthansa
LH 3494	Zagreb		15:30		H35	Lufthansa
LH 3642	Graz		15:30		664	Lufthansa
LH 3938	Verona		15:30		661	Lufthansa
LH 4342	Bordeaux		15:30		666	Lufthansa
LH 4386	Toulouse		15:30		662	Lufthansa
LH 366	Bremen		15:35		631	Lufthansa
LH 052	Hamburg		15:40		632	Lufthansa
LH 979	Frankfurt/Main		15:40		630	Lufthansa
LH 3652	Bern		15:40		641	Lufthansa

Gniazdo sieciowe

- Identyfikacja procesu, który ma odebrać daną porcję danych, odbywa się na podstawie numeru portu
 - **numer portu** jest 16-bitową liczbą związaną z danym typem komunikacji w sieci – przykładowo, serwer WWW odbierając zapytanie i następnie przesyłając stronę odbiorcy działa na porcie 80
- **Gniazdo sieciowe** (*network socket*) to para liczb (numer IP oraz numer portu), które identyfikują zarówno odbiorcę jak i dany proces (aplikację)
 - zapis: 62.211.243.226:80 (IP:port)
 - zakres portów: 0 - 65 535
 - (opcjonalnie) gniazdo sieciowe może zawierać informację o protokole (np. TCP)



“Well-known ports”

- Istnieje lista zarezerwowanych portów (*well-known ports*), które są przypisane do różnych aplikacji (usług)
- Organizacja IANA prowadzi rejestr zarezerwowanych portów:

<https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml>

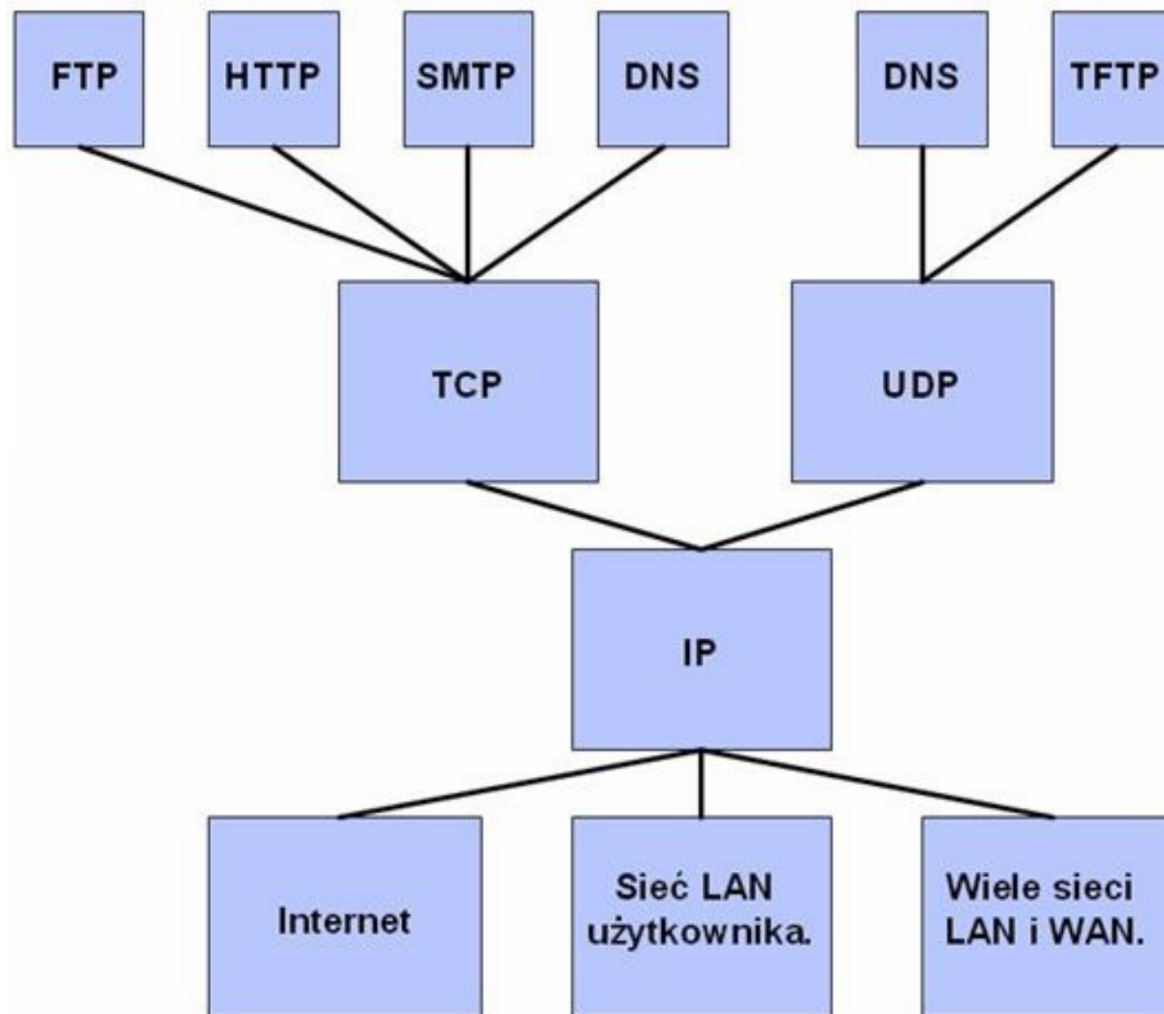
- Obecnie otwarte porty możemy sprawdzić poleceniem **netstat** (zarówno Windows i Linux)

```
C:\Users\opal>netstat -o
Active Connections
Proto Local Address Foreign Address State PID
TCP 127.0.0.1:1028 raft:5905 ESTABLISHED 1424
TCP 127.0.0.1:1029 raft:5905 ESTABLISHED 1424
TCP 127.0.0.1:5905 raft:1032 ESTABLISHED 1424
TCP 127.0.0.1:5905 raft:1033 ESTABLISHED 1424
TCP 192.168.1.101:1344 poczta:imaps ESTABLISHED 5876
TCP 192.168.1.101:1421 poczta:imaps ESTABLISHED 5876
TCP 192.168.1.101:1422 poczta:imaps ESTABLISHED 5876
TCP 192.168.1.101:1423 poczta:imaps ESTABLISHED 5876
TCP 192.168.1.101:3027 rmfstream3:8009 ESTABLISHED 5372
TCP 192.168.1.101:3116 wg-in-f189:https ESTABLISHED 5372
TCP 192.168.1.101:3286 www:http ESTABLISHED 5372
TCP 192.168.1.101:3306 poczta:imaps ESTABLISHED 5876
TCP 192.168.1.101:3327 74.125.133.106:https ESTABLISHED 5372
TCP 192.168.1.101:3332 host-213:https ESTABLISHED 5372
TCP 192.168.1.101:3349 74.125.133.95:http ESTABLISHED 5372
TCP 192.168.1.101:3351 74.125.206.95:http ESTABLISHED 5372
TCP 192.168.1.101:3352 74.125.206.95:http ESTABLISHED 5372
TCP 192.168.1.101:3359 wg-in-f94:http ESTABLISHED 5372
TCP 192.168.1.101:3360 wg-in-f94:http ESTABLISHED 5372
TCP 192.168.1.101:3361 wg-in-f101:https ESTABLISHED 5372
TCP 192.168.1.101:3362 wg-in-f94:http ESTABLISHED 5372
TCP 192.168.1.101:3363 wg-in-f94:http ESTABLISHED 5372
TCP 192.168.1.101:3376 wj-in-f84:https ESTABLISHED 5372
```

Protocol	Port	Protocol	Purpose
echo	7	TCP/UDP	Echo is a test protocol used to verify that two machines are able to connect by having one echo back the other's input.
discard	9	TCP/UDP	Discard is a less useful test protocol in which all data received by the server is ignored.
daytime	13	TCP/UDP	Provides an ASCII representation of the current time on the server.
FTP data	20	TCP	FTP uses two well-known ports. This port is used to transfer files.
FTP	21	TCP	This port is used to send FTP commands like put and get.
SSH	22	TCP	Used for encrypted, remote logins.
telnet	23	TCP	Used for interactive, remote command-line sessions.
smtp	25	TCP	The Simple Mail Transfer Protocol is used to send email between machines.
time	37	TCP/UDP	A time server returns the number of seconds that have elapsed on the server since midnight, January 1, 1900, as a four-byte, signed, big-endian integer.
whois	43	TCP	A simple directory service for Internet network administrators.
finger	79	TCP	A service that returns information about a user or users on the local system.
HTTP	80	TCP	The underlying protocol of the World Wide Web.
POP3	110	TCP	Post Office Protocol Version 3 is a protocol for the transfer of accumulated email from the host to sporadically connected clients.

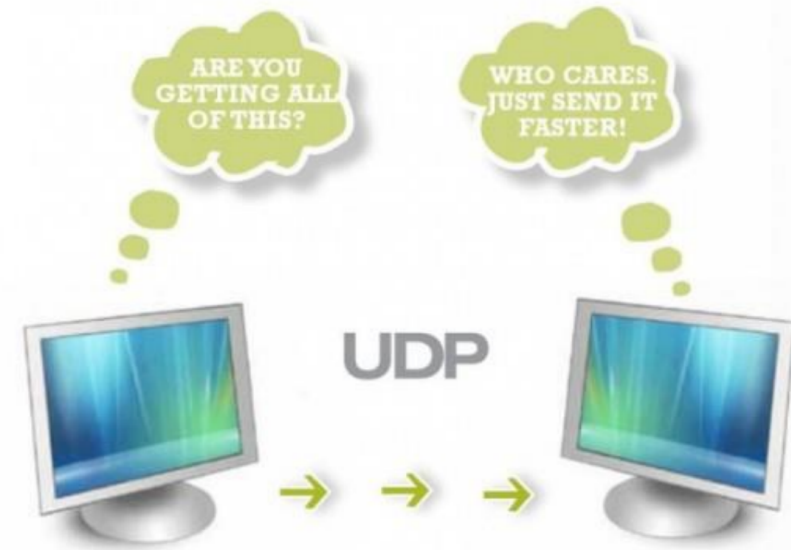
Protokoły TCP i UDP

- Zadaniem protokołów TCP i UDP jest dzielenie (łączenie) danych z warstw wyższych na segmenty oraz przekazywanie (odbieranie) ich z warstwy Internetu (protokołu IP)



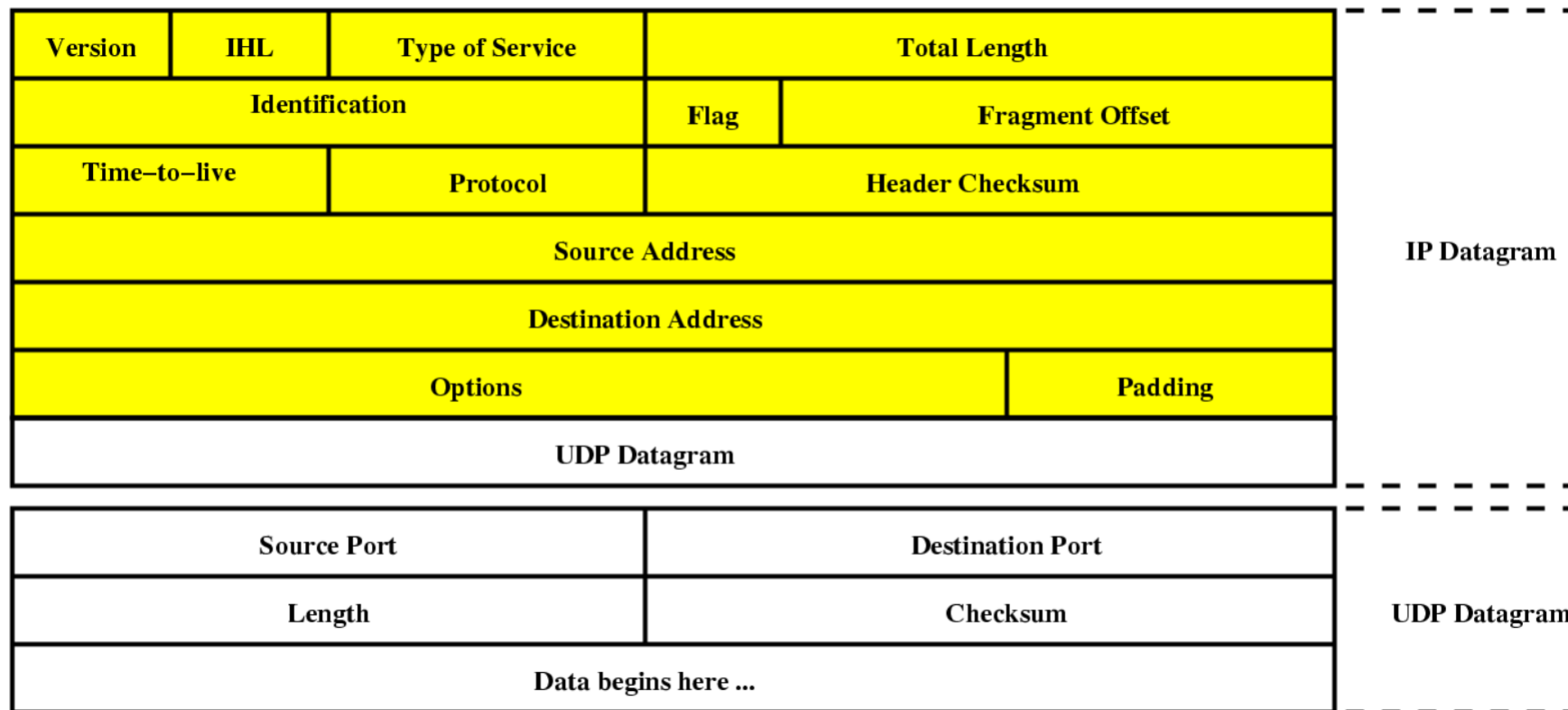
Protokół UDP

- **UDP** (*User Datagram Protocol*)
 - **bezpołączeniowy** – nie sprawdza gotowości odbiorcy do odbioru oraz tego, czy odbiorca faktycznie wiadomość odebrał
 - **brak kontroli przepływu informacji** (to mogą obsługiwać programy w warstwie aplikacji)
 - **datagram z krótkim nagłówkiem**
- **Po co go używać?**
 - większa szybkość
 - brak dodatkowych zadań adresata
 - obsługa multicast/broadcast



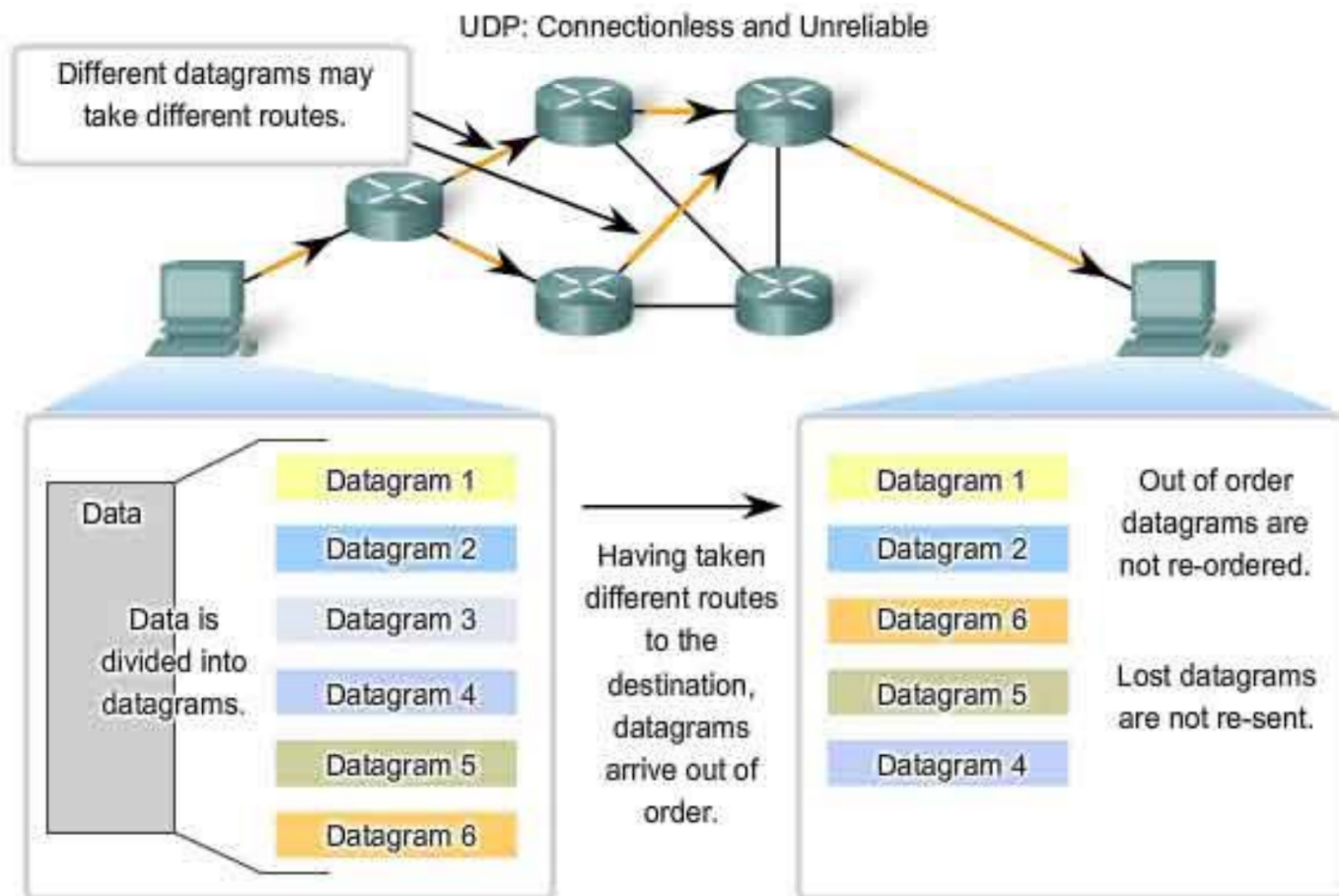
Protokół UDP

- Datagram UDP (jako część składowa datagramu IP)



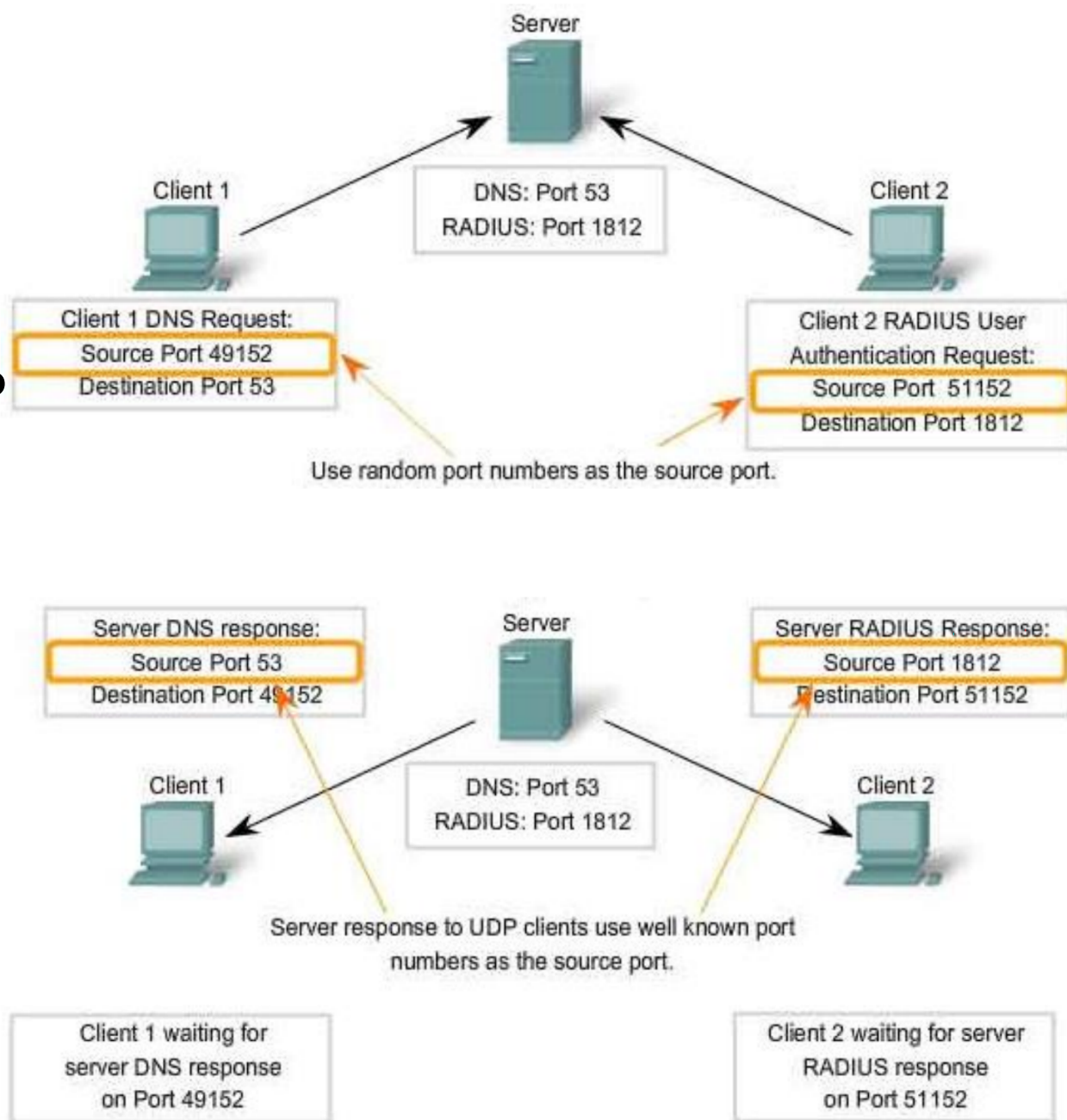
Protokół UDP

- Dane dzielone są na datagramy (“pakiety”)
- Datagramy docierają do adresata różną drogą (trasą) i mogą dotrzeć w różnej kolejności
- Brak możliwości retransmisji zgubionych datagramów



Protokół UDP

- W komunikacji klient-serwer używane są losowe porty nadawcy (source port) – ustalane dynamicznie
- Port serwera usługi UDP zawsze z puli “well-known ports”
- Losowe wybieranie portów zwiększa bezpieczeństwo



Protokół UDP

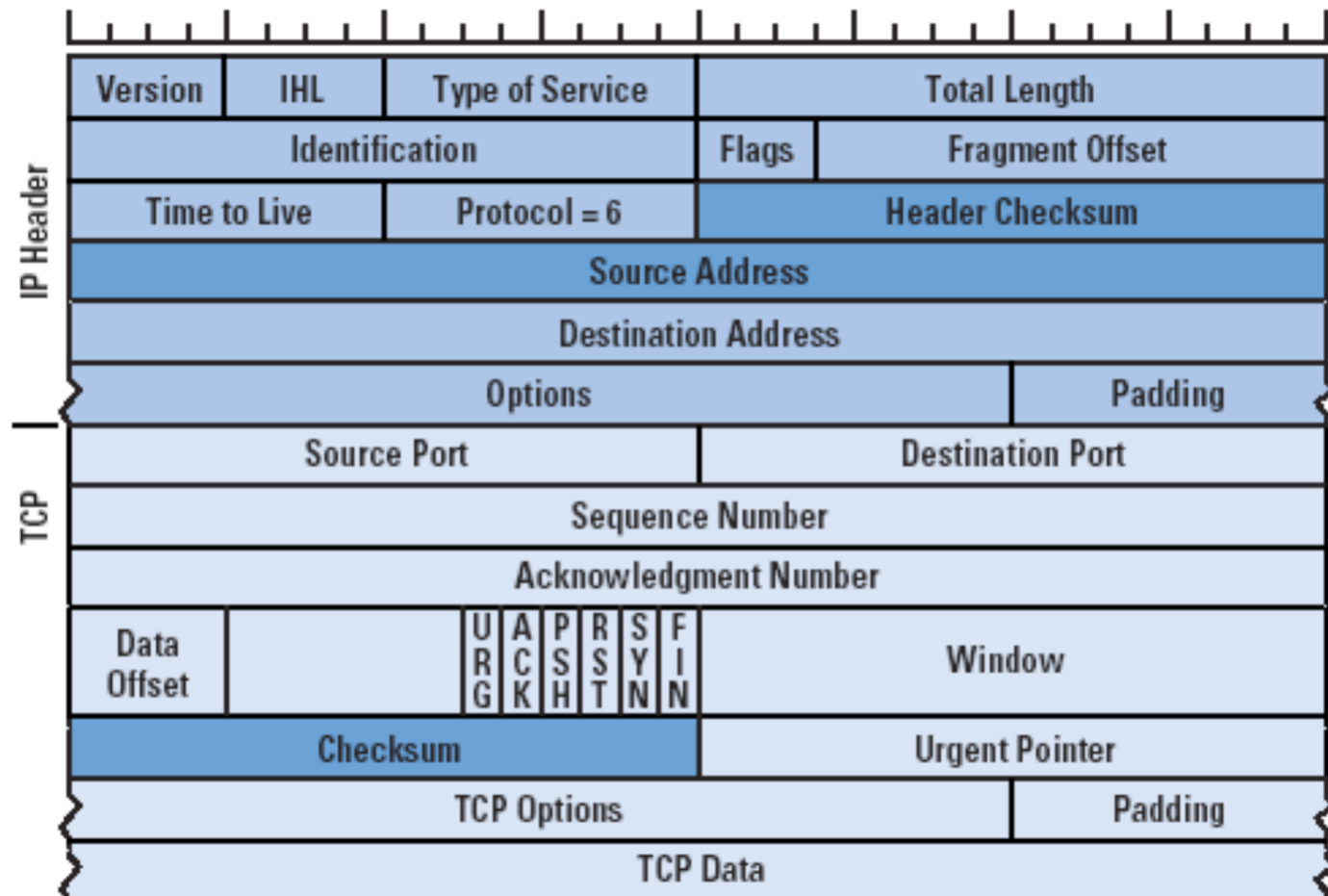
- Zastosowania UDP:
 - komunikacja multimedialna
 - strumieniowe przesyłanie dźwięku i obrazu
 - gry sieciowe (możemy stracić pojedyncze klatki)
- ... ale!!!
 - np. YouTube używa TCP (bezpieczeństwo)
 - Google wdraża też własne protokoły na potrzeby swoich usług

Protokół TCP

- **TCP** (*Transmission Control Protocol*)
 - **połączeniowy** – następuje nawiązanie połączenia i potwierdzanie gotowości do odbioru
 - **wiarygodny** – dostarczane są wszystkie dane, w odpowiedniej kolejności
 - możliwość **sterowania przepływem danych** (np. podział jednego datagramu na mniejsze lub łączenie kilku w jeden)
 - **kontrola przeciążeń**
- Datagramy TCP, tak samo jak w UDP, docierają w różnej kolejności, działa natomiast **mechanizm porządkowania danych**
- Większy rozmiar nagłówka TCP oraz narzuty na nawiązywanie połączenia – większe obciążenie sieci i wolniejszy niż UDP

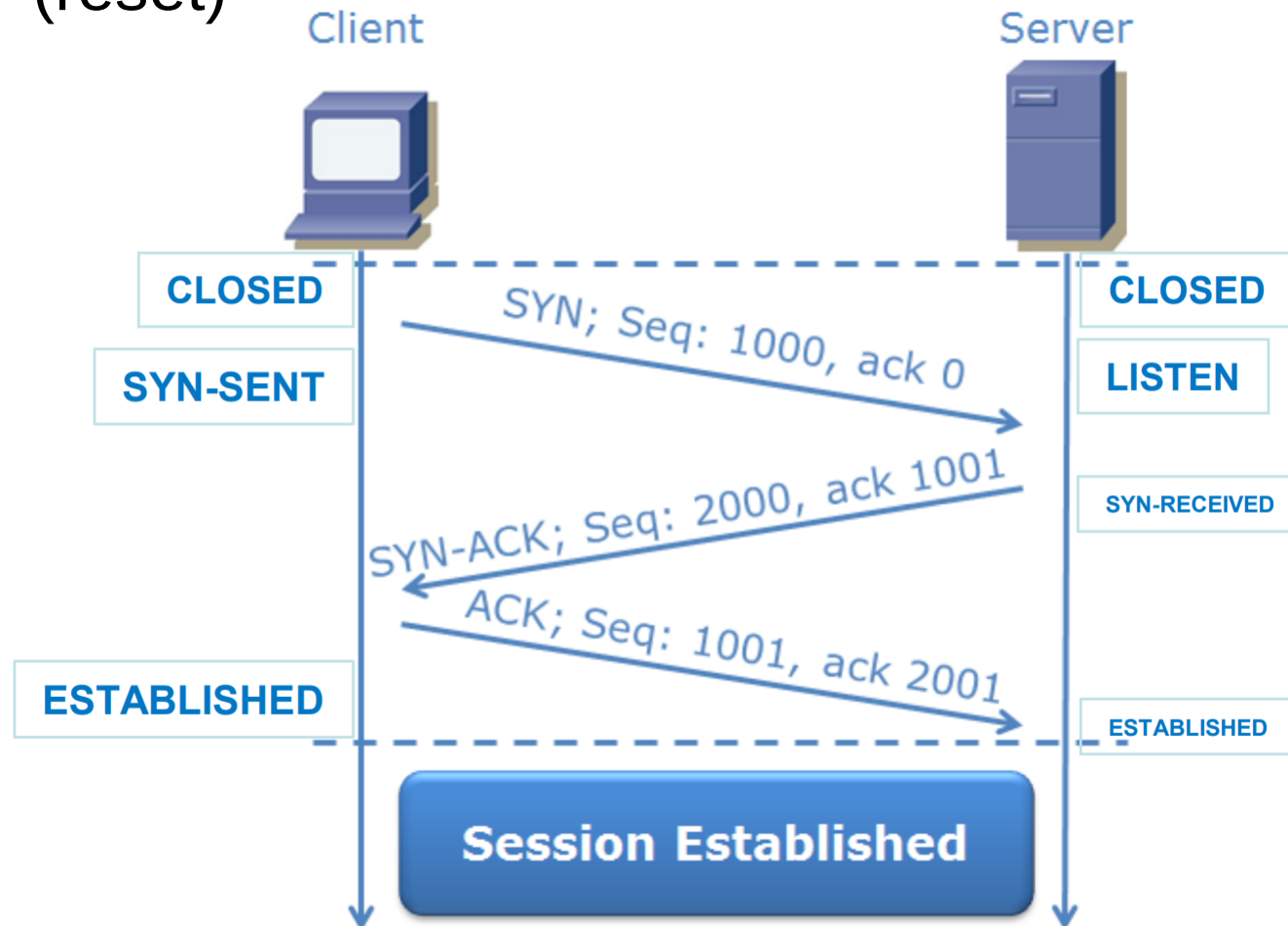
Protokół TCP

- Segment TCP (jako część składowa datagramu IP)
- Od razu widać różnicę w stosunku do UDP jeśli chodzi o jego wielkość...



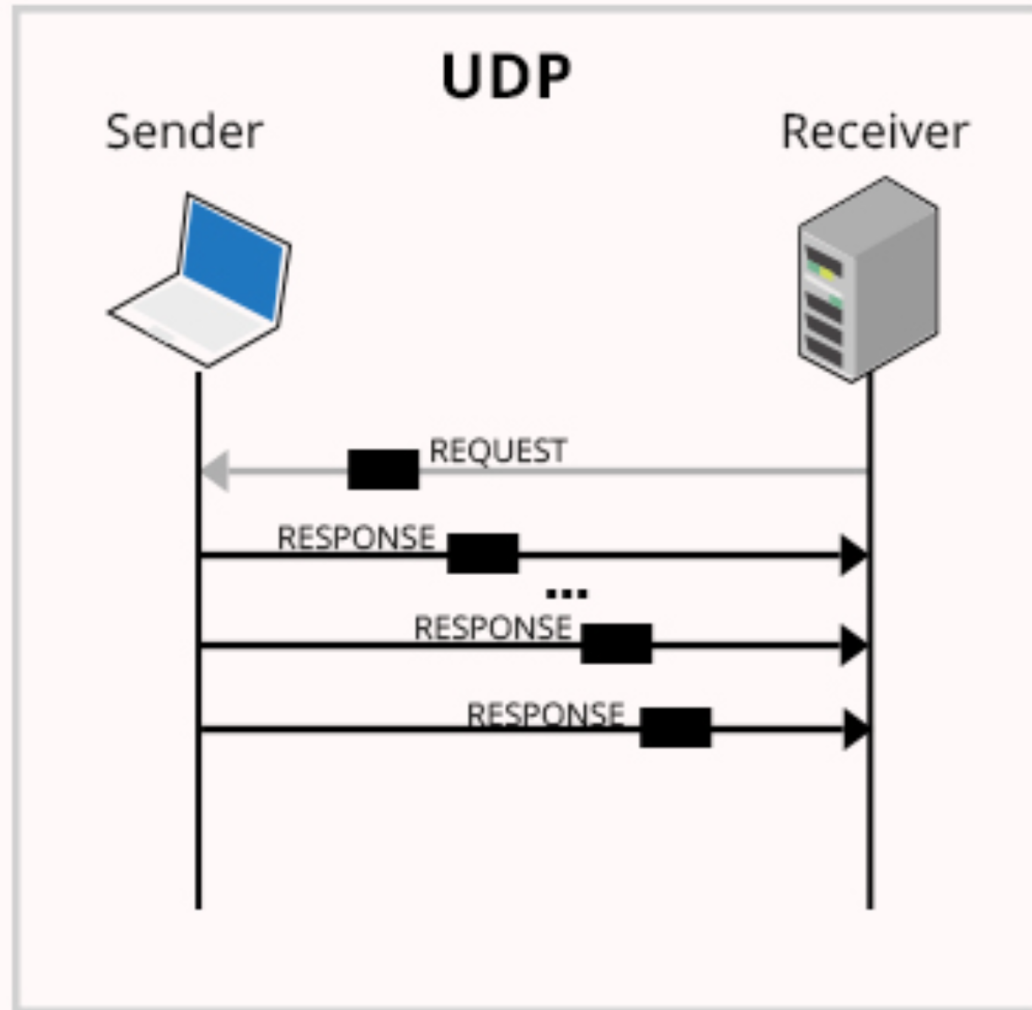
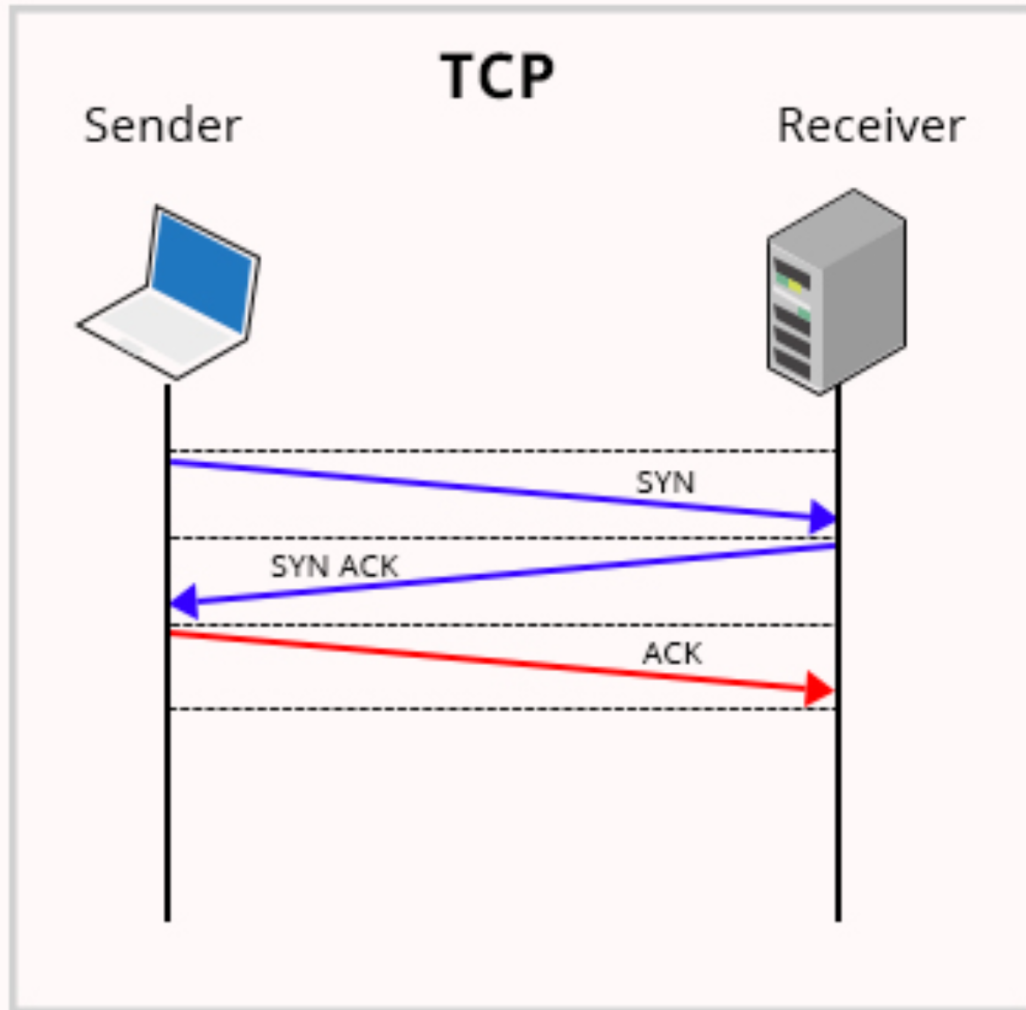
Protokół TCP

- Nawiązywanie połączenia w TCP następuje za pomocą mechanizmu “**three-way handshake**”
- Jeśli serwer nie chce (albo nie może) ustanowić połączenia, odsyła RST (reset)



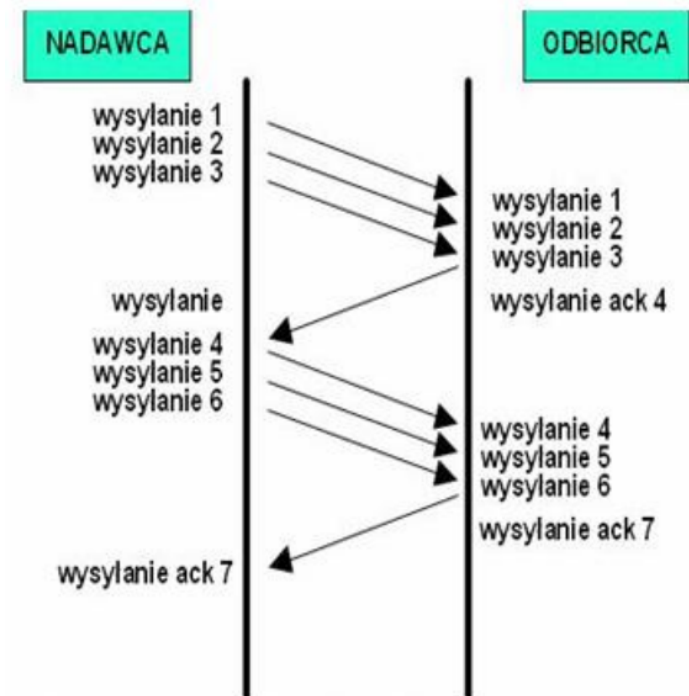
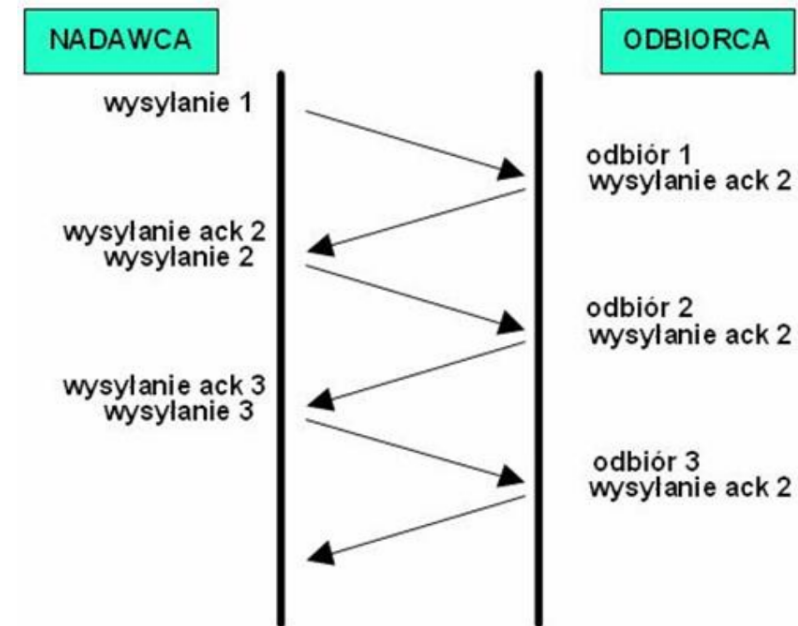
Protokół UDP

- Porównanie połączenia w TCP i “połączenia” w UDP



Przesyłanie danych w TCP

- Teoretycznie możemy sobie wyobrazić następujący scenariusz:
 - po każdym wysłaniu jednego segmentu danych odbiorca potwierdza otrzymanie danych
 - jest to jednak **mało efektywne**
- Rozwiązanie – **buforowanie** danych (przechowywanie danych po stronie klienta i serwera)



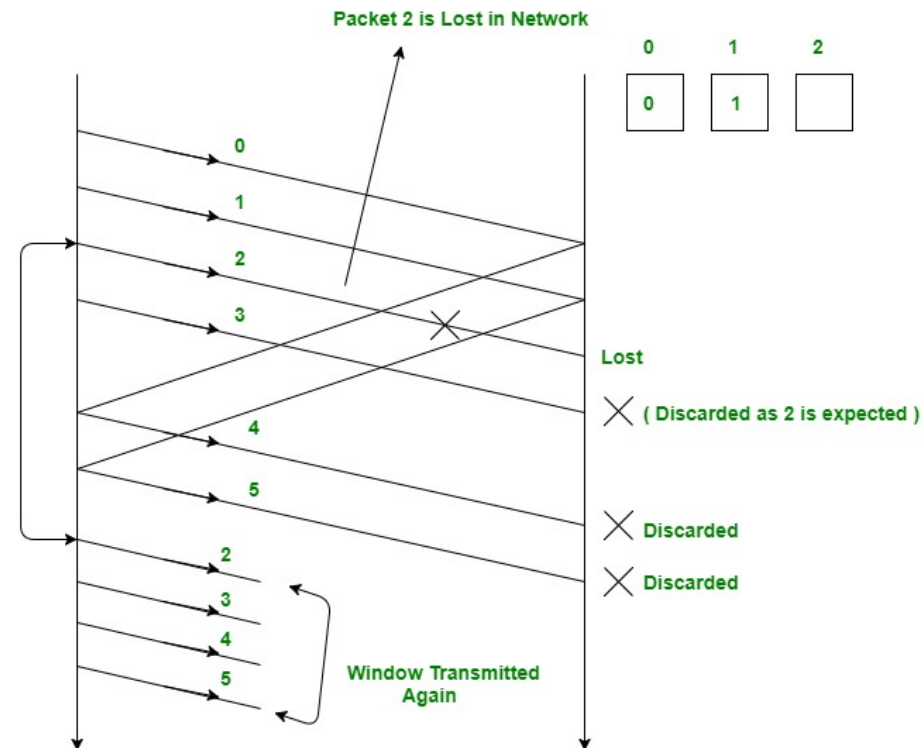
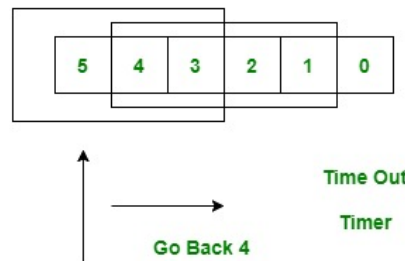
wysyłanie ack=numer potwierdzenia
(ang.acknowledgement number)

Przesyłanie danych w TCP

- Mechanizm **przesuwającego okna** (*sliding window*), wysyłanie okna zawierającego konkretne segmenty
- Wysyłamy segmenty odpowiadające rozmiarowi okna bez potwierdzania każdego z nich przed wysłaniem kolejnego
- Powtarzamy wysłanie okna, gdy nie któryś segment nie dojdzie w założonym czasie (**timeout**)

- **Kontrola przepływu:**

- zmienny rozmiar okna
- blokowanie nadawcy by nie przeciążyć bufora



Porównanie UDP i TCP

Cecha	UDP	TCP
Opis	Prosty protokół dużych przepustowości (przeniesienie funkcjonalności na warstwy wyższe)	W pełni funkcjonalny, niezawodny protokół komunikacyjny z mechanizmami obsługi błędów warstwy sieciowej
Ustanawianie połączenia	bezpołączeniowy	Połączeniowy, faza nawiązania połączenia
Interfejs danych dla aplikacji	Zorientowany na wiadomości	Zorientowany strumieniowo
Wiarygodność i potwierdzenia	Zawodny, bez potwierdzeń	Niezawodny, wymaga potwierdzeń dostarczenia datagramów
Retransmisje	Nie obsługiwane (przeniesione do warstw wyższych)	Obsługiwane automatycznie
Kontrola przepływu	Brak	Okno przesuwne zmiennych rozmiarów, mechanizmy zapobiegania przeciążeniom
Narzut	Bardzo mały	Mały
Prędkość transmisji	Bardzo duża	Duża
Typ danych (wielkość, rozmiar)	od małych do średnich	Od małych do bardzo dużych

Enkapsulacja danych

- Przykład enkapsulacji danych
- ramka Ethernet → datagram IPv4 → segment TCP

Encapsulation Payloads



Ethernet "Frame"
OSI Layer 2 - Data Link

IP "Packet"
OSI Layer 3 - Network

TCP "Segment"
OSI Layer 4 - Transport

TCP segment in IPv4 packet in Ethernet frame

Ethernet	Octets	IPv4	Bits	TCP	Bits
Preamble	7				
Start of frame delimiter	1				
MAC destination	6				
MAC source	6				
802.1Q tag (opt.)	4				
Ethertype or length	2				
Payload	46 -1500	Version	4		
		Header Length	4		
		Differentiated Services Code Point	6		
		Explicit Congestion Notification	2		
		Total Length	16		
		Identification	16		
		Flags	3		
		Fragment Offset	13		
		Time to Live	8		
		Protocol	8		
		Header Checksum	16		
		Source IP Address	32		
		Destination IP Address	32		
		Options (if Header Length > 5)	?		
		Payload	1440-1480 Bytes	Source Port	16
				Destination Port	16
				Sequence number	32
				Acknowledgment number	32
				Data offset	4
				Reserved	4
				Flag	8
				Window Size	16
				Checksum	16
				Urgent pointer	16
				Options (if Data Offset > 5)	varies
				padding	8
				Payload	Payload
CRC	4				
Interframe gap	12				



KONIEC