Computer Network

                          REPORT ON

**WHAT HAPPENS WHEN YOU TYPE FACEBOOK.COM AND HIT ENTER**

**Kalpana Acharya(PAS078BEI016)**

**Sharon Adhikari(PAS078BEI037)**

**Subina Chhetri(PAS078BEI039)**

**Swostika Poudel(PAS078BEI043)**

Table of Contents

[Introduction 3](#_Toc173867087)

[URL Parsing 3](#_Toc173867088)

[Components of a URL 3](#_Toc173867089)

[DNS Lookup Process 3](#_Toc173867090)

[Browser Cache 3](#_Toc173867091)

[Operating System Cache 3](#_Toc173867092)

[Router Cache 4](#_Toc173867093)

[ISP DNS Cache 4](#_Toc173867094)

[Recursive DNS Lookup 4](#_Toc173867095)

[DNS Recursive Resolver 4](#_Toc173867096)

[DNS Root Name server 4](#_Toc173867097)

[TLD Nameserver 4](#_Toc173867098)

[Authoritative Nameserver 4](#_Toc173867099)

[Establishing a Connection Opening a Socket 4](#_Toc173867100)

[TCP Handshake 5](#_Toc173867101)

[SYN Packet: 5](#_Toc173867102)

[SYN-ACK Packet: 5](#_Toc173867103)

[ACK Packet: 5](#_Toc173867104)

[Encapsulation into Frames: 5](#_Toc173867105)

[HTTP/HTTPS Request 6](#_Toc173867106)

[Secure Communication Using HTTPS: 6](#_Toc173867107)

[Request Message 6](#_Toc173867108)

[Processing the Request: 6](#_Toc173867109)

[Fetching Data: 6](#_Toc173867110)

[Sending the Response: 7](#_Toc173867111)

[Server Response 7](#_Toc173867112)

[Data Transmission and Reception 7](#_Toc173867113)

[Rendering the Web Page 9](#_Toc173867114)

[Conclusion 10](#_Toc173867115)

[References 10](#_Toc173867116)

# Introduction

When a user clicks on facebook.com, a series of events is triggered, starting with the parsing of the URL and followed by a DNS lookup to resolve the domain name into an IP address. Once the IP address is obtained, the browser establishes a TCP/IP connection with the Facebook server. This connection involves a three-way handshake to ensure reliable communication. After establishing the connection, the browser sends an HTTP request to the server and receives an HTTP response. Finally, the browser processes the response and renders the Facebook homepage. Each of these steps is crucial in ensuring that the user can access the desired web page quickly and efficiently.

## URL Parsing

When a user types "facebook.com" into their browser and presses enter, the browser begins by parsing the URL into its fundamental components. This process ensures that each part of the URL is correctly identified and processed for subsequent steps.

### Components of a URL

1. **Protocol**: The protocol is the set of rules used for data transmission. In the case of Facebook, it is usually https://, indicating a secure communication channel.
2. **Domain**: The domain is the main address of the website. For Facebook, the domain is www.facebook.com. Domain name of the site is simply the human readable address you type into your web browser to visit a website, making it easier to remember than a string of numbers(IP Address).
3. **Port**: This is often implicit and not visible in the URL. For HTTPS, the default port is 443, while for HTTP, it is 80. In URL parsing, the port is an optional component that specifies the port number on the host to which the request is directed.
4. **Path**: This specifies the exact location of the resource on the server. For instance, if you visit https://www.facebook.com/profile, /profile is the path.
5. Resource and Path together specify the resource on the server we want to load.

## DNS Lookup Process

-Once the URL is parsed, the next step involves resolving the domain name to an IP address via the Domain Name System (DNS). This process includes multiple layers of caching and recursive queries to ensure quick and accurate resolution.

### Browser Cache

-Initially, the browser checks its own cache to see if it has a recent DNS record for www.facebook.com. If the record is found, the browser uses this cached IP address, bypassing further lookup steps.

### Operating System Cache

-If the browser cache does not have the needed DNS record, the browser queries the operating system's cache. The OS maintains its own cache of recent DNS queries.

### Router Cache

-Should the OS cache also lack the DNS record, the request is sent to the local router. Many routers keep their own DNS caches to expedite the lookup process for frequently visited domains.

### ISP DNS Cache

-If the router does not have the DNS information, the request is forwarded to the Internet Service Provider's (ISP) DNS server. ISPs maintain a cache of DNS records for their users, which helps reduce the load on the global DNS infrastructure and improves lookup speed.

### Recursive DNS Lookup

-If the ISP's DNS server does not have the required DNS record, it starts a recursive DNS lookup. This process involves querying multiple DNS servers in a hierarchical manner to resolve the domain name.

### DNS Recursive Resolver

- The recursive resolver is the first stage in this process. Acting as an intermediary, it queries other DNS servers on behalf of the client. It first checks its own cache, and if necessary, sends a request to one of the root DNS servers.

### DNS Root Name server

- Root nameservers are the highest level in the DNS hierarchy. There are 13 root name servers globally, each with multiple copies to ensure reliability and performance. While the root nameserver does not have the exact IP address for www.facebook.com, it knows the authoritative servers for the top-level domain (.com).

### TLD Nameserver

- The recursive resolver then queries a TLD name server responsible for .com domains. This server directs the resolver to the authoritative nameserver for facebook.com.

### Authoritative Nameserver

-The authoritative nameserver for facebook.com holds the DNS records for all subdomains of facebook.com, including www.facebook.com. It responds to the recursive resolver with the IP address of the Facebook server.

The recursive resolver caches this information and returns the IP address to the browser, completing the DNS lookup. This enables the browser to connect to the Facebook server using the resolved IP address.

## Establishing a Connection Opening a Socket

• **TCP Socket:** The browser requests a TCP socket connection to the IP address and appropriate port (443 for HTTPS, 80 for HTTP).

# TCP Handshake

• Three-Way Handshake:

## SYN Packet:

 - The client (browser) sends a SYN packet to the Facebook server.

-The client sends a SYN packet to the server to start the connection process. This packet contains a sequence number used to track the order of bytes sent..  
-The SYN flag in the TCP header is set to indicate that this packet is for initiating a  
   
connection.

## SYN-ACK Packet:

**-** The Facebook server responds with a SYN-ACK packet upon receiving the SYN packet.  
 - It synchronizes the server's sequence numbers with the client by including the server's own initial sequence number.  
- Both the SYN and ACK flags in the TCP header are set.

## ACK Packet:

-The client responds to the server’s SYN-ACK with an ACK packet.  
-This packet acknowledges the server’s SYN-ACK by setting the acknowledgment number to one more than the server's sequence number.  
 -The ACK flag is set in the TCP header, and the SYN flag is not set, indicating that the connection is now established.

# Encapsulation into Frames:

**What Happens**: The data that needs to be sent HTTP/HTTPS request) is encapsulated into a frame. This frame includes important information like source and destination MAC addresses (unique hardware addresses of network interfaces).

**Source MAC Address:** This is the MAC address of your computer's network interface.

**Destination MAC Address:** This is the MAC address of the next device in the network path (e.g., your router).

**Why It Matters:** The frame ensures that the data can be correctly addressed and managed on the local network. It includes error-checking information to ensure the data isn't corrupted during transmission.

# HTTP/HTTPS Request

## Secure Communication Using HTTPS:

**What Happens:** When you visit a secure website like Facebook, your browser uses HTTPS HyperText Transfer Protocol Secure) instead of plain HTTP. HTTPS ensures that all data sent between your browser and the website is encrypted.

**Encryption with SSL/TLS:** HTTPS uses SSL Secure Sockets Layer) or its successor TLS Transport Layer Security) to encrypt the data. This means that anyone intercepting the data cannot read it because it appears as gibberish without the proper decryption keys.

**Why It Matters:** Encryption ensures that sensitive information, such as your login credentials and personal data, is protected from eavesdroppers and hackers while being transmitted over the internet.

## Request Message

 Encrypted HTTP GET Request:

**What Happens**: Your browser prepares an HTTP GET request, which is a type of message that asks the server to send back the contents of a specific web page (in this case, the Facebook homepage). This GET request is encrypted using SSL/TLS.

**Sending the Request:** The browser sends this encrypted HTTP GET request to the IP address of Facebook's server

**Why It Matters:** By sending an encrypted request, your browser ensures that the communication remains private and secure, protecting the data from being tampered with or read by unauthorized parties.

## Processing the Request:

**What Happens:** When Facebook's server receives your encrypted HTTP GET request, it decrypts the request using its SSL/TLS keys. The server then processes the request to fetch the necessary data needed to display the Facebook homepage.

## Fetching Data:

This includes:  
 **HTML:** The structure and content of the web page.  
 **CSS:** The styling information (colors, fonts, layout).  
 **JavaScript**: Scripts that add interactivity and functionality to the page. **Images** Any pictures, icons, or graphics needed for the page.

**Why It Matters**: The server must gather all these resources to build and display the web page correctly on your browser.

## Sending the Response:

**What Happens:** Once the server has collected all the necessary resources, it creates an HTTP response message. This message contains the requested resources HTML, CSS, JavaScript, images, etc.). The server then encrypts the response using SSL/TLS to ensure secure communication back to your browser.

**Encryption:** Just like the request, the response is encrypted to protect the data as it travels over the internet, ensuring privacy and security.

**Why It Matters:** Encrypting the response ensures that the data remains secure and private while being transmitted back to your browser. This prevents eavesdropping and tampering by unauthorized parties.

## Server Response

* The server processes the request and sends back an HTTP response. This response includes:
  + **Status Line**: Indicates the HTTP version, status code (e.g., 200 OK, 404 Not Found), and a reason phrase.
  + **Headers**: Provide metadata such as Content-Type (e.g., text/html), Content-Length, Set-Cookie (for session management), and others.
  + **Body**: Contains the actual content requested, such as HTML, images, or JSON data.

These steps complete the process of sending an HTTP request and receiving an HTTP response, ultimately resulting in the rendering of the requested web page in the browser.

# Data Transmission and Reception

**Breaking Down the HTTP Response**: The HTTP response from Facebook's server is too large to be sent all at once. It's broken down into smaller packets.

**TCP Transmission Control Protocol):** TCP is responsible for ensuring that these packets are delivered reliably and in the correct order. It assigns sequence numbers to the packets so they can be reassembled correctly on the receiving end.

**Error Handling:** If any packets are lost or corrupted during transmission, TCP handles retransmissions, ensuring that all data arrives intact.

 TCP ensures that the entire HTTP response is delivered accurately and completely, even if the data is split across multiple packets.

**Adding IP Addresses**: Each packet is given a header that includes the source IP address Facebook's server) and the destination IP address (your computer). This allows the packets to be routed correctly across the internet.

**Routing:** The packets are sent through a series of routers and other network devices that use these IP addresses to determine the best path to your computer. IP addresses ensure that the packets are routed from Facebook's server to your computer across the complex network of the internet.

**Frame Creation:** At each hop along the way (e.g., from one router to another), the packets are encapsulated into frames, which include source and destination MAC addresses. These addresses are specific to each network interface on the local network.

**Error-Checking** Frames also include error-checking information to detect and correct any errors that occur during transmission.

**Transmission Over Physical Medium**:  
  Frames are converted into physical signals appropriate for the transmission medium:

**Ethernet**: Frames are converted into electrical signals that travel over copper cables.

**Fiber Optic:** Frames are converted into light pulses that travel through fiber optic cables.

**Wi-Fi:** Frames are converted into radio waves that travel through the air.

**Transmission and Reception:** These signals are sent over the physical medium, received by the next device in the network path, and converted back into frames and then packets.

- The Data Link and Physical Layers handle the actual physical transmission of data, ensuring that packets travel across the network medium and reach their destination.

# Rendering the Web Page

**Receiving the HTTP Response**: Your browser receives the encrypted HTTP response from Facebook's server. This response contains the HTML, CSS, JavaScript, and other resources needed to display the Facebook homepage.

**Decrypting the Response** The browser decrypts the response using SSL/TLS to access the contents securely.

**Parsing the Resources**:**HTML** The browser parses the HTML to understand the structure and content of the web page. This includes elements like headings, paragraphs, images, and links.

**CSS:** The browser processes the CSS to apply styles to the HTML elements, defining how they should look (colors, fonts, layout, etc.).

**JavaScript:** The browser executes any JavaScript code to add interactivity and functionality to the web page (e.g., loading new content without refreshing the page).

**Rendering the Page**:  
 The browser combines the parsed HTML, CSS, and JavaScript to render the Facebook homepage on your screen. This process involves constructing the Document Object Model DOM and applying styles and scripts to create the final visual representation.

 This step transforms the raw data from the server into a visually appealing and interactive web page that you can see and interact with.

**User Interaction Interacting with the Web Page**:

Once the Facebook homepage is rendered, you can interact with it in various ways, such as liking a post, posting a comment, or navigating to another page.

# Conclusion

-When a user clicks on facebook.com, a series of sophisticated steps occur behind the scenes to deliver the requested web page efficiently and reliably. The process begins with URL parsing, where the browser dissects the URL into its components. The DNS lookup then resolves the domain name into an IP address, involving multiple caching layers and recursive lookups.

Once the IP address is obtained, the browser establishes a TCP/IP connection with the Facebook server using a three-way handshake. This connection ensures reliable communication. The HTTP request is then sent to the server, which processes it and generates an appropriate HTTP response. Server-side load balancing mechanisms.

# References

1. **HTTP/1.1: Protocol Overview**

* **Fielding, R., et al. "Hypertext Transfer Protocol -- HTTP/1.1." RFC 2616, 1999.**
* **URL:** [**https://tools.ietf.org/html/rfc2616**](https://tools.ietf.org/html/rfc2616)

1. **DNS and BIND**

* **Albitz, P., & Liu, C. "DNS and BIND." O'Reilly Media, 5th Edition, 2006.**
* **URL:** [**https://www.oreilly.com/library/view/dns-and-bind/0596100574/**](https://www.oreilly.com/library/view/dns-and-bind/0596100574/)

1. **TCP/IP Illustrated, Volume 1: The Protocols**

* **Stevens, W. R. "TCP/IP Illustrated, Volume 1: The Protocols." Addison-Wesley, 1994.**
* **URL:** [**https://www.informit.com/store/tcp-ip-illustrated-volume-1-the-protocols-9780201633467**](https://www.informit.com/store/tcp-ip-illustrated-volume-1-the-protocols-9780201633467)

1. **Understanding Load Balancing in Networking**

* **Incapsula, "What is Load Balancing?" Imperva, 2021.**
* **URL:** [**https://www.imperva.com/learn/performance/load-balancing/**](https://www.imperva.com/learn/performance/load-balancing/)

1. **TLS and SSL Protocols**

* **Rescorla, E. "The Transport Layer Security (TLS) Protocol Version 1.3." RFC 8446, 2018.**
* **URL:** [**https://tools.ietf.org/html/rfc8446**](https://tools.ietf.org/html/rfc8446)