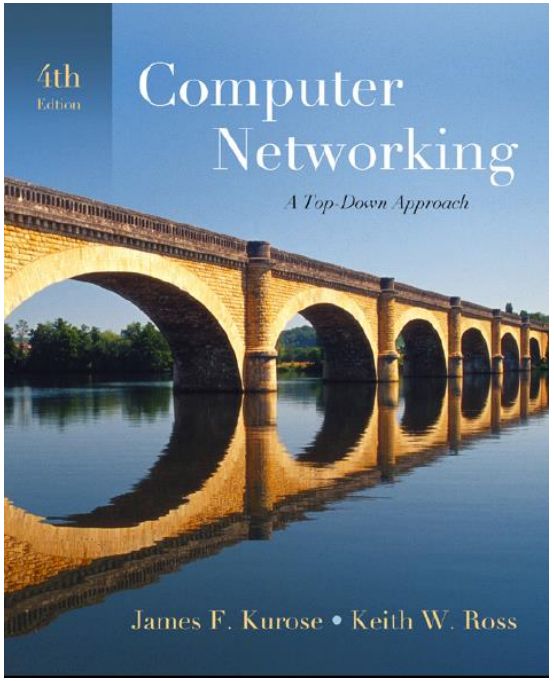
3. Bölüm
Taşıma Katmanı



Computer Networking: A Top Down Approach 4th edition.
Jim Kurose, Keith Ross
Addison-Wesley, July 2007.

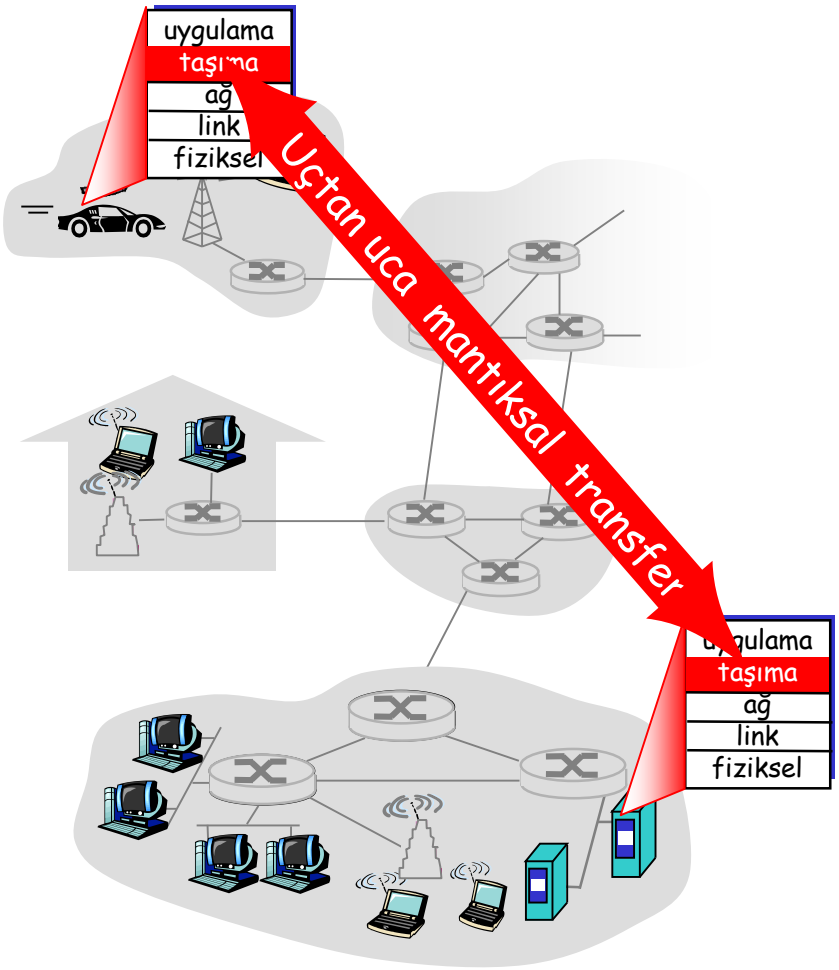
3. Bölüm: Taşıma Katmanı

Amaçlarımız:

- Taşıma katmanı servislerinin arkasındaki prensipleri anlamak:
  - multiplexing/demultiplexing
  - Güvenilir data transferi
  - akış kontrolü
  - Sıkışıklık kontrolü
- İnternetin taşıma katmanı protokollerini öğrenmek:
  - UDP: bağlantısız taşıma
  - TCP: bağlantılı taşıma
  - TCP sıkışıklık kontrolü

Taşıma servisleri ve protokolleri

- Farklı hostlarda çalışan uygulamalar için *mantıksal haberleşme* sağlar
- Taşıma protokolleri uç sistemlerde çalışır
  - Gönderici taraf: mesajları önce segmentlere böler,sonra ağ katmanına aktarır
  - Alıcı taraf: segmentleri birleştirerek mesajı tekrar oluşturur ve uygulama katmanına aktarır
- Uygulamalar için birden fazla taşıma protokolü vardır
  - Internet: TCP ve UDP

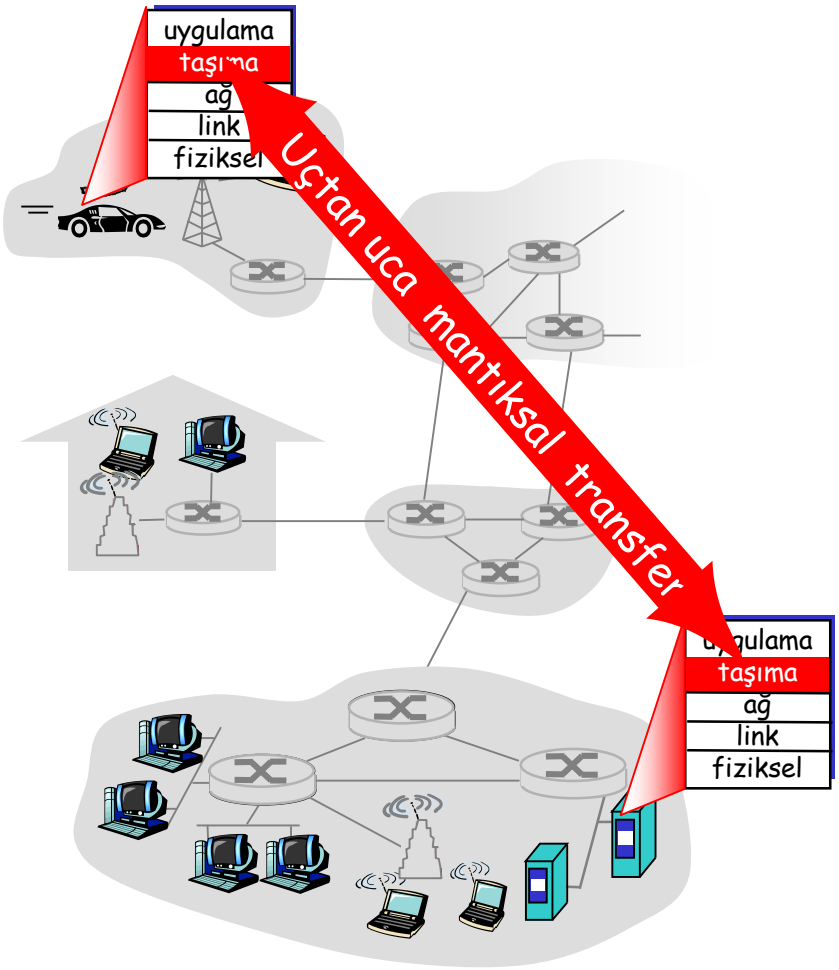


Taşıma vs. Ağ katmanı

- *Ağ katmanı:* hostlar arasında mantıksal haberleşme
- *Taşıma katmanı:* işlemler arasında mantıksal haberleşme
  - Ağ katmanı servislerine dayanır ve onları iyileştirir

Internet taşıma katmanı protokolleri

- güvenilir, sıralı iletim (TCP)
  - Sıkışıklık kontrolü
  - Akış kontrolü
  - Bağlantı kurulumu
- Güvenilir değil, sıralı olmayan iletim: UDP
  - Elinden gelenin en iyisini yapma "best-effort" IP
- Olmayan servisler:
  - Gecikme garantisi
  - Bant genişliği garantisi



Multiplexing/demultiplexing

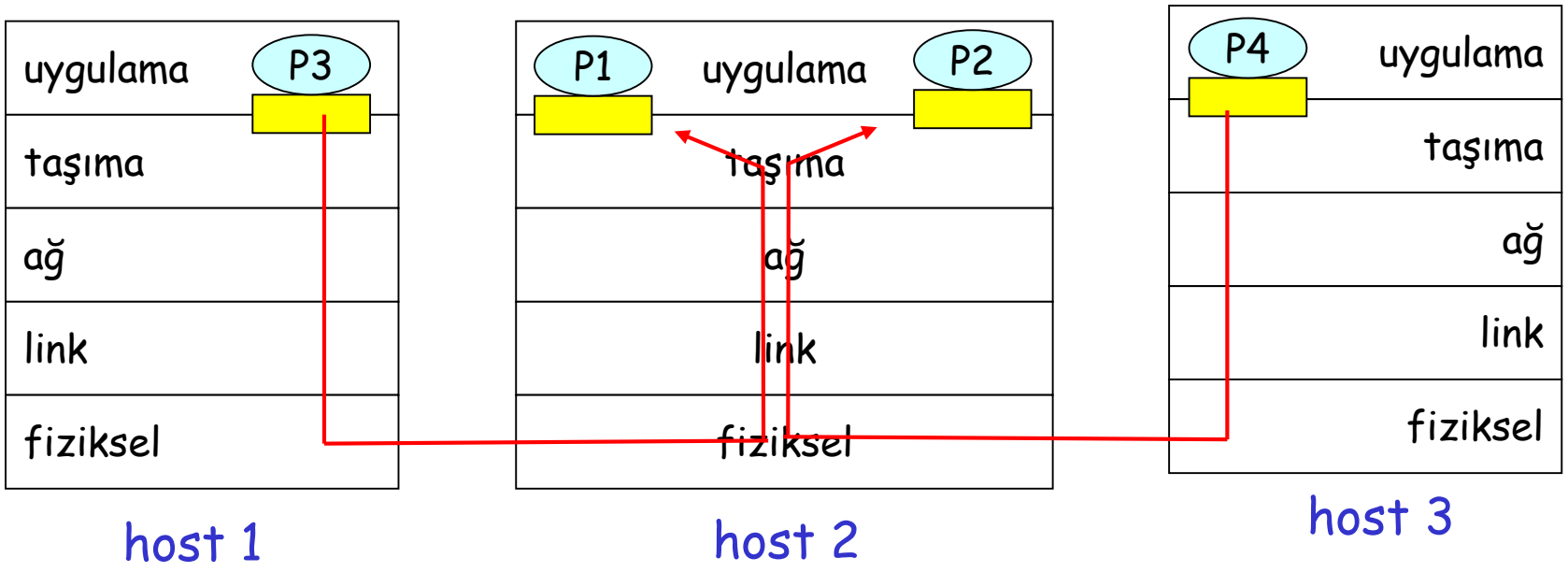
Alıcıda demultiplexing:

Alınan segmentleri doğru hosta göndermek

Göndericide Multiplexing

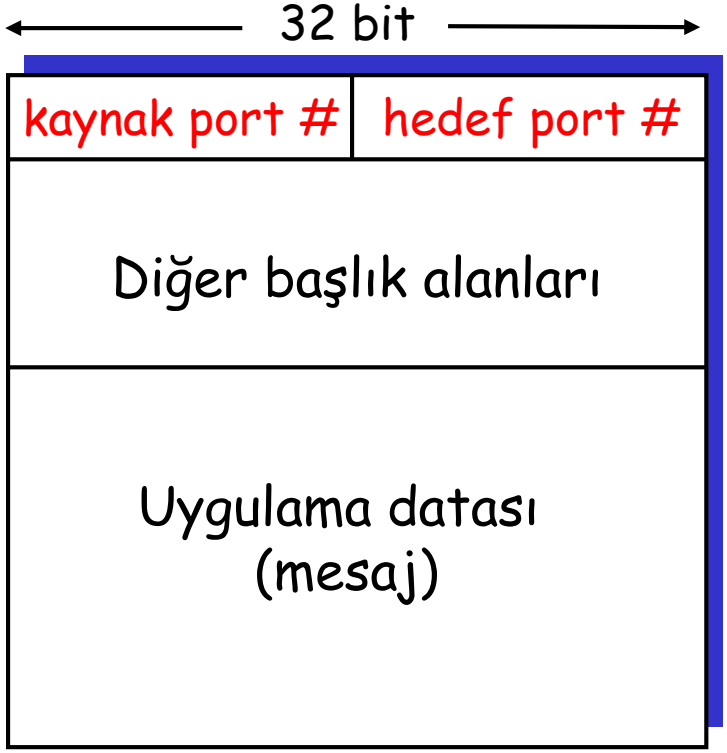
Birçok soketten data toplayıp veri başlığı ile sarmak (daha sonra demultiplexing için kullanılmak üzere)

= soket = işlem



Demultiplexing nasıl çalışır

- host IP datagramlarını alır
  - herbir datagram kaynak IP adresine ve hedef IP adresine sahiptir
  - Herbir datagram 1 taşıma katmanı segmenti taşır
  - Herbir segment kaynak ve hedef port numarası vardır
- Host, IP adreslerini ve port numaralarını segmentleri uygun soketlere yönlendirmek için kullanır



TCP/UDP segment formatı

Bağlantısız demultiplexing

- Port numaralarıyla soketleri oluşturur:

```
DatagramSocket mySocket1 = new
   DatagramSocket(12534);
DatagramSocket mySocket2 = new
   DatagramSocket(12535);
```

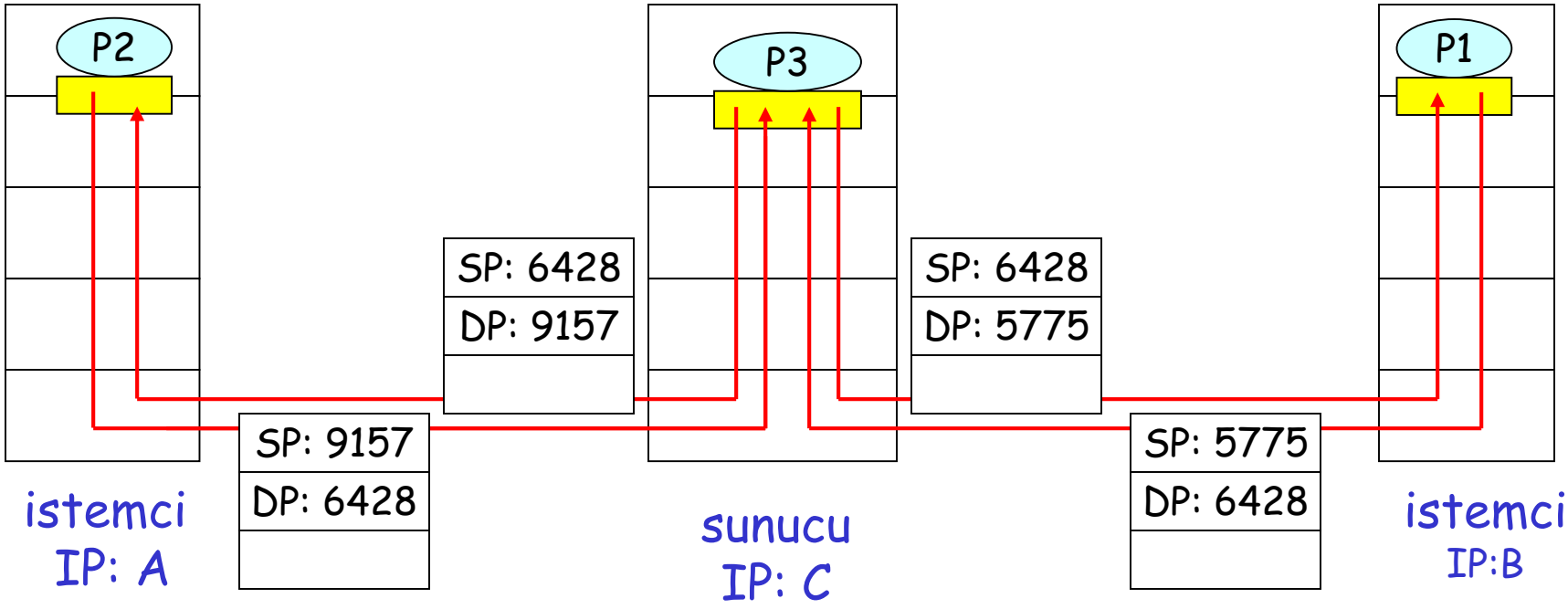
- UDP soketi iki unsurla tanımlanır:

(hedef IP adresi, hedef port numarası)

- host UDP segmentini alınca:
  - Segmentteki hedef port numarasını kontrol eder
  - UDP segmentini ait olduğu sokete yönlendirir
- Farklı kaynak IP adresine ve kaynak port numarasına sahip datagramlar aynı sokete geliyor

Bağlantısız demultiplexing

```
DatagramSocket serverSocket = new DatagramSocket(6428);
```



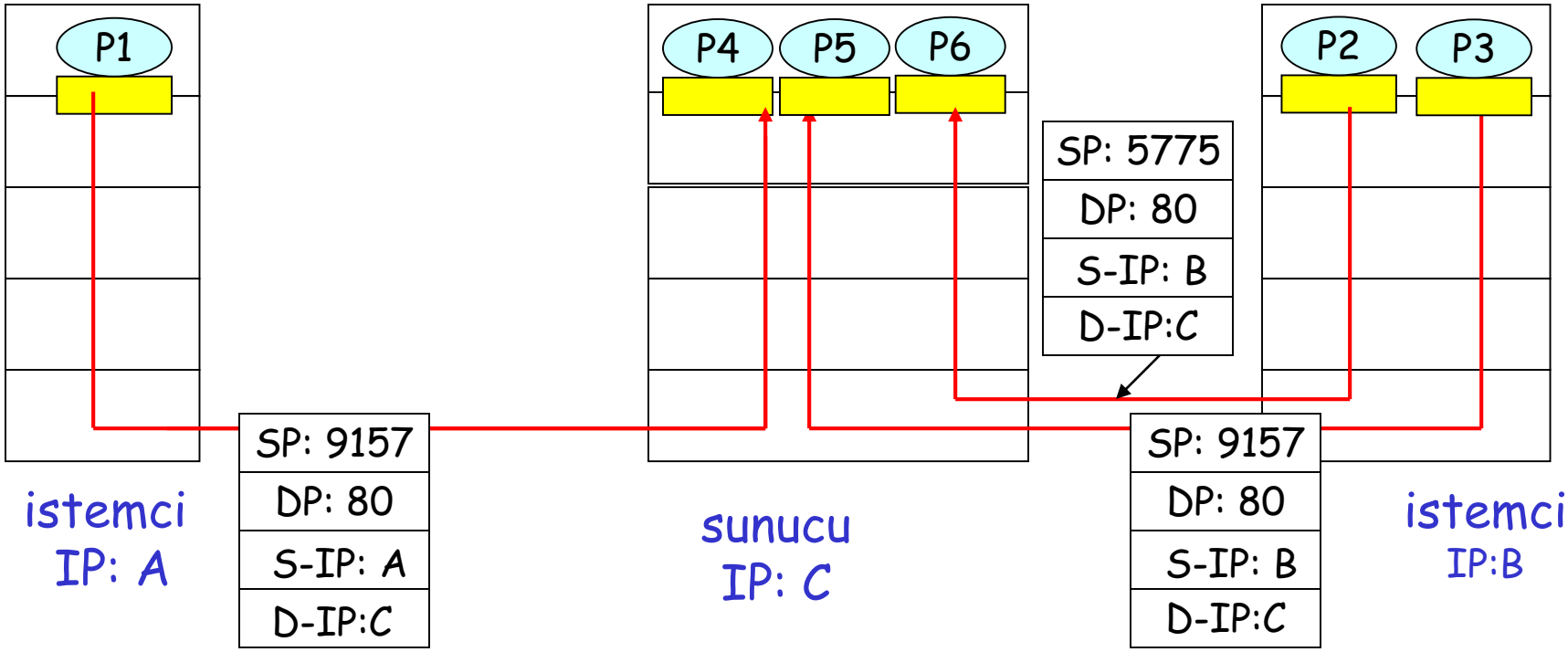
SP "dönüş adresini" verir

Bağlantılı demux

- TCP soketi 4 unsurla tanımlanır:
  - kaynak IP adresi
  - kaynak port numarası
  - hedef IP adresi
  - hedef port numarası
- alıcı host bütün 4 unsuru da segmenti uygun sokete yönlendirmek için kullanır

- Sunucu hostu eşzamanlı olarak birçok TCP soketini destekleyebilir:
  - her soket kendi 4 unsuru ile tanımlanır
- Web sunucuları her bağlanan istemci için farklı soketlere sahiptir
  - Kalıcı olmayan HTTP'de her istek için farklı soket olacaktır

Bağlantılı demux



UDP: User Datagram Protocol [RFC 768]

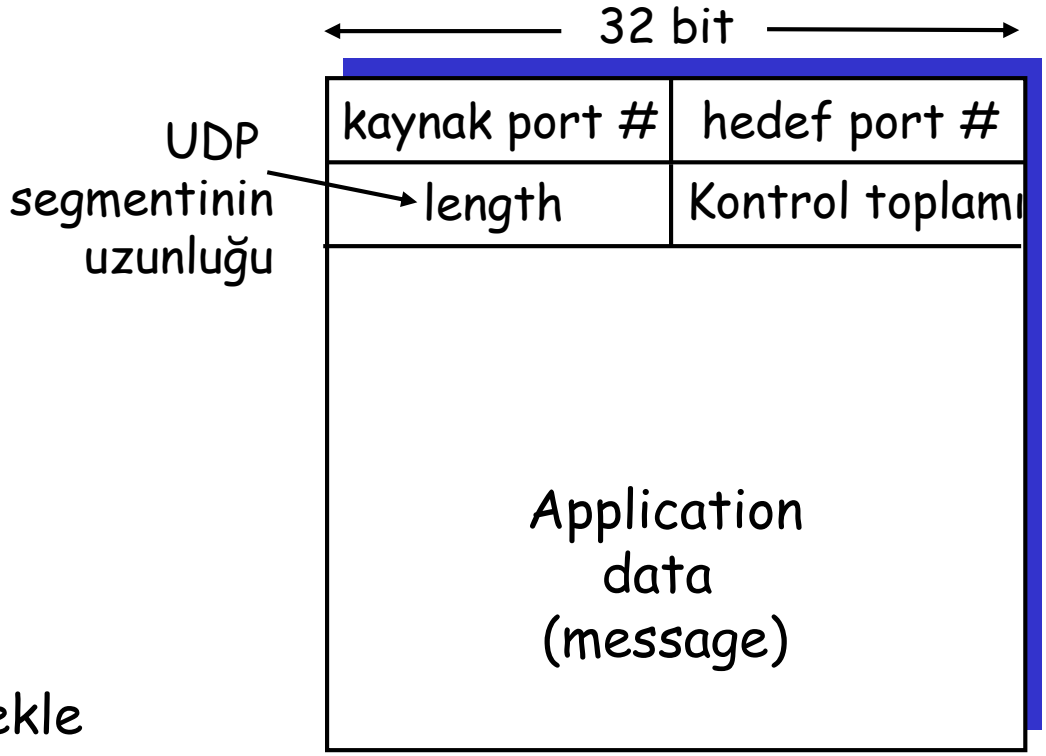
- "no frills," "bare bones" Internet taşıma protokolü
- "en iyi çaba" servisi, UDP segmentleri:
  - kaybolabilir
  - Sırasız olarak uygulamaya gelebilir
- *Bağlatısız:*
  - UDP gönderici ve alıcı arasında el sıkışma yok
  - her UDP segmenti diğerlerinden bağımsızdır

Neden UDP var?

- Bağlantı kurulumu(gecikme ekleyebilecek olan yok)
- basit: gönderici ve alıcı bağlantı durumu takibi yapmaz
- Küçük small segment başlığı
- Sıkışıklık kontrolü yok: UDP istendiği kadar hızlı gidebilir

UDP:

- sıklıkla streaming multimedya uygulamaları için kullanılır
  - Kayba çok duyarlı değil
  - Hıza duyarlı
- UDP kullananlar...
  - DNS
  - SNMP
- UDP üzerinden güvenilir transfer: uygulama katmanında güvenilirlik ekle
  - Uygulamaya özel hata tespbiti !



UDP segment formatı

UDP kontrol toplamı

Amaç: iletilen segmentte "hata" tesbiti

Gönderici:

- segment içeriklerine contents as sequence of 16-bitlik tamsayılar dizisi gibi davranır
- Kontrol toplamı: segment içeriğinin toplaması
- gönderici UDP kontrol toplamı alanına kontrol toplamı değeri ekler

Alıcı:

- Alınan segmentin kontrol toplamını hesaplar
- Hesaplanan kontrol toplamı değerinin kontrol toplamı değerine eşit olup olmadığına bakar:
  - Eşit değil – hata tesbit edildi
  - Eşit – hata tesbit edilemedi. *Fakat buna rağmen hala hata olabilir mi?* ….

Internet Kontrol Toplamı Örneği

- Not
  - Rakamları eklerken, en anlamlı bitten gelen eldenin sonuca eklenmesi gerekir
- Örnek: iki 16-bit'lik tamsayıyı ekle