Bilgisayar Ağları
Computer Networks

Hazırlayan: M. Ali Akcayol
Gazi Üniversitesi
Bilgisayar Mühendisliği Bölümü

Bu dersin sunumları, "James Kurose, Keith Ross, Computer Networking: A Top-Down Approach 6/e, Pearson, 2013." kitabı kullanılarak hazırlanmıştır.

İçerik

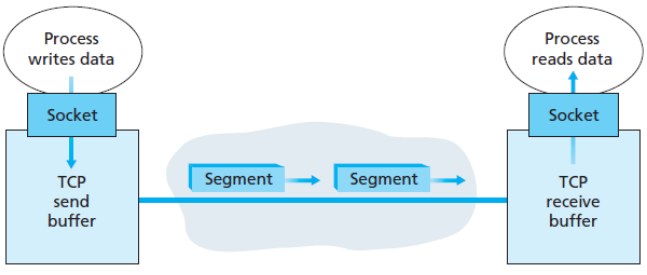
- Bağlantı temelli iletişim: TCP
- TCP segment yapısı
- RTT tahmini
- Güvenilir veri aktarımı
- Akış denetimi
- TCP bağlantı yönetimi

Bağlantı temelli iletişim: TCP

- **TCP,** İnternetin **bağlantı temelli ve güvenilir iletişim protokolüdür.**
- TCP kullanan process'ler iletişime başlamadan önce **handshake** yapmak zorundadır.
- **TCP** bağlantısı **full-duplex servis sağlar.**
- TCP bağlantısı her zaman point-to-point **(bir gönderici-bir alıcı)** iletişim yapar.
- İstemci-sunucu, sunucu-istemci ve istemci-sunucu (three-way hand-shake) arasında **özel TCP segmentleri ile bağlantı başlatılır.**
- Maksimum segment size (MSS), link layer'daki maksimum transmission unit'e (MTU) göre belirlenir (Ethernet ve PPP için 1500byte).

Bağlantı temelli iletişim: TCP

- TCP **istemci** ve **sunucu** tarafta **buffer vardır.**
- TCP segmenti, IP datagramı içerisine encapsulate edilir.
- İki taraf arasında **buffer doluluk durumuna göre akış denetimi yapılır.**



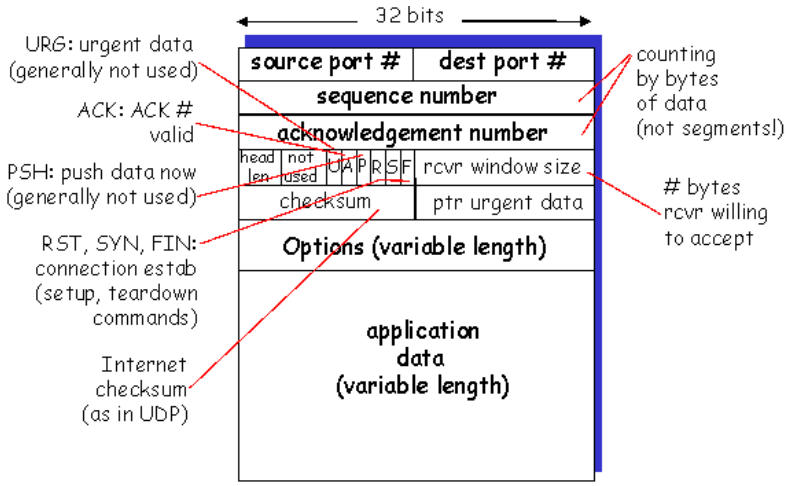
TCP send and receive buffers

İçerik

- Bağlantı temelli iletişim: TCP
- TCP segment yapısı
- RTT tahmini
- Güvenilir veri aktarımı
- Akış denetimi
- TCP bağlantı yönetimi

TCP segment yapısı

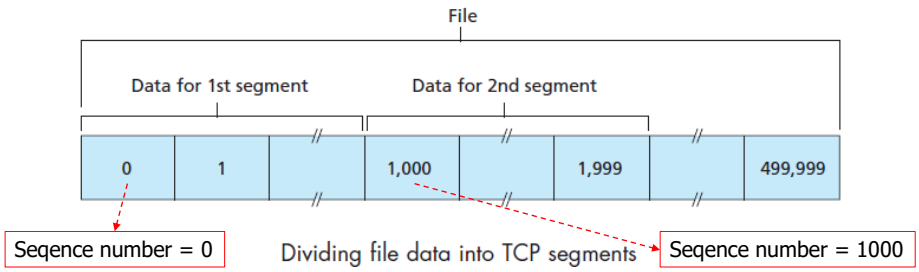
- TCP segmenti, başlık alanları ile veri alanına sahiptir.



TCP segment yapısı

ACK ve sequence number

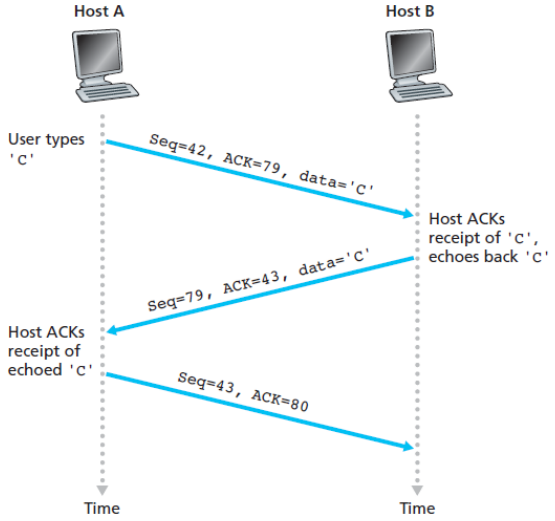
- **TCP byte bazında sıralama yapar.**
- **Seq(n):** segmentteki **ilk byte'ın sıra numarasını gösterir.**
- Segment içindeki **tüm byte'lar sayılır.**
- ACK(n) karşı taraftan beklenen byte'ın sıra numarasını gösterir.
- **ACK(n):** Gönderici cumulative ACK yapar.



Dividing file data into TCP segments

TCP segment yapısı

- Telnet senaryosu



Sequence and acknowledgment numbers for a simple Telnet application over TCP

İçerik

- Bağlantı temelli iletişim: TCP
- TCP segment yapısı
- RTT tahmini
- Güvenilir veri aktarımı
- Akış denetimi
- TCP bağlantı yönetimi

RTT tahmini

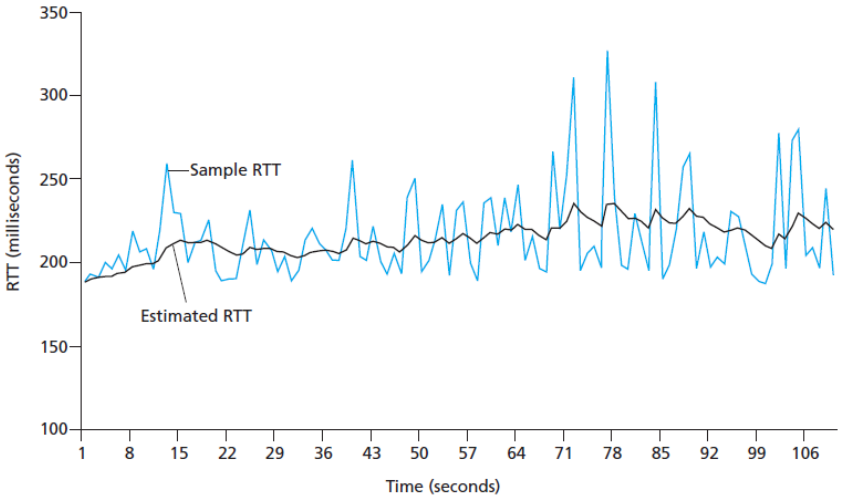
- Timeout **çok kısa** olursa **erken timeout** (premature) olur ve **gereksiz retransmit yapılır.**
- Timeout süresi **çok uzun** olursa kayıp paketleri **retransmit etmek çok gecikir.**
- Bir segmentin gönderilip cevabının (ACK) gelmesi için geçen süre ölçülür (**`SampleRTT`**).
- **`SampleRTT`** değerleri ile ortalama **`EstimatedRTT`** hesaplanır.
- **`SampleRTT`** hızlı değişir, **`EstimatedRTT`** yavaş değişir.
- Eski tahmin değeri azaltılarak kullanılır (exponential weighted moving average).

$$\texttt{EstimatedRTT} = (1 - \alpha) \cdot \texttt{EstimatedRTT} + \alpha \cdot \texttt{SampleRTT}$$

$\alpha = 0.125$ [RFC 6298]

RTT tahmini

- **`EstimatedRTT`** güven aralığı belirler.



RTT samples and RTT estimates

RTT tahmini

Timeout süresinin hesaplanması

- **`SampleRTT`**'nin **`EstimatedRTT`**'den ne kadar saptığı tahmin edilir (**`DevRTT`**).

$$DevRTT = (1 - \beta) \cdot DevRTT + \beta \cdot |SampleRTT - EstimatedRTT|$$

$\beta = 0.25$

- Timeout aralığı hesaplanırken **`DevRTT`**'nin ağırlığı daha yüksektir.

$$TimeoutInterval = EstimatedRTT + 4 \cdot DevRTT$$

İçerik

- Bağlantı temelli iletişim: TCP
- TCP segment yapısı
- RTT tahmini
- Güvenilir veri aktarımı
- Akış denetimi
- TCP bağlantı yönetimi

Güvenilir veri aktarımı

- TCP:
  - **IP'nin** unreliable best-effort servisinin üstüne **reliable servis oluşturur.**
  - Bir tane retransmission timer'ı kullanır.
  - **Cumulative AKC** kullanır.
  - **Pipelining** ile segment gönderimi yapar.
- TCP'de retransmit:
  - **Timeout** olduğunda yapılır.
  - **Duplicate ACK** gelince yapılır.
- TCP:
  - **Byte-stream sıra numarası** verir.
  - **Seq# segmentteki ilk byte'ın sırasını** gösterir.

Güvenilir veri aktarımı

- Application layer'dan veri geldiğinde:
  - **Sonraki Seq#** ile bir segment oluşturulur.
  - Seq# ilk byte'ın sırasıdır.
  - **Timer çalışmıyorsa başlatılır** (Timer en eski ACK beklenen paket için çalışır).
  - **Expiration interval,** TimeOutInterval ile **belirlenir.**
- Timeout:
  - Timeout'a neden olan **segment retransmit edilir.**
  - **Timer restart** yapılır.
- ACK# alındı:
  - ACK gelen **segmentin durumu güncellenir.**
  - ACK **bekleyen segment varsa timer start edilir.**

Güvenilir veri aktarımı

```
NextSeqNum=InitialSeqNumber
SendBase=InitialSeqNumber

loop (forever) {
    switch(event)

        event: data received from application above
            create TCP segment with sequence number NextSeqNum
            if (timer currently not running)*
                start timer
            pass segment to IP
            NextSeqNum=NextSeqNum+length(data)*
            break;

        event: timer timeout
            retransmit not-yet-acknowledged segment with
                smallest sequence number*
            start timer
            break;

        event: ACK received, with ACK field value of y
            if (y > SendBase) {*
                SendBase=y
                if (there are currently any not-yet-acknowledged segments)
                    start timer
                }
            break;

    } /* end of loop forever */
```

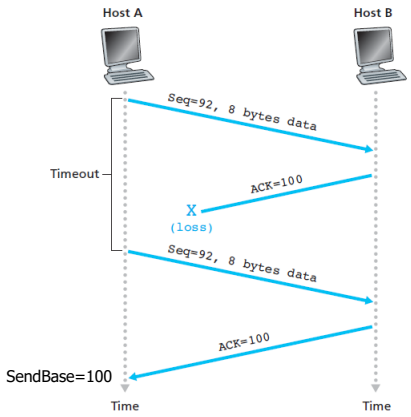
Simplified TCP sender

Varsayımlar:
- Flow kontrol yok
- Congestion kontrol yok
- Veri MSS'den küçük
- Aktarım tek yönlü

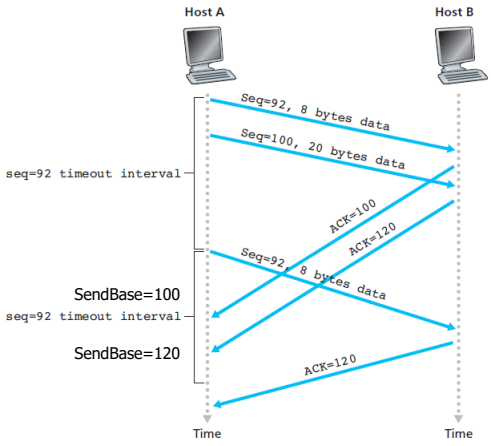
Açıklama:
- SendBase-1: En son toplu ACK'lanan byte
- y > SendBase ise yeni segment ACK'lanır

Örnek:
- **SendBase-1 = 21** ve **y = 73** ise, **alıcı 73+ bekliyor.**

Güvenilir veri aktarımı



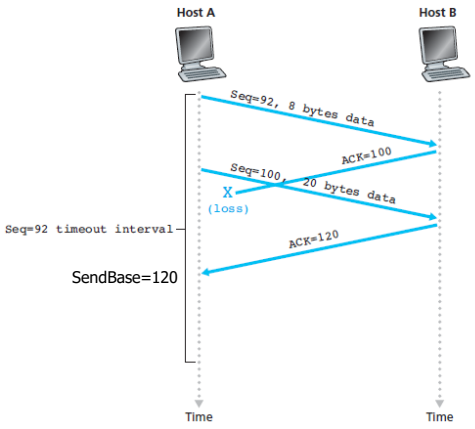
Retransmission due to a lost acknowledgment



Segment 100 not retransmitted

Premature timeout

Güvenilir veri aktarımı



A cumulative acknowledgment avoids retransmission of the first segment

Güvenilir veri aktarımı

TCP alıcının ACK oluşturma kuralları

Event	TCP Receiver Action
Arrival of in-order segment with expected sequence number. All data up to expected sequence number already acknowledged.	Delayed ACK. Wait up to 500 msec for arrival of another in-order segment. If next in-order segment does not arrive in this interval, send an ACK.
Arrival of in-order segment with expected sequence number. One other in-order segment waiting for ACK transmission.	Immediately send single cumulative ACK, ACKing both in-order segments.
Arrival of out-of-order segment with higher-than-expected sequence number. Gap detected.	Immediately send duplicate ACK, indicating sequence number of next expected byte (which is the lower end of the gap).
Arrival of segment that partially or completely fills in gap in received data.	Immediately send ACK, provided that segment starts at the lower end of gap.

TCP ACK Generation Recommendation [RFC 5681]

Güvenilir veri aktarımı

Fast retransmit

- **Timeout süresi genellikle uzundur.**
- Kaybolan paketi **retransmit etmek için beklenen süre uzundur.**
- TCP alıcı **sırasız gelen segmentlerde** en eski **beklenen byte'ın sıra numarasını ACK ile ister.**
- **Gönderici** art arda **çok sayıda segment gönderir.**
- **Kaybolan** bir **segment için çok sayıda ACK alınır.**
- Gönderici **aynı segment için 3 ACK alırsa** segmentin **kaybolduğunu varsayar.**
- **3 ACK aldığında** timer timeout olmadan **retransmit eder** (fast retransmit).

Güvenilir veri aktarımı

Fast retransmit

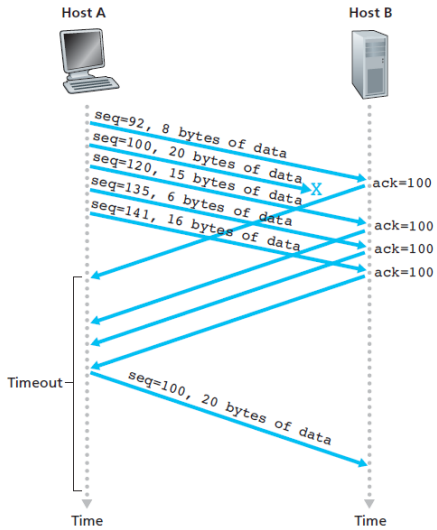
```
event: ACK received, with ACK field value of y
            if (y > SendBase) {
                    SendBase=y
                    if (there are currently any not yet
                                acknowledged segments)
                        start timer
                    }
            else { /* a duplicate ACK for already ACKed
                    segment */
                increment number of duplicate ACKs
                    received for y
                if (number of duplicate ACKS received
                    for y==3)
                    /* TCP fast retransmit */
                    resend segment with sequence number y
                }
            break;
```

Duplicate ACK

Fast retransmit

Güvenilir veri aktarımı

Fast retransmit



Fast retransmit: retransmitting the missing segment before the segment's timer expires

İçerik

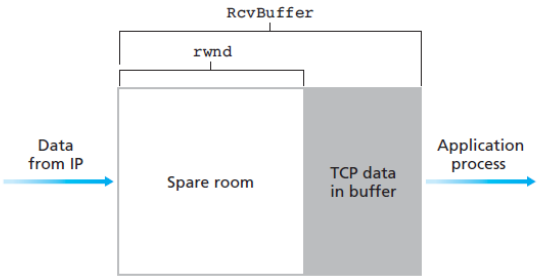
- Bağlantı temelli iletişim: TCP
- TCP segment yapısı
- RTT tahmini
- Güvenilir veri aktarımı
- Akış denetimi
- TCP bağlantı yönetimi

Akış denetimi

- TCP'de gönderici ve alıcı buffer'a sahiptir.
- **Buffer'a gelme hızı çıkma hızından yüksekse dolmaya başlar.**
- TCP alıcı kendi **buffer'ı dolmadan göndericiyi uyarır** (flow control).
- Akış denetimi **hız uyumlama servisidir.**
- Akış denetimini **alıcı başlatır ve yönetir.**
- Tıkanıklık denetimini (congestion control) **gönderici kendisi başlatır ve yönetir.**

Akış denetimi

- **`LastByteRcvd`**: Buffer'a son eklenen byte.
- **`LastByteRead`**: Process'in buffer'dan son okuduğu byte.
- **`rwnd`**, segmentin içindeki ReceiveWindow alanına yazılır.



The receive window (`rwnd`) and the receive buffer (`RcvBuffer`)

$$\texttt{LastByteRcvd} - \texttt{LastByteRead} \leq \texttt{RcvBuffer}$$

$$\texttt{rwnd} = \texttt{RcvBuffer} - [\texttt{LastByteRcvd} - \texttt{LastByteRead}]$$

İçerik

- Bağlantı temelli iletişim: TCP
- TCP segment yapısı
- RTT tahmini
- Güvenilir veri aktarımı
- Akış denetimi
- TCP bağlantı yönetimi

TCP bağlantı yönetimi

- TCP istemci ve TCP sunucu arasında **veri aktarımı başlamadan önce bağlantı kurulur** (three-way handshake).
- İki taraf TCP değişkenlerine **başlangıç değerleri** atar:
  - **Seq# belirlenir** (rastgele).
  - **Buffer boyutları belirlenir.**
  - Akış denetimi için **RcvWindow boyutu belirlenir.**
- TCP istemci bağlantı isteğini TCP sunucuya iletir.

```
Socket clientSocket = new  Socket("hostname","port number");
```

- TCP sunucu istemci ile bağlantı kurar.

```
Socket connectionSocket = welcomeSocket.accept();
```


TCP bağlantı yönetimi

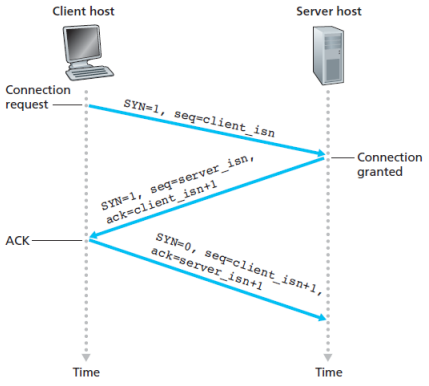
Three-way handshake

- Adım 1: (Gönderici)
  - TCP **istemci özel bir segmenti TCP sunucuya gönderir.**
  - Bu özel TCP segmentinde **SYN biti 1** yapılır (SYN segment).
  - İstemci rastgele bir **Seq# belirler** (`client_isn`).
  - Bu segment IP paketine encapsulate edilir ve gönderilir.
- Adım 2: (Alıcı)
  - TCP **sunucu** IP datagramından **SYN segmentini extract eder.**
  - TCP buffer ve değişkenlerini belirler (SYN saldırısı için zayıflık (DoS)).
  - TCP sunucu **özel bir segmenti** (SYNACK) **istemciye gönderir.**
  - **SYN biti 1 yapılır**, segmentteki ACK# = `client_isn+1` yapılır.
  - Sunucu rastgele bir **Seq# belirler** (`server_isn`).
- Adım 3: (Gönderici)
  - TCP istemci SYNACK alınca buffer ve değişkenleri belirler.
  - TCP **istemci sunucuya özel segment gönderir** (veri içerebilir).
  - **SYN biti 0 yapılır**, ACK# = `server_isn+1` yapılır.

TCP bağlantı yönetimi

Three-way handshake

- Üçüncü segment uygulama katmanı verisi içerebilir.



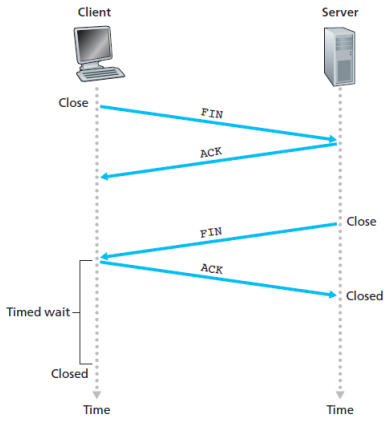
TCP three-way handshake: segment exchange

TCP bağlantı yönetimi

- Bağlantı sonlandırmayı TCP istemci başlatır.
- Adım 1:
  - TCP **istemci bağlantı sonlandırma isteğini** içeren **özel bir segmenti** sunucuya iletir.
  - Bu özel segmentin **FIN biti 1** yapılır.
- Adım 2:
  - **TCP sunucu FIN biti 1 olan segmenti alır** ve **ACK gönderir.**
  - Sunucu bağlantıyı kapatır ve **FIN biti 1 olan özel bir segmenti istemciye gönderir.**
- Adım 3:
  - **TCP istemci FIN biti 1 olan segmenti alır** ve **ACK gönderir.**
  - **İstemci** bir süre bekler ve **bağlantıyı sonlandırır.**
- Adım 4:
  - **Sunucu ACK segmentini alır** ve **bağlantıyı sonlandırır.**

TCP bağlantı yönetimi

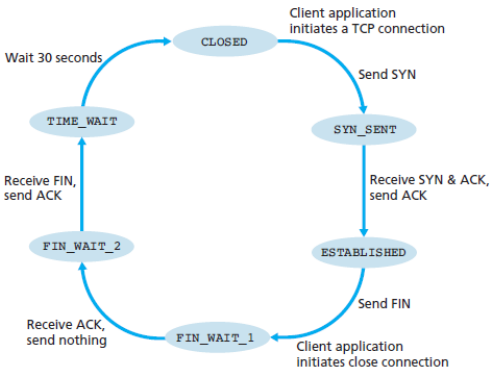
- Bağlantı sonlandırıldığında kaynaklar (buffer, değişkenler) serbest bırakılır.



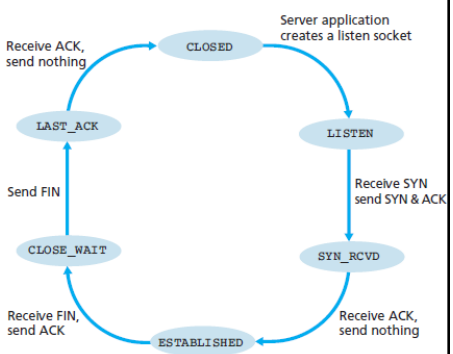
Closing a TCP connection

TCP bağlantı yönetimi

- TCP istemci ve sunucu bağlantı sonlandırma sürecinde farklı durumlar arasında geçiş yapar.



A typical sequence of TCP states visited by a client TCP



A typical sequence of TCP states visited by a server-side TCP