Application Layer: Application Layer: Principles of Network Applications, The Web and HTTP, File Transfer: FTP, Electronic Mail in the Internet, DNS-The Internet's Directory Service: Services provided by DNS, overview of how DNS works.

*****

Principles of Network Applications:

1. Introduction:

- The application layer is responsible for providing network services directly to user applications.
- Applications interact with the network using application-layer protocols.

Client-Server Architecture

- A **server** provides services; a **client** requests services.
- The server is always on, while the client initiates communication.
- Example: Web browsing (HTTP), Email (SMTP).

Peer-to-Peer (P2P) Architecture

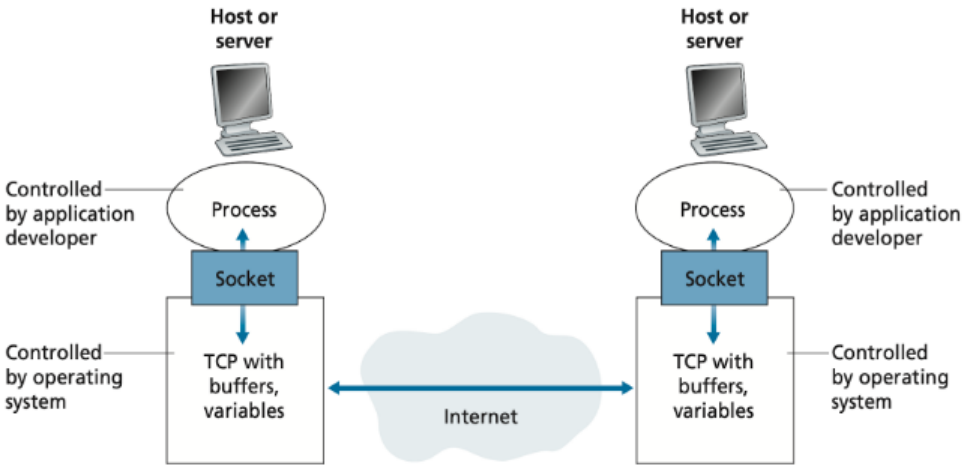
- No always-on server; peers communicate directly.
- More scalable than client-server.
- Example: File sharing (BitTorrent), VoIP (Skype).

Hybrid Architectures

- Combination of client-server and P2P models.
- Example: Instant messaging (centralized login, P2P communication).

2. Processes Communicating

- Process: A program running on a host.
- Inter-process communication occurs between processes on the same host (e.g., via IPC mechanisms).
- Network applications use socket programming to communicate over networks.
- Sockets serve as endpoints for sending/receiving data as shown below.



3. Transport Services Required by Applications

Applications rely on transport-layer protocols for communication.

Key services:

- Reliable data transfer (e.g., TCP for web browsing, email).
- Unreliable data transfer (e.g., UDP for streaming, gaming).
- Throughput requirements (some apps need high bandwidth).
- Timing constraints (e.g., VoIP needs low delay).
- Security (encryption for sensitive data).

4. Application-Layer Protocols

Define how processes communicate over the network.

Components:

- Message syntax – Format of messages exchanged.
- Message semantics – Meaning of messages.
- Message rules – When and how messages are sent/received.
- Transport layer dependencies – Use TCP or UDP.

5. Common Application-Layer Protocols

Protocol	Transport Layer	Purpose
HTTP	TCP	Web browsing
SMTP	TCP	Email transfer
IMAP/POP3	TCP	Email retrieval
FTP	TCP	File transfer
DNS	UDP (sometimes TCP)	Domain name resolution
BitTorrent	TCP/UDP	P2P file sharing
VoIP (SIP, RTP)	UDP	Internet telephony

5.1 HTTP (Hypertext Transfer Protocol)

- Used for web page retrieval.
- Stateless: No session memory between requests.
- Uses TCP (port 80 for HTTP, port 443 for HTTPS).
- Request-response model:
- Client sends HTTP request (GET, POST, etc.).
- Server responds with HTTP response (status codes: 200 OK, 404 Not Found).

5.2 SMTP (Simple Mail Transfer Protocol)

- Used for sending email.
- Push-based protocol (emails are pushed from sender to recipient mail server).
- Uses TCP port 25.

5.3 DNS (Domain Name System)

- Converts human-readable domain names (e.g., google.com) into IP addresses.
- Uses UDP port 53 for quick lookups.
- Hierarchical structure:
- Root DNS servers

- TLD (Top-Level Domain) servers (e.g., .com, .org).
- Authoritative DNS servers (store domain-specific records).

5.4 FTP (File Transfer Protocol)

- Used for file transfers between client and server.
- Uses TCP (port 20, 21).
- Supports active and passive modes.

5.5 BitTorrent (P2P File Sharing)

- Uses tracker-based P2P architecture.
- Files split into chunks, which peers exchange.
- Peers download from multiple sources simultaneously.

5.6 VoIP Protocols (SIP, RTP)

- SIP (Session Initiation Protocol) sets up calls.
- RTP (Real-time Transport Protocol) carries voice/video data.
- Uses UDP for low-latency delivery.

The Web and HTTP:

1. The Web: Overview

- The World Wide Web (WWW) is a distributed system of interconnected documents and resources, accessed via the Internet.
- It uses the Hypertext Transfer Protocol (HTTP) for communication.
- Web pages are written in HTML (HyperText Markup Language) and can include text, images, videos, and links.

1.1 Components of the Web

Web Clients (Browsers)

- Examples: Chrome, Firefox, Edge.
- Fetches web pages using HTTP and displays them to users.

Web Servers

- Store web pages and serve them upon request.
- Examples: Apache, Nginx, Microsoft IIS.

Web Pages

- Built using HTML, CSS, and JavaScript.
- Can contain hyperlinks to other pages.

URLs (Uniform Resource Locators)

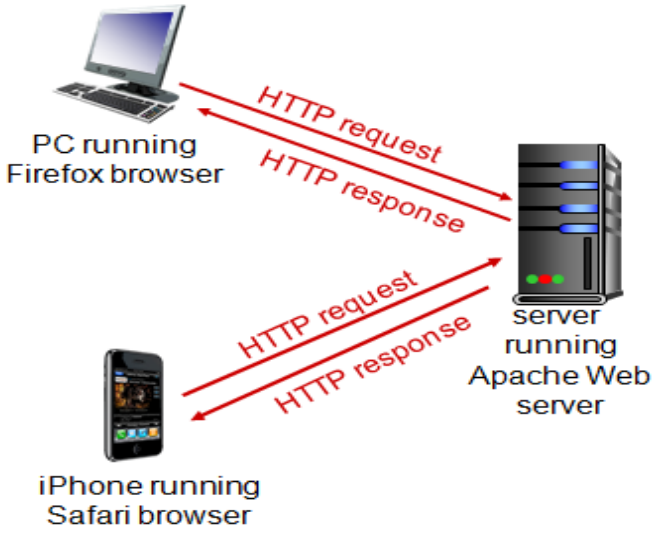
- Identify the location of resources on the web.
- Structure: protocol://hostname[:port]/path
- Example: https://www.example.com/index.html

2. HTTP: The Hypertext Transfer Protocol

- Application-layer protocol for web communication.
- Uses a client-server model:
  - Client: Sends HTTP requests (browser).
  - Server: Responds with requested web content.

3. HTTP Request and Response Structure

HTTP Request Message Structure



HTTP request-response behavior

A request message consists of:

1. Request line – Contains the request method, URL, and HTTP version.
   - Example: GET /index.html HTTP/1.1
2. Header fields – Provide additional information (e.g., browser type, content type).

- **Example:** User-Agent: Mozilla/5.0

3. Body (optional) – Used in POST requests to send form data.

Common HTTP Methods:

Method	Description
GET	Requests a resource from the server (e.g., webpage, image).
POST	Sends data to the server (e.g., form submission).
PUT	Uploads or updates a resource.
DELETE	Deletes a resource from the server.
HEAD	Retrieves only headers, not the full content.

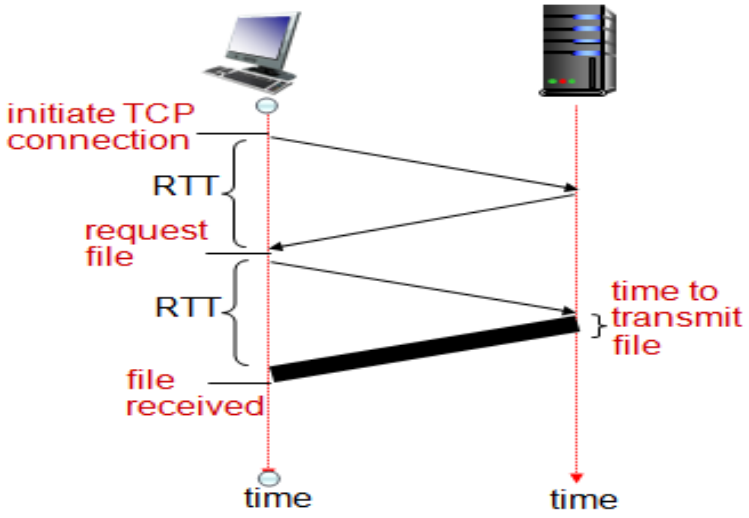
HTTP Response Message Structure

A response message consists of:

1. Status line – Contains the HTTP version, status code, and status phrase.
   Example: HTTP/1.1 200 OK
2. Header fields – Provide metadata about the response (e.g., content type, date).
   Example: Content-Type: text/html
3. Body – Contains the requested web page or object (e.g., HTML, image).

4. Non Persistent and Persistent Connections

- Non persistent connections create a new connection for each requested object. Persistent connections allow multiple objects to be sent over the same connection, improving efficiency.
- Although HTTP uses persistent connections in its default mode, HTTP clients and servers can be configured to use non persistent connections.



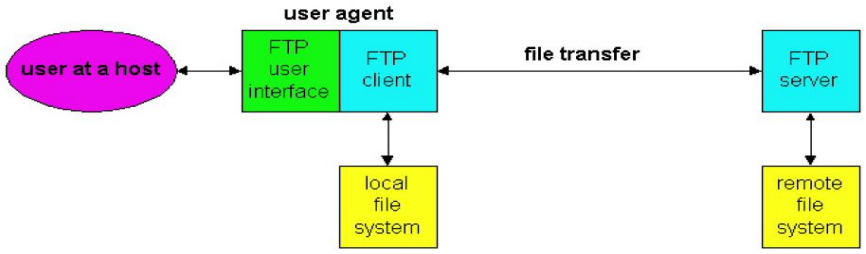
- Here RTT is round trip time, which is the time it takes for a small packet to travel from client to server and then back to the client. The RTT includes packet propagation delays, packet queuing delays in intermediate routers and switches, and packet processing delays.
- The three way handshake involves the client and server exchanging messages, taking one round trip time (RTT) for this process.
- After completing the handshake, the client sends an HTTP request message along with an acknowledgment into the TCP connection.
- The server responds by sending the HTML file over the established connection.

Persistent vs Non-Persistent Response Time

Type	Number of RTTs per Object	Total Response Time for N Objects
Non-Persistent HTTP (HTTP/1.0)	2 RTTs per object (1 for TCP handshake, 1 for data transfer)	2N x RTT + N x Tansmission Time
Persistent HTTP (HTTP/1.1)	1 RTT for the first object, then 1 RTT for all others combined	1 x RTT + N x Transmission Time

Introduction to FTP (File Transfer Protocol)

- FTP (File Transfer Protocol) is a standard application layer protocol used for transferring files between a client and a server over the Internet.
- Unlike HTTP, which transfers web content, FTP is specifically designed for efficient file sharing.
- FTP follows the client-server model, where:
  - The client requests file operations (upload/download).
  - The server stores and manages files.



FTP moves files between local and remote file systems

Features of FTP

- Allows uploading and downloading of files.
- Supports authentication (username and password) for secure access.
- Provides two modes of data transfer: Active Mode & Passive Mode.
- Uses separate control and data connections for communication.
- Supports directory listing, file deletion, renaming, and permissions management.

FTP Architecture & Working

FTP operates using two parallel TCP connections:

1. Control Connection (Port 21):
   - Used for sending commands and responses between client and server.
   - Maintains the session throughout the file transfer process.
2. Data Connection (Port 20 - Active Mode / Random Port - Passive Mode):
   - Used for actual file transfer.
   - A new connection is established for each file transfer operation.

FTP Modes of Operation

(A) Active Mode FTP

- The FTP client initiates the control connection to the server on port 21.
- When a file transfer is requested:
  - The server initiates a data connection back to the client using port 20.
- Limitation: If the client is behind a firewall or NAT, the incoming server connection may be blocked.

(B) Passive Mode FTP

- Used to overcome firewall/NAT restrictions.
- The client initiates both the control and data connections.
- The server provides a random port number, and the client connects to it for file transfer.
- Advantage: Works better in restricted network environments.

FTP Commands and Responses

(A) Common FTP Commands (Sent by Client)

Command	Description
USER <username>	Provides the username for authentication.
PASS <password>	Provides the password for authentication.
LIST	Lists files in the current directory.
RETR <filename>	Downloads a file from the server.
STOR <filename>	Uploads a file to the server.
DELE <filename>	Deletes a file from the server.
CWD <directory>	Changes the working directory.
QUIT	Ends the session.

(B) Server Response Codes

Code	Meaning
200	Command successful.
220	Service ready for new user.
331	Username OK, password required.
425	Cannot open data connection.
530	Authentication failure (Invalid login).

Anonymous FTP

- Some FTP servers allow anonymous access where users can connect without authentication.
- Used for public file sharing (e.g., downloading software, documents).
- Users typically log in using "anonymous" as the username and their email as the password.

Security Issues in FTP

(A) Plaintext Authentication & Data Transfer

- FTP sends login credentials (username & password) and files in plaintext, making it vulnerable to packet sniffing.

(B) FTP Bounce Attack

- Attackers exploit the FTP protocol to redirect connections through the server to attack other hosts.

(C) Brute-Force Attacks

- Hackers attempt to guess FTP login credentials using automated tools.

(D) Firewall and NAT Issues

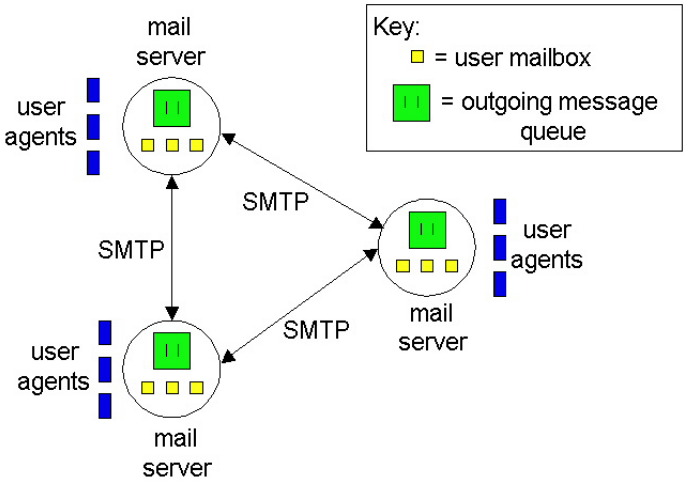
- Active mode FTP is often blocked due to firewalls rejecting incoming server connections.

Introduction to Electronic Mail (Email)

Electronic mail (email) is one of the most widely used network applications, allowing users to send and receive messages over the Internet. It operates based on the client-server model and relies on several standardized protocols for message transmission and retrieval.

Components of Email System

The email system consists of three major components as shown in the figure.



1. User Agents (UA):
   - Software applications like Gmail, Outlook, and Thunderbird that allow users to compose, send, receive, and manage emails.
   - Provides functions like attaching files, replying, and forwarding messages.
2. Mail Servers:
   - Store and manage incoming and outgoing emails.
   - Each domain has at least one mail server responsible for handling emails (e.g., mail.google.com for Gmail).
3. Protocols for Email Transmission & Retrieval:
   - SMTP (Simple Mail Transfer Protocol)
   - POP3 (Post Office Protocol v3)
   - IMAP (Internet Message Access Protocol)

Email Protocols

(A) Simple Mail Transfer Protocol (SMTP)

- Purpose: Used for sending emails between mail servers and from a user's device to the mail server.
- Characteristics:
  - Uses TCP port 25 (or 587 for secure submission).
  - Works in a push model (emails are pushed from the sender's server to the recipient's server).
  - Operates in three phases:
    1. Handshake: The client establishes a connection with the SMTP server.
    2. Message Transfer: Email content is sent using commands like MAIL FROM, RCPT TO, and DATA.
    3. Closure: The connection is terminated after successful transmission.
- Limitations of SMTP:
  - Sends emails in 7-bit ASCII text format, making it unsuitable for non-text attachments (handled by MIME).

(B) Mail Retrieval Protocols

Once an email reaches the recipient's mail server, users retrieve it using one of the following protocols:

1. Post Office Protocol v3 (POP3)
   - Purpose: Downloads emails from the server to the local device.
   - Characteristics:
     - Uses TCP port 110 (or 995 for encrypted POP3S).
     - Two modes:
       - Download-and-delete: Emails are removed from the server after being downloaded.

        - Download-and-keep: Copies of emails remain on the server.
    - Suitable for users accessing email from a single device.

2. Internet Message Access Protocol (IMAP)
    - Purpose: Allows users to access and manage their emails directly on the server.
    - Characteristics:
        - Uses TCP port 143 (or 993 for encrypted IMAPS).
        - Supports server-side organization (folders, labels, flags).
        - Synchronization across multiple devices (ideal for users accessing email from different locations).
        - Emails remain on the server until explicitly deleted by the user.

Email Message Format

Emails are structured into two main components:

1. Header:
    - Contains metadata, including:
        - From: (sender's email address)
        - To: (recipient's email address)
        - Subject: (brief description of the email)
        - Date: (timestamp of sending)
        - CC: (carbon copy) and BCC: (blind carbon copy)
2. Body:
    - Contains the actual message content (plain text or HTML).
    - May include attachments (handled using MIME).

Multipurpose Internet Mail Extensions (MIME)

- Purpose: Allows emails to include attachments, images, audio, video, and non-ASCII characters.
- How It Works:
    - Encodes binary files (like images) into text using Base64 encoding.

- Defines headers to specify content type (e.g., Content-Type: image/jpeg).

- Example MIME Types:
  - text/plain → Plain text
  - text/html → HTML email
  - image/jpeg → JPEG image attachment
  - application/pdf → PDF document attachment

Working of Email System – Step-by-Step Process

1. Sending an Email:
   - The user composes an email in a User Agent (UA) (e.g., Outlook, Gmail).
   - The UA sends the email to an SMTP server.
   - The SMTP server forwards the email to the recipient's mail server.
2. Receiving an Email:
   - The recipient's mail server stores the email in the mailbox.
   - The user retrieves the email using a Mail Access Protocol (POP3 or IMAP).
   - The email is displayed in the User Agent.

Security Issues & Solutions in Email

- Spam & Phishing:
  - Solution: Spam filters, email authentication protocols like SPF, DKIM, and DMARC.
- Email Spoofing:
  - Solution: Digital signatures and email authentication mechanisms.
- Email Encryption:
  - Solution: PGP (Pretty Good Privacy), S/MIME for securing email content.

Introduction to DNS:

- The Domain Name System (DNS) is a hierarchical and distributed naming that translates human-readable domain names (e.g., www.google.com) into IP addresses (142.250.183.206).
- Since IP addresses are difficult to remember, DNS provides an efficient way to map domain names to their corresponding numerical addresses.
- DNS operates at the application layer of the Internet protocol stack and follows a client-server architecture.

Services Provided by DNS

DNS provides several important services for the Internet:

(A) Name Resolution (Hostname to IP Address Mapping)

- The primary function of DNS is to map human-readable domain names (e.g., www.facebook.com) to their corresponding IP addresses (e.g., 157.240.22.35).
- This allows users to access websites without memorizing complex numerical addresses.

(B) Host Aliasing

- Many websites have multiple domain names that point to the same IP address.
- DNS supports aliasing, where multiple names refer to the same machine:
    - Example: www.example.com → server1.example.com → 192.168.1.1

(C) Mail Server Address Resolution (Mail Exchange - MX Records)

- DNS helps email services by mapping a domain name to its associated mail server.
- Example:
    - mail.example.com → IP of the mail server
    - MX records ensure that emails are sent to the correct server.

(D) Load Balancing (Distributing Traffic Among Multiple Servers)

- Large websites (e.g., Google, Amazon) have multiple servers hosting the same content.
- DNS can distribute traffic among these servers to improve reliability and performance.
- Example: www.google.com may resolve to different IPs based on user location.

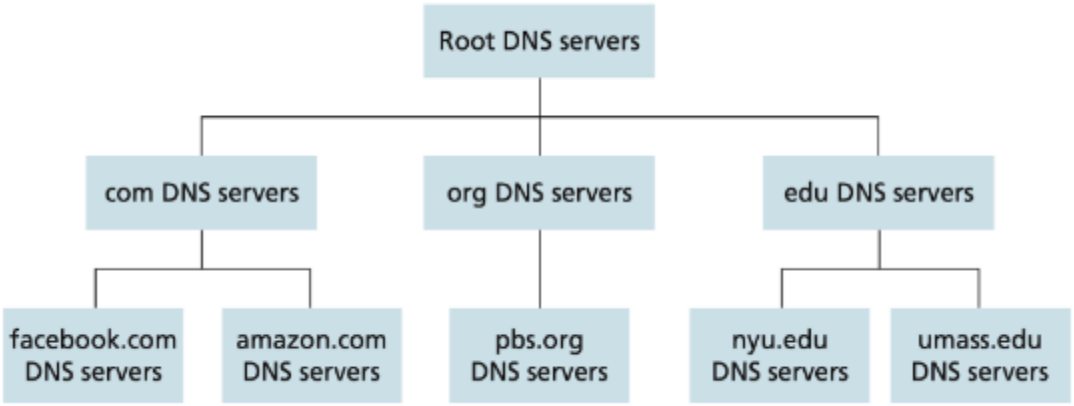
(E) Reverse DNS Lookup (IP to Hostname Mapping)

- In addition to resolving domain names to IPs, DNS also allows reverse lookup, where an IP address is mapped back to a domain name.
- Example:
  - 8.8.8.8 → dns.google.com

How DNS Works – An Overview

(A) DNS Hierarchy & Structure

DNS follows a hierarchical structure, divided into multiple levels as shown in the figure:



1. Root DNS Servers:
   - The top level of the DNS hierarchy.
   - Contains information about top-level domain (TLD) servers.
   - Example: If querying www.example.com, the root server directs to the .com TLD server.

2. Top-Level Domain (TLD) Servers:
    - Responsible for domains like .com, .org, .net, .edu, .gov.
    - Maintains records for authoritative DNS servers of specific domains.
    - Example: The .com TLD directs queries for example.com to its authoritative name server.
3. Authoritative DNS Servers:
    - Stores actual domain name to IP address mappings.
    - Example: The example.com authoritative DNS server returns the IP address of www.example.com.
4. Local DNS Resolver (Recursive Resolver):
    - A DNS resolver is responsible for querying other DNS servers.
    - Located within Internet Service Providers (ISPs) or organizations.
    - Helps speed up responses by caching DNS records.

(B) Steps in DNS Query Resolution

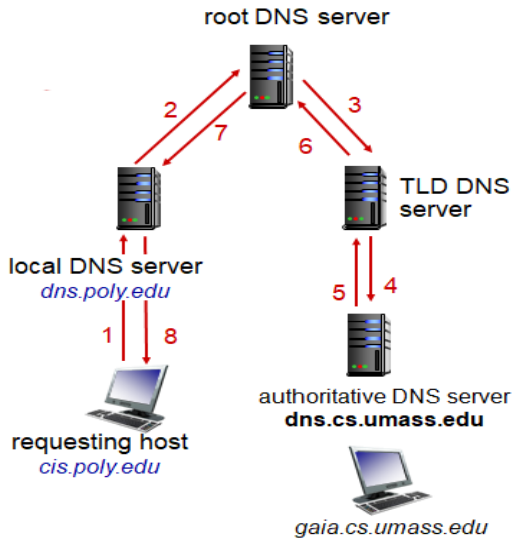
1. User enters www.example.com in a web browser.
2. The request is sent to the Local DNS Resolver (usually managed by the ISP).
3. If the resolver has the IP cached, it returns the result immediately. Otherwise:
    - The resolver queries the Root DNS Server.
    - The root server directs it to the TLD DNS Server (for .com).
    - The TLD server then points to the Authoritative DNS Server for example.com.
    - The authoritative server provides the IP address (192.168.1.1).
4. The resolver caches the result and returns the IP to the user's browser.
5. The browser establishes a connection with the web server and loads the website.

Types of DNS Queries

DNS queries can be classified into two types:

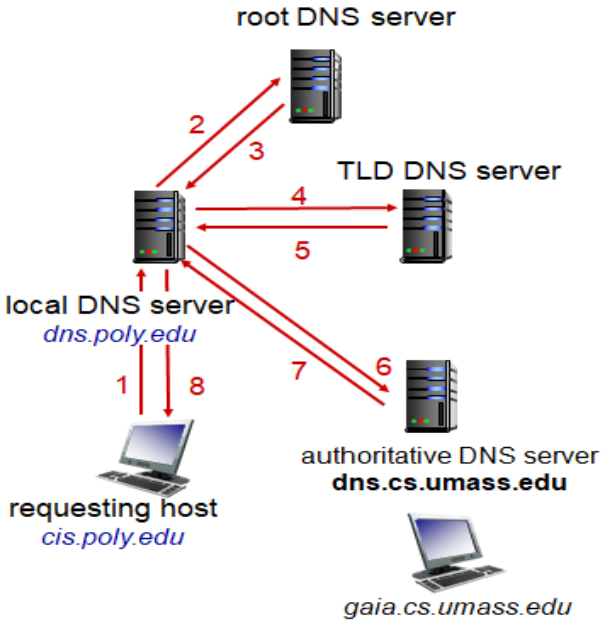
(A) Recursive Query

- The DNS client asks the resolver to find the exact IP address.
- The resolver performs all necessary lookups and returns the final result.



(B) Iterative Query

- The resolver contacts each server step by step, instead of fetching the entire result at once.
- The server returns the next DNS server's address instead of the final IP.



DNS Caching & Performance Optimization

To reduce lookup time and improve performance, DNS uses caching at multiple levels:

(A) Local Caching in Browsers & Operating Systems

- Web browsers (Chrome, Firefox) and operating systems (Windows, Linux) store recent DNS resolutions.
- Example: If you visit www.google.com, the IP is cached for faster future access.

(B) ISP-Level Caching

- ISPs cache popular DNS responses to avoid repeated lookups.

(C) Time-To-Live (TTL) in DNS Records

- DNS records have a TTL (Time-To-Live) value, which determines how long a DNS entry is cached before needing an update.
- Example: TTL = 3600 means the record is cached for 1 hour.

Security Issues in DNS

(A) DNS Spoofing (Cache Poisoning)

- Attackers insert fake IP addresses into DNS caches, redirecting users to malicious websites.
- Solution: DNSSEC (DNS Security Extensions) ensures DNS responses are digitally signed.

(B) DDoS Attacks on DNS Servers

- Attackers flood DNS servers with requests, making websites inaccessible.
- Solution: Load balancing and using multiple redundant DNS servers.

(C) Privacy Issues in DNS

- DNS queries are not encrypted, allowing ISPs to track users' browsing behavior.
- Solution:
  - DNS over HTTPS (DoH): Encrypts DNS requests over HTTPS.
  - DNS over TLS (DoT): Encrypts DNS traffic using TLS.

*****