Chapter 3
Application Layer Functionality and Protocols
Applications: Interface Between the Networks
The Application layer, Layer seven, is the top layer of both the OSI and TCP/IP models.

- Provides the interface between the applications we use to communicate and the underlying network.
Application layer protocols are used to exchange data between programs running on the source and destination hosts.

There are many Application layer protocols and new protocols are always being developed.
Functionality of the TCP/IP application layer protocols fit roughly into the framework of the top three layers of the:
- OSI model: Application, Presentation and Session layers.

Most early TCP/IP application layer protocols were developed before the emergence of:
- personal computers, graphical user interfaces and multimedia objects.

These protocols implement very little of the functionality that is specified in the OSI model Presentation and Session layers.
The Presentation layer has three primary functions:

- **Coding and conversion** of Application layer data to ensure that data from the source device can be interpreted by destination device.
- **Compression** of the data in a manner that can be decompressed by the destination device.
- **Encryption** of the data for transmission and the decryption of data upon receipt by the destination.

Compression and Coding formats:

- Graphics Interchange Format (GIF)
- Joint Photographic Experts Group (JPEG)
- Tagged Image File Format (TIFF).
The Session Layer

- Create and maintain dialogs between source and destination applications.
- Handles the exchange of information to:
  - initiate dialogs
  - keep them active
  - restart sessions that are disrupted or idle for a long period of time

Most applications, like web browsers or e-mail clients, incorporate functionality of the OSI layers 5, 6 and 7.
Application Layer: OSI and TCP/IP Models

Note: Usually a single server will function as a server for multiple applications.

- **Common TCP/IP Protocols**
  - **Domain Name Service Protocol (DNS)** is used to resolve Internet names to IP addresses.
  - **Hypertext Transfer Protocol (HTTP)** is used to transfer files that make up the Web pages of the World Wide Web.
  - **Simple Mail Transfer Protocol (SMTP)** is used for the transfer of mail messages and attachments.
  - **Telnet**, a terminal emulation protocol, is used to provide remote access to servers and networking devices.
  - **File Transfer Protocol (FTP)** is used for interactive file transfer between systems.
RFCs: Request For Comments

The protocols in the TCP/IP suite are generally defined by Requests for Comments (RFCs).
  - Maintained by IETF (Internet Engineering Task Force)
Application Layer Software

- Within the Application layer, there are two forms of software programs or processes that provide access to the network:
  - applications
  - services

- Network-Aware Applications
  - Applications are the software programs used by people to communicate over the network.
    - Email Clients
    - Web Browsers

- Application layer Services
  - These services are the programs that interface with the network and prepare the data for transfer.
    - File transfer
    - Network print spooling
**Application Layer Software**

- **Application layer** uses protocols that are implemented within applications and services.
  - **Applications** provide people a way to create messages.
  - Application layer **services** establish an interface to the network.
  - **Protocols** provide the rules and formats that govern how data is treated.

- **Bottom line:**
  - When discussing an application like "Telnet" we could be referring to the application, the service, or the protocol.
Application Layer Protocol Functions

Application layer protocols are used by both the source and destination devices during a communication session.

The application layer protocols implemented on the source and destination host must match.

**Protocols:**
- Establish consistent rules for exchanging data.
- Specify the structure and type of messages that are exchanged.
  - Types: Request, response, acknowledgement, error message, etc.
Application Layer Protocol Functions

- Applications and services can use multiple protocols.
  - Encapsulate the protocol or encapsulated by this protocol
  - Invoke other protocols

- Using a web browser (HTTP):
  - May invoke:
    - DNS, ARP, ICMP
  - May use:
    - TCP, UDP, Ethernet, PPP
  - Uses
    - IP
Client Server Model

- **Client**: the device requesting the information
- **Server**: the device responding to the request is called a server.
- The client begins the exchange by requesting data from the server.
- Server responds by sending one or more streams of data to the client.
- In addition to the actual data transfer, this exchange may also require control information, such as:
  - user authentication
  - the identification of a data file to be transferred
Servers

- A server is usually a computer that contains information to be shared with many client systems.
  - Web server
  - Email server
  - File or database server
  - Applications server
- Some servers may require authentication of user account information and vary permissions.
Servers

- The server runs a service, or process, sometimes called a server **daemon**.
- **Daemons** (like other services) typically run in the background and are not under an end user's direct control.
- Daemons are described as "**listening**" for a request from a client.
  - Programmed to respond whenever the server receives a request for the service provided by the daemon.
- When a daemon "**hears**" a request from a client:
  - It exchanges appropriate messages with the client, as required by its protocol,
  - Proceeds to send the requested data to the client in the proper format.
Application Layer
Services and Protocols

- Servers typically have multiple clients requesting information at the same time.

- For example, a Telnet server may have many clients requesting connections to it.
  - These individual client requests must be handled simultaneously and separately for the network to succeed.
  - The Application layer processes and services rely on support from lower layer functions to successfully manage the multiple conversations.
Application Layer Protocols
HTTP
(WWW)

FTP
(file transfer)

SMTP
(email)

Telnet
(remote login)

DHCP
(IP address resolution)

DNS
(domain name resolution)

SMB
(file sharing)

P2P
(file sharing)
Reminder of encapsulation/decapsulation

<table>
<thead>
<tr>
<th>Data Link Header</th>
<th>IP Header</th>
<th>TCP Header</th>
<th>HTTP Header</th>
<th>Data</th>
<th>Data Link Trailer</th>
</tr>
</thead>
</table>

Data Link Header | IP Packet | Data Link Trailer

Data Link Header | IP Packet | Data Link Trailer

Data Link Header | IP Packet | Data Link Trailer

Data

Data Link

Data Link

Data Link

Data

Data Link

Data Link

Data Link

Data
HTTP (HyperText Transfer Protocol)

- HTTP – The Web’s application layer protocol.
- RFC 1945 and RFC 2616
- Implemented in:
  - Client program
  - Server program
- Current version: HTTP/1.1
- Encapsulated in TCP
HTTP (HyperText Transfer Protocol)

- **Web page** (also called a html document)
- Web page consists of **objects**
  - Objects (examples):
    - HTML file
    - JPEG image
    - GIF image
    - JAVA applet
    - Audio file

```
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title>Rick Graziani, Cabrillo College</title><style type="text/css">
 <!--
body {
  margin-left: 0px;
  margin-top: 0px;
  margin-right: 0px;
  margin-bottom: 0px;

  The base HTML file references other objects in the page.
```
Web Browser - Client

- **Browser** – The user agent for the Web.
  - Displays requested Web page and provides navigational and configuration features.

- **Browser and client** may be used *interchangeably* in this discussion.

- HTTP has nothing to do with how a Web page is interpreted (displayed) by the client (browser).
- **Web Server** – Stores web objects, each addressable by a URL.
- Implement the server side of HTTP.
- Examples:
  - Apache
  - Microsoft Internet Information Server
HTTP Request Message

**Request line**: Method field
- GET, POST and HEAD
  - The great majority of Requests are GETs

**Header lines**
- Accept-Language: en-us
- User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 6.0; SLCC1; .NET CLR 2.0.50727; Media Center PC 5.0; .NET CLR 3.0.04506; InfoPath.1)
- Host: www.cabrillo.edu
- Connection: Keep-Alive

**ASCII Text**

```
GET /~rgraziani/ HTTP/1.1  
Accept-Language: en-us   
User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 6.0; SLCC1; .NET CLR 2.0.50727; Media Center PC 5.0; .NET CLR 3.0.04506; InfoPath.1)
Host: www.cabrillo.edu
Connection: Keep-Alive
```
HTTP Request Message

GET /~rgraziani/ HTTP/1.1
Accept-Language: en-us
User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 6.0; SLCC1; .NET CLR 2.0.50727; Media Center PC 5.0; .NET CLR 3.0.04506; InfoPath.1)
Host: www.cabrillo.edu
Connection: Keep-Alive

Request Line

GET     - Browser/client is requesting an object
/~rgraziani/  - Browser is requesting this object in this directory (default is index.html)
HTTP/1.1  - Browser implements the HTTP/1.1 (1.1 is backwards compatible with 1.0)

Note: HTTP GET is also used by some P2P applications like Gnutella and Bittorrent.
HTTP Request Message

GET /~rgraziani/ HTTP/1.1
Accept-Language: en-us
User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 6.0; SLCC1; .NET CLR 2.0.50727; Media Center PC 5.0; .NET CLR 3.0.04506; InfoPath.1)
Host: www.cabrillo.edu
Connection: Keep-Alive

Request Line

GET: - Used by browser/client to request an object.
POST: - Used when user has filled out a form and sending information to the server. (Forms do not have to use POST.)
        - Example: words in a search engine
HEAD: - Similar to a GET, but the server will responds with a HTTP message but leaves out the requested object.
PUT:  - Used with Web publishing tools, upload objects.
DELETE: - Used with Web publishing tools, delete objects.
HTTP Response Message

HTTP/1.1 200 OK
Date: Fri, 22 Feb 2008 16:34:18 GMT
Server: Apache/2.0.52 (Red Hat)
Last-Modified: Thu, 15 Nov 2007 19:33:12 GMT
Content-Length: 15137
Connection: close
Content-Type: text/html

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">

Some data omitted for brevity
HTTP Response Message

HTTP/1.1 200 OK
Date: Fri, 22 Feb 2008 16:34:18 GMT
Server: Apache/2.0.52 (Red Hat)
Last-Modified: Thu, 15 Nov 2007 19:33:12 GMT
Content-Length: 15137
Connection: close
Content-Type: text/html

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" 
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">

● Response message:
  ● Status line
  ● Header lines
  ● Entity body
HTTP Response Message

HTTP/1.1 200 OK
Date: Fri, 22 Feb 2008 16:34:18 GMT
Server: Apache/2.0.52 (Red Hat)
Last-Modified: Thu, 15 Nov 2007 19:33:12 GMT
Content-Length: 15137
Connection: close
Content-Type: text/html

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">

Status Line

HTTP/1.1 – Server is using HTTP/1.1
200 OK - Status code, request succeeded and information is returned in response
HTTP Response Message

HTTP/1.1 404

Status Codes

200 OK
- Status code, request succeeded and information is returned in response.

301 Moved Permanently
- Requested object has been permanently moved.

400 Bad Request
- Generic error message, request not understood by server.

404 Not Found:
- The requested document does not exist on server.

505 HTTP Version Not Supported
- The requested HTTP protocol version not supported by server.
HTTP Request and Response Messages

GET /~rgraziani/ HTTP/1.1
Accept-Language: en-us
User-Agent: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 6.0; SLCC1; .NET CLR 2.0.50727; Media Center PC 5.0; .NET CLR 3.0.04506; InfoPath.1)
Host: www.cabrillo.edu
Connection: Keep-Alive

HTTP/1.1 200 OK
Date: Fri, 22 Feb 2008 16:34:18 GMT
Server: Apache/2.0.52 (Red Hat)
Last-Modified: Thu, 15 Nov 2007 19:33:12 GMT
Content-Length: 15137
Connection: close
Content-Type: text/html

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
User-Server Interaction: Cookies

- **Web servers** are considered **stateless** – they do not maintain state information, keep track of the user.
  - **Higher performance** – allowing the server to handle thousands of simultaneous TCP connections (later).
- Web servers use **cookies to track users**.
- Cookies defined in RFC 2109
User-Server Interaction: Cookies

- Web server installs cookies on client when:
  - Accessed the web site for the **first time** (Web server does not know client by name.)
  - and/or
  - User **provides information** to the web server. (Web server now knows client by name.)

- HTTP on Web server responds with a **Set-cookie**: header with an **ID**.
  - This ID is stored on the client’s computer.

- Each time client/browser accesses web site. The **GET** includes **Cookie:** or **User ID** or similar with the **ID**.
HTTP Request and Response Messages

GET /jpeg/cap81/cam0.36705623.rgb888.enc HTTP/1.1

Cookie: SLSPOTNAME5=Cowells; SLSPOTNAME4=Waimea%20Bay;
        SLSPOTNAME3=Pipeline; SLSPOTNAME2=38th%20Ave%2E; SLSPOTNAME1=Cowells;
        SLSPOTID5=4189; SLSPOTID4=4755; SLSPOTID3=4750; SLSPOTID2=4191;
        SLSPOTID1=4189; OAX=R8bfwEbCU08ABCBu; USER_ID=5551212  <not my actual
        user-id>; <rest of information omitted for brevity>

HTTP: Cookie 5551212 included

HTTP data customized
for Rick Graziani

HTTP/1.1 200 OK
Date: Fri, 22 Feb 2008 19:00:15 GMT
Server: Apache/1.3.34 (Unix)
Last-Modified: Fri, 22 Feb 2008 18:51:47 GMT
ETag: "760a31-18ce-47bf19c3"
Accept-Ranges: bytes
Content-Length: 6350
Keep-Alive: timeout=15, max=257
Connection: Keep-Alive
Content-Type: text/plain <information omitted>
**Web Caching**

- **Web cache** or **proxy server** – Web cache satisfies HTTP requests on the behalf of the Origin Web server.
  - Own disk storage
  - Keeps copies of recently requested objects
- Typically installed at ISP or larger institutions.
- **Advantages:**
  - Reduces the response time for client requests, especially if there are any bottlenecks in the network.
  - Reduces traffic on institution’s access link to the ISP (Internet).
Web Caching

1. Client/browser sends HTTP Request to Web cache (Proxy server).
2. Web cache checks to see if it has a local copy of the object.
   - 2a. Local copy: Web cache sends object to client’s browser.
   - 2b. No Local copy: Web cache sends HTTP request to origin server.
3. Origin server sends object to Web cache.
4. Web cache stores a local copy of the object.
5. Web cache forwards copy of the object to the client browser.
HTTPS

- **HTTPS** (Hypertext Transfer Protocol over Secure Socket Layer) is a URL scheme used to indicate a **secure HTTP connection**.
- HTTPS is not a separate protocol
  - combination of a normal **HTTP** interaction over an encrypted:
    - Secure Sockets Layer (SSL) or
    - Transport Layer Security (TLS) connection
FTP (File Transfer Protocol)

- FTP was developed to allow for **file transfers** between a client and a server.
- Used to **push** and **pull** files from a **server running the FTP daemon** (FTPD).
- Uses **get** and **put** commands.
- RFC 959
FTP (File Transfer Protocol)

- Client initiates a TCP control connection with FTP server using port 21.
  - This connection remains open until the user quits the FTP application.
  - TCP port 21 connection includes:
    - Username and password is sent over TCP port 21.
    - Remote directory changes
      - This state information significantly reduces total number of sessions on server.
- For each file transferred, TCP opens and closes a TCP data connection on port 20.
SMTP – Simple Mail Transfer Protocol
Internet mail involves:

- **User agents**
  - Allows users to read, reply, compose, forward, save, etc., mail messages
  - GUI user agents: Outlook, Eudora, Messenger
  - Text user agents: mail, pine, elm

- **Mail servers**
  - Stores user mail boxes, communicates with local user agents and other mail servers.

- **SMTP**
  - Principle application layer protocol for Internet mail
  - Sent over TCP

- **Mail access protocols**: POP3, IMAP, HTTP
SMTP – Simple Mail Transfer Protocol

- **SMTP**
  - RFC 2821
  - Transfers messages **from sender’s mail server to recipient’s mail server**
    - **Push protocol**, not a pull protocol
      - Push (from client to server or server to server)
      - Pull (from server to client)

- **Retrieving email**
  - Historically, users would log into local mail server to read mail.
  - Since early 1990’s, clients use mail access protocols:
    - POP3
    - IMAP
    - HTTP
SMTP – Simple Mail Transfer Protocol

- POP3 (Post Office Protocol)
  - RFC 1939
  - Uses TCP port 110
  - **Download-and-delete mode**
    - Retrieves messages on server and store the locally
    - Delete messages on server
  - **Download-and-keep mode**
    - Does not delete messages on server when retrieved.
  - **Problem**
    - Difficult to access email from multiple computers – work and home.
    - Some email may have already been downloaded on another computer (work) – download-and-delete
    - To read email from another computer, must leave on server – download-and-keep
**SMTP – Simple Mail Transfer Protocol**

- **IMAP (Internet Message Access Protocol)**
  - RFC 2060
  - Mail not downloaded, but kept on server
  - Received email is associated with user’s INBOX
  - Users can create and manage remote folders
  - Users can retrieve portions of the email:
    - Message header: Subject line and Sender

- **Web-based email**
  - Introduced with Hotmail in mid-1990’s
  - Communicates with remote mailbox using HTTP
  - HTTP is used to push (client to server) and pull the email (server to client)
SMTP

Mail software, processes used: MTA and MDA

- **MUA (Mail User Agent)** – Email client software.
- **MTA (Mail Transfer Agent)** – Software that governs transfer of email between mail servers.
  - Includes UNIX sendmail, Microsoft Exchange Server, Postfix, and Exim
- **MDA (Mail Delivery Agent)** – Software that governs transfer of email from mail servers to clients.
  - On Unix systems, procmail and maildrop are the most popular MDAs.
Telnet provides a standard method of emulating text-based terminal devices over the data network.
Telnet

- Allows a user to remotely access another device (host, router, switch).
- A connection using Telnet is called a Virtual Terminal (VTY) session, or connection.
- Telnet uses software to create a virtual device that provides the same features of a terminal session with access to the server command line interface (CLI).
- Telnet clients:
  - Putty
  - Teraterm
  - Hyperterm
● Telnet supports user authentication, but does not encrypt data.
● All data exchanged during a Telnet session is transported as plain text.
● **Secure Shell (SSH)** protocol offers an alternate and secure method for server access.
  - Stronger authentication
  - Encrypts data
DHCP – Dynamic Host Configuration Protocol

- IP addresses and other information can be obtained:
  - Statically
  - Dynamically (DHCP)
DHCP

- DHCP Information can include:
  - IP address
  - Subnet mask
  - Default gateway
  - Domain name
  - DNS Server

- DHCP servers can be:
  - Server on LAN
  - Router
  - Server at ISP
DHCP

- We will discuss DHCP more when we discuss IPv4.
DNS – Domain Name System

- DNS allows users (software) to use domain names instead of IP addresses
Name Resolution

Resolver
● DNS client programs used to look up DNS name information.

Name Resolution
● The two types of queries that a DNS resolver (either a DNS client or another DNS server) can make to a DNS server are the following:

Recursive queries
● Queries performed by Host to Local DNS Server

Iterative queries
● Queries performed Local DNS server to other servers

Need the IP address
DNS Name Resolution

- User types http://www.example.com

Step 1.
- The DNS resolver on the DNS client sends a recursive query to its configured **Local DNS server**.
- Requests IP address for "www.example.com".
- The DNS server for that client is responsible for resolving the name
  - Cannot refer the DNS client to another DNS server.
Step 2.
● Local DNS Server forwards the query to a **Root DNS server**.

Step 3.
● Root DNS server
  • Makes note of `.com` suffix
  • Returns a list of IP addresses for **TLD** (Top Level Domain Servers) responsible for `.com`. 
DNS Name Resolution

- **Root DNS Servers**
  - There are 13 Root DNS servers (labeled A through M)

- **TLD Servers**
  - Responsible for domains such as .com, edu, org, .net, .uk, jp, fr
  - Network Solutions maintains TLD servers for .com
  - Educause maintains TLD servers for .edu
  - There are redundant servers throughout the world.
Step 4.
- The **local DNS server** sends query for www.example.com to one of the **TLD servers**.

Step 5.
- TLD Server
  - Makes note of **example.com**
  - Returns **IP address** for authoritative server example.com (such as dns.example.com server)
Step 6.
- **Local DNS server** sends query for www.example.com directly to DNS server for example.com

Step 7.
- example.com DNS server responds with its IP address for www.example.com
Step 8.
- **Local DNS server** sends the IP address of www.example.com to the DNS client.

DNS Caching
- When a DNS server receives a DNS reply (mapping hostname to an IP address) it can cache the information in its local memory.
- DNS servers discard cached information after a period of time (usually 2 days)
- A local DNS server can cache TLD server addresses, bypassing the root DNS servers in the query chain.
DNS Name Resolution

- In the worst cases, you'll get a dialog box that says the domain name doesn't exist - even though you know it does.

- This happens because the authoritative server is slow replying to the first, and your computer gets tired of waiting so it times-out (drops the connection) or the domain name does not exist.

- But if you try again, there's a good chance it will work, because the authoritative server has had enough time to reply, and your name server has stored the information in its cache.
nslookup

- Displays default DNS server for your host
- Can be used to query a domain name and get the IP address
DNS Name Resolution

- **ipconfig /displaydns**
  - After a certain amount of time, specified in the Time to Live (TTL) associated with the DNS resource record, the resolver discards the record from the cache.

- **ipconfig /flushdns** – Manually deletes entries

- The default TTL for positive responses is 86,400 seconds (1 day).
- The default TTL for negative responses is 300 seconds.
<table>
<thead>
<tr>
<th>No.</th>
<th>Time</th>
<th>Source</th>
<th>Destination</th>
<th>Protocol</th>
<th>Info</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>2.73829s</td>
<td>192.168.1.101</td>
<td>Broadcast</td>
<td>ARP</td>
<td>Who has 192.168.1.1? Tell 192.168.1.101</td>
</tr>
<tr>
<td>4</td>
<td>2.73906s</td>
<td>192.168.1.1</td>
<td>192.168.1.101</td>
<td>ARP</td>
<td>192.168.1.1 is at 00:0f:66:09:4e:0f</td>
</tr>
<tr>
<td>5</td>
<td>2.73907s</td>
<td>192.168.1.101</td>
<td>204.127.199.8</td>
<td>DNS</td>
<td>Standard query A <a href="http://www.ucsc.edu">www.ucsc.edu</a></td>
</tr>
<tr>
<td>6</td>
<td>2.77840s</td>
<td>204.127.199.8</td>
<td>192.168.1.101</td>
<td>DNS</td>
<td>Standard query response CNAME ucsc.edu A 128.114.124.7</td>
</tr>
<tr>
<td>7</td>
<td>2.78462s</td>
<td>192.168.1.101</td>
<td>128.114.124.7</td>
<td>ICMP</td>
<td>Echo (ping) request</td>
</tr>
<tr>
<td>8</td>
<td>2.87576s</td>
<td>192.168.1.101</td>
<td>128.114.124.7</td>
<td>ICMP</td>
<td>Echo (ping) reply</td>
</tr>
<tr>
<td>9</td>
<td>3.78742s</td>
<td>192.168.1.101</td>
<td>128.114.124.7</td>
<td>ICMP</td>
<td>Echo (ping) request</td>
</tr>
<tr>
<td>10</td>
<td>3.88614s</td>
<td>128.114.124.7</td>
<td>192.168.1.101</td>
<td>ICMP</td>
<td>Echo (ping) reply</td>
</tr>
</tbody>
</table>
The Server Message Block (SMB) is a client/server file sharing protocol. IBM developed Server Message Block (SMB) in the late 1980s to describe the structure of shared network resources, such as directories, files, printers, and serial ports.

SMB is a client-server, request-response protocol. Servers can make their resources available to clients on the network.
• Request-response protocol.
• Client can access the resources on the server as if the resource is local to the client host.
• SMB is sent over TCP
• Linux/UNIX have similar protocol: SAMBA
SMB messages can:

- Start, authenticate, and terminate sessions
- Control file and printer access
- Allow an application to send or receive messages to or from another device
In addition to the client/server model for networking, there is also a peer-to-peer model. Two or more computers are connected via a network and can share resources (such as printers and files) without having a dedicated server. End devices (peers) can function as either a server or client.
P2P File Sharing

- P2P (Peer-to-Peer) file sharing accounts for more traffic on the Internet than any other application (2004).
- Peers (hosts) act as both clients and servers.
- No centralized file server.
- HTTP GET and responses are commonly used.
● Challenge with P2P – locating content across thousands or millions of peers.

● One solution – centralized directory
  ● Approach done by Napster

● Problems (non-legal problems)
  ● Single point of failure
  ● Performance bottlenecks
Gnutella – public domain file sharing application

- Fully distributed approach
  - No centralized server
- Gnutella peer maintains peering relationship (TCP connection – later) which a number of other peers (usually fewer than 10).