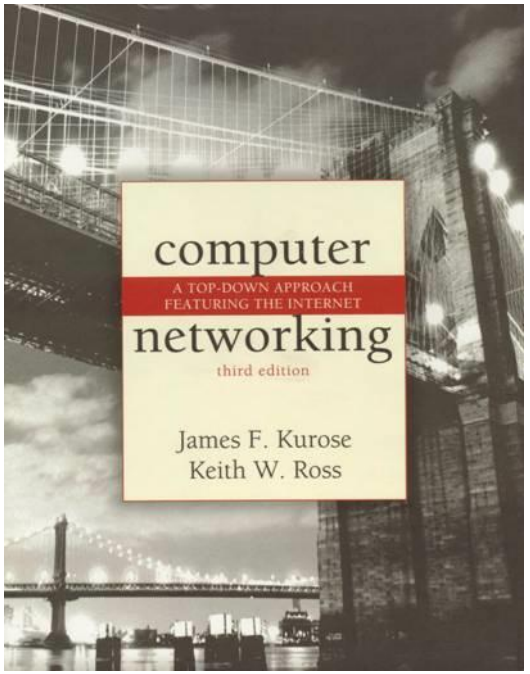
Uygulama Katmanı - 1



Computer Networking: A Top Down Approach Featuring the Internet, 3rd edition.
Jim Kurose, Keith Ross
Addison-Wesley, July 2004.

A note on the use of these ppt slides:

We're making these slides freely available to all (faculty, students, readers). They're in PowerPoint form so you can add, modify, and delete slides (including this one) and slide content to suit your needs. They obviously represent a *lot* of work on our part. In return for use, we only ask the following:

- If you use these slides (e.g., in a class) in substantially unaltered form, that you mention their source (after all, we'd like people to use our book!)
- If you post any slides in substantially unaltered form on a www site, that you note that they are adapted from (or perhaps identical to) our slides, and note our copyright of this material.

Thanks and enjoy! JFK/KWR

Uygulama Katmanı

- 2.1 Ağ uygulamalarının prensipleri
- 2.2 Web ve HTTP
- 2.3 FTP
- 2.4 Elektronik Posta
  - SMTP, POP3, IMAP
- 2.5 DNS
- 2.6 Eşler arası uygulamalar P2P

Uygulama Katmanı

Hedefler:

- Ağ uygulama protokollerinin kavramsal ve uygulama yönleri
  - Taşıma katmanı servis modelleri
  - İstemci-sunucu modeli
  - Eşler arası (P2P) modeli
- Protokolleri popüler uygulama katmanı protokollerini inceleyerek öğrenme
  - HTTP
  - FTP
  - SMTP / POP3 / IMAP
  - DNS
- Ağ uygulamalarını programlama
  - soket API

Bazı ağ uygulamaları

- E-mail
- Web
- Anında mesajlaşma (Instant messaging)
- Uzaktan erişim (Remote login)
- Eşler arası dosya paylaşımı (P2P file sharing)
- Çok kullanıcılı ağ oyunları (Multi-user network games)
- Streaming stored video clips
- Internet telefonu
- Gerçek zamanlı video konferans (Real-time video conference)
- Yoğun paralel işleme (Massive parallel computing)

Ağ uygulaması oluşturmak

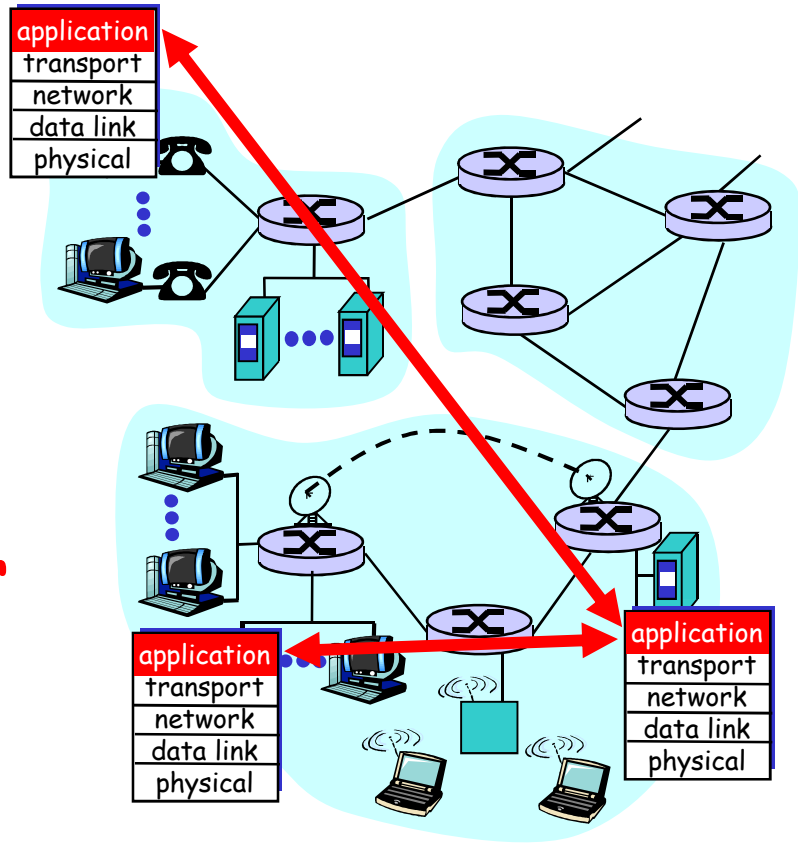
- Farklı uç sistemlerde çalışan
- Ağ üzerinden birbiriyle iletişime geçebilen

programlar yazılır.

- Örn. Web: web sunucusu yazılımları tarayıcı yazılımları ile iletişim kurarlar.

Ağ çekirdeğindeki araçlar için yazılım yazılmaz:

- Ağ çekirdeği araçları uygulama katmanında çalışmazlar
- Bu tasarım hızlı uygulama geliştirmeye izin verir.



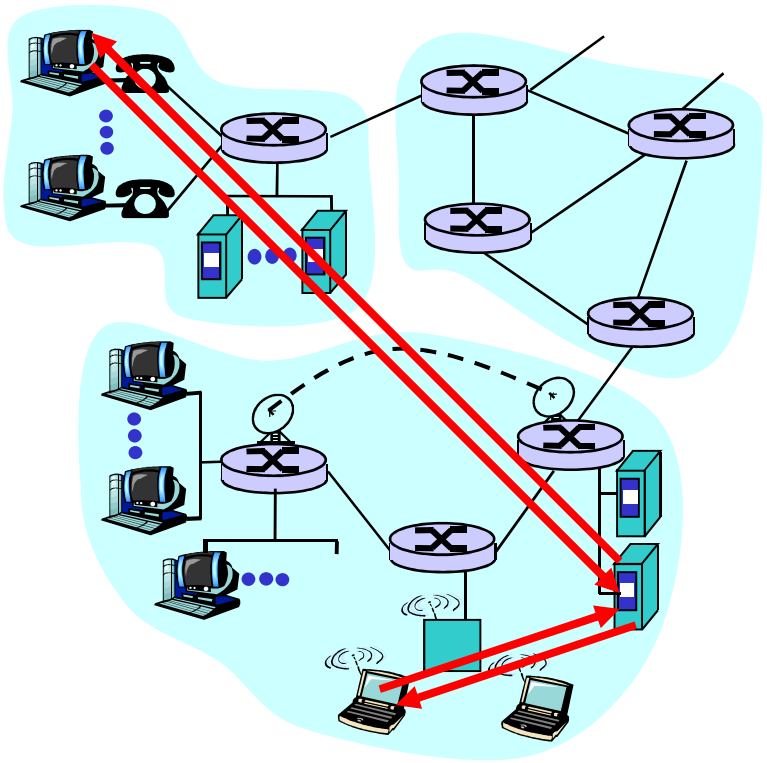
Uygulama Katmanı

- 2.1 Ağ uygulamalarının prensipleri
- 2.2 Web ve HTTP
- 2.3 FTP
- 2.4 Elektronik Posta
  - SMTP, POP3, IMAP
- 2.5 DNS
- 2.6 Eşler arası uygulamalar P2P
- 2.7 TCP ile soket programlama
- 2.8 UDP ile soket programlama
- 2.9 Web sunucusu oluşturma

Uygulama mimarileri

- İstemci sunucu (Client-server)
- Eşler arası (Peer-to-peer (P2P))
- İstemci sunucu ve P2P karışımı melez mimari (Hybrid of client-server and P2P)

İstemci sunucu mimarisi



sunucu:

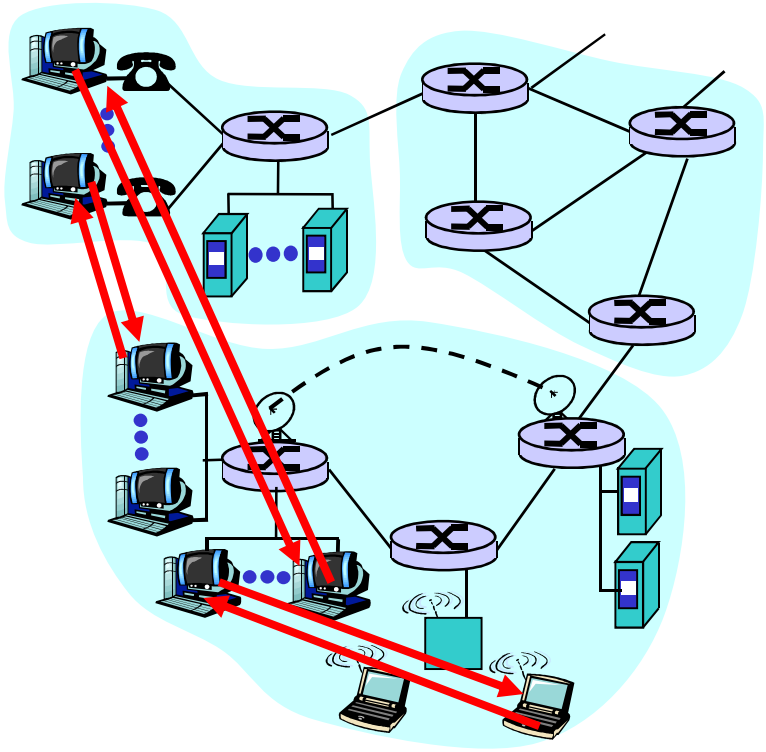
- Daima açık bir ana sistem
- Sabit IP adresli
- Ana sistem kümesi – sunucu çiftliği (server farms)

istemci:

- sunucu ile iletişim kurar
- may be her zaman açık olmak zorunda değildir
- Dinamik (değişebilir) IP adresine sahip olabilir
- Birbirleriyle direk olarak iletişime geçmezler

P2P mimarisi

- Her zaman açık olan sunuculara ihtiyaç yoktur
- Gelişigüzel uç sistemler birbirleriyle direk olarak iletişime geçerler
- Uç sistemler birbirleriye ara sıra bağlanırlar ve IP adreslerini değiştirirler
- örn: Gnutella



Büyük oranda ölçeklenebilirler (scalable)

Fakat yönetilmeleri zordur

İstemci-sunucu ve P2P mimarilerini birleştiren melez mimariler

Napster

- Dosya transferi P2P olarak gerçekleştirilir.
- Dosya arama merkezidir:
  - Peers register content at central server
  - Peers query same central server to locate content

Anında mesajlaşma

- İki kullanıcının mesajlaşması P2P dir
- Varlık tespiti/bulması (presence detection/location) merkezidir :
  - Kullanıcılar IP adreslerini çevrimiçi olunca merkezi sunucuya kayıt ederler.
  - Kullanıcılar diğerlerinin IP adreslerini bulmak için merkezi sunucu ile iletişime geçerler.

Süreç İletişimleri

Süreç: bir uç sistemde çalışan program.

- Aynı uç sistem üzerinde, iki süreç iç-süreç iletişimi (işletim sistemi tarafından tanımlanan) kullanarak birbirleriyle iletişime geçebilirler.
- Farklı sistemler üzerinde çalışan süreçler mesaj alış verişi ile haberleşirler

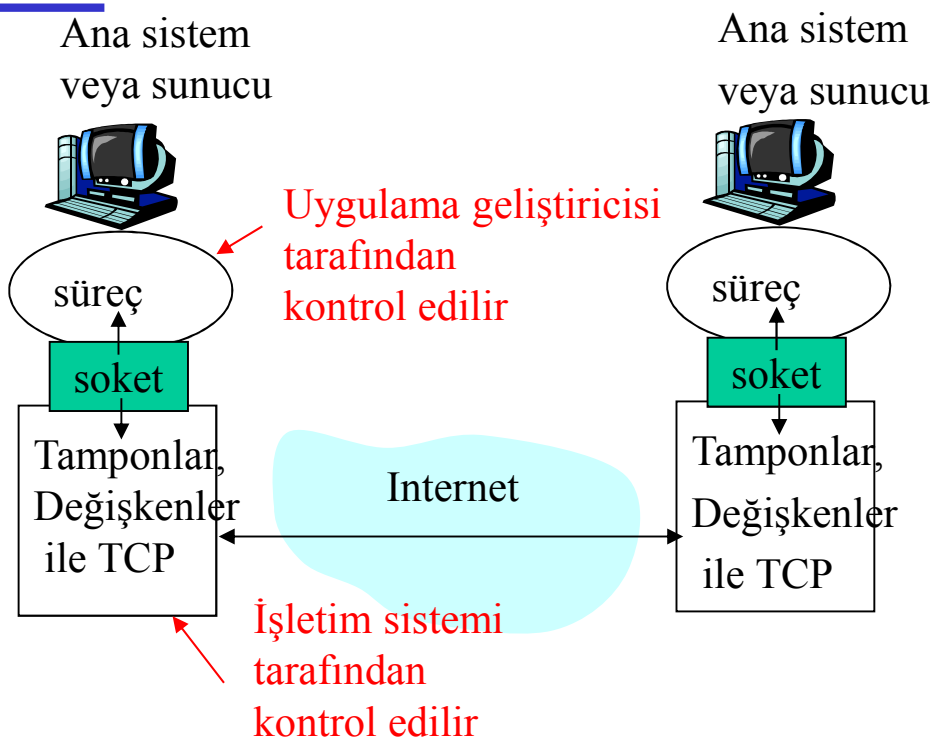
İstemci süreçleri:
iletişimi başlatan süreç

Sunucu süreçleri:
iletişime geçilmeyi bekleyen süreç

- Not: P2P mimarideki uygulamalarda hem istemci süreçleri hem de sunucu süreçleri yer alır.

Süreç ve Bilgisayar ağı arasındaki arabirimler: Soketler

- Bir süreç, mesajlara ağa soket adındaki yazılımlar aracılığıyla gönderir ve alır.
- Soket kapı benzeşmesi
  - Gönderici süreç mesajı kapıya (sokete) doğru iter
  - Gönderici süreç, kapının diğer tarafında mesajı hedef sürecin kapısına taşıyacak olan bir taşıma latyapısının olduğunu varsayar.



- Soketlere API (uygulama arabirimi) de denir
- Uygulama geliştiriciler soketin
  - uygulama tarafında her şey üzerinde kontrolleri vardır
  - Taşıma tarafında ise sadece (1) taşıma protokolünün seçimi; (2) bazı parametreleri ayarlayabilme(maksimum tampon ve minimum segment boyutları gibi) üzerinde kontroller olabilir

Süreçleri işaret etmek

- Bir sürecin mesaj alabilmesi için bir tanımlayıcısı (identifier) olması gerekir.
- Bir ana sistem 32 bit'lik benzersiz bir IP adresine sahiptir.
- Q: Süreci tanımlayabilmek için sürecin çalıştığı ana sistemin sadece IP adresini bilmesi yeterli midir?
- Cevap: Hayır, aynı ana sistem üzerinde pek çok süreç çalışıyor olabilir
- Tanımlayıcı hem IP adresini hem de süreçle ilgili hedef bağlantı noktası numarasına (port numbers) sahiptir
- Örnek port numaraları
  - HTTP sunucusu: port 80
  - Posta sunucusu: port 25

Uygulama katmanı protokolleri neleri tanımlar?

- Karşılıklı iletilen mesajlar, örn. istek&cevap mesajları
- Mesajların sentaksı: mesajlardaki alanlar ve bu alanların nasıl tanımlandığı
- Alanların anlamları, alanlardaki bilgilerin anlamları
- Süreçlerin ne zaman mesajları alıp göndereceklerine dair kurallar.

Genel etki alanı protokolleri

- RFC lerde tanımlanmıştır
- Birlikte işlerliğe (interoperability) olanak sağlar
- Örn., HTTP, SMTP

Özel protokoller:

- Örn., KaZaA

Uygulamalar tarafından ihtiyaç duyulan taşıma servisleri

Veri kaybı (data loss)

- Bazı uygulamalar bir miktar veri kaybını tolere edebilir (örn., ses/video)
- Diğerleri 100% güvenli veri transferine ihtiyaç duyarlar (örn., dosya transferi, telnet)

Zamanlama (timing)

- Bazı uygulamalar "etkili" olabilmek için az gecikme gerektirirler (örn., Internet telefonu, etkileşimli oyunlar)

Bant genişliği (bandwidth)

- Bazı uyglamalar "etkili" olabilmek için belli bir miktarda ant genişliğine ihtiyaç duyarlar (örn., multimedia)
- Diğerleri bant genişliğini ne kadarını kullanabilirlerse onunla idare edebilirler ("elastik uygulamalar)

Ağ uygulamalarının taşıma katmanı gereklilikleri

Uygulama	Veri kaybı	Bant genişliği	Zaman Duyarlılık
Dosya transferi	Kayıp yok	esnek	hayır
e-posta	Kayıp yok	esnek	hayır
Web belgeleri	Kayıp yok	esnek	hayır
Gerçek zamanlı ses/görüntü	Kayıp toleranslı	ses: 5kbps-1Mbps video:10kbps-5Mbps	evet, 100's msn
Depolanmış ses/video	Kayıp toleranslı	Yukarıdakiyle aynı	evet, birkaç sn
Etkileşimli oyunlar	Kayıp toleranslı	Birkaç kbps ve üstü	evet, 100's msn
Anında mesajlaşma	Kayıp yok	esnek	Evet ve hayır

Internet tarafından sunulan Taşıma Servisleri

TCP servisleri:

- *Bağlantı-yönelimli (connection-oriented):* istemci ve sunucu süreçler arasında bağlantı kurulumu gerekir
- Gönderici ve alıcı süreçler arasında *güvenilir taşıma (reliable transport)*
- *Akış kontrolü (flow control):* gönderici alıcıyı sıkıştırmaz
- *Tıkanıklık kontrolü (congestion control): ağ çok yüklendiğinde göndericinin gönderdiği veri miktarını kısar*
- *Zamanlama, minimum bant genişliği garantisi sağlamaz.*

UDP servisleri:

- Gönderici ve alıcı süreçler arasında güvenilir olmayan veri iletimi
- Bağlantı kurulumu (connection setup), güvenilirlik (reliability), akış kontrolü (flow control), tıkanıklık kontrolü (congestion control), zamanlama (timing), veya bant genişliği garantisi (bandwidth guarantee) SAĞLAMAZ

Q: O zaman bununla neden ilgileniyoruz? Niye UDP var?

Internet uygulamaları: uygulama ve altında çalışan taşıma protokolleri

Uygulama	Uygulama katmanı protokolü	Altta yatan taşıma protokolü
e-posta	SMTP [RFC 2821]	TCP
Uzak terminal erişimi	Telnet [RFC 854]	TCP
Web	HTTP [RFC 2616]	TCP
Dosya transferi	FTP [RFC 959]	TCP
akan multimedia	HTTP (örneğin, YouTube), RTP	TCP veya UDP
Internet telefonculuğu	SIP, RTP, veya Markaya özgü (Skype)	tipik olarak UDP

Uygulama Katmanı

- 2.1 Ağ uygulamalarının prensipleri
- 2.2 Web ve HTTP
- 2.3 FTP
- 2.4 Elektronik Posta
  - SMTP, POP3, IMAP
- 2.5 DNS
- 2.6 Eşler arası uygulamalar P2P

Web ve HTTP

Bir miktar web terminolojisi

- Bir web sayfası, nesnelerden oluşur
- Bir nesne bir HTML dosyası, bir JPEG görüntüsü, bir Java appleti, veya bir video klip olabilir.
- Çoğu web sayfası birçok başvuru nesnesini içeren temel bir HTML dosyasından meydana gelir
- Her nesne sadece bir tek URL ile adreslenebilen bir dosyadır
- Example URL:

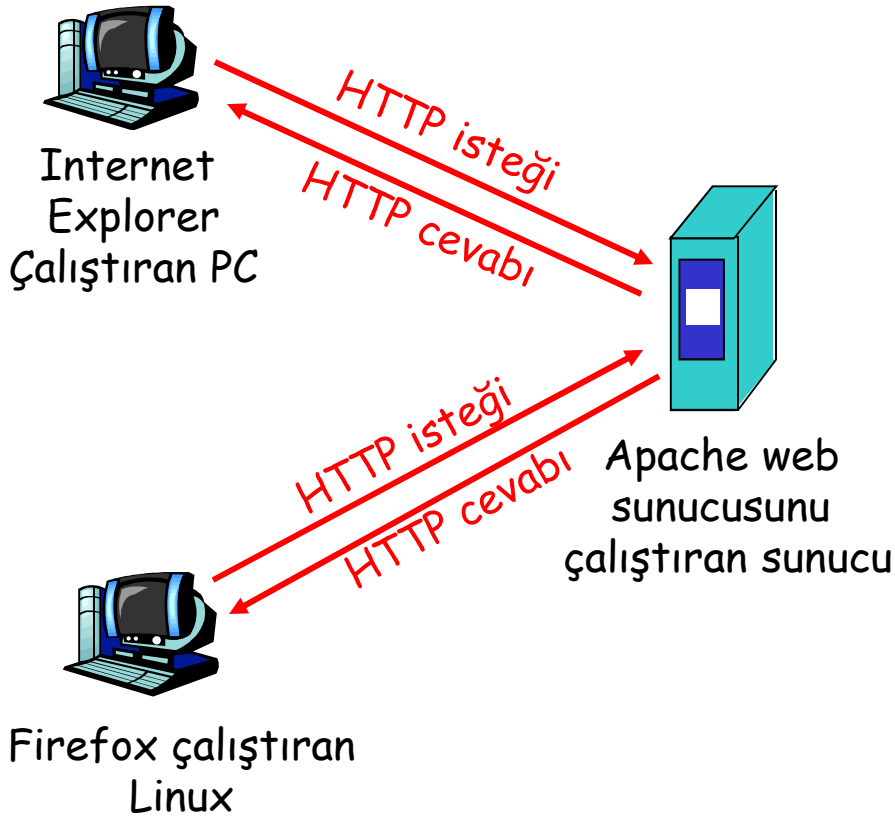
`www.someschool.edu/someDept/pic.gif`

ana sistem adı — nesne yolunun adı

HTTP'ye genel bakış

HTTP: hypertext transfer protocol

- Web'in uygulama katmanı protokolüdür.
- İstemci/sunucu modeli
  - *istemci:* Web nesnelerini isteyen, alan ve gösteren tarayıcı
  - *sunucu:* isteklere karşılık olarak nesneleri gönderen web sunucusu
- HTTP 1.0: RFC 1945
- HTTP 1.1: RFC 2068



HTTP'ye genel bakış (devam)

TCP kullanır:

- İstemci sunucuyla port 80'den TCP bağlantısı (soket oluşturur) başlatır
- Sunucu istemcinin TCP bağlantı isteğini kabul eder.
- HTTP mesajları (uygulama katmanı protokolü mesajları) tarayıcı (HTTP istemcisi) ve web sunucusu (HTTP sunucusu) arasında karşılıklı gönderilir
- TCP bağlantısı kapatılır.

HTTP "durumsuz"dur

- Sunucu istemcinin önceki istekleri ile ilgili bilgi tutmaz.

diğer tarafdan

"durum" bilgisi tutan protokoller karmaşıktır!

- Geçmiş durum bilgisi tutulmalıdır
- Eğer sunucu/istemci çökerse, birbirlerindeki "durum" bilgisi tutarsız hale gelebilir ve yeniden oluşturulması gerekir.

HTTP bağlantıları

Kalıcı olmayan (nonpersistent) HTTP

- Genellikle bir TCP bağlantısı ile bir nesne gönderilir.
- HTTP/1.0 kalıcı olmayan (nonpersistent) HTTP'yi kullanır

Kalıcı (persistent) HTTP

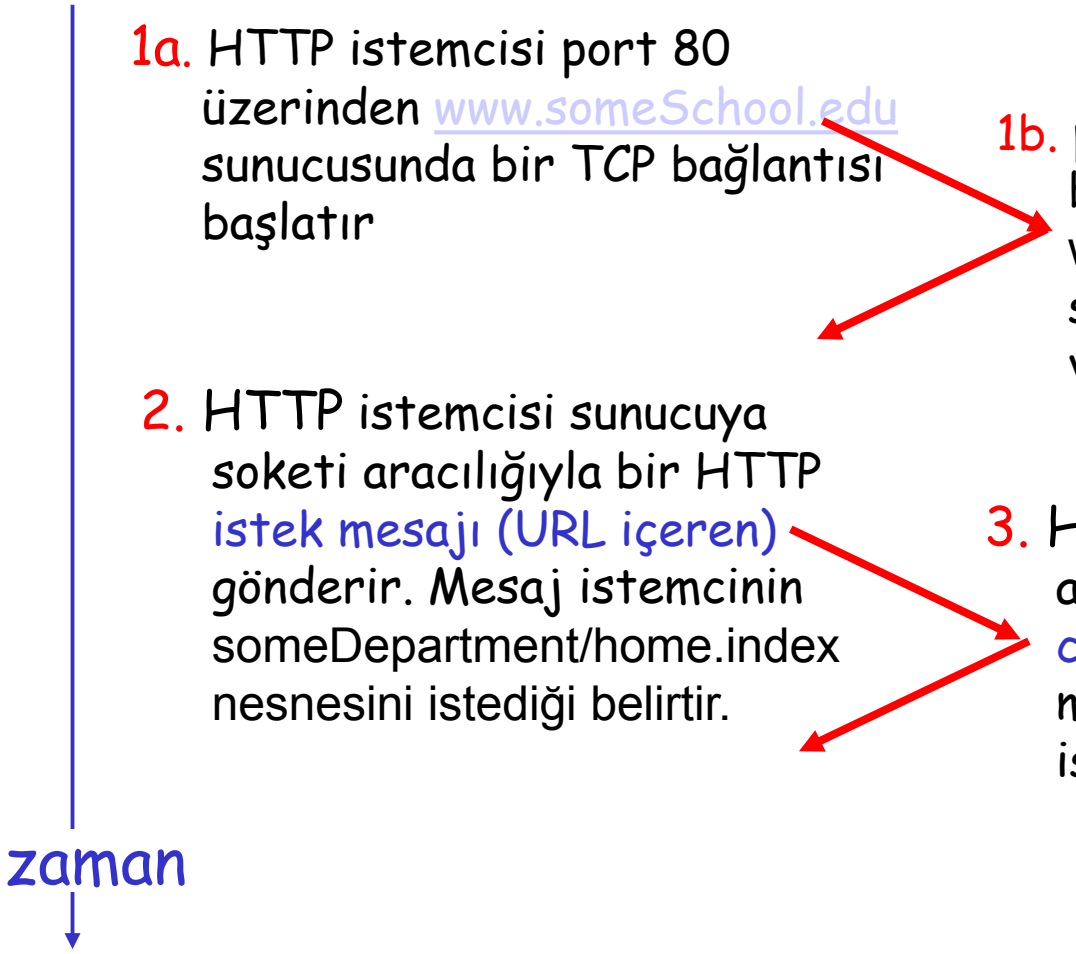
- İstemci ile sunucu arasında tek bir TCP bağlantısı ile birden fazla nesne gönderilebilir.
- HTTP/1.1 varsayılan modunda kalıcı (persistent) bağlantı kullanır

Kalıcı olmayan (nonpersistent) HTTP

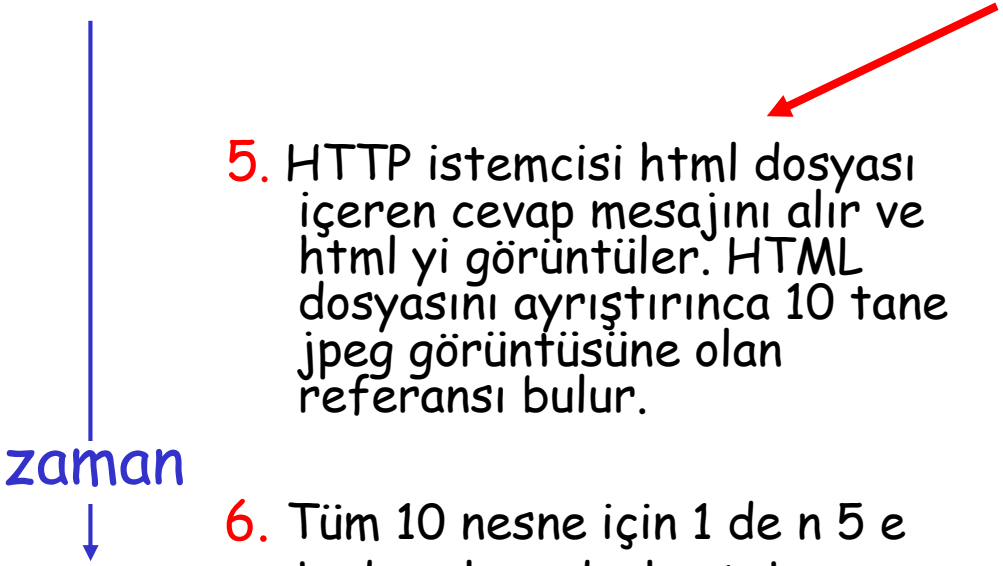
Kullanıcın girdiği URL:

`www.someSchool.edu/someDepartment/home.index`

(metin ve 10 tane jpeg görüntüsüne referans içermekte)



Kalıcı olmayan (nonpersistent) HTTP



4. HTTP sunucusu TCP bağlantısını kapatır

5. HTTP istemcisi html dosyası içeren cevap mesajını alır ve html yi görüntüler. HTML dosyasını ayrıştırınca 10 tane jpeg görüntüsüne olan referansı bulur.

6. Tüm 10 nesne için 1 de n 5 e kadar olan adımlar tekrar edilir.

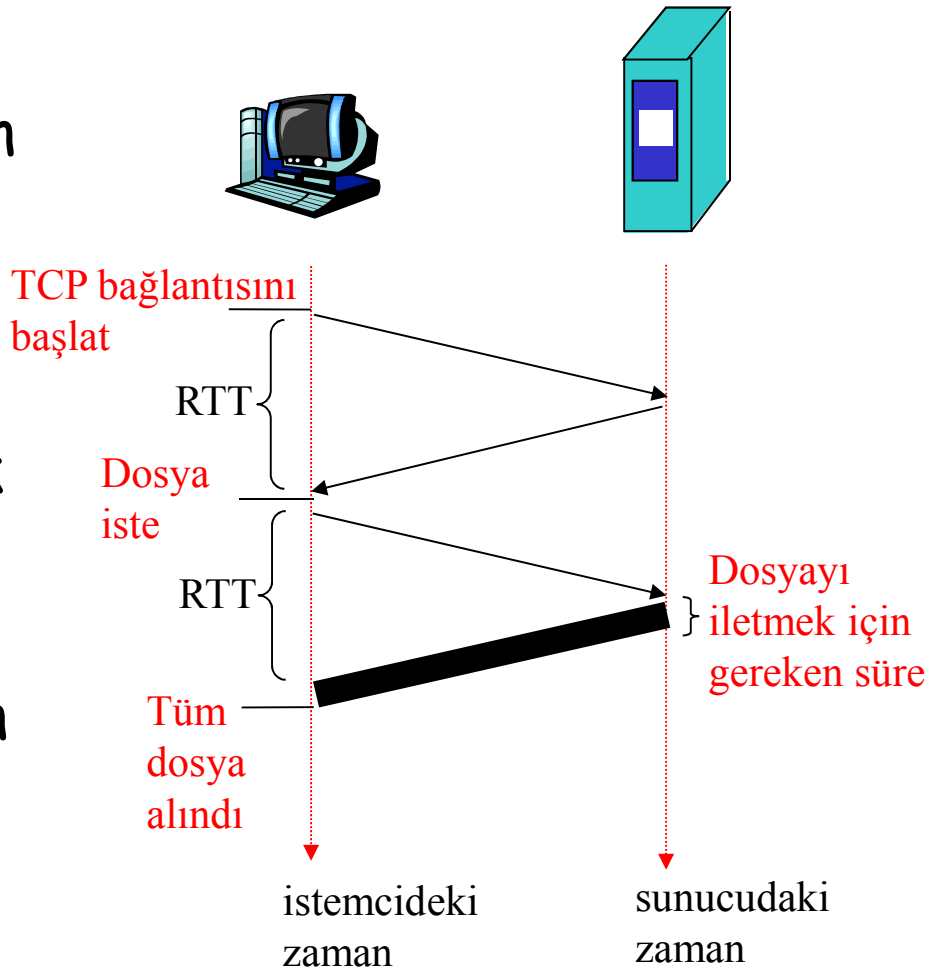
Cevap zamanı (response time)

RTT'nin tanımı: İstemciden sunucuya giden bir paketin gidiş dönüş zamanı

Cevap zamanı (response time):

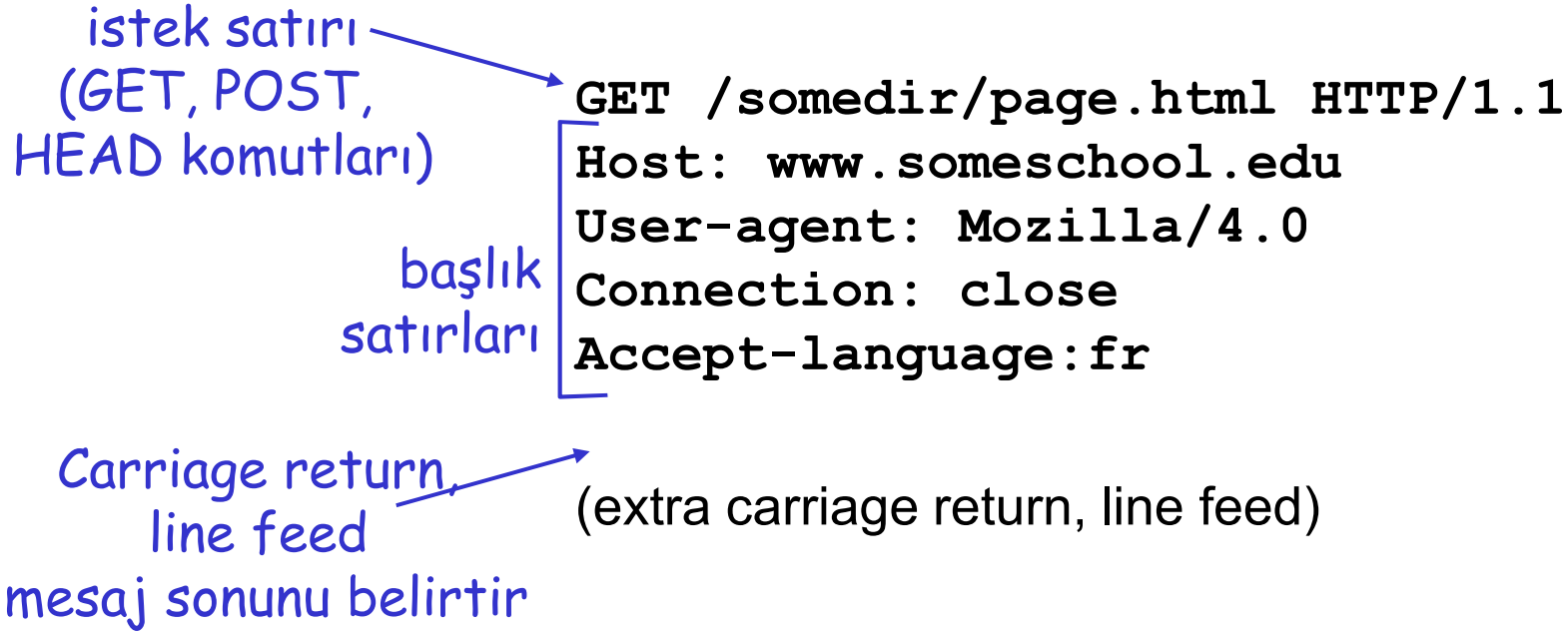
- TCP bağlantısnı başlatmak için bir RTT
- HTTP isteği ve HTTP cevabının ilk birkaç bitinin dönüşü için bir RTT
- Dosya iletim zamanı

toplam= 2RTT+iletim zamanı

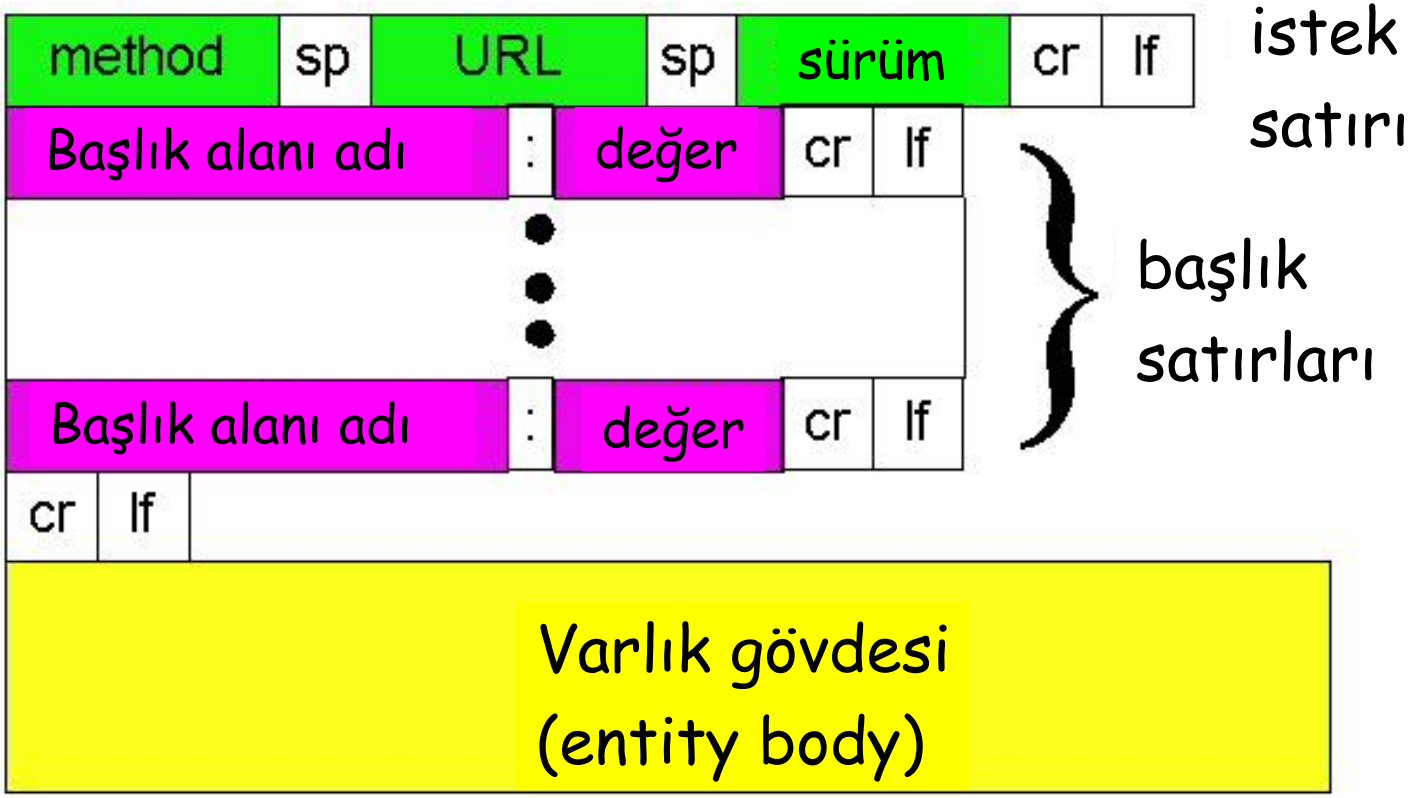


HTTP istek mesajı

- İki çeşit http mesajı: *istek, cevap*
- HTTP istek mesajı:
  - ASCII (insanlar tarafında da anlaşılabilir)



HTTP istek mesajı: genel biçim



Formlara girilen bilgilerin gönderilmesi

Post metodu:

- Web sayfalarında sıklıkla veri girişi yapılan formlar yer alır.
- Veri girişleri sunucuya varlık gövdesi (entity body) içerisinde gönderilir

URL metodu:

- GET metodunu kullanır
- Girilen veriler istek satırının URL alanı ile gönderilir:

```
www.somesite.com/animalsearch?monkeys&banana
```

Metod tipleri

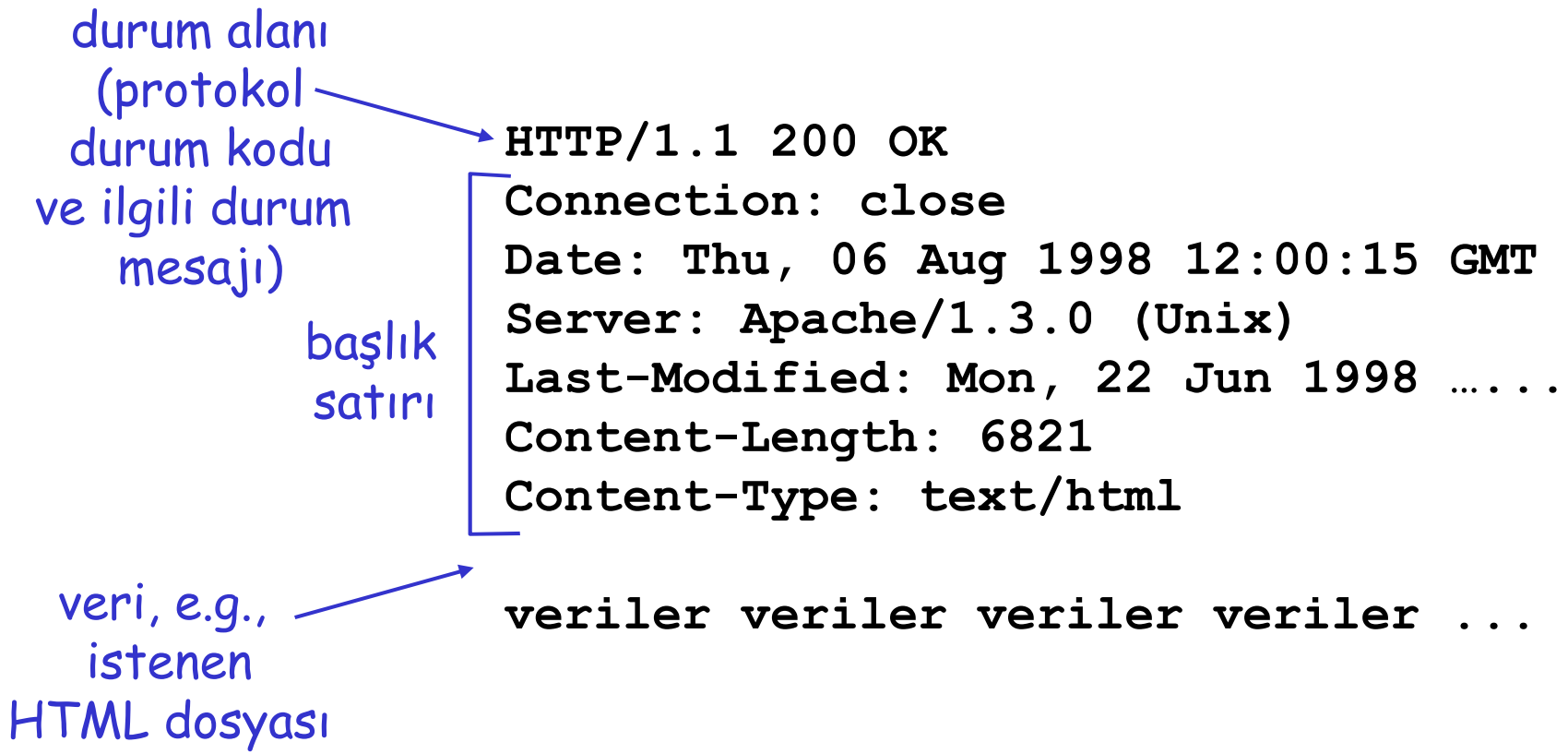
HTTP/1.0

- GET
- POST
- HEAD
  - Sunucudan istenen nesneyi dışarıda bırakması istenir

HTTP/1.1

- GET, POST, HEAD
- PUT
  - URL alanında belirlenen spesifik bir yola bir nesne yüklemeyi sağlar
- DELETE
  - URL alanında gönderilen nesnenin sunucudan silinmesini sağlar

HTTP cevap mesajı



HTTP durum kodları

Sunucu → istemci cevap mesajının ilk satırında yer alır

Birkaç örnek kod:

200 OK

- İstek başarılı olmuştur ve bilgi cevap ile döndürülmüştür

301 Moved Permanently

- İstenen nesne kalıcı olarak taşınmıştır; yeni URL, cevap mesajının Location: başlığında belirtilmiştir.

400 Bad Request

- İstek sunucu tarafından anlaşımlamıştır

404 Not Found

- İstenilen belge bu sunucu üzerinde mevcut değildir.

505 HTTP Version Not Supported

- İstenilen HTTP protokolü sunucu tarafından desteklenmemektedir.

Gerçek bir HTTP cevap mesajı görmek

1. Favori web sunucunuza bir Telnet oturumu açalım:

```
telnet cis.poly.edu 80
```

Port 80 (varsayılan HTTP sunucu portu) üzerinden cis.poly.edu da TCP bağlantısı açar. Yazılan herşey port 80 üzerinden cis.poly.edu ya gönderilir

2. Aşağıdaki GET HTTP isteğini yazalım:

```
GET /~ross/ HTTP/1.1
Host: cis.poly.edu
```

Bunu yazarak HTTP sunucusuna bu minimum GET isteğini göndermiş oluyoruz

3. HTTP sunucusu tarafından gönderilen cevap mesajını inceleyelim!

Kullanıcı taraflı etkileşim: Çerezler (cookies)

Pek çok web sitesi cookie kullanır

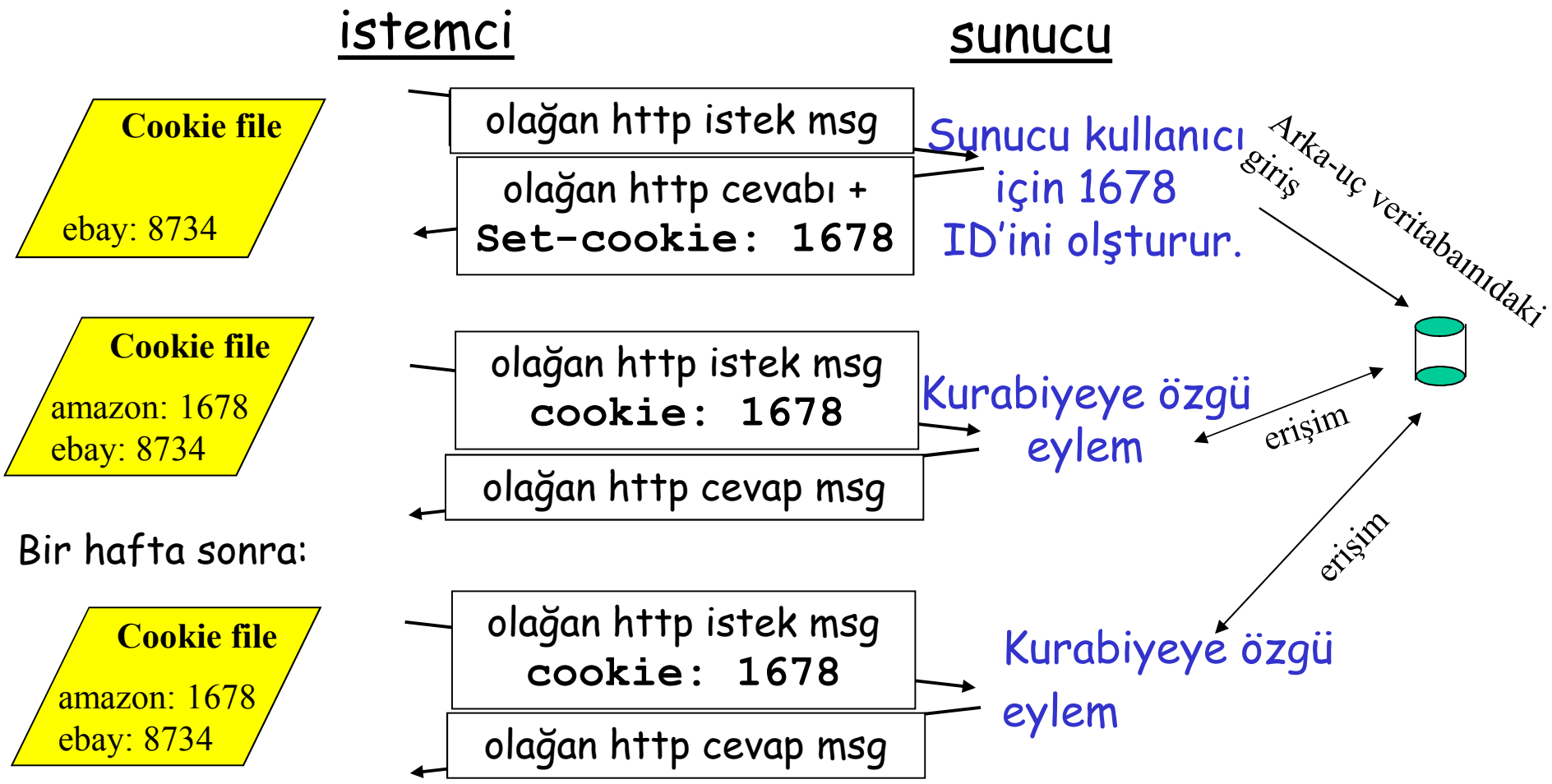
Dört bileşen:

1) HTTP cevap mesajındaki bir cookie başlık satırı
2) HTTP istek mesajındaki bir cookie başlık satırı
3) Kullanıcın uç sisteminde tutulan ve tarayıcı tarafından yönetilen bir cookie dosyası
4) Web sitesindeki bir arka uç veritabanı

Example:

- Suzan Internet'e her zaman aynı PC den ulaşmaktadır
- Diyelim ki bir e-ticaret sitesini ilk defa ziyaret ediyor
- İstek siteye ilk defa ulaştığında, site benzersiz bir tanımlama numarası oluşturur ve arka-uç veritabanında sunucu tanımlama numarası ile indekslenen bir giriş oluşturur.

Çerez (cookie) ile kullanıcı durumunu korumak



Cookieler (devam)

Cookie lerin sağladıkları:

- Yetkilendirme (authorization)
- Alışveriş sepeti(shopping carts)
- Tavsiyeler
- Kullanıcı oturum durumu (user session state) (Web e-mail)

Bunun yanında

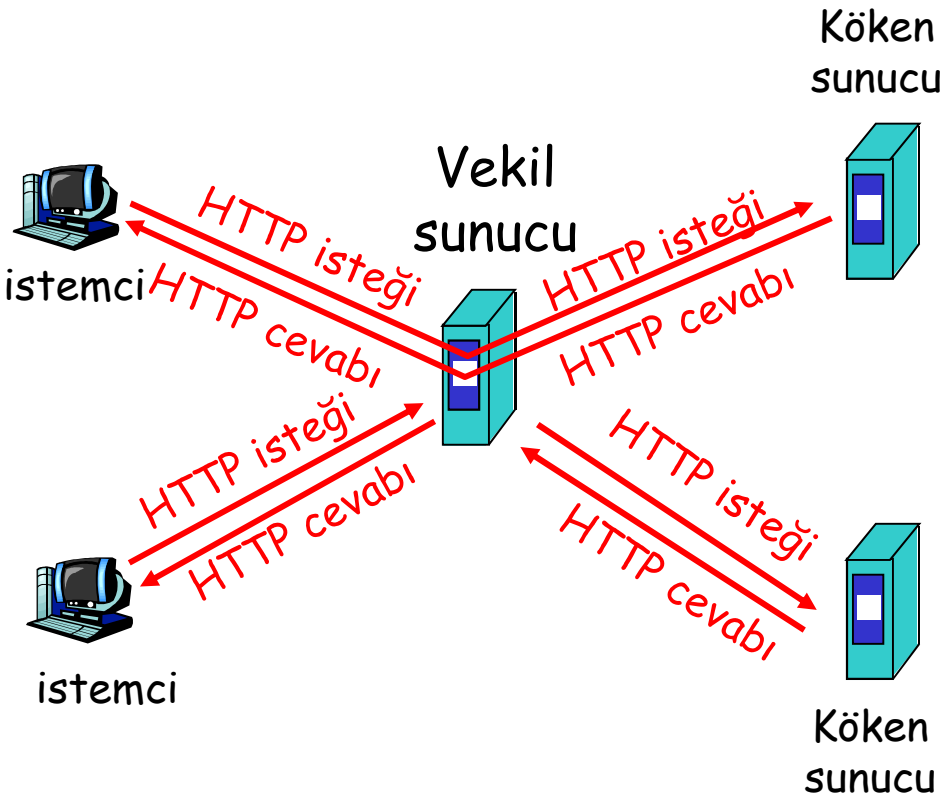
Cookie ler ve gizlilik:

- Cookie ler sitelerin sizin hakkınızda pek çok şey öğrenmesine izin verir.
- Sitelere e-mail ve isim bilgilerinizi verebilirsiniz.
- Arama motorları daha fazlasını öğrenmek redirection & cookie leri kullanırlar.
- Reklam şirketleri sitelerden bilgileri alırlar

Web tampon belleği (caches) (vekil sunucusu -proxy server)

Amaç: HTTP isteklerini köken Web sunucusu yerine karşılayan bir ağ varlığıdır.

- Kullanıcı tarayıcısını, kullanıcın tüm HTTP isteklerini önce bu Web tampon belleğine yönlendireceği şekilde yapılandırabilir
- Tarayıcı tüm HTTP isteklerini bu tampon belleğe yönlendirir.
  - Nesne tampon bellekte ise: tampon bellek nesneyi istemciye döndürür.
  - Nesne tampon bellekte değilse: tampon bellek köken sunucudan nesneyi ister ve sonrasında nesneyi istemciye döndürür.



Web tampon belleği ile ilgili daha fazlası

- Tampon bellek aynı anda hem sunucu hem istemci gibi davranır
- Genel olarak tampon bellekler bir ISP tarafından kurulurlar (üniversite, şirket, yerleşim yeri ISP'si)

Neden web tampon belleği?

- Bir istemci isteği için cevap zamanını azaltır
- Kurumun erişim hattındaki trafiği azaltır.
- Tampon belleklerin yoğun olduğu bir Internet "zayıf" içerik sağlayıcıların içeriği daha iyi sunmalarını sağlar (fakat aynısını P2P dosya paylaşımı da yapabilir)

Tampon bellek örneği

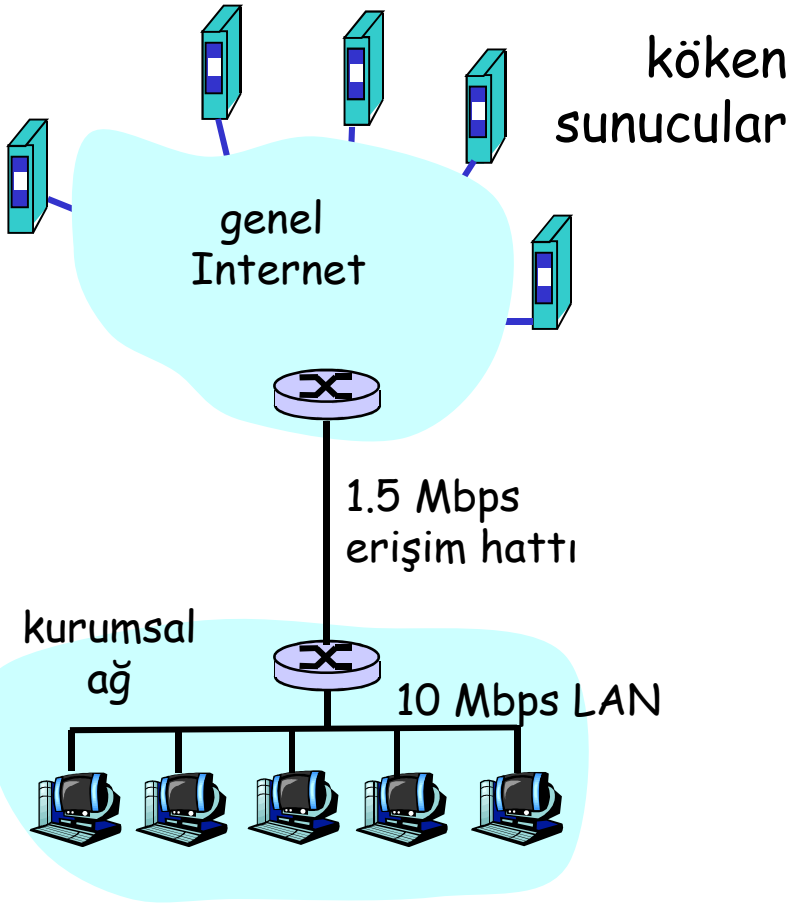
Varsayımlar

- Ortalama nesne büyüklüğü = 100,000 bits
- Kurumun tarayıcılarından köken sunuculara gelen ort. istek oranı = 15/sec
- Kurumsal yönlendiriciden köken sunuculara ya da geri yönlendiricilere olan gecikme = 2 sec

Sonuçlar

- LAN'daki ağ yükü/trafik yoğunluğu= 0.15
- Erişim hattındaki ağ yükü/trafik yoğunluğu = 1
- Toplam gecikme= Internet gecikmesi + erişim gecikmesi + LAN gecikmesi

  = 2 sn + dakikalar + millisaniyeler



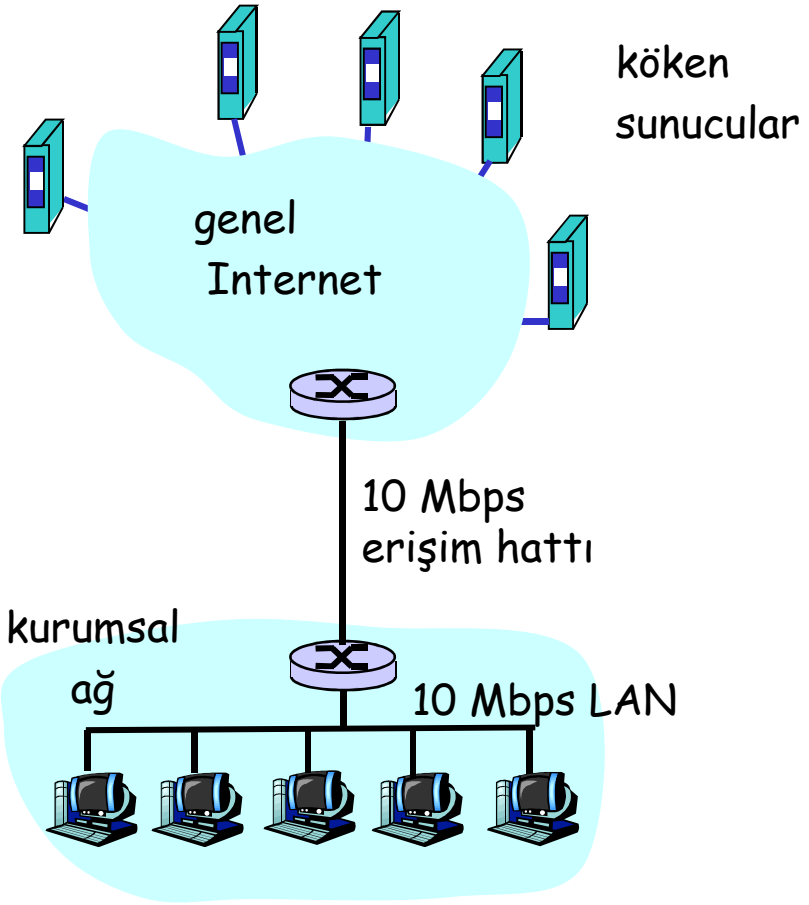
Tampon bellek örneği (devamı)

Olası çözüm

- Erişim hattının bant genişliğini diyelim ki 10 Mbps'ye arttırırsak

Sonuçları

- LAN'daki yarar = 15%
- Erişim hattındaki yarar = 15%
- Toplam gecikme= Internet gecikmesi + erişim gecikmesi + LAN gecikmesi
  = 2 sn + msn + msn
- Maliyetli bir artırım



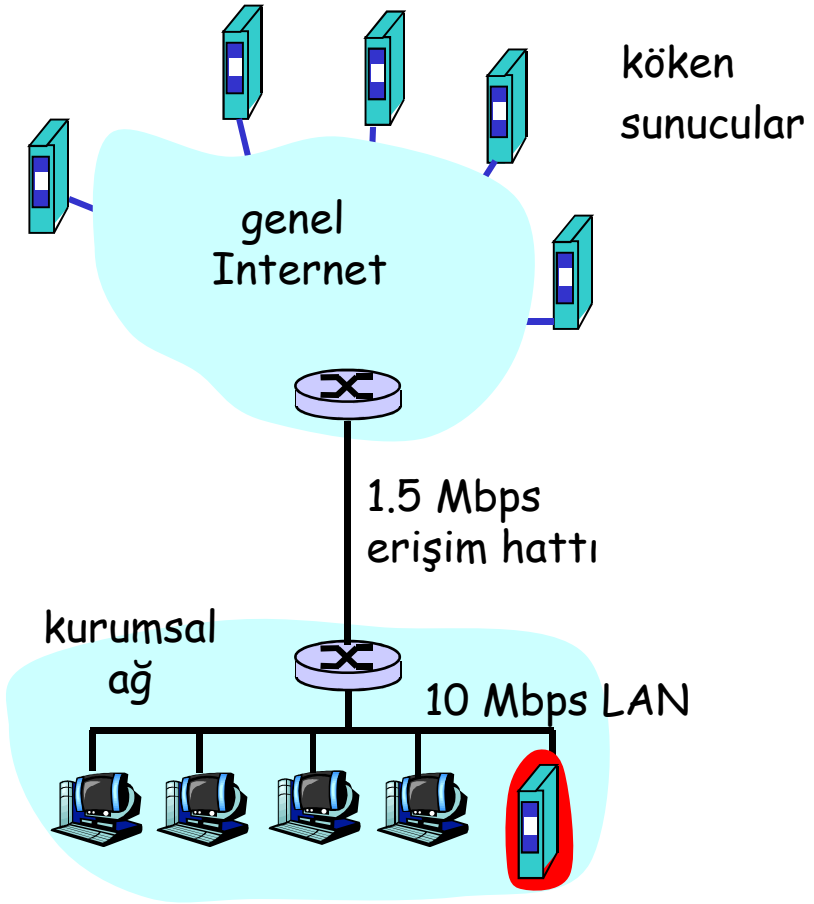
Tampon bellek örneği (devamı)

Tampon bellek kurarsak

- Hit alma oranı .4 varsayılırsa

Sonuçları

- İsteklerin 40% 'ı neredeyse hemen karşılanır
- İsteklerin 60% 'ı köken sunucular tarafından karşılanır
- Erişim hattının kullanılması 60 % 'a azaltıldı, önemsiz gecikmeler olabilir (10 msn kadar)
- Toplam ort. gecikme = Internet gecikmesi + erişim gecikmesi + LAN gecikmesi



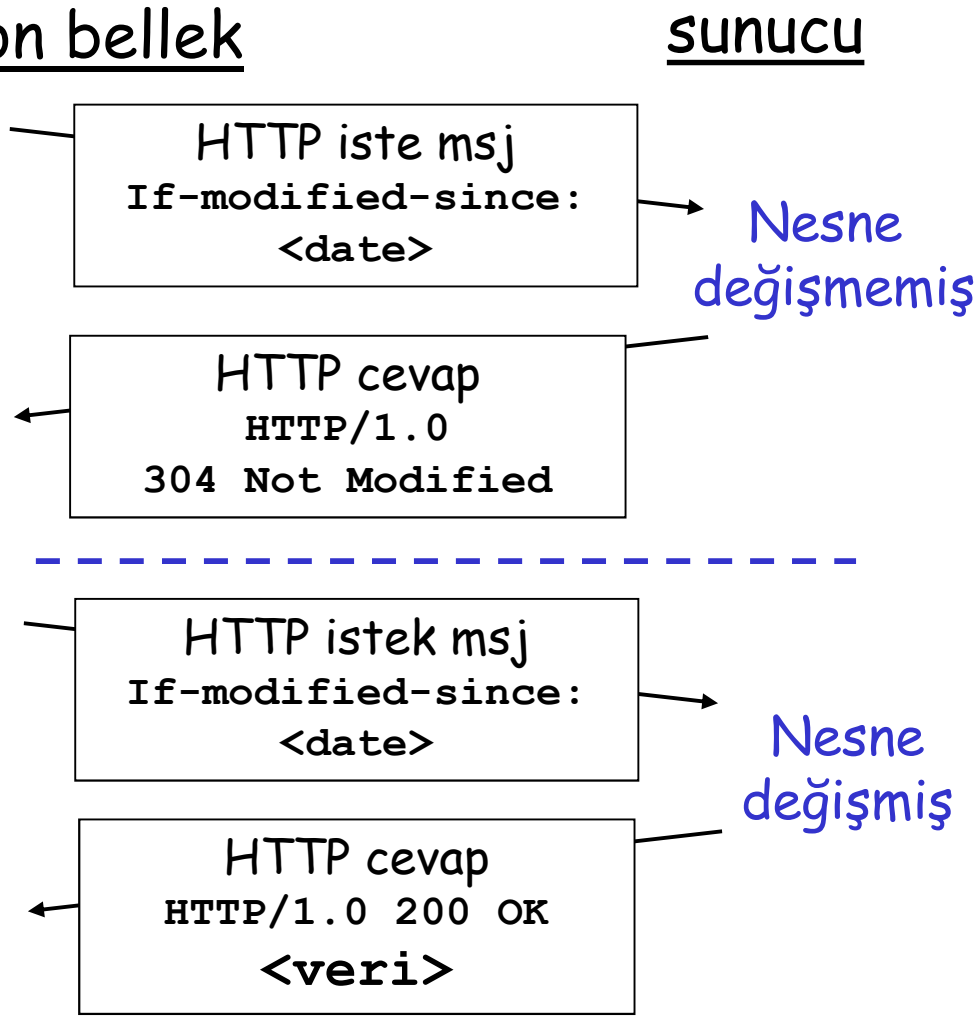
Koşullu GET

- Amaç: eğer tampon bellek güncel sürüme sahipse nesneyi gönderme
- Tampon bellek: HTTP isteği içerisinde belleğe alınmış kopyanın tarihini belirler

```
If-modified-since:
  <date>
```

- sunucu: eğer tampondaki kopya güncelse cevapta nesne bulunmaz:

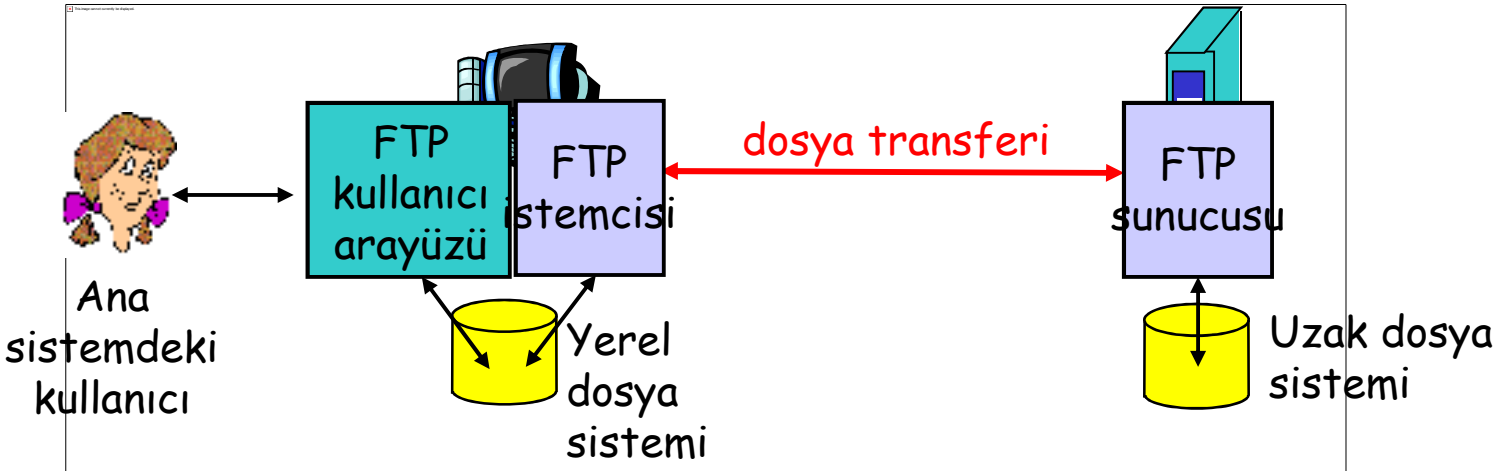
```
HTTP/1.0 304 Not
  Modified
```



Uygulama Katmanı

- 2.1 Ağ uygulamalarının prensipleri
- 2.2 Web ve HTTP
- 2.3 FTP
- 2.4 Elektronik Posta
  - SMTP, POP3, IMAP
- 2.5 DNS
- 2.6 Eşler arası uygulamalar P2P

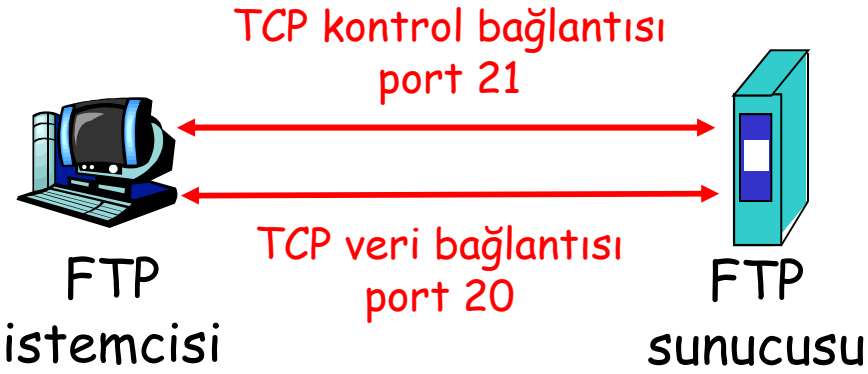
Dosya transfer protokolü :FTP (file transfer protocol)



- Uzak ana sistem e/den dosya transferi
- İstemci sunucu modeli
  - *istemci:* transferi başlatan taraf
  - *sunucu:* uzaktaki ana sistem
- ftp: RFC 959
- ftp sunucus: port 21

FTP: ayrı kontrol ve veri bağlantıları

- FTP istemcisi FTP sunucusu ile TCP'yi taşıma protokolü olarak kullanarak port 21 üzerinden iletişim kurar
- İstemci kontrol bağlantısı üzerinden yetkilendirme alır.
- İstemci uzaktaki dizinleri kontrol bağlantısı üzerinden gönderdiği komutlar aracılığıyla tarar.
- Sunucu dosya transferi ile ilgili bir komut aldığında, sunucu istemciye TCP veri bağlantısı açar.
- Dosyayı transfer ettikten sonra sunucu bağlantıyı kapatır.



- Sunucu başka bir dosya transfer edilmek istenirse yeni bir TCP veri bağlantısı açar.
- Kontrol bağlantısı: "bant dışı (out of band)"
- FTP sunucusu "durum" korumalıdır: halihazırdaki dizin, önceki yetkilendirmeler

FTP komutları, cevapları

Örnek komutlar:

- 7 bit ASCII biçiminde kontrol bağlantısı boyunca gönderilir.
- **USER *kullanıcıadı***
- **PASS *parola***
- **LIST** sunucuda halihazırda bulunan dizindeki dosyaları listeler
- **RETR dosyaadı** dosya almak için kullanılır
- **STOR dosyaadi** uzaktaki sisteme dosya depolamak için kullanılır

Örnek durum kodları

- Durum kodu ve deyimi (HTTP'deki gibi)
- **331 Username OK, password required**
- **125 data connection already open; transfer starting**
- **425 Can't open data connection**
- **452 Error writing file**

Uygulama Katmanı

- 2.1 Ağ uygulamalarının prensipleri
- 2.2 Web ve HTTP
- 2.3 FTP
- 2.4 Elektronik Posta
  - SMTP, POP3, IMAP
- 2.5 DNS
- 2.6 Eşler arası uygulamalar P2P
- 2.7 TCP ile soket programlama
- 2.8 UDP ile soket programlama

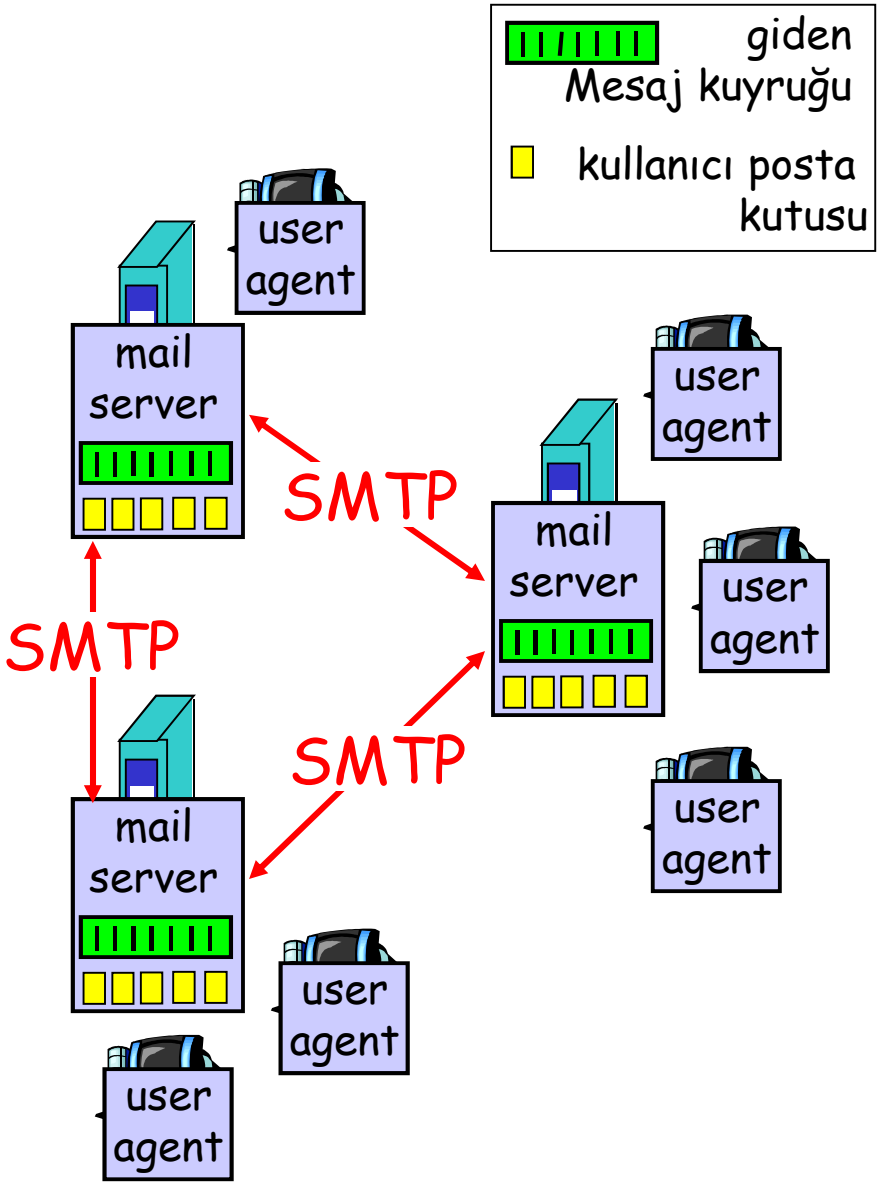
Elektronik Posta

Üç ana bileşen

- Kullanıcı temsilcileri (user agents)
- Posta sunucuları (mail servers)
- Basit posta transfer protokolü (simple mail transfer protocol: SMTP)

User Agent

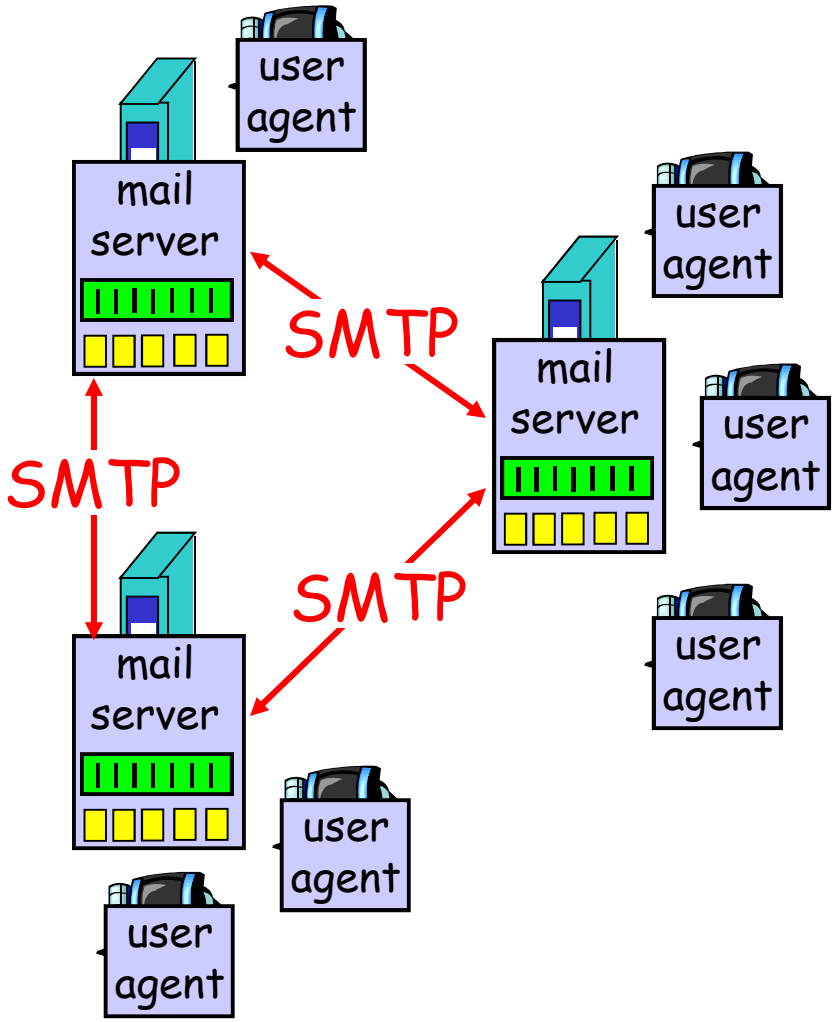
- a.k.a. "posta okuyucuları"
- Mesaj okuma, cevap verme, iletme, kaydetme ve oluşturma
- e.g., Eudora, Outlook, elm, Netscape Messenger
- Sunucuda depolanan gelen ve giden mesajlar



Elektronik posta: posta sunucuları

Posta Sunucuları

- Posta kutusu kullanıcıya gelen mesajları içerir.
- Mesaj kuyruğu giden (gönderilecek) mesajları içerir
- SMTP protokolü posta sunucuları arasında posta göndermek için kullanılır
  - istemci: gönderen posta sunucusu
  - "sunucu": alıcı posta sunucusu

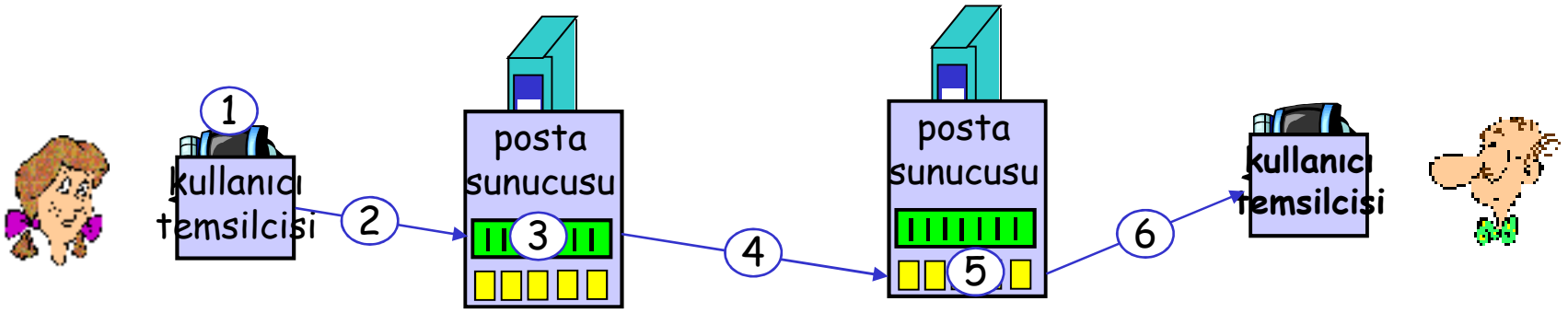


Elektronik Posta: SMTP [RFC 2821]

- İstemciden sunucuya eposta mesajlarını güvenilir olarak transfer etmek için port 25 üzerinden TCP kullanır
- Direk transfer: gönderen sunucudan alıcı sunucuya
- Üç fazlı transfer
  - El sıkışma (greeting)
  - Mesajın iletimi
  - Kapatma (closure)
- Komut /cevap etkileşimi (command/response interaction)
  - komutlar: ASCII metin
  - cevaplar: durum kodları ve ilgili deyimler
- Mesajlar 7 bit'lik ASCII metinleridir.

Senaryo: Alice Bob'a mesaj gönderir

1) Alice kullanıcı temsilcisi kullanarak mesaj gönderir ve "to" satırına `bob@someschool.edu` yazar
2) Alice'in kullanıcı temsilcisi mesajı kendi posta sunucusuna gönderir; mesaj mesaj kuyruğuna konur
3) SMTP'nin istemci tarafı Bob'un posta sunucusuna TCP bağlantısı açar
4) SMTP istemcisi Alice'in mesajını TCP bağlantısı üzerinden gönderir.
5) Bob'un posta sunucusu mesajı Bob'un posta kutusuna koyar
6) Bob mesajı okumak için kendi kullanıcı temsilcisini çalıştırır.



Örnek SMTP etkileşimi

```
S: 220 hamburger.edu
C: HELO crepes.fr
S: 250  Hello crepes.fr, pleased to meet you
C: MAIL FROM: <alice@crepes.fr>
S: 250 alice@crepes.fr... Sender ok
C: RCPT TO: <bob@hamburger.edu>
S: 250 bob@hamburger.edu ... Recipient ok
C: DATA
S: 354 Enter mail, end with "." on a line by itself
C: Do you like ketchup?
C: How about pickles?
C: .
S: 250 Message accepted for delivery
C: QUIT
S: 221 hamburger.edu closing connection
```

SMTP: son notlar

- SMTP kalıcı (persistent) bağlantı kullanır
- SMTP mesajları 7 bit'lik ASCII metinden oluşur (header & body)
- SMTP sunucusu msaj sonlarını belirtmek için `CRLF.CRLF` kullanır

CR : carriage return

LF : line feed

HTTP ile farkı:

- HTTP: çekme (pull)
- SMTP: itme (push)

- Her ikisi de etkileşim için ASCII komut ve cevaplarını, durum kodlarını kullanırlar

- HTTP: her nesne kendi cevap mesajı içerisine sarmalanır
- SMTP: tüm nesneler bir mesaj içerisine yerleştirilir.

- HTTP bant içi SMTP bant dışı kontrolü kullanır

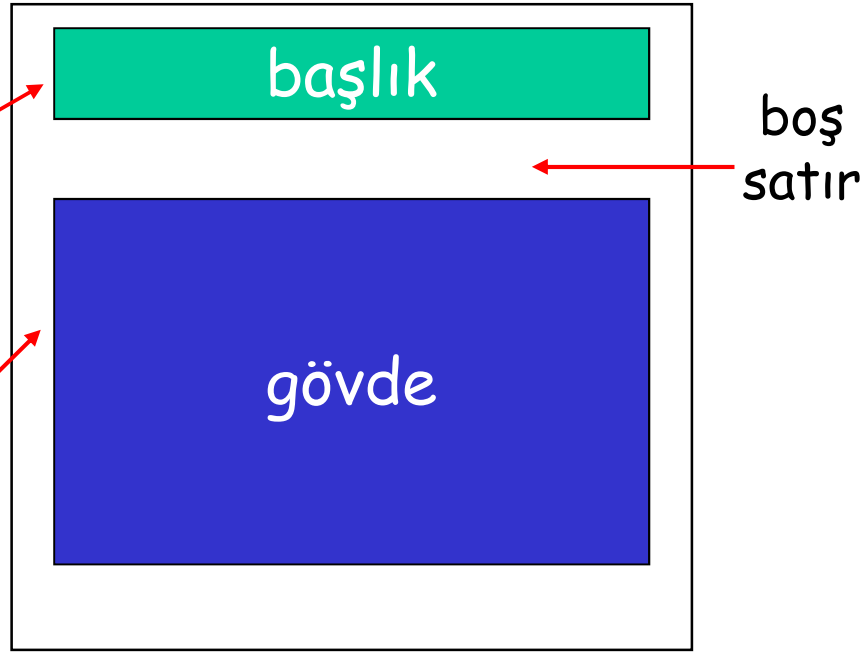
Posta mesajı biçimleri

SMTP: eposta mesajlarının gönderimi için bir protokoldür

RFC 822: metin mesaj biçimini belirleyen standarttır:

- Başlık satırları, e.g.,
  - To:
  - From:
  - Subject:

  SMTP komutlarından farklıdır !
- gövde
  - Sadece ASCII karakterlerden oluşan "mesaj"



Mesaj biçimi: multimedia için

- MIME: çok amaçlı Internet posta uzantıları (multi purpose mail extension), RFC 2045, 2056
- MIME içerik tipini tanımlamak için mesaj başlığına ek başlıklar eklenmelidir

MIME sürümü

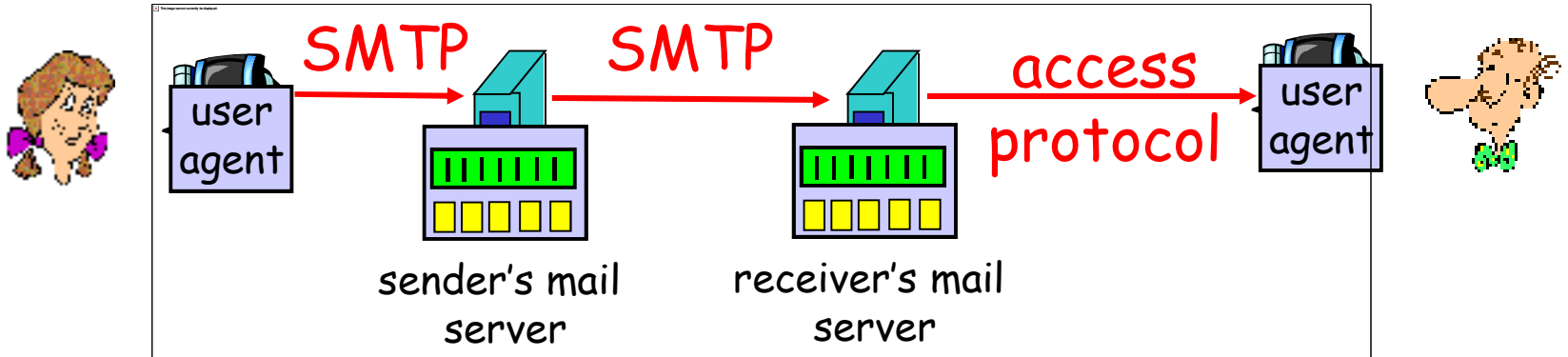
veriyi kodlamada kullanılan metod

multimedia veri tipi, alttipi, parametre tanımı

kodlanmış veri

```
From: alice@crepes.fr
To: bob@hamburger.edu
Subject: Picture of yummy crepe.
MIME-Version: 1.0
Content-Transfer-Encoding: base64
Content-Type: image/jpeg

base64 encoded data .....
.........................
......base64 encoded data
```

Posta erişim protokolleri



- SMTP: alıcı sunucuya gönderme/depolama ile ilgilidir
- Posta erişim protokolleri: sunucudan alma ile ilgilidir
  - POP: Post Office Protocol [RFC 1939]
    - yetkilendirme (agent <-->server) ve download
  - IMAP: Internet Mail Access Protocol [RFC 1730]
    - Daha fazla özellik (daha karmaşık)
    - Sunucuda depolanan mesajların idaresi
  - HTTP: Hotmail , Yahoo! Mail, etc.

Uygulama katmanı

- 2.1 Ağ uygulamalarının prensipleri
- 2.2 Web ve HTTP
- 2.3 FTP
- 2.4 Elektronik Posta
  - SMTP, POP3, IMAP
- 2.5 DNS
- 2.6 Eşler arası uygulamalar P2P

DNS: Internet'in Dizin Servisi

İnsanlar: pek çok tanımlayıcı:
- SSN, isim, pasaport #

Internet ana sistemleri, yönlendiriciler:
- IP adreslerini (32 bit) - datagram ların adreslenmesi için kullanılırlar
- İnsanlar "isim", e.g., ww.yahoo.com – kullanırlar

Q: IP adresleri ile isimler arasında eşleşme nasıl sağlanır?

Etki Alanı Ad Sistemi (Domain Name System):
- Bir *DNS sunucu* hiyerarşisi içerisinde uygulanan *dağıtık bir veritabanı (distributed database)*
- Ana sistemlerin dağıtık veritabanı sorgulamasını sağlayan bir *uygulama katmanı protokolüdür.*
  - not: ana Internet fonksiyonu,uygulama katmanı protokolü olarak uygulanır
  - complexity at network's "edge"

DNS

DNS servisleri

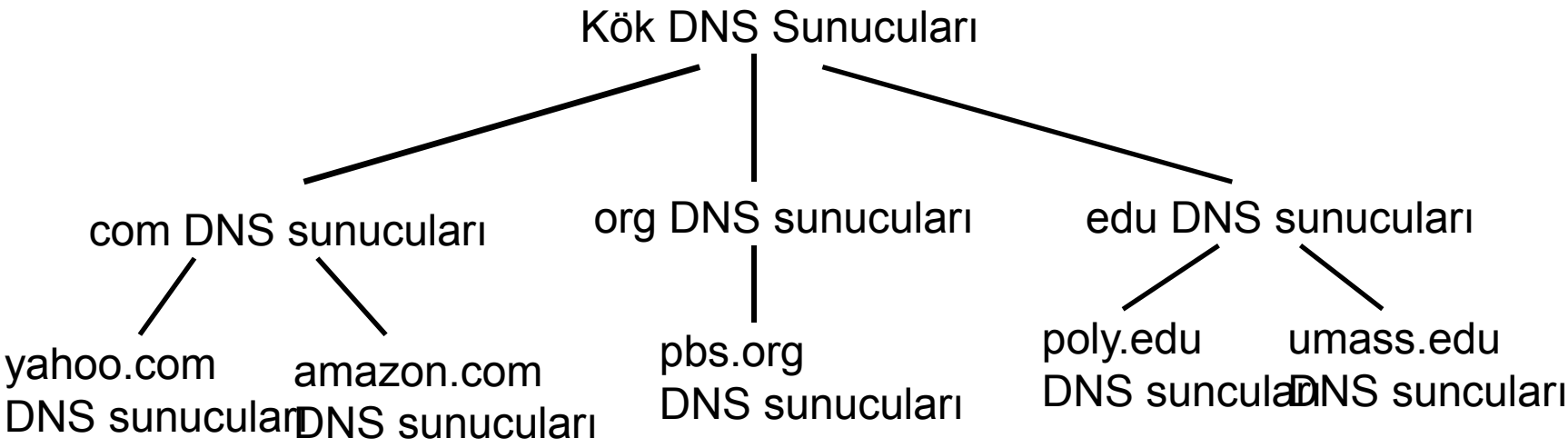
- Ana sistem isimleri-IP adres çevirisi
- Ana sistem lakapları (aliasing)
  - Kurallı (canonical) ve lakap (alias) adları
- Posta sunucusu lakapları
- Yük dağıtımı
  - Çoğaltılan web sunucuları için bir IP adresi seti, bir kurallı isimle ilişkilendirilir

Neden DNS merkezileştirilmiyor?

- Bir tek başarısızlık noktası
- Trafik hacmi
- Uzakta merkezileştirilmiş veritabanı
- bakım

ölçeklenemez!

Dağıtık, Hiyerarşik Veritabanı



İstemci www.amazon.com için IP istiyor; 1inci tahmin:

- İstemci com DNS sunucusunu bulmak için kök dizin sorgusu yapar
- İstemci com DNS sunucusunu amazon.com DNS sunucusunu bulmak için sorgular
- İstemci amazon.com DNS sunucusunu www.amazon.com un IP adresini almak için sorgular

DNS: Kök sunucuları

- İsim eşleştirmesini çözemeyen yerel DNS sunucuları tarafından başvurulurlar
- kök DNS sunucuları:
  - İsim eşleştirmesi bilinmiyorsa yetkili sunucuya başvururlar.
  - Eşleştirmeyi alırlar
  - Eşleştirmeyi yerel sunucuya gönderirler

13 kök DNS sunucusu

TLD and Yetkili Sunucular

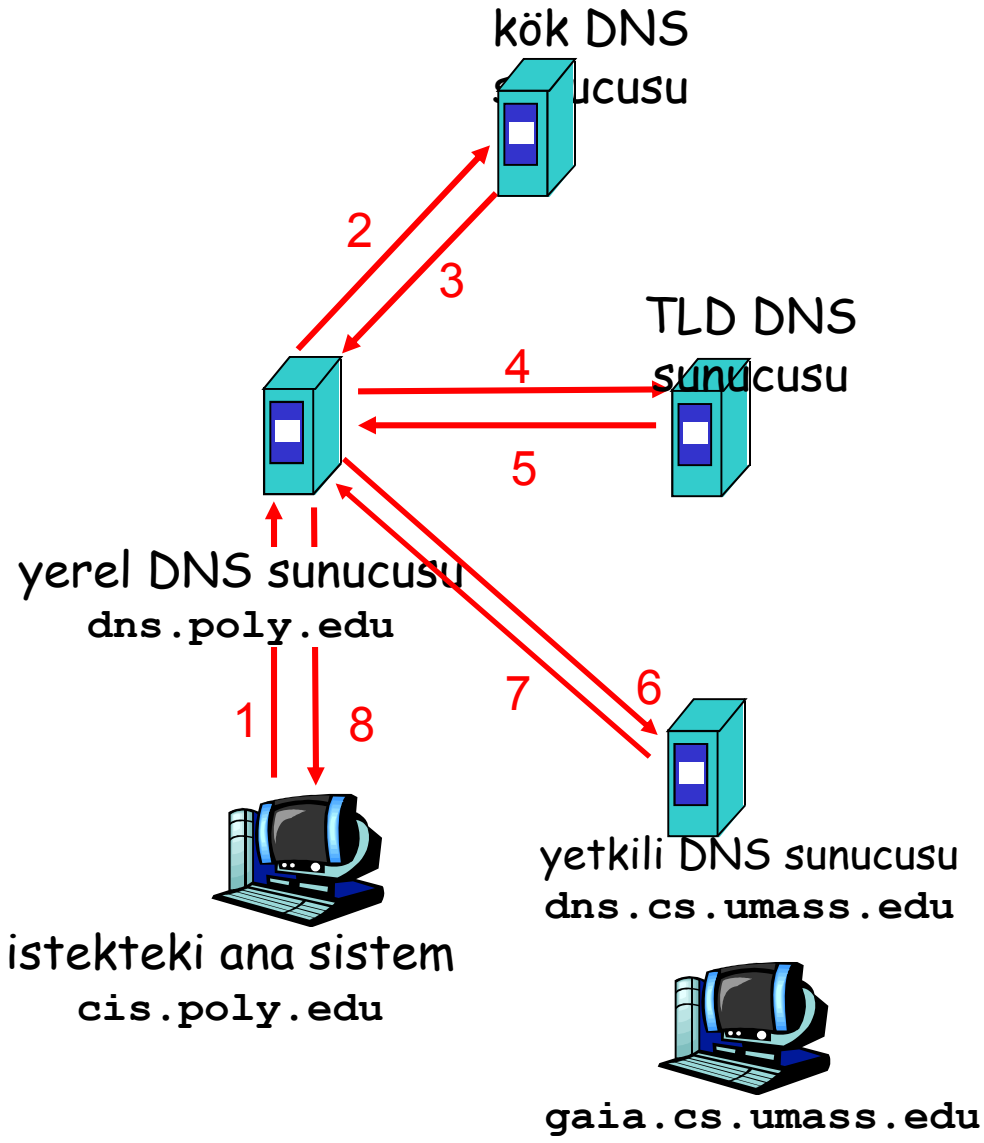
- Üst-seviye etki alanı (TLD) sunucuları: com, org, net, edu, gibi, ve tüm üst düzey ülke etki alanlarından uk, fr, ca, jp, gibi sorumludurlar
  - Network solutions com TLD sunucularını işletmektedir
  - Educause edu TLD sunucularını işletmektedir
- Yetkili DNS sunucuları: kurumların DNS sunucuları, diğer kurum sunucularına yetkili alan adı ve IP eşleşmesi sunarlar (e.g., Web ve posta).
  - Kurum tarafından ya da servis sağlayıcı tarafından işletilebilirler

Yerel DNS Sunucuları

- Katı bir şekilde hiyerarşiye ait değildir
- Her ISP'ye ait bir tane vardır (yerel ISP, şirket, üniversite).
  - "default name server"
- Bir ana sistem DNS sorgusu yaptığında, sorgu kendi yerel DNS sunucuna gönderilir
  - Vekil (proxy) gibi davranır, sorguyu hiyerarşiye gönderir.

Örnek

- cis.poly.edu deki ana sistem gaia.cs.umass.edu için IP adresi ister



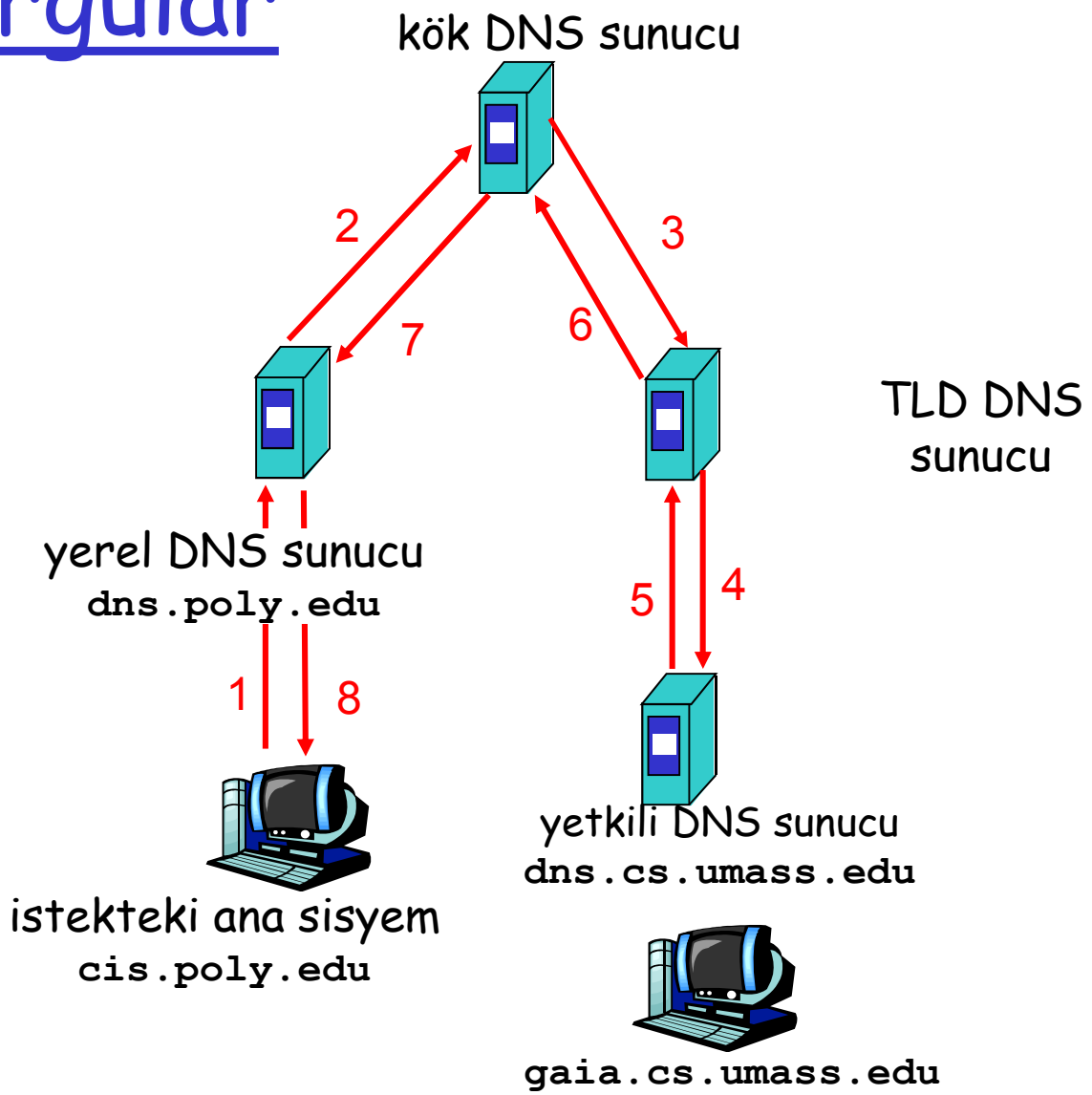
Yinelenen sorgular

yinelenen sorgu:

- İsim eşleştirmenin zorluğunu başvurulan sunucuya yükler
- ağır yük?

tekrarlanan sorgu:

- Başvurulan sunucu başvurulacak sunucu adıyla cevap verir
- "Ben bilmiyorum ama şu sunucuya sorabilirsin"



DNS: tampon bellek ve kayıtları güncelleme

- Bir sunucu eşleştirmeyi öğrendiğinde, eşleştirmeyi tampon belleğe alır
  - Tampon bellek girdileri belli bir süre sonra kaybolurlar (timeout)
  - TLD sunucuları tipik olarak yerel DNS sunucularının tampon belleğinde yer alırlar
    - Bu nedenle kök DNS sunucularına sıklıkla başvurulmaz
- Güncelleme/uyarı mekanizması IETF'nin tasarımındadır
  - RFC 2136
  - http://www.ietf.org/html.charters/dnsind-charter.html

DNS kayıtları

DNS: kaynak kayırlarını (resource records (RR)) depolayan dağıtık veritabanı

RR biçimi: **(name, value, type, ttl)**

- Type=A
  - **name** sistem adıdır
  - **value** IP adresine eşleme sunar
- Type=NS
  - **name** etki alanıdır (e.g. foo.com)
  - **value** bu etki alanına ait yetkili DNS 'in IP sine eşleme sunar
- Type=CNAME
  - **name** lakab ana sistem adıdır

    `www.ibm.com` aslında

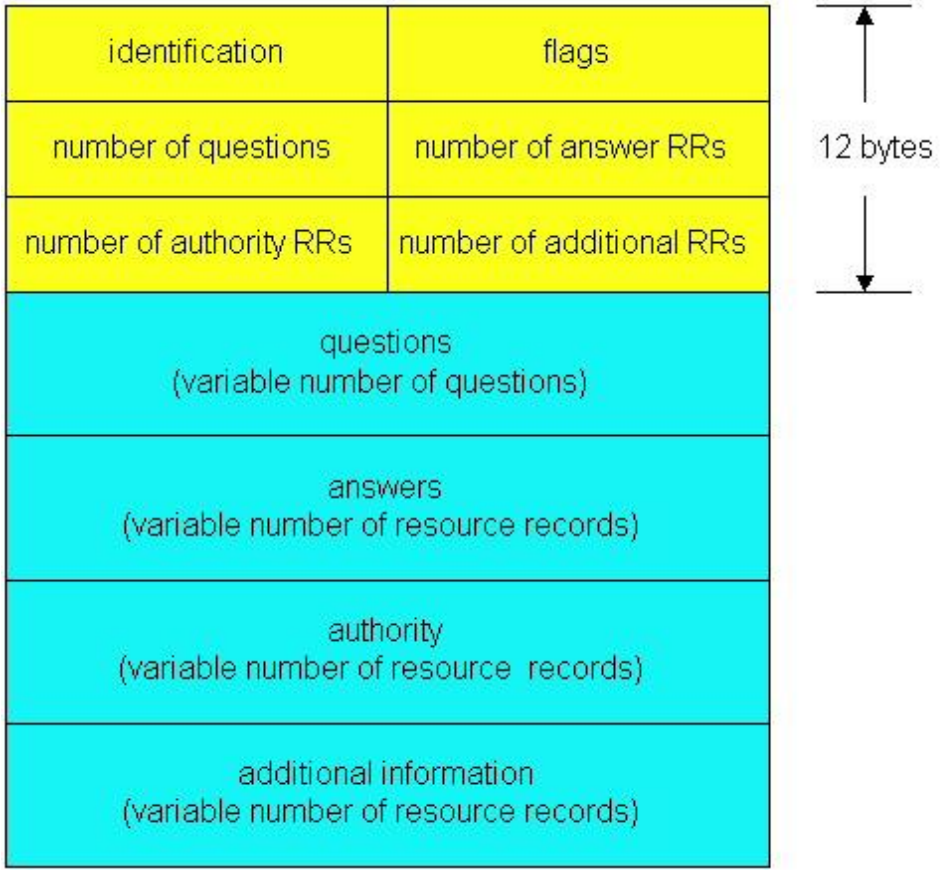
    `servereast.backup2.ibm.com`
  - **value** kurallı ana sistem adıdır
- Type=MX
  - **Value**, **name** ana sistemi lakabına sahip olan bir posta sunucusunun kurallı adıdır.

DNS mesajları

DNS protokolü : *sorgu* ve *cevap* mesajları, aynı *mesaj biçimindedir.*

msg başlığı

- Tanımlama (identification): sorgu için 16 bit lik alan kullanılır ,
- Bayraklar (flags):
  - Sorgu veya cevap
  - İstenen yineleme
  - Mevcut yineleme
  - Cevap yetkilidir



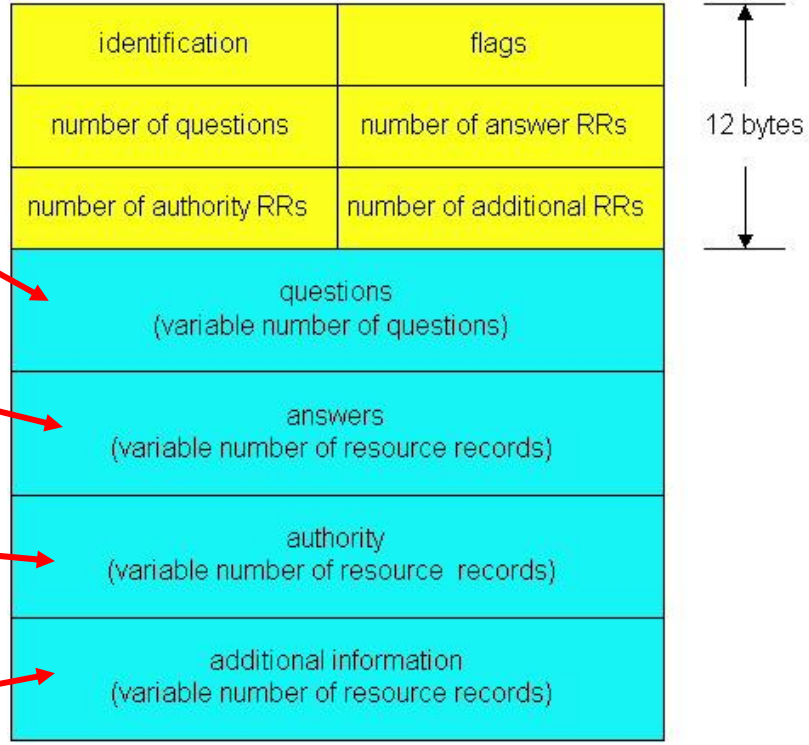
DNS protocol, messages

Bir sorgunun ad, tip alanları

Sorguya verilen Cevaptaki RR'ler

Yetkili sunucuların kayıtları

Kullanılabilen ek "yardımcı" bilgi



DNS veritabanına kayıt girmek

- Örnek: "Network Utopia" yı yeni oluşturduk
- networkuptopia.com adını kaydedici (registrar) 'ye kayıt ettiririz. (örn, Network Solutions)
  - Kaydediciye yekiliDNS sunucumuzun adı ve IP adresini de vermek gerekir (birincil ve ikincil)
  - Kaydedici TLD sunucusuna iki RR ekler:

```
(networkutopia.com, dns1.networkutopia.com, NS)
(dns1.networkutopia.com, 212.212.212.1, A)
```

- Yetkili sunucuya www.networkuptopia.com için Type A kaydı ve mail.networkutopia.com için Type MX kaydı konulur
- İnsanlar sizin sitenizin IP adresine nasıl erişecekler?

Uygulama Katmanı

- 2.1 Ağ uygulamalarının prensipleri
- 2.2 Web ve HTTP
- 2.3 FTP
- 2.4 Elektronik Posta
  - SMTP, POP3, IMAP
- 2.5 DNS
- 2.6 Eşler arası uygulamalar P2P

P2P dosya paylaşımı

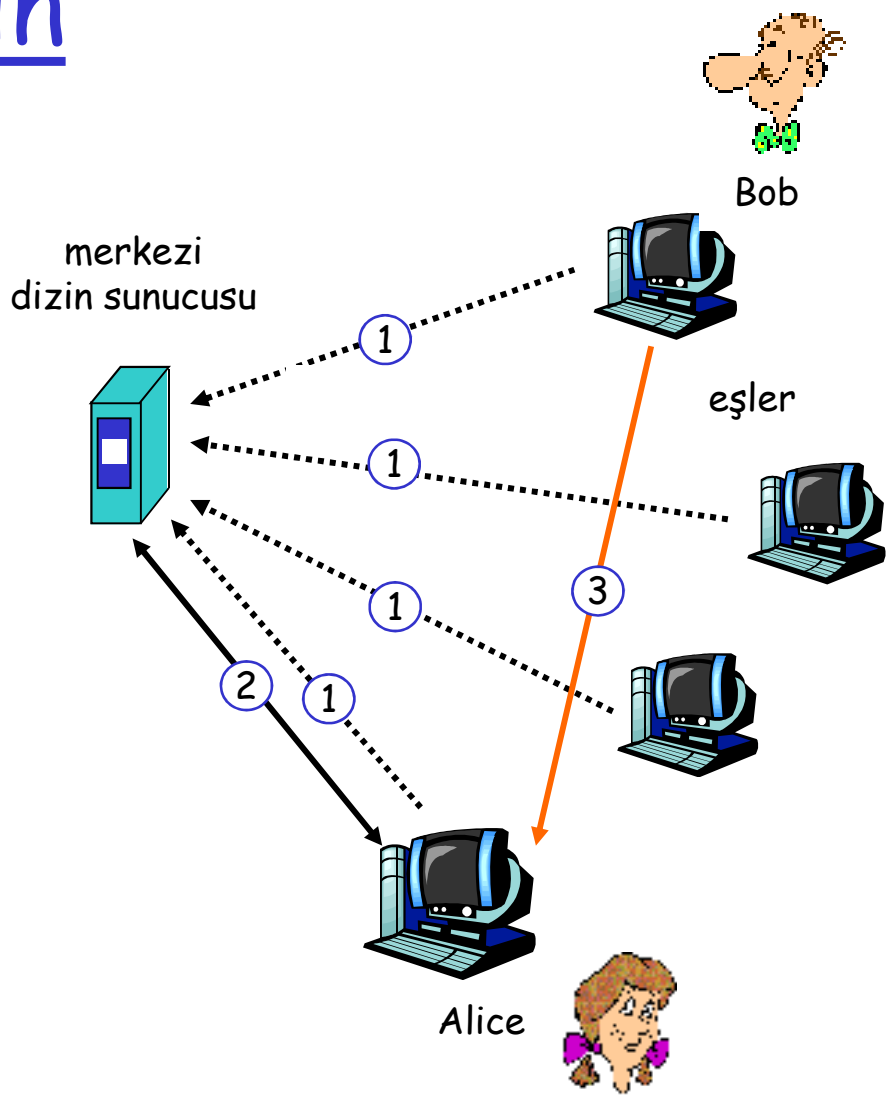
Örnek

- Alice laptop'unda eşler arası istemci uygulaması çalıştırıyor
- Zaman zaman Internet'e bağlanıp, her bağlantı için yeni bir IP adresi alıyor
- "Hey Jude"'u arıyor
- Uygulama Hey Jude'un kopyasına sahip olan diğer eşleri görüntülüyor
- Alice eşlerden birini, Bob'u seçiyor.
- Dosya Bob'un bilgisayarından Alice'in laptop una kopyalanıyor.
- Alice dosya indirirken, diğer kullanıcılar da Alice'den dosya indiriyorlar.
- Alice'in eşleri hem web istemcisi hem de Web sunucusu olarak davranıyorlar
- Tüm eşler sunucu = fazlasıyla ölçeklenebilir!

P2P: merkezi dizin

orijinal "Napster" tasarımı

1) Bir eş bağlandığında, merkezi sunucuyu bilgilendirir:
   - IP adresi
   - içerik
2) Alice "Hey Jude"'u sorgular
3) Alice Bob dan dosyayı ister



P2P: merkezi dizin ile ilgili problemler

- Tek hata noktası
- Performans düşüklüğü
- Copyright ihlali

Sorgu baskını (query flooding): Gnutella

- Tümüyle dağıtık
  - Merkezi sunucu yok
- Genel alan protokolü (public domain protocol)
- Çok sayıda Gnutella istemcisi protokolü uygular

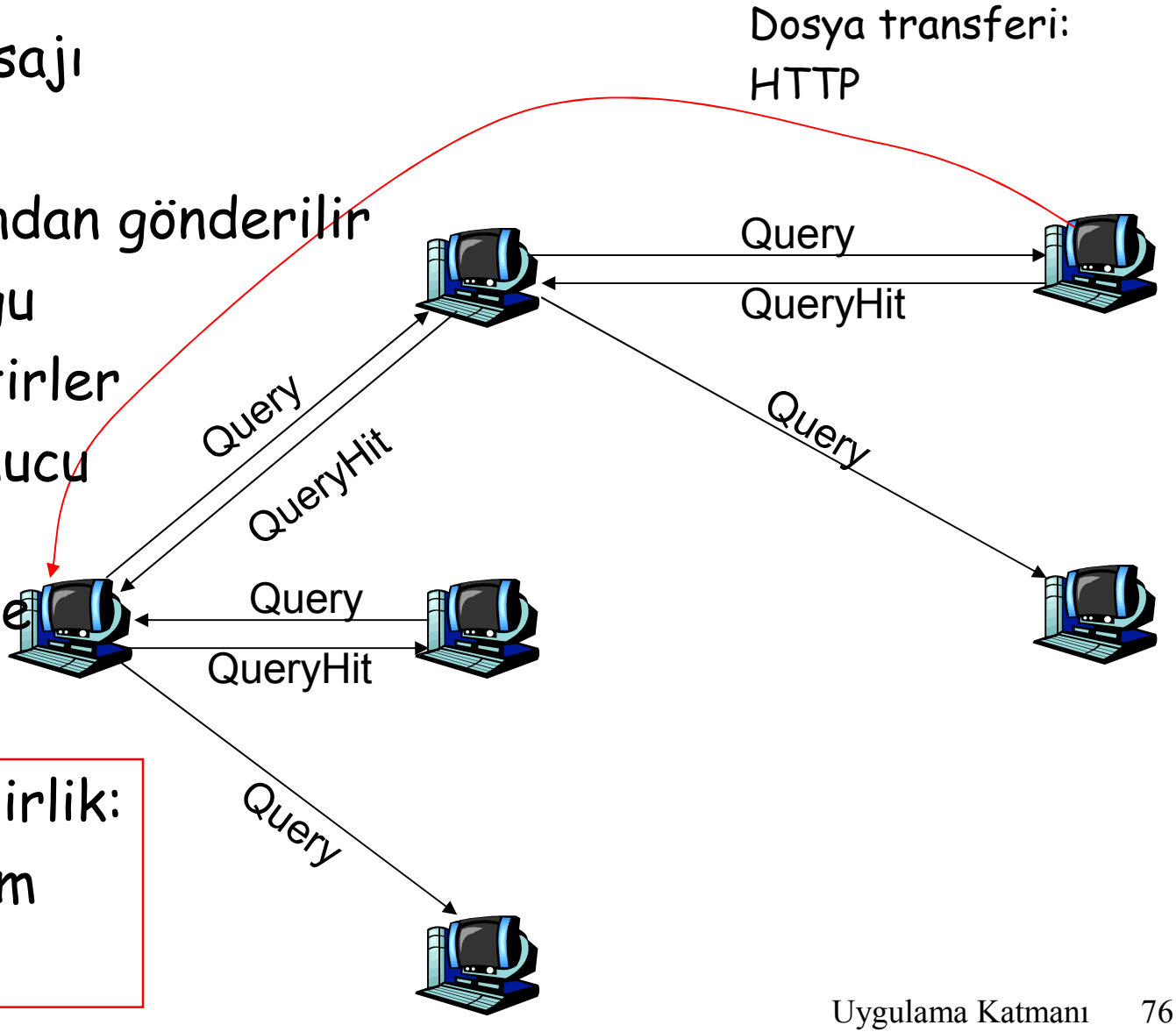
Kaplama ağ (overlay network): grafik

- Eğer TCP bağlantısı varsa, X ve Y eşleri arasında sınır var
- Tüm aktif eşler ve sınırlar kaplama ağı oluşturur
- Sınır fiziksel bir hat değildir
- Herhangi bir eş < 10 kaplama komşu ile bağlıdır

Gnutella: protocol

- Sorgu mesajı varolan TCP bağlantılarından gönderilir
- eşler sorgu mesajını iletirler
- Sorgu sonucu (QueryHit) yoldan geriye gönderilir

Ölçeklenebilirlik: Kısıtlı kapsam flooding

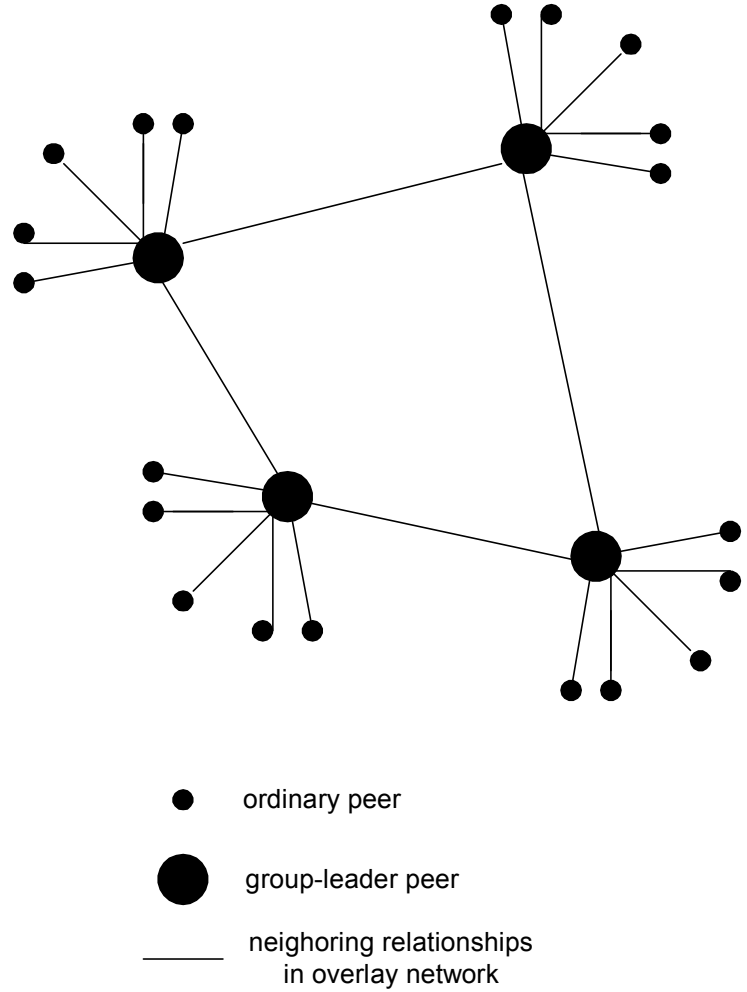


Gnutella: Eşin katılımı

1. Katılan eş X Gnutella ağında başka bir eş bulmalıdır: aday eşler listesini kullanır
2. X Yile bağlantı kurana kadar sırayla listedeki eşlerle TCP yapmayı dener
3. X Y ye ping mesajı gönderir; Y Ping mesajını iletir.
4. Ping mesajını alan tüm eşler Pong mesajı ile cevap verir
5. X pek çok Pong mesajı alır. Sonra ek TCP bağlantıları da kurabilir.

Heterojenliği kullanma: KaZaA

- Her eş bir grup lideridir veya bir grup liderine atanmıştır.
  - Eş ve grup lideri arasında TCP bağlantısı.
  - Bazı grup lideri çiftleri arasında TCP bağlantıları.
  - Grup liderleri kendi çocuklarındaki içeriğin bilgisini izlerler.



KaZaA: Sorgulama

- Her dosyanın bir hash i ve tanımlayıcısı (descriptor) vardır.
- İstemci grup liderine bir anahtar kelime içeren sorgu gönderir
- Grup lideri eşleşmelerle cevap verir:
  - Her bir eşleşme için: metadata, hash, IP address
- Eğer grup lideri sorguyu başka grup liderlerine gönderirse, onlarda eşleşmelerle cevap verirler.
- İstemci daha sonra indireceği dosyayı seçer
  - Hash i tanımlayıcı olarak kullanan HTTP istekleri istenen dosyayı tutan eşlere gönderilir.

Uygulama Katmanı

- 2.1 Ağ uygulamalarının prensipleri
- 2.2 Web ve HTTP
- 2.3 FTP
- 2.4 Elektronik Posta
  - SMTP, POP3, IMAP
- 2.5 DNS
- 2.6 Eşler arası uygulamalar P2P

Soket programlama

Amaç: soketleri kullanarak iletişim kuran bir istemci/sunucu uygulamasının nasıl yapılacağını öğrenmek.

Soket API'si

- 1981'de BSD4.1 UNIX ile tanıtıldı
- Uygulamalar tarafından yaratıldı, kullanıldı ve sunuldu
- İstemci/sunucu paradigması
- Soket API'si ile iki taşıma hizmeti:
  - Güvenilir olmayan datagram
  - güvenilir, byte stream yönelimli

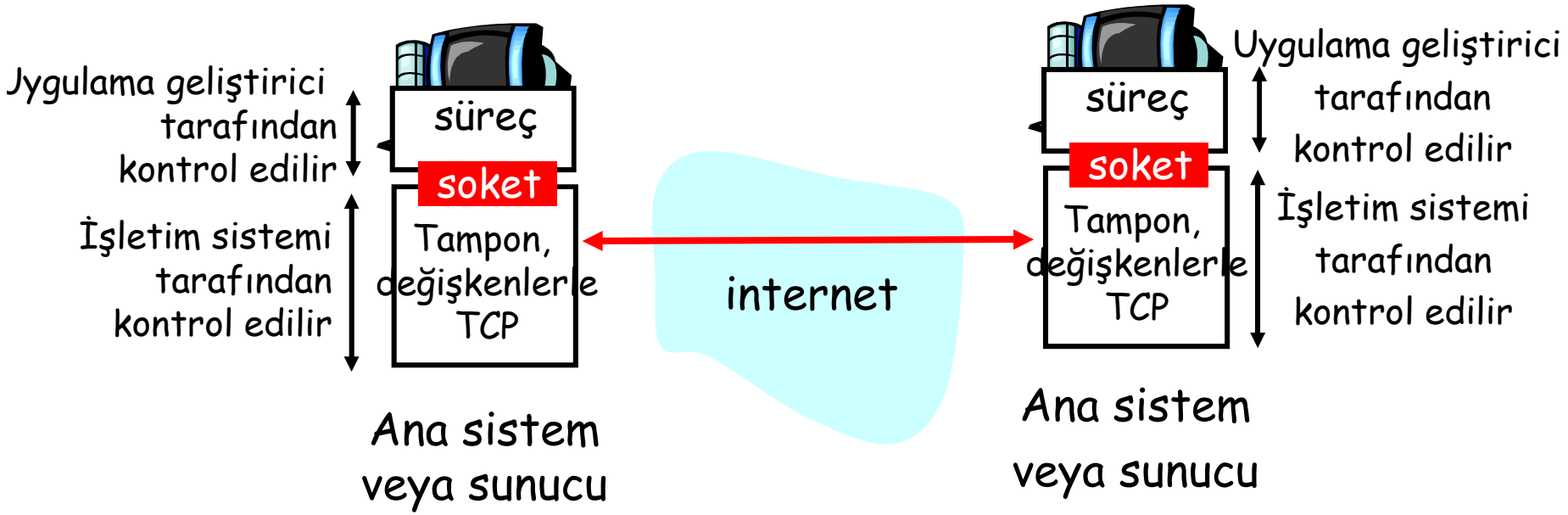
soket

bir uygulama sürecinin bir diğerine/diğerinden mesaj *alması ve göndermesini* sağlayan *ana sisteme özel*, *uygulama tarafından yapılmış*, *İşletim sistemi kontrollü* arayüz ("kapı")

TCP ile soket programlama

Soket: uygulama süreci ve uçtan uca taşıma protokolü (UCP veya TCP) arasında bir kapı

TCP servisi: bir süreçten diğerine bitlerin güvenli iletimi



TCP ile soket programlama

İstemci sunucu ile iletişime geçmeli

- Sunucu süreci öncelikle çalışır olmalı
- Sunucu istemcinin iletişimini karşılayacak soket (kapı) oluşturmalı

İstemci sunucuyla:

- İstemciye özel TCP soketi oluşturarak
- Sunucu sürecin IP adresini ve port numarasını belirleyerek iletişim kurar.
- İstemci soket oluşturduğunda: istemci TCP sunucu TCP si ile bağlantı kurmuş olur

- İstemci tarafından iletişime geçen sunucu TCP si sunucu sürecinin istemciyle haberleşmesi için yeni bir soket oluşturur. server TCP creates new socket
  - Sunucunun birden fazla istemci ile konuşmasını sağlar
  - Kaynak port numaraları istemcilerin ayırt edilmesini sağlar

uygulama bakış açısı

TCP istemci ile sunucu arasında güvenilir, sıralı byte ("boru") transferini sağlar

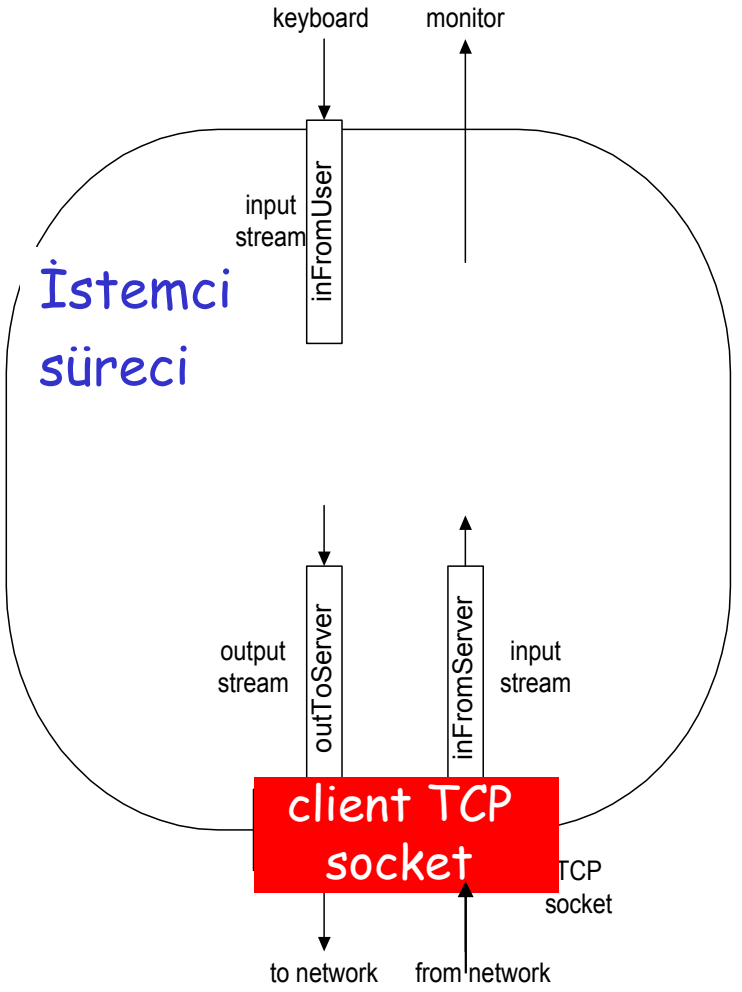
Akım (Stream) jargon

- Akım (stream) sürecin içine ya da içinden akan karakterler sırasıdır.
- Girdi akımı (input stream) süreç için bir girdi kaynağına bağlıdır, örn, klavye veya soket.
- Çıktı akımı (output stream) bir çıktı kaynağına bağlıdır, eg, monitor veya soket.

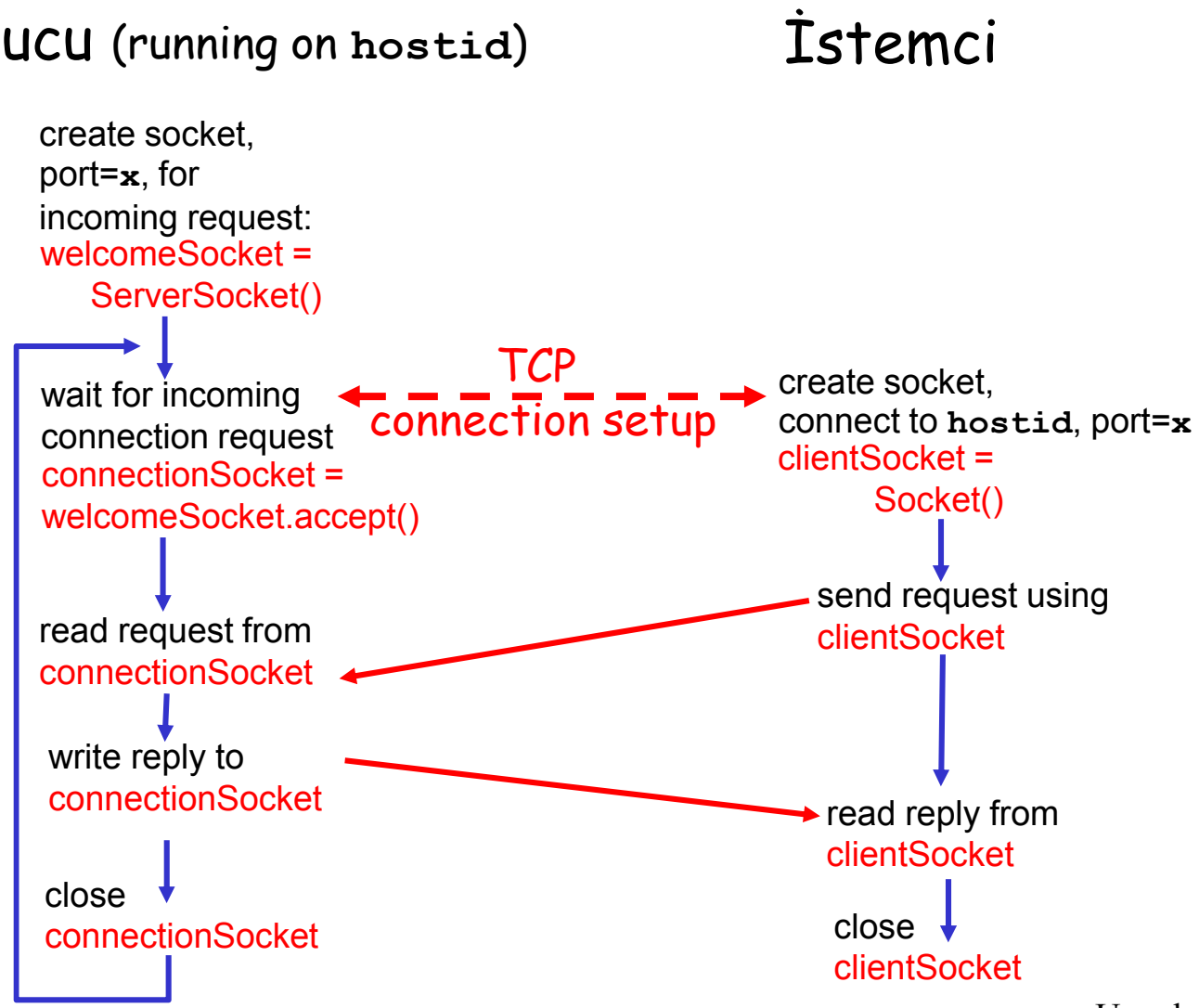
TCP ile soket programlama

Örnek istemci/sunucu uygulaması

1) İstemci standart girdi den satır okur (**`inFromUser`** stream) , soket aracılığıyla sunucuya gönderir (**`outToServer`** stream)
2) Sunucu soketten satırı okur
3) Sunucu satırı büyük harfe çevirir, istemciye geri gönderir
4) İstemci okur, soketteki değişen satırı çıktı alır (**`inFromServer`** stream)



İstemci sunucu soket etkileşimi: TCP



Örnek: Java istemcisi (TCP)

```
import java.io.*;
import java.net.*;
class TCPClient {

    public static void main(String argv[]) throws Exception
    {
        String sentence;
        String modifiedSentence;

        BufferedReader inFromUser =
          new BufferedReader(new InputStreamReader(System.in));

        Socket clientSocket = new Socket("hostname", 6789);

        DataOutputStream outToServer =
          new DataOutputStream(clientSocket.getOutputStream());
```

Create input stream

Create client socket, connect to server

Create output stream attached to socket

Örnek: Java istemcisi (TCP), dev.

Create input stream attached to socket

```
BufferedReader inFromServer =
  new BufferedReader(new
  InputStreamReader(clientSocket.getInputStream()));

sentence = inFromUser.readLine();

outToServer.writeBytes(sentence + '\n');

modifiedSentence = inFromServer.readLine();

System.out.println("FROM SERVER: " + modifiedSentence);

clientSocket.close();

    }
}
```

Send line to server

Read line from server

Örnek: Java sunucusu (TCP)

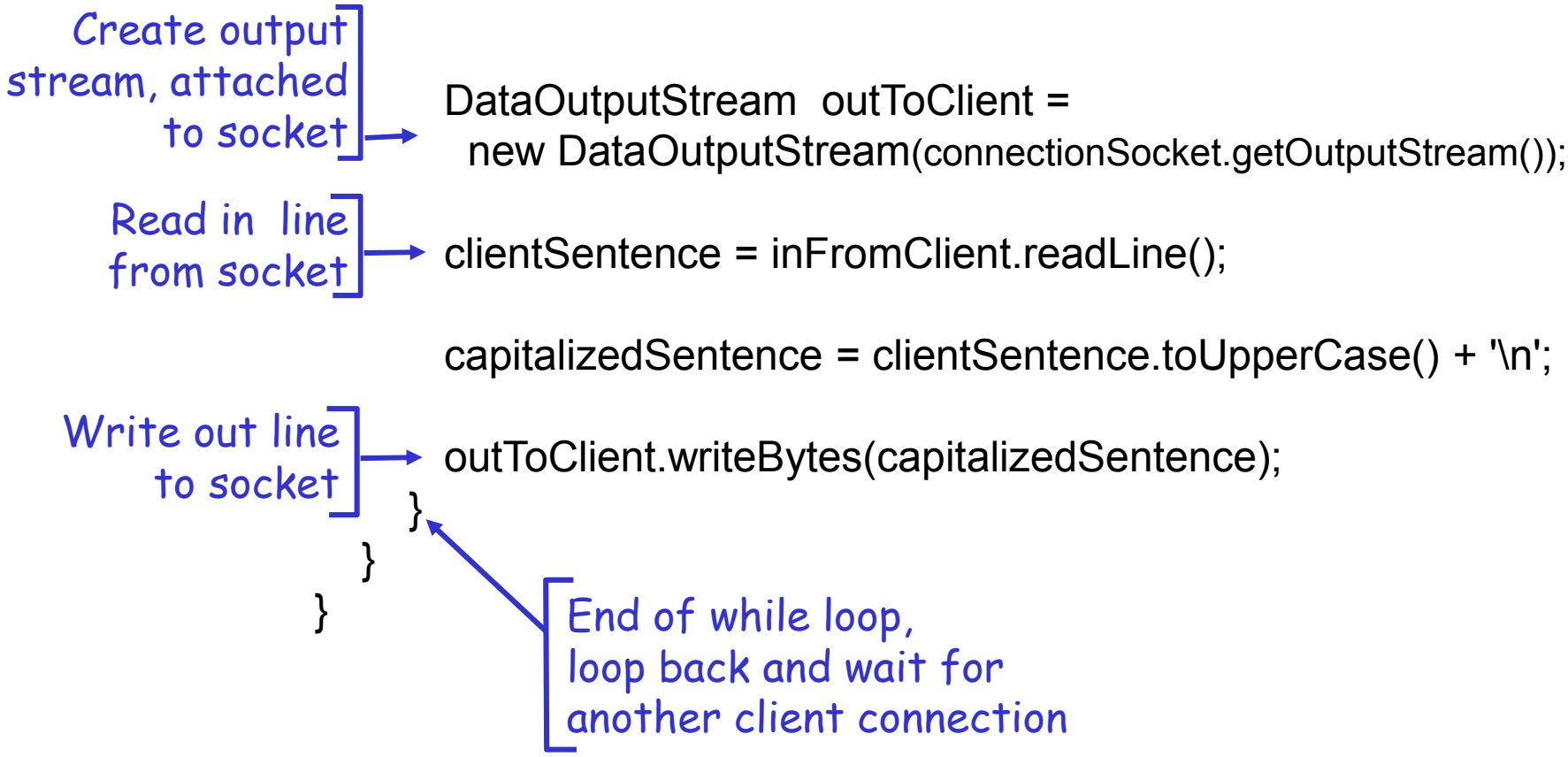
```
import java.io.*;
import java.net.*;

class TCPServer {

  public static void main(String argv[]) throws Exception
   {
     String clientSentence;
     String capitalizedSentence;

     ServerSocket welcomeSocket = new ServerSocket(6789);

     while(true) {

       Socket connectionSocket = welcomeSocket.accept();

      BufferedReader inFromClient =
        new BufferedReader(new
        InputStreamReader(connectionSocket.getInputStream()));
```

Create welcoming socket at port 6789

Wait, on welcoming socket for contact by client

Create input stream, attached to socket

Örnek: Java sunucusu (TCP), dev.



Uygulama Katmanı

- 2.1 Ağ uygulamalarının prensipleri
- 2.2 Web ve HTTP
- 2.3 FTP
- 2.4 Elektronik Posta
  - SMTP, POP3, IMAP
- 2.5 DNS
- 2.6 Eşler arası uygulamalar P2P
- 2.7 TCP ile soket programlama
- 2.8 UDP ile soket programlama

UDP ile Soket Programlama

UDP: istemci ile sunucu arasında "bağlantı" yoktur

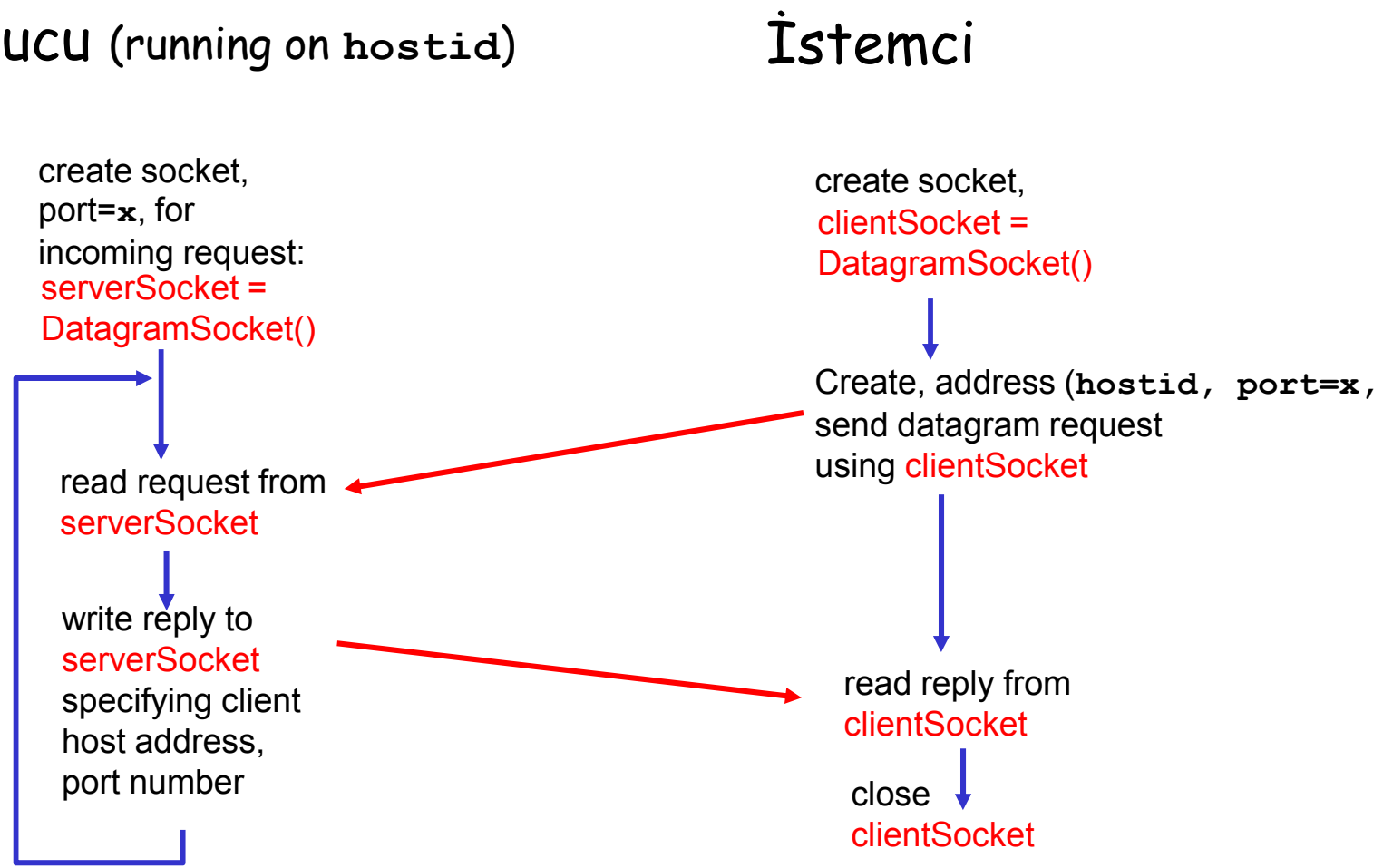
- El sıkışma yoktur
- Gönderici her pakete IP adresi ve hedef port numarasını ekler
- Sunucu gelen paketteki IP adresi ve portu açmak zorundadır.

UDP: gönderilen veri sırasız bir şekilde alınabilir ya da kaybolabilir

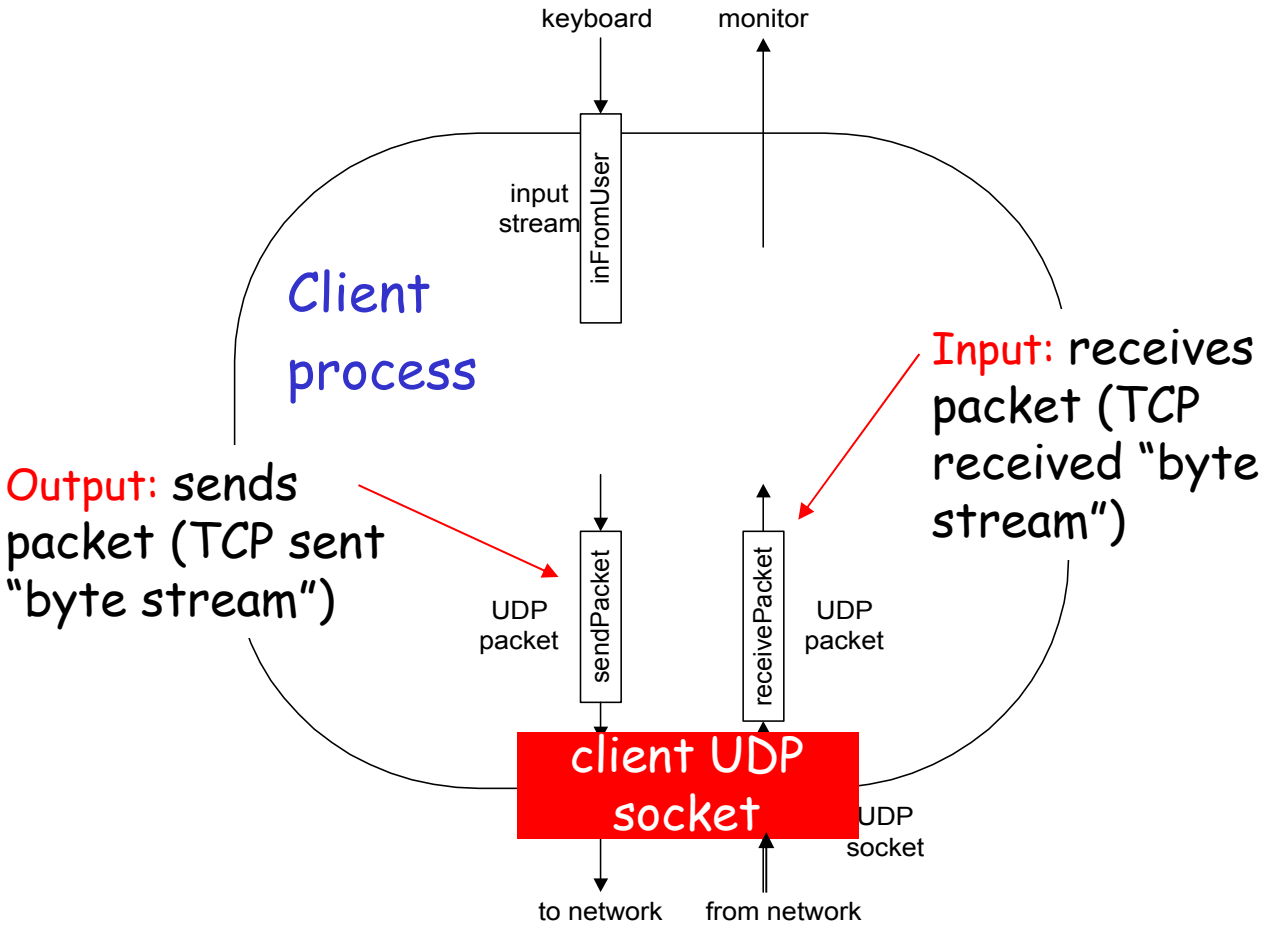
uygulama bakış açısı

UDP istemci ve sunucu arasında byte grupları ("datagram") için güvenilir olmayan transfer sağlar

İstemci sunucu soket etkileşimi: UDP



Örnek: Java istemcisi (UDP)



Örnek: Java istemcisi (UDP)

```
import java.io.*;
import java.net.*;

class UDPClient {
    public static void main(String args[]) throws Exception
    {

      BufferedReader inFromUser =
        new BufferedReader(new InputStreamReader(System.in));

      DatagramSocket clientSocket = new DatagramSocket();

      InetAddress IPAddress = InetAddress.getByName("hostname");

      byte[] sendData = new byte[1024];
      byte[] receiveData = new byte[1024];

      String sentence = inFromUser.readLine();

      sendData = sentence.getBytes();
```

Create input stream

Create client socket

Translate hostname to IP address using DNS

Örnek: Java istemcisi (UDP), dev.

Create datagram with data-to-send, length, IP addr, port

Send datagram to server

Read datagram from server

```
                DatagramPacket sendPacket =
                  new DatagramPacket(sendData, sendData.length, IPAddress, 9876);

                clientSocket.send(sendPacket);

                DatagramPacket receivePacket =
                   new DatagramPacket(receiveData, receiveData.length);

                clientSocket.receive(receivePacket);

                String modifiedSentence =
                    new String(receivePacket.getData());

                System.out.println("FROM SERVER:" + modifiedSentence);
                clientSocket.close();
                }

            }
```

Örnek: Java sunucusu (UDP)

```
import java.io.*;
import java.net.*;

class UDPServer {
  public static void main(String args[]) throws Exception
    {
      DatagramSocket serverSocket = new DatagramSocket(9876);

      byte[] receiveData = new byte[1024];
      byte[] sendData  = new byte[1024];

      while(true)
        {
          DatagramPacket receivePacket =
            new DatagramPacket(receiveData, receiveData.length);
          serverSocket.receive(receivePacket);
```

Create datagram socket at port 9876

Create space for received datagram

Receive datagram

Örnek: Java sunucusu(UDP), dev

```
String sentence = new String(receivePacket.getData());

InetAddress IPAddress = receivePacket.getAddress();

int port = receivePacket.getPort();

String capitalizedSentence = sentence.toUpperCase();

sendData = capitalizedSentence.getBytes();

DatagramPacket sendPacket =
   new DatagramPacket(sendData, sendData.length, IPAddress,
                      port);

serverSocket.send(sendPacket);
    }
  }
}
```

Get IP addr port #, of sender

Create datagram to send to client

Write out datagram to socket

End of while loop, loop back and wait for another datagram

ÖZET: Ne öğrendik??

- Uygulama mimarileri
  - İstemci/sunucu
  - P2P
  - hybrid
- Uygulama servis gereksinimleri:
  - güvenilirlik, bant genişliği, gecikme
- Internet taşıma servis modeli
  - Balantı yönelimli, güvenilir: TCP
  - Güvenilr olmayan, datagrams: UDP
- Bazı özel protokoller:
  - HTTP
  - FTP
  - SMTP, POP, IMAP
  - DNS
- Soket programlama

ÖZET

En önemlisi: protokoller hakkında öğrendik...

- Tipik istek/cevap mesaj değişimi:
  - İstemci bilgi ya da servis ister
  - Sunucu veri ya da durum kodu ile cevap verir
- Mesaj biçimleri:
  - başlıklar: veril ile ilgili bilgi veren alanlar
  - veri: iletilecek bilgi
- kontrol vs. veri mesajları
  - hat içi, hat dışı
- merkezi vs. merkezi olmayan
- durumsuz vs. durumlu
- güvenilir vs. güvenilir olmayan mesaj transferi
- "ağ sınırında karmaşıklık"