

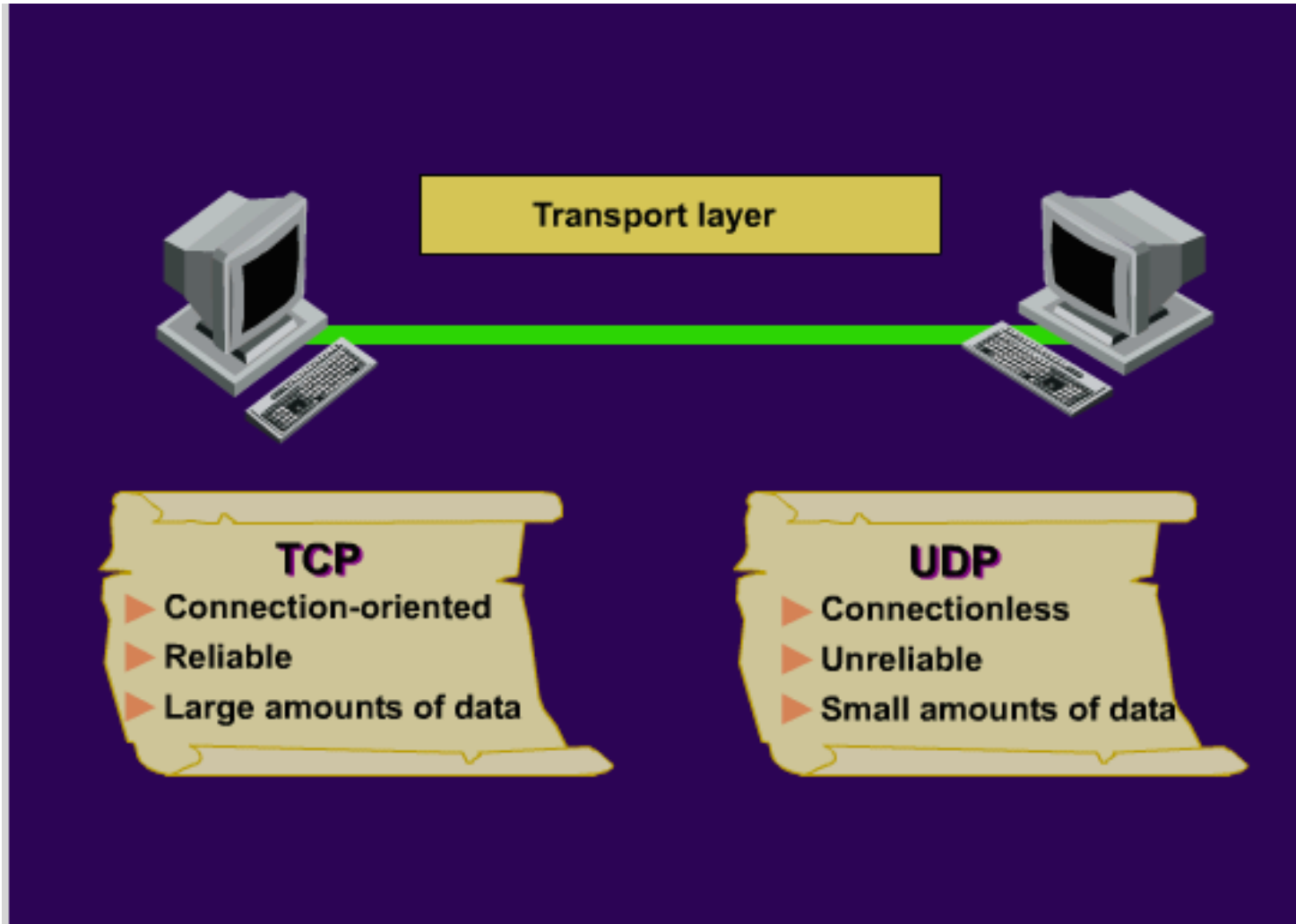
# Minggu 6

## Transport Layer

# Overview

- ▣ Layer Transport bertugas melakukan sesi komunikasi antara komputer dalam jaringan. Menentukan bagaimana data ditransmisikan.
- ▣ Dua Protocol Transport Layer yang dipakai :
  - Transmission Control Protocol (TCP)
  - User Datagram Protocol (UDP).
- ▣ User Datagram Protocol
  - UDP merupakan connectionless communication, bekerja tidak menjamin data sampai ditujuan secara utuh.
  - Normalnya untuk mentransmisikan data dalam jumlah kecil pada satu waktu.
  - Reliabilitasnya/penjaminan data sampai pada penerima tergantung dari aplikasi.
- ▣ Transmission Control Protocol (TCP)
  - Connection-oriented dan reliable communication yang artinya data dijamin sampai tujuan
  - Untuk menjamin diperlukan komunikasi awal dengan penerima sebelum transfer data dilakukan
  - Membutuhkan ack setiap penerimaan data
  - Dipakai untuk mentransmisikan data dalam jumlah besar

# Overview



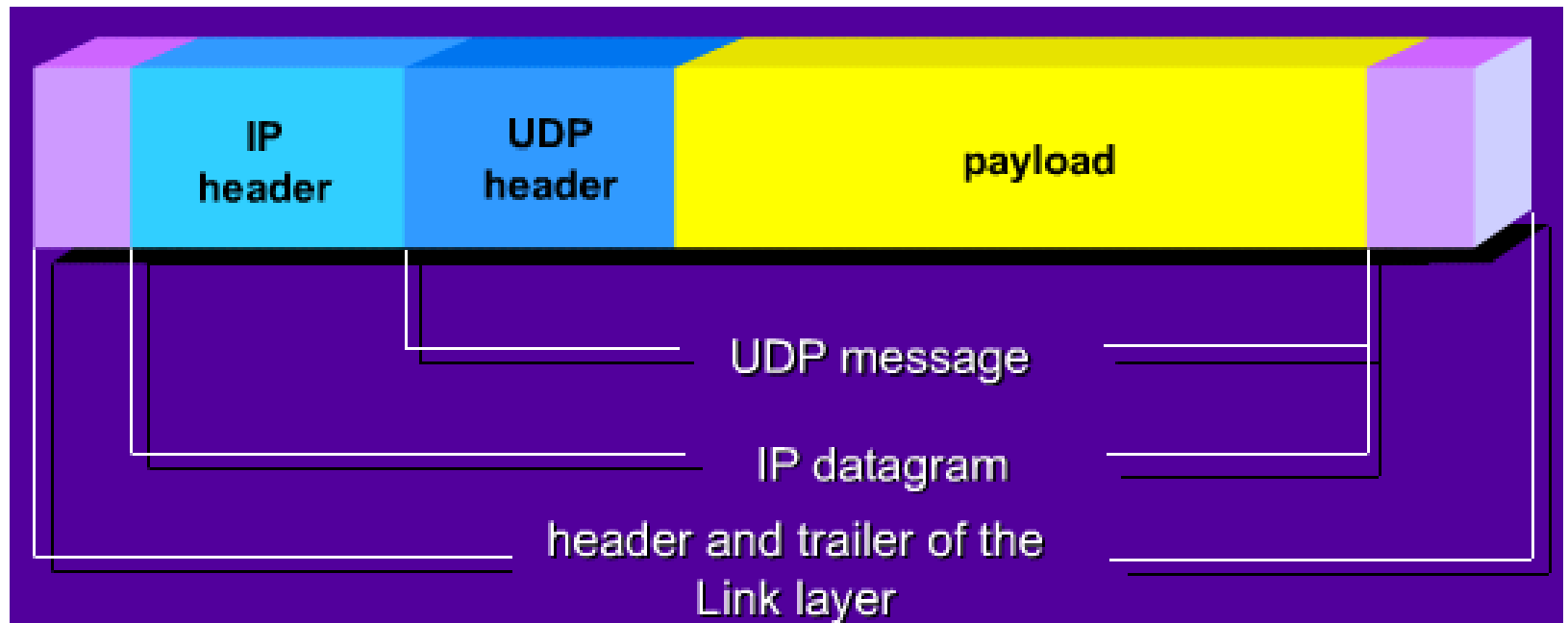
# Port and Socket

- ▣ Ada dua komponen yang biasa dipakai selama komunikasi pada layer transport yaitu port dan socket
- ▣ Port
  - Port bisa dikatakan internal address yang disediakan untuk aplikasi tertentu pada komputer. Setiap aplikasi mempunyai port yang berbeda
  - Port bisa TCP atau UDP, tergantung pada pemakaian protocol apa pada layer transport apakah Udp atau TCP
  - Nomor Port antara 0 and 65,535.
  - Aplikasi TCP/IP biasanya menggunakan nomor port dibawah 1,024, dimana setiap aplikasi biasanya nomornya sudah pasti. Port ini biasa disebut "Well-Known Ports".
- ▣ Socket
  - Merupakan kombinasi dari IP address dan TCP atau UDP port.
  - Aplikasi men-generate socket ketika berkomunikasi dengan komputer lain
  - IP address menentukan tujuan komputer dan Port menentukan aplikasi yang dipakai.

# UDP

- ▣ UDP merupakan protokol connectionless, artinya tidak ada sesi komunikasi awal ketika data ditransmisikan.
- ▣ UDP merupakan unreliable protokol. Berarti pesan yang dikirim tanpa ada nomor urut dan tanpa acknowledgment dari penerima sehingga pengirim tidak pernah tahu apakah pesan sudah diterima penuh atau tidak. Untuk masalah ini ditangani oleh aplikasi
- ▣ Jika terjadi Lost paket data harus di-retrieve oleh layer di atasnya (aplikasi).
- ▣ Biasanya message UDP ditransmisikan secara regular dalam interval waktu tertentu atau setelah ditentukan batas waktu habis
- ▣ Hanya membutuhkan sedikit resource memori dan processor
- ▣ Contoh aplikasi yang menggunakan Protocol UDP Domain Name System(DNS) dan Dynamic Host Configuration Protocol(DHCP).

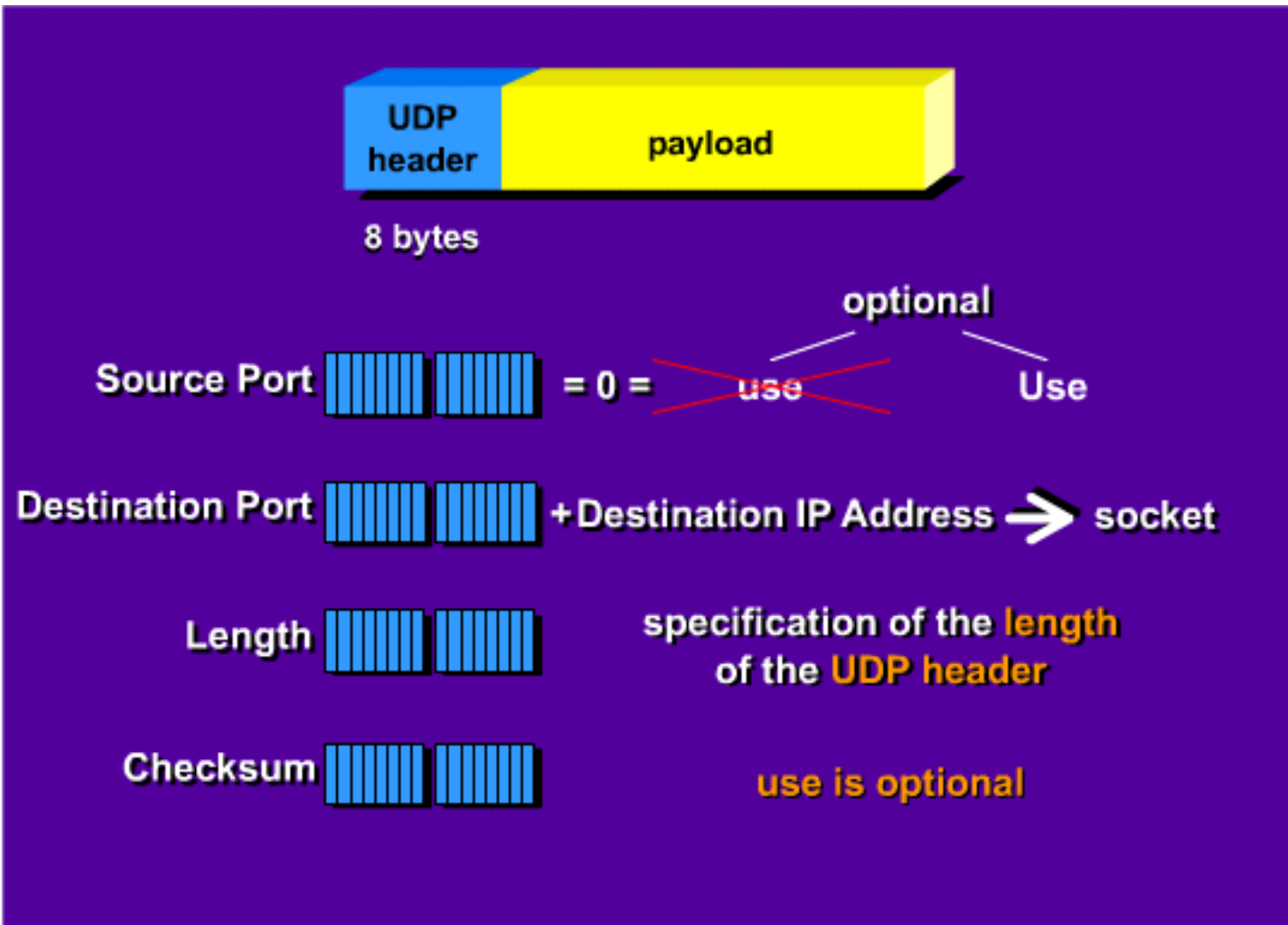
# IP Datagram UDP



# IP Datagram

- ▣ Message UDP ditransmisikan dalam bentuk IP datagrams.
- ▣ Message UDP, terdiri dari :
  - IP Header
  - UDP header
  - Payload
- ▣ IP header terdiri dari Source IP dan Destination IP :
  - Source IP berisi IP address host yang mengirim paket
  - Destination IP berisi alamat penerima paket, bisa broadcast address atau multicast address.

# UDP Header Structure





# Struktur Header UDP

- ▣ Header UDP header mempunyai panjang yang tetap yaitu 8 bytes, Terdiri dari 4 field : Source Port, Length field dan Checksum
- ▣ Source Port terdiri dari 2 byte yang mengidentifikasi Port pengirim yang dipakai untuk mentransmisikan data. Source Port merupakan optional bisa diisi bisa tidak, jika tidak diisi diset 0. Misal pengirim data video yang tidak butuh reply/pengiriman balik
- ▣ Destination Port, berisi Port tujuan yang dikirim data. Gabungan Destination IP dan Destination Port membentuk Socket.
- ▣ Length field mengindikasikan panjang Header UDP.
- ▣ Checksum field, menyediakan integriti checker. Optional, jika diset 0 berarti tidak dipakai, Pengirim tidak melakukan proses perhitungan.

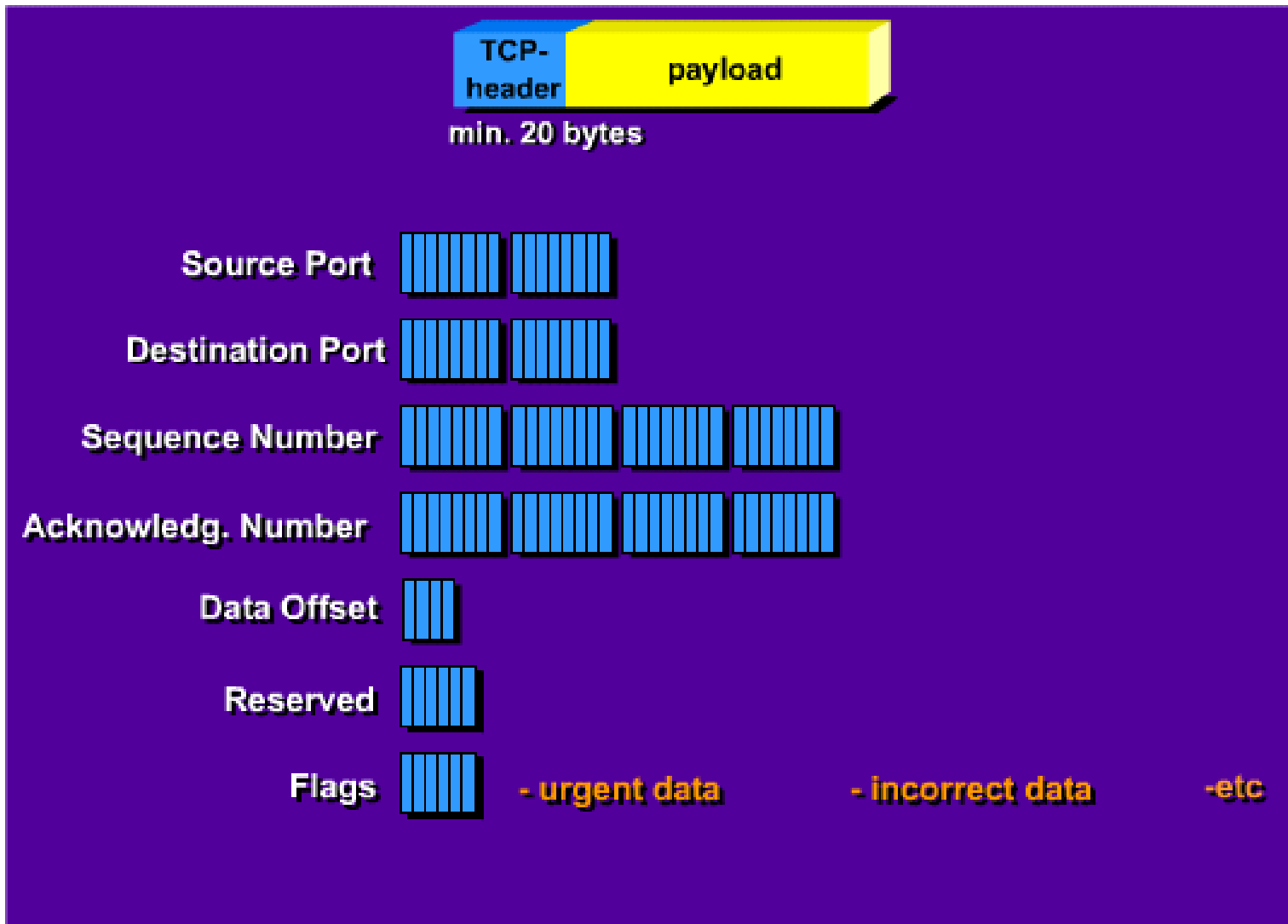
# TCP Overview

- ▣ TCP merupakan protocol connection-oriented, yang artinya data hanya bisa ditransmisikan setelah ada proses negosiasi terlebih dahulu antara pengirim dan penerima
- ▣ Negosiasi diantaranya berupa : Berapa data yang bisa dikirim dalam satu waktu, nomor urut yang dipakai setiap pengiriman data dll.
- ▣ TCP biasanya merupakan komunikasi fully duplex, yang artinya Setiap host yang berkomunikasi mempunyai dua chanel logical untuk mengirim dan menerima message
- ▣ TCP Menyediakan transmisi data yang reliable, dengan cara:
  - Setiap paket data diberi sequence number, dan positive acknowledgement oleh receiver is expected, jika tidak harus retransmite data
  - Receiver akan membuang jika terjadi duplikasi data, dan resequences packets jika kedatangan tidak urut

# Flow Control

- Selain itu, TCP mensupport Flow Control untuk menghindari terlalu banyak data yang dikirim pada satu waktu dan overload pada jaringan router
- Flow Control artinya harus ada kesepakatan berada besar data yang dikirim dalam satu waktu antara pengirim dan penerima.
- Flow Control mengindikasikan ukuran buffer penerima yang free yang bisa diisi dalam waktu tertentu

# Struktur TCP



# TCP Header

- TCP header panjangnya bervariasi. Panjang minimal 20 bytes.
- Terdiri dari 7 field : Source Port, Destination Port, Sequence Number, Ack. Number, Data Offset, Reserver dan Flag.
- 2 byte masing –masing untuk Source Port and the Destination Port. Sama seperti UDP.
- 4 byte sequence Number yang berisi nomor urut transmisi data dalam satu segment
  - Ini digunakan cekung ketika semua byte telah diterima
- Acknowledgement Number gterdiri dari 4 byte.
  - Berisi Sequence number berikutnya dari penerima
- Data Offset mengindikasikan awal data. Ini berhubungan dengan ukuran TCP header.
- Diikuti 6 bit reserve untuk penggunaan kedepan, diset 0.
- Flags menentukan tipe informasi pada segment.

# Flag

Flags 

Urgent		
Acknowledgement		<b>!sending acknowledgement!</b>
Push		<b>!push function!</b>
Reset		<b>!connection reset!</b>
Synchronization		<b>!connection set-up!</b>
Final		<b>!end of the byte stream!</b>

# Tahapan Komunikasi pada TCP

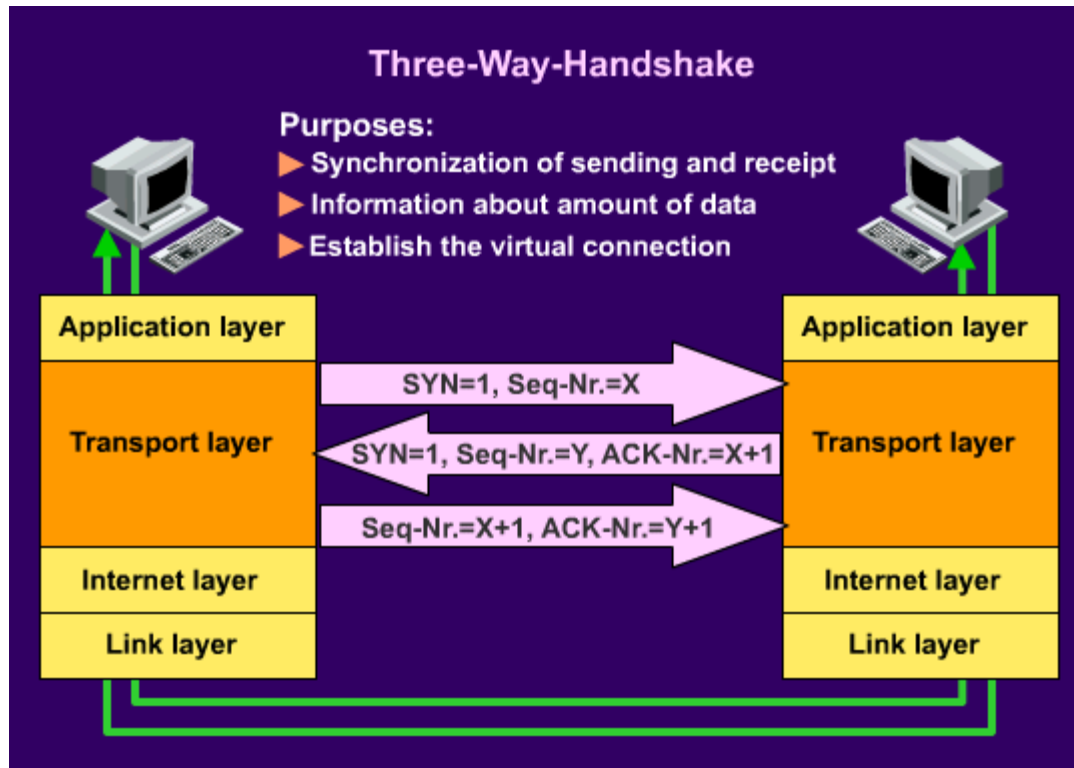
- TCP adalah Protocol connection-oriented.
- Sebelum data ditransmisikan, koneksi yang dibuat bisa diset atau dirubah sesuai keadaan.
- Tiga tahap komunikasi connection set-up kemudian data transfer dan connection release
- Data yang ditransmisikan bisa dipakai untuk tiga keadaan tadi

# Three Way Handshake

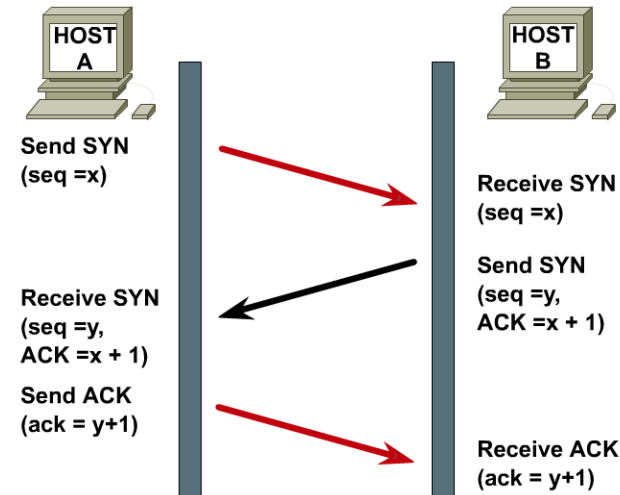
- Koneksi TCP diawali oleh prosedur yang biasa disebut dengan Three-Way-Handshake. Tujuannya untuk melakukan sinkronisasi antara pengirim dan penerima. Hal yang diinformasikan selama Three Way Handshake adalah Jumlah data yang bisa ditransmisikan dalam satu waktu, Sequence number yang dipakai.
- Untuk setup koneksi, host melakukan session inisialisasi dengan menset flag sinkronisasi ke 1.
- Segment juga berisi sequence number yang mengindikasikan awal byte yang ingin dikirim berikutnya. Juga berisi acknowledgement yang terdiri dari sequence number berikutnya untuk menerima data.
- Setelah Three Way Handshake dilakukan baru dianggap session established, dan koneksi dua arah siap dilaksanakan



# Three Way Handshake



## TCP Three-Way Handshake/ Open Connection



Packet 1: source: 130.57.20.10 dest.:130.57.20.1

TCP: ----- TCP header -----

TCP: Source port = 1026  
TCP: Destination port = 524  
TCP: Initial sequence number = 12952  
TCP: Next expected Seq number = 12953  
TCP: .... ..1. = SYN  
TCP: Window = 8192  
TCP: Checksum = 1303 (correct)  
TCP: Maximum segment size = 1460 (TCP Option)

Packet 2: source: 130.57.20.1 dest: 130.57.20.10

TCP: ----- TCP header -----

TCP: Source port = 524  
TCP: Destination port = 1026  
TCP: Initial sequence number = 2744080  
TCP: Next expected Seq number = 2744081  
TCP: Acknowledgment number = 12953  
TCP: .... ..1. = SYN  
TCP: Window = 32768  
TCP: Checksum = D3B7 (correct)  
TCP: Maximum segment size = 1460 (TCP Option)

Packet 3: source: 130.57.20.10 dest: 130.57.20.1

TCP: ----- TCP header -----

TCP: Source port = 1026  
TCP: Destination port = 524  
TCP: Sequence number = 12953  
TCP: Next expected Seq number = 12953  
TCP: Acknowledgment number = 2744081  
TCP: ...1 .... = Acknowledgment  
TCP: Window = 8760  
TCP: Checksum = 493D (correct)  
TCP: No TCP options

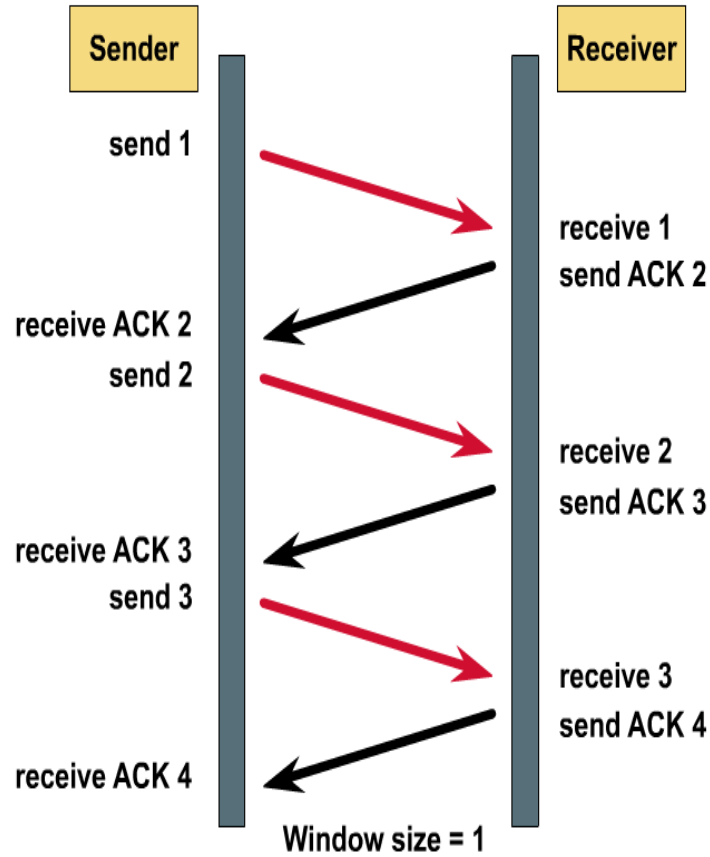
■ Maksimum ukuran segment dan ukuran windows yang dinegosiasikan juga dikirim

# PAR

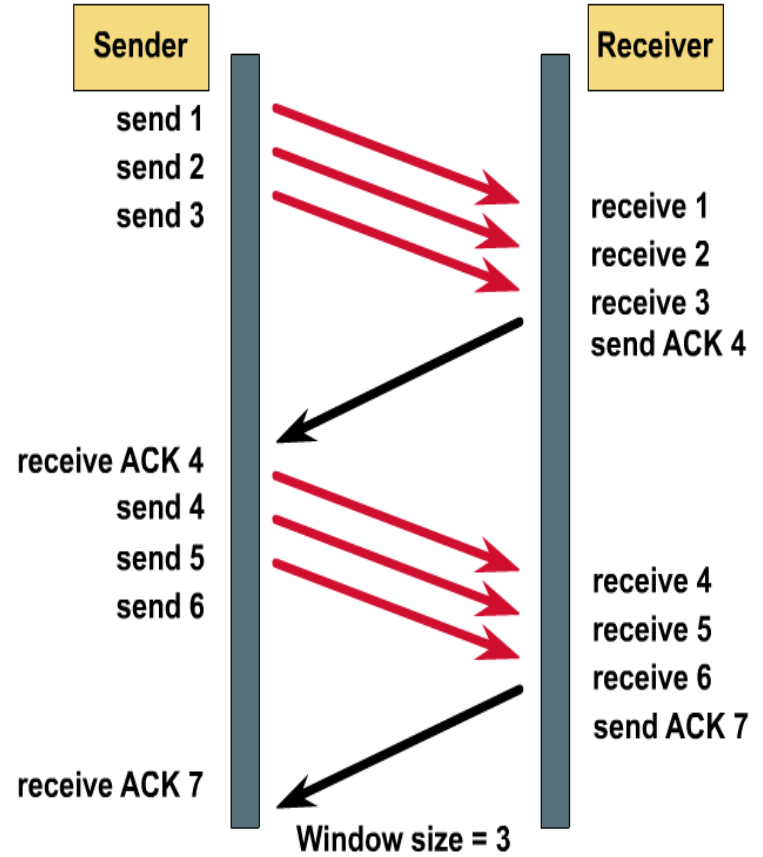
- Setelah koneksi establishe, ini harus dimantain sampai salah satu partner komunikasi ingin mengakhir komunikasi.
- System Transfer Data didasarkan pada mekanisme PAR (*Positive Acknowledgement with Retransmission*).
- Artinya bahwa untuk kebenaran data yang diterima maka penerima data harus mengirimkan acknowledgement ke pengirim.
- Untuk efisiensi, acknowledgements hanya berisi paket selanjutnya yang harus dikirim, tidak untuk setiap individu paket

# PAR

## TCP Simple Acknowledgment



## TCP Sliding Window

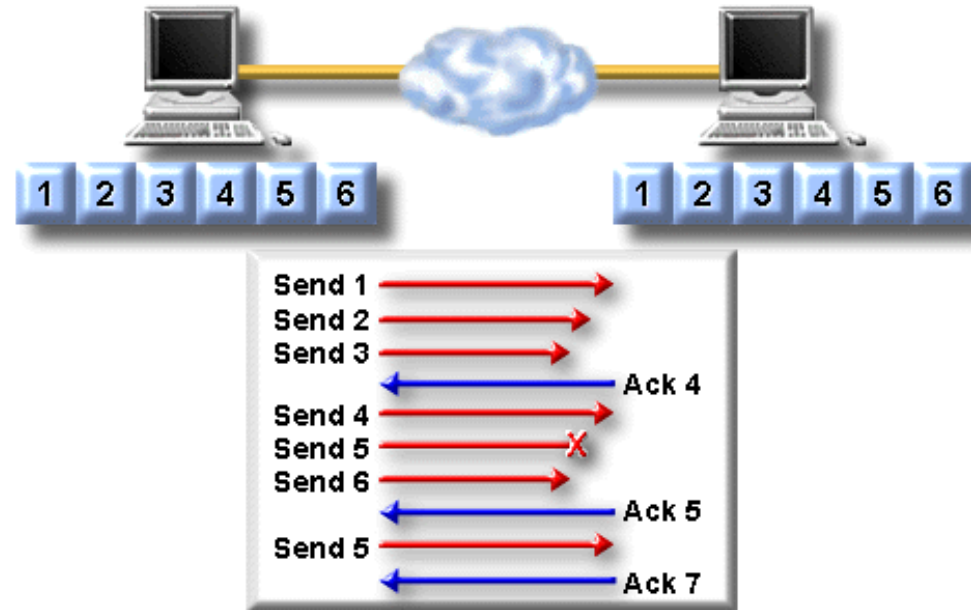


15 16 31

16-bit Source Port Number		16-bit Destination Port Number	
32-bit Sequence Number			
32 bit Acknowledgement Number			
4-bit Header Length	6-bit (Reserved)	U R G	A C K
		P R S T	S S T
		F I N	S Y N
16-bit TCP Checksum			16-bit Window Size
16-bit Urgent Pointer			
Options (if any)			
Data (if any)			

## TCP Header

## An Acknowledgment Technique

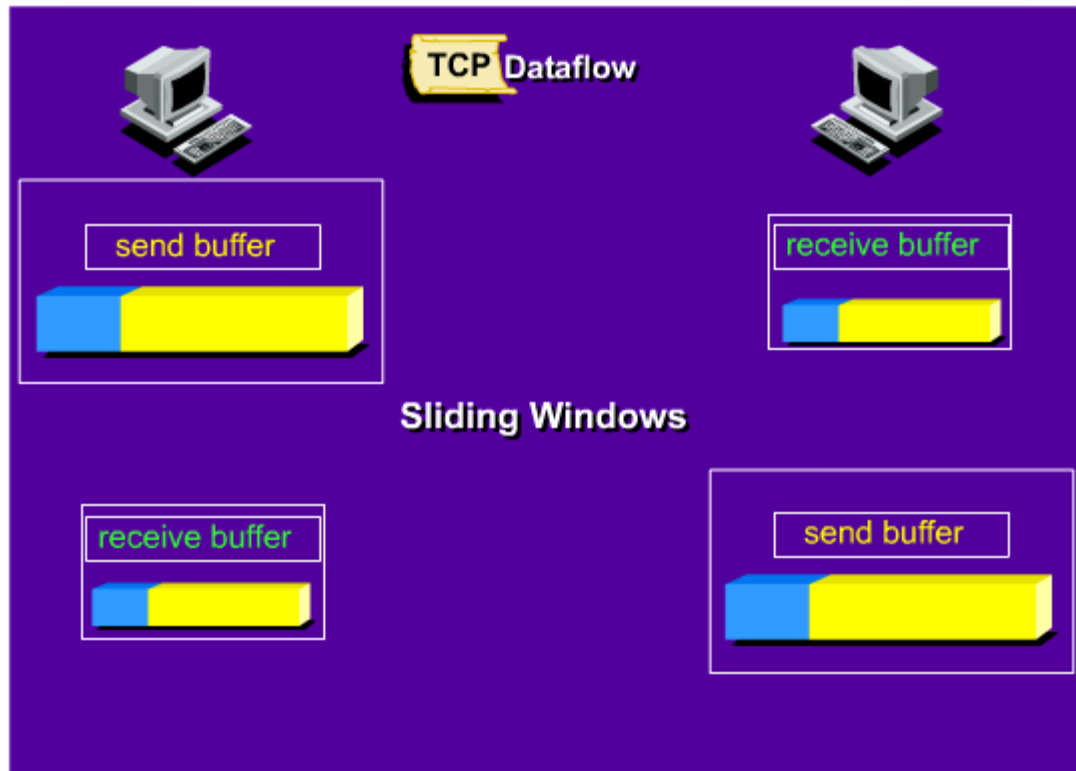


- Paket mungkin didrop sepanjang jalan, time out atau rusak
- Jika misal 4, 5, dan 6 dikirim, tapi 5 lost, receiver hanya akan memberi ack sampai 4, mengirim ack 5.
- Pengirim akan mengirim ulang paket 5 dan menunggu untuk mendengar dari penerima paket mana yang selanjutnya dikirim
- Receiver mengirim Ack 7, jadi pengirim tahu dapat memulai lagi mengirim paket ke-7 dan seterusnya

# Sliding Window

- Untuk melakukan transmisi, data penerima menyiapkan buffer.
- Untuk mekanisme ini TCP menggunakan mekanisme sliding windows.
- Setiap host mempunyai akses ke dua windows: satu mengirim data dan yang lain menerima data.
- Ukuran windows mengindikasikan jumlah buffer yang disiapkan untuk data

# Sliding Window



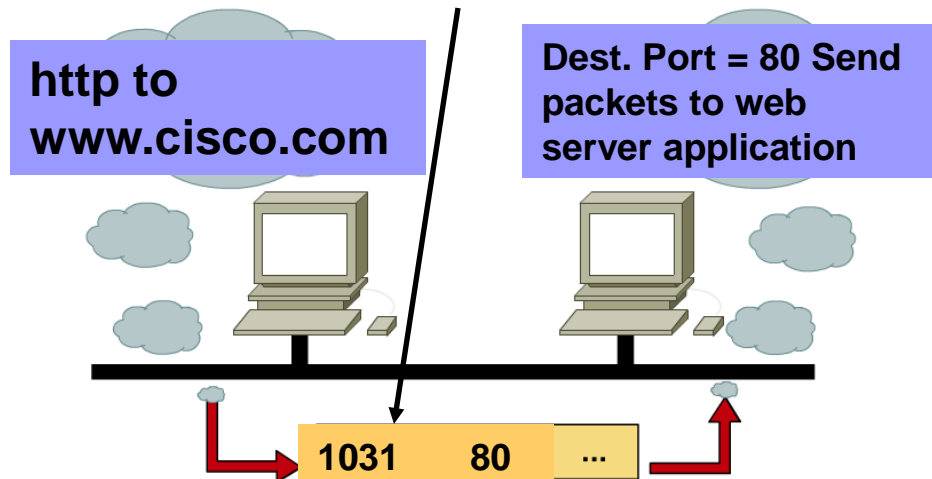
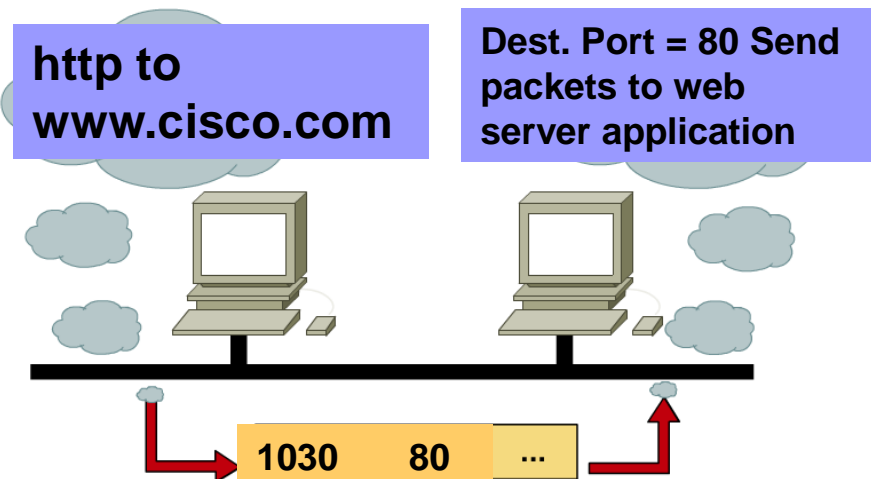
# Contoh Penggunaan Transport Layer



# TCP/UDP Port Numbers

Source Port	Destination Port	...
-------------	------------------	-----

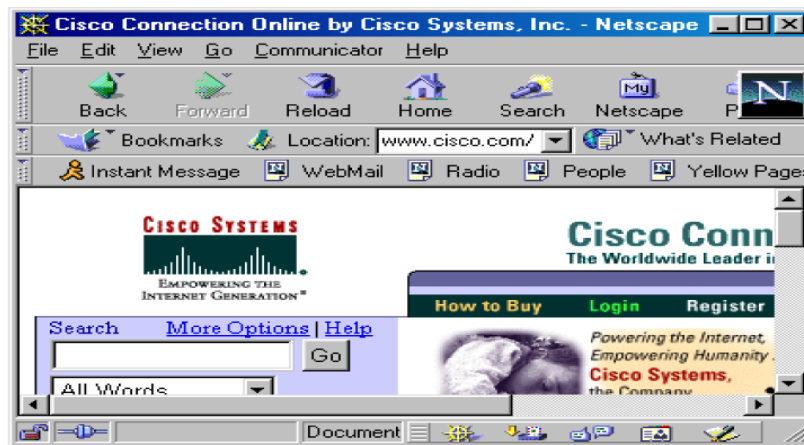
Pada setiap session http antara client dan server yang sama, mempunyai destination port yang sama, tapi berbeda Source Port (unik) untuk mengidentifikasi setiap session sehingga pengembalian permintaan masuk ke sesi yang benar



## Netscape Navigator

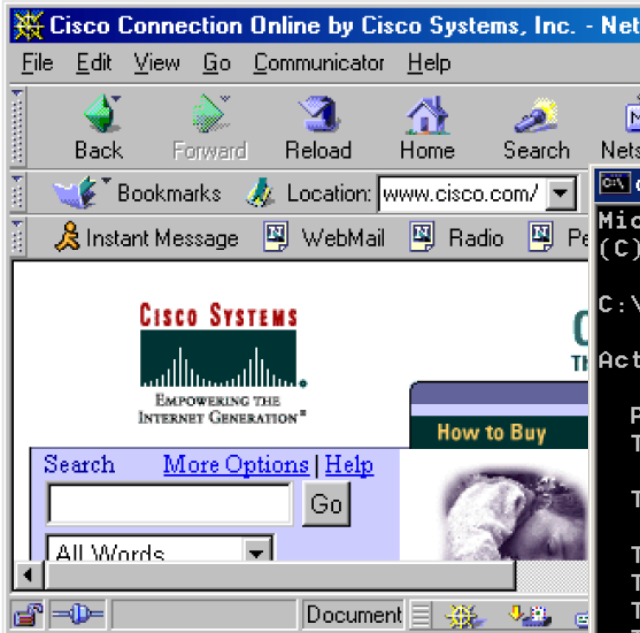


## Netscape Navigator



Ini menunjukkan contoh dua browser windows dengan URL yang sama. TCP/IP menggunakan source port numbers untuk pengembalian informasi

# Netscape Navigator



TCP or UDP

Source IP

Destination IP

Connection State

Source Port

Destination Port

[www.google.com](http://www.google.com)

[www.cisco.com](http://www.cisco.com)

```
Microsoft Windows [Version 5.00.2195]
(C) Copyright 1985-2000 Microsoft Corp.

C:\>netstat

Active Connections

Proto Local Address          Foreign Address        State
TCP    RICK-GRAZIANI:1430    RICK-GRAZIANI.cabrillo-lab.net:1431 ESTABLISHED
TCP    RICK-GRAZIANI:1431    RICK-GRAZIANI.cabrillo-lab.net:1430 ESTABLISHED
TCP    RICK-GRAZIANI:1414    65.104.27.219:http    ESTABLISHED
TCP    RICK-GRAZIANI:1432    boris.cabrillo.cc.ca.us:imap ESTABLISHED
TCP    RICK-GRAZIANI:1919    216.239.37.126:http   CLOSE_WAIT
TCP    RICK-GRAZIANI:1920    216.239.37.126:http   CLOSE_WAIT
TCP    RICK-GRAZIANI:1937    216.239.37.126:http   CLOSE_WAIT
TCP    RICK-GRAZIANI:1938    216.239.37.126:http   CLOSE_WAIT
TCP    RICK-GRAZIANI:1975    208.44.195.174:http   TIME_WAIT
TCP    RICK-GRAZIANI:1977    www.cisco.com:http    ESTABLISHED
TCP    RICK-GRAZIANI:1978    www.cisco.com:http    ESTABLISHED
TCP    RICK-GRAZIANI:1979    www.cisco.com:http    ESTABLISHED
TCP    RICK-GRAZIANI:1980    www.cisco.com:http    ESTABLISHED
TCP    RICK-GRAZIANI:1985    cco-sj-1.cisco.com:http ESTABLISHED
TCP    RICK-GRAZIANI:1986    63.209.4.18:http     TIME_WAIT

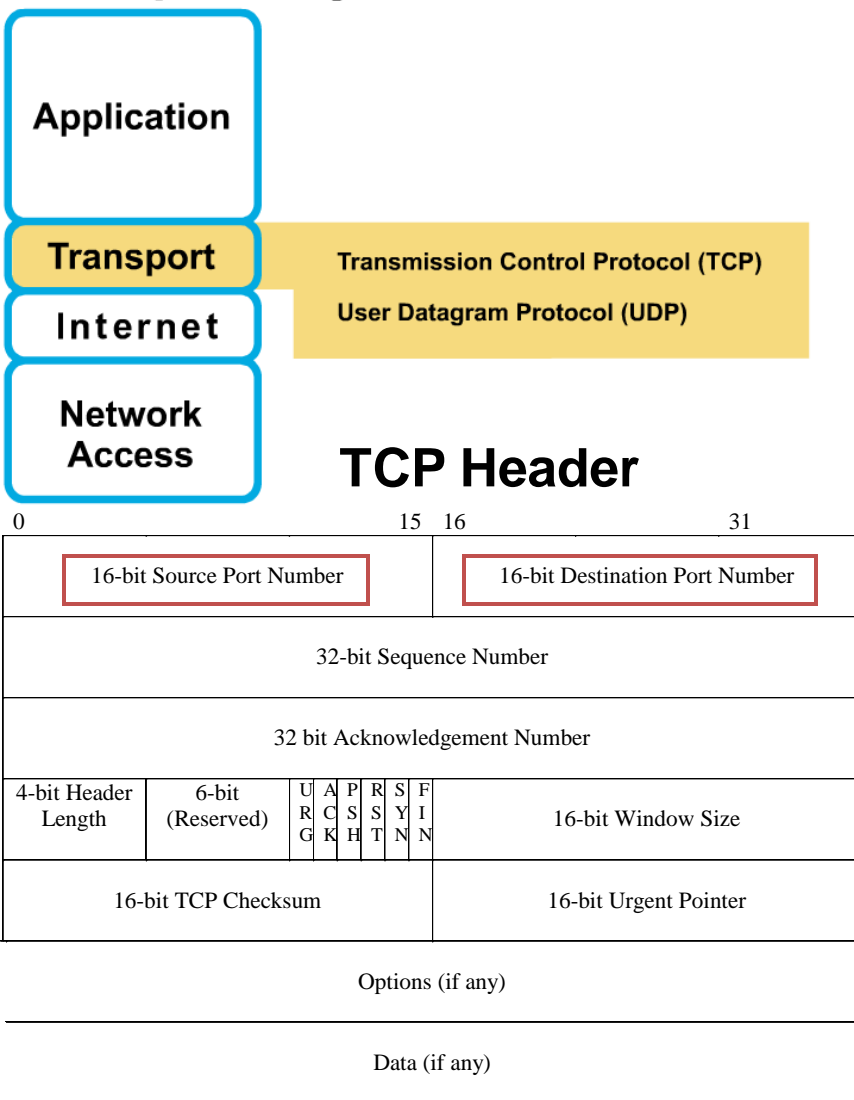
C:\>
```

**netstat command**

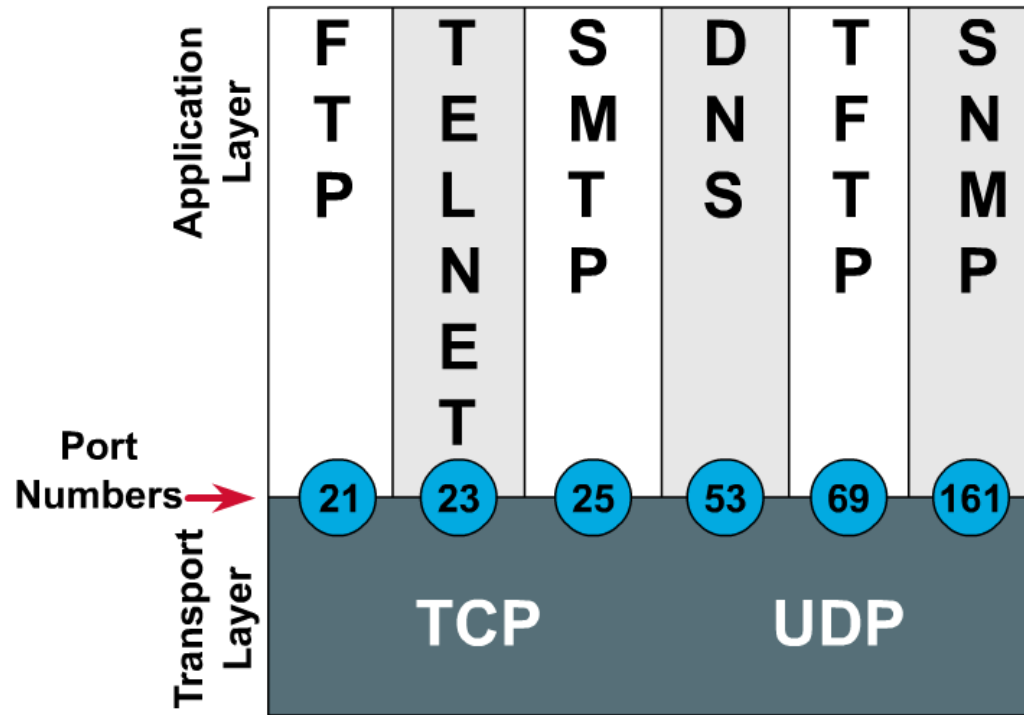
- **Aktualnya**, ketika kita membuka sebuah halaman html, maka session TCP dibangun.
- Jika membuka beberapa halaman web maka multiple koneksi TCP dibentuk, setiap koneksi di client akan membentuk port yang berbeda-beda, dengan port tujuan sama.

# Hubungan antara Aplikasi, Port dan Transport Layer

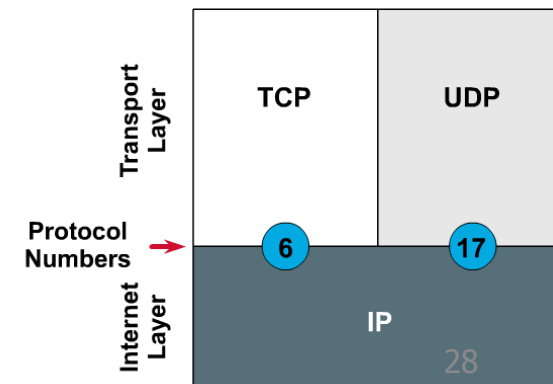
# Transport Layer Overview



# Port Numbers



# The Protocol Field

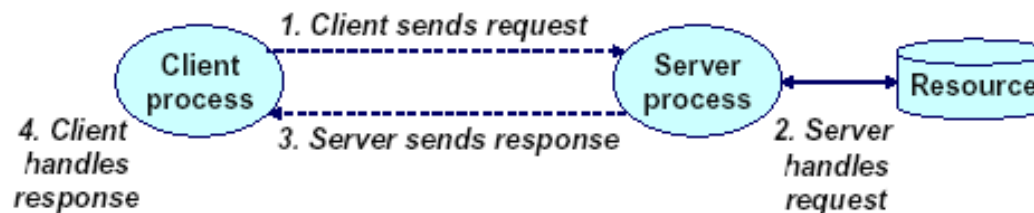


- **Keduanya TCP dan UDP** menggunakan Port untuk meneruskan informasi ke layer di atasnya

# Socket Programming

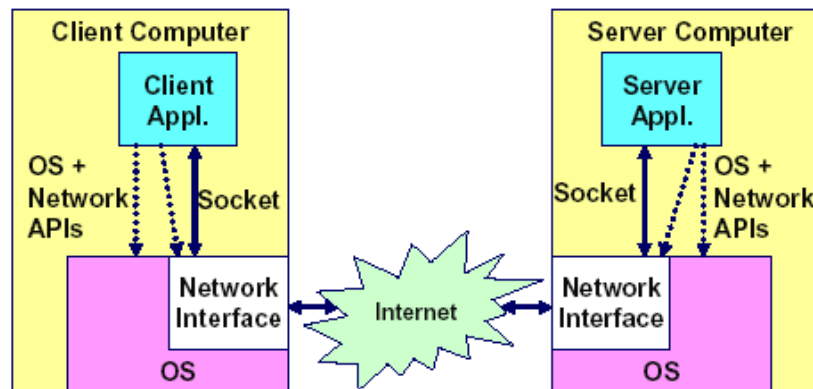
# Pendahuluan

- aplikasi di jaringan, transaksinya didasarkan pada konsep *client-server*. Sebuah atau beberapa *client* meminta/*request* pelayanan ke server.
- Aplikasi *client-server* menggunakan protokol *transport* untuk saling berinteraksi.
- Ketika proses interaksi terjadi, suatu aplikasi harus memberikan informasi-informasi secara detail tentang :
  - Informasi tentang apakah dia *client* atau *server*.
  - Pengirim memberikan informasi tentang data yang dikirim.
  - Penerima memberikan informasi tentang dimana data diletakkan, dll



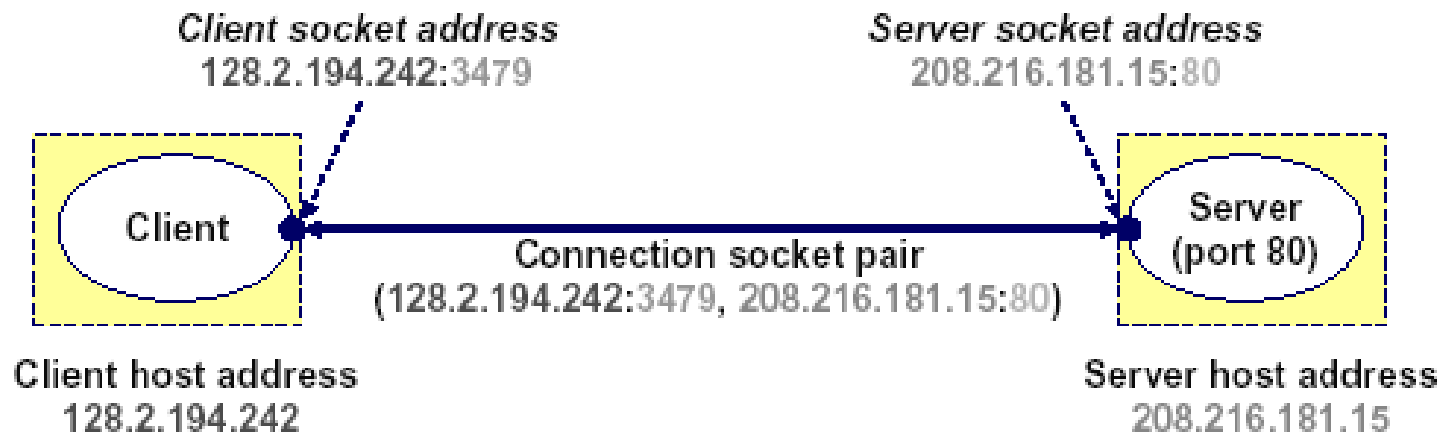
# Konsep Socket

- Antarmuka (*interface*) antara program aplikasi dengan protokol komunikasi pada suatu sistem operasi disebut *Application Program Interface (API)*. *API* didefinisikan sebagai suatu kumpulan instruksi yang mendukung proses interaksi antara suatu perangkat lunak dengan suatu protokol yang digunakan.
- Pada mesin keluarga *Linux*, *socket* terintegrasi dengan *I/O* sehingga aplikasi yang berkomunikasi dengan *socket*, cara kerjanya sama dengan suatu aplikasi yang mengakses peralatan *I/O*. Oleh karena itu untuk memahami cara kerja *socket* pada *Linux*, sebelumnya harus juga memahami fasilitas *I/O* pada *Linux*.



# Konsep Socket

- Pada saat suatu aplikasi berkomunikasi, awalnya aplikasi membuat *socket* baru, maka pada aplikasi tersebut akan diberikan nomer yang digunakan sebagai referensi *socket*.
- Jika ada suatu sistem yang menggunakan nomer referensi *socket* tersebut, maka akan terjalin suatu jaringan komunikasi antar komputer sebaik transfer data lokal



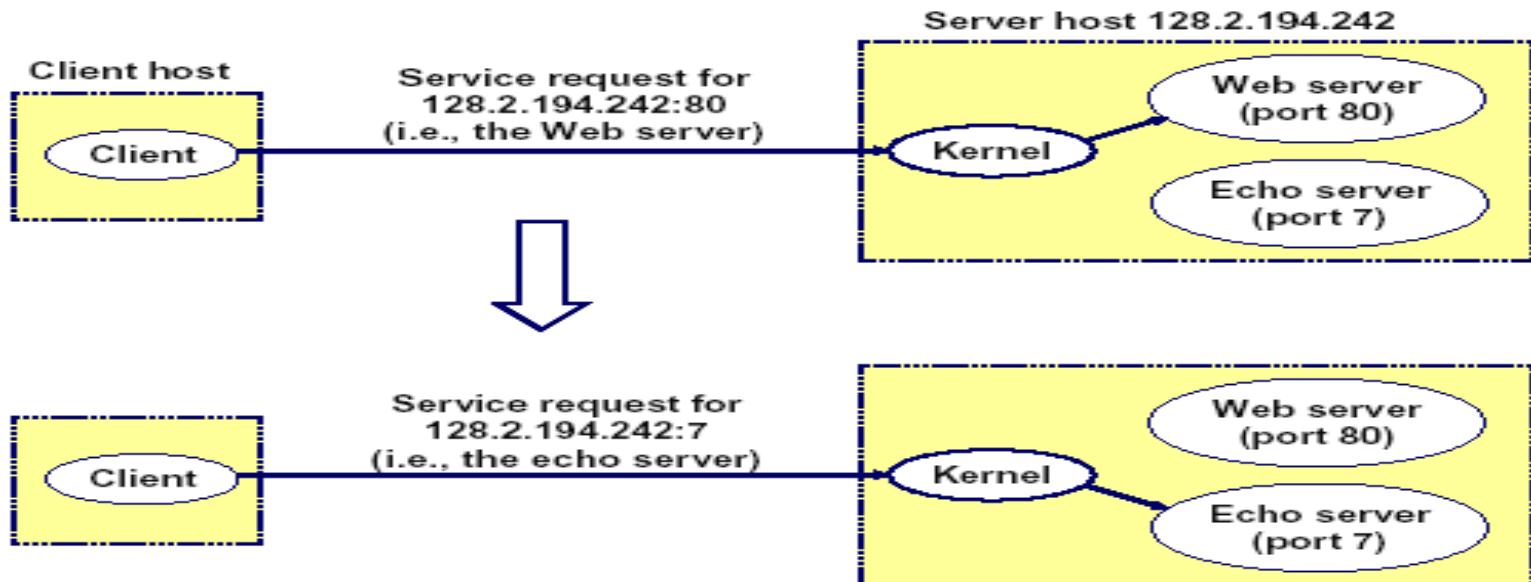
*Note: 3479 is an ephemeral port allocated by the kernel*

*Note: 80 is a well-known port associated with Web servers*  
15-441, Spring 2004



# Konsep Socket...

- Untuk berkomunikasi dengan *server*, *client* harus tahu nomor *IP server* begitu juga nomor *port* yang dituju, nomor port menunjukkan service yang dijalankan. Contoh port 23 untuk *Telnet Server*, port 25 untuk *Mail Server* dan port 80 untuk *Web Server*. Dalam hal ini aplikasi di client sudah mengetahui port yang akan dituju. Contoh program aplikasi di client yang meminta service di server ada;ah ftp, telnet, ssh. Untuk melihat service bisa dilihat pada file */etc/services*.
- Program yang berjalan di server, akan berjalan sepanjang waktu (disebut sebagai daemon) sampai mesin/service dimatikan, menunggu request dari *client* sesuai service yang diminta.



# Procedure Utama Socket

- Client, meminta layanan, langkah :
  - Membuka koneksi client ke server, yang di dalamnya adalah :
    - Membuat socket dengan perintah `socket()`
    - melakukan pengalamatan ke server.
    - Menghubungi server dengan `connect()`
  - Melakukan komunikasi (mengirim dan menerima data), dengan menggunakan perintah `write()` dan `read()`
  - Menutup hubungan
- Server, menyediakan layanan, langkah:
  - Melakukan prosedur pembukaan koneksi yang di dalamnya berupa langkah – langkah : membuat socket, mengikat socket, menyiapkan socket menerima koneksi, pengalamatan socket
  - Looping utama adalah menerima koneksi, dan melakukan komunikasi data (mengirim dan menerima).

# Latihan Soal

1. Sebutkan perbedaan kegunaan antara TCP dan UDP !
2. Jelaskan apa yang dimaksud dengan Port dan Socket !
3. Sebutkan urutan-urutan proses Three Way Handshake !
4. Cari di internet dan buat contoh program sederhana menggunakan protokol TCP !
5. Cari di internet dan buat contoh program sederhana menggunakan protokol UDP !