COMPUTER NETWORK

REPORT ON

WHAT HAPPENS WHEN YOU TYPE FACEBOOK.COM AND HIT ENTER

Submitted by:

ANISHA BHANDARI (PAS078BEI005)

SUCHANA SUBEDI (PAS078BEI040)

UTKARSHA GUPTA (PAS078BEI047)

SRIYANKA BARAL (PAS078BEI048)

Abstract

This report details the intricate series of events that occur when a user types "facebook.com" and presses Enter. It covers the entire process, from key press events to the rendering of the Facebook homepage, providing an in-depth analysis of the multiple layers of technology and communication protocols involved. These layers include URL parsing, DNS resolution, TCP and TLS handshakes, HTTP request and response, and page rendering. Each step is essential to ensure a smooth and secure browsing experience. By examining these processes, this report aims to demystify the complex interactions between hardware, software, and network systems that allow users to access web content effortlessly and securely. Understanding these mechanisms can provide valuable insights into the efficiency and security measures inherent in modern web browsing.

Contents

Introduction:

The act of typing "facebook.com" into a web browser and pressing Enter is deceptively simple on the surface. However, it initiates a complex series of processes that involve multiple layers of technology and intricate communication protocols. Each step, from the initial key press to the final rendering of the Facebook homepage, involves precise coordination between hardware components, operating systems, network infrastructure, and server-side applications.

This report aims to explore and elucidate these processes, providing a comprehensive breakdown of what happens behind the scenes. By understanding these mechanisms, we can appreciate the sophistication and efficiency of modern web browsing, which seamlessly integrates various technologies to deliver quick and secure access to web content.

From the moment a key is pressed, the journey to rendering a web page involves numerous checks and handshakes designed to ensure the reliability, security, and accuracy of the requested content. This report will delve into each stage of this journey, highlighting the role of each component and protocol involved in making a simple URL entry result in a fully rendered web page.

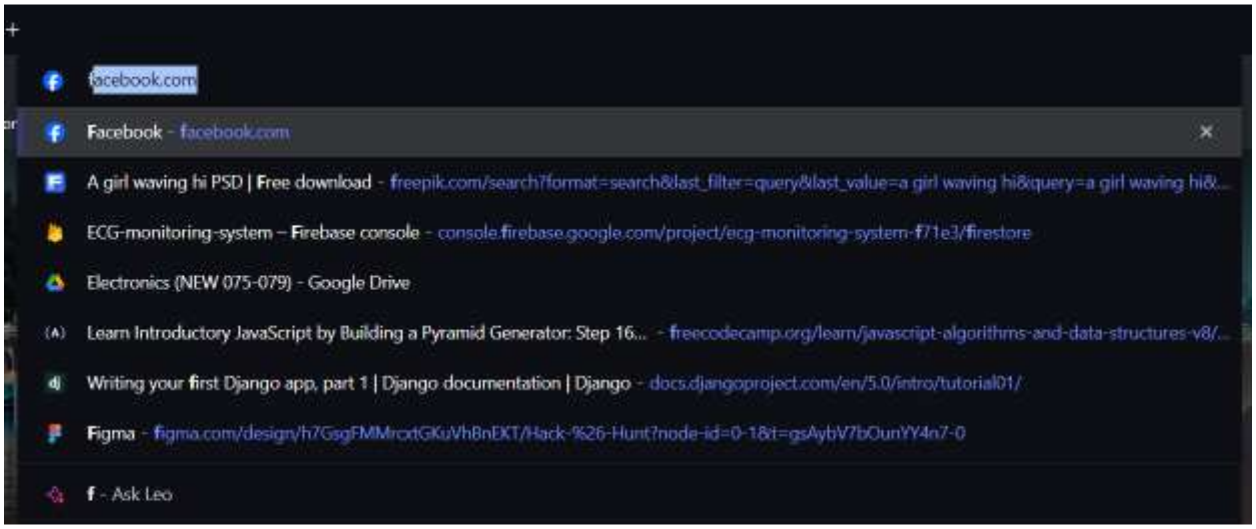
Purpose:

The purpose of this report is to examine the steps involved when a user types "facebook.com" and presses Enter. This action triggers multiple layers of technology, including hardware and software interactions during key presses, URL interpretation, security protocols like HTTP Strict Transport Security (HSTS), and DNS resolution. It also covers establishing secure connections through TCP and TLS handshakes, constructing and transmitting HTTP requests, server response preparation, and rendering the final web page. Additionally, it addresses handling additional resource requests. By understanding these processes, we gain insights into the efficiency, complexity, and security of modern web browsing, making these technical details accessible to all readers.

1. Key Press Events

The "f" Key is Pressed

- **Physical Action:** We press the "f" key on our keyboard.
- **Electrical Signal:** This action closes a specific circuit within the keyboard, generating an electrical signal.
- **Key Code Generation:** The controller converts this signal into a keycode (e.g., 70 for "f"), which is sent to our computer via USB, Bluetooth, or another connection method.
- **OS Handling:** The operating system (OS) receives this keycode and forwards it to the active application, which is our web browser.


facebook.com

Facebook - facebook.com

A girl waving hi PSD | Free download - freepik.com/search?format=search&last_filter=query&last_value=a girl waving hi&query=a girl waving hi&...

ECG-monitoring-system – Firebase console - console.firebase.google.com/project/ecg-monitoring-system-f71e3/firestore

Electronics (NEW 075-079) - Google Drive

Learn Introductory JavaScript by Building a Pyramid Generator: Step 16... - freecodecamp.org/learn/javascript-algorithms-and-data-structures-v8/...

Writing your first Django app, part 1 | Django documentation | Django - docs.djangoproject.com/en/5.0/intro/tutorial01/

Figma - figma.com/design/h7GsgFMMrcxtGKuVhBnEXT/Hack-%26-Hunt?node-id=0-1&t=gsAybV7bOunYY4n7-0

f - Ask Leo

Figure 1: Initial typing

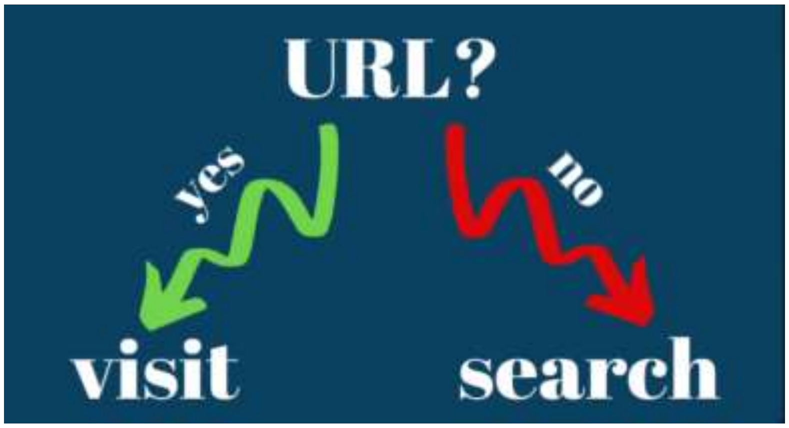
The "Enter" Key is Pressed

- **Physical Action:** We press the "Enter" key.
- **Electrical Signal:** An electrical circuit specific to the "Enter" key is closed, sending a signal.
- **Debouncing:** The signal is debounced by the keyboard controller.
- **Key Code Generation:** The controller generates a keycode (usually 13 for Enter) and sends it to the computer.
- **OS Handling:** The OS processes this keycode and sends it to the active application.

2. URL Parsing

Determining the Input

- **Input Evaluation:** The browser examines the entered text to determine whether it is a valid URL or a search term.
    - **URL:** If the input contains a protocol (e.g., "http://", "https://"), a domain name, or other URL-specific components, it is treated as a URL.
    - **Search Term:** If the input doesn't meet URL criteria, it is treated as a search query and sent to the default search engine (e.g., Google, Bing).



Handling Unicode

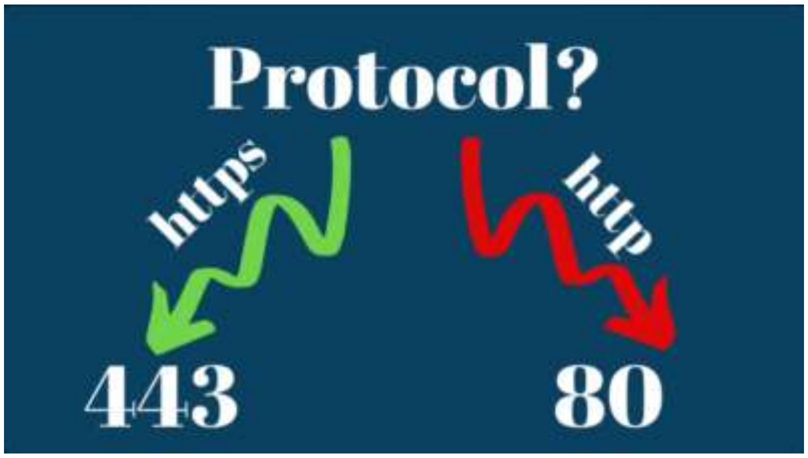
- **Internationalized Domain Names (IDN):** If the URL contains non-ASCII characters, the browser converts it to Punycode, ensuring compatibility with the DNS system.

3. HTTP or HTTPS?

HSTS (HTTP Strict Transport Security) Check

Preloaded HSTS List:

- The browser checks if "facebook.com" is in its preloaded HSTS list.
- **HSTS Policy:** If present, the browser enforces HTTPS for all connections to this domain.
- **Default Protocol:** If not in the HSTS list, the browser may initially try HTTP and then be redirected to HTTPS by the server.



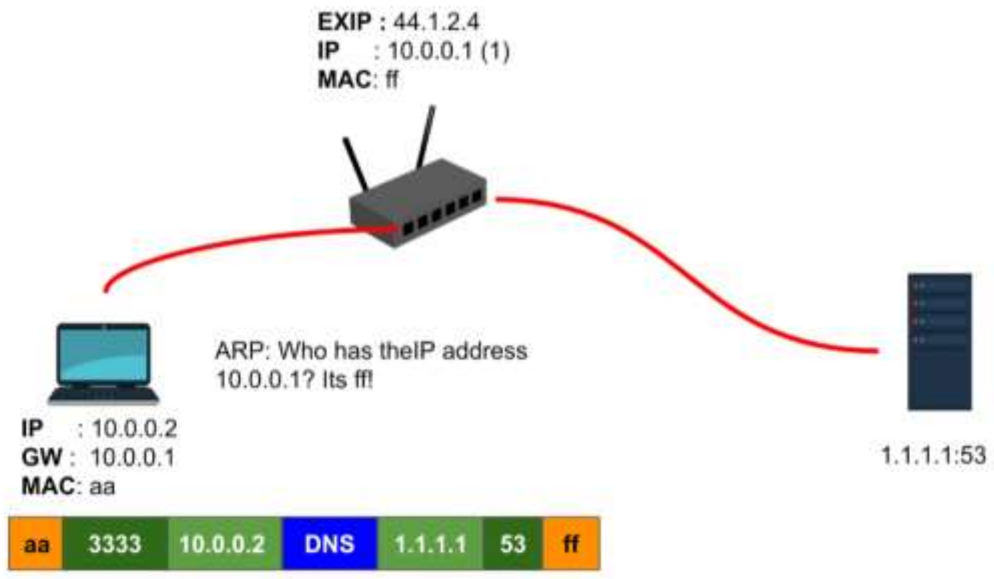
4. DNS Resolution

Checking Cache

- **Local DNS Cache:** The browser first checks its local cache for the IP address of "facebook.com" to reduce latency.

DNS Query

- **Hosts File:** If the address is not cached, the browser checks the local hosts file for a manual entry of the IP address.
- **DNS Server:** If not found in the hosts file, the browser sends a DNS query to the configured DNS resolver (often provided by our ISP or manually set).



DNS Query Process

Recursive DNS Query:

- **Root Servers:** The DNS resolver queries the root DNS servers to find the authoritative servers for the top-level domain (TLD) ".com".
- **TLD Servers:** The root servers respond with the addresses of the TLD servers for ".com".
- **Authoritative DNS Servers:** The DNS resolver queries the TLD servers, which respond with the authoritative DNS servers for "facebook.com".
- **Final Query:** The resolver queries the authoritative servers to get the IP address for "facebook.com".

ARP (Address Resolution Protocol)

- **MAC Address Resolution:** If the DNS resolver is on a different subnet, an ARP request is sent to determine the MAC address of the router or gateway.

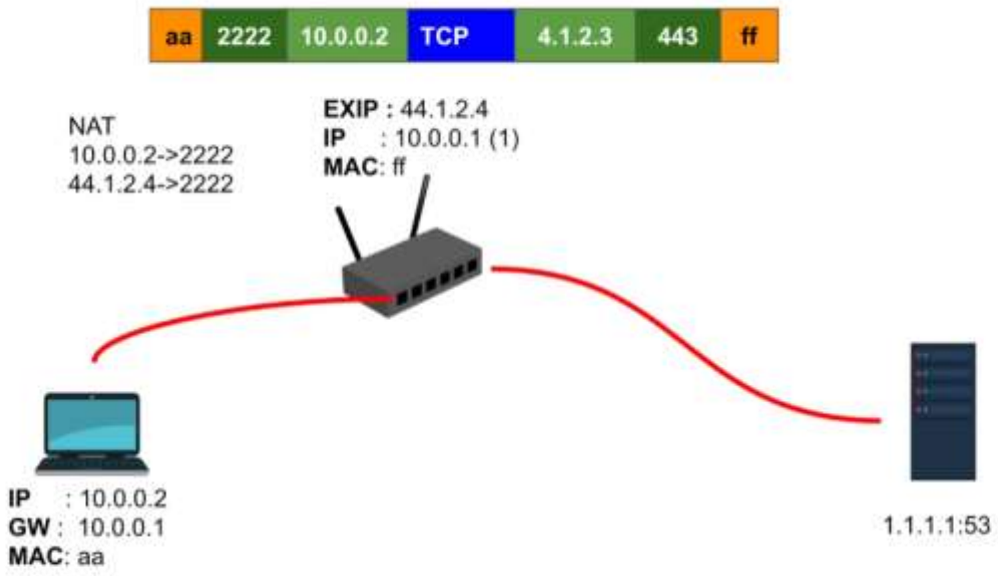
DNS Response

- **IP Address:** The DNS resolver returns the IP address (e.g., 157.240.22.35) of "facebook.com" to the browser.

5. 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).

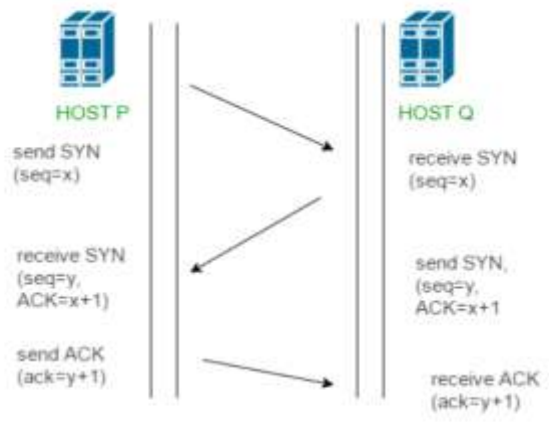


TCP Handshake

- **Three-Way Handshake:**
    - **SYN Packet:**
        - The client (browser) sends a SYN packet to the Facebook server.
        - The packet includes an initial sequence number (ISN), a randomly chosen number serving as the starting point for the sequence of bytes.
        - 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.
    - This packet acknowledges the client's SYN packet with an acknowledgment number one more than the client's sequence number.
    - 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.

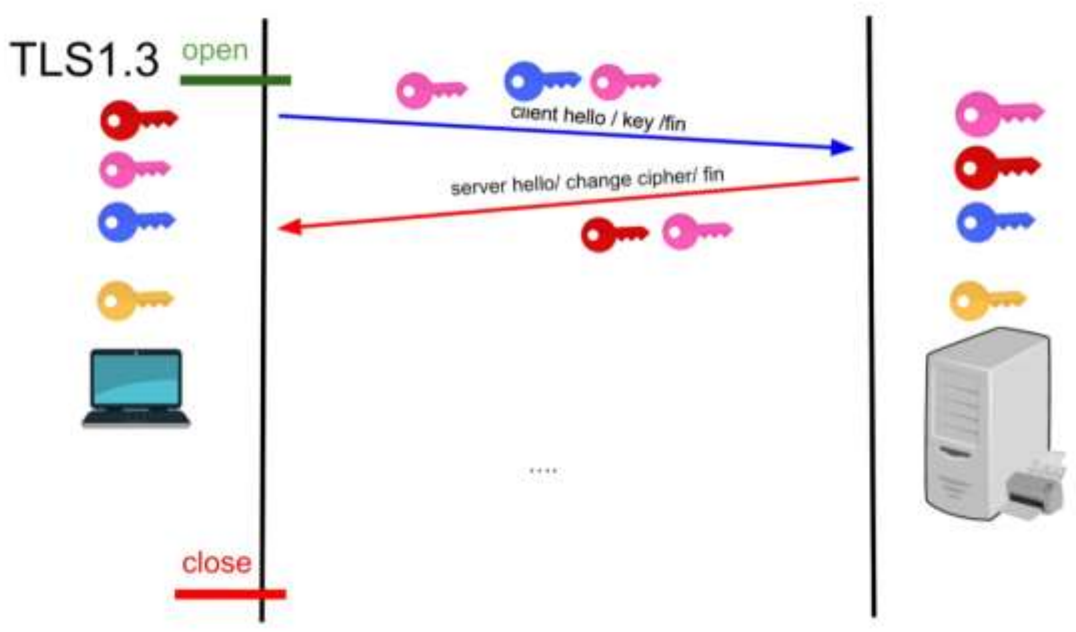
- **ACK Packet:**
    - The client responds to the server's SYN-ACK with an ACK packet.
    - This packet acknowledges the server's SYN-ACK by setting the acknowledgment number to one more than the server's sequence number.
    - The ACK flag is set in the TCP header, and the SYN flag is not set, indicating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



TLS Handshake (for HTTPS)

- **ClientHello:** Client send a ClientHello message to the server, specifying supported TLS versions and cipher suites.
- **ServerHello:** The server responds with a ServerHello message, selecting the TLS version and cipher suite, and sends its digital certificate.

- **Certificate Verification:** Client verify the server's certificate using a chain of trust up to a trusted Certificate Authority (CA).
- **Key Exchange:** Browser and the server exchange keys to establish a secure session.
- **Session Establishment:** Both parties agree on session keys, completing the TLS handshake.



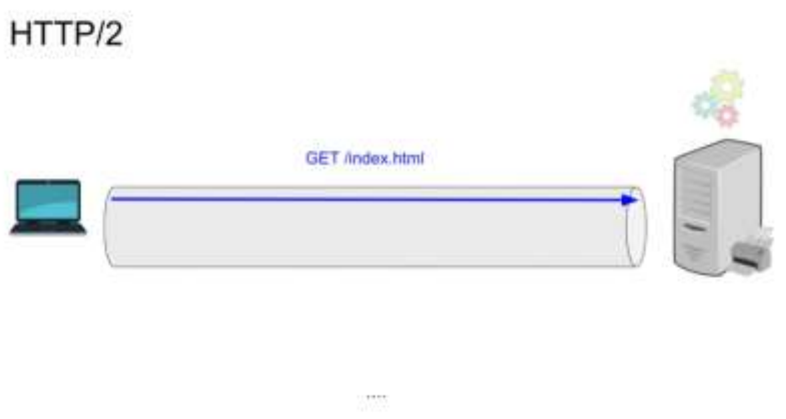
6. Sending the HTTP Request

Crafting the Request

- **HTTP GET Request:** We construct an HTTP GET request for "facebook.com".

Sending the Request

- **Packet Segmentation:** The HTTP request is broken into TCP segments and sent to the server.



7. Server-Side Processing

Receiving the Request

- **HTTP Server:** The server receives the request through its HTTP server software (e.g., Nginx, Apache).
- **Virtual Host:** The server determines the appropriate virtual host (facebook.com) based on the "Host" header.

Request Handling

- **Method Verification:** The server checks that the HTTP method (GET) is supported.
- **Security Checks:** The server performs security checks, such as IP whitelisting, rate limiting, and access control.
- **Rewriting Rules:** The server applies URL rewriting rules if configured (e.g., redirecting HTTP to HTTPS).

Content Retrieval

- **Static Content:** The server retrieves the requested static file (e.g., index.html) from the file system.
- **Dynamic Content:** For dynamic content, the server processes scripts or applications (e.g., PHP, Python, Node.js) to generate the HTML response.

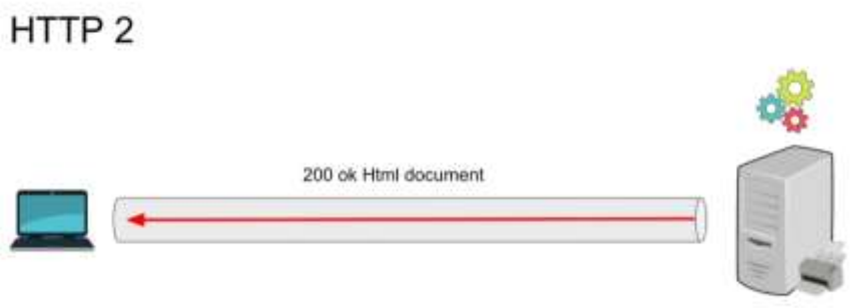
8. Sending the Response

Crafting the Response

- **HTTP Response:** The server constructs an HTTP response with the requested content.

Transporting the Response

- **TCP Segmentation:** The response is segmented into TCP packets and sent to browser.
- **ACKs:** Client acknowledge each received segment to ensure reliable delivery.



9. Rendering the Page

Receiving the Response

- **Reassembly:** Client reassemble the TCP segments to form the complete HTTP response.

Parsing and Rendering

- **HTML Parsing:** The browser parses the HTML content to construct the Document Object Model (DOM).
- **CSS Parsing:** The browser fetches and parses CSS files to construct the CSS Object Model (CSSOM).
- **JavaScript Execution:** The browser fetches and executes JavaScript files.
- **Render Tree:** The browser constructs the render tree by combining the DOM and CSSOM.
- **Layout:** The browser calculates the layout, determining the position and size of elements on the page.
- **Painting:** The browser paints the pixels on the screen based on the layout.

10. Additional Requests

Resource Loading

- **Embedded Resources:** The HTML may reference additional resources (e.g., images, CSS, JS).
    - **Concurrent Requests:** The browser makes concurrent HTTP requests to fetch these resources.
- **Resource Parsing:** The browser parses and processes these resources as they are received.

Caching

- **Browser Cache:** The browser caches resources (e.g., images, CSS, JS) for future use to reduce load times and bandwidth usage.
- **Cache-Control Headers:** The server may provide cache-control headers to instruct the browser on how to cache resources.

Summary

When we type "facebook.com" and press Enter, an intricate series of events unfold involving hardware interactions, software processes, network communications, and security protocols. From the initial key press to the final rendering of the Facebook homepage, this process highlights the complexity and efficiency of modern web browsing, involving multiple systems and layers working seamlessly together.

References

1) "The TCP Three-Way Handshake," TCP/IP Guide.
   Link:
   http://www.tcpipguide.com/free/t_TCPConnectionEstablishmentProcessTheThreeWayHandsh-3.htm

2) "HTTP Strict Transport Security (HSTS)," Cloudflare.
   Link:
   https://developers.cloudflare.com/ssl/edge-certificates/additional-options/http-strict-transport-security/

3) "HTTP Requests and Responses," MDN Web Docs.
   Link: HTTP request methods - HTTP | MDN (mozilla.org)

4) "TLS Handshake Overview," DigiCert.
   Link: What is a TLS/SSL Handshake? | DigiCert FAQ