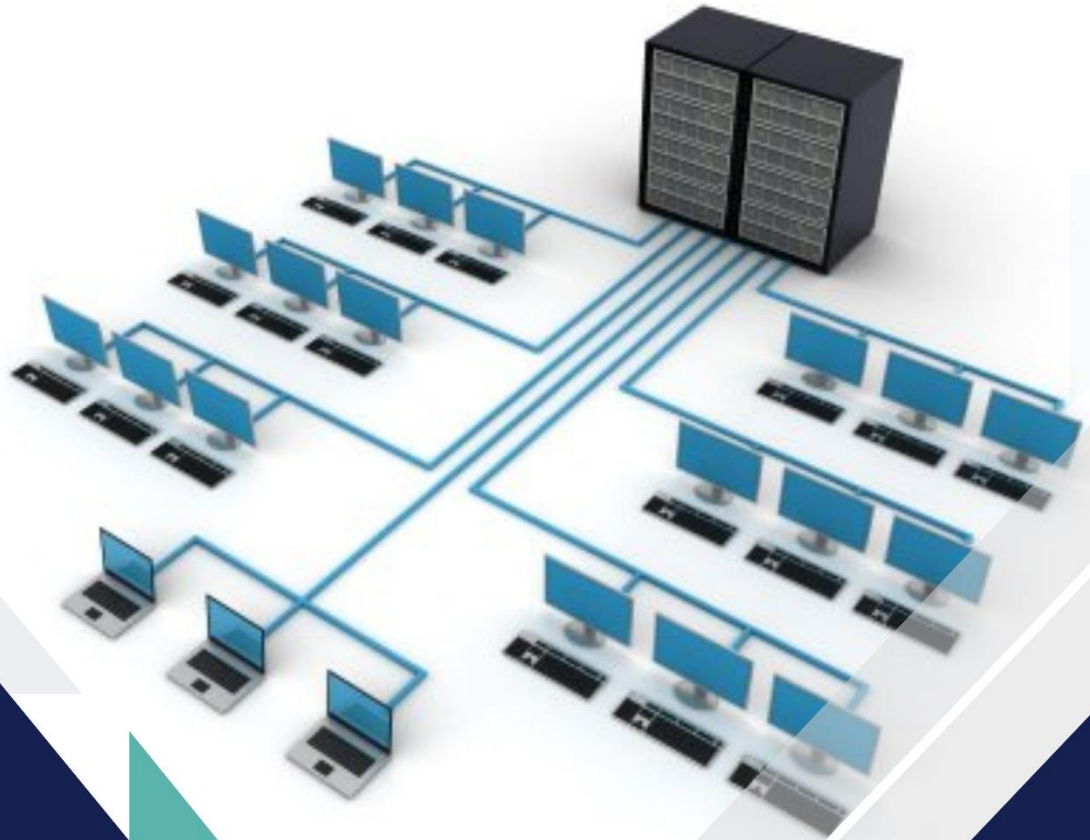
UNIVERSITAS HAYAM WURUK PERBANAS

Pertemuan 11

TRANSPORT LAYER



AUDIO MODUL 11

TUJUAN PEMBELAJARAN

Setelah mempelajari bab ini, mahasiswa seharusnya mampu:

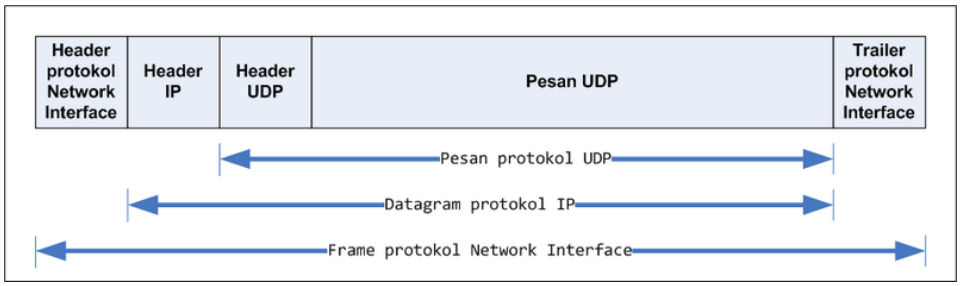
1. Mahasiswa mampu menjelaskan protokol UDP.
2. Mahasiswa mampu menjelaskan protokol TCP.

UDP (User Datagram Protocol)

UDP (User Datagram Protocol) adalah salah satu protokol lapisan transport TCP/IP yang mendukung komunikasi yang tidak handal (unreliable), tanpa koneksi antara host to host dan tidak memiliki kendali aliran. Dengan UDP, sebuah aplikasi komputer dimungkinkan untuk mengirim pesan kepada komputer lain tanpa perlu melalui proses komunikasi awal yang disebut sebagai *"*connectionless*"*. Selain itu, karakteristik UDP lainnya adalah "unreliable", yang artinya semua pesan yang dikirimkan tidak memiliki pesan pemberitahuan terlebih dahulu. Jika selama transmisi ada pesan-pesan yang hilang, maka protokol aplikasi yang letaknya di atas UDP harus memulihkan pesan tersebut.

Pesan UDP

Sebuah pesan UDP berisi header UDP akan dikirimkan ke protokol lapisan berikutnya setelah dibungkus menjadi datagram IP. Pesan UDP memiliki besar maksimum sebesar 65507 byte. Enkapsulasi terhadap pesan-pesan UDP oleh protokol IP dilakukan dengan menambahkan header IP dengan protokol IP nomor 17 (0x11). Datagram IP yang dihasilkan dari proses enkapsulasi tersebut, kemudian akan dienkapsulasi kembali dengan menggunakan header dan trailer protokol.



Gambar 11.1 UDP

Header UDP

Pada header UDP terdapat 4 buah field memiliki ukuran yang tetap diantarnya sebagai berikut:

Tabel 11. 1 Header UDP

Field	Panjang	Keterangan
Source Port	16 bit (2 byte)	Digunakan untuk mengidentifikasikan sumber protokol lapisan aplikasi yang mengirimkan pesan UDP yang bersangkutan. Pengunaan Source Port dapat berupa client atau server.
Destination Port	16 bit (2 byte)	Digunakan untuk mengidentifikasikan tujuan protokol lapisan aplikasi yang menjadi tujuan pesan UDP yang bersangkutan. Pengunaan Destination Port dapat berupa client atau server.
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 dan ukuran paling besar adalah 65515 bytes.
Checksum	16 bit (2 byte)	Berisi informasi pengecekan integritas dari pesan UDP yang dikirimkan apakah terjadi kesalahan dalam UDP.

Port UDP

UDP mempunyai port untuk mengirimkan informasi antar host, yang disebut dengan UDP Port. Sebuah UDP port berfungsi sebagai multiplexed message queue, yang artinya UDP port dapat menerima beberapa pesan sekaligus. Beberapa contoh aplikasi yang menggunakan UDP Port antara lain : DNS (Domain Name System), SNMP (Simple Network Management Protocol), TFTP dan SunRPC.

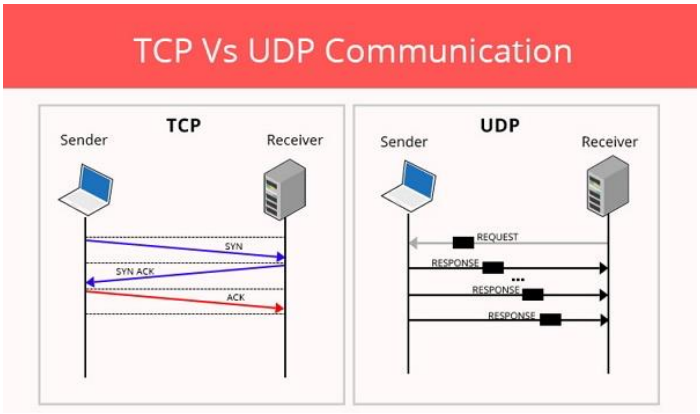
Layanan UDP

UDP dapat melakukan beberapa tugas sebagai berikut:

a. Lightweight: Beberapa protokol lapisan aplikasi membutuhkan penggunaan protokol yang ringan yang dapat melakukan fungsi-fungsi spesifik dengan saling bertukar pesan agar dapat menghemat sumber daya memori dan prosesor.
b. Protokol yang tidak membutuhkan keandalan. Contohnya adalah protokol Routing Information Protocol (RIP).
c. Transmisi broadcast: UDP merupakan protokol yang tidak perlu membuat koneksi terlebih dahulu dengan sebuah host, yang memungkinkan transmisi broadcast dilakukan. Sebuah protokol lapisan aplikasi dapat mengirimkan paket data ke beberapa tujuan dengan menggunakan mekanisme alamat multicast atau broadcast. Tentu berbeda dengan protokol TCP yang hanya dapat mengirimkan transmisi one-to-one.

Cara Kerja UDP

UDP memiliki sebuah saluran (channel) yang berguna untuk menghubungkan host antar host untuk saling berkirim informasi. Channel ini kemudian disebut dengan port UDP. Agar dapat terhubung dengan protokol UDP, aplikasi pada komputer terlebih dahulu perlu menyediakan alamat IP serta nomor port UDP dari host yang ingin dituju.



Gambar 11.2 Proses Komunikasi TCP vs UDP

Port UDP ini berguna sebagai sebuah multiplexed message queue. Artinya, port UDP tersebut kemudian mampu bekerja dengan menerima beberapa pesan secara bersamaan. Setiap port UDP memiliki identifikasi dengan nomor yang unik, namun memiliki pembagian tersendiri seperti yang telah dijelaskan sebelumnya.

Walaupun cukup bermanfaat, namun UDP sendiri memiliki beberapa kelemahan dalam cara kerjanya. Misalnya saja, UDP tidak memfasilitasi mekanisme *buffering* data masuk dan keluar. Selain itu, UDP juga tak memfasilitasi segmentasi data yang ukurannya besar untuk disederhanakan ke dalam segmen-segmen data, yang bisa dilakukan dengan TCP. Dalam protokol UDP juga tidak terdapat mekanisme *flow-control* seperti layaknya TCP.

TCP (Transmission Control Protocol)

TCP adalah standar komunikasi yang memungkinkan program aplikasi dan perangkat komputer dapat bertukar informasi melalui jaringan. TCP dirancang untuk mengirimkan data/informasi dan memastikannya terkirim lewat jaringan. Server dan klien dapat saling mentransmisikan data yang telah diatur oleh TCP, integritas data/informasi yang dikirimkan melalui jaringan juga akan terjamin. Sebelum mentransmisikan data, TCP membuat koneksi antara sumber dan tujuannya, kemudian barulah menguraikan data besar menjadi lebih kecil dan memastikan integritas data selama proses transmisi berlangsung.

Karakteristik TCP

TCP memiliki karakteristik sebagai berikut:

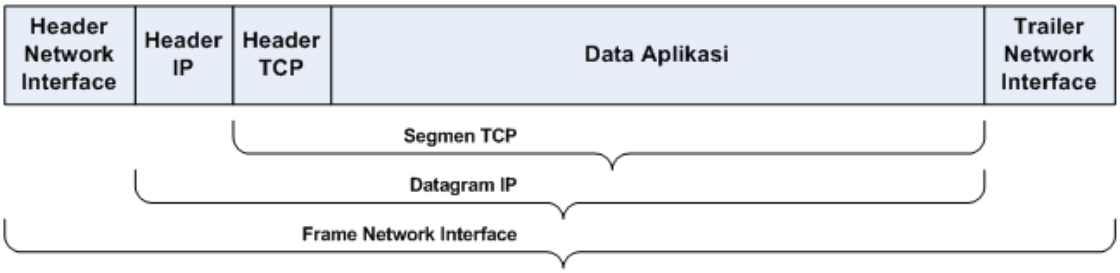
- Berorientasi sambungan (*connection-oriented*): Sebelum data dapat ditransmisikan antara dua host, dua proses yang berjalan pada lapisan aplikasi harus melakukan negosiasi untuk membuat sesi koneksi terlebih dahulu. Koneksi TCP ditutup dengan menggunakan proses terminasi koneksi TCP (TCP connection termination).
- *Full-duplex*: Untuk setiap host TCP, koneksi yang terjadi antara dua host terdiri atas dua buah jalur, yakni jalur keluar dan jalur masuk. Dengan menggunakan teknologi lapisan yang lebih rendah yang mendukung full-duplex, maka data pun dapat secara simultan diterima dan dikirim. Header TCP berisi nomor urut (TCP sequence number) dari data yang ditransmisikan dan sebuah acknowledgment dari data yang masuk.
- Dapat diandalkan (*reliable*): Data yang dikirimkan ke sebuah koneksi TCP akan diurutkan dengan sebuah nomor urut paket dan akan mengharapkan paket *positive acknowledgment* dari penerima. Jika tidak ada paket Acknowledgment dari penerima, maka segmen TCP (protocol data unit dalam protokol TCP) akan ditransmisikan ulang. Pada pihak penerima, segmen-segmen duplikat akan diabaikan dan segmen-segmen yang datang tidak sesuai dengan urutannya akan diletakkan di belakang untuk mengurutkan segmen-segmen TCP. Untuk menjamin integritas setiap segmen TCP, TCP mengimplementasikan penghitungan TCP Checksum.
- *Byte stream*: TCP melihat data yang dikirimkan dan diterima melalui dua jalur masuk dan jalur keluar TCP sebagai sebuah *byte stream* yang berdekatan (kontigu). Nomor urut TCP dan nomor acknowlegment dalam setiap header TCP didefinisikan juga dalam bentuk byte. Meski demikian, TCP tidak mengetahui batasan pesan-pesan di dalam byte stream TCP tersebut. Untuk melakukannya, hal ini diserahkan kepada protokol lapisan aplikasi (dalam DARPA Reference Model), yang harus menerjemahkan byte stream TCP ke dalam "bahasa" yang ia pahami.
- Memiliki layanan *flow control*: Untuk mencegah data terlalu banyak dikirimkan pada satu waktu, yang akhirnya membuat "macet" jaringan internetwork IP, TCP mengimplementasikan layanan *flow control* yang dimiliki oleh pihak pengirim yang secara terus menerus memantau dan membatasi jumlah data yang dikirimkan pada satu waktu. Untuk mencegah pihak penerima untuk memperoleh data yang tidak dapat disangganya (*buffer*), TCP juga mengimplementasikan *flow control* dalam pihak penerima, yang mengindikasikan jumlah *buffer* yang masih tersedia dalam pihak penerima.
- Melakukan segmentasi terhadap data yang datang dari lapisan aplikasi (dalam *DARPA Reference Model*)
- Mengirimkan paket secara "*one-to-one*": hal ini karena memang TCP harus membuat sebuah sirkuit logis antara dua buah protokol lapisan aplikasi agar saling

dapat berkomunikasi. TCP tidak menyediakan layanan pengiriman data secara *one-to-many*.

Segmen TCP

Segmen-segmen TCP akan dikirimkan sebagai datagram-datagram IP (datagram merupakan satuan protocol data unit pada lapisan *internetwork*). Sebuah segmen TCP terdiri atas sebuah *header* dan segmen data (*payload*), yang dienkapsulasi dengan menggunakan *header* IP dari protokol IP.

Sebuah segmen dapat berukuran hingga 65495 byte: 2^{16}-(ukuran header IP terkecil (20 byte)+ukuran header TCP terkecil (20 byte)). *Datagram* IP tersebut akan dienkapsulasi lagi dengan menggunakan header protokol network interface (lapisan pertama dalam DARPA Reference Model) menjadi frame lapisan Network Interface. Gambar berikut mengilustrasikan data yang dikirimkan ke sebuah host. Di dalam header IP dari sebuah segmen TCP, *field* **Source IP Address** diatur menjadi alamat *unicast* dari sebuah antarmuka host yang mengirimkan segmen TCP yang bersangkutan. Sementara itu, field **Destination IP Address** juga akan diatur menjadi alamat *unicast* dari sebuah antarmuka host tertentu yang dituju. Hal ini dikarenakan, protokol TCP hanya mendukung transmisi *one-to-one*.



Gambar 11.3 Segemen TCP

Header TCP

Ukuran dari header TCP adalah bervariasi, yang terdiri atas beberapa field yang ditunjukkan dalam gambar dan tabel berikut. Ukuran TCP header paling kecil (ketika tidak ada tambahan opsi TCP) adalah 20 byte.

Tabel 11.2 Header TCP

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 **source *socket***, 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.
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.
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.
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*.

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.

Port TCP

Port TCP mampu mengindikasikan sebuah lokasi tertentu untuk menyampaikan segmen-segmen TCP yang dikirimkan yang diidentifikasi dengan **TCP Port Number**. Nomor-nomor di bawah angka 1024 merupakan port yang umum digunakan dan ditetapkan oleh [[IANA|IANAplikasi, sementara port UDP merepresentasikan sebuah antrean pesan UDP untuk protokol lapisan aplikasi. Selain itu, protokol lapisan aplikasi yang menggunakan port TCP dan port UDP dalam nomor yang sama juga tidak harus sama. Sebagai contoh protokol Extended Filename Server (EFS) menggunakan port TCP dengan nomor 520, dan protokol Routing Information Protocol (RIP) menggunakan port UDP juga dengan nomor 520. Jelas, dua protokol tersebut sangatlah berbeda! Karenanya, untuk menyebutkan sebuah nomor port, sebutkan juga jenis port yang digunakannya, karena hal tersebut mampu membingungkan (ambigu).

Beberapa contoh aplikasi yang menggunakan protokol TCP antara lain TELNET, FTP (File Transfer Protocol), dan SMTP (Simple Mail Transfer Protocol).

TCP Flag

Sebuah segmen TCP dapat memiliki flag (tanda-tanda) khusus yang mengindikasikan segmen yang bersangkutan, seperti yang disebutkan dalam tabel berikut:

Tabel 11.4 TCP Flag

Nama Flag	Keterangan

URG	Mengindikasikan bahwa beberapa bagian dari segmen TCP mengandung data yang sangat penting, dan field Urgent Pointer dalam header TCP harus digunakan untuk menentukan lokasi di mana data penting tersebut berada dalam segmen.
ACK	Mengindikasikan field Acknowledgment mengandung oktet selanjutnya yang diharapkan dalam koneksi. Flag ini selalu diset, kecuali pada segmen pertama pada pembuatan sesi koneksi TCP.
PSH	Mengindikasikan bahwa isi dari TCP Receive buffer harus diserahkan kepada protokol lapisan aplikasi. Data dalam receive buffer harus berisi sebuah blok data yang berurutan (kontigu), dilihat dari ujung paling kiri dari buffer. Dengan kata lain, sebuah segmen yang memiliki flag PSH diset ke nilai 1, tidak bolah ada satu byte pun data yang hilang dari aliran byte segmen tersebut; data tidak dapat diberikan kepada protokol lapisan aplikasi hingga segmen yang hilang tersebut datang. Normalnya, TCP Receive buffer akan dikosongkan (dengan kata lain, isi dari buffer akan diteruskan kepada protokol lapisan aplikasi) ketika buffer tersebut berisi data yang kontigu atau ketika dalam "proses perawatan". Flag PSH ini dapat mengubah hal seperti itu, dan membuat akan TCP segera mengosongkan TCP Receive buffer. Flag PSH umumnya digunakan dalam protokol lapisan aplikasi yang bersifat interaktif, seperti halnya Telnet, karena setiap penekanan tombol dalam sesi terminal virtual akan dikirimkan dengan sebuah flag PSH diset ke nilai 1. Contoh dari penggunaan lainnya dari flag ini adalah pada segmen terakhir dari berkas yang ditransfer dengan menggunakan protokol FTP. Segmen yang dikirimkan dengan flag PSH aktif tidak harus segera di-acknowledge oleh penerima.
RST	Mengindikasikan bahwa koneksi yang dibuat akan digagalkan. Untuk sebuah koneksi TCP yang sedang berjalan (aktif), sebuah segmen dengan flag RST diset ke nilai 1 akan dikirimkan sebagai respons terhadap sebuah segmen TCP yang diterima yang ternyata segmen tersebut bukan yang diminta, sehingga koneksi pun menjadi gagal. Pengiriman segmen dengan flag RST diset ke nilai 1 untuk sebuah koneksi aktif akan menutup koneksi secara paksa, sehingga data yang disimpan dalam buffer akan dibuang (dihilangkan). Untuk sebuah koneksi TCP yang sedang dibuat, segmen dengan flag RST aktif akan

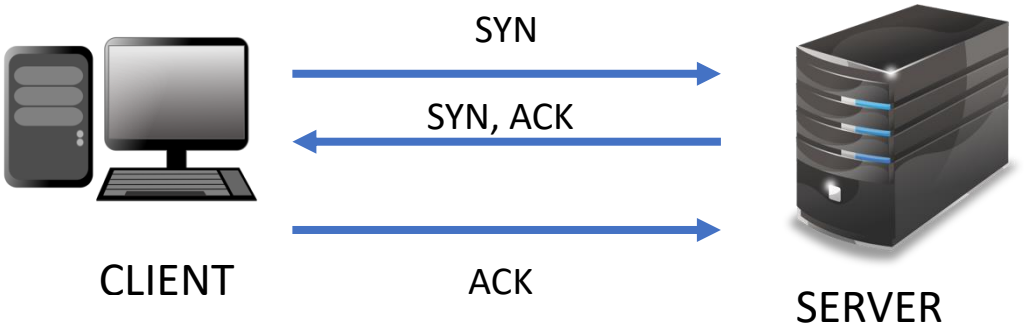
	dikirimkan sebagai respons terhadap request pembuatan koneksi untuk mencegah percobaan pembuatan koneksi.
SYN	Mengindikasikan bahwa segmen TCP yang bersangkutan mengandung Initial Sequence Number (ISN). Selama proses pembuatan sesi koneksi TCP, TCP akan mengirimkan sebuah segmen dengan flag SYN diset ke nilai 1. Setiap host TCP lainnya akan memberikan jawaban (acknowledgment) dari segmen dengan flag SYN tersebut dengan menganggap bahwa segmen tersebut merupakan sekumpulan byte dari data. Field Acknowledgment Number dari sebuah segmen SYN diatur ke nilai ISN + 1.
FIN	Menandakan bahwa pengirim segmen TCP telah selesai dalam mengirimkan data dalam sebuah koneksi TCP. Ketika sebuah koneksi TCP akhirnya dihentikan (akibat sudah tidak ada data yang dikirimkan lagi), setiap host TCP akan mengirimkan sebuah segmen TCP dengan flag FIN diset ke nilai 1. Sebuah host TCP tidak akan mengirimkan segmen dengan flag FIN hingga semua data yang dikirimkannya telah diterima dengan baik (menerima paket acknowledgment) oleh penerima. Setiap host akan menganggap sebuah segmen TCP dengan flag FIN sebagai sekumpulan byte dari data. Ketika dua host TCP telah mengirimkan segmen TCP dengan flag FIN dan menerima acknowledgment dari segmen tersebut, maka koneksi TCP pun akan dihentikan.

TCP Three-way handshake

Proses pembuatan koneksi TCP disebut juga dengan "**Three-way Handshake**". Tujuan metode ini adalah agar dapat melakukan sinkronisasi terhadap nomor urut dan nomor acknowledgement yang dikirimkan oleh kedua pihak dan saling bertukar ukuran TCP Window. Prosesnya dapat digambarkan sebagai berikut:

- *Host* pertama (yang ingin membuat koneksi) akan mengirimkan sebuah segmen TCP dengan flag SYN diaktifkan kepada *host* kedua (yang hendak diajak untuk berkomunikasi).
- *Host* kedua akan meresponsnya dengan mengirimkan segmen dengan *acknowledgment* dan juga SYN kepada *host* pertama.
- *Host* pertama selanjutnya akan mulai saling bertukar data dengan host kedua.

TCP menggunakan proses jabat tangan yang sama untuk mengakhiri koneksi yang dibuat. Hal ini menjamin dua *host* yang sedang terkoneksi tersebut telah menyelesaikan proses transmisi data dan semua data yang ditransmisikan telah diterima dengan baik. Itulah sebabnya, mengapa TCP disebut dengan koneksi yang *reliable*.



Gambar 11.4 Proses Three way Handshake

Cara Kerja TCP

Pertama, data dipecah menjadi bagian-bagian kecil yang bertujuan agar dapat ditransmisikan dapat berjalan dengan akurat tanpa harus mengirim ulang ketika mengalami kendala. Data yang telah dipecah akan otomatis kembali tergabung seperti semula ketika sudah sampai ke tujuan yang diinginkan. Data yang dikirimkan akan melewati rute yang berbeda-beda antara sumber dan perangkat tujuan.Sebelum sampai di perangkat tujuan, data akan melewati beberapa layer. Tahap ini akan menerjemahkan sinyal dan memastikan data akan terkirim ke tujuan.

Daftar Pustaka

1. Lukas, J., 2006, Jaringan Komputer, Graha Ilmu, Yogyakarta
2. Sutanta, E., 2005, Komunikasi Data & Jaringan Komputer, Graha Ilmu, Yogyakarta
3. Kurose, Ross, 2017, Computer Networking, A Top-Down Approach (Seventh Edition), Pearson, New York