Jaringan Komputer (KP041)

edisi kerjasama dengan Univ Kalabahi

Pertemuan 11

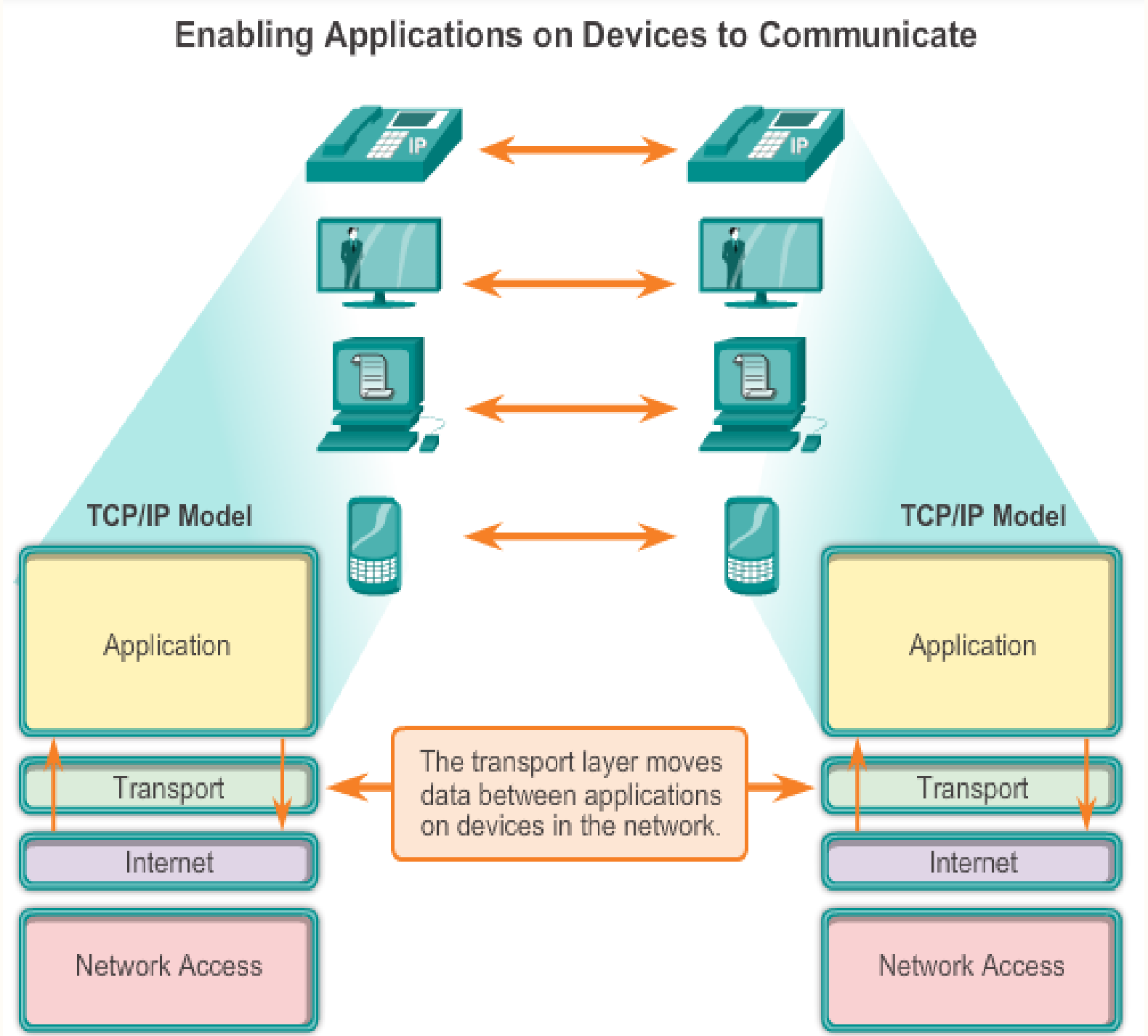
Review

- Pada pertemuan sebelumnya telah dibahas:
  - Fungsi switching: circuit switching dan packet switching
- Pada pertemuan ini akan dibahas:
  - Transport layer
  - Protokol TCP dan UDP

Transport Layer

OSI transport layer

- Pada bab ini kita akan membahas:
  1. Menjelaskan pentingnya transport layer
  2. Mengidentifikasikan peran transport layer sebagai penyedia layanan transfer end-to-end antara aplikasi
  3. Menjelaskan fungsi utama transport layer
  4. Menjelaskan peran protokol TCP dan UDP dan contoh aplikasinya



OSI transport layer

Fungsi OSI transport layer

- Secara umum, fungsi layer ini ialah:
  1. Memungkinkan banyak aplikasi berkomunikasi melalui jaringan secara bersamaan pada suatu perangkat
  2. Menjamin (bila dibutuhkan) reliabilitas data yang diterima dan pengurutan potongan segment
  3. Menangani mekanisme error

Tugas OSI transport layer

– Untuk memenuhi fungsinya, beberapa tugas yang dilakukan oleh transport layer adalah:

1. Melakukan “tracking” komunikasi antar aplikasi sumber dan tujuan
2. Mengidentifikasikan aplikasi dan yang menggunakan
3. Segmentasi data dan re-assembly
4. Conversation control (jika diperlukan)

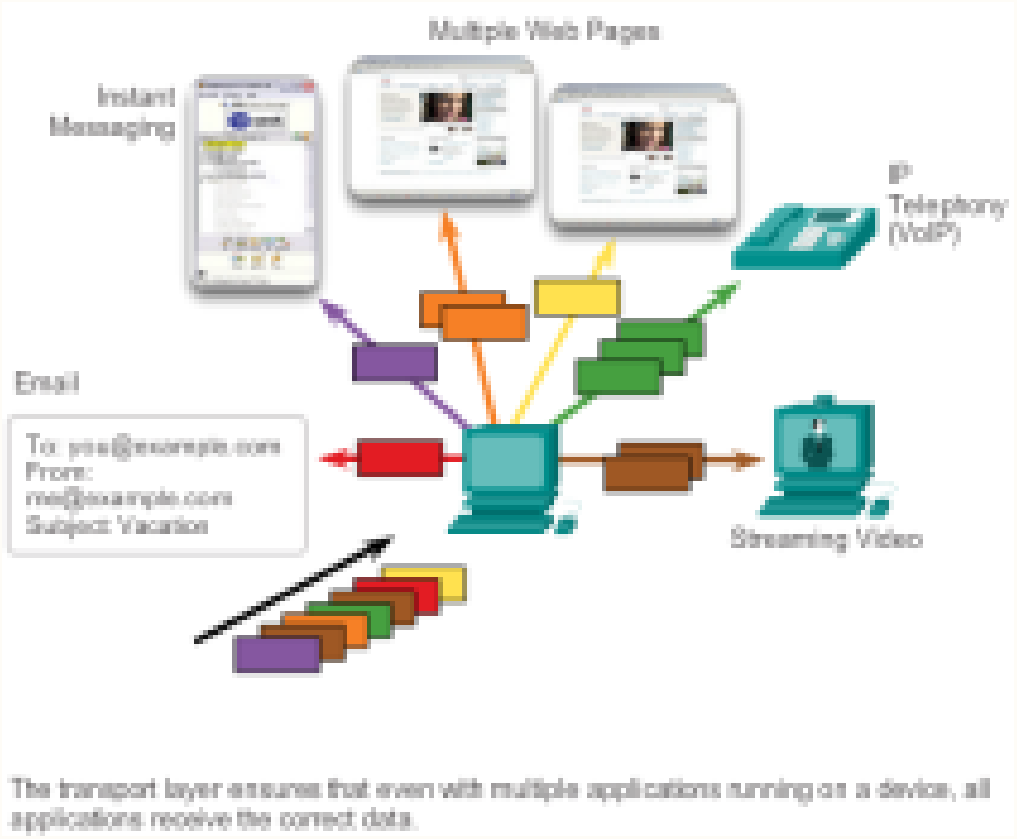
Tugas transport layer → 1. “tracking”

- Setiap host dapat memiliki berbagai aplikasi yang memanfaatkan jaringan.
- Tiap aplikasi ini akan berkomunikasi dengan aplikasi lain atau host lain, ini disebut dengan conversation (komunikasi).
- Sebuah host dapat memiliki lebih dari satu conversation secara bersamaan
- Tugas layer 4 ialah mempertahankan berbagai jalur conversation diantara aplikasi ini

Tugas transport layer → 2. Identifikasi aplikasi

- Transport layer dapat mengetahui aplikasi pengguna data dengan cara memberikan identifier khusus.

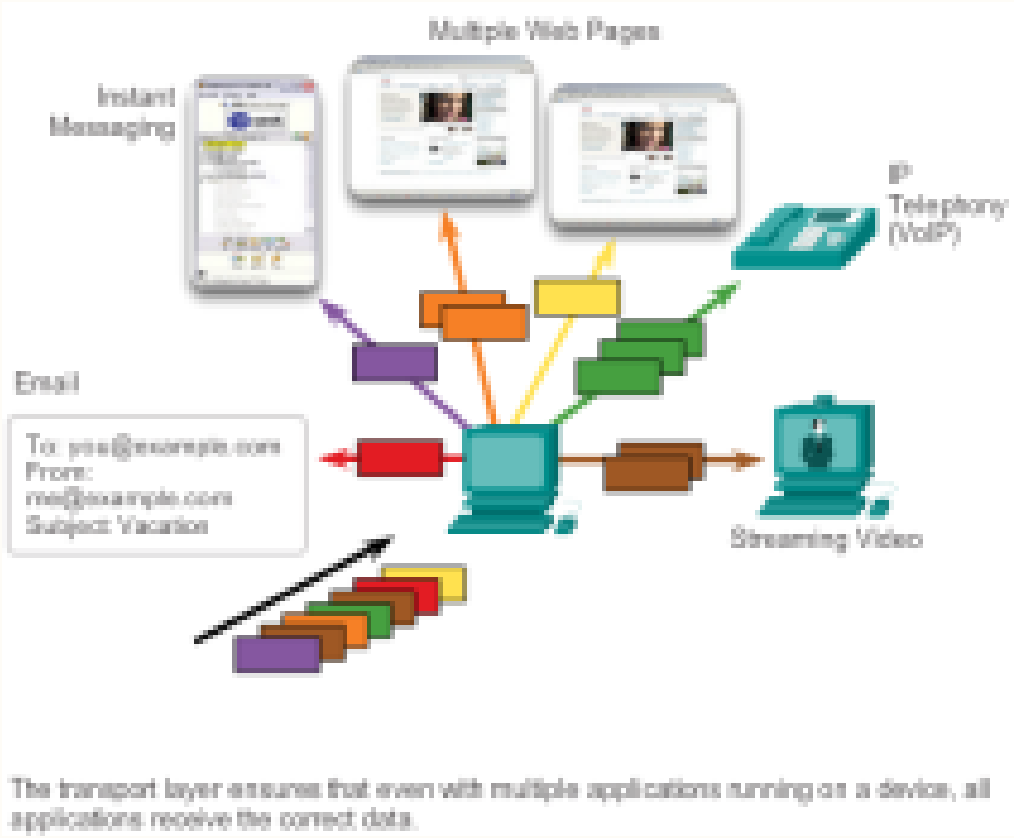
- Dengan bantuan layer transport, aplikasi yang memanfaatkan jaringan tidak perlu mengetahui detail operasi jaringan, seperti : jenis host, tipe media, jalur pengiriman, dan ukuran jaringan



Tugas transport layer → 2. Identifikasi aplikasi

- Layer dibawah transport juga tidak mengetahui aplikasi apa yang mengirimkan data. Tanggung jawab layer dibawah hanya mengirimkan data ke tujuan.

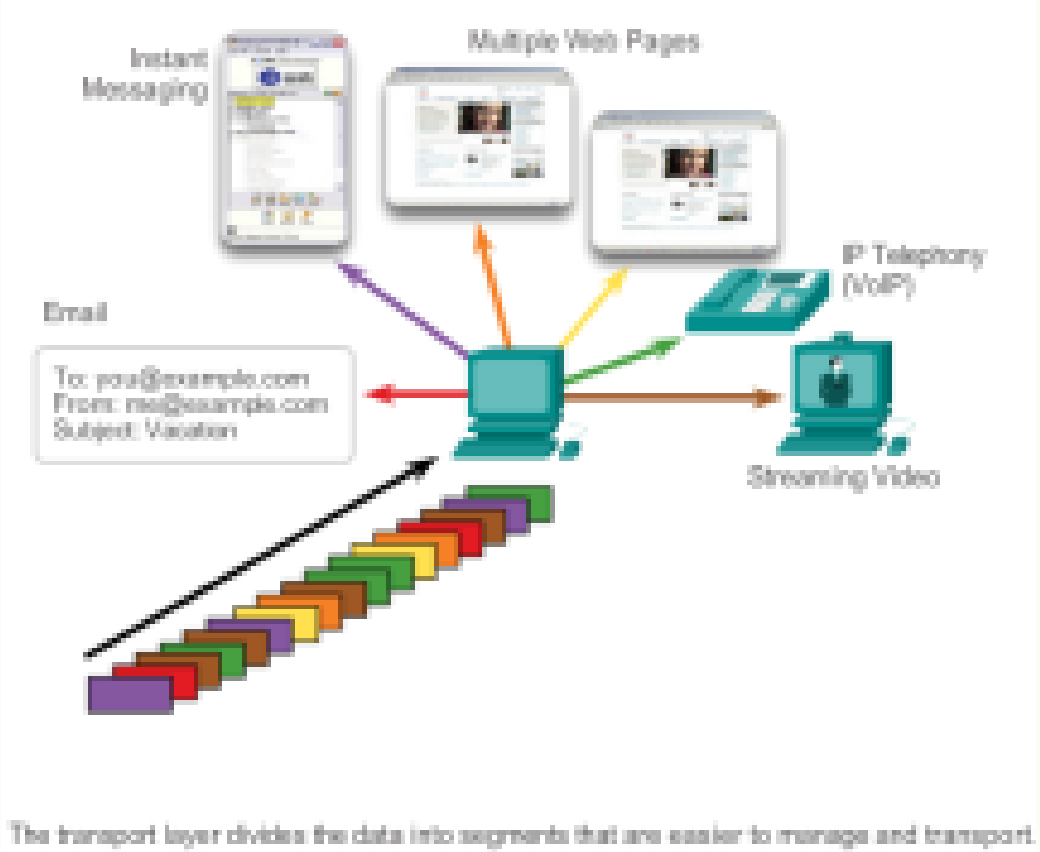
- Pada protokol TCP dan UDP, identifier ini disebut nomor port (port number)



Tugas transport layer
→ 3. segmentasi data dan re-assembly

- Setiap aplikasi membuat stream data untuk dikirim, agar data ini bisa dikirim dengan lebih optimal, maka harus dipecah (di segmentasi) agar bisa dikirim melalui media (ingat metode packet – swithing).

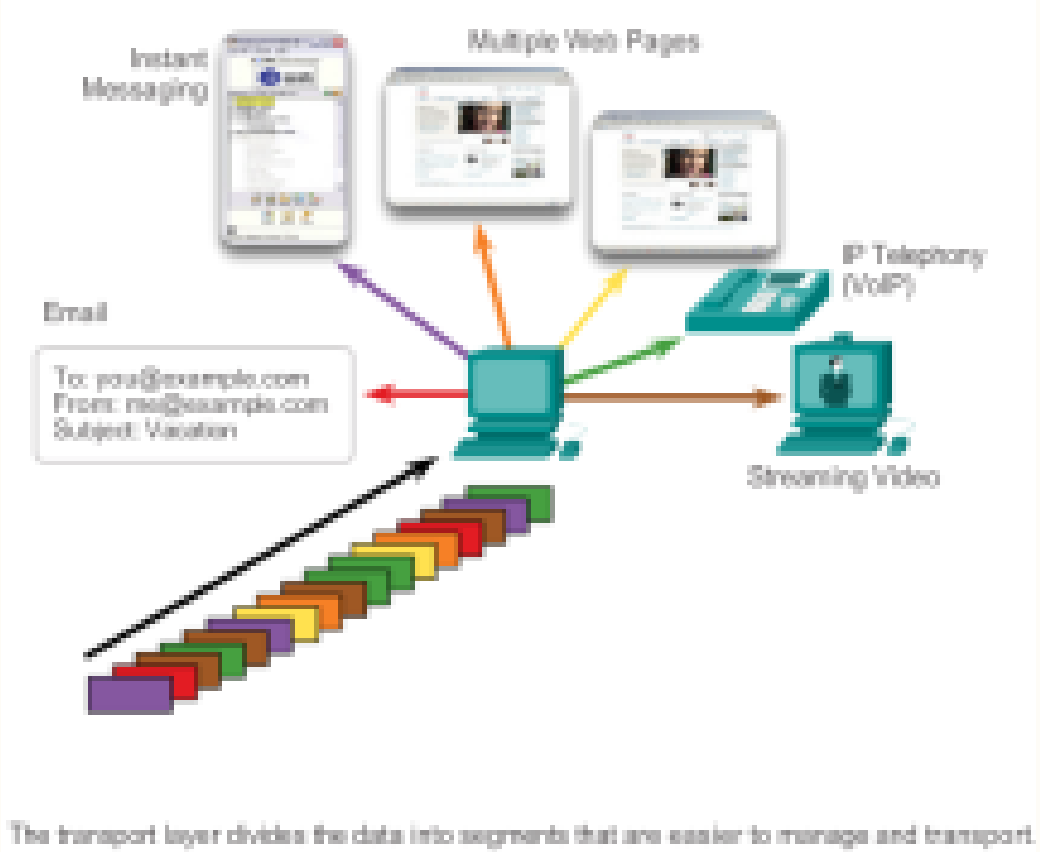
- Membagi data aplikasi menjadi segment atau datagram menjamin data tersebut dikirim dalam batasan (kemampuan) media, dan data dari aplikasi berbeda dapat di –multiplex (digabungkan) menggunakan media yang sama.



Tugas transport layer
→ 3. segmentasi data dan re-assembly

- Oleh protokol pada layer transport, segmen data akan di-enkapsulasi dengan tambahan header transport yang berisi informasi untuk identifikasi potongan.

- Saat data diterima, tugas layer 4 adalah menyusun ulang potongan tersebut dan mengirimkannya ke aplikasi yang tepat.

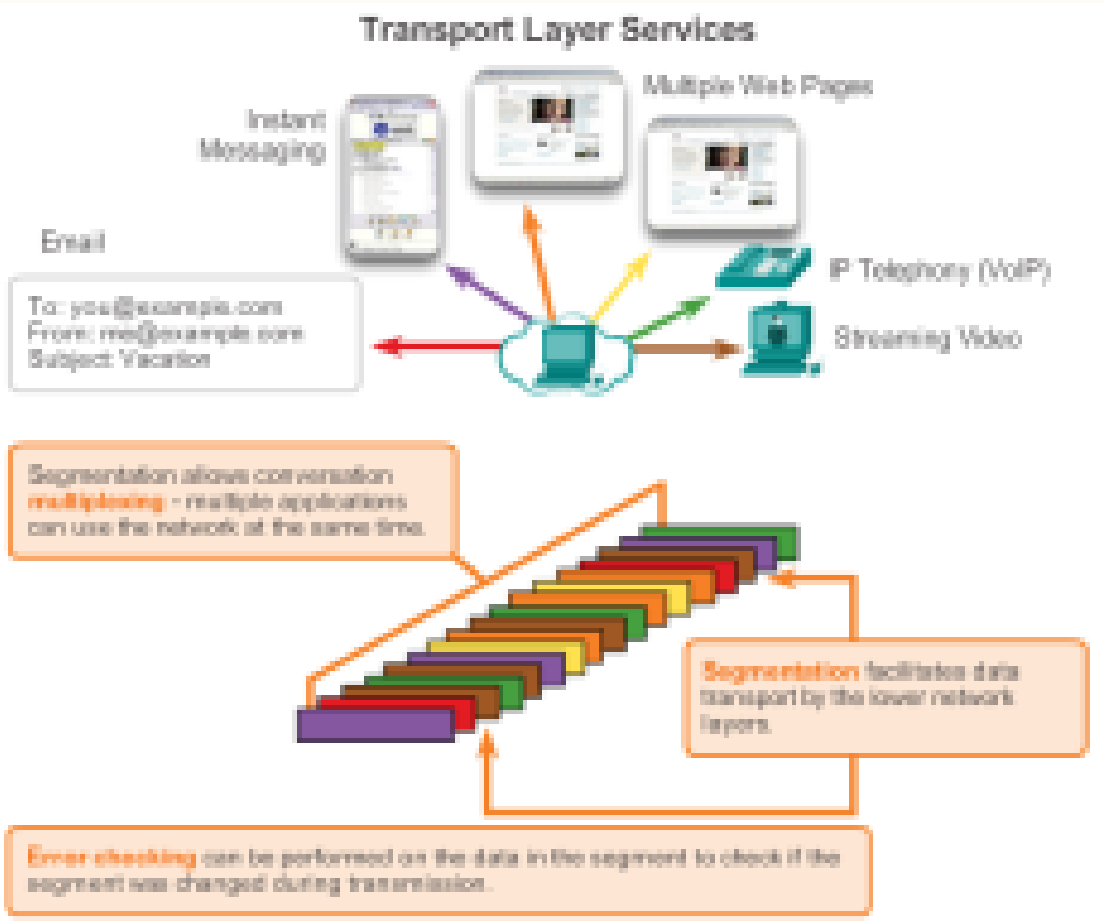


Multiplexing

- Karena protokol pada transport layer melakukan segmentasi, maka data yang dikirim berupa potongan yang lebih kecil yang lebih manageable, dan memungkinkan optimalisasi media dengan metode multiplexing.

- Multiplexing memungkinkan banyak jenis komunikasi, dari user yang berbeda menggunakan media yang sama secara bersamaan.

- Tanpa segmentasi, maka multiplexing tidak bisa dilakukan , yang berarti hanya satu aplikasi yang bisa mengirim atau menerima data pada suatu saat.



Tugas transport layer
→ 4. Conversation control

– Berikut adalah tugas tambahan pada layer 4 **jika diperlukan** , karena tidak setiap aplikasi memerlukan fitur dibawah ini:

1. Establishing a Session
   Menjamin aplikasi siap menerima data dengan cara membuka session dan melakukan manajemen session

2. Reliable Delivery
   Protokol pada Layer 4 bertanggung jawab menjamin setiap potongan (segmen) mencapai tujuan dan bila diperlukan melakukan transmisi ulang

Tugas transport layer
→ 4. Conversation control

– Berikut adalah tugas tambahan pada layer 4 **jika diperlukan** , karena tidak setiap aplikasi memerlukan fitur dibawah ini:

3. Same Order Delivery
   Karena jalur yang ditempuh dapat berbeda-beda, maka layer 4 harus menjamin bahwa segmen-segmen tersebut dapat disusun ulang dalam urutan yang tepat

4. Flow Control
   Jaringan memiliki sumber daya terbatas (seperti memory / bandwith), beberapa jenis protokol layer 4 dapat mengurangi kecepatan kirim bila kondisi kelebihan beban terdeteksi. Hal ini mengurangi kemungkinan hilangnya segmen data karena kondisi jaringan yang penuh.

Jenis Transport layer protocol berdasarkan reliabilitas

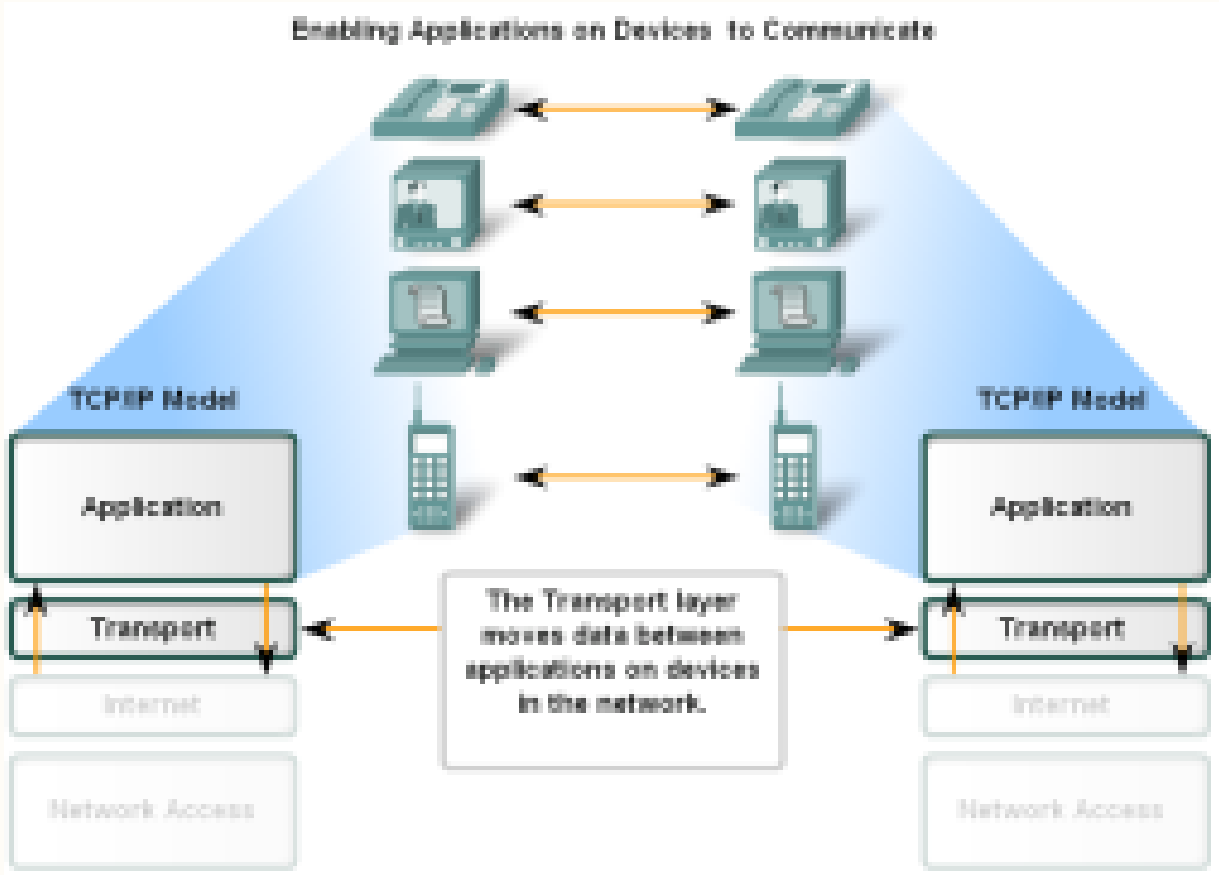
- Perbedaan aplikasi dan pemanfaatan data membedakan metode penjaminan reliabilitas pengiriman. Ada dua jenis utama, yaitu :

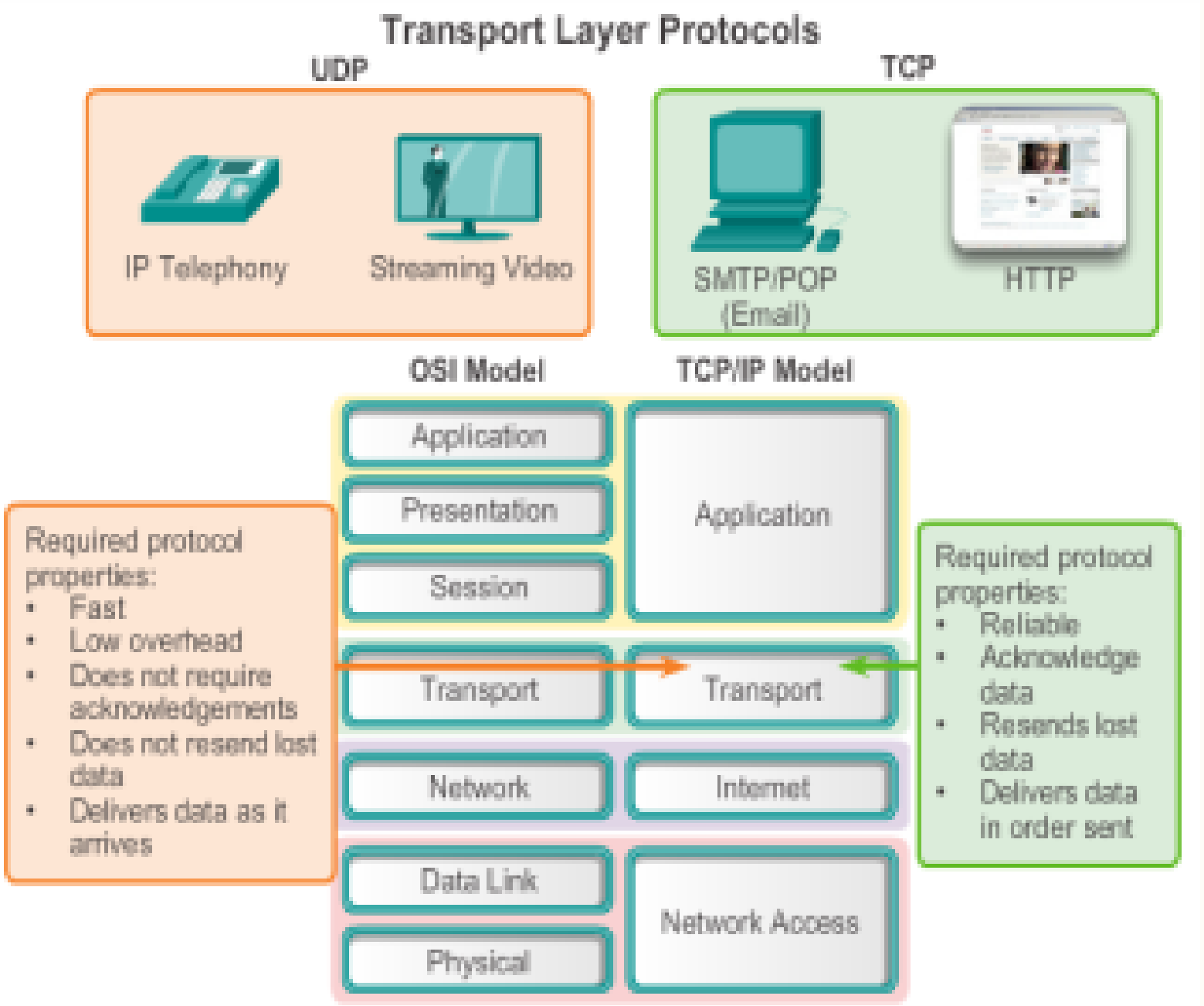
1. Reliable- ada acknowledgement- pengiriman data ulang – penyusunan ulang
2. Cepat – beban kecil – tidak perlu acknowledgement – tidak perlu resent – tanpa penyusunan ulang

- Jenis pertama cocok untuk data sensitif yang harus lengkap seperti data transaksi, email, web, dsb
- Jenis kedua cocok untuk data yang bersifat streaming seperti Voip dan Streaming

contoh protokol Transport layer

- Ada banyak protokol yang terdapat pada layer ini : ATP (Apple Talk Transaction Protocol), CUDP (Cyclic UDP), DCCP (Datagram Congestion Control Protocol), FCP (Fiber Channel Protocol), dll,
- Namun yang digunakan terbanyak dan de-facto pada jaringan komputer, khususnya LAN, dan internet adalah dari keluarga protokol pada TCP/IP, yang terdiri dari dua jenis:
  1. TCP → Reliabilitas tinggi
  2. UDP →Cepat dan beban rendah



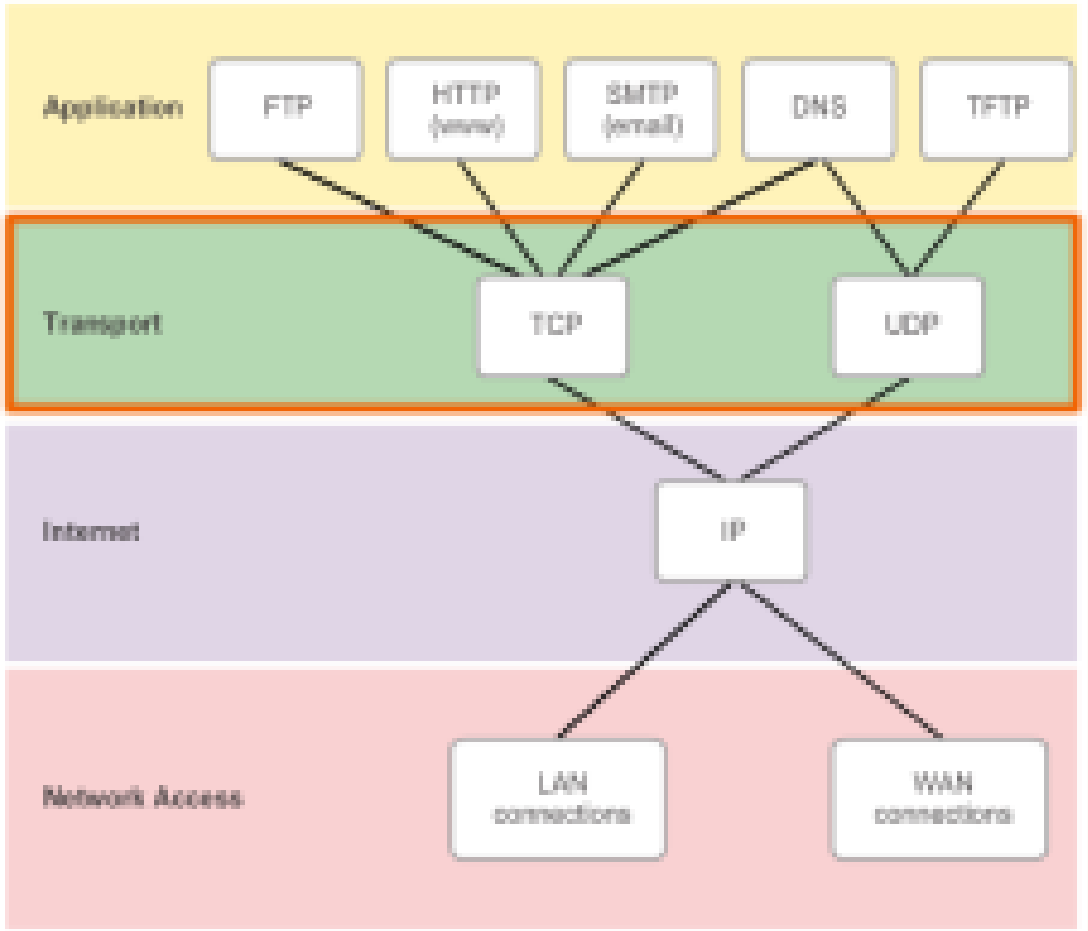


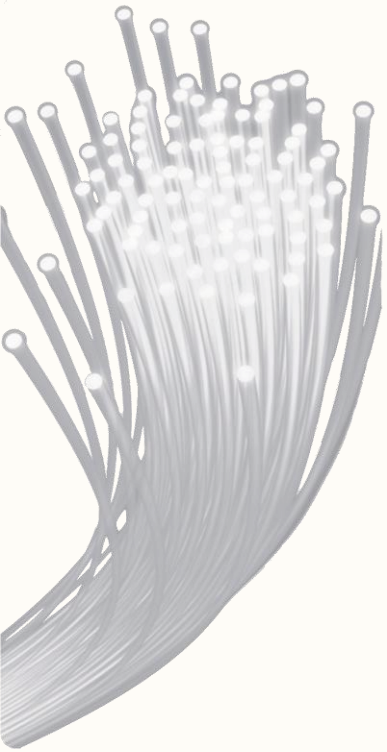
Pilih TCP atau UDP ?

Tergantung kebutuhan layanan

Penggunaan protokol TCP dan UDP

- Secara rata-rata, jumlah paket UDP yang lewat berkisar di 14 % dari seluruh paket yang lewat.
- Secara rata-rata, besar data UDP yang lewat berkisar di 6 % dari seluruh data yang lewat.
- Riset oleh CAIDA, data tahun 2002-2009 di USA dan swedia https://www.caida.org/research/traffic-analysis/tcpudpratio/



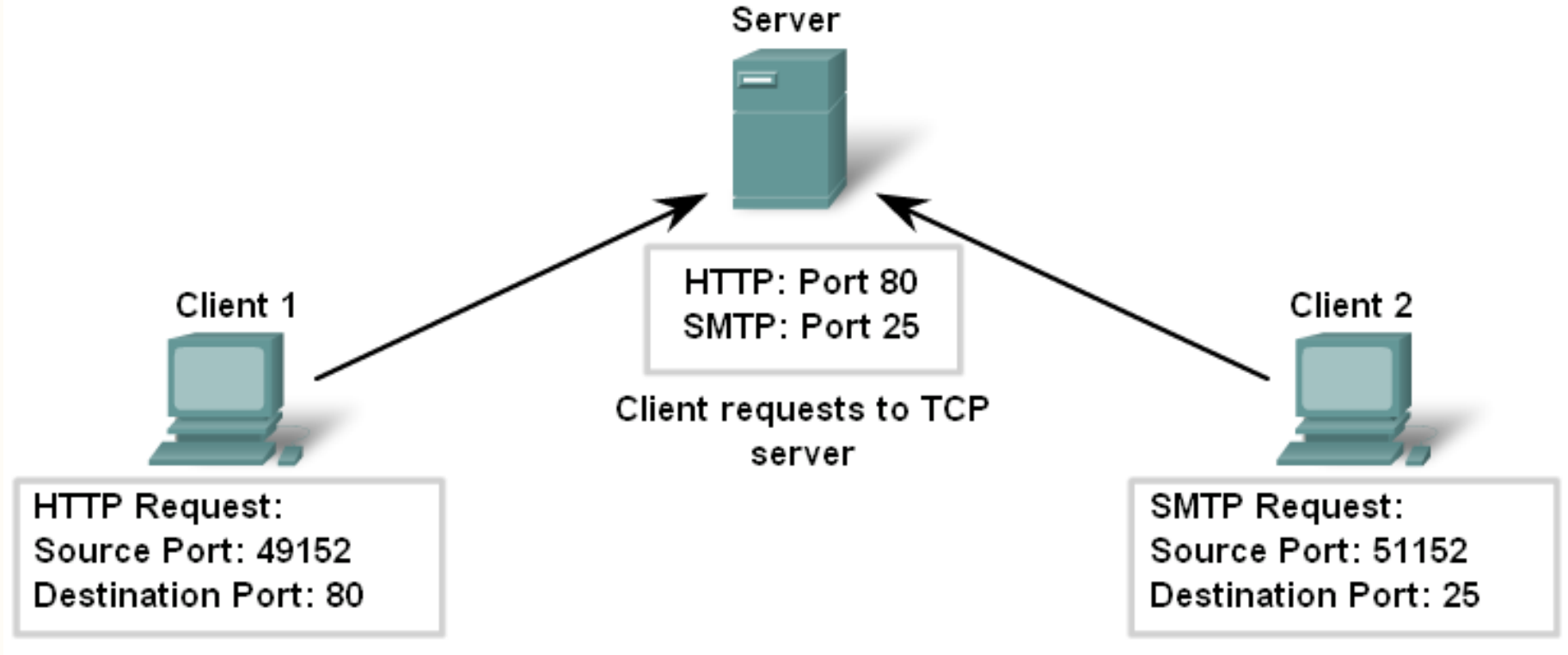


TCP dan UDP

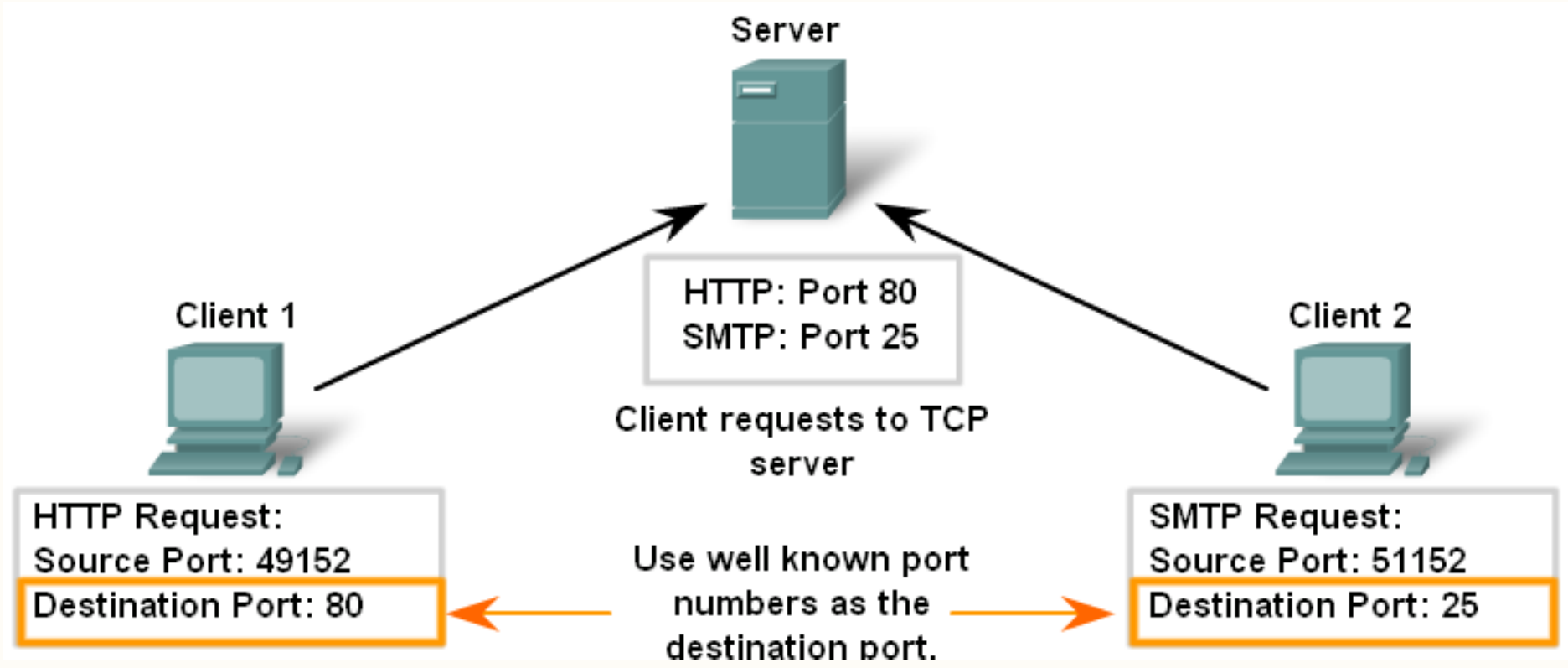
- TCP dan UDP adalah 2 protokol yang bekerja pada layer Transport
- segment atau datagram yang dibentuk akan menyertakan nomor port

- Berikut adalah contoh dari cara kerja protokol TCP dan UDP

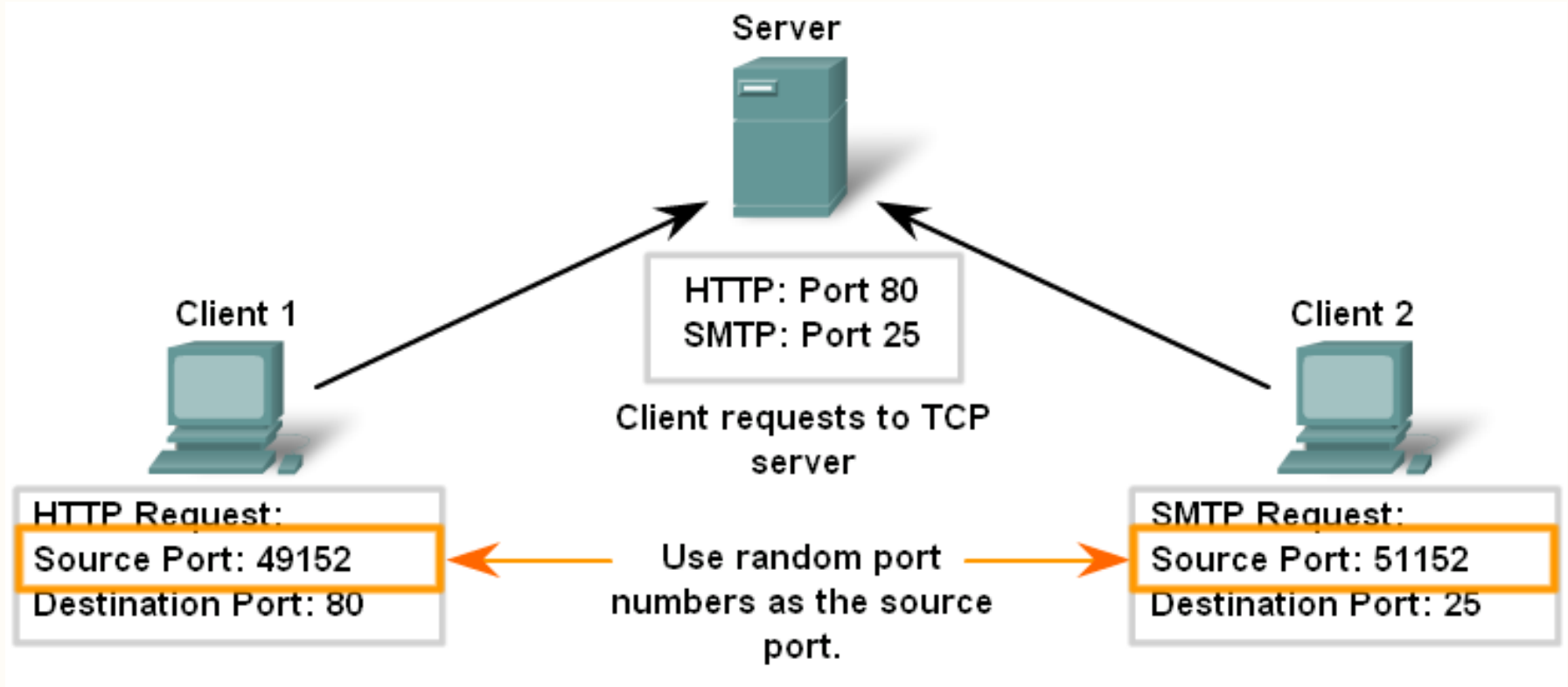
TCP- contoh penggunaan port http dan email



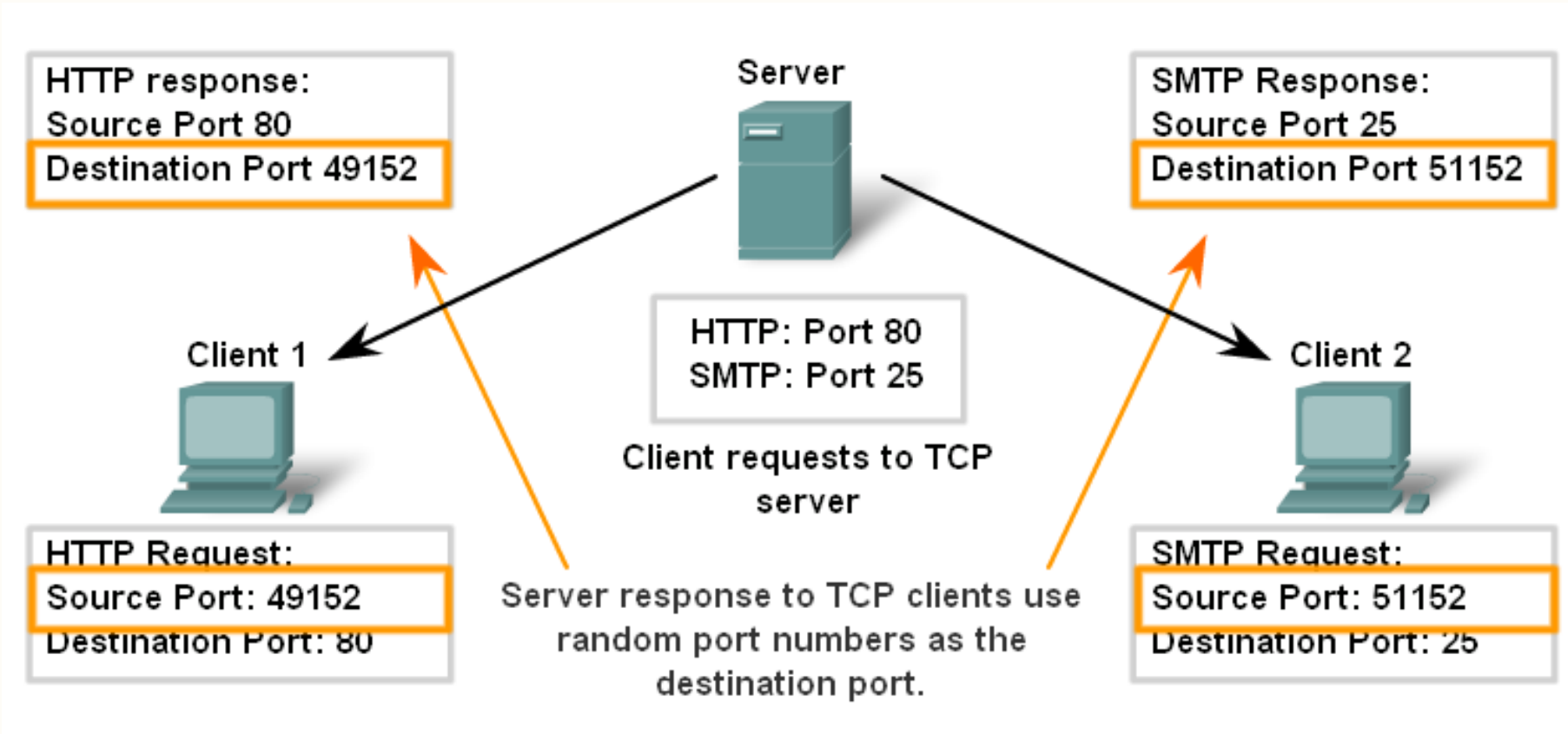
TCP- contoh penggunaan port http dan email



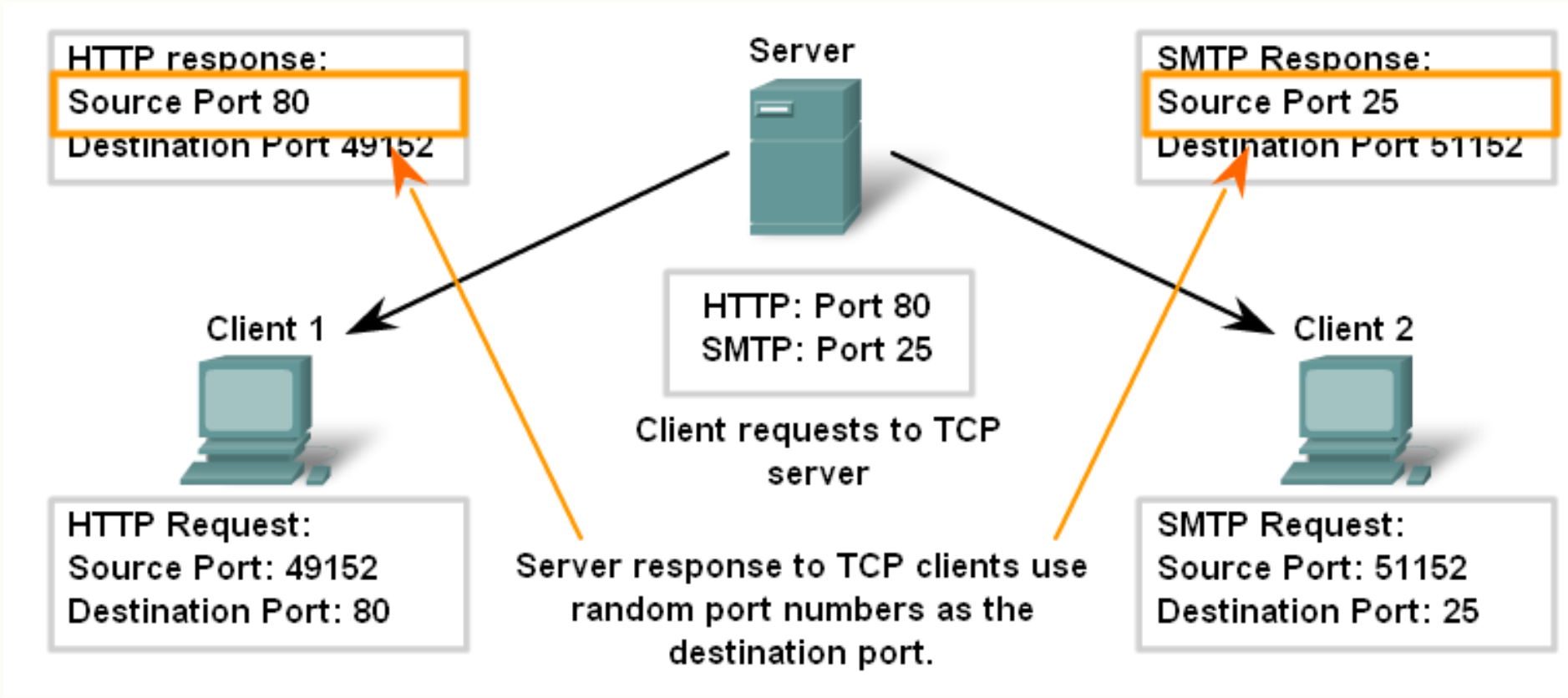
TCP- contoh penggunaan port http dan email



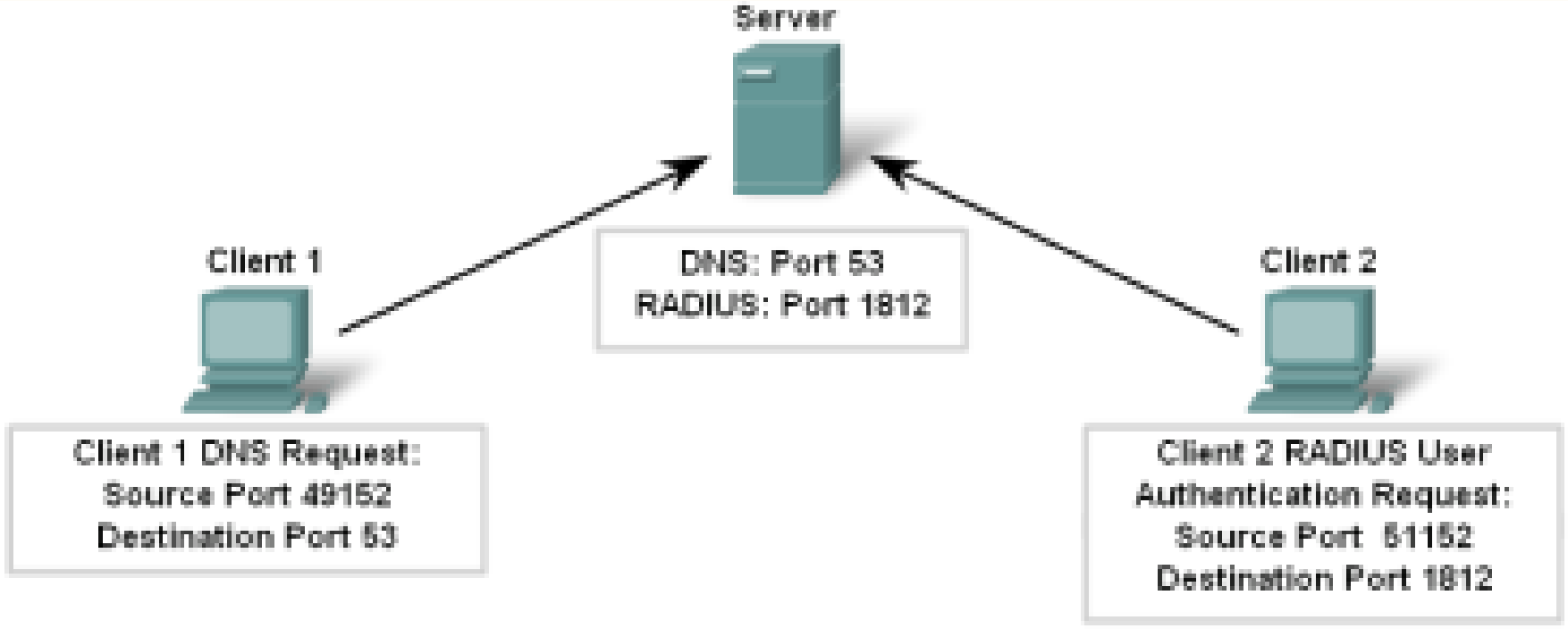
TCP- contoh penggunaan port http dan email



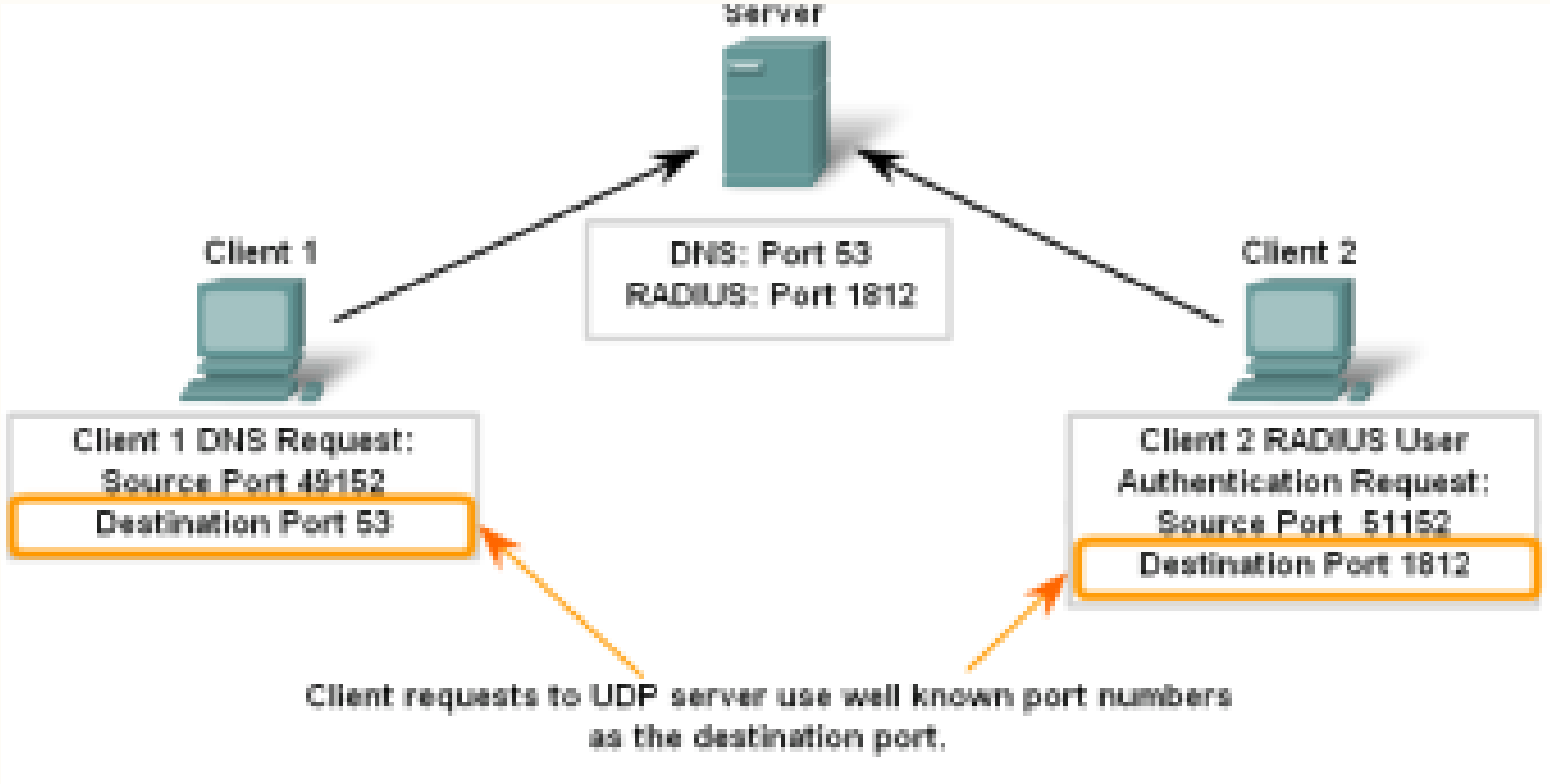
TCP- contoh penggunaan port http dan email



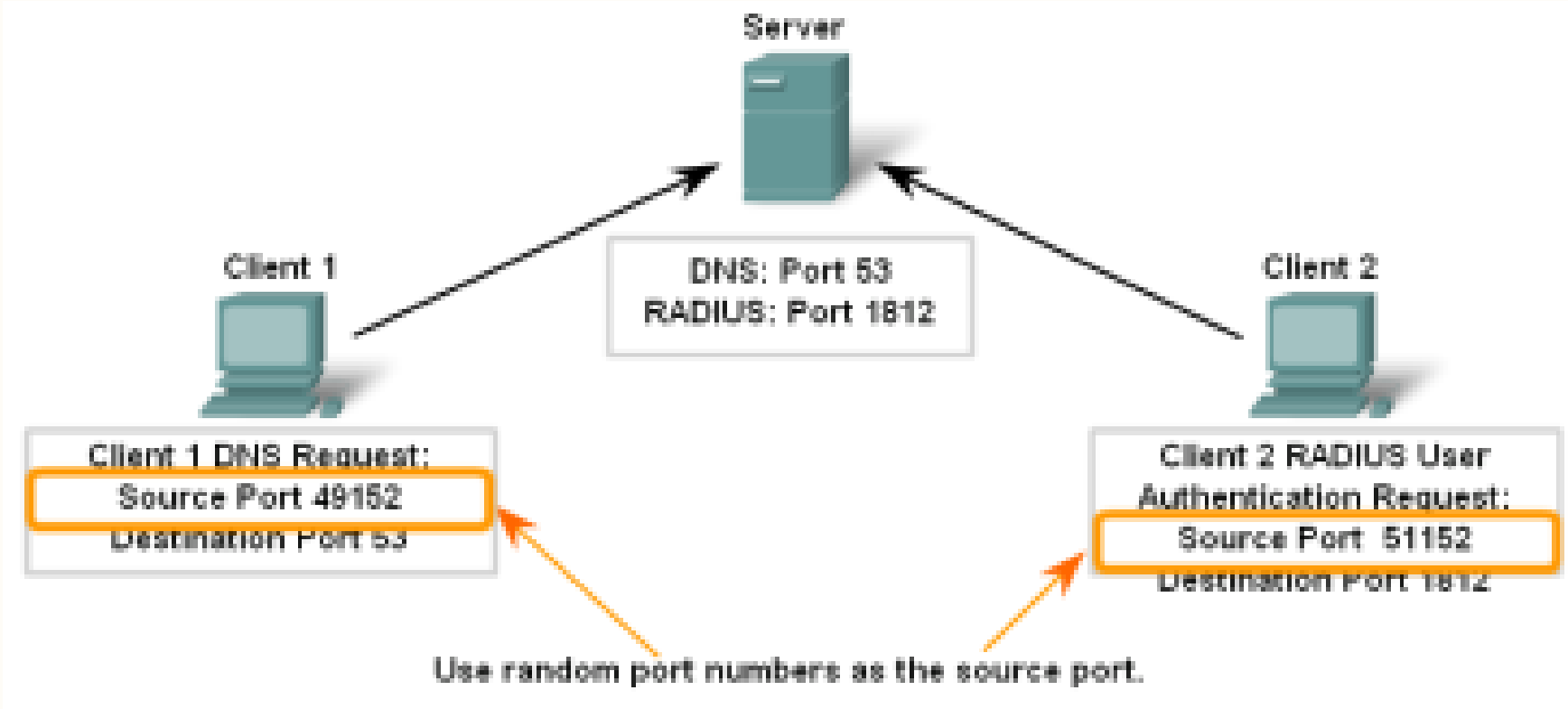
UDP- contoh penggunaan port DNS dan radius



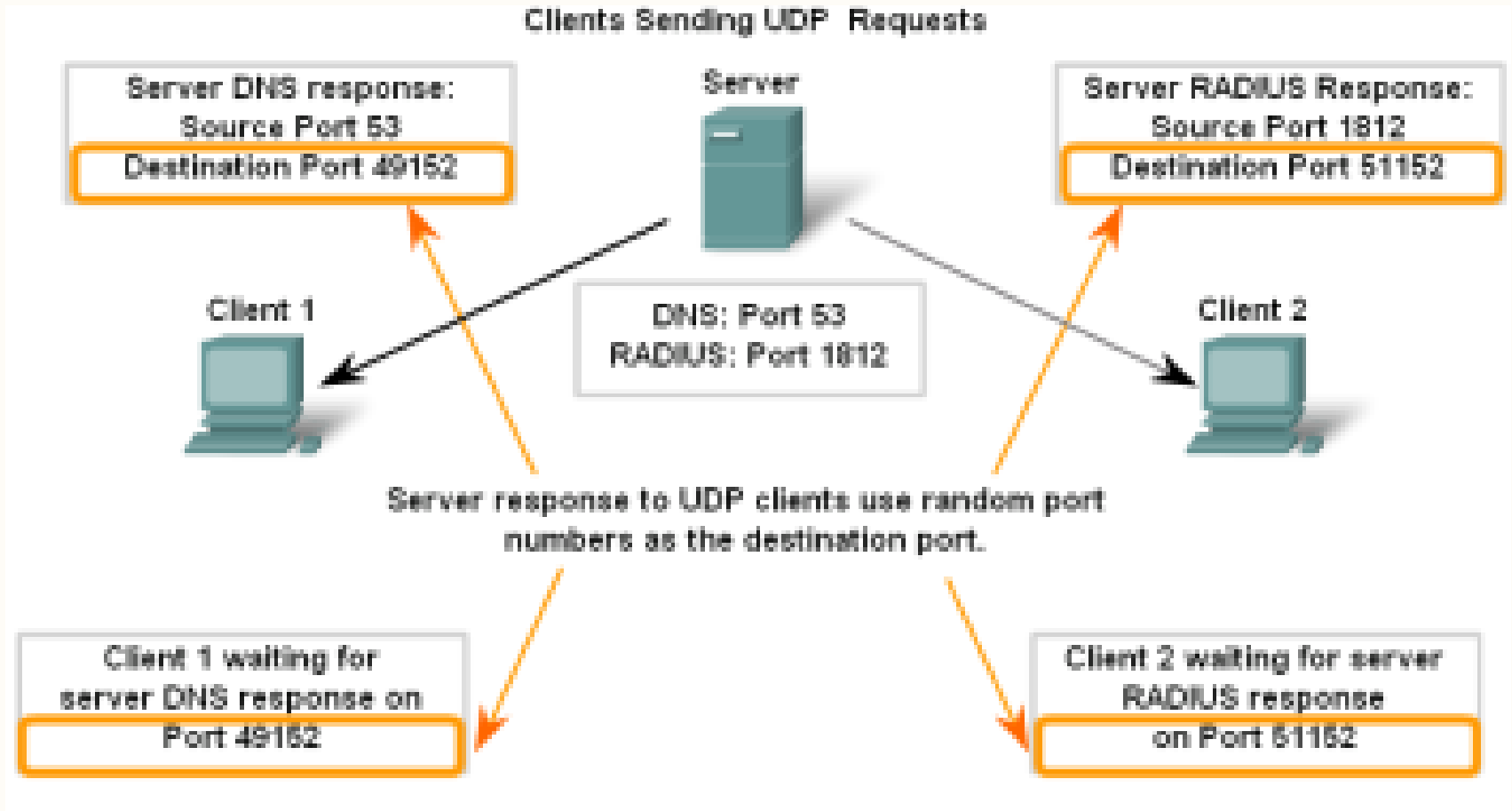
UDP- contoh penggunaan port DNS dan radius



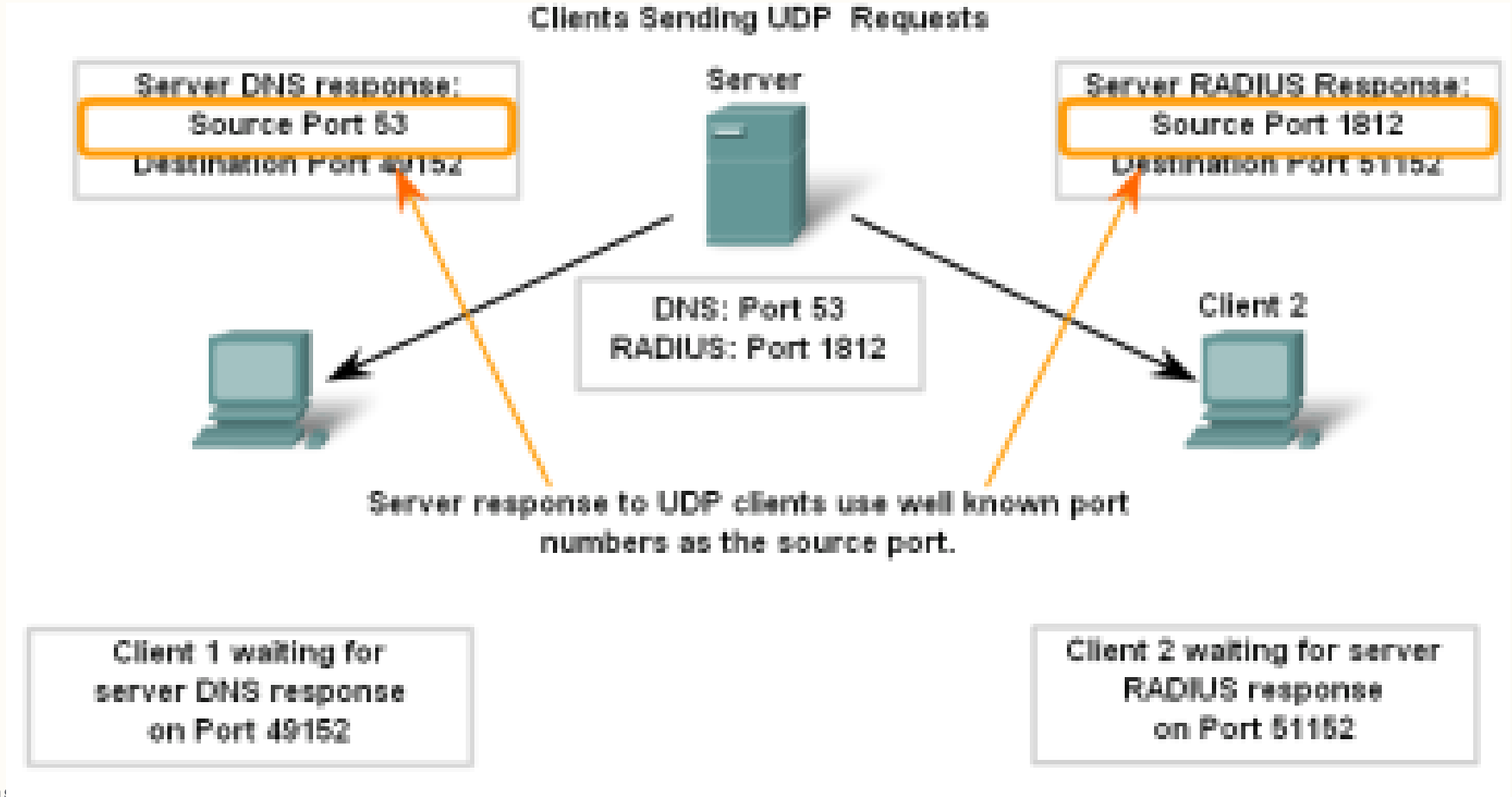
UDP- contoh penggunaan port DNS dan radius



UDP- contoh penggunaan port DNS dan radius

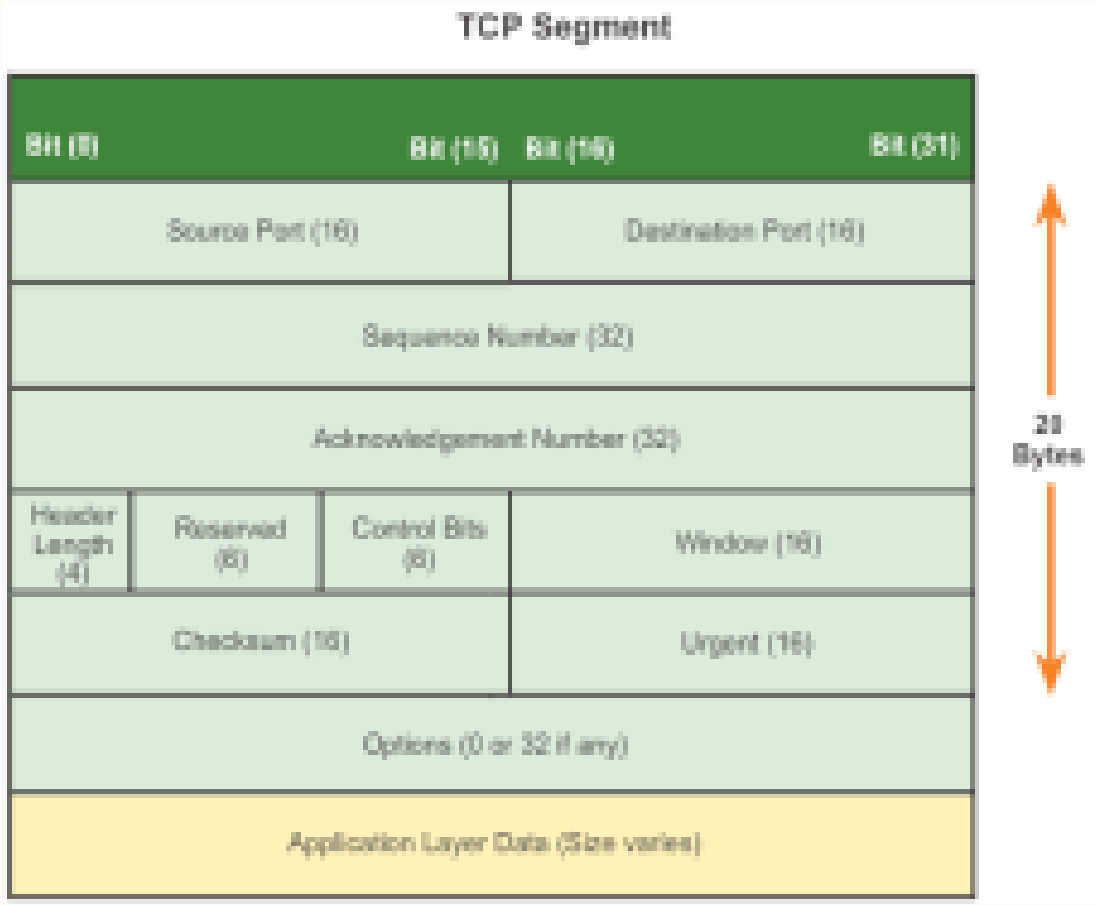


UDP- contoh penggunaan port DNS dan radius

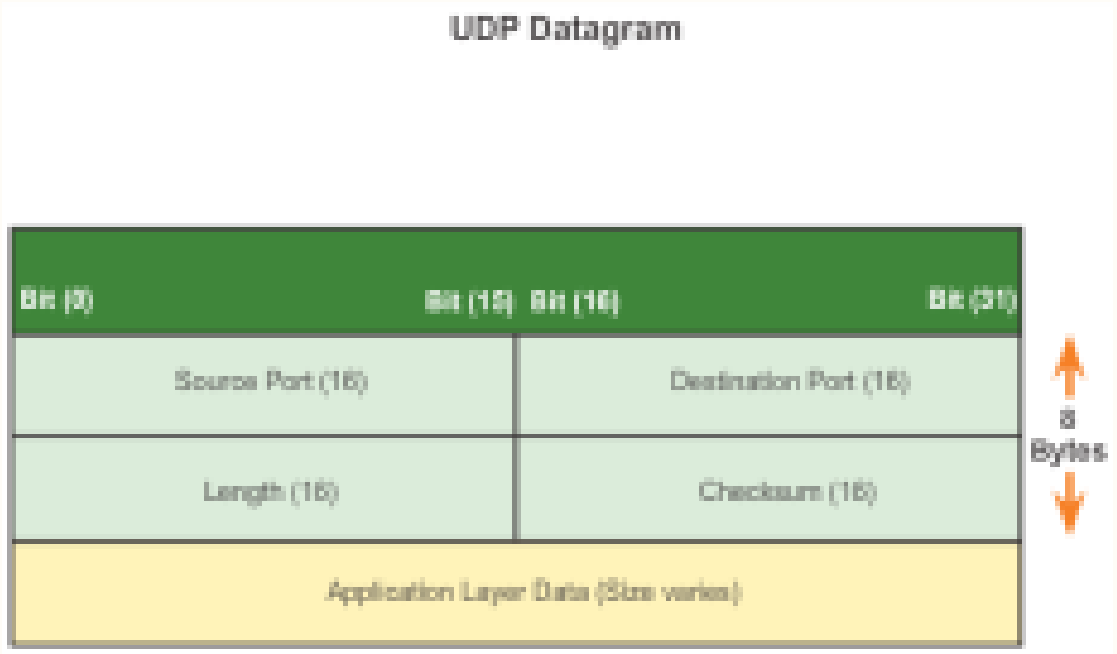


Protokol layer 4

Berikut adalah struktur header dari protocol TCP:

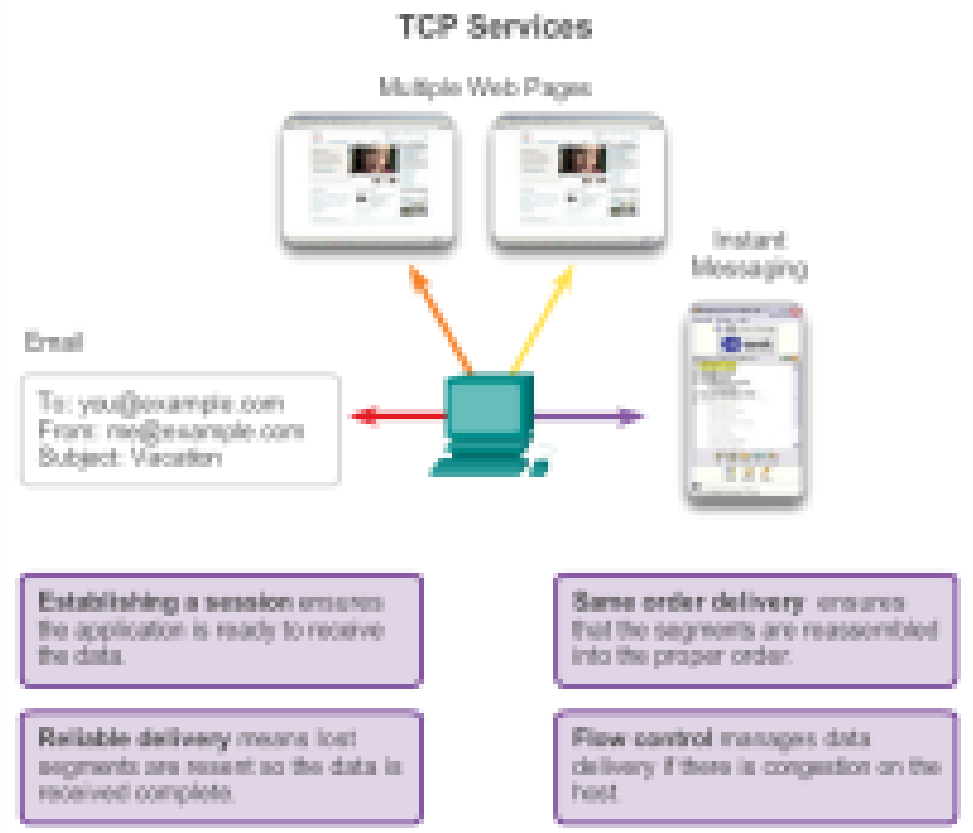


Berikut adalah struktur header dari protocol UDP:



Gambaran umum - TCP

- Transmission Control Protocol (TCP)
- Adalah protokol berorientasi koneksi (connection-oriented protocol), dinyatakan pada RFC 793.
- Fungsi tambahan yang didukung adalah:
  1. Establishing a session
  2. Same order delivery
  3. Reliable delivery
  4. Flow control.



Gambaran umum - TCP

- Menambah overhead tambahan untuk fungsionalitas diatas, dimana tiap segment TCP memiliki 20 byte overhead (tambahan) pada data yang di-enkapsulasi

- Contoh aplikasi yang menggunakan adalah web Browsers, E-mail client, file Transfers

Fungsi tambahan TCP
1.Establishing a session

- Sebelum memulai komunikasinya, TCP akan membuka sesi koneksi terlebih dahulu
- Sesudah komunikasi selesai, TCP akan menutup sesi koneksi
- Detil proses buka dan tutup sesi koneksi dibahas pada slide-slide berikutnya

Fungsi tambahan TCP
2.Same order delivery

- Karena dapat tersedia banyak jalur menuju tujuan, data dapat tiba dengan urutan yang tidak teratur.

- TCP memiliki nomor segment yang memungkinkan proses pengurutan ulang potongan segment menjadi data utuh kembali

Fungsi tambahan TCP
3.Reliable delivery

- Untuk menjamin reliabilitas data yang dikirim, protokol pada layer 4 harus melakukan operasi berikut:
  1. tracking transmitted data
  2. acknowledging received data
  3. retransmitting any unacknowledged data

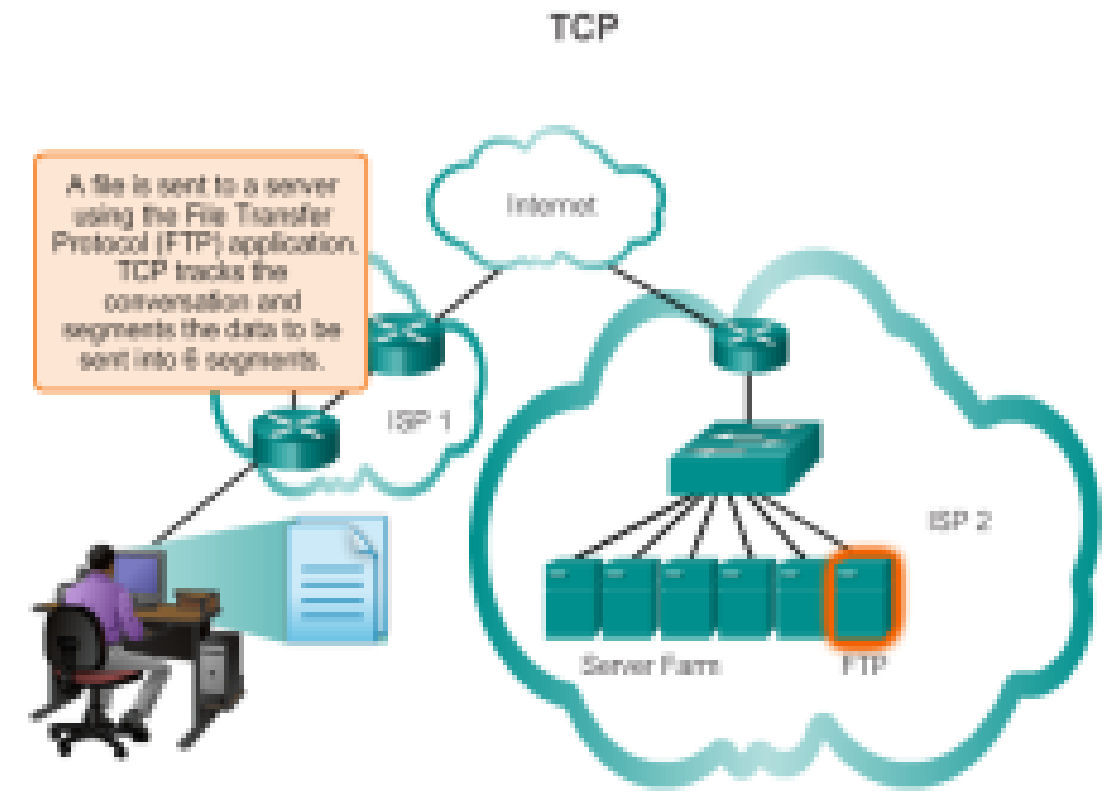
- Untuk melakukan manajemen tersebut, digunakan informasi yang terdapat pada header layer 4, dan pengiriman control data melalui jaringan.
- Hal ini merupakan beban tambahan bagi bandwith jaringan, dan tidak selalu dibutuhkan.

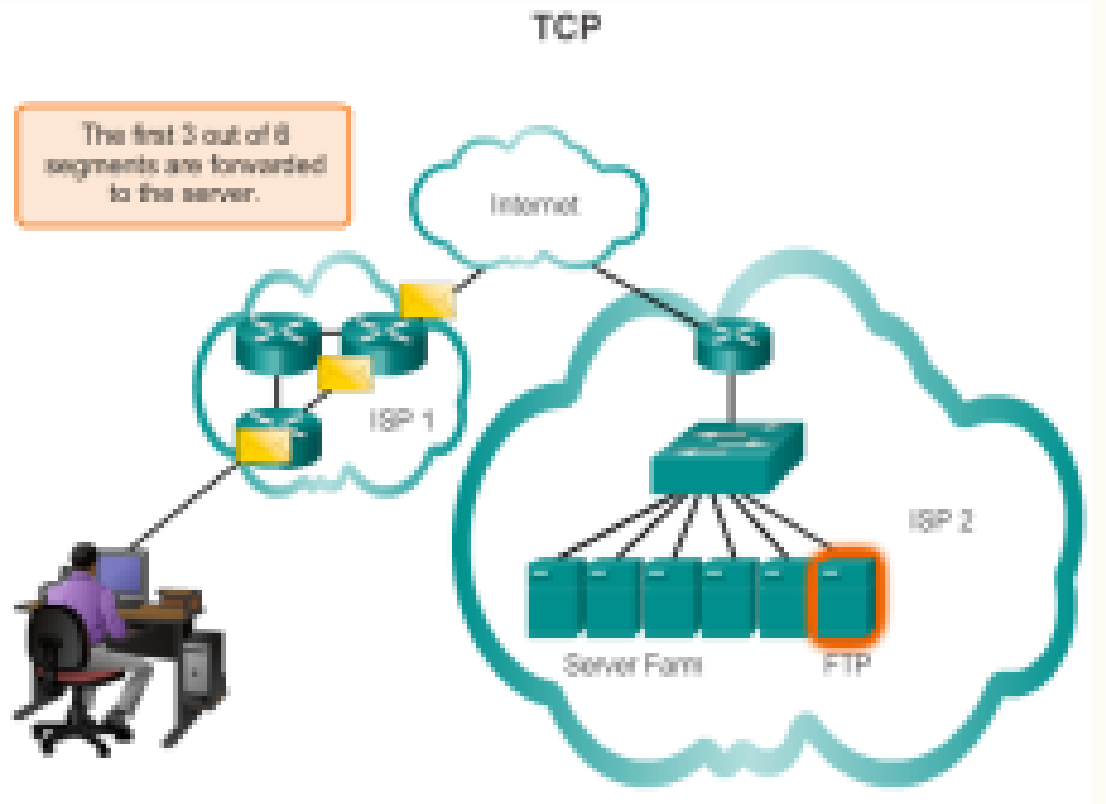
Fungsi tambahan TCP
4.FLOW control

- Jaringan memiliki sumber daya terbatas, seperti memory, bandwidth dan kemampuan proses.

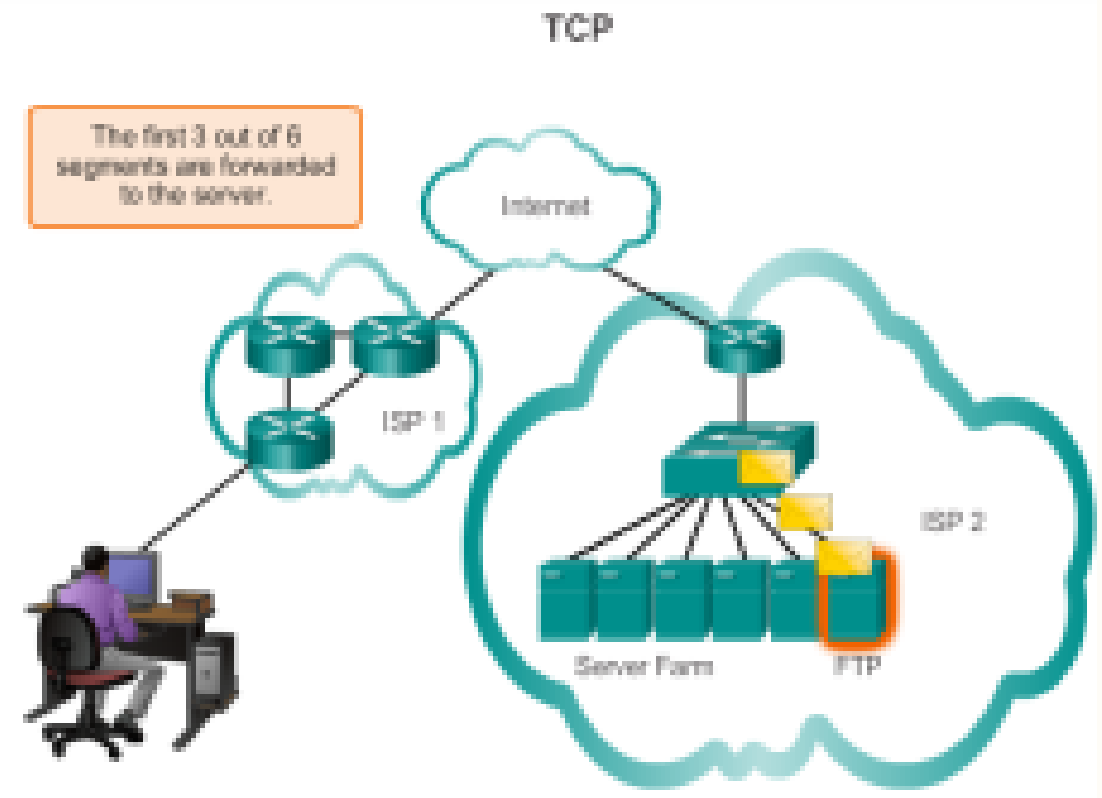
- TCP merupakan protokol yang “aware” terhadap kondisi ini, dan mendukung proses pengaturan kecepatan kirim secara dinamis, sehingga kecepatan kirim dapat disesuaikan dengan kemampuan media.
- Flow control dapat mencegah kehilangan segment pada jaringan dan mengurangi jumlah retransmisi, juga secara tidak langsung mencegah “overtaxing” sumber daya jaringan.

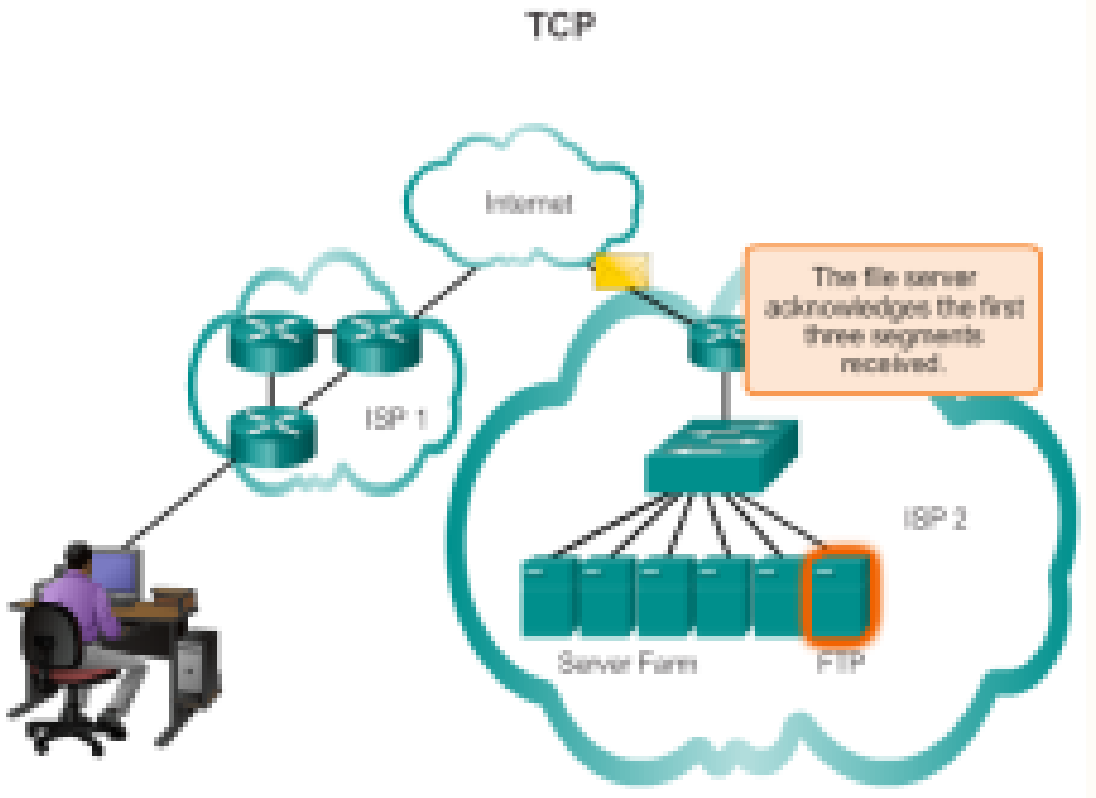
Contoh komunikasi TCP



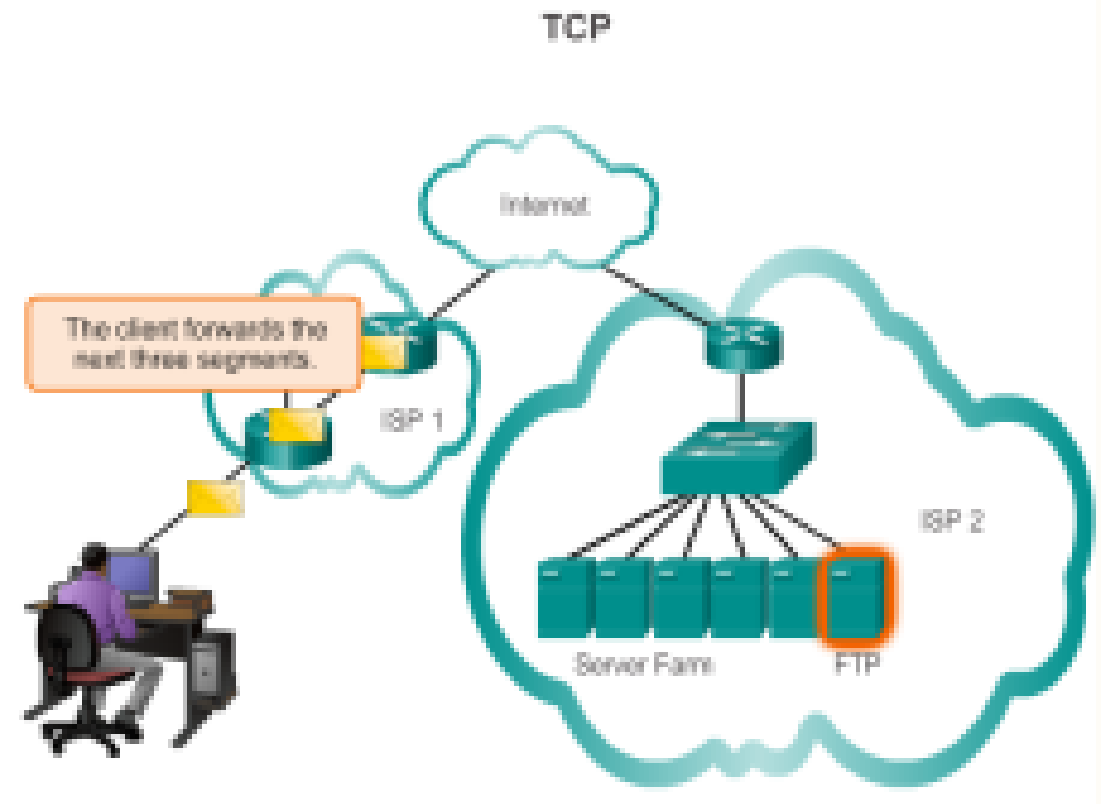


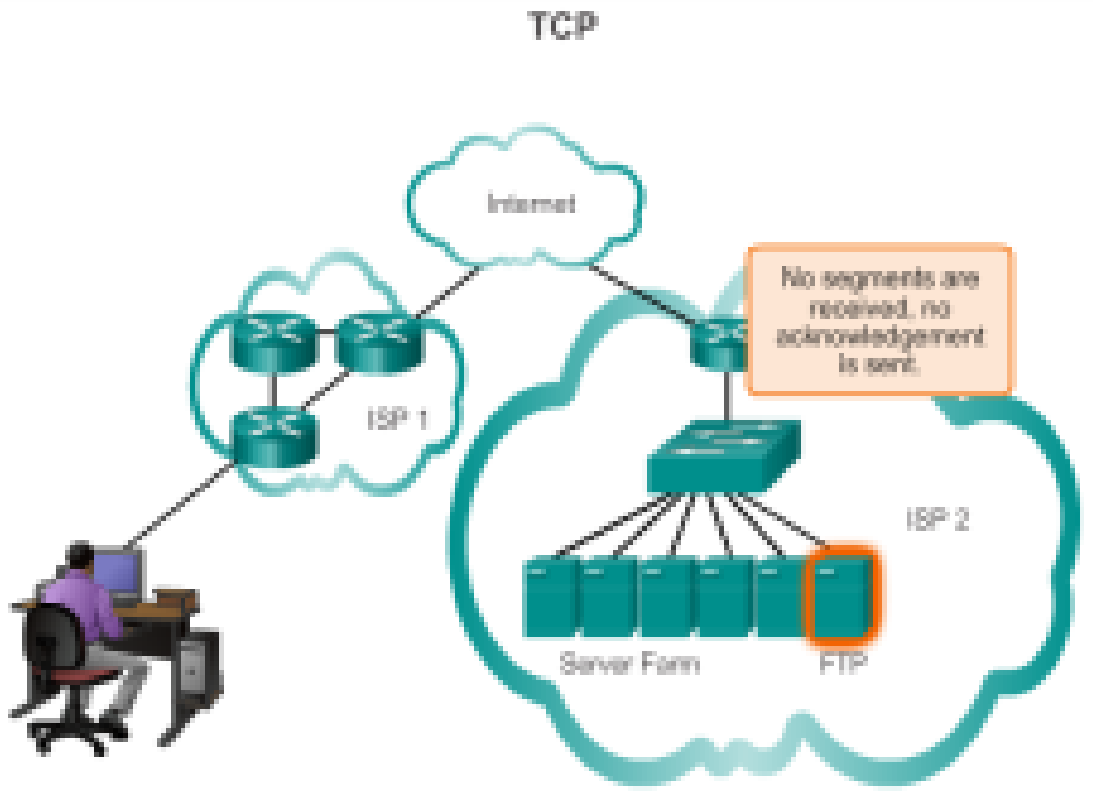
Contoh komunikasi TCP



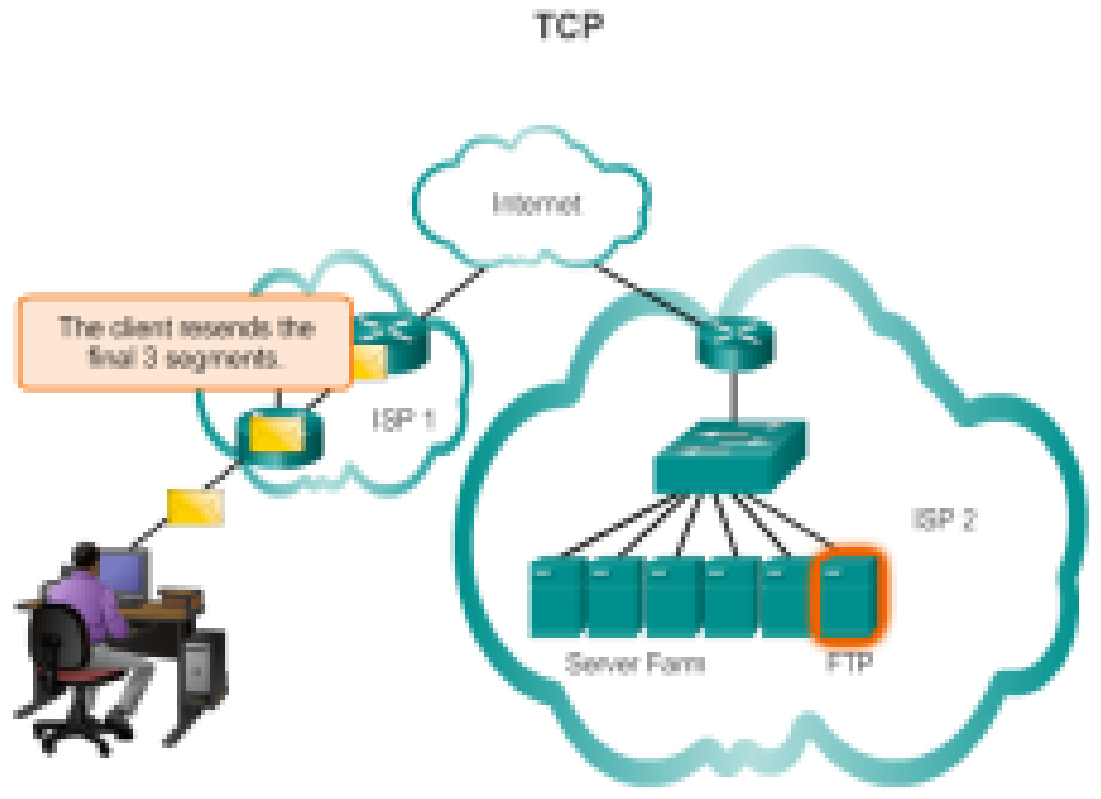


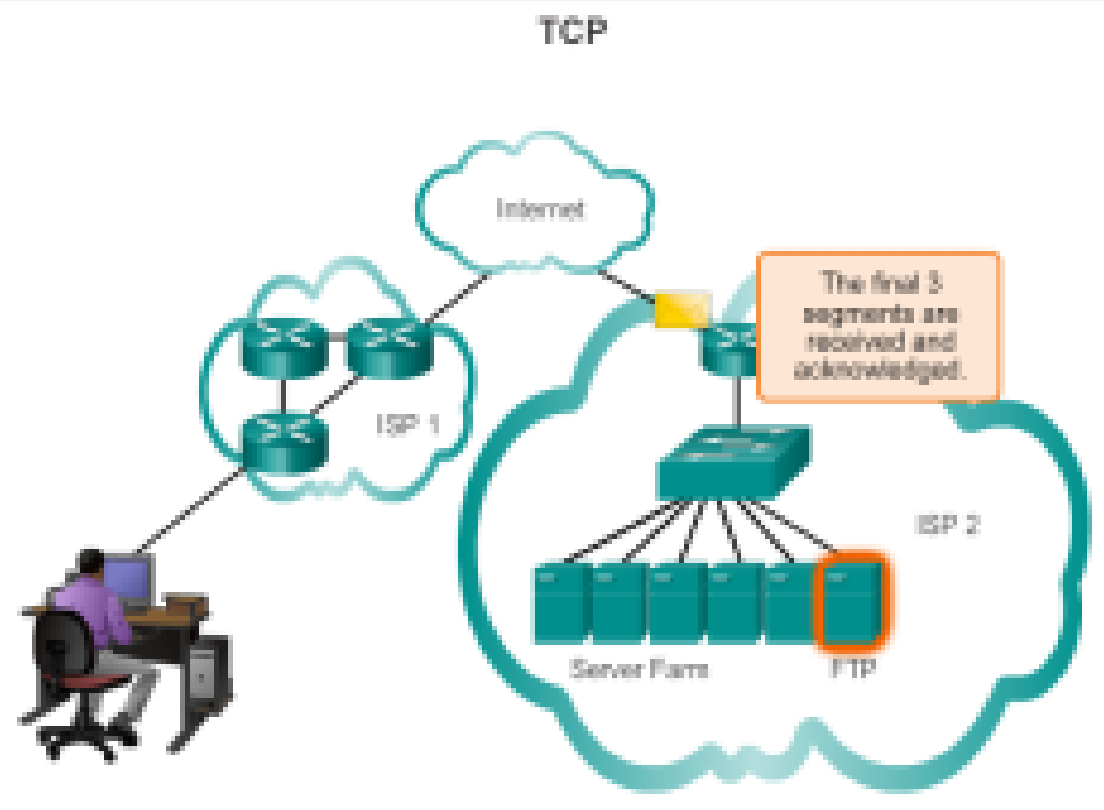
Contoh komunikasi TCP





Contoh komunikasi TCP





Header TCP

Nama field	Ukuran	Keterangan
Source Port	2 byte (16 bit)	Mengindikasikan sumber protokol lapisan aplikasi yang mengirimkan segmen TCP yang bersangkutan. Gabungan antara *field* **Source IP Address** dalam *header IP*dan *field* **Source Port** dalam *field header TCP* disebut juga sebagai ***socket*sumber**, yang berarti sebuah alamat global dari mana segmen dikirimkan. Lihat juga Port TCP.
Destination Port	2 byte (16 bit)	Mengindikasikan tujuan protokol lapisan aplikasi yang menerima segmen TCP yang bersangkutan. Gabungan antara field Destination IP Address dalam header IP dan field Destination Port dalam field header TCP disebut juga sebagai ***socket* tujuan**, yang berarti sebuah alamat global ke mana segmen akan dikirimkan.

Header TCP

Nama field	Ukuran	Keterangan
Sequence Number	4 byte (32 bit)	Mengindikasikan nomor urut dari oktet pertama dari data di dalam sebuah segmen TCP yang hendak dikirimkan. Field ini harus selalu diset, meskipun tidak ada data (payload) dalam segmen. Ketika memulai sebuah sesi koneksi TCP, segmen dengan flag SYN (Synchronization) diset ke nilai 1, field ini akan berisi nilai Initial Sequence Number (ISN). Hal ini berarti, oktet pertama dalam aliran byte (byte stream) dalam koneksi adalah ISN+1.
Acknowledgment Number	4 byte (32 bit)	Mengindikasikan nomor urut dari oktet selanjutnya dalam aliran byte yang diharapkan oleh untuk diterima oleh pengirim dari si penerima pada pengiriman selanjutnya. Acknowledgment number sangat dipentingkan bagi segmen-segmen TCP dengan flag ACK diset ke nilai 1.

Header TCP

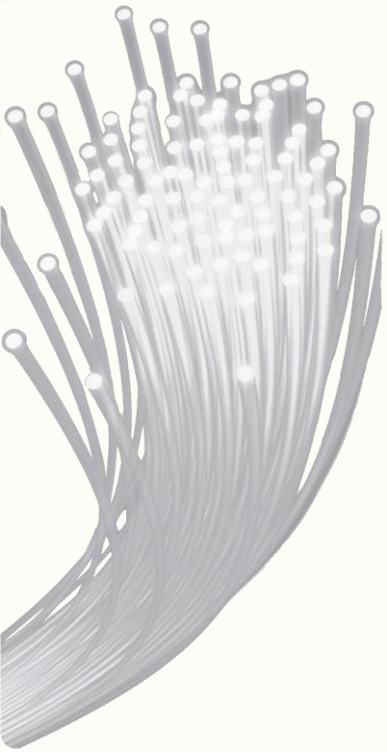
Data Offset	4 bit	Mengindikasikan di mana data dalam segmen TCP dimulai. Field ini juga dapat berarti ukuran dari header TCP. Seperti halnya field **Header Length** dalam header IP, field ini merupakan angka dari word 32-bit dalam header TCP. Untuk sebuah segmen TCP terkecil (di mana tidak ada opsi TCP tambahan), field ini diatur ke nilai 0x5, yang berarti data dalam segmen TCP dimulai dari oktet ke 20 dilihat dari permulaan segmen TCP. Jika field Data Offset diset ke nilai maksimumnya (2^4=16) yakni 15, header TCP dengan ukuran terbesar dapat memiliki panjang hingga 60 byte.
Reserved	6 bit	Direservasikan untuk digunakan pada masa depan. Pengirim segmen TCP akan mengeset bit-bit ini ke dalam nilai 0.

Header TCP

Flags	6 bit	Mengindikasikan flag-flag TCP yang memang ada enam jumlahnya, yang terdiri atas: URG (Urgent), ACK (Acknowledgment), PSH (Push), RST (Reset), SYN (Synchronize), dan FIN (Finish).
Window	2 byte (16 bit)	Mengindikasikan jumlah byte yang tersedia yang dimiliki oleh buffer host penerima segmen yang bersangkutan. Buffer ini disebut sebagai Receive Buffer, digunakan untuk menyimpan byte stream yang datang. Dengan mengimbuhkan ukuran window ke setiap segmen, penerima segmen TCP memberitahukan kepada pengirim segmen berapa banyak data yang dapat dikirimkan dan disangga dengan sukses. Hal ini dilakukan agar si pengirim segmen tidak mengirimkan data lebih banyak dibandingkan ukuran Receive Buffer. Jika tidak ada tempat lagi di dalam Receive buffer, nilai dari field ini adalah 0. Dengan nilai 0, maka si pengirim tidak akan dapat mengirimkan segmen lagi ke penerima hingga nilai field ini berubah (bukan 0). Tujuan hal ini adalah untuk mengatur lalu lintas data atau *flow control*.

Header TCP

Checksum	2 byte (16 bit)	Mampu melakukan pengecekan integritas segmen TCP (*header*-nya dan *payload*-nya). Nilai field Checksum akan diatur ke nilai 0 selama proses kalkulasi checksum.
Urgent Pointer	2 byte (16 bit)	Menandakan lokasi data yang dianggap "urgent" dalam segmen.
Options	4 byte (32 bit)	Berfungsi sebagai penampung beberapa opsi tambahan TCP. Setiap opsi TCP akan memakan ruangan 32 bit, sehingga ukuran header TCP dapat diindikasikan dengan menggunakan field Data offset.



TCP session

- TCP adalah protokol yang “sopan” dan membuka sesi koneksi sebelum mengirimkan data
- Sesudah pengiriman selesai, TCP juga akan memutup sesi koneksi
- Sifat kerja ini disebut connection oriented

TCP-
pembukaan dan penutupan sesi

- Untuk membuka sesi koneksi, dilakukan **three-way handshake**
- Tujuan dari handshake ini ialah:
  1. Memastikan bahwa host penerima ada pada jaringan.
  2. Mem-verifikasi bahwa host penerima memiliki service yang aktif dan menerima request pada port yang ingin dituju oleh pengirim.
  3. Menginformasikan host penerima bahwa pengirim ingin membuka sesi komunikasi pada nomor port tersebut.

TCP- three way handshake

Source Port	Dest Port	Sequence No	ACK No	Header Length	Flags	Window Size	TCP Checksum	Urgent Pointer	DATA
← ==========TCP Segme ==========→									

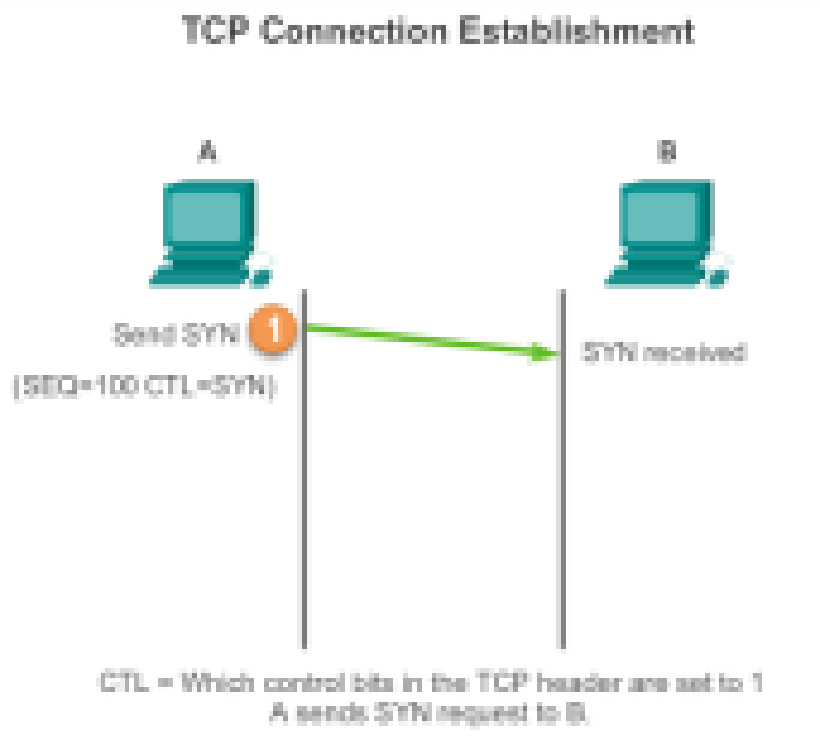
– Untuk lebih memahami proses yang terjadi pada TCP, harus diperhatikan nilai dari flag control info. Yaitu:

1. URG - Urgent pointer field significant
2. ACK - Acknowledgement field significant
3. PSH - Push function
4. RST - Reset the connection
5. SYN - Synchronize sequence numbers
6. FIN- No more data from sender

URG	ACK	PSH	RST	SYN	FIN
0	0	0	0	0	0

– Disebut flag, karena nilainya hanya 1 atau 0 menandakan aktif atau tidaknya control tersebut

TCP-
three way handshake

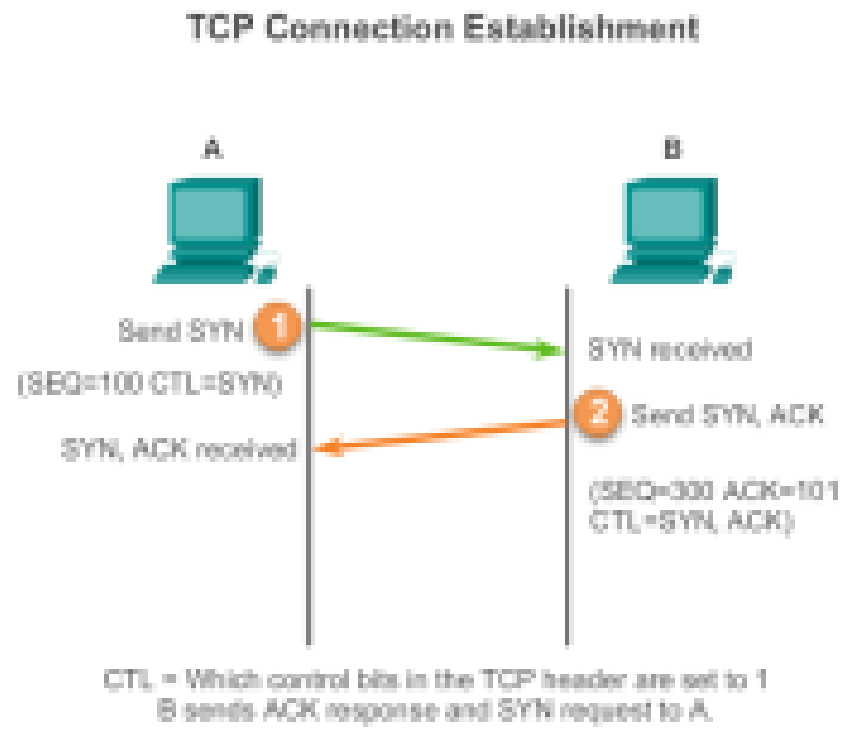


- Berikut ialah langkah kerja three way handshake.
- Host pengirim akan disebut client dan host penerima akan disebut server

1. Client mengirimkan segment yang berisi nomor sequence, yang berfungsi sebagai permohonan pembukaan sesi, ini adalah segment SYN

Source Port	Dest Port	Sequence No	ACK No	Header Length	Flags						Window Size	TCP Checksum	Urgent Pointer	DATA
					URG	ACK	PSH	RST	SYN	FIN				
		100	0		0	0	0	0	1	0				

TCP-
three way handshake

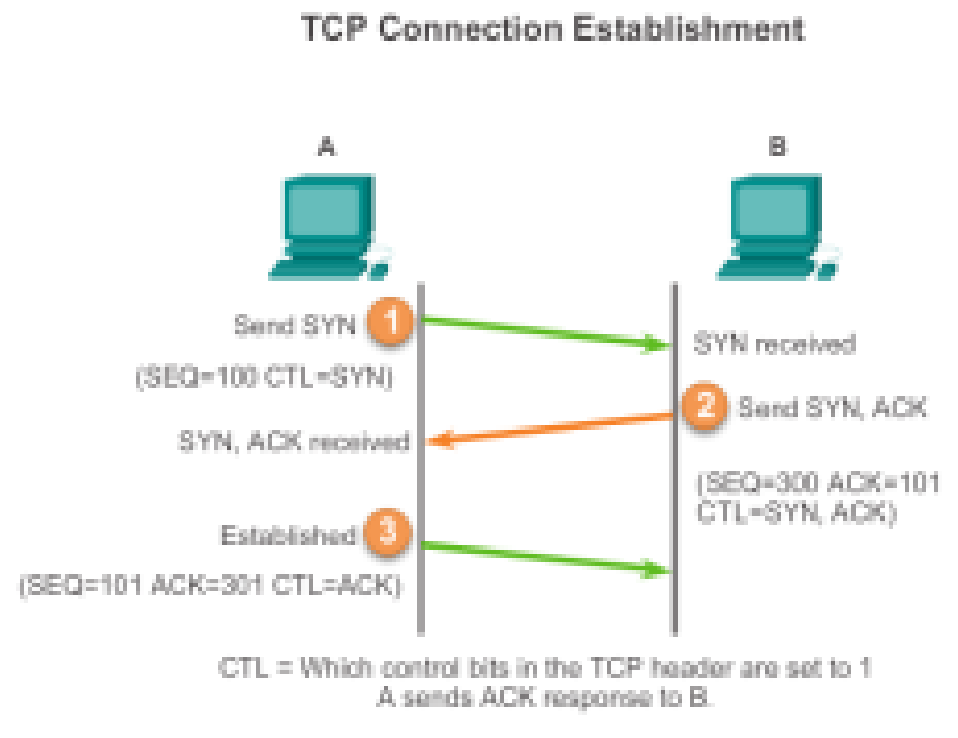


2. Server merespon dengan segment SYN, ACK yang berisi nilai ack (nomor sequence client diterima +1), dan nomor sequence nya sendiri.

   SYN,ACK ini memungkinkan PC A untuk menghubungkan response tersebut dengan segment sebelumnya yang dia kirim.

Source Port	Dest Port	Sequence No	ACK No	Header Length	Flags						Window Size	TCP Checksum	Urgent Pointer	DATA
					URG	ACK	PSH	RST	SYN	FIN				
		300	101		0	1	0	0	1	0				

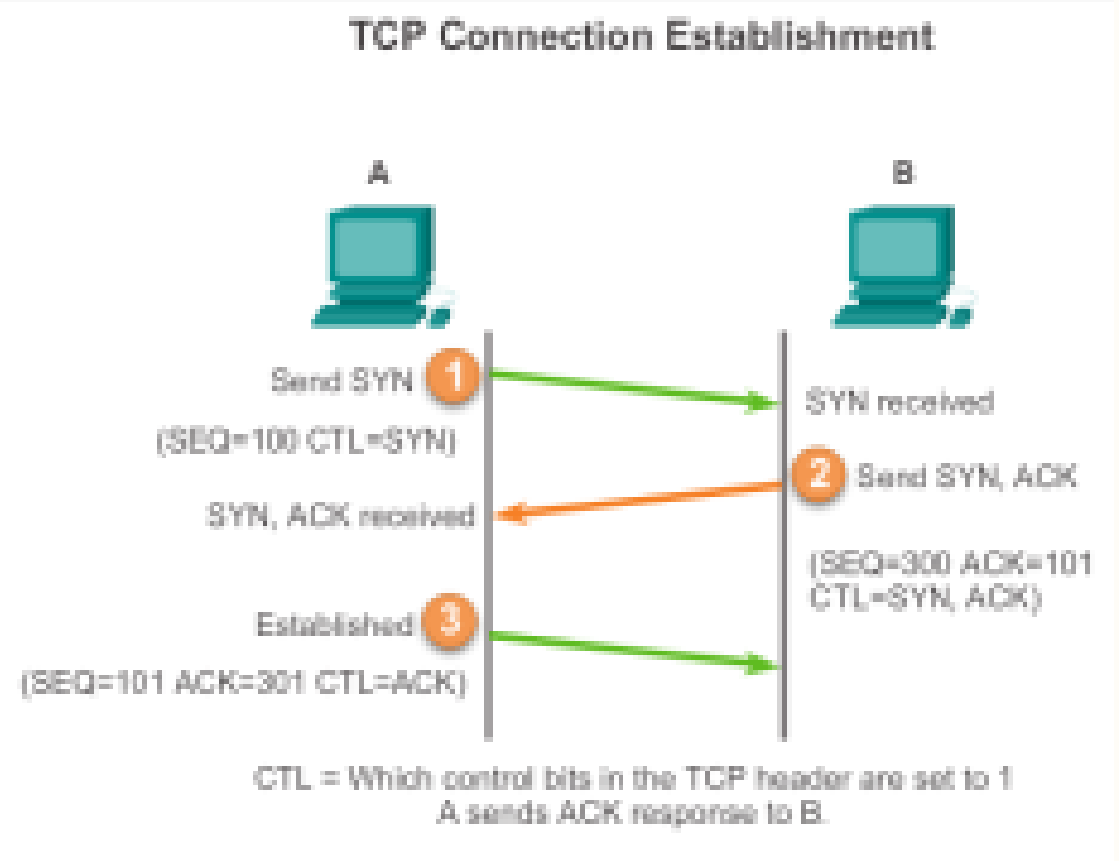
TCP-
three way handshake



3. Client akan merespon dengan ack yang nilai sequencenya +1 dari sequence sebelumnya, dan nilai ack = nilai sequence server + 1

Source Port	Dest Port	Sequence No	ACK No	Header Length	Flags						Window Size	TCP Checksum	Urgent Pointer	DATA
					URG	ACK	PSH	RST	SYN	FIN				
		101	301		0	1	0	0	0	0				

TCP- three way handshake

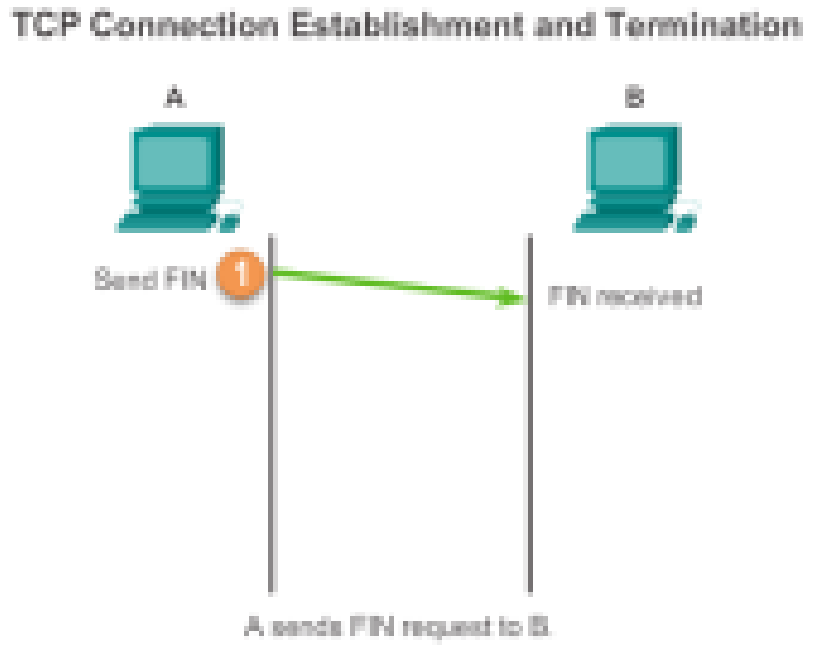


	Source Port	Dest Port	Sequence No	ACK No	Header Length	Flags						Window Size	TCP Checksum	Urgent Pointer	DATA
						URG	ACK	PSH	RST	SYN	FIN				
1			100	0		0	0	0	0	1	0				
2			300	101		0	1	0	0	1	0				
3			101	301		0	1	0	0	0	0				

TCP- penutupan sesi

- Untuk menutup koneksi, control flag FIN (Finish) harus di set.
- Diperlukan two way handshake untuk menutup satu sesi searah, terdiri dari FIN dan ACK.
- Sehingga dibutuhkan 4 kali pergantian segment untuk menutup sebuah komunikasi TCP
- Catatan : digunakan bahasa client dan server untuk kemudahan pemahaman, sebenarnya penutupan sesi bisa diawali baik dari server maupun client

TCP- penutupan sesi

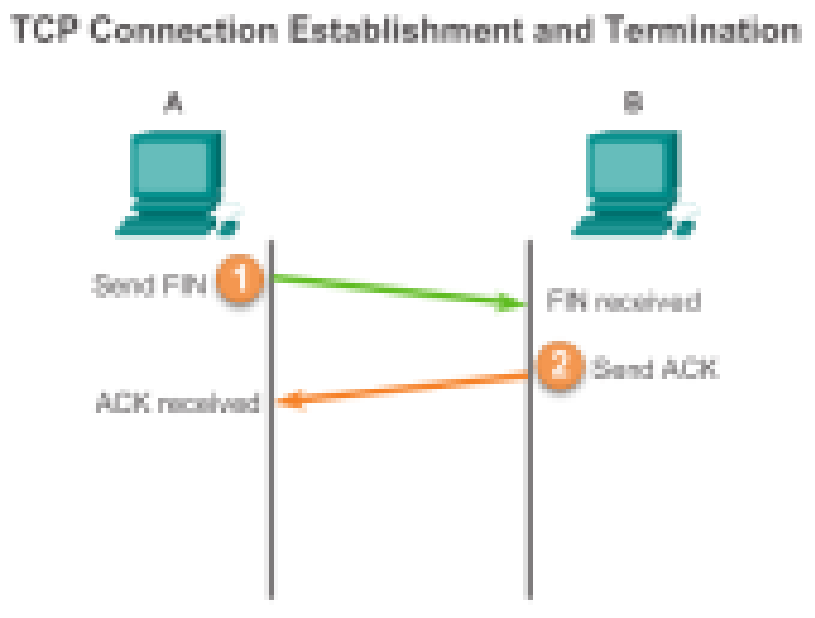


1. Saat client tidak memiliki data untuk dikirim lagi, ia mengirim segment dengan flag FIN bernilai 1

Nomor seq dan ack mengikuti nomor berikutnya dari pengiriman data terakhir, dalam contoh ini tidak dibahas agar lebih sederhana

Source Port	Dest Port	Sequence No	ACK No	Header Length	Flags						Window Size	TCP Checksum	Urgent Pointer	DATA
					URG	ACK	PSH	RST	SYN	FIN				
					0	0	0	0	0	1				

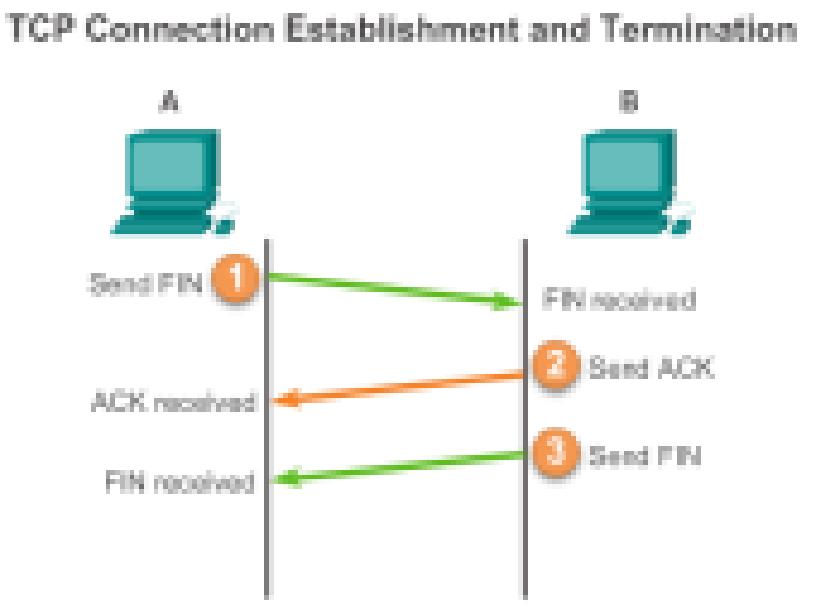
TCP- penutupan sesi



2. Server akan mengirim ACK sebagai tanda terima untuk penutupan sesi

Source Port	Dest Port	Sequence No	ACK No	Header Length	Flags						Window Size	TCP Checksum	Urgent Pointer	DATA
					URG	ACK	PSH	RST	SYN	FIN				
					0	1	0	0	0	0				

TCP- penutupan sesi

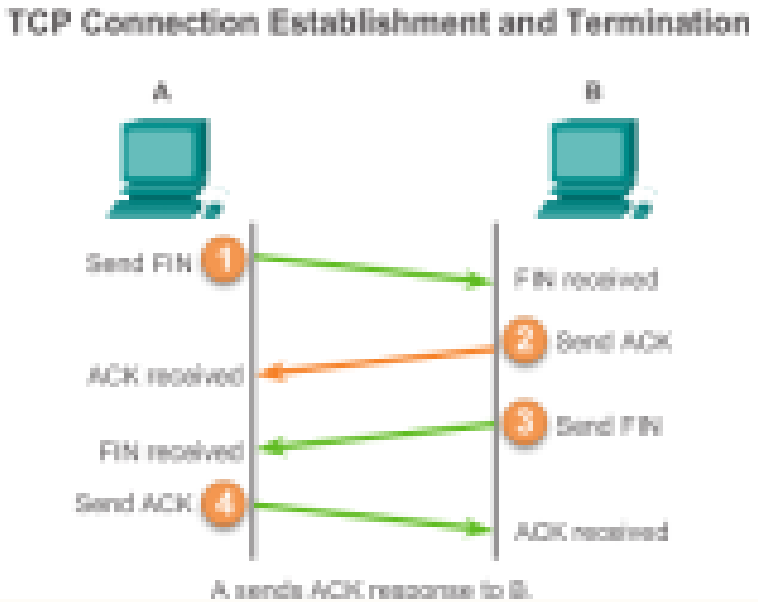


3. Server mengirim FIN pada client

Source Port	Dest Port	Sequence No	ACK No	Header Length	Flags						Window Size	TCP Checksum	Urgent Pointer	DATA
					URG	ACK	PSH	RST	SYN	FIN				
					0	0	0	0	0	1				

TCP- penutupan sesi

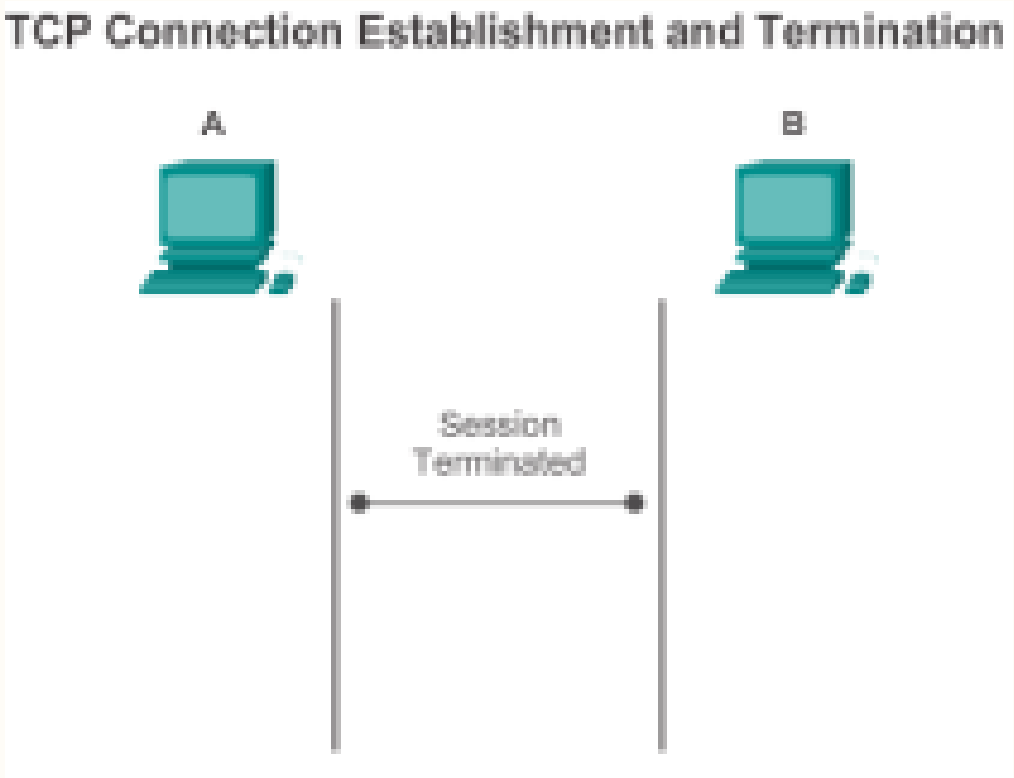
4. Client merespon dengan ACK



Source Port	Dest Port	Sequence No	ACK No	Header Length	Flags						Window Size	TCP Checksum	Urgent Pointer	DATA
					URG	ACK	PSH	RST	SYN	FIN				
					0	1	0	0	0	0				

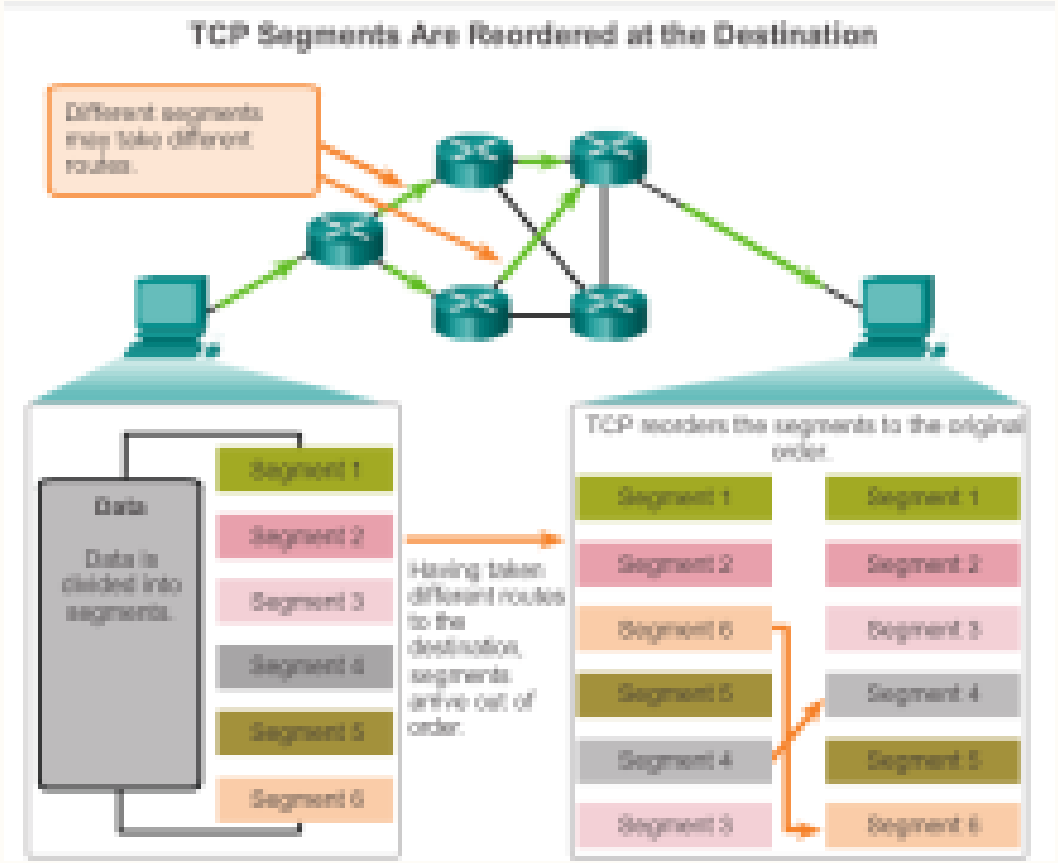
TCP- penutupan sesi

– Dengan 4 langkah tersebut, sesi dianggap selesai, jika ada pengiriman data lagi, sesi harus dibuka ulang



TCP
Sending the data

- Setelah sesi koneksi dibuka, TCP dapat mengirimkan data ke tujuan
- TCP memiliki kemampuan untuk mengirimkan ulang data dan re-ordering data yang yang tiba dengan urutan salah.



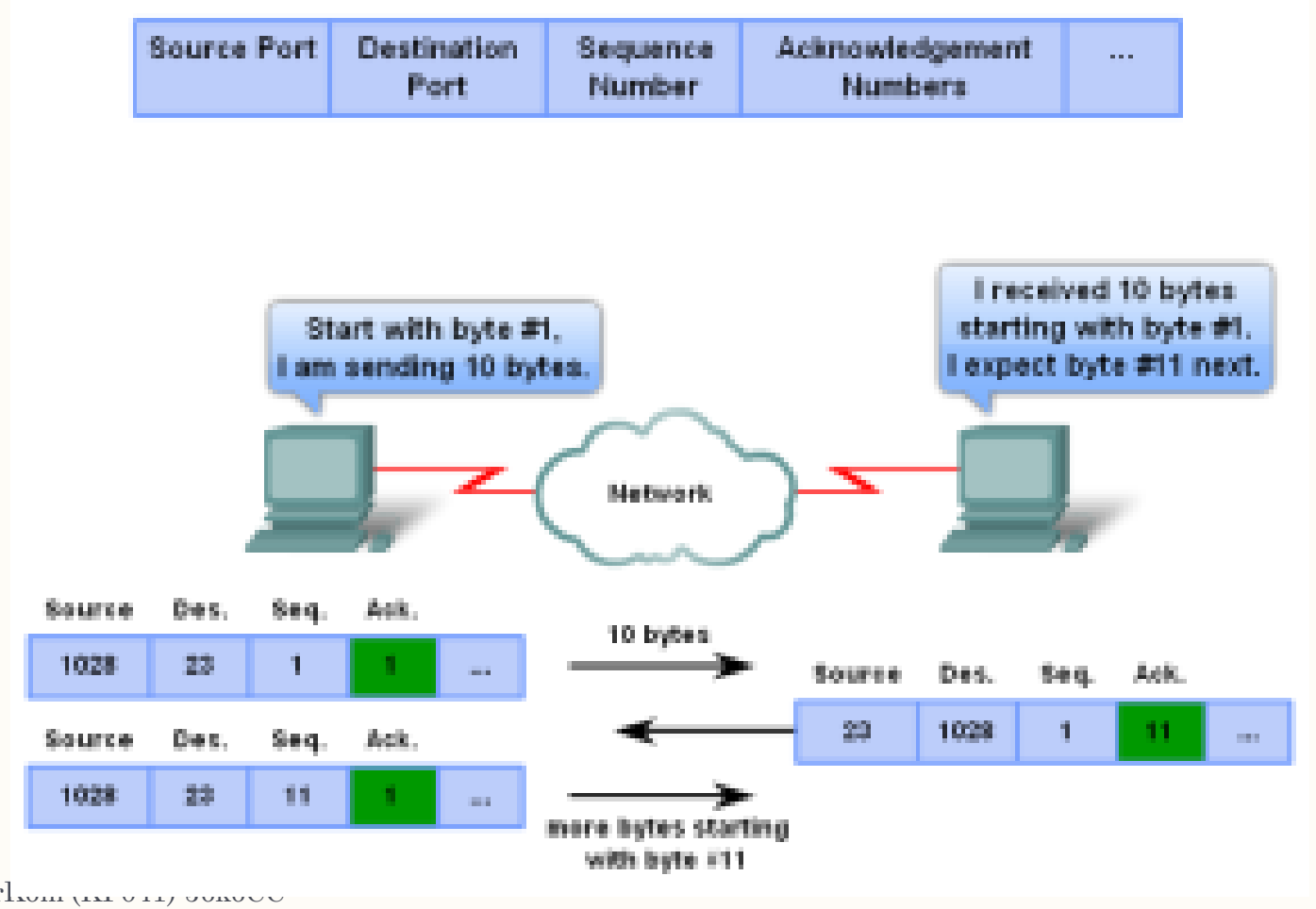
TCP acknowledgement dengan windowing

- Untuk menjamin data sampai ditujuan, digunakan tanda terima (acknowledgement). TCP akan menunggu acknowledgment sebelum mengirimkan data lanjutan.
- Resend dan re-ordering dimungkinkan karena adanya sequence number dan acknowledgement number (selanjutnya disebut seq dan ack)
- Seq no dan ack no digunakan untuk mengkonfirmasi penerimaan segmen data.
- Seq mengindikasikan jumlah byte relatif yang telah ditransmisikan, termasuk byte pada segmen saat ini.
- Ack pada balasan, mengindikasikan byte berikutnya yang diharapkan untuk diterima . Metode ini disebut **expectational acknowledgement**

TCP retransmission

- Sebaik apapun desain jaringan, pasti terjadi data loss, sehingga TCP harus menyediakan mekanisme retransmisi data yang hilang
- Perhatikan video untuk memahami mekanisme retransmisi TCP

TCP acknowledgement dengan windowing



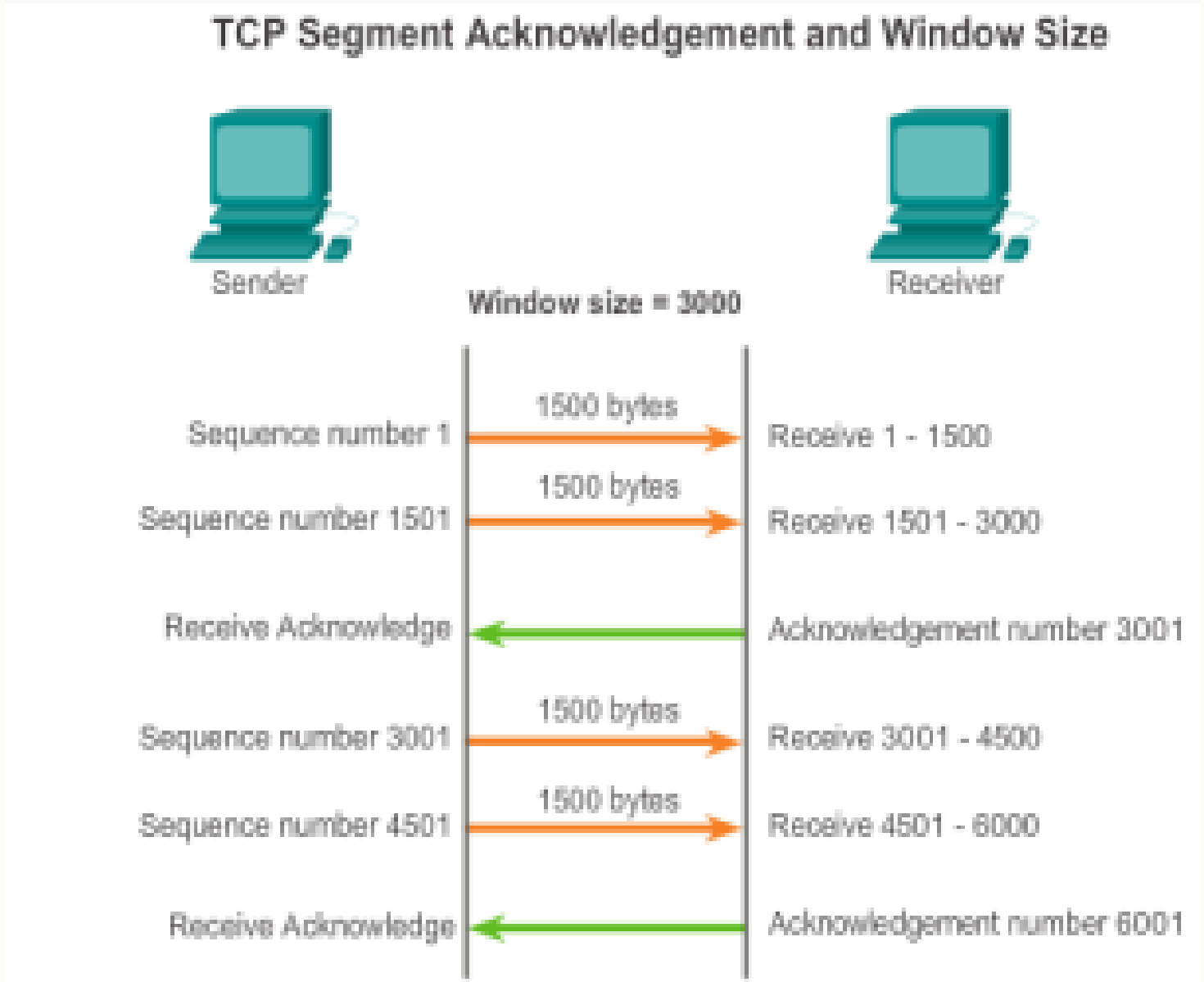
TCP acknowledgement dengan windowing

- Menggunakan model pada contoh, host pengirim harus menunggu ack untuk tiap 10 byte, jaringan akan memiliki banyak overhead (beban).
- Untuk mengurangi beban jaringan, banyak segment dapat dikirim sebelum menerima ack.
- Ack yang digunakan bisa berupa summary (ack untuk banyak segment sekaligus)
- Contoh , jika pengiriman menggunakan seq number = 2000 dan mengirim 10 segment berukuran 1000 byte, maka ack nya akan bernilai 12001
- Jumlah data yang dapat dikirimkan sebelum menerima ack disebut dengan **window size**

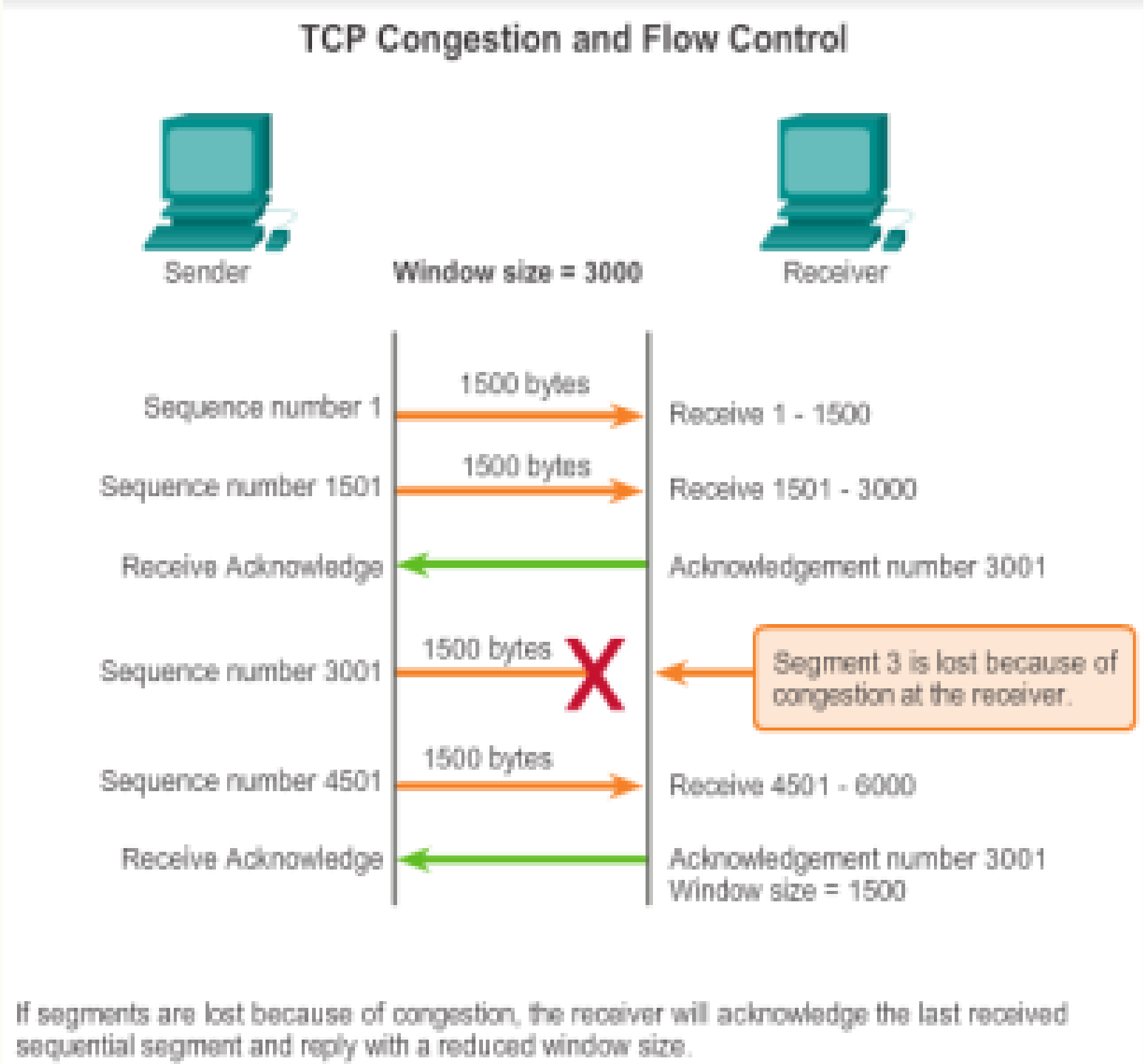
TCP congestion control – minimizing segment loss

- TCP juga menyediakan mekanisme untuk kontrol flow, yang membagntu reliabilitas dengan mengatur kecepatan data flow yang efektif.

- Field window size pada header mengatur berapa banyak data yang bisa dikirim sebelum ack diterima, nilai awal ditentukan saat three way handshake



TCP congestion control – minimizing segment loss



TCP congestion control – minimizing segment loss

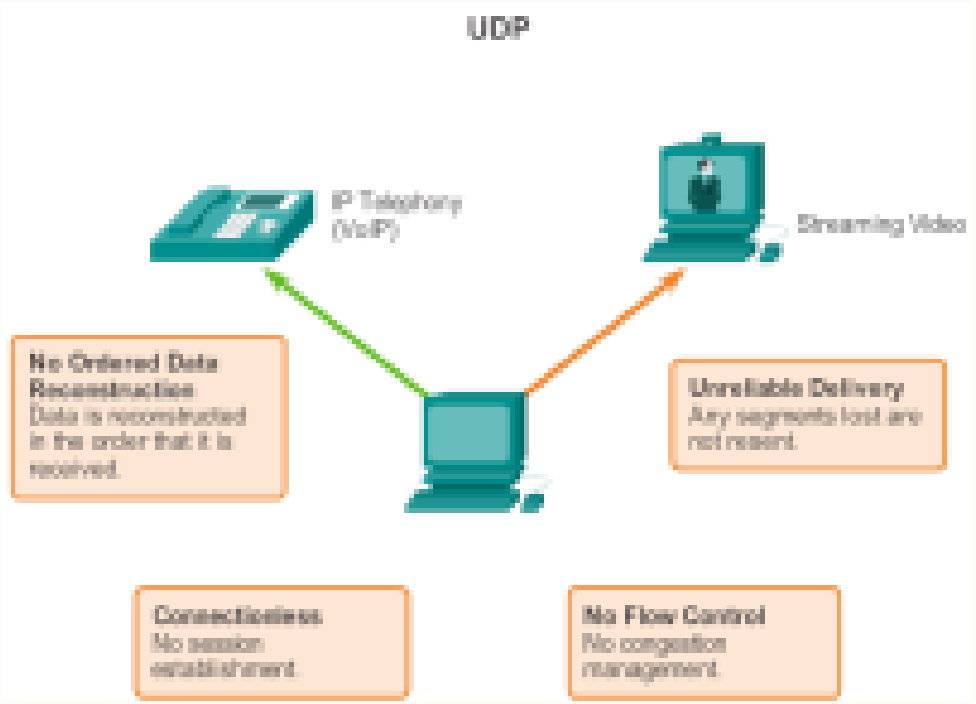
Window size dapat diubah, jika terjadi kehilangan data, atau beban terlalu tinggi.

LATIHAN

– Dosen akan memberikan contoh kasus baru terkait proses pengiriman data dengan TCP –flow control dan resend

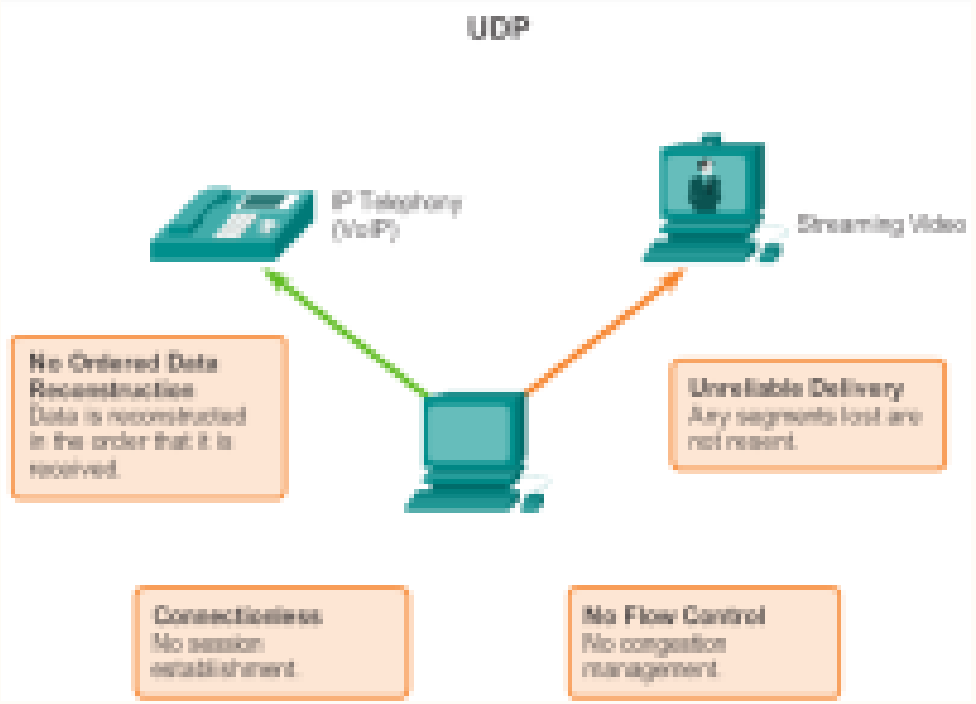
Gambaran Umum - UDP

- User Datagram Protocol (UDP)
  Adalah protokol sederhana yang tidak berorientasi koneksi (connectionless protocol), dinyatakan pada RFC 768.
- Tiap potongan data dari UDP disebut datagram.
- Kelebihan UDP adalah overhead yang rendah, besar enkapsulasi tiap segment hanya 8 byte dan tidak memerlukan tanda terima.

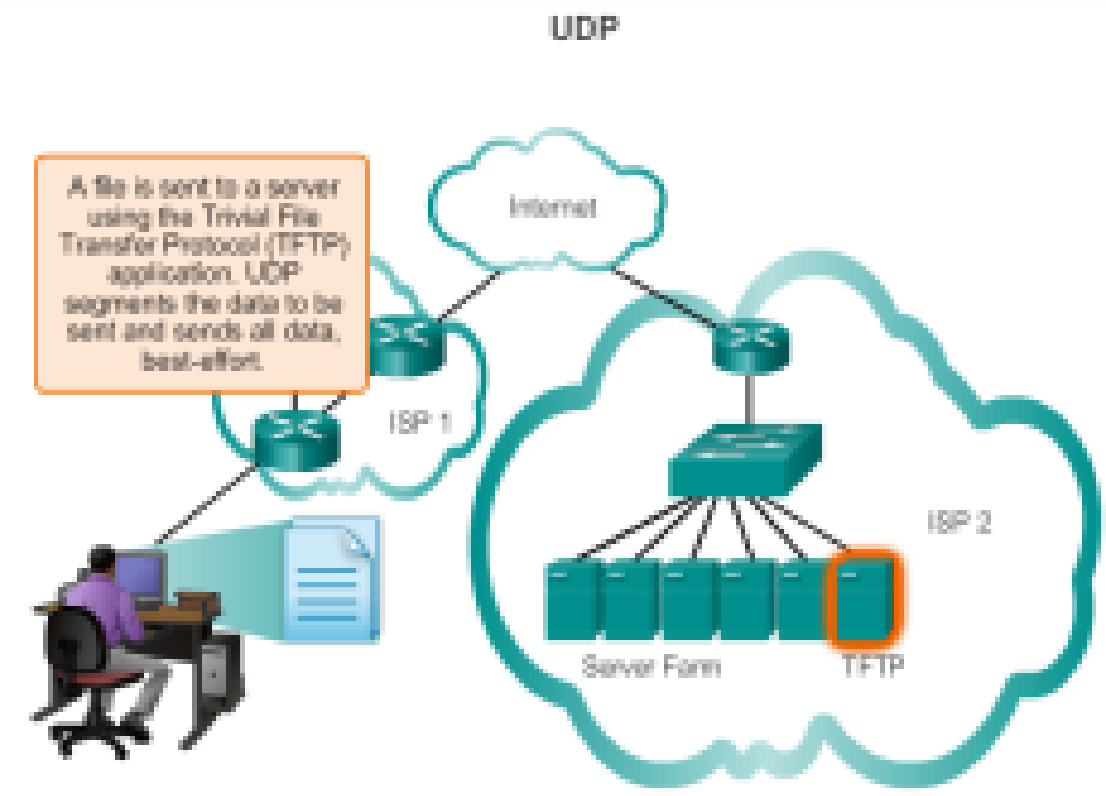


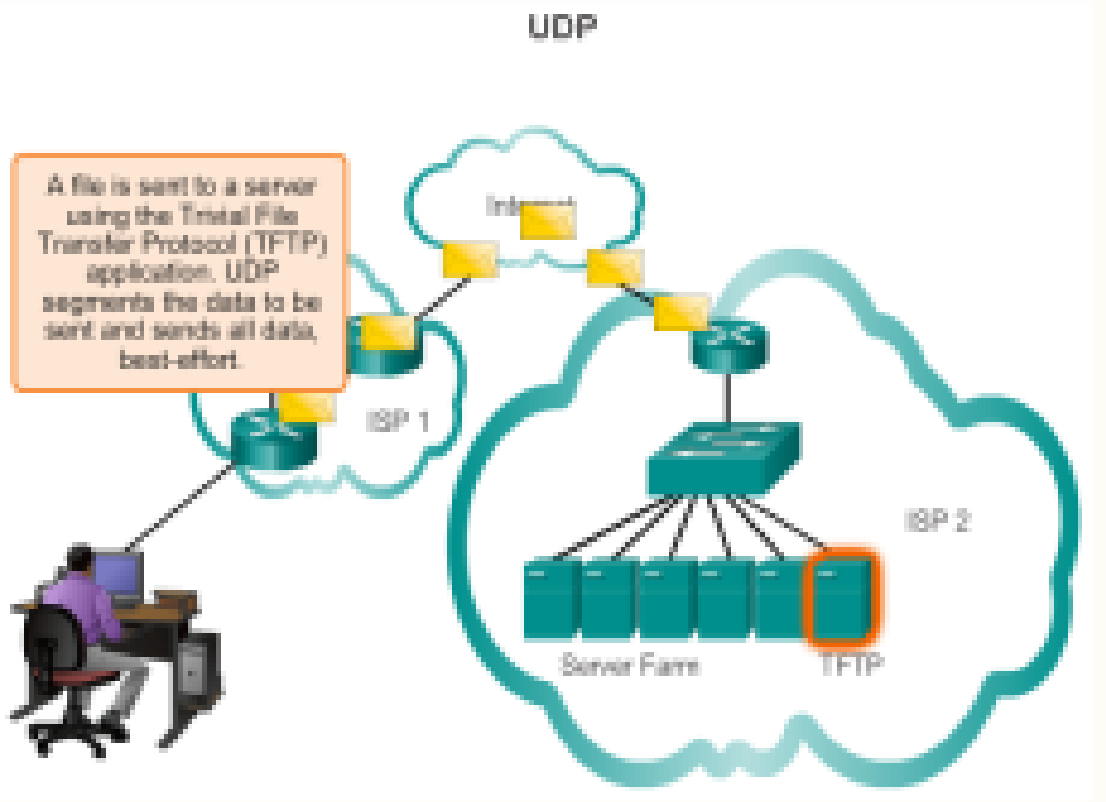
Gambaran Umum - UDP

- Datagram dikirim dengan metode usaha terbaik (best effort) sering juga disebut unreliable delivery
- Tidak ada rekonstruksi data, datagram yang diterima akan diurutkan sesuai waktu penerimaan
- Tidak mendukung Flow control
- Aplikasi yang menggunakan : Domain Name System (DNS), Video Streaming, Voice over IP (VoIP)

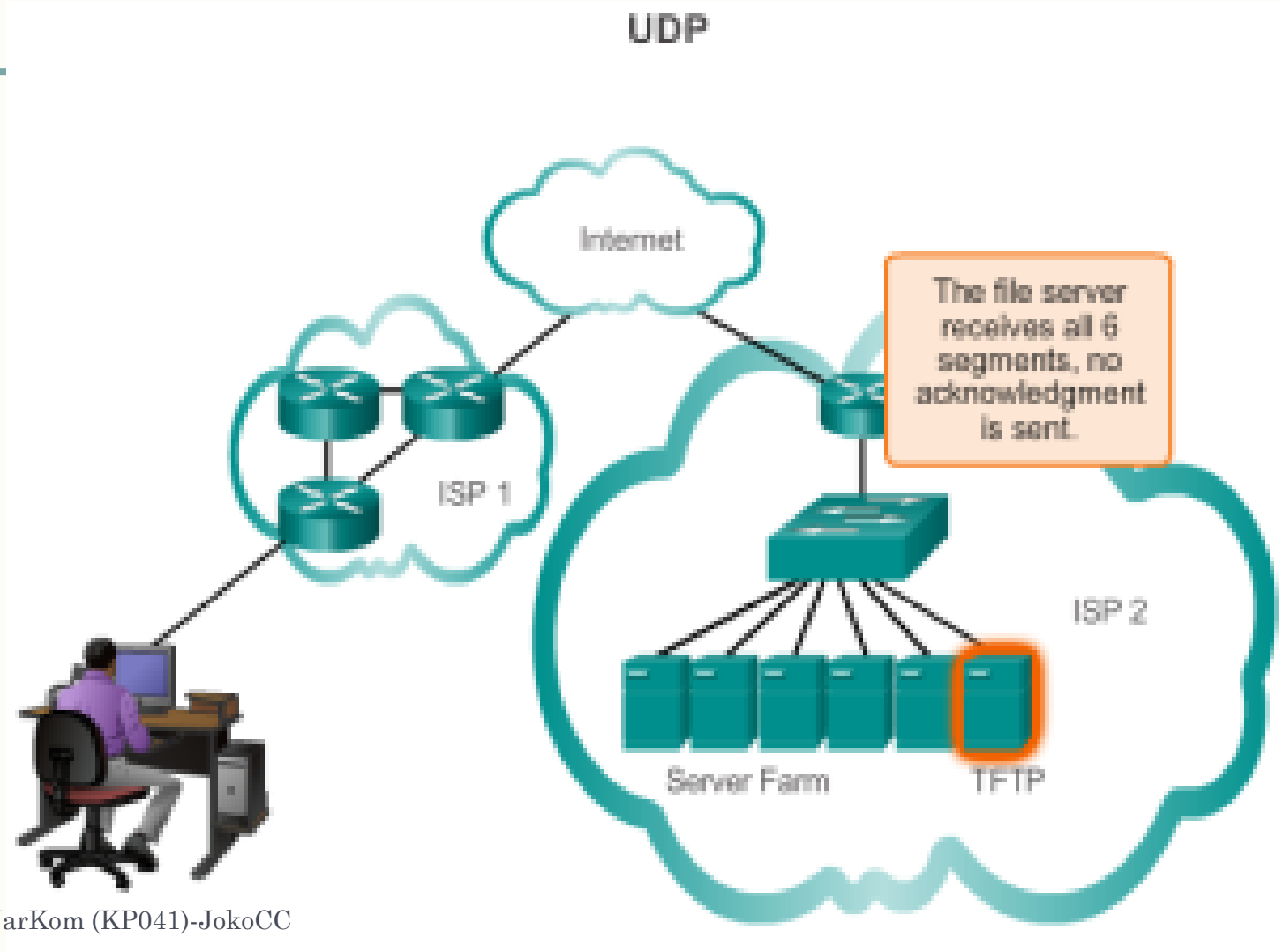


Contoh komunikasi UDP





Contoh komunikasi UDP



Header UDP

Field	Panjang	Keterangan
Source Port	16 bit (2 byte)	Digunakan untuk mengidentifikasikan sumber protokol lapisan aplikasi yang mengirimkan pesan UDP yang bersangkutan. Penggunaan field ini adalah opsional, dan jika tidak digunakan, akan diset ke angka 0. Beberapa protokol lapisan aplikasi dapat menggunakan nilai field ini dari pesan UDP yang masuk sebagai nilai field port tujuan (Destination Port, lihat baris selanjutnya) sebagai balasan untuk pesan tersebut.
Destination Port	16 bit (2 byte)	Digunakan untuk mengidentifikasikan tujuan protokol lapisan aplikasi yang menjadi tujuan pesan UDP yang bersangkutan. Dengan menggunakan kombinasi antara alamat IP dengan nilai dari field ini untuk membuat sebuah alamat yang signifikan untuk mengidentifikasikan proses yang berjalan dalam sebuah host tertentu yang dituju oleh pesan UDP yang bersangkutan.

Header UDP

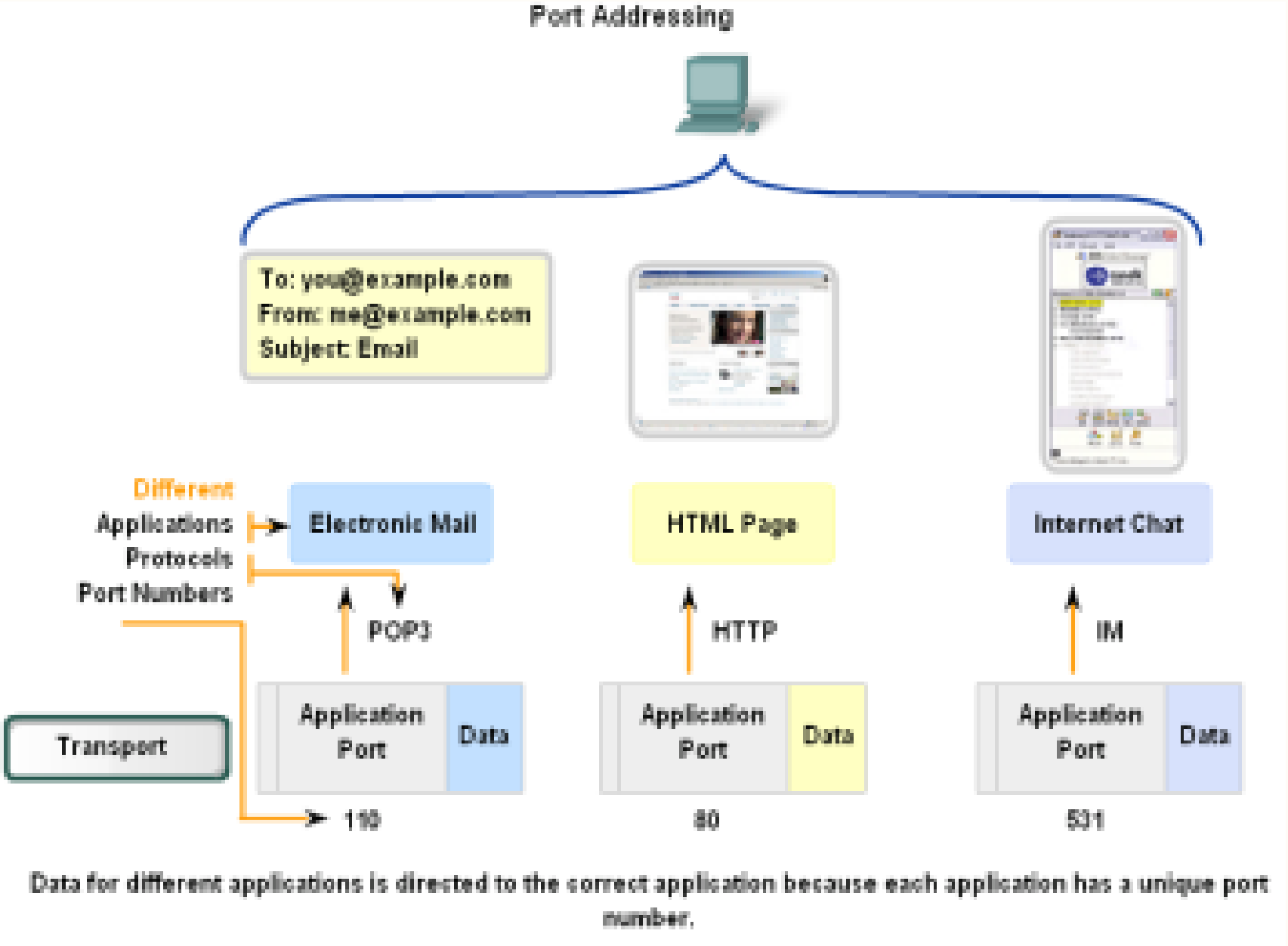
Length	16 bit (2 byte)	Digunakan untuk mengindikasikan panjang pesan UDP (pesan UDP ditambah dengan header UDP) dalam satuan byte. Ukuran paling kecil adalah 8 byte (ukuran header UDP, ketika tidak ada isi pesan UDP), dan ukuran paling besar adalah 65515 bytes (65535 [2^{16}] -20 [ukuran header protokol IP]). Panjang maksimum aktual dari pesan UDP akan disesuaikan dengan menggunakan nilai Maximum Transmission Unit (MTU) dari saluran di mana pesan UDP dikirimkan. Field ini bersifat redundan (terulang-ulang). Panjang pesan UDP dapat dihitung dari field Length dalam header UDP dan field IP Header Length dalam header IP.
Checksum	16 bit (2 byte)	Berisi informasi pengecekan integritas dari pesan UDP yang dikirimkan (header UDP dan pesan UDP). Penggunaan field ini adalah opsional. Jika tidak digunakan, field ini akan bernilai 0.

Memisahkan data aplikasi

- Bagaimana protokol pada transport layer membedakan data aplikasi yang satu dengan yang lain?
- Hal ini dapat dilakukan dengan menggunakan nomor port
- Protokol TCP dan UDP mendukung nomor port

Port addressing

- Untuk mengetahui aplikasi mana yang menggunakan, tiap segmen atau datagram dilengkapi dengan nomor port.
- Server memiliki static port number yang telah ditentukan, misalnya , port untuk web server ialah 80, untuk FTP adalah 20 dan 21
- Client akan menggunakan alamat port yang dinamis acak

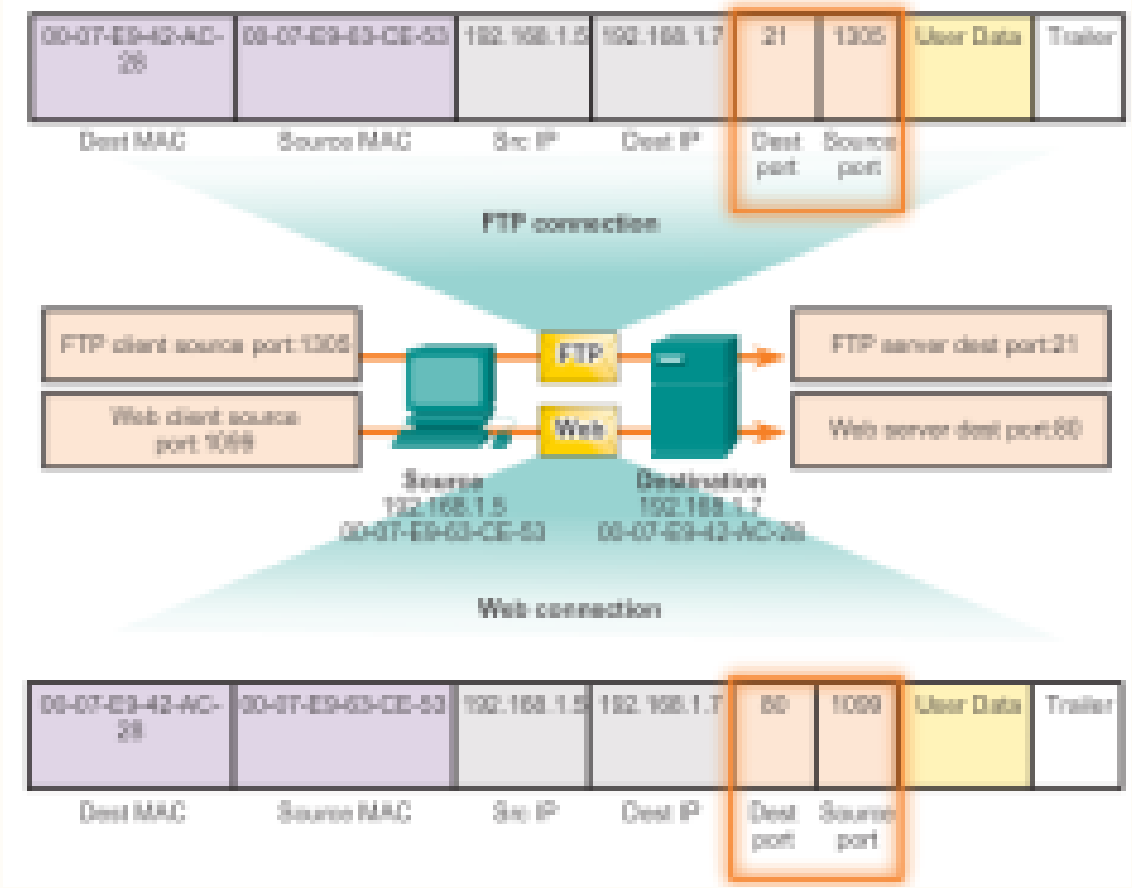


Port addressing

Port addressing

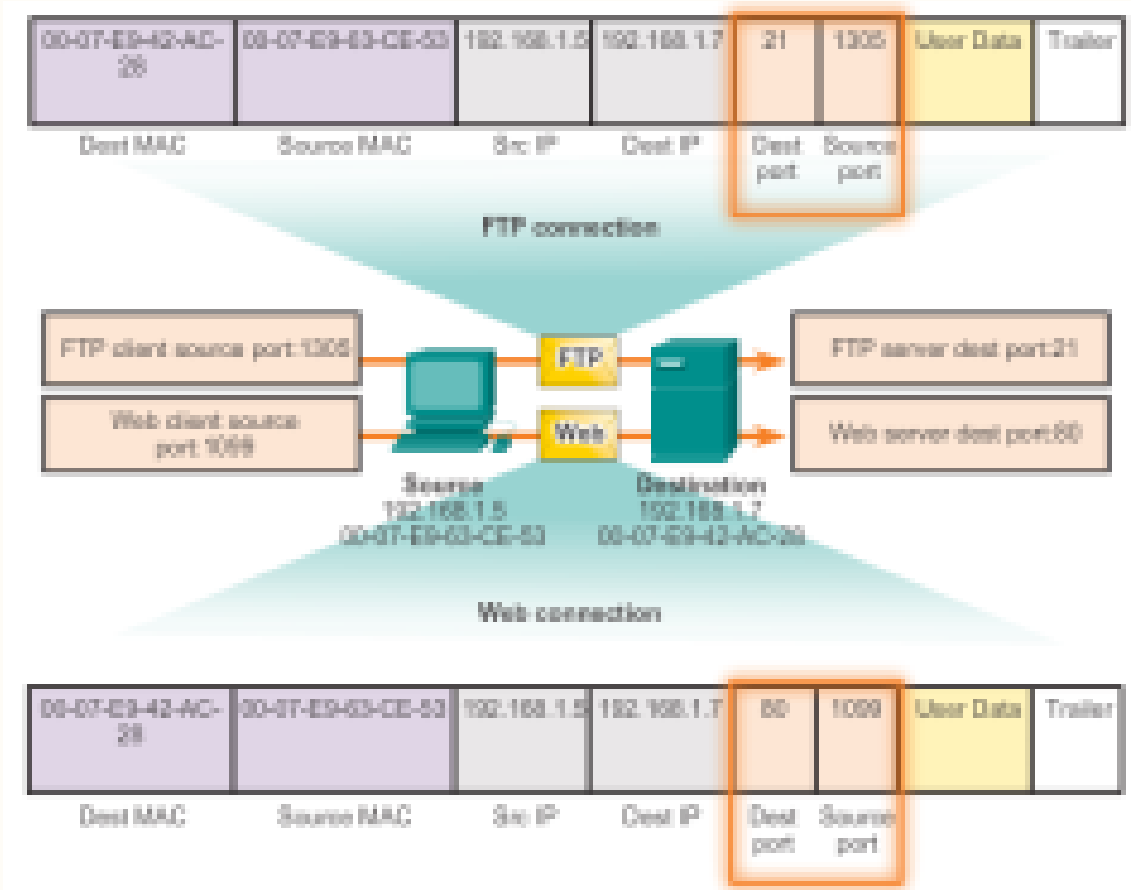
- Contoh , jika sebuah client dengan ip 192.168.1.5 membuka FTP client untuk mengakses layanan FTP pada server dengan ip 192.168.1.7

- Pada saat yang bersamaan, client tersebut juga mengaktifkan browser dan mengakses halaman pada server yang sama.



Port addressing

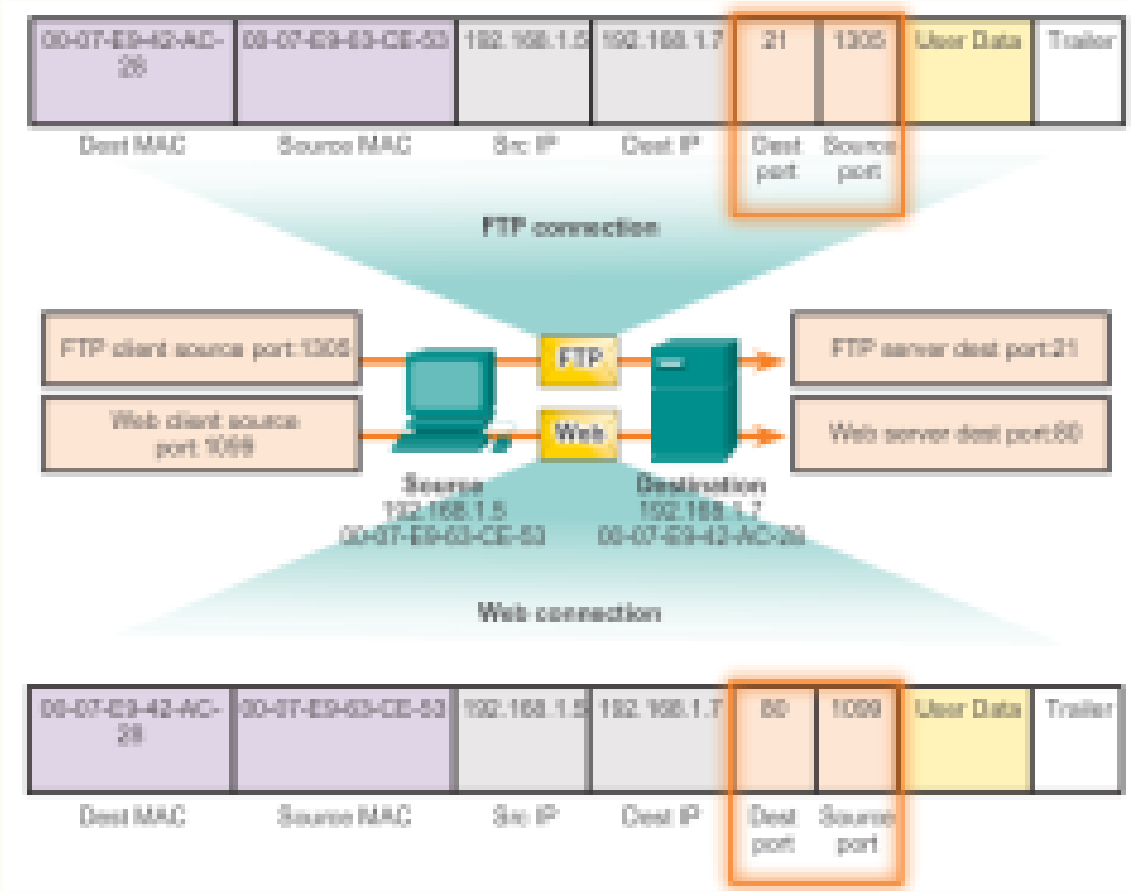
- FTP server berjalan di port 21 (standar)
- FTP client akan memilih nomor port secara dinamis (diluar port standar), misalnya 1305.
- Web server berjalan di port 80 (standar)
- Browser akan memilih nomor port secara dinamis yang tidak dipakai pada host tersebut. Misalnya 1099



Port addressing

- Segment FTP akan dikirim dari port 1305 kepada port 21
- Dapat dituliskan : 192.168.1.5:1305 ➔192.168.1.7:21

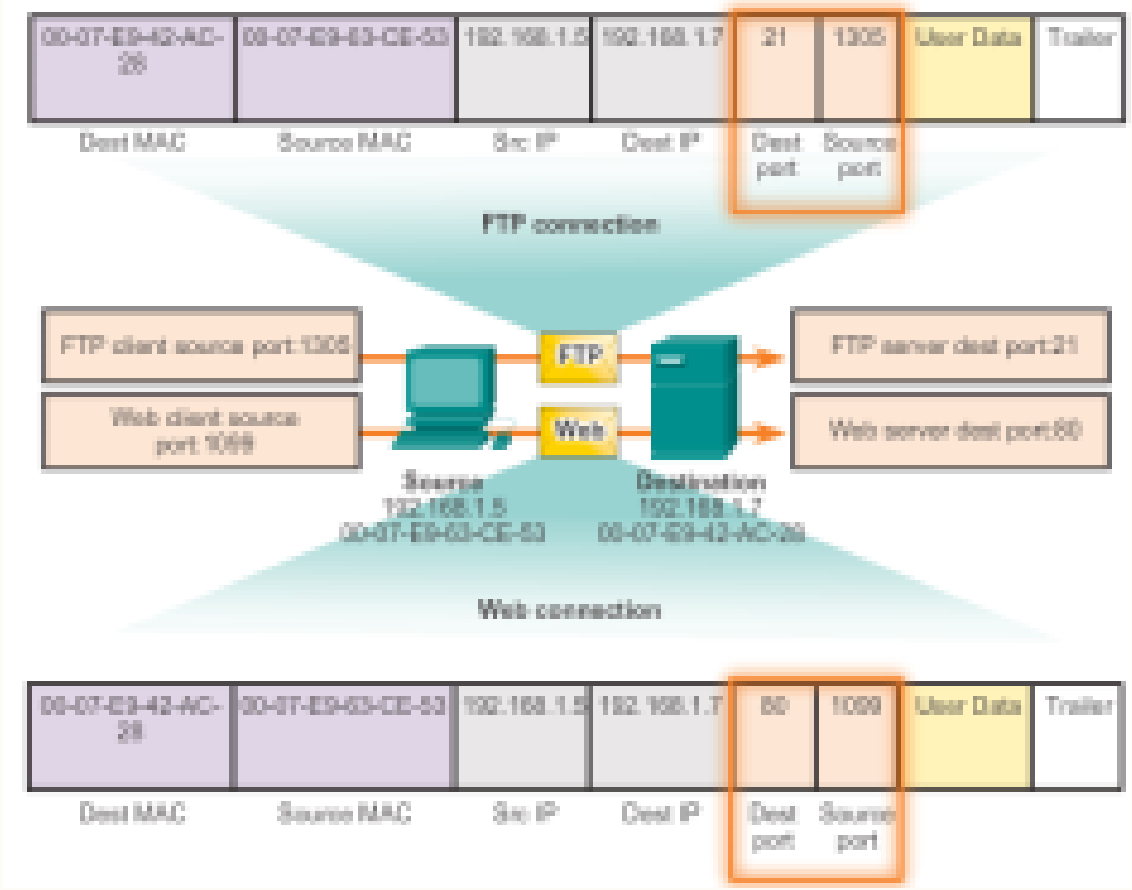
- Segment HTTP akan dikirim dari port 1099 kepada port 80
- Dapat dituliskan : 192.168.1.5:1099 ➔ 192.168.1.7:80



Port addressing

- Balasan dari server FTP akan berupa segment dengan tujuan 192.168.1.7:21 ➔ 192.168.1.5:1305

- Balasan dari server HTTP akan berupa segment dengan tujuan 192.168.1.7:80 ➔ 192.168.1.5:1099



Kategori Port addressing

– Berikut ialah pembagian jatah nomor port oleh IANA (Internet Assigned Numbers Authority).

1. Well Known Ports (Numbers 0 to 1023) –
   Reservasi oleh aplikasi dan layanan umum seperti web server, email, telnet, dsb.

2. Registered Ports (Numbers 1024 to 49151) –
   Untuk proses user atau aplikasi, umumnya berupa aplikasi individual yang diinstal pada host, juga dapat digunakan secara dinamis.

3. Dynamic or Private Ports (Numbers 49152 to 65535) –
   juga disebut sebagai Ephemeral Ports. Umumnya digunakan secara dinamis oleh aplikasi client.

– Daftar port number yang distandarisasi saat ini dapat dilihat pada **http://www.iana.org/assignments/port-numbers .**

Port addressing

Port Numbers

Port Number Range	Port Group
0 to 1023	Well Known (Contact) Ports
1024 to 49151	Registered Ports
49152 to 65535	Private and/or Dynamic Ports

Port addressing- contoh TCP

Port Numbers

Port Number Range	Port Group
0 to 1023	Well Known (Contact) Ports
1024 to 49151	Registered Ports
49152 to 65535	Private and/or Dynamic Ports

Registered TCP Ports:

1863	MSN Messenger
8008	Alternate HTTP
8080	Alternate HTTP

Well Known TCP Ports

21	FTP
23	Telnet
25	SMTP
80	HTTP
110	POP3
194	Internet Relay Chat (IRC)
443	Secure HTTP (HTTPS)

Port addressing- contoh UDP

Port Numbers

Port Number Range	Port Group
0 to 1023	Well Known (Contact) Ports
1024 to 49151	Registered Ports
49152 to 65535	Private and/or Dynamic Ports

Registered UDP Ports:
1812 RADIUS Authentication Protocol
2000 Cisco SCCP (VoIP)
5004 RTP (Voice and Video Transport Protocol)
5060 SIP (VoIP)

Well Known UDP Ports:
69 TFTP
520 RIP

Port addressing –contoh TCP dan UDP

Port Numbers

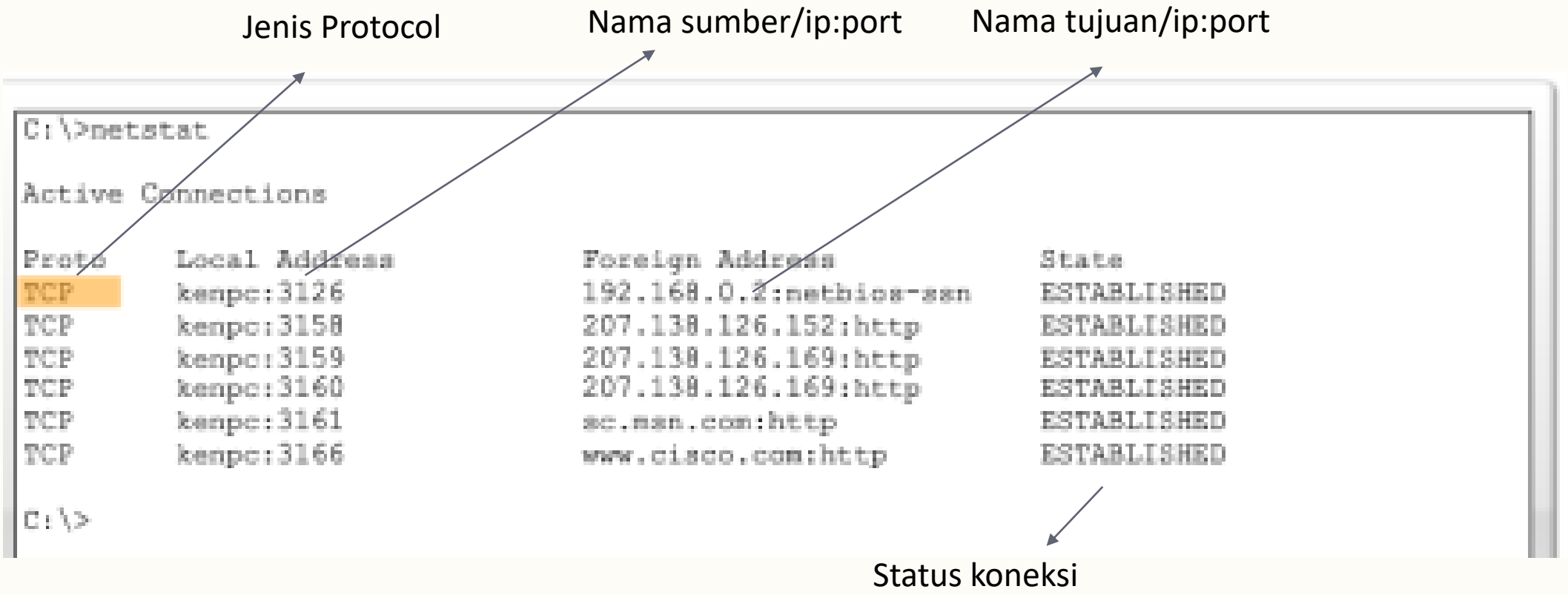
Port Number Range	Port Group
0 to 1023	Well Known (Contact) Ports
1024 to 49151	Registered Ports
49152 to 65535	Private and/or Dynamic Ports

Registered TCP/UDP Common Ports:
1433 MS SQL
2948 WAP (MMS)

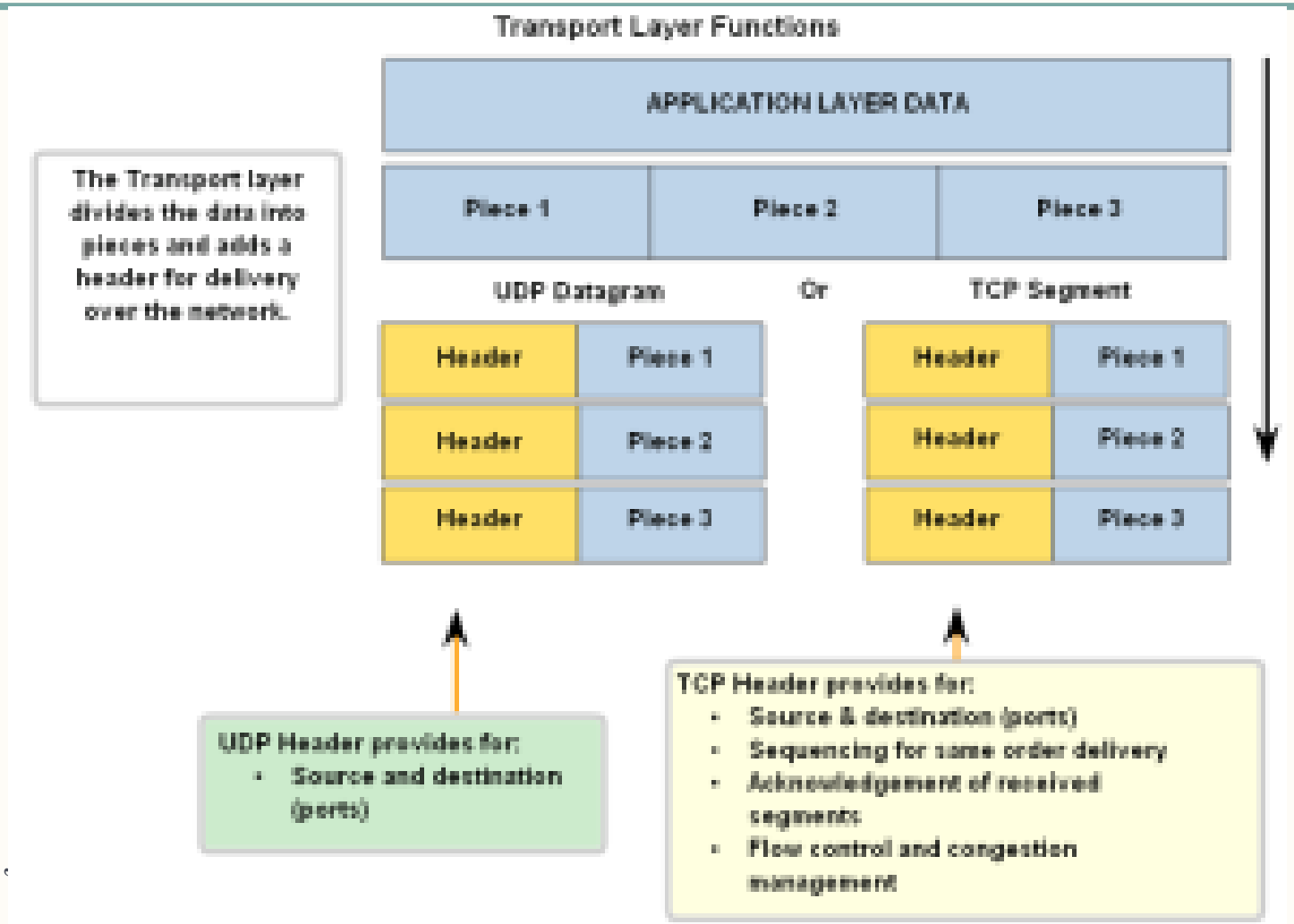
Well Known TCP/UDP Common Ports:
53 DNS
161 SNMP
531 AOL Instant Messenger, IRC

Port Addressing - tool

- Untuk memonitor koneksi dan port yang digunakan, dapat digunakan aplikasi netstat
- Dapat diketahui protokol, alamat local, tujuan, port, dan status koneksi

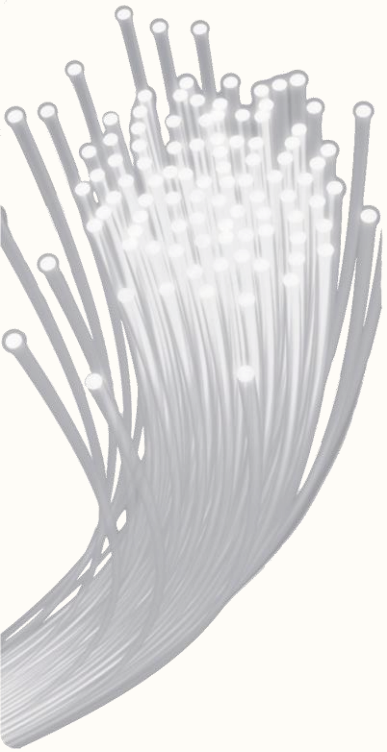


Segmentasi dan reassembly pada TCP VS UDP



Segmentasi dan reassembly pada TCP VS UDP

- Pada dasarnya UDP tidak memberikan nomor pengurutan, sehingga sebuah datagram dapat sampai pada urutan yang berbeda-beda.
- Kondisi ini harus diantisipasi oleh aplikasi yang menggunakan.
- Umumnya protokol aplikasi yang menggunakan UDP mengirimkan pesan ukuran kecil yang muat dalam satu datagram (contoh DNS, SNMP)

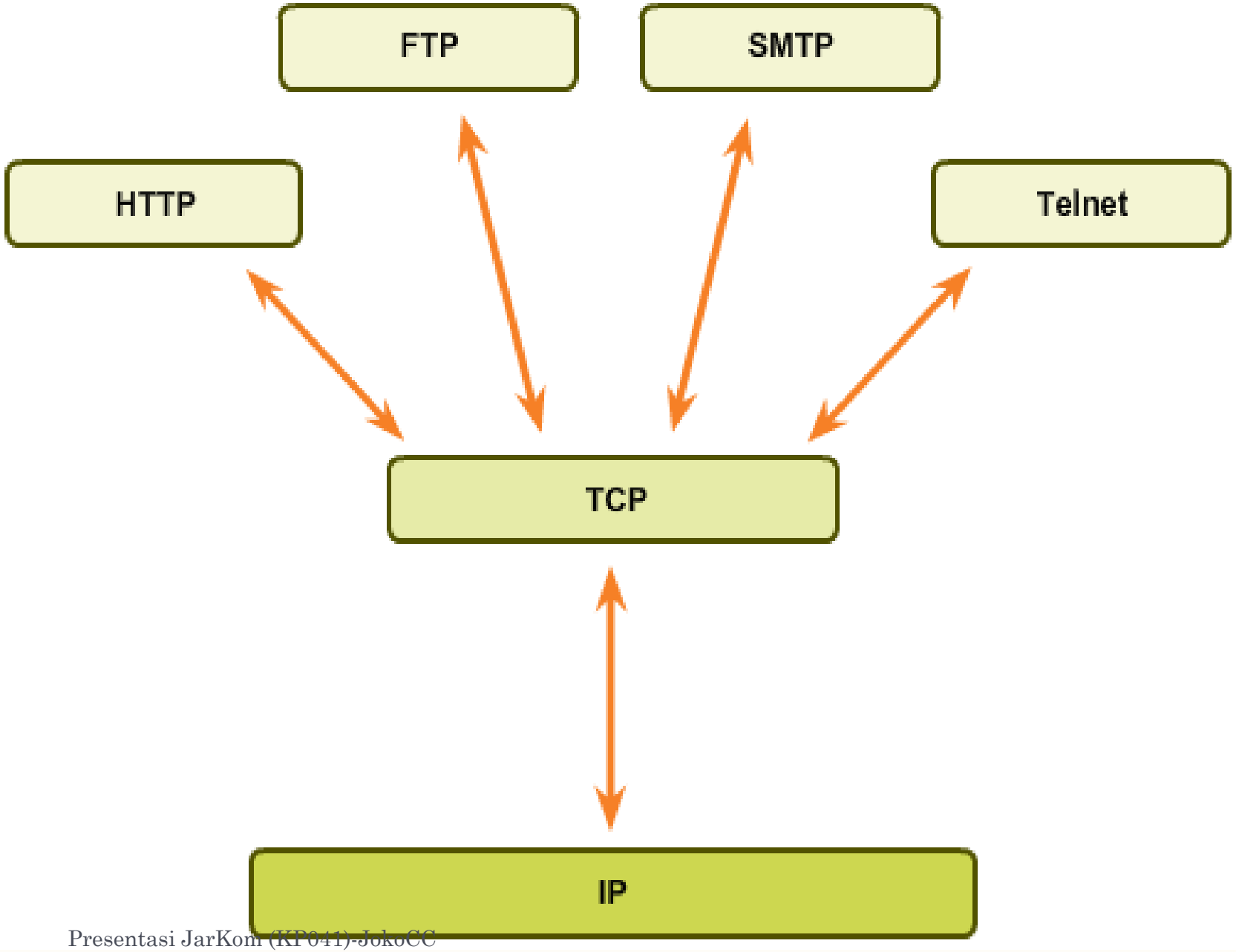


rangkuman TCP VS UDP

keterangan	TCP	UDP
Akronim dari	Transmission Control Protocol	User Datagram Protocol
Nama Protocol Data Unit (PDU)	Segment	Datagram
Tipe Koneksi	Connection Oriented (membuka sesi koneksi)	Connectionless (tidak punya sesi koneksi)
Penggunaan cocok untuk aplikasi yang :	membutuhkan reliabilitas tinggi, dan waktu kirim tidak begitu kritis	Membutuhkan pengiriman cepat, efisien seperti game, live broadcasting & communication dan query kecil dari banyak client (SNMP)
Reliabilitas	Ada jaminan informasi bahwa data sampai/ tidak sampai ke tujuan	Tidak ada jaminan data sampai ke tujuan, tidak ada informasi.

rangkuman TCP VS UDP

keterangan	TCP	UDP
Ukuran header	20 Bytes	8 Bytes
Data Flow Control (mengatur kecepatan kirim secara dinamis) atau congestion control	Tersedia	Tidak ada
Cek Error	Ada, dengan pengiriman ulang jika segment rusak/ hilang	Ada, tidak memiliki mekanisme pengiriman ulang
Ada tanda terima	Ada	Tidak
Pengurutan paket	Dapat dilakukan	Tidak dapat dilakukan



Aplikasi yang menggunakan TCP

Aplikasi yang menggunakan TCP memerlukan jaminan reliabilitas, beberapa contoh diantaranya adalah :

- http,https
- ftp, nfs
- Smtp, pop, imap
- telnet, ssh
- Netbios
- Bgp
- irc

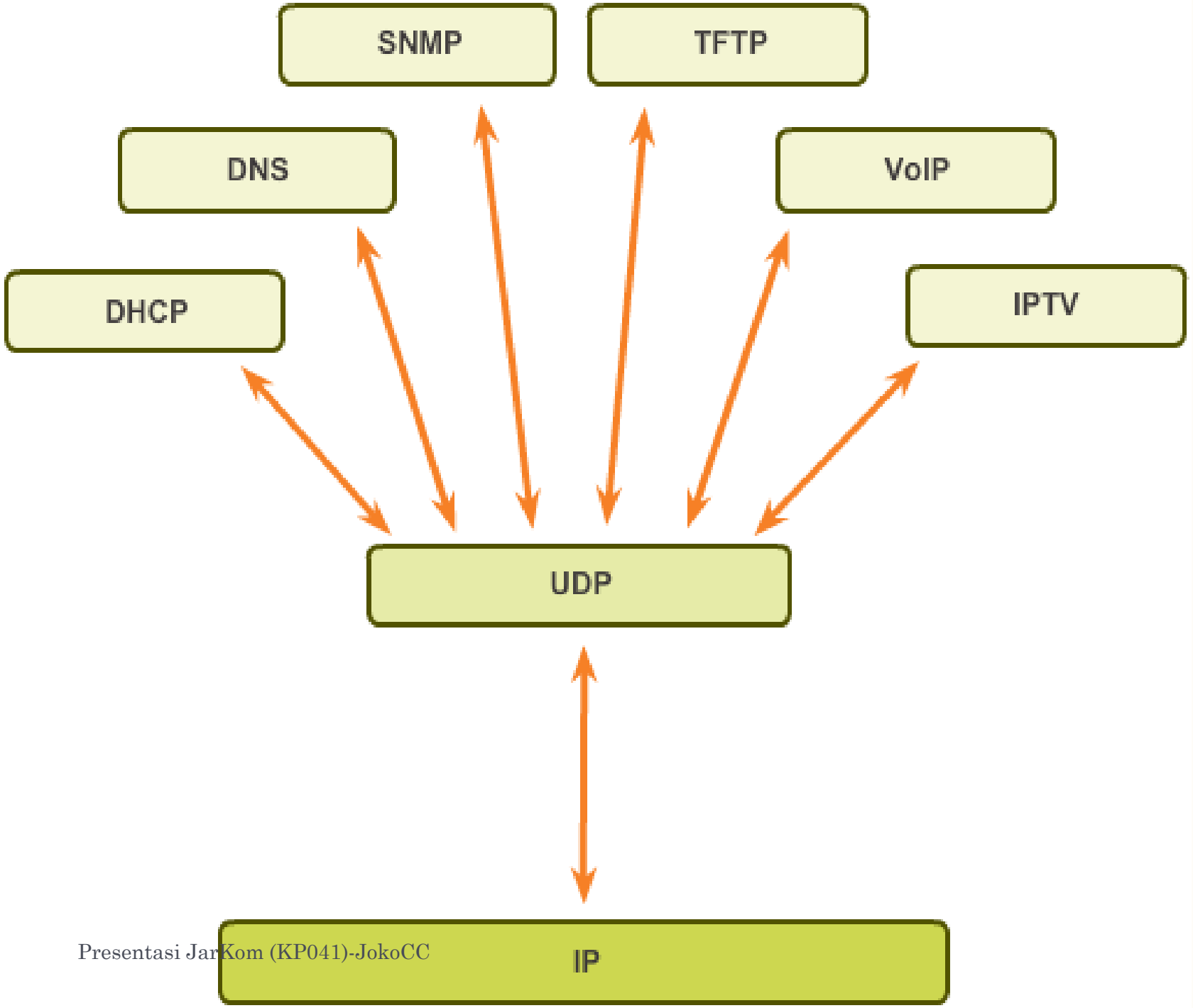
UDP

- Karena UDP didesain untuk beban rendah, maka UDP tidak memiliki mekanisme resend, re-ordering, maupun flow control.

- UDP tidak membuka sesi koneksi dan termasuk kategori connectionless

- Aplikasi yang menggunakan UDP umumnya adalah aplikasi dengan karakteristik:

1. Dapat mentolerasi kehilangan sebagian data, dan membutuhkan delay yang sangat kecil
2. Aplikasi dengan permintaan sederhana dan dengan reply sederhana
3. Unidirectional communication (searah) yang tidak memerlukan reliabilitas, atau sisi reliabilitas ditanggung aplikasi.



UDP

Contoh aplikasi yang menggunakan UDP adalah :

- DHCP
- DNS
- SNMP
- TFTP
- VoIP
- IPTV

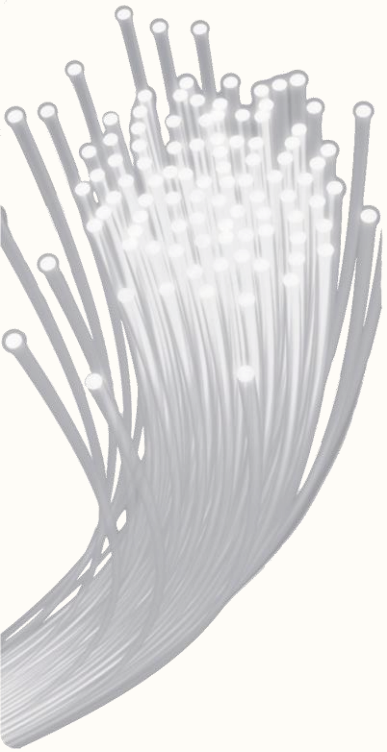
Penggunaan tandem TCP dan UDP

- Beberapa protokol mendukung penggunaan kedua transport protocol seperti DNS.
- DNS menggunakan TCP port 53 untuk zone transfer (sinkronisasi antar server). sedangkan query pertama dari komputer ke server DNS menggunakan UDP port 53, jika belum ada respon dari server setelah 3-5 detik, permintaan ulang dikirimkan via TCP.
- Utility dan diagnostic protocol seperti echo, discard dan time protocol juga didesain untuk memanfaatkan kedua transport protokol TCP dan UDP.

Apa pentingnya saya mengetahui TCP dan UDP?

- Jika anda adalah network admin ➔ ini adalah salah satu fundamental dasar variasi metode pengiriman data yang didukung, dan pengetahuan ini berperan amat besar dalam kaitan diagnostic, troubleshoot, rekonfigurasi dan pengamanan.

- Jika anda adalah developer ➔ anda dapat memilih metode yang tepat dipakai terkait kinerja dan reliabilitas komunikasi antar program, misalnya aplikasi game online mungkin memerlukan UDP agar latensi lebih rendah.



Kesimpulan

- Transport layer memungkinkan komunikasi antar aplikasi pengirim dan penerima.
- Transport layer memiliki mekanisme untuk mengenali aplikasi sumber data dan aplikasi tujuan
- Port number digunakan untuk mengidentifikasi aplikasi, dan terbagi dalam 3 kategori besar : well known, registered, and dynamic.
- TCP dan UDP adalah dua protokol de-facto yang umum digunakan pada jaringan internet dan LAN saat ini.
- TCP merupakan connection oriented protocol, akan membuka dan menutup sesi koneksi
- TCP mendukung resend dan reordering dan flow control
- UDP merupakan connectionless oriented protocol yang lebih rendah beban jaringan, lebih cepat tapi dengan reliabilitas dibawah TCP.

Akhir pertemuan 11

- Terima kasih

- Materi ini bisa di-download melalui link yang tersedia di :

- https://sites.google.com/site/jokocc