Chapter 2:

Communication protocols : Application Layer

Master 1 IS , Guelma University, 2024-2025

Mohamed Amine Ferrag, PhD

Introduction (1/2)

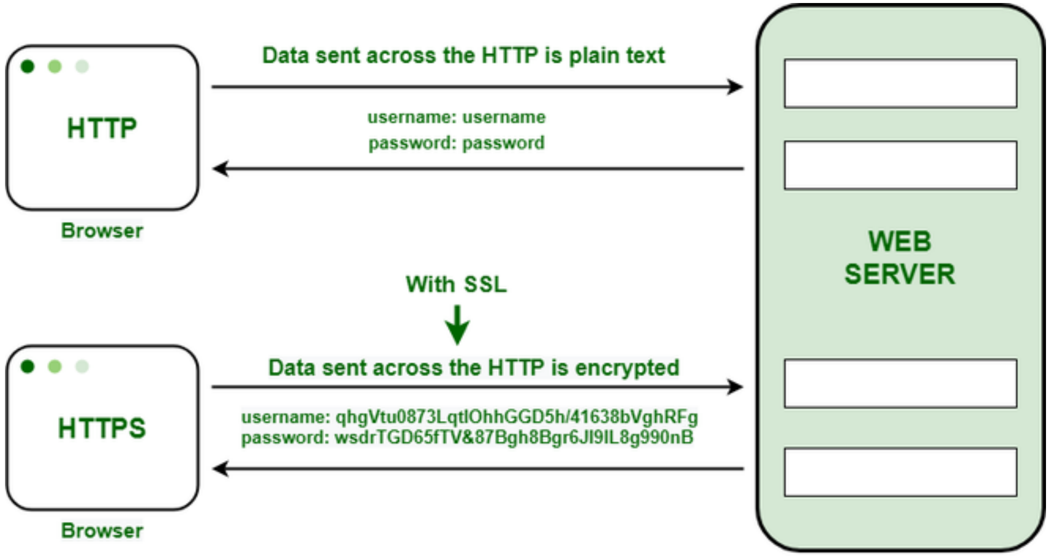
- The application layer is the top layer in both the OSI and TCP/IP models, and it encompasses protocols that enable direct communication with end-user applications. Here are some widely used application layer protocols:
- **HTTP/HTTPS:**
  - **HTTP (Hypertext Transfer Protocol)** is used for transferring web pages and data over the World Wide Web.
  - **HTTPS (HTTP Secure)** adds an encryption layer (SSL/TLS) for secure communication.
- **FTP/SFTP:**
  - **FTP (File Transfer Protocol)** is used for transferring files between clients and servers.
  - **SFTP (SSH File Transfer Protocol)** provides a secure alternative for file transfers over SSH.
- **Email Protocols:**
  - **SMTP (Simple Mail Transfer Protocol)** is used for sending emails.
  - **POP3 (Post Office Protocol 3)** and **IMAP (Internet Message Access Protocol)** are used for retrieving emails from a server.
- **DNS (Domain Name System):**
  - This protocol translates human-friendly domain names (e.g., www.example.com) into IP addresses that computers use to identify each other on the network.

Introduction (2/2)

- **Remote Access Protocols:**
  - **Telnet** allows remote command-line access to another computer (though it is largely replaced by more secure protocols).
  - **SSH (Secure Shell)** provides a secure method for remote login and command execution.
- **Network Management and Multimedia:**
  - **SNMP (Simple Network Management Protocol)** is used for network management, monitoring, and configuration.
  - **SIP (Session Initiation Protocol)** is used for establishing, managing, and terminating real-time communication sessions, like VoIP calls.
  - **RTP (Real-time Transport Protocol)** is used for delivering audio and video over IP networks.
- **Other Specialized Protocols:**
  - **LDAP (Lightweight Directory Access Protocol)** is used for accessing and maintaining distributed directory information services.
  - **NTP (Network Time Protocol)** is used for clock synchronization between computer systems.
  - **MQTT (Message Queuing Telemetry Transport)** and **CoAP (Constrained Application Protocol)** are examples of protocols designed for lightweight communication, often used in IoT (Internet of Things) applications.

Introduction to HTTP/HTTPS

- **What is HTTP?**
  - Stands for HyperText Transfer Protocol.
  - The foundation of data communication for the World Wide Web.
  - Operates on a simple request-response model over TCP/IP.
  - Designed as a stateless protocol, meaning each request is independent.
- **What is HTTPS?**
  - HTTPS is HTTP layered over TLS (or formerly SSL).
  - Provides a secure channel by encrypting data during transmission.
  - Uses digital certificates issued by trusted Certificate Authorities (CAs) to authenticate servers.
- **Evolution & Importance**
  - Originally, HTTP was used without encryption, which exposed sensitive data.
  - HTTPS emerged to protect user privacy and secure financial, personal, and confidential data online.


HTTP
Browser
Data sent across the HTTP is plain text
username: username
password: password
With SSL
HTTPS
Browser
Data sent across the HTTP is encrypted
username: qhgVtu0873LqtlOhhGGD5h/41638bVghRFg
password: wsdrTGD65fTV&87Bgh8Bgr6JI9IL8g990nB
WEB
SERVER

How HTTP/HTTPS Work

HTTP Communication Process

- **Step 1: DNS Lookup**
  The browser resolves the domain name to an IP address.
- **Step 2: TCP Connection**
  A TCP connection is established between the client and the server.
- **Step 3: Request & Response**
  The client sends an HTTP request (e.g., GET, POST) and the server replies with the appropriate response (HTML, JSON, etc.).
- **Statelessness**
  Each request is independent; no session information is retained between requests.
- **HTTPS Communication Process**

TLS Handshake

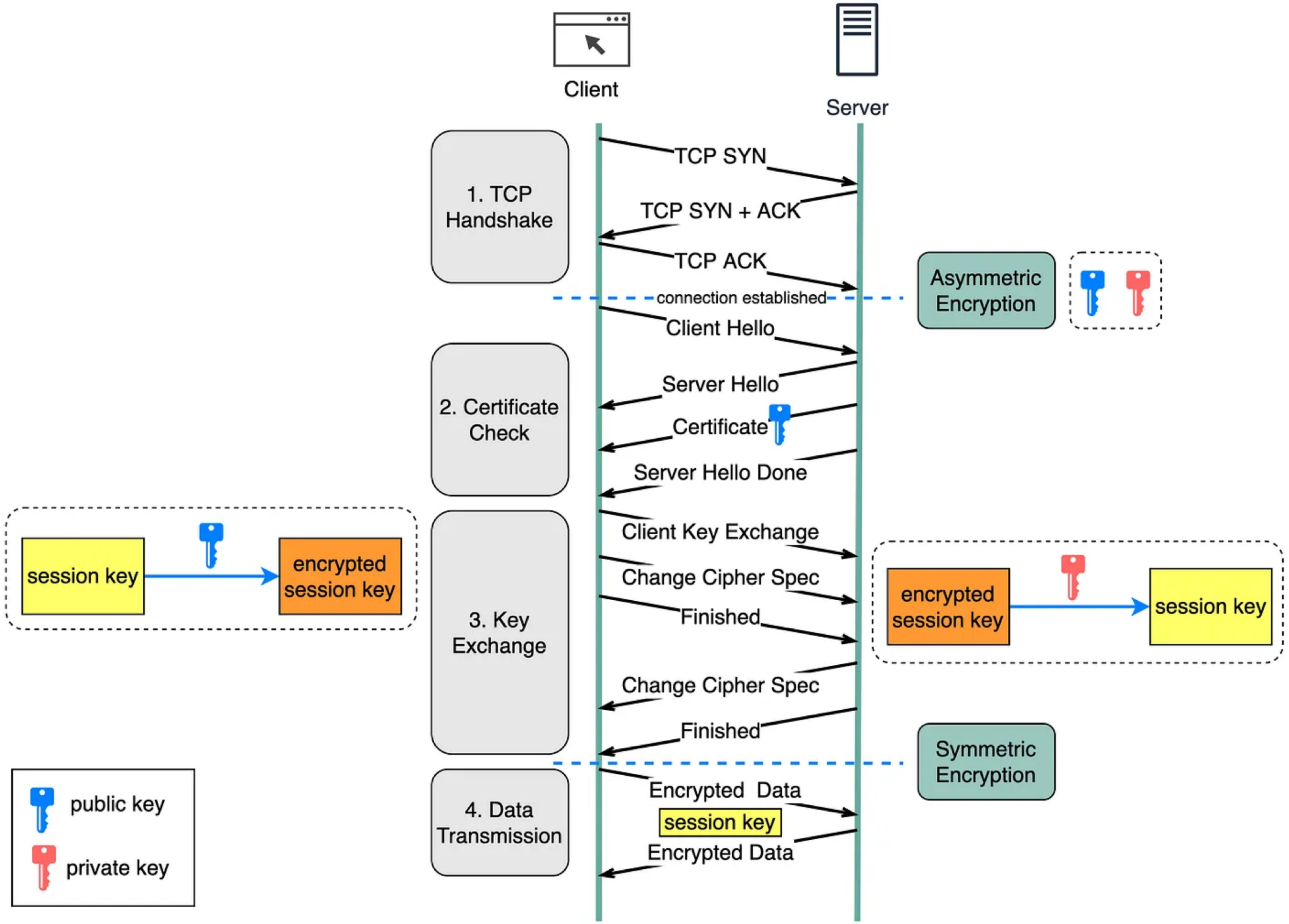
- The client and server negotiate encryption algorithms.
- The server presents its digital certificate to prove its identity.
- A secure session key is generated for encrypting data.

Encrypted Data Transfer
All subsequent data transfers between client and server are encrypted.

Connection Termination
The secure session ends when the connection is closed, ensuring no lingering session data.

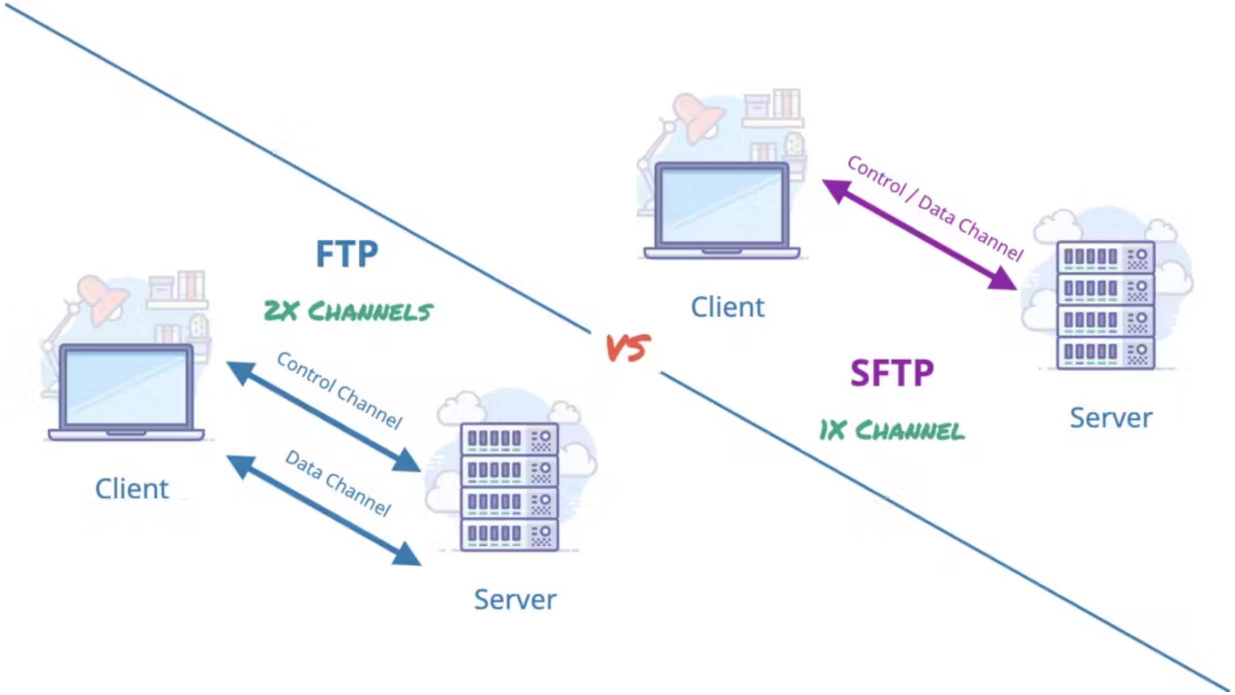
- **Key Differences**

- HTTP sends data in plain text, making it vulnerable to eavesdropping.
- HTTPS secures data via encryption, ensuring confidentiality, data integrity, and authentication.

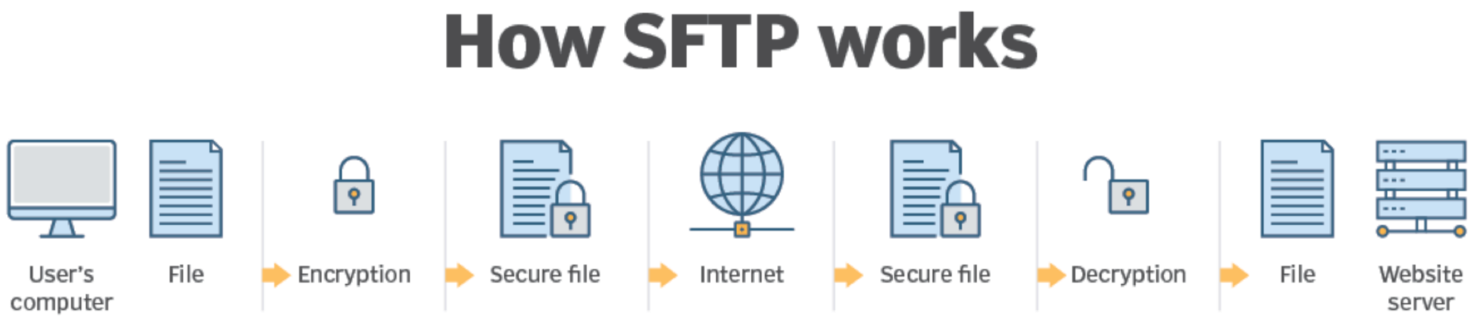


Introduction to FTP/SFTP

- **FTP (File Transfer Protocol)**
- **Overview:**
  Developed in the early days of the internet, FTP is designed to transfer files between a client and a server using a client-server architecture.
- **How It Operates:**
  Uses separate channels for control (commands) and data (file transfer). Typically operates on port 21 for control and port 20 for data.
- **Limitations:**
  Transmits data, including login credentials, in plaintext, making it vulnerable to eavesdropping and attacks.
- **SFTP (Secure File Transfer Protocol)**
- **Overview:**
  SFTP is not simply "FTP over SSH" but a distinct protocol built on the SSH (Secure Shell) framework, designed for secure file transfers.
- **How It Operates:**
  Uses a single encrypted connection over port 22 to transmit both commands and data, ensuring end-to-end security.
- **Key Advantages:**
  Encrypts all data in transit, provides robust authentication (e.g., password or key-based), and ensures data integrity.



How FTP and SFTP Work



- **FTP Communication Process**
  - **Control Connection:**
    - Client initiates a connection on port 21 to send commands (e.g., LIST, GET, PUT).
    - Operates using a stateless request-response model.
  - **Data Connection:**
    - A separate connection (usually on port 20) is established for transferring files.
    - Modes:
      - **Active Mode:** Server initiates the data connection back to the client.
      - **Passive Mode:** Client initiates both control and data connections.
  - **Key Note:**
    FTP's separate channels and lack of encryption make it unsuitable for transferring sensitive information.
- **SFTP Communication Process**
  - **Single Encrypted Connection:**
    - Leverages the SSH protocol to create a secure channel, handling both file commands and data transfer within one tunnel.
  - **Authentication & Security:**
    - Uses SSH methods (password-based or key-based) for authentication.
    - Encrypts all transmitted data, ensuring confidentiality and integrity.
  - **Operations:**
    - Supports file transfer, remote file management (e.g., rename, delete), and file access over a secure channel without the need for multiple ports.

DHCP Protocol Overview

- **DHCP** stands for *Dynamic Host Configuration Protocol*.
- Automates the assignment of IP addresses to devices.
- Simplifies network management by centralizing configuration.
- Provides additional network parameters (e.g., subnet mask, default gateway, DNS).

The DORA Process

1.Discover: The client broadcasts a DHCPDISCOVER message to locate available DHCP servers.

2.Offer: DHCP servers respond with a DHCPOFFER message containing an available IP address and configuration details.

3.Request: The client replies with a DHCPREQUEST message to accept one of the offered configurations.

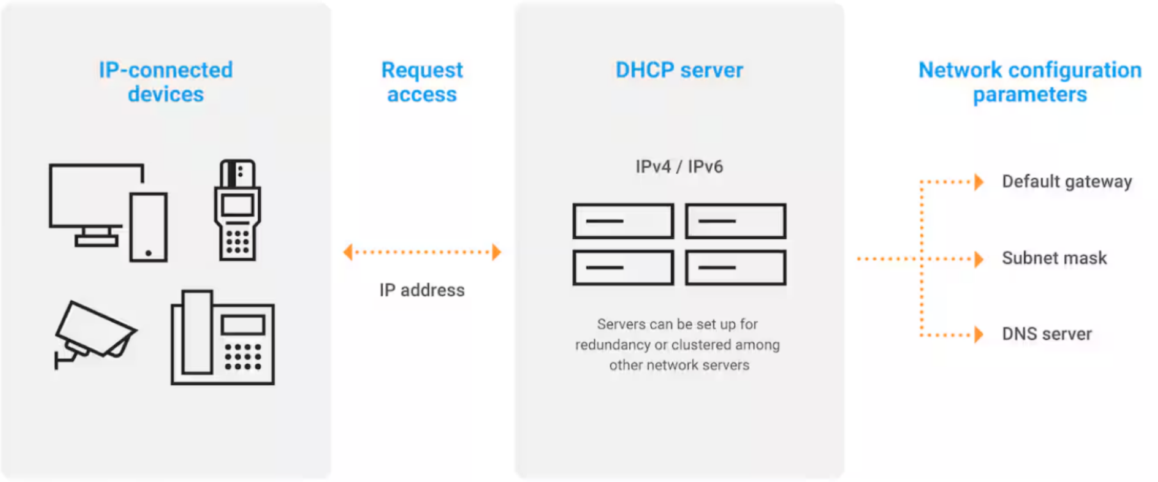
4.Acknowledge: The DHCP server finalizes the process by sending a DHCPACK message, confirming the lease.

DHCP Features and Benefits

•**Dynamic Allocation:** Automatically assigns IP addresses as devices join the network.

•**Lease Management:** IP addresses are leased for a specific duration, allowing reuse.

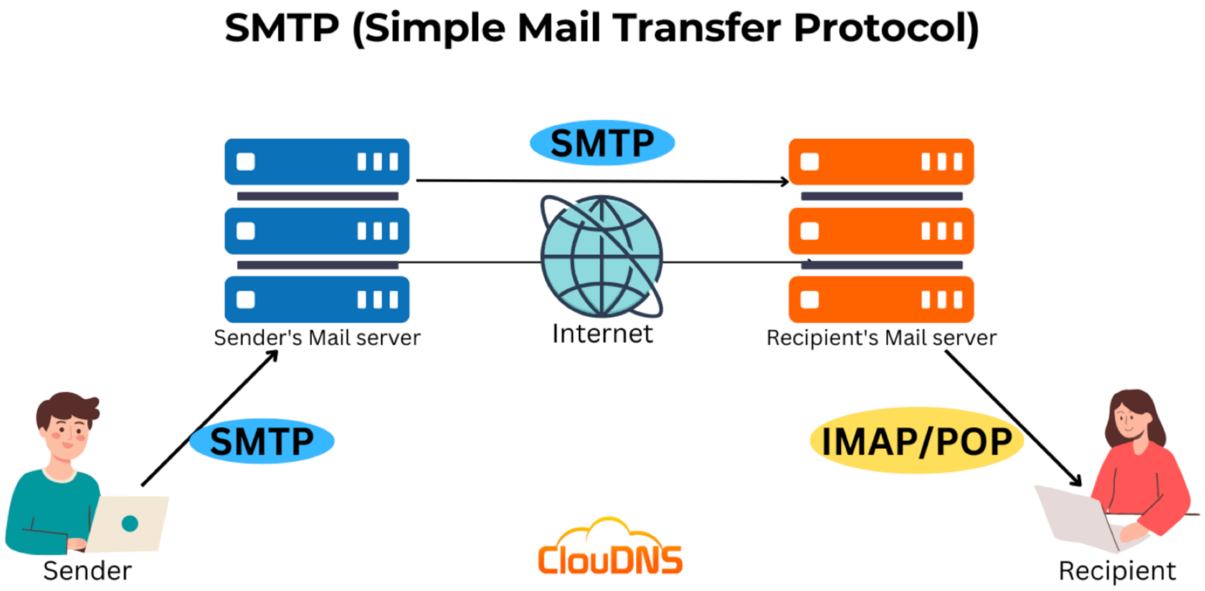
•**Centralized Control:** Simplifies administration of network settings across all devices.



IP-connected devices
Request access
IP address
DHCP server
IPv4 / IPv6
Servers can be set up for redundancy or clustered among other network servers
Network configuration parameters
Default gateway
Subnet mask
DNS server

In-Depth Look at SMTP

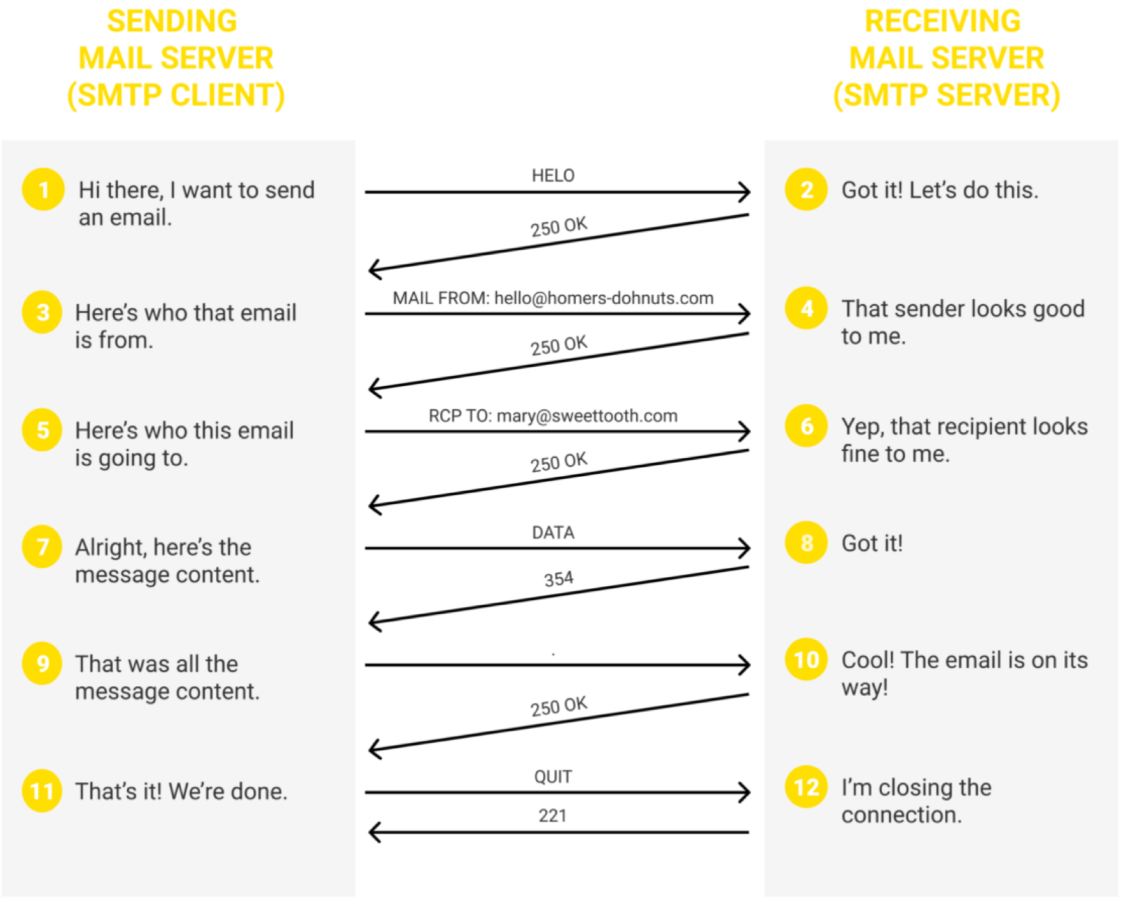
- **SMTP Fundamentals:**
  - **Primary Function:**
    Transfers outgoing emails from the sender's email client to the recipient's mail server.
  - **Communication Model:**
    Uses a push model—mail is pushed from one server to the next.
- **Technical Aspects:**
  - **Ports & Security:**
    - Commonly operates on port 25 for standard communication, port 465 or 587 for secure (encrypted) transmissions.
  - **Process Overview:**
    - The sender's client connects to the SMTP server.
    - The server verifies and routes the email to the recipient's mail server.
    - Includes handling of relay servers and error notifications.
- **Security Considerations:**
  - **Authentication:**
    Often requires user authentication to prevent unauthorized use.
  - **Encryption:**
    Use of TLS/SSL (when on secure ports) protects data in transit.
  - **Anti-Spam Measures:**
    Incorporates techniques such as SPF, DKIM, and DMARC to validate the sender's authenticity.



Basic SMTP commands

SMTP commands are a set of codes that power the transmission of email messages between servers. Here are the basic SMTP commands you should be aware of:

- **HELO or EHLO (Hello):** This is a crucial command for beginning the entire email sending process. The email client is identifying itself to the SMTP server. It is the beginning of a conversation and usually involves the server sending a HELO command back complete with its domain name/IP address.
- **MAIL FROM:** Following the identification command, the sender will share code that specifies who the mail is from. This outlines the email address and tells the SMTP server that a new transaction is about to start. From here, the server resets everything and is ready to accept the email address. Once accepted, it will reply with a 250 OK reply code.
- **RCPT TO (Recipient To):** The next command follows the 250 OK reply code identifying who the email is being sent to. Again, the SMTP server responds with the same code, at which point another RCPT TO command can be sent with a different recipient's email address. This can go back and forth as many times as required depending on how many people will receive the email.
- **DATA:** This triggers the transfer of data between the client and the server. All of the message contents will be moved to the SMTP server, which will respond with a 345 reply code. The contents of the messages are transferred to the server, where a single dot is sent in a line by itself to signal the end of the message. If accepted and ready for delivery, the server sends another 250 OK code. At this point, the message is on its way to the recipients.
- **QUIT:** When the email has been sent, the client sends the QUIT command to the server, severing the connection. If it has been successfully closed, the server will reply with a 221 code.
- **RSET (Reset):** This command is sent to the server when the mail transaction needs to be aborted. It doesn't close the connection, but it does reset everything and remove all previous data about the email and the parties involved. You will commonly use this when there has been an error, like inputting the wrong recipient information, and the process needs to be restarted.

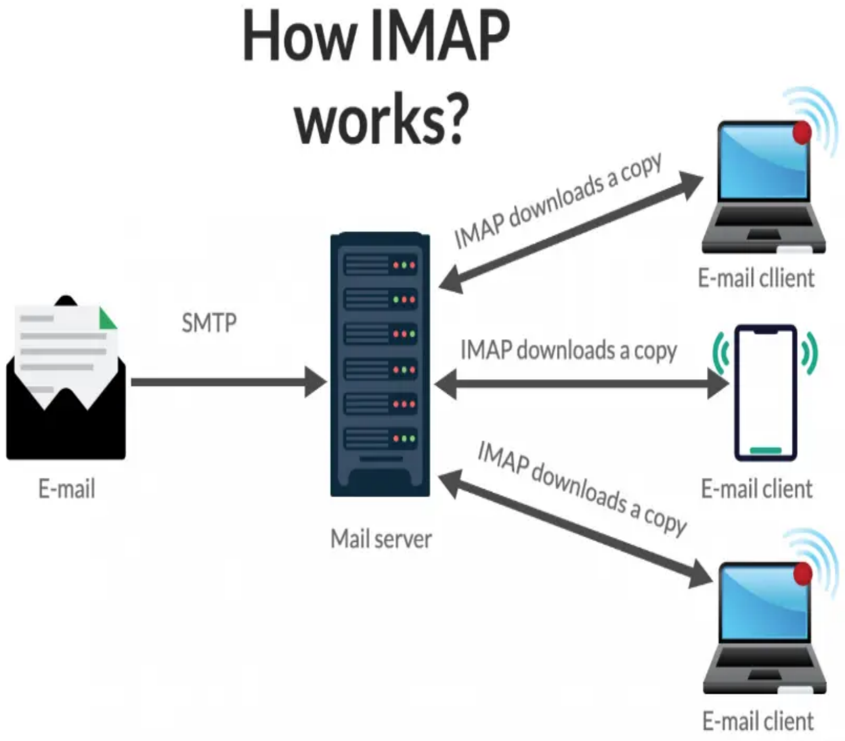

SENDING MAIL SERVER (SMTP CLIENT)
RECEIVING MAIL SERVER (SMTP SERVER)
1 Hi there, I want to send an email.
HELO
2 Got it! Let's do this.
250 OK
3 Here's who that email is from.
MAIL FROM: hello@homers-dohnuts.com
4 That sender looks good to me.
250 OK
5 Here's who this email is going to.
RCP TO: mary@sweettooth.com
6 Yep, that recipient looks fine to me.
250 OK
7 Alright, here's the message content.
DATA
8 Got it!
354
9 That was all the message content.
.
10 Cool! The email is on its way!
250 OK
11 That's it! We're done.
QUIT
12 I'm closing the connection.
221

POP3 and IMAP for Email Retrieval

- **POP3 (Post Office Protocol 3):**
- **Basic Functionality:**
  - Downloads emails from the server to the client.
  - Typically removes emails from the server after download (though many clients offer settings to leave copies behind).
- **Advantages & Limitations:**
  - Simplicity and offline access once downloaded.
  - Limited synchronization; not ideal for accessing emails across multiple devices.
- **IMAP (Internet Message Access Protocol):**
- **Basic Functionality:**
  - Allows users to view, manage, and organize emails directly on the server.
  - Maintains email states (read, unread, folder organization) across multiple devices.
- **Advantages & Limitations:**
  - Ideal for users accessing email from various locations or devices.
  - Requires a constant internet connection for optimal performance.
- **Choosing Between POP3 and IMAP:**

User Needs:

- POP3 for offline storage and simple email retrieval.
- IMAP for full synchronization, flexible organization, and server-side management.



Introduction to DNS

- DNS, which stands for Domain Name System, is a distributed naming system that allows us to assign user-friendly domain names to the numeric IP addresses associated with websites, servers, and other network resources. Without DNS, we would need to remember complex IP addresses like “192.168.1.1” instead of simple domain names like www.google.com.

- DNS is an essential part of the internet’s infrastructure and is often referred to as the “internet’s address book” because it provides the means to map human-readable domain names to the IP addresses that machines use to communicate with each other.

Components of DNS - DNS Servers

The DNS system consists of several key components, each serving a specific role in the process of resolving domain names to IP addresses:

- DNS Servers: DNS servers are specialized computers that store databases of domain names and their corresponding IP addresses. There are different types of DNS servers, including:
- **Root Servers:** These servers are at the top of the DNS hierarchy and store information about the top-level domains (TLDs) like .com, .org, and .net.
- **Top-Level Domain (TLD) Servers:** These servers manage domain names within specific TLDs (e.g., .com, .org, .gov).
- **Authoritative Name Servers:** These servers store DNS records for specific domains. Each domain typically has one or more authoritative name servers.
- **Recursive DNS Servers:** Also known as resolver servers, these servers interact with clients to resolve domain names by recursively querying other DNS servers until they find the authoritative server for a given domain.

Common DNS Records

DNS records are used to store various types of information associated with a domain. Here are some common DNS record types and their purposes:

- A Record (Address Record)

Purpose: Maps a domain name to an IPv4 address.

Example: If you have an A record that maps “www.example.com” to “192.168.1.1,” it means that “www.example.com” points to the server with the IPv4 address 192.168.1.1.

- AAAA Record (IPv6 Address Record)

Purpose: Maps a domain name to an IPv6 address.

Example: Similar to the A record, but for IPv6. It associates a domain name with a 128-bit IPv6 address.

- CNAME Record (Canonical Name Record)

Purpose: Creates an alias or nickname for an existing domain name. It allows one domain to point to another domain.

Example: You can create a CNAME record that maps “blog.example.com” to “www.example.com.” Now, both addresses point to the same location.

- MX Record (Mail Exchanger Record)

Purpose: Specifies the mail servers responsible for receiving email on behalf of a domain.

Example: An MX record for “example.com” might point to the mail server “mail.example.com,” indicating where email for that domain should be delivered.

- TXT Record (Text Record)

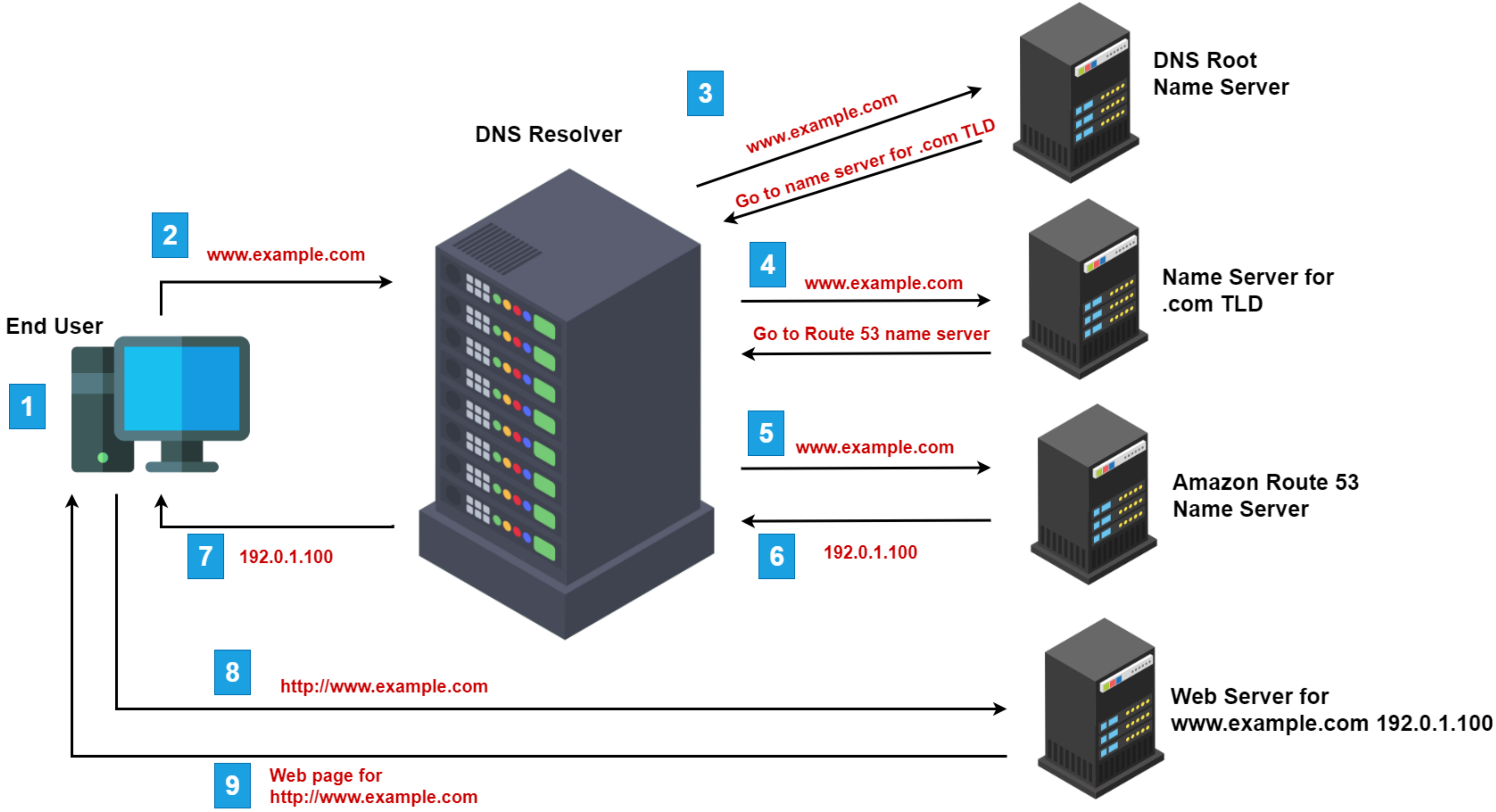
Purpose: Allows domain owners to add arbitrary text information to a domain’s DNS record.

Example: TXT records are commonly used for domain verification, email authentication (SPF, DKIM), and other purposes. For example, a TXT record may contain a verification code provided by a domain registrar.

- NS Record (Name Server Record)

Purpose: Specifies the authoritative name servers for a domain.

Example: An NS record for “example.com” would point to the authoritative DNS servers that hold DNS records for that domain.



1
End User
2
www.example.com
DNS Resolver
3
www.example.com
Go to name server for .com TLD
DNS Root
Name Server
4
www.example.com
Go to Route 53 name server
Name Server for
.com TLD
5
www.example.com
6
192.0.1.100
Amazon Route 53
Name Server
7
192.0.1.100
8
http://www.example.com
9
Web page for
http://www.example.com
Web Server for
www.example.com 192.0.1.100

How DNS Works

- DNS operates as a hierarchical and distributed system. When you enter a URL into your web browser's address bar and press Enter, your device initiates a DNS query to resolve the domain name into an IP address. Here's a simplified overview of how DNS works:

1. **Local DNS Cache:** Your device checks its local DNS cache to see if it already knows the IP address for the domain. If it's not in the cache or has expired, it proceeds to the next step.
2. **Recursive DNS Server:** Your device sends a request to a recursive DNS server provided by your internet service provider (ISP) or a third-party DNS resolver like Google's 8.8.8.8.
3. **Iterative Query:** The recursive DNS server may not have the IP address for the requested domain in its cache either. In this case, it begins an iterative query by contacting one of the root DNS servers.
4. **Root DNS Server:** The root DNS server responds to the query with a referral to the appropriate TLD server based on the top-level domain of the requested domain name. For example, if you requested "www.example.com," it would refer to the .com TLD server.
5. **TLD DNS Server:** The TLD server provides a referral to the authoritative name server for the requested domain, which is responsible for storing the actual IP address associated with the domain.
6. **Authoritative DNS Server:** The recursive DNS server contacts the authoritative DNS server for the specific domain (e.g., "example.com") and requests the IP address associated with "www.example.com."
7. **Response:** The authoritative DNS server responds with the IP address, and the recursive DNS server caches this information for future requests.
8. **Local Cache Update:** The recursive DNS server sends the IP address back to your device, which also caches the information locally to speed up future requests.
9. **Access the Website:** Your device now uses the obtained IP address to establish a connection with the webserver hosting the website associated with the domain name you entered. The webserver responds by serving the requested web page.

Introduction to TELNET

- **TELNET** stands for Teletype Network. It is a **client/server application protocol** that provides access to virtual terminals of remote systems on local area networks or the Internet. The local computer uses a telnet client program and the remote computers use a telnet server program.
- The **Telnet protocol** originated in the late 1960s, it was created to provide remote terminal access and control over mainframes and minicomputers. Initially, it was designed to be a simple and secure method of connecting to a remote system. This protocol allowed users to access remote computers using a terminal or command-line interface. Over time, Telnet’s use has diminished due to security concerns, and alternatives like **SSH** are now preferred for secure remote management
- **Logging in TELNET**

The logging process can be further categorized into two parts:

❖Local Login

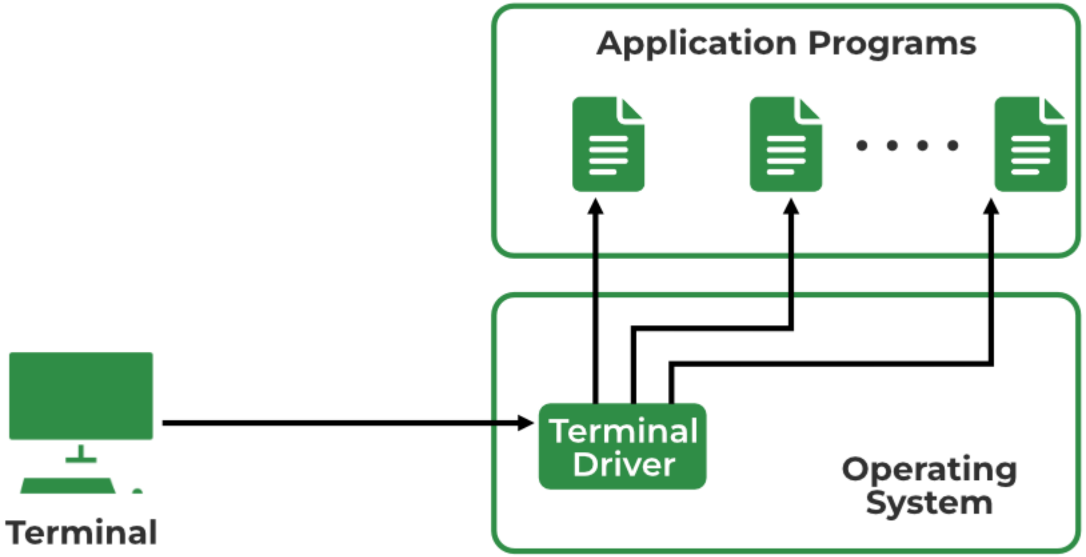
❖Remote Login

TELNET - 1. Local Login

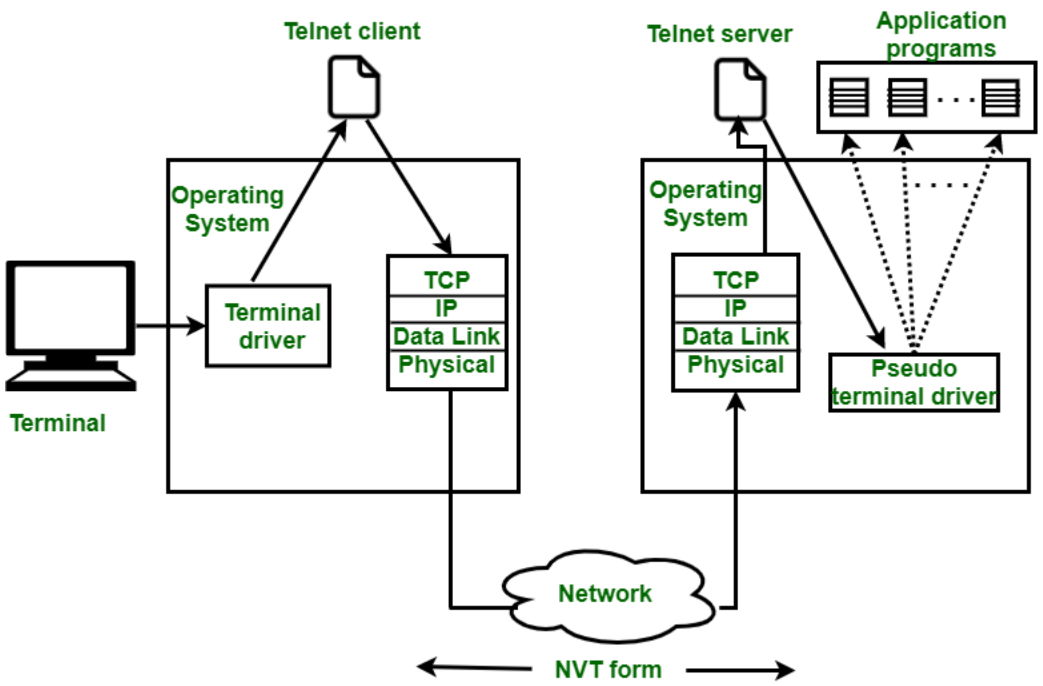
- Whenever a user logs into its local system, it is known as local login.

The Procedure of Local Login

- Keystrokes are accepted by the terminal driver when the user types at the terminal.
- Terminal Driver passes these characters to OS.
- Now, OS validates the combination of characters and opens the required application.



TELNET - 2. Remote Login



- Remote Login is a process in which users can log in to a remote site i.e. computer and use services that are available on the remote computer. With the help of remote login, a user is able to understand the result of transferring the result of processing from the remote computer to the local computer.

The Procedure of Remote Login

•When the user types something on the local computer, the local operating system accepts the character.

•The local computer does not interpret the characters, it will send them to the TELNET client.

•TELNET client transforms these characters to a universal character set called Network Virtual Terminal (NVT) characters and it will pass them to the local TCP/IP protocol Stack.

•Commands or text which are in the form of NVT, travel through the Internet and it will arrive at the TCP/IP stack at the remote computer.

•Characters are then delivered to the operating system and later on passed to the TELNET server.

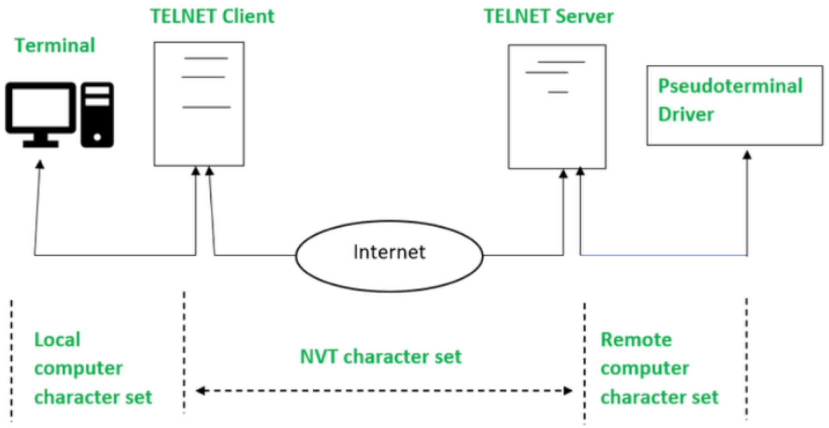
•Then TELNET server changes those characters to characters that can be understandable by a remote computer.

•The remote operating system receives characters from a pseudo-terminal driver, which is a piece of software that pretends that characters are coming from a terminal.

•The operating system then passes the character to the appropriate application program.

How TELNET Works?

Network Virtual Terminal(NVT) in Telnet



- **Client-Server Interaction**
  - The **Telnet client** initiates the connection by sending requests to the Telnet server.
  - Once the connection is established, the client can send **commands** to the server.
  - The server processes these commands and responds accordingly.
- **Character Flow**
  - When the user types on the **local computer**, the local operating system accepts the characters.
  - The Telnet client transforms these characters into a universal character set called **Network Virtual Terminal (NVT)** characters.
  - These NVT characters travel through the Internet to the remote computer via the local TCP/IP protocol stack.
  - The remote Telnet server converts these characters into a format understandable by the remote computer.
  - The remote operating system receives the characters from a pseudo-terminal driver and passes them to the appropriate application program[3].
- **Network Virtual Terminal (NVT)**
  - NVT is a virtual terminal in Telnet that provides a common structure shared by different types of real terminals.
  - It ensures communication compatibility between various terminals with different operating systems.

Frequently Asked Questions on TELNET

- **What is Telnet used for?**

Terminal programs typically use TELNET to allow you to log into a remote host.

- **Is Telnet TCP or UDP?**

A TELNET connection is a Transmission Control Protocol (**TCP**).

- **Is Telnet secure?**

Telnet is inherently insecure. Credential information (usernames and passwords) supplied over telnet is not encrypted and hence vulnerable to identity theft.

- **Which layer is Telnet?**

TELNET is an application layer protocol of OSI model.

What is SSH (Secure Shell) and How Does It Work?

- The **SSH (Secure Shell)** is an access credential that is used in the SSH Protocol. In other words, it is a cryptographic network protocol that is used for transferring encrypted data over the network. The port number of SSH is 22. It allow users to connect with server, without having to remember or enter password for each system. It always comes in key pairs:
- **Public key –** Everyone can see it, no need to protect it. (for encryption function).
- **Private key –** Stays in computer, must be protected. (for decryption function).

Key pairs can be of the following types:

- **User Key –** If the public key and private key remain with the user.
- **Host Key –** If public key and private key are on a remote system.
- **Session key –** Used when a large amount of data is to be transmitted.

What is the Secure Shell Key?

- Secure Shell or SSH, is a protocol that allows you to connect securely to another computer over an unsecured network. It developed in 1995. SSH was designed to replace older methods like Telnet, which transmitted data in plain text.
- Imagine a system administrator working from home who needs to manage a remote server at a company data center. Without SSH, they would have to worry about their login credentials being intercepted, leaving the server vulnerable to hackers. Instead of it after using SSH, the administrator establishes a secure connection that encrypts all data sent over the internet. They can now log in with their username and a private key, allowing them to safely execute commands on the server, transfer files, and make necessary updates, all of these without the risk of spying eyes watching their actions. This secure access is essential for maintaining the integrity of sensitive information of the company. **SSH (Secure Shell)** is an access credential that is used in the SSH Protocol. In other words, it is a cryptographic network protocol that is used for transferring encrypted data over the network.

Features and Functions of SSH

- **Features of SSH**
- **Encryption:** Encrypted data is exchanged between the server and client, which ensures confidentiality and prevents unauthorized attacks on the system.
- **Authentication:** For authentication, SSH uses public and private key pairs which provide more security than traditional password authentication.
- **Data Integrity:** SSH provides Data Integrity of the message exchanged during the communication.
- **Tunneling:** Through SSH we can create secure tunnels for forwarding network connections over encrypted channels.

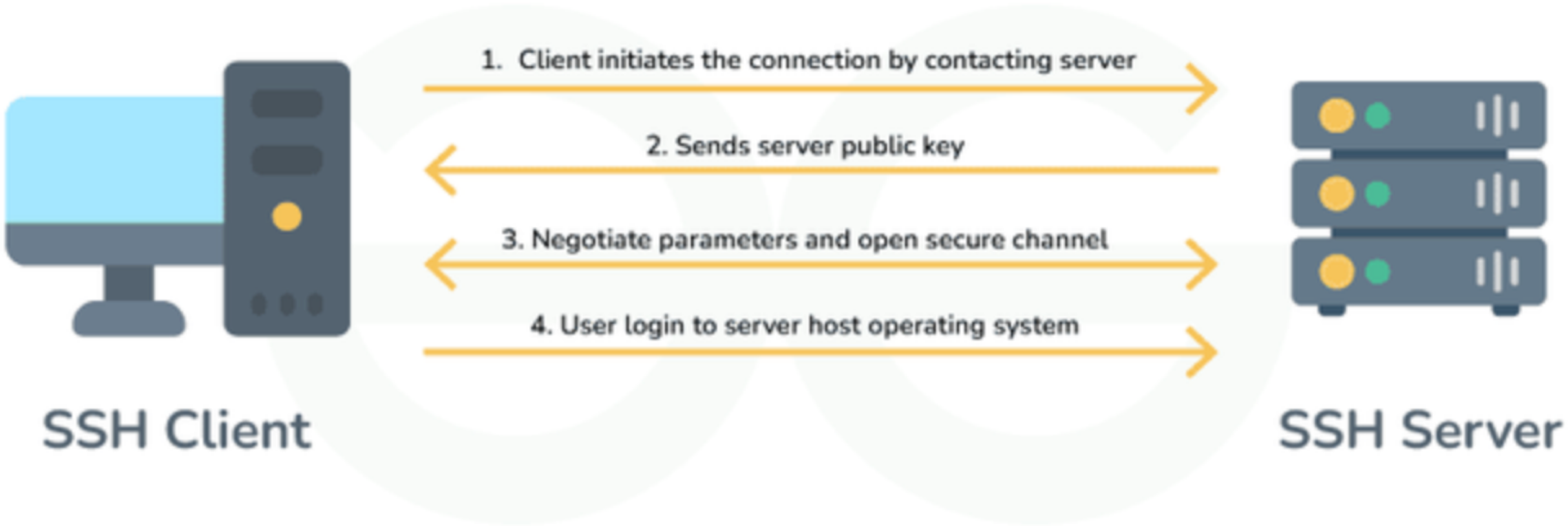
- **SSH Functions**

There are multiple functions performed by SSH Function, here below are some functions:

- SSH provides high security as it encrypts all messages of communication between client and server.
- SSH provides confidentiality.
- SSH allows remote login, hence is a better alternative to TELNET.
- SSH provides a secure File Transfer Protocol, which means we can transfer files over the Internet securely.
- SSH supports tunneling which provides more secure connection communication.

SSH Protocol

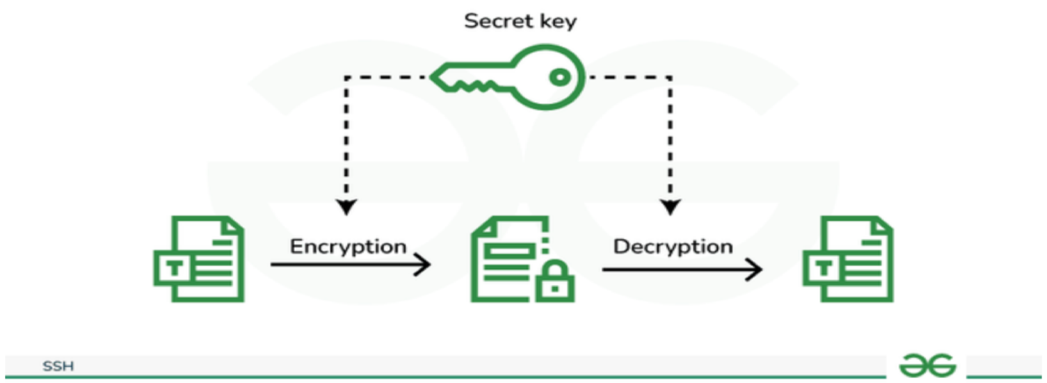
- To provide security between a client and a server the SSH protocol uses encryption. All user authentication and file transfers are encrypted to protect the network against attacks.


1. Client initiates the connection by contacting server
2. Sends server public key
3. Negotiate parameters and open secure channel
4. User login to server host operating system
SSH Client
SSH Server

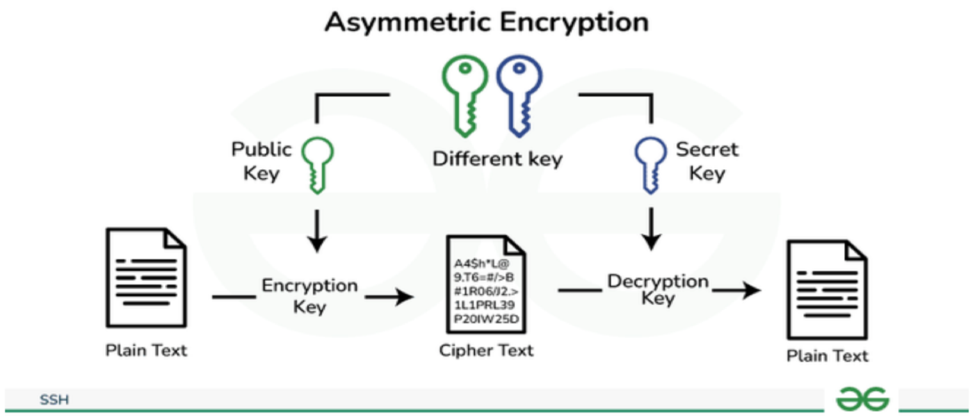
Techniques Used in SSH

There are majorly three major techniques used in SSH, which are

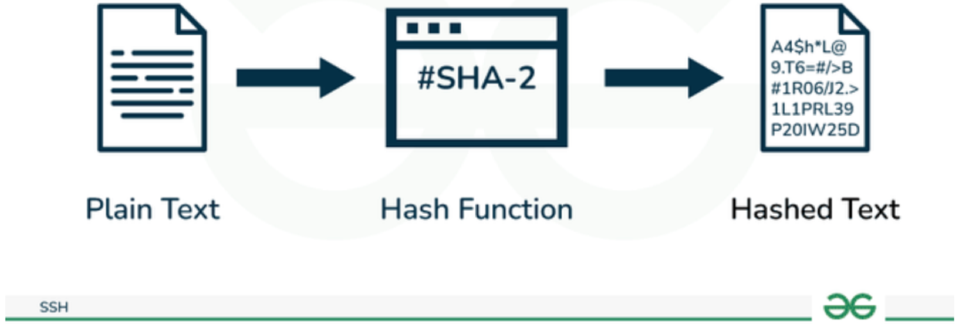
- **Symmetric Cryptography:** In Symmetric key cryptography the same key used for encrypting and decrypting the message, a unique single shared key is kept between the sender and reciever. For ex: DES (Data Encryption Standard) and AES (Advanced Encryption Standard).
- **Asymmetric Cryptography:** In Asymmetric key cryptography the key used for encrypting is different from the key used for decrypting the message. For ex: RSA (Rivest–Shamir–Adleman) and Digital Signature Algorithm.
- **Hashing:** Hashing is a procedure used in cryptography which convert variable length string to a fixed length string, this fixed length value is called hash value which is generated by hash function.



Symmetric Cryptography



Asymmetric Cryptography



Hashing