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

- E-Mail
  - SMTP
  - SMTP ve HTTP
  - Mail mesaj formatları
  - Mail erişim protokolleri : POP3, IMAP
- DNS
  - DNS servisleri
  - DNS'in çalışması
- Peer-to-Peer Uygulamalar
  - P2P dosya dağıtımı
- TCP ile soket programlama
- UDP ile soket programlama

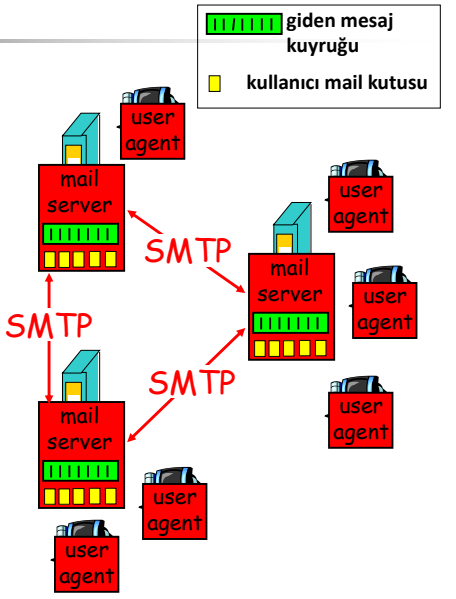
Elektronik posta

Üç ana bileşen vardır:

- user agents (kullanıcı arayüzleri)
- mail server'lar
- simple mail transfer protocol: SMTP

User Agent

- posta okuyucu
- Mesaj oluşturma, düzenleme, okuma
- Outlook, Gmail app, Netscape Messenger
- Giden ve gelen mesajlar server'da saklanır

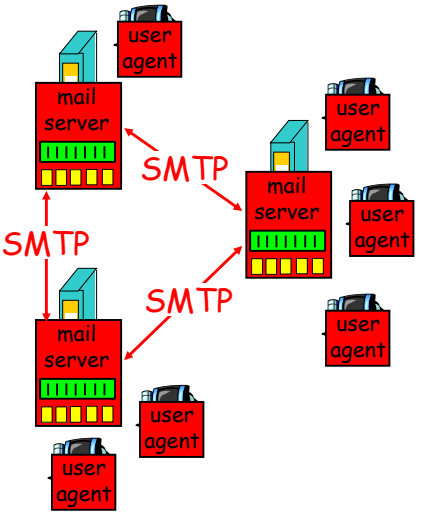


3/101

Elektronik posta: mail server'lar

Mail Server'lar

- mailbox kullanıcı için gelen mesajları saklar
- message queue gönderilen mesajları tutar
- SMTP protocol mail server'lar arasında mesaj göndermek için kullanılır
  - client: gönderen mail sunucu
  - "server": alan mail sunucu



4/101

Ders konuları

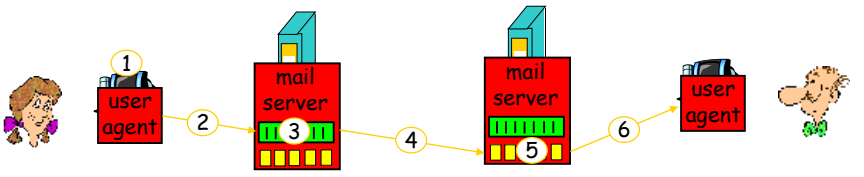
- E-Mail
  - SMTP
  - SMTP ve HTTP
  - Mail mesaj formatları
  - Mail erişim protokolleri : POP3, IMAP
- DNS
  - DNS servisleri
  - DNS'in çalışması
- Peer-to-Peer Uygulamalar
  - P2P dosya dağıtımı
- TCP ile soket programlama
- UDP ile soket programlama

Elektronik posta: SMTP [RFC 2821]

- Client'tan Server'a e-mail mesajını güvenilir göndermek için TCP kullanır, port 25
- direct transfer: gönderen server'dan alan server'a
- Üç aşamalı transfer
  - handshaking
  - mesajların transferi
  - kapatma
- command/response etkileşimi
  - commands: ASCII metin
  - response: durum kodları ve deyimler

Elektronik posta: posta gönderme

1) A, user agent kullanarak bir mesaj hazırlar `aaa@gazi.edu.tr`
2) A'nın user agent programı mesajı kendi mail sunucusuna gönderir, mesaj kuyruğa atılır
3) SMTP'de client taraftaki mail sunucu bir TCP bağlantısı açar B kullanıcısının mail server'ına
4) SMTP client A'nın mesajını TCP bağlantısı üzerinden gönderir
5) B'nin mail server'ı gelen mesajı B'nin mail kutusuna yerleştirir
6) B kendi user agent programı ile mesajı okur



Ders konuları

- E-Mail
  - SMTP
  - SMTP ve HTTP
  - Mail mesaj formatları
  - Mail erişim protokolleri : POP3, IMAP
- DNS
  - DNS servisleri
  - DNS'in çalışması
- Peer-to-Peer Uygulamalar
  - P2P dosya dağıtımı
- TCP ile soket programlama
- UDP ile soket programlama

SMTP: final words

- SMTP persistent connection kullanır
- SMTP 7-bit ASCII mesaj gerektirir (header & body)
- SMTP server `CRLF.CRLF` ile mesaj sonunu belirler

HTTP ile karşılaştırma:

- HTTP: pull
- SMTP: push
- İkisi de ASCII command/response etkileşimi, durum kodları
- HTTP: her nesne kendi cevap mesajına encapsulate edilir.
- SMTP: bir mesajla çok sayıda nesne gönderilir.

Ders konuları

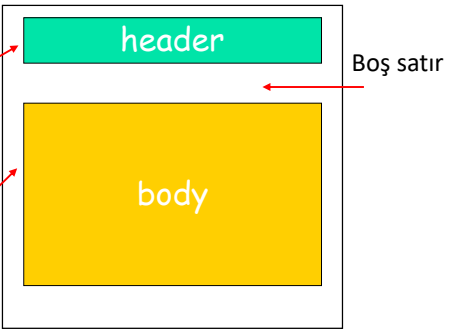
- E-Mail
  - SMTP
  - SMTP ve HTTP
  - Mail mesaj formatları
  - Mail erişim protokolleri : POP3, IMAP
- DNS
  - DNS servisleri
  - DNS'in çalışması
- Peer-to-Peer Uygulamalar
  - P2P dosya dağıtımı
- TCP ile soket programlama
- UDP ile soket programlama

Mail mesaj formatı

SMTP: e-mail mesajlarını gönderen/alan protokol

RFC 822: metin mesaj formatı

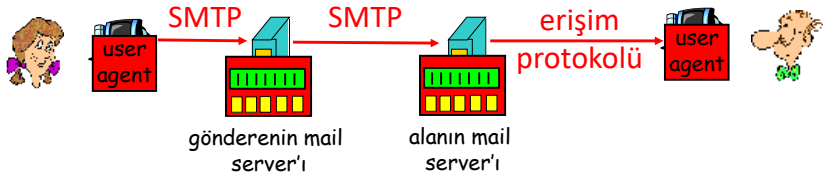
- header lines,
  - To:
  - From:
  - Subject:
- body
  - mesaj, ASCII karakterler



Ders konuları

- E-Mail
  - SMTP
  - SMTP ve HTTP
  - Mail mesaj formatları
  - Mail erişim protokolleri : POP3, IMAP
- DNS
  - DNS servisleri
  - DNS'in çalışması
- Peer-to-Peer Uygulamalar
  - P2P dosya dağıtımı
- TCP ile soket programlama
- UDP ile soket programlama

Mail erişim protokolleri



- SMTP: Alıcı sunucuda gönderme ve saklama
- Mail access protocol: sunucudan alma işlemini yapar
  - POP: Post Office Protocol [RFC 1939]
    - authorization (agent <-->server) ve download
  - IMAP: Internet Mail Access Protocol [RFC 1730]
    - daha fazla özellik (daha fazla karmaşık)
    - sunucuda saklanmış mesajlar üzerinde işlem
  - HTTP: Hotmail , Yahoo! Mail, v.b.

POP3 protocol

authorization aşaması

- client komutları:
  - **`user:`** username tanımla
  - **`pass:`** password
- server cevapları
  - **`+OK`**
  - **`-ERR`**

transaction aşaması, client:

- **`list:`** mesaj numaralarını listele
- **`retr:`** numara ile mesaj al
- **`dele:`** delete
- **`quit`**

```
S: +OK POP3 server ready
C: user bob
S: +OK
C: pass hungry
S: +OK user successfully logged on
```

```
C: list
S: 1 498
S: 2 912
S: .
C: retr 1
S: <message 1 contents>
S: .
C: dele 1
C: retr 2
S: <message 1 contents>
S: .
C: dele 2
C: quit
S: +OK POP3 server signing off
```

POP3 ve IMAP

POP3

- Önceki örnek “download ve delete” modlarını göstermektedir.
- Bir kullanıcı client’ı değiştirmişse mesaj okuyamaz

IMAP

- Tüm mesajlar aynı sunucuda tutulur
- Kullanıcıya dizinlerdeki mesajlarını organize etme izni verir
- IMAP kullanıcı durumunu oturum sonuna kadar saklar:
  - Dizin adları ve mesaj ID ile dizin eşleştirme yapar

Ders konuları

- E-Mail
  - SMTP
  - SMTP ve HTTP
  - Mail mesaj formatları
  - Mail erişim protokolleri : POP3, IMAP
- DNS
  - DNS servisleri
  - DNS’in çalışması
- Peer-to-Peer Uygulamalar
  - P2P dosya dağıtımı
- TCP ile soket programlama
- UDP ile soket programlama

DNS: Domain Name System

İnsanlar için: çok sayıda tanımlayıcı vardır:

- TCKimlik, Vergi No, Pasaport No

Internet host'ları, router'lar:

- IP adres (32 bit) – datagram adreslemek için kullanılır
- "ad", örn., www.yahoo.com – insanlar kullanır

IP adresleri ile adlar arasında eşleme gerekmektedir.

Domain Name System:

- *distributed database* bir çok *DNS server* ile hiyerarşik olarak oluşturulmuştur.
- *application-layer protocol* host, router, name server'lar adres/ad dönüşümünü yaparak adları elde eder.
  - Internet temel bir fonksiyonu application-layer protokolü ile gerçekleştirilir.
  - Ağ uç birimleri karmaşıktır.

Ders konuları

- E-Mail
  - SMTP
  - SMTP ve HTTP
  - Mail mesaj formatları
  - Mail erişim protokolleri : POP3, IMAP
  - Web tabanlı e-mail
- DNS
  - DNS servisleri
  - DNS'in çalışması
- Peer-to-Peer Uygulamalar
  - P2P dosya dağıtımı
- TCP ile soket programlama
- UDP ile soket programlama

DNS

DNS servisleri

- Hostname ile IP adres dönüşümü
- Host aliasing (takma isim)
  - Canonical ve alias name
- Mail server aliasing
- Load distribution (yük dağıtımı)
  - Bir isim için birden fazla sunucu farklı IP numaralarıyla çalışır ve sırayla işler dağıtılır

DNS neden merkezi değildir?

- Tek noktadaki hata
- Trafik yoğunluğu
- distant centralized database
- maintenance

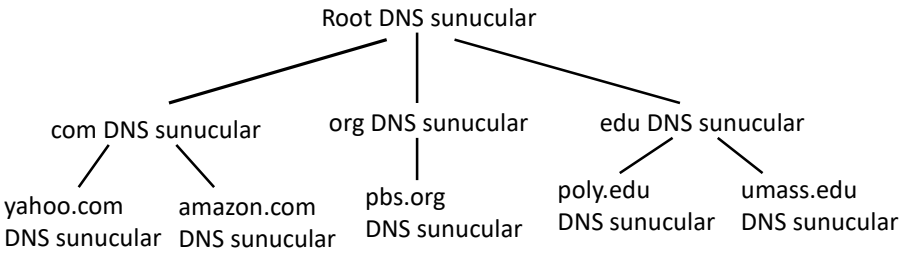
DNS

- UDP kullanır
- Port 53

Ders konuları

- E-Mail
  - SMTP
  - SMTP ve HTTP
  - Mail mesaj formatları
  - Mail erişim protokolleri : POP3, IMAP
  - Web tabanlı e-mail
- DNS
  - DNS servisleri
  - DNS'in çalışması
- Peer-to-Peer Uygulamalar
  - P2P dosya dağıtımı
- TCP ile soket programlama
- UDP ile soket programlama

Dağıtık, Hiyerarşik Veritabanı



Client www.amazon.com için IP adresini istemektedir.

- Client, com DNS server'ı bulmak için a root server'ı sorgular
- Client, amazon.com DNS server için com DNS server'ı sorgular
- Client, www.amazon.com için IP adresi almak için amazon.com DNS server'ı sorgular

DNS: Root name server'lar

- Çözümlenemeyen isimler için local name server'la iletişim yapılır
- root name server:
  - İsim eşleşmesi bilinmiyorsa bağlantıya geçilir
  - Eşleşme alınır
  - Eşleşme local name server'a gönderilir

dünyada 13 root name server vardır

TLD ve Authoritative Server'lar

- Top-level domain (TLD) server'lar: com, org, net, edu, vb. ile en üst seviye ülke domainleri uk, fr, ca, jp vb. den sorumludur.
  - Ağ çözümleri com TLD için sunucu bulundurur
  - Eğitim sunucuları edu TLD bulundurur
- Authoritative DNS server'lar: kurumların DNS server'ları, hostname ile IP eşleşmesi için kullanılır (örn., Web ve mail).
  - Kurum veya ISP tarafından oluşturulur

Local Name Server

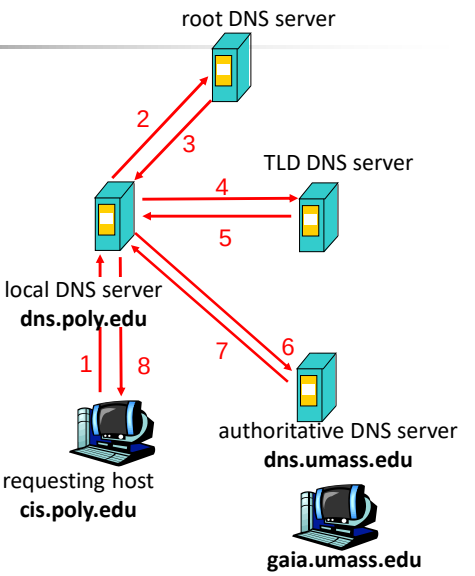
- Her ISP (şirket, üniversite) bir tane Local Name Server'a sahiptir.
  - "default name server" olarak adlandırılır
- Bir host DNS sorgu yaptığında, sorgu kendi local DNS server'ına gönderilir
  - Proxy olarak çalışır, sorgu hiyerarşide yönlendirilir.

Örnek

- cis.poly.edu üzerindeki host, gaia.cs.umass.edu için IP adresi istemektedir

iterated query:

- Tüm sorguyu lokal DNS sunucu yapar.

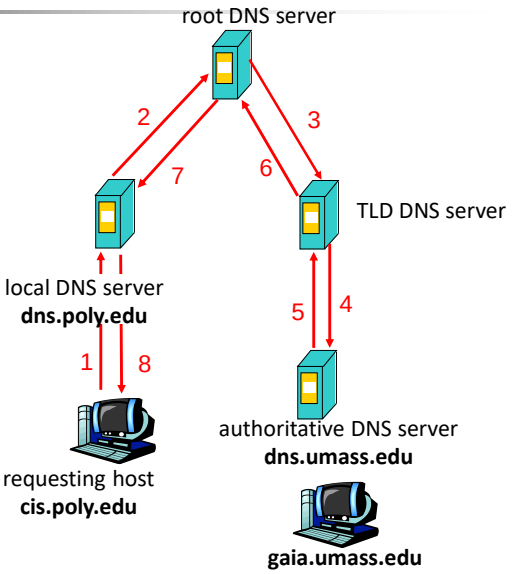


25/101

Özyinelemeli (recursive) sorgular

recursive query:

- Tüm sorgu recursive gerçekleşir.



26/101

DNS: kayıtları cache'e yazmak ve update etmek

- name server eşleşmeyi öğrendiğinde, kendi cache'ine aktarır
  - Cache'ten belirli süre sonunda atılır
  - TLD server'lar local name server'larda cache'lenir
    - Böylece root name server'lar sıklıkla ziyaret edilmezler
- update/notify mekanizmaları IETF tarafından tasarlanmaktadır
  - RFC 2136

DNS kayıtları

DNS: distributed db storing resource records (RR)

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

- Type=A
  - **`name`**, hostname
  - **`value`**, IP adres
  - `(relay1.bar.foo.com, 145.37.93.126, A)`
- Type=NS
  - **`name`**, domain (örn. foo.com)
  - **`value`**, bu domain için authoritative name server'ın IP adresi
  - `(foo.com, dns.foo.com, NS)`
- Type=CNAME
  - **`value`** canonical isimdir, **`name`** takma isimdir
  - `(foo.com, relay1.bar.foo.com, CNAME)`
- Type=MX
  - **`value`**, eposta sunucunun canonical ismidir, **`name`** takma isimdir
  - `(foo.com, mail.bar.foo.com, MX)`

DNS içine kayıtların eklenmesi

- Örnek: “Network Utopia” oluşturuldu
- Kayıt adı networkuptopia.com olur (registrar’da (örn., Network Solutions))
  - Registrar’lar adları ve IP adresleri ile authoritative name server’a sağlanır
  - Registrar iki RRs kaydını com TLD server’a saklar:

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

- authoritative DNS server’a www.networkuptopia.com için Type A kayıt ve networkutopia.com için Type MX kayıt eklenir

```
(www.networkutopia.com, 212.212.212.4, A)
(networkutopia.com, mail.networkutopia.com, MX)
```


Ders konuları

- E-Mail
  - SMTP
  - SMTP ve HTTP
  - Mail mesaj formatları
  - Mail erişim protokolleri : POP3, IMAP
  - Web tabanlı e-mail
- DNS
  - DNS servisleri
  - DNS’in çalışması
- Peer-to-Peer Uygulamalar
  - P2P dosya dağıtımı
- TCP ile soket programlama
- UDP ile soket programlama

P2P dosya paylaşımı

Örnek

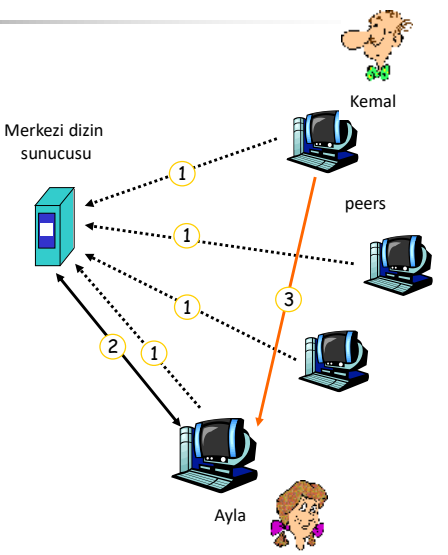
- Ayla P2P client uygulamasını kendi notebook'u üzerinde çalıştırmaktadır.
- Internet'e bağlanır; her bağlantı için bir IP adres alınır.
- Jale'ye "Merhaba Jale" gönderir.
- Uygulama diğer peer'da Merhaba Jale'yi görüntüler.
- Ayla bir peer seçer, Kemal.
- Dosya Kemal'in bilgisayarından Ayla'nın notebook'una aktarılır: HTTP
- Ayla download yaparken, diğer kullanıcılar Ayla'dan upload yapabilir.
- Ayla hem Web client hem de Web server olarak çalışır.

Tüm peer'ların server olmasından dolayı yüksek ölçeklenebilir.

P2P: merkezi dizin

Orijinal "Napster" tasarımı

1) Bir peer bağlandığında, merkezi server'ı bilgilendirir:
   - IP adres
   - içerik
2) Ayla istediği sorguyu gönderir
3) Ayla, Kemal'den istediği dosyayı aktarır



P2P: merkezi dizindeki problemler

- Hatanın tek noktada olması
- Performans yoğunluktan etkilenir.
- Copyright infringement

Query flooding: Gnutella

- Tam dağıtıktır
  - Merkezi server yoktur.
- Query routing protokol ile Query routing table aktarılır.
- Çok sayıda Gnutella client vardır ve ilgili protokolü bulundurur.
- Paket türleri:
  - **Ping**: ağdaki kullanıcıların bulunması
  - **Pong**: ping cevabı
  - **Query**: dosya arama
  - **Query hit**: sorgu cevabı
  - **Push**: dosya download

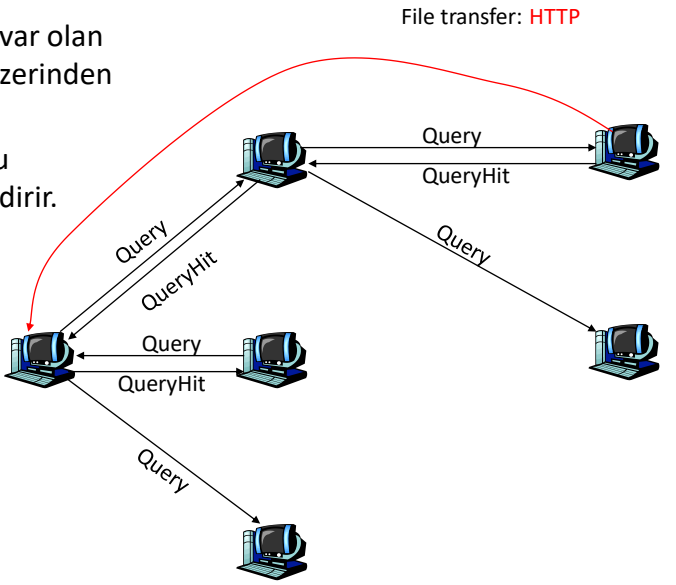
Ağ yapısı : graph

- X ve Y arasında TCP bağlantısı varsa kenar çizilir
- Tüm aktif peer'lar ve kenarlar ağı oluşturur
- Kenar fiziksel bağlantı değildir
- Bir peer genel olarak < 10 komşuya bağlıdır

Gnutella: protokol

- Sorgu mesajı var olan TCP bağlantısı üzerinden gönderilir.
- peer'lar sorgu mesajını yönlendirir.
- QueryHit aynı yoldan ters yönden gönderilir.

Scalability:
limited scope flooding

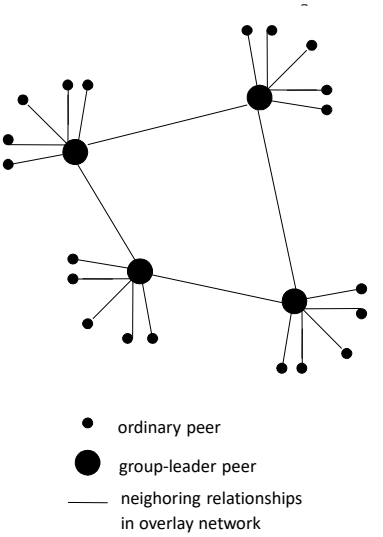


Gnutella: Peer joining

1. peer X, Gnutella ağında başka bir peer bulmak zorundadır.
2. X, Y ile bağlantı kurana kadar ard arda TCP bağlantısı yapmaya çalışır.
3. X Ping mesajını Y'ye gönderir; Y Ping mesajını yönlendirir.
4. Tüm peer'lar Ping mesajını alır ve Pong mesajını gönderir.
5. X çok sayıda Pong mesajını alır. Çok sayıda TCP bağlantısı yapar.

Heterojen yapı: KaZaA

- Her peer bir grup lideridir veya bir grup liderine atanır.
  - Peer'lar arasında TCP bağlantısı
  - Grup liderleri arasında TCP bağlantısı.
- Grup lideri tüm bağlı peer'ların içeriğini izler.



37/101

KaZaA: Querying

- Client sorguyu kendi grup liderine gönderir.
- Grup lideri uyan kayıtlarla geri döner.
- Eğer bir grup lideri sorguyu diğer grup liderlerine yönlendirirse, onlar uyan kayıtları döndürür.
- Client download için dosyaları seçer.

38/101

Kazaa özellikleri

- Eşzamanlı sınırlı upload
- Request kuyruğu
- Incentive önceliklendirme
- Paralel download

39/101

Ders konuları

- E-Mail
  - SMTP
  - SMTP ve HTTP
  - Mail mesaj formatları
  - Mail erişim protokolleri : POP3, IMAP
  - Web tabanlı e-mail
- DNS
  - DNS servisleri
  - DNS'in çalışması
- Peer-to-Peer Uygulamalar
  - P2P dosya dağıtımı
- TCP ile soket programlama
- UDP ile soket programlama

40/101

Soket programlama

Amaç: client/server uygulamaların soket kullanılarak haberleşmesi

Soket API

- 1981 yılında BSD4.1 UNIX ile önerildi
- Uygulamalar tarafından doğrudan oluşturulur, kullanılır ve yayınlanır.
- client/server yaklaşımı
- Soket API ile iki tür transport servisi:
  - unreliable datagram
  - reliable, byte stream-oriented

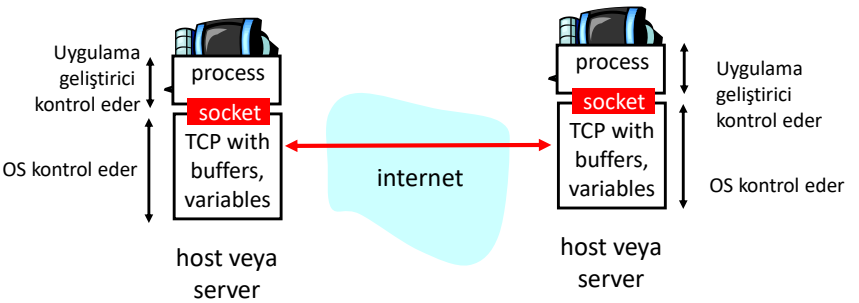
socket

Bir *host-local*, *application-created*, *OS-controlled* arayüz (bir "kapı"). Uygulama işlemleri (process) arasında mesaj gönderme ve alma yapılır.

TCP kullanarak soket programlama

Soket: uygulama prosesleri ve end-to-end transport protokolü (TCP veya UDP) arasında

TCP servisi: bir prosesten diğerine reliable **byte** transferi



TCP ile soket programlama

Client server'la bağlantı sağlar

- Server proses çalışır olmalıdır.
- Server, client'ın bağlantısı için bir soket oluşturur.

- Client lokal TCP soket oluşturur.
- IP adres belirlenir ve port numarası server proses için belirlenir.
- Client TCP ile server TCP arasında bağlantı oluşturulmuş olur.

- Başka bir client server'a bağlanmak isterse,
  - Server birden çok client ile bağlantı sağlar.
  - Kaynak port numaraları client'ları birbirinden ayırır.

application viewpoint

TCP client ile server güvenilir ve sıralı byte transferi sağlayan bir bağlantı sağlar

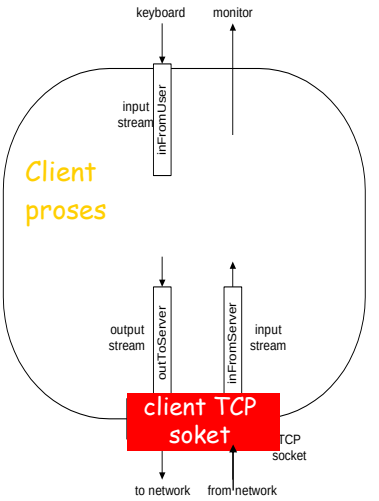
Stream kavramı

- Bir stream bir prosese gelen veya prosesten çıkan karakterler kümesidir
- input stream bir proses için giriş sağlayan kaynaktır, örn., keyboard veya soket.
- output stream bir prosesten çıkış alan kaynaktır, örn., monitor veya soket.

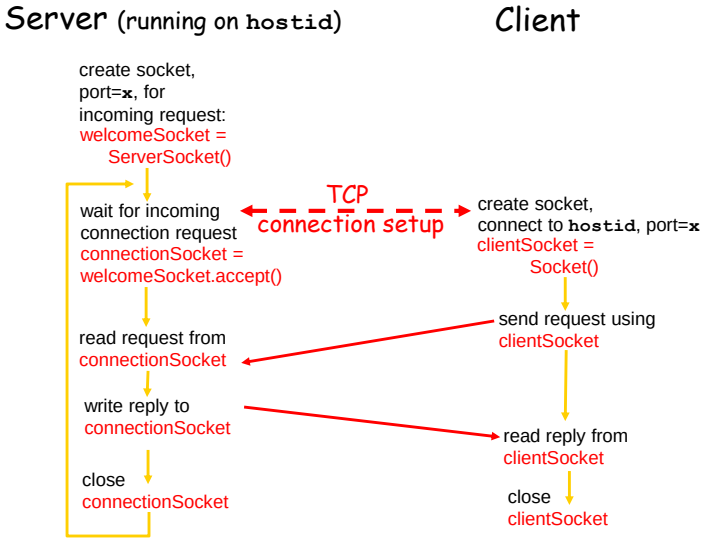
TCP ile soket programlama

Örnek client-server uygulama:

1) client girişten bir satır okur (**`inFromUser`** stream) , bir soket ile sunucuya gönderir (**`outToServer`** stream)
2) server soket ile satırı okur
3) server satırı büyük harfe dönüştürür ve client'a geri gönderir
4) client soket ile okur, yazdırır ve değişiklik yapar (**`inFromServer`** stream)



Client/server soket etkileşimi: TCP



Örnek: Java client (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

47/101

Örnek: Java client (TCP) - devam

```
      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();

   }
}
```

Create input stream attached to socket

Send line to server

Read line from server

48/101

Örnek: Java server (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 server (TCP) - devam

```
DataOutputStream  outToClient =
new DataOutputStream(connectionSocket.getOutputStream());

clientSentence = inFromClient.readLine();

capitalizedSentence = clientSentence.toUpperCase() + '\n';

outToClient.writeBytes(capitalizedSentence);
        }
    }
}
```

Create output stream, attached to socket

Read in line from socket

Write out line to socket

End of while loop, loop back and wait for another client connection

Ders konuları

- E-Mail
  - SMTP
  - SMTP ve HTTP
  - Mail mesaj formatları
  - Mail erişim protokolleri : POP3, IMAP
  - Web tabanlı e-mail
- DNS
  - DNS servisleri
  - DNS'in çalışması
- Peer-to-Peer Uygulamalar
  - P2P dosya dağıtımı
- TCP ile soket programlama
- UDP ile soket programlama

UDP ile soket programlama

UDP: client ve server arasında bağlantı yapılmaz

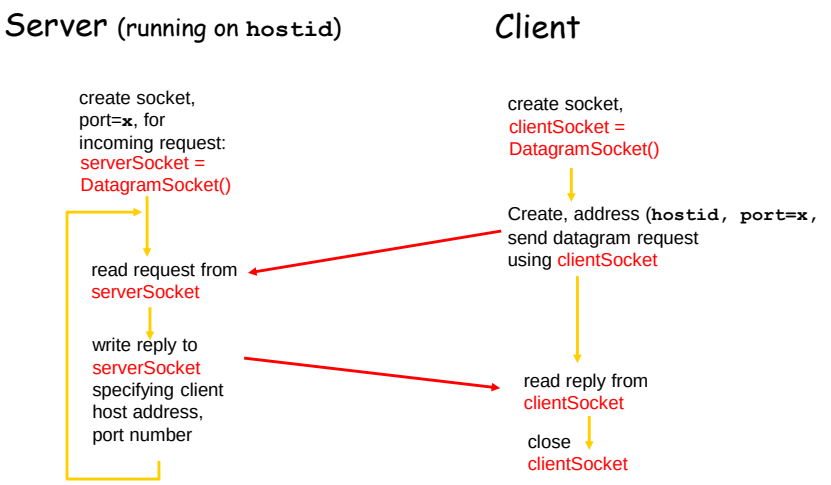
- Handshake yapılmaz
- Gönderen hedef IP adres ve port numarasını pakete ekler
- server IP adresi ve gönderenin port numarasını algılar

UDP: iletilen data sırasız gidebilir veya kaybolabilir

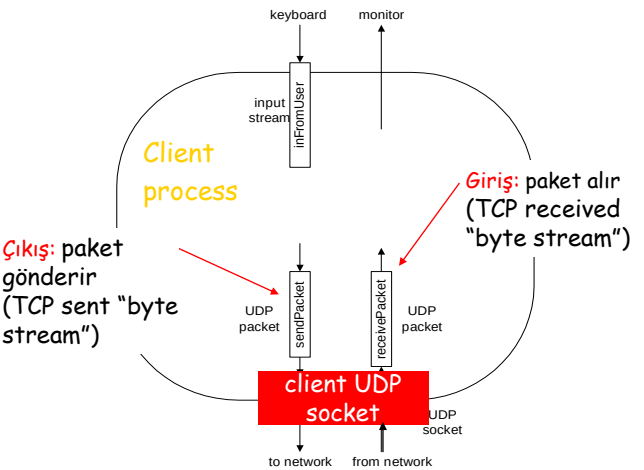
application viewpoint

UDP, cient ile server arasında byte gruplarının güvenilir olmayan transferini yapar

Client/server soket etkileşimi: UDP



Örnek: Java client (UDP)



Örnek: Java client (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 client (UDP) - devam

```
      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();
      }
}
```

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

Send datagram to server

Read datagram from server

Örnek: Java server (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 server (UDP) - devam

```
        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

Ödev

TCP protokolünü kullanarak Java programlama diliyle anlık mesajlaşma uygulaması geliştiriniz.

Aşağıdaki özelliklere sahip olacak:

- Gelen mesajların kullanıcı bazında kaydı tutulacak.
- Geçmiş mesaj bilgilerine erişilebilecek.
- Kullanıcıların bağlantı listesi oluşturulacak.
- Liste üzerinde gruplandırmalar (arkadaşlarım, ailem, diğer, v.b.) yapılabilecek.
- Geçmiş mesajlar üzerinde anahtar kelimeyle arama yapılabilecek.