- ✓ Each URL has two components: the hostname of the server that houses the object and the object's path name. For example, the URL http://www.someSchool.edu/someDepartment/picture.gif.

### 2.1.1 HTTP

- ✓ HTTP defines how Web clients request Web pages from Web servers and how servers transfer Web pages to clients.
- ✓ The HTTP client first initiates a TCP connection with the server.
- ✓ Once the connection is established, the browser and the server processes access TCP through their socket interfaces.
- ✓ On the client side the socket interface is the door between the client process and the TCP connection.
- ✓ On the server side it is the door between the server process and the TCP connection.
- ✓ The client sends HTTP request messages into its socket interface and receives HTTP response messages from its socket interface. Similarly, the HTTP server receives request messages from its socket interface and sends response messages into its socket interface.
- ✓ Once the client sends a message into its socket interface, the message is out of the client's hands and is "in the hands" of TCP. TCP provides a reliable data transfer service to HTTP.
- ✓ HTTP follows client/server model
  - **client**: a browser that requests, receives, (using HTTP protocol) and "displays" Web objects
  - **server**: Web server sends (using HTTP protocol) objects in response to requests
- ✓ HTTP connection types
  1. Non-persistent HTTP
  2. Persistent HTTP



## 2.2 Non-Persistent and Persistent Connections

Depending on the application and on how the application is being used, the series of requests may be made back-to-back, periodically at regular intervals, or intermittently. When this client-server interaction is taking place over TCP, the application developer needs to make an important decision—should each request/response pair be sent over a *separate* TCP connection.

### 2.2.1 Non-persistent HTTP

- ✓ A non-persistent connection is the one that is closed after the server sends the requested object to the client. In other words, the connection is used exactly for one request and one response as shown in figure 7.
- ✓ For downloading multiple objects it required multiple connections.
- ✓ Non-persistent connections are the default mode for HTTP/1.0.
- ✓ suppose a user enters URL: www.someSchool.edu/someDepartment/home.index
- ✓ Above link contains text and references to 10 jpeg images.

- ✓ Non-persistent connections have some shortcomings.
  - First, a brand-new connection must be established and maintained for *each requested object*.
  - For each of these connections, TCP buffers must be allocated and TCP variables must be kept in both the client and server.
  - Each object suffers a delivery delay of two RTTs— one RTT to establish the TCP connection and one RTT to request and receive an object.

### 2.2.2 HTTP with Persistent Connections

- ✓ The server leaves the TCP connection open after sending a response. Subsequent requests and responses between the same client and server can be sent over the same connection.
- ✓ An entire Web page can be sent over a single persistent TCP connection. Moreover, multiple Web pages residing on the same server can be sent from the server to the same client over a single persistent TCP connection.
- ✓ These requests for objects can be made back-to-back, without waiting for replies to pending requests (pipelining).
- ✓ The HTTP server closes a connection when it isn't used for a certain time.
- ✓ When the server receives the back-to-back requests, it sends the objects back-to-back.
- ✓ It requires as little as one RTT for all the referenced objects
- ✓ With persistent connections, the performance is improved by 20%.
- ✓ Persistent connections are the default mode for HTTP/1.1



### 2.3 HTTP Message Format

- ➢ There are two types of HTTP messages, request messages and response messages

#### 1.3.4.1 HTTP Request Message

- ✓ Below we provide a typical HTTP request message:

```
GET /somedir/page.html HTTP/1.1
Host: www.someschool.edu
Connection: close
User-agent: Mozilla/5.0
Accept-language: fr
```

- ✓ The message consists of five lines, each followed by a **carriage return** and **a line feed**.
- ✓ The last line is followed by an additional carriage return and line feed.
- ✓ The first line of an HTTP request message is called the **request line**; the subsequent lines are called the **header lines**.
- ✓ The **request line** has three fields**: the method field, the URL field, and the HTTP version field.**
- ✓ The **method field** can take on several different values, including **GET, POST, HEAD, PUT, and DELETE.**

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

- This opens a TCP connection to port 80 of the host cis.poly.edu and then sends the HTTP request message.
- You should see a response message that includes the base HTML file of Professor Ross's homepage.
- If you'd rather just see the HTTP message lines and not receive the object itself, replace GET with HEAD.
- Finally, replace /~ross/ with /~banana/ and see what kind of response message you get.

- The HTTP specification defines many, many more header lines that can be inserted by browsers, Web servers, and network cache servers.

**3. User-Server Interaction**

**3.1 Cookies**

✓ A small text file created by a Web site that is stored in the user's computer either temporarily for that session only or permanently on the hard disk (persistent cookie).
✓ Cookies provide a way for the Web site to recognize you and keep track of your preferences.
✓ Web servers that can handle thousands of simultaneous TCP connections.
✓ A Web site to identify users, either because the server wishes to restrict user access or because it wants to serve content as a function of the user identity.
✓ For these purposes, HTTP uses cookies. Cookies, defined in [RFC 6265], allow sites to keep track of users.
✓ **Cookie technology has four components:**

(1) a cookie header line in the HTTP response message;
(2) a cookie header line in the HTTP request message;
(3) a cookie file kept on the user's end system and managed by the user's browser;
(4) a back-end database at the Web site.

- When it receives requests from and sends responses to a browser, it is a server.
- When it sends requests to and receives responses from an origin server, it is a client.
- Typically a Web cache is purchased and installed by an ISP.
- A major residential ISP (such as AOL) might install one or more caches in its network and preconfigure its shipped browsers to point to the installed caches.

✓ Web caching has seen deployment in the Internet for two reasons.

- First, a Web cache can substantially reduce the response time for a client request, particularly if the bottleneck bandwidth between the client and the origin server is much less than the bottleneck bandwidth between the client and the cache.
- If there is a high-speed connection between the client and the cache, as there often is, and if the cache has the requested object, then the cache will be able to deliver the object rapidly to the client.
- Through the use of **Content Distribution Networks (CDNs)**, Web caches are increasingly playing an important role in the Internet.
- A CDN company installs many geographically distributed caches throughout the Internet, thereby localizing much of the traffic. There are shared CDNs and dedicated CDNs.

✓ **Why Web caching is needed (Required)? OR Advantages of Caching**

- To reduce response time for a client request
- **To reduce traffic on an institution's access link**
- To enable "poor" content providers to effectively deliver content



**3.3 The Conditional GET**

✓ The object housed in the Web server may have been modified since the copy was cached at the client.

✓ Fortunately, HTTP has a mechanism that allows a cache to verify that its objects are up to date. This mechanism is called the **conditional GET**.

✓ An HTTP request message is a so-called conditional GET message if

1. the request message uses the GET method and
2. the request message includes an If-Modified- Since: header line.

✓ First, on the behalf of a requesting browser, a proxy cache sends a request message to a Web server:

**GET /fruit/kiwi.gif HTTP/1.1**
**Host: www.exotiquecuisine.com**

✓ Second, the Web server sends a response message with the requested object to the cache:

**HTTP/1.1 200 OK**
**Date: Sat, 8 Oct 2011 15:39:29**
**Server: Apache/1.3.0 (Unix)**
**Last-Modified: Wed, 7 Sep 2011 09:23:24**
**Content-Type: image/gif**
**(data data data data data ...)**

- ✓ Once the server has authorized the user, the user copies one or more files stored in the local file system into the remote file system (or vice versa).
- ✓ FTP uses two parallel TCP connections to transfer a file, as shown in figure 12.
  1. control connection
  2. data connection


Fig.12 : Control and data connections

- ✓ The **control connection** is used for sending control information between the two hosts such as user identification, password, commands to change remote directory and commands to put and get files.
- ✓ The data connection is used to actually send a file.
- ✓ Because FTP uses a separate control connection, FTP is said to send its control information out-of-band.
- ✓ When a user starts an FTP session with a remote host, the client side of FTP (user) first initiates a control TCP connection with the server side (remote host) on server port number 21.
- ✓ The client side of FTP sends the user identification and password over this control connection.
- ✓ The client side of FTP also sends, over the control connection, commands to change the remote directory.
- ✓ When the server side receives a command for a file transfer over the control connection (either to or from, the remote host), the server side initiates a TCP data connection to the client side.
- ✓ FTP sends exactly one file over the data connection and then closes the data connection.
- ✓ If during the same session, the user wants to transfer another file, FTP opens another data connection.
- ✓ Thus, with FTP, the control connection remains open throughout the duration of the user session, but a new data connection is created for each file transferred within a session (that is, the data connections are non-persistent).



### 4.1 FTP Commands and Replies

The commands, from client to server, and replies, from server to client, are sent across the control connection in 7-bit ASCII format. Thus, like HTTP commands, FTP commands are readable by people.

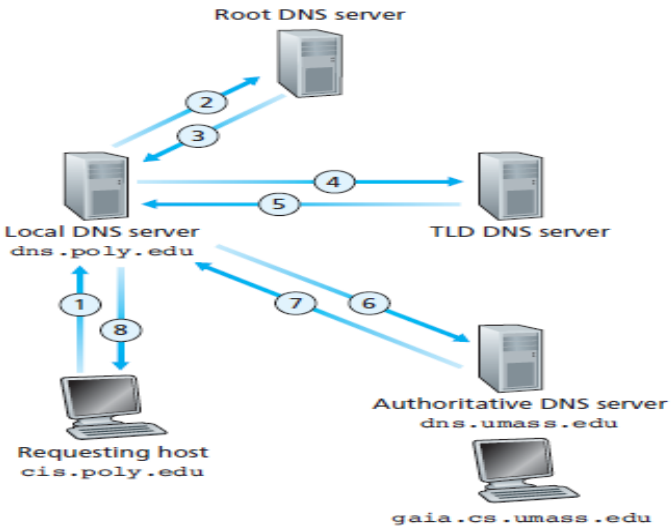


**Figure 17 : Interaction of the various DNS servers**

- The query message contains the hostname to be translated, namely, gaia.cs.umass.edu.
- The local DNS server forwards the query message to a root DNS server.
- The root DNS server takes note of the edu suffix and returns to the local DNS server a list of IP addresses for TLD servers responsible for edu.
- The local DNS server then resends the query message to one of these TLD servers.
- The TLD server takes note of the umass.edu suffix and responds with the IP address of the authoritative DNS server for the University of Massachusetts, namely, dns.umass.edu.
- Finally, the local DNS server resends the query message directly to dns.umass.edu, which responds with the IP address of gaia.cs.umass.edu.
- Note that in this example, in order to obtain the mapping for one hostname, eight DNS messages were sent: four query messages and four reply messages
- Also suppose that each of the departments at the University of Massachusetts has its own DNS server, and that each departmental DNS server is authoritative for all hosts in the department.
- In this case, when the intermediate DNS server, dns.umass.edu, receives a query for a host with a hostname ending with cs.umass.edu, it returns to dns.poly.edu the IP address of dns.cs.umass.edu, which is authoritative for all hostnames ending with cs.umass.edu.
- The local DNS server dns.poly.edu then sends the query to the authoritative DNS server, which returns the desired mapping to the local DNS server, which in turn returns the mapping to the requesting host. In this case, a total of 10 DNS messages are sent!



### 6.2.2 DNS Caching

✓ In a query chain, when a DNS server receives a DNS reply, it can cache the mapping in its local memory.

- ✓ If a hostname/IP address pair is cached in a DNS server and another query arrives to the DNS server for the same hostname, the DNS server can provide the desired IP address, even if it is not authoritative for the hostname.
- ✓ Because hosts and mappings between hostnames and IP addresses are by no means permanent, DNS servers discard cached information after a period of time.

### 6.3 DNS Records and Messages

- ✓ The DNS servers that together implement the DNS distributed database store **resource records (RRs)**, including RRs that provide hostname-to-IP address mappings.
- ✓ Each DNS reply message carries one or more resource records.
- ✓ A resource record is a four-tuple that contains the following fields:

(Name, Value, Type, TTL)

TTL is the time to live of the resource record; it determines when a resource should be removed from a cache. In the example records given below, we ignore the TTL field. The meaning of Name and Value depend on Type:

• **If Type=A,** then Name is a hostname and Value is the IP address for the hostname. Thus, a Type A record provides the standard hostname-to-IP address mapping. As an example, (relay1.bar.foo.com, 145.37.93.126, A) is a Type A record.

• **If Type=NS,** then Name is a domain (such as foo.com) and Value is the hostname of an authoritative DNS server that knows how to obtain the IP addresses for hosts in the domain. This record is used to route DNS queries further along in A resource record is a four-tuple that contains the following fields: (Name, Value, Type, TTL) TTL is the time to live of the resource record; it determines when a resource should be removed from a cache. In the example records given below, we ignore the TTL field. The meaning of Name and Value depend on Type:



• **If Type=CNAME,** then Value is a canonical hostname for the alias hostname Name. This record can provide querying hosts the canonical name for a hostname. As an example, (foo.com, relay1.bar.foo.com, CNAME) is a CNAME record.

• **If Type=MX**, then Value is the canonical name of a mail server that has an alias hostname Name. As an example, (foo.com, mail.bar.foo.com, MX) is an MX record. MX records allow the hostnames of mail servers to have simple aliases. Note that by using the MX record, a company can have the same aliased name for its mail server and for one of its other servers (such as its Web server). To obtain the canonical name for the mail server, a DNS client would query for an MX record; to obtain the canonical name for the other server, the DNS client would query for the CNAME record. If a DNS server is authoritative for a particular hostname, then the DNS server will contain a Type A record for the hostname.

#### 6.3.1 DNS Messages

These are the only two kinds of DNS messages as shown in figure 18.