BM 402 Bilgisayar Ağları
(Computer Networks)

M.Ali Akcayol
Gazi Üniversitesi
Bilgisayar Mühendisliği Bölümü

Not: Bu dersin sunumları, ders kitabının yazarları James F. Kurose ve Keith W. Ross tarafından sağlanan sunumlar üzerinde değişiklik yapılarak hazırlanmıştır.

Ders konuları

- Ağ uygulamaları
  - Ağ uygulama mimarileri
  - Process'ler arası iletişim
  - Uygulamalar için transport layer hizmetleri
  - İnternet'in sağladığı transport layer hizmetleri
  - Uygulama katmanı protokolleri
- Web ve HTTP
  - HTTP 'nin özellikleri
  - Kalıcı ve kalıcı olmayan bağlantı
  - HTTP mesaj formatı
  - Cookie'ler
  - Web caching
  - Şartlı Get
- FTP
  - FTP komutları ve cevapları

2

Ağ uygulamalarının temelleri

Amaç:

- Ağ uygulama protokollerinin oluşturulması
  - transport-layer hizmet modelleri
  - client-server yaklaşım
  - Peer-to-peer yaklaşım
- Popüler application layer protokolleri
  - HTTP
  - FTP
  - SMTP / POP3 / IMAP
  - DNS
- Ağ uygulamaları programlama
  - socket API (application programming interface)

3

Bazı ağ uygulamaları

- E-posta
- Web
- Instant messaging
- Remote login
- P2P dosya paylaşımı
- Çok kullanıcılı ağ oyunları
- Streaming
- Internet telefon
- Real-time video konferans
- Paralel işlem

4

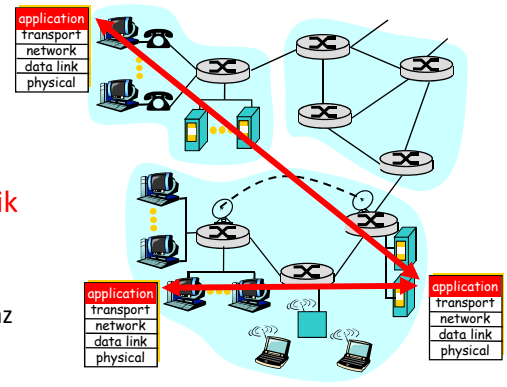
Ağ uygulaması oluşturma

Yazılan programlar

- Farklı uç sistemlerde çalışır
- Ağ üzerinden haberleşir
- örnek, Web: Web server yazılımı browser yazılımı ile haberleşir

Ağ temel elemanlarına yönelik yazılım yapılmaz

- Network core cihazlar application layer'da çalışmaz
- Bu tasarım hızlı uygulama geliştirmeye izin verir



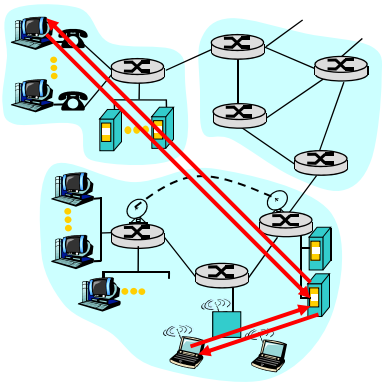
5

Uygulama mimarileri

- Client-server
- Peer-to-peer (P2P)
- Hibrid (client-server ve P2P)

6

Client-server mimari



server:

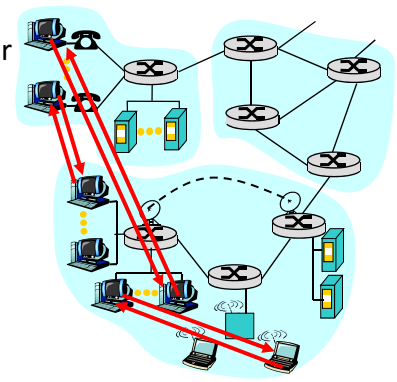
- Always-on host (her zaman açık)
- Sabit IP adres
- Ölçekleme için çok sayıda sunucu

clients:

- Sunucuyla haberleşir
- Aralıklarla birbirine bağlanabilir
- Dinamik IP adres olabilir
- Birbirine doğrudan bağlı değil

P2P mimari

- Her zaman açık sunucu yoktur
- Uç sistemler doğrudan bağlanır
- Uç sistemler aralıklarla doğrudan bağlanabilir ve IP adres değiştirebilir
- örnek: Gnutella, eDonkey



Yüksek ölçeklenirdir

Yönetim zordur

Hibrid (client-server ve P2P)

Napster

- Dosya transferi P2P
- Dosya arama merkezi:
  - Merkezi sunucuya uç birimler kayıt olur
  - Uç birimler içerik aramayı merkezi sunucuda yapar

Instant messaging

- İki kullanıcı arasında chat P2P yapılır
- Açık olup olmadığı denetimi merkezi:
  - Kullanıcı online olduğunda merkezi sunucuya IP adresi kayıt edilir
  - Kullanıcılar IP adres arayacaklarında merkezi sunucuyla iletişime geçerler

9

İşlemlerin iletişimi (process communication)

Process: host üzerinde çalışan program.

- Aynı host üzerinde, iki process inter-process communication ile haberleşir (OS tanımlar).
- Farklı host'lardaki process'ler mesajlarla haberleşir

Client process: iletişimi başlatan process

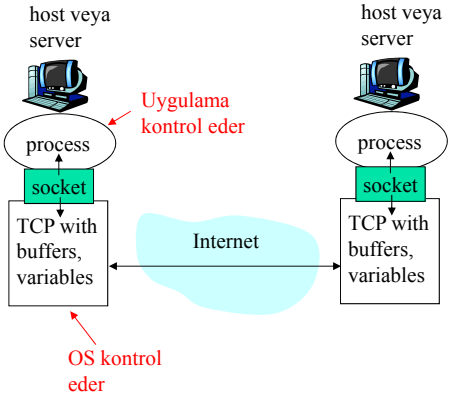
Server process: iletişim başvurusu için bekleyen process

- P2P uygulama mimarileri client ve server işlemlerine sahiptir

10

Soketler

- Process'ler kendi soketlerine mesaj gönderir veya alır
- soketler kapılara benzer
  - Gönderici process mesajı kapıdan dışarı gönderir
  - Gönderici process kapının diğer tarafındaki transport altyapısına güvenir



- API: (1) transport protokol seçer; (2) Parametre belirler

11

Process adresleme

- Mesajı alan process için bir tanımlayıcı gerekir
- Host 32-bit IP adrese sahiptir
- Çok sayıda process aynı host üzerinde açlıştığı için IP adres tanımlayıcı olamaz
- Tanımlayıcı hem IP adresini hemde port numarasını bir process'le ilişkilendirir.
- Örnek port numaraları:
  - HTTP server: 80
  - Mail server: 25

12

Application layer protokol tanımları

- İletilen mesaj tipleri: request, response
- Mesaj tiplerinin yazımı (syntax): mesaj içindeki alanlar tanımlanır
- Alanların anlamları (semantic) tanımlanır
- Process'in gönderdiği ve cevapladığı mesajların kuralları belirlenir

Public-domain protokoller:

- RFCs dökümanları ile tanımlanır
- Birlikte çalışabilirliği sağlar
- HTTP, SMTP

Özel protokoller:

- KaZaA

13

Transport servisleri application'ın isteklerini karşılar

Veri kaybı

- Bazı uygulamalar (örnek, audio, video) belirli kaybı tolere eder
- Diğer uygulamalar (örnek, file transfer, telnet) 100% güvenilir veri transferi ister

Zamanlama

- Bazı uygulamalar (örnek, Internet telefon, interaktif oyunlar) iyi performans için düşük gecikme ister

Bant genişliği

- Bazı uygulamalar (örnek, multimedya) gerekli minimum bantgenişliğini ister
- Diğer uygulamalar bant genişliği ne olursa olsun çalışabilir

14

Yaygın uygulamalar için transport servis ihtiyaçları

Application	Data loss	Bandwidth	Time Sensitive
file transfer	no loss	elastic	no
e-mail	no loss	elastic	no
Web documents	no loss	elastic	no
real-time audio/video	loss-tolerant	audio: 5kbps-1Mbps video:10kbps-5Mbps	yes, 100's ms
stored audio/video	loss-tolerant	Üsttekiyle aynı	yes, birkaç saniye
interactive games	loss-tolerant	birkaç kbps	yes, 100's ms
instant messaging	no loss	elastic	yes/no

15

Internet transport prokol servisleri

TCP servisi:

- *connection-oriented:* client ve server process'leri arasında setup gerekir
- *reliable transport: gönderen ve alan process'ler arasında*
- *flow control:* gönderen alana çok fazla göndermez
- *congestion control:* ağ yoğunluğuna göre düzenleme yapılır
- *sağlayamadıkları:* zamanlama, minimum bandwidth garatisi

UDP servisi:

- Gönderici ve alıcı arasında güvenilir olmayan veri transferi
- sağlayamadıkları: connection setup, reliability, flow control, congestion control, timing, veya bandwidth garantisi

16

Internet transport prokol uygulamaları

Application	Application layer protocol	Underlying transport protocol
e-mail	SMTP [RFC 2821]	TCP
remote terminal access	Telnet [RFC 854]	TCP
Web	HTTP [RFC 2616]	TCP
file transfer	FTP [RFC 959]	TCP
streaming multimedia	özel (RealNetworks)	TCP veya UDP
Internet telephony	proprietary (Dialpad)	UDP

17

Ders konuları

- Ağ uygulamaları
  - Ağ uygulama mimarileri
  - Process'ler arası iletişim
  - Uygulamalar için transport layer hizmetleri
  - İnternet'in sağladığı transport layer hizmetleri
  - Uygulama katmanı protokolleri
- Web ve HTTP
  - HTTP 'nin özellikleri
  - Kalıcı ve kalıcı olmayan bağlantı
  - HTTP mesaj formatı
  - Cookie'ler
  - Web caching
  - Şartlı Get
- FTP
  - FTP komutları ve cevapları

18

Web ve HTTP

Kavramlar

- Web sayfası (Web page), nesnelerden (objects) oluşur
- Objectler HTML dosyası, JPEG image, Java applet, audio file,… olabilir
- Her object URL ile adreslenebilir
- URL:

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

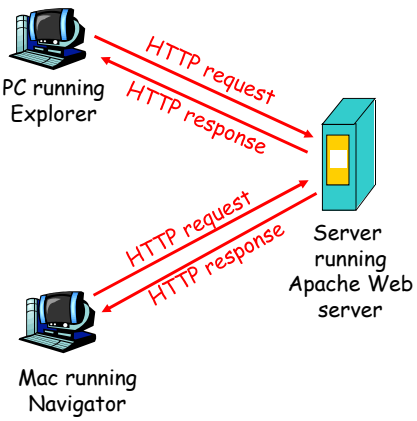
host adı | path adı

19

HTTP genel bakış

HTTP: hypertext transfer protocol

- Web'in application layer protokolü
- client/server model
  - *client:* browser istek yapar, Web nesnelerini görüntüler
  - *server:* Web server nesneleri istek yapanlara gönderir
- HTTP 1.0: RFC 1945
- HTTP 1.1: RFC 2068



20

HTTP genel bakış (devam)

TCP kullanır:

- Client, server ile TCP bağlantısını başlatır (soket oluşturur), port 80
- Server, client'tan gelen TCP bağlantı isteğini kabul eder
- HTTP mesajları (application layer protocol mesajları) browser ile web server arasında değiştirilir.
- TCP bağlantısı kapatılır

HTTP is "stateless"

- server geçmiş client istekleri hakkında bilgi tutmaz

21

HTTP connections

Nonpersistent HTTP

- En fazla bir nesne TCP bağlantısı ile iletilir.
- HTTP/1.0 nonpersistent HTTP kullanır

Persistent HTTP

- Çok sayıda nesne bir TCP bağlantısı ile client ve server arasında gönderilebilir.
- HTTP/1.1 persistent HTTP kullanır

22

Nonpersistent HTTP

Kullanıcı aşağıdaki URL'yi girerse
`www.someSchool.edu/someDepartment/home.index`

(text içerir, 10 jpeg image vardır)

1a. HTTP client, HTTP server ile TCP bağlantıyı başlatır, (www.someSchool.edu, port 80)

1b. www.someSchool.edu 'deki HTTP server TCP bağlantısı bekler, port 80. bağlantı bilgisini client'a gönderir

2. HTTP client, HTTP istek mesajı gönderir (URL) TCP bağlantı soketyle. Mesaj client'ın someDepartment/home.index nesnelerini isteğini gösterir

3. HTTP server istek mesajını alır, istenen nesnelerle cevap mesajını oluşturur ve kendi soketine gönderir

zaman

23

Nonpersistent HTTP - devam

4. HTTP server TCP bağlantısını kapatır.

5. HTTP client html file'dan oluşan cevabı alır ve ekranda gösterir. html file, 10 tane jpeg nesnesi içermektedir

zaman

6. Adım 1-5 her 10 jpeg nesne için tekrarlanır

24

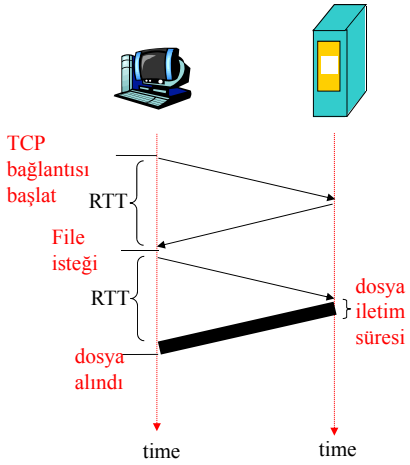
Cevap süresi

RTT tanımı: gönderilen küçük bir paketin client'tan server'a gitmesi ve geri gelmesi için geçen süre

Response time:

- TCP bağlantısını başlatmak için bir RTT
- HTTP isteği ve dönen HTTP cevabı için bir RTT
- Dosya iletim süresi (transmit time)

toplam = 2RTT+transmit time



25

Persistent HTTP

Nonpersistent HTTP :

- Her nesne için 2 RTTs gerekir
- OS her TCP bağlantısı için kaynak ayırır

Persistent HTTP

- server bağlantıyı cevap gönderdikten sonrada açık tutar
- Diğer HTTP mesajları aynı bağlantıdan gönderilir

Persistent without pipelining:

- client önceki cevabı aldıktan sonra yeni bir istek yapar
- Bir RTT süresi her nesne için gerekir

Persistent with pipelining:

- HTTP 1.1 de default
- client bir nesneyle karşılatığında istek gönderir
- Bir RTT süresi tüm nesneler için

26

HTTP request message

- İki tür HTTP mesajı vardır: *request, response*
- HTTP request mesajı:
  - ASCII (okunabilir format)

request line (GET, POST, HEAD komutları)

header lines

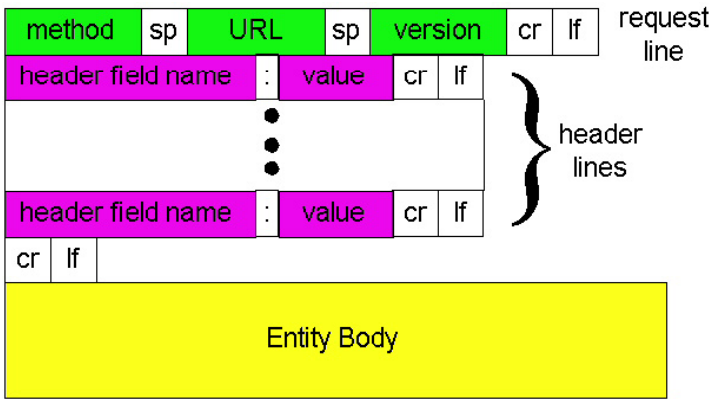
```
GET /somedir/page.html HTTP/1.1
Host: www.someschool.edu
User-agent: Mozilla/4.0
Connection: close
Accept-language:fr
```

return, mesajın sonunu gösterir

(fazla return satır açar)

27

HTTP request message : genel format



28

Form girişlerini gönderme

Post method:

- Web sayfaları genellikle form girişleri bulundurur
- Giriş server'a body içinde upload edilir

URL method:

- GET metodunu kullanır
- Giriş server'a URL içinde upload edilir

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

29

Metod tipleri

HTTP/1.0

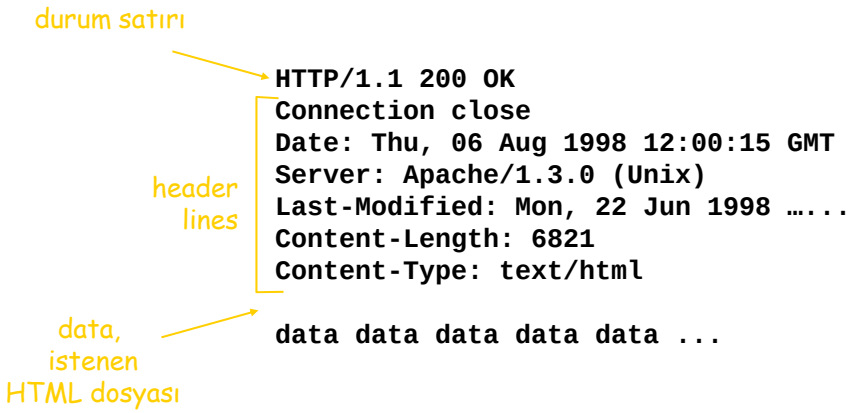
- GET
- POST
- HEAD
  - Server'a istenen nesnenin gönderilip gönderilmediğini sorar

HTTP/1.1

- GET, POST, HEAD
- PUT
  - Dosyayı body içinde URL ile belirlenen yola upload eder
- DELETE
  - URL alanında belirtilen dosyayı siler

30

HTTP cevap mesajı



31

HTTP cevap durum kodları

Server -> client cevap mesajında ilk satır.

Birkaç örnek kod:

200 OK
- İstek başarılı, istenen nesne mesajın sonunda

301 Moved Permanently
- İstenen nesne taşınmış, yeni yer mesajın sonunda (Location)

400 Bad Request
- İstek mesajı server tarafından anlaşılamadı

404 Not Found
- İstenen döküman server üzerinde bulunamadı

505 HTTP Version Not Supported

32

Kullanıcı-sunucu durumları: cookies

Çok sayıda önemli web sitesi cookie kullanır

Dört bileşeni vardır:

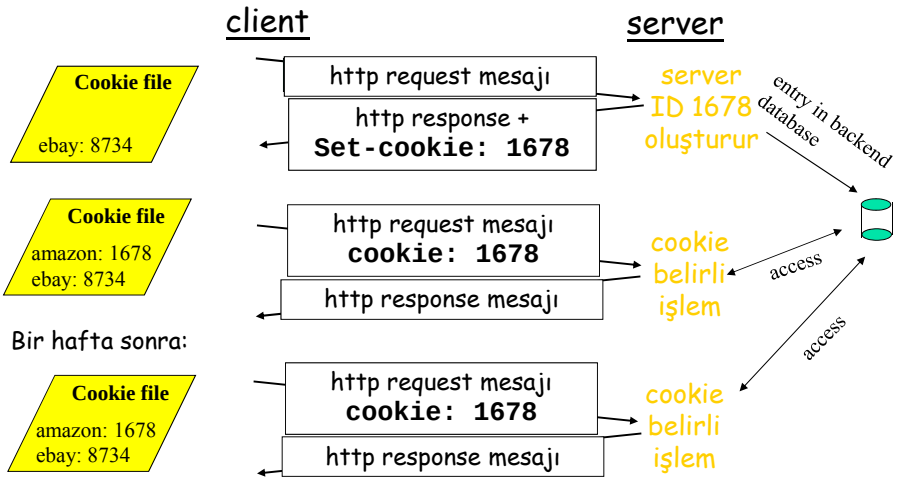
1) cookie başlık satırı (HTTP cevap mesajında)
2) cookie başlık satırı(HTTP istek mesajında)
3) cookie dosyası kullanıcı bilgisayarında saklandı ve kullanıcı browser'ı tarafından kullanılıyor
4) Web sitesinde back-end veritabanı

Örnek:

- Bir kullanıcı hep aynı bilgisayardan Internet'e erişiyor
- Bir e-ticaret sitesini ilk defa ziyaret ediyor
- Başlatma HTTP isteği web sitesine geldiğinde, site bir unique ID oluşturur

33

Cookie'ler



34

Cookie'ler - devam

Cookie'ler ne sağlar

- yetkilendirme
- alışveriş kartları
- öneriler

Cookie'ler ve gizlilik:

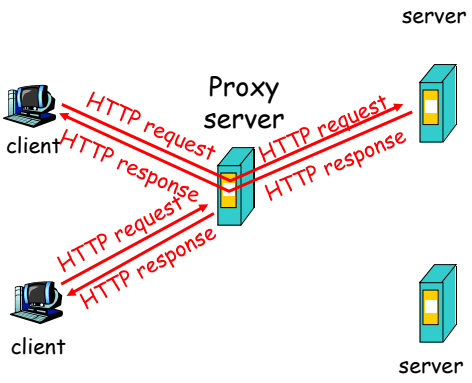
- Cookie'ler web sitesinin sizin hakkınızda birçok şeyi öğrenmesini sağlar
- Web sitesine adınızı ve e-postanızı verebilirsiniz
- Reklam şirketleri bilgi edinebilir

35

Web cache'ler (proxy server)

Amaç: client isteğinin orijinal server kullanılmadan karşılanması

- Web erişimi cache ile olabilir
- browser tüm HTTP isteklerini cahe'e gönderir
  - Nesne cahe'te ise: cache nesneyi döndürür
  - Değilse cache nesneyi orijinal server'dan ister



36

Web caching

- Cache hem client hemde server işlevi görür
- Genellikle cache ISP tarafından kurulur (universite, şirket, bölgesel ISP)

Neden Web caching?

- Client isteklerine cevap süresini azaltır
- Kurumun erişim linkindeki trafiği azaltır
- P2P file sharing uygulamalarında daha etkin içerik sunmayı sağlar

37

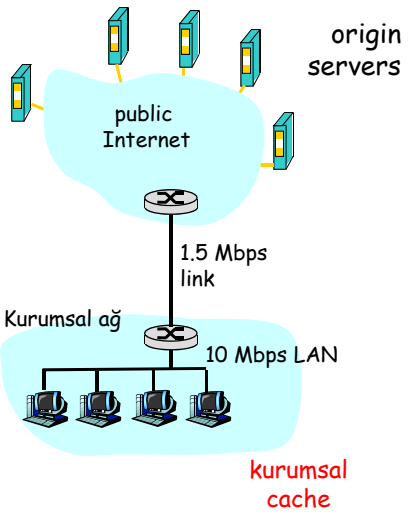
Caching örnek

Varsayımlar

- Ortalama nesne boyutu = 100,000 bit
- Orijinal sunucudan ortalama istek oranı = 15 istek/s
- İnternet gecikmesi: router ile server arasındaki gecikme süresi = 2 s

Sonuçlar

- LAN kullanımı = 15%
- Erişim bağlantısı kullanımı = 100%
- toplam gecikme = İnternet gecikmesi + erişim gecikmesi + LAN gecikmesi = 2 s + dakikalar + milisaniyeler



38

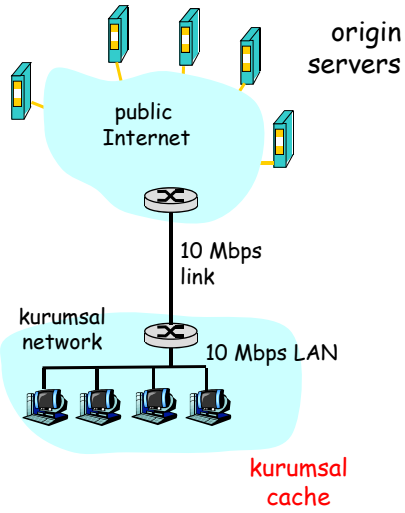
Caching örnek - devam

Muhtemel çözüm

- Erişim bağlantısının bant genişliğini artırmak, 10 Mbps olabilir

Sonuçlar

- LAN kullanımı = 15%
- Erişim bağlantısı kullanımı = 15%
- toplam gecikme = İnternet gecikmesi + erişim gecikmesi + LAN gecikmesi
  = 2 s + dakikalar + milisaniyeler
- Maliyeti yüksek upgrade



39

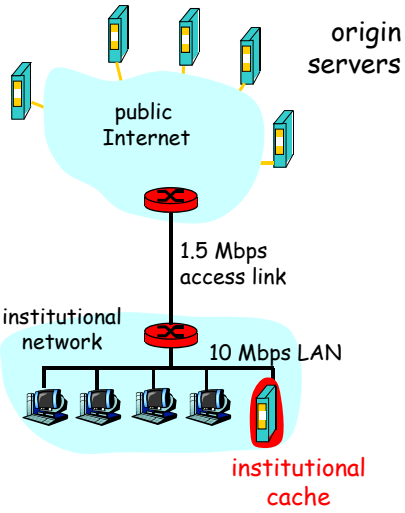
Caching örnek - devam

Install cache

- hit rate değeri 0.4 olsun

Sonuçlar

- 40% istek hemen karşılanabilir
- 60% istek orijinal server tarafından karşılanır
- Erişim bağlantı kullanımı 60% oranına düşer (cache bakmak için geçen ek süre 0,01s).
- Toplam ortalama gecikme = İnternet gecikmesi + erişim gecikmesi + LAN gecikmesi = 0.6 * (2 + 0.01) s + 0.6 * dakikalar + milisaniyeler



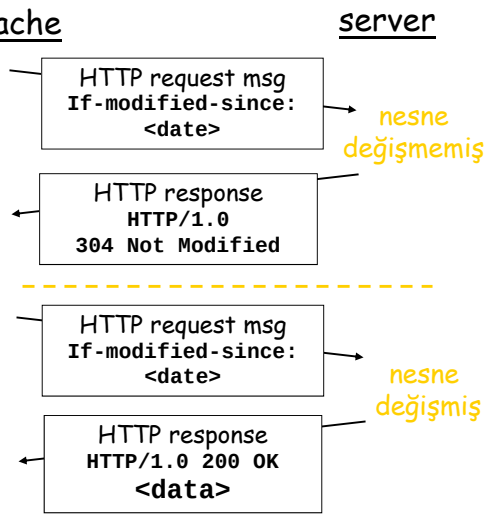
40

Şartlı GET

- Amaç: eğer cahce güncelse nesneyi gönderme
- cache: HTTP isteğindeki cache kopyanın tarihi belirlenir

  `If-modified-since: <date>`
- server: cevap nesne içermez eğer cache güncelse:

  `HTTP/1.0 304 Not Modified`



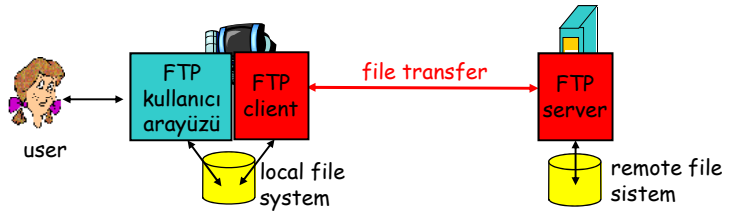
41

Ders konuları

- Ağ uygulamaları
  - Ağ uygulama mimarileri
  - Process'ler arası iletişim
  - Uygulamalar için transport layer hizmetleri
  - İnternet'in sağladığı transport layer hizmetleri
  - Uygulama katmanı protokolleri
- Web ve HTTP
  - HTTP 'nin özellikleri
  - Kalıcı ve kalıcı olmayan bağlantı
  - HTTP mesaj formatı
  - Cookie'ler
  - Web caching
  - Şartlı Get
- FTP
  - FTP komutları ve cevapları

42

FTP: File Transfer Protocol



- Uzaktaki bir host ile dosya transferi yapar
- client/server model
  - *client:* transferi başlatan taraf
  - *server:* remote host
- ftp: RFC 959
- ftp server: port 21

43

FTP: ayrı kontrol ve veri bağlantısı

- FTP client FTP server'a port 21 ile bağlanır, transport protokolünü TCP olarak belirler
- Client yetkilendirmeyi kontrol bağlantısı ile alır
- Client, uzaktaki dizini kontrol bağlantısı üzerinden komutlar gönderek listeler
- Server bir dosya transfer komutu aldığında client'a bir TCP bağlantısı açar
- Bir dosya transferi tamamlandığında, server bağlantıyı kapatır

- Server başka bir dosya için ikinci bir TCP veri bağlantısı açar
- Kontrol bağlantısı: "out of band" durumuna geçer
- FTP server durumunu korur.

44

FTP komutlar, cevaplar

Örnek komutlar:

- Kontrol kanalı üzerinden ASCII metin gönder
- **USER *username***
- **PASS *password***
- **LIST** aktif dizindeki dosyaların listesi alınır
- **RETR filename** dosya alınır(get)
- **STOR filename** uzaktaki bilgisayara dosya gönderilir (put)

Örnek return kodları

- Durum kodu ve deyim (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**

45

Ödev

- C# veya Java programlama diliyle temel işlevlere sahip bir browser geliştiriniz.

46