TCP & UDP Protocols, HTTP, HTTPs & ICMP Protocols and Working of it

1.TCP Protocol :

What is TCP (Transmission Control Protocol)?

Transmission Control Protocol (TCP) is a connection-oriented protocol for communications that helps in the exchange of messages between different devices over a network. It is one of the main protocols of the TCP/IP suite. In OSI model, it operates at the transport layer(Layer 4). It lies between the Application and Network Layers which are used in providing reliable delivery services. The Internet Protocol (IP), which establishes the technique for sending data packets between computers, works with TCP.

- TCP establishes a reliable connection between sender and receiver using the three-way handshake (SYN, SYN-ACK, ACK) and it uses a four-step handshake (FIN, ACK, FIN, ACK) to close connections properly.
- It ensures error-free, in-order delivery of data packets.
- It uses acknowledgments (ACKs) to confirm receipt.
- It prevents data overflow by adjusting the data transmission rate according to the receiver's buffer size.
- It prevents network congestion using algorithms like Slow Start, Congestion Avoidance, Fast Retransmit, and Fast Recovery.
- TCP header uses checksum to detect corrupted data and requests retransmission if needed.
- It is used in applications requiring reliable and ordered data transfer, such as web browsing, email, and remote login.

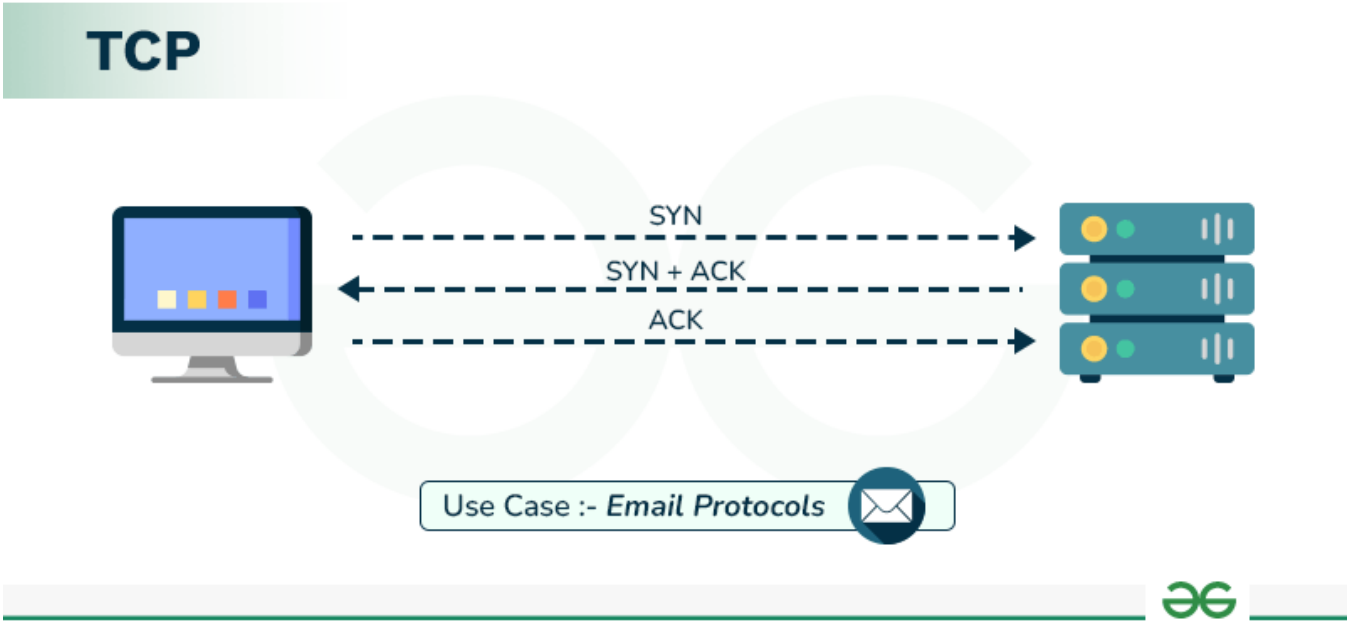
Internet Protocol (IP)

Internet Protocol (IP) is a method that is useful for sending data from one device to another from all over the internet. It is a set of rules governing how data is sent and received over the internet. It is responsible for addressing and routing packets of data so they can travel from the sender to the correct destination across multiple networks. Every device contains a unique IP Address that helps it communicate and exchange data across other devices present on the internet.

Working of Transmission Control Protocol (TCP)

Transmission Control Protocol (TCP) model breaks down the data into small bundles and afterward reassembles the bundles into the original message on the opposite end to make sure that each message reaches its target location intact. Sending the information in little bundles of information makes it simpler to maintain efficiency as opposed to sending everything in one go.

After a particular message is broken down into bundles, these bundles may travel along multiple routes if one route is jammed but the destination remains the same.



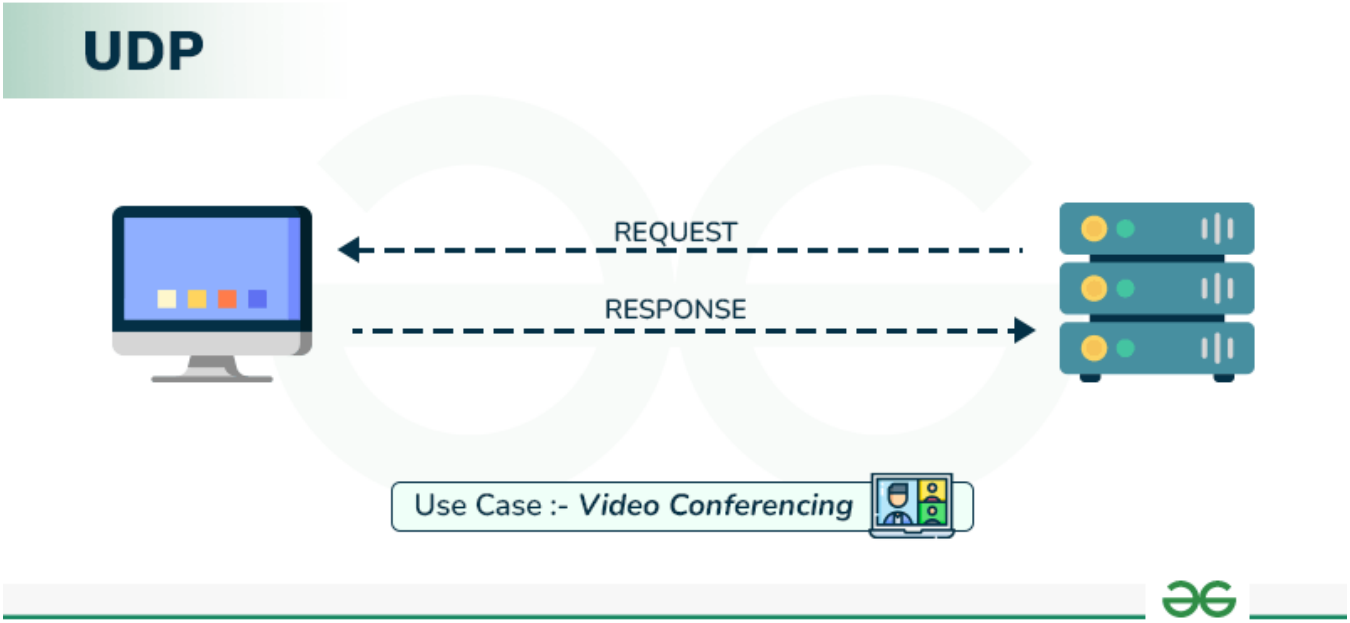
For Example: When a user requests a web page on the internet, somewhere in the world, the server processes that request and sends back an HTML Page to that user. The server makes use of a protocol called the HTTP Protocol. The HTTP then requests the TCP layer to set the required connection and send the HTML file.
Now, the TCP breaks the data into small packets and forwards it toward the Internet Protocol (IP) layer. The packets are then sent to the destination through different routes.
The TCP layer in the user's system waits for the transmission to get finished and acknowledges once all packets have been received.

2.UDP Protocol :

User Datagram Protocol (UDP) is a Transport Layer protocol. UDP is a part of the Internet Protocol suite, referred to as UDP/IP suite. Unlike TCP, it is an **unreliable and connectionless protocol.** So, there is no need to establish a connection before data transfer. The UDP helps to establish low-latency and loss-tolerating connections over the network. The UDP enables process-to-process communication.

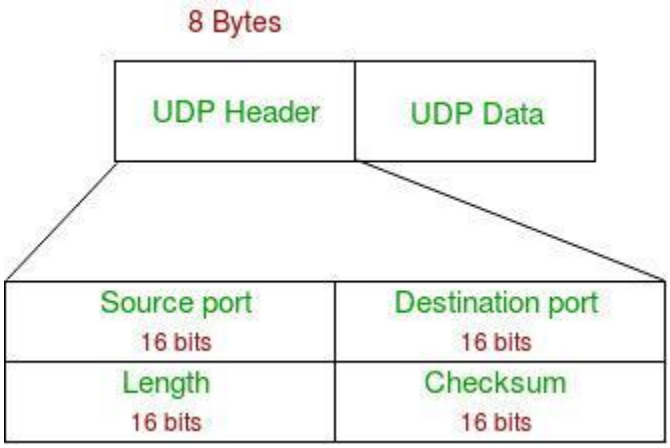
What is User Datagram Protocol?

User Datagram Protocol (UDP) is one of the core protocols of the Internet Protocol (IP) suite. It is a communication protocol used across the internet for time-sensitive transmissions such as video playback or DNS lookups . Unlike Transmission Control Protocol (TCP), UDP is connectionless and does not guarantee delivery, order, or error checking, making it a lightweight and efficient option for certain types of data transmission.



UDP Header

UDP header is an **8-byte** fixed and simple header, while for TCP it may vary from 20 bytes to 60 bytes. The first 8 Bytes contain all necessary header information and the remaining part consists of data. UDP port number fields are each 16 bits long, therefore the range for port numbers is defined from 0 to 65535; port number 0 is reserved. Port numbers help to distinguish different user requests or processes.



UDP Header

- **Source Port:** Source Port is a 2 Byte long field used to identify the port number of the source.
- **Destination Port:** It is a 2 Byte long field, used to identify the port of the destined packet.
- **Length:** Length is the length of UDP including the header and the data. It is a 16-bits field.
- **Checksum:** Checksum is 2 Bytes long field. It is the 16-bit one's complement of the one's complement sum of the UDP header, the pseudo-header of information from the IP header, and the data, padded with zero octets at the end (if necessary) to make a multiple of two octets.

3.HTTP Protocol :

HTTP stands for **Hypertext Transfer Protocol**, and it's the system that allows communication between web browsers (like Google Chrome or Firefox) and websites. When you visit a website, your browser uses HTTP to send a request to the server hosting that site, and the server sends back the data needed to display the page.

Think of HTTP as a set of rules or a language used by your browser and the web server to talk to each other, ensuring that websites load properly when you type in their URLs.

The Full Form of HTTP

HTTP is short for **Hypertext Transfer Protocol**. The word "hypertext" refers to text that links to other text or documents. "Protocol" means a set of rules for

communication. So, HTTP is a protocol that enables transferring hypertext (like web pages with links) between web servers and browsers.

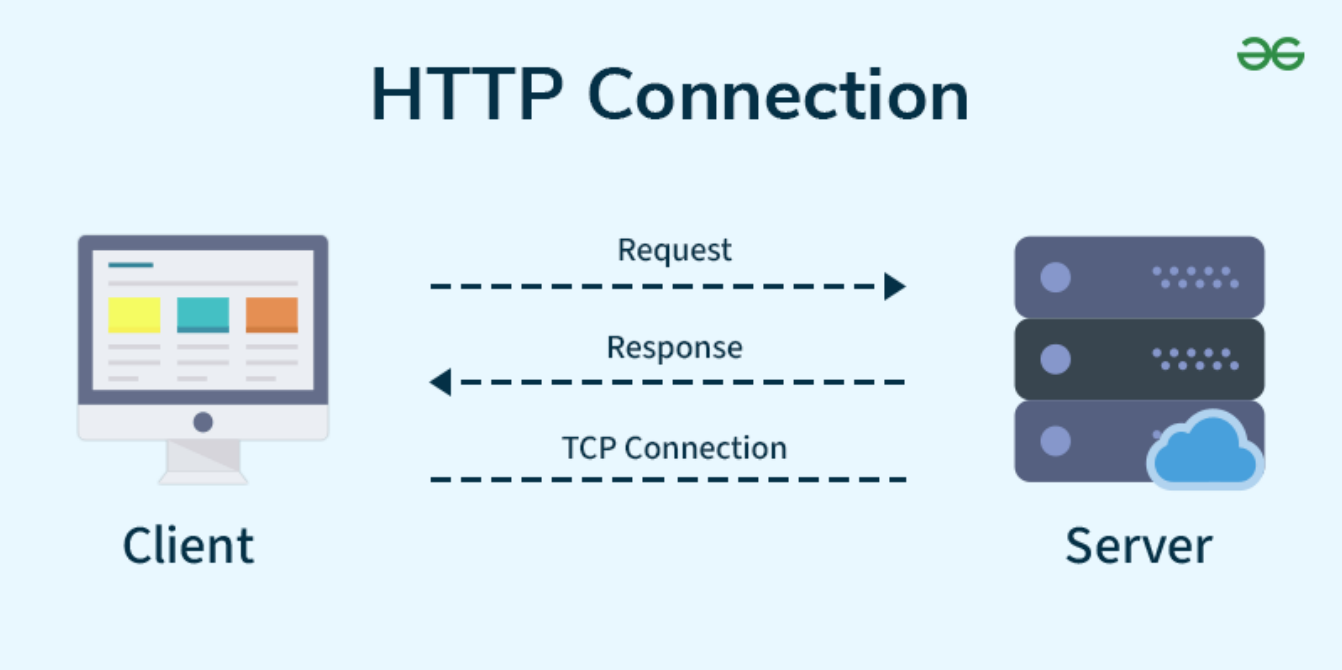
When you type a website address (like www.example.com) into your browser, HTTP is what makes sure the correct data is sent from the server to your browser.

How HTTP Works: Step-by-Step Process

Here's how HTTP works when you visit a website:

1. **Open Web Browser**: First, you open your web browser and type a website URL (e.g., www.example.com).
2. **DNS Lookup**: Your browser asks a Domain Name System (DNS) server to find out the IP address associated with that URL. Think of this as looking up the phone number of the website.
3. **Send HTTP Request**: Once the browser has the website's IP address, it sends an HTTP **request** to the server. The request asks the server for the resources needed to display the page (like text, images, and videos).
4. **Server Response**: The server processes your request and sends back an HTTP **response**. This response contains the requested resources (like HTML, CSS, JavaScript) needed to load the page.
5. **Rendering the Web Page**: Your browser receives the data from the server and displays the webpage on your screen.

After the page is loaded, the connection between the browser and server is closed. If you request a new page, a new connection will be made.



What is HyperText?

HyperText is a way of structuring text so that it can contain links (called "hyperlinks") to other documents or resources. When you click on a link in a webpage, you are typically directed to another page or resource on the internet. HTML (HyperText Markup Language) is used to create and format this type of text for web pages.

HTTP is the protocol used to transfer this hypertext between the web browser and the server, allowing you to click links and move around the web.

Understanding HTTP Request and Response

1. HTTP Request

An HTTP request is how your browser asks the server for something. It includes:

- **HTTP Version**: The version of HTTP (like HTTP/1.1 or HTTP/2) being used.
- **URL**: The specific address of the resource (e.g., https://www.example.com/about).

- **HTTP Method**: The type of action being requested (e.g., GET to retrieve information or POST to send data).
- **HTTP Request Headers**: Extra information about the request, like what kind of browser you're using or what kind of content you're expecting.
- **HTTP Request Body**: In some cases, the request will include a body that contains data (e.g., when you submit a form).

2. HTTP Response

- An HTTP response is the server's answer to your request. It includes:
- **HTTP Status Code**: A number that tells you if the request was successful or not (e.g., 200 OK means everything is fine, 404 Not Found means the requested page doesn't exist).
- **Response Headers**: Information about the response, like what kind of data is being sent (e.g., Content-Type: text/html means it's an HTML page).
- **Response Body**: The content that the server sends back (e.g., HTML code that the browser will use to display the webpage).

What is HTTP Status Code?

- **HTTP Status codes** are three-digit numbers that servers use to tell your browser what happened with the request you sent. There are different types of status codes:
- Informational **(1xx)**: These codes just give you information (e.g., 100 Continue means the request is still being processed).
- Successful**(2xx)**: These codes tell you everything went fine (e.g., 200 OK means the request was successful).

- Redirection**(3xx)**: These codes tell the browser to take additional action (e.g., 301 Moved Permanently means the requested page has moved to a new address).
- Client Error **(4xx)**: These codes indicate that there was a problem with your request (e.g., 404 Not Found means the page doesn't exist).
- Server Error **(5xx)**: These codes tell you that something went wrong on the server side (e.g., 500 Internal Server Error means the server had an issue processing the request).

History of HTTP

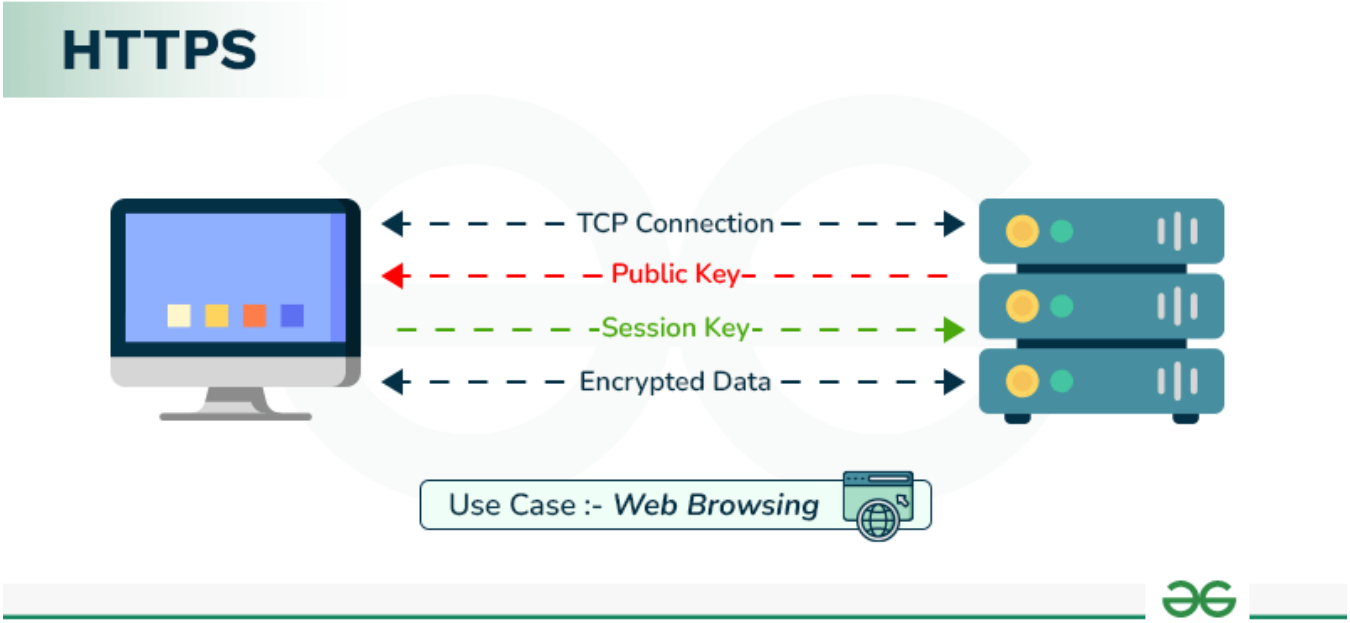
Tim Berners-Lee and his team at CERN get credit for inventing the original HTTP and associated technologies.

- **HTTP version 0.9:** This was the first version of HTTP, which was introduced in 1991.
- **HTTP version 1.0:** In 1996, RFC 1945 (Request For Comments) was introduced in HTTP version 1.0.
- **HTTP version 1.1:** In January 1997, RFC 2068 was introduced in HTTP version 1.1. Improvements and updates to the HTTP version 1.1 standard were released under RFC 2616 in June 1999.
- **HTTP version 2.0:** The HTTP version 2.0 specification was published as RFC 7540 on May 14, 2015.
- **HTTP version 3.0:** HTTP version 3.0 is based on the previous RFC draft. It is renamed as Hyper-Text Transfer Protocol QUIC which is a transport layer network protocol developed by Google.

4.HTTPS Protocol :

HTTPS stands for HyperText Transfer Protocol Secure. It is the most common protocol for sending data between a web browser and a website. HTTPS is the secure variant of HTTP and is used to communicate between the user's browser and the website, ensuring that data transfer is encrypted for added security.

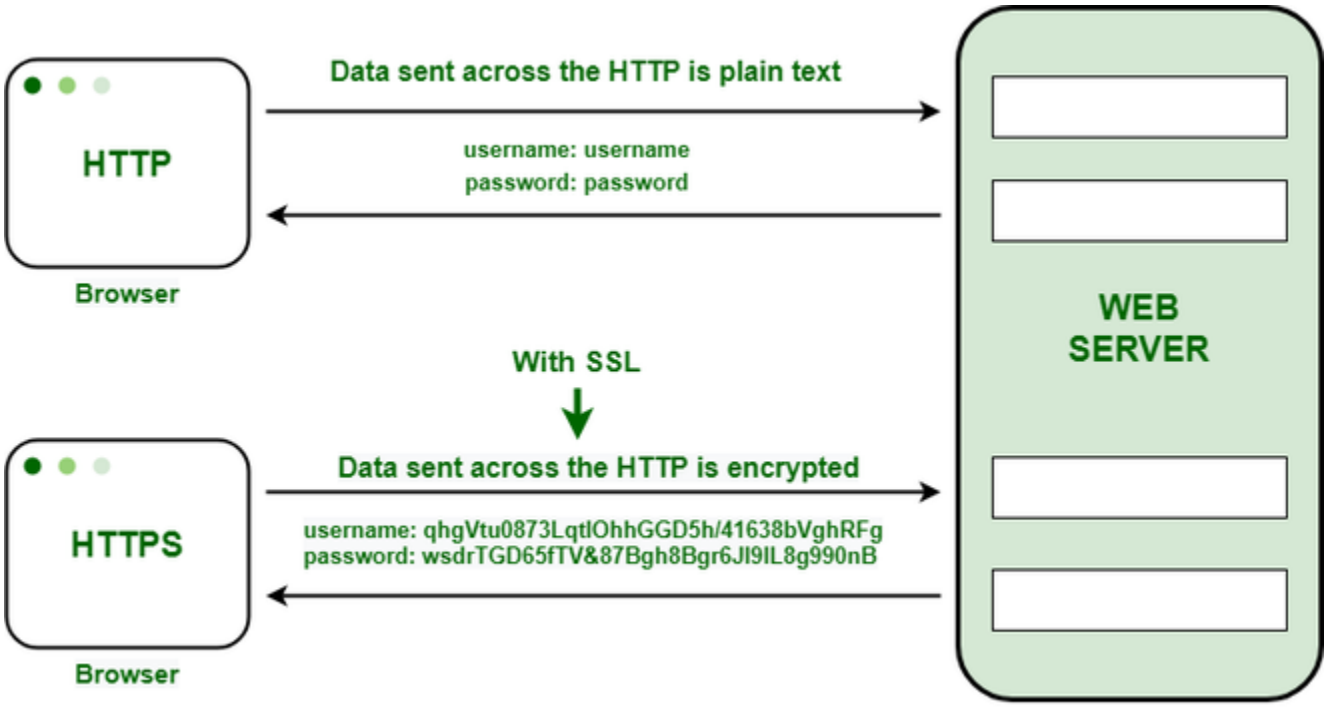
Any website, especially those requiring login details, should use HTTPS. You can see a padlock icon in the URL bar, which means the page is secure. Browsers, like Google Chrome, treat HTTPS seriously and mark non-HTTPS websites as "Not Secure."



How Does HTTPS Work?

HTTPS establishes the communication between the browser and the web server. It uses the **Secure Socket Layer** (SSL) and **Transport Layer Security** (TLS) protocol for establishing communication. The new version of SSL is **TLS(Transport Layer Security)**.

HTTPS uses the conventional HTTP protocol and adds a layer of SSL/TLS over it. The workflow of HTTP and HTTPS remains the same, the browsers and servers still communicate with each other using the HTTP protocol. However, this is done over a secure SSL connection. The SSL connection is responsible for the encryption and decryption of the data that is being exchanged to ensure data safety.



Why HTTPS Matters and What Happens Without It?

HTTPS is important because it keeps the information on websites safe from being easily viewed or stolen by anyone who might be spying on the network. When a website uses regular HTTP, data is sent in small chunks called packets that can easily be intercepted using free software. This makes communication, especially over public Wi-Fi, very vulnerable to attacks.

On the other hand, HTTPS encrypts the data, so even if someone manages to intercept the packets, they will appear as random, unreadable characters. For example:

- **Before encryption:**

"This is a string of text that is completely readable"

- **After encryption:**

"ITM0IRyiEhVpa6VnKyExMiEgNveroyWBPlgGyfkflYjDaaFf/Kn3bo3OfghBPD
Wo6AfSHlNtL8N7ITEwIXc1gU5X73xMsJormzzXlwOyrCs+9XCPk63Y+z0="

Secure Socket Layer (SSL)

The main responsibility of SSL is to ensure that the data transfer between the communicating systems is **secure and reliable**. It is the standard security technology that is used for encryption and decryption of data during the transmission of requests.

As discussed earlier, HTTPS is basically the same old HTTP but with SSL. For establishing a secure communication link between the communicating devices, SSL uses a digital certificate called **SSL certificate**.

There are two major roles of the SSL layer

- Ensuring that the browser communicates with the required server directly.
- Ensuring that only the communicating systems have access to the messages they exchange.

Encryption in HTTPS

HTTP transfers data in a hypertext format between the browser and the web server, whereas HTTPS transfers data in an encrypted format. As a result, HTTPS protects websites from having their information broadcast in a way that anyone eavesdropping on the network can easily see. During the transit between the browser and the web server, HTTPS protects the data from being accessed and altered by hackers. Even if the transmission is intercepted, hackers will be unable to use it because the me ssage is encrypted.

It uses an asymmetric public key infrastructure for securing a communication link. There are two different kinds of keys used for encryption -

- **Private Key**: It is used for the decryption of the data that has been encrypted by the public key. It resides on the server-side and is controlled by the owner of the website. It is private in nature.
- **Public Key**: It is public in nature and is accessible to all the users who communicate with the server. The private key is used for the decryption of the data that has been encrypted by the public key.

HTTP vs HTTPS

Below are the basic differences between HTTP and HTTPS.

HTTP	HTTPS
HTTP stands for HyperText Transfer Protocol.	HTTPS stands for HyperText Transfer Protocol Secure.
URL begins with "http://".	URL starts with "https://".
HTTP Works at the Application Layer.	HTTPS works at Transport Layer.
HTTP speed is faster than HTTPS.	HTTPS speed is slower than HTTP.

5.ICP Protocol :

Internet Control Message Protocol is known as ICMP. The protocol is at the network layer. It is mostly utilized on network equipment like routers and is utilized for error handling at the network layer. Since there are various kinds of network layer faults, ICMP can be utilized to report and troubleshoot these errors.

Since IP does not have an inbuilt mechanism for sending error and control messages. It depends on Internet Control Message Protocol(ICMP) to provide error control. In this article, we are going to discuss ICMP in detail along with their uses, messages, etc.

What is ICMP?

ICMP is used for reporting errors and management queries. It is a supporting protocol and is used by network devices like routers for sending error messages and operations information. For example, the requested service is not available or a host or router could not be reached.

Since the IP protocol lacks an error-reporting or error-correcting mechanism, information is communicated via a message. For instance, when a message is sent to its intended recipient, it may be intercepted along the route from the sender. The sender may believe that the communication has reached its destination if no one reports the problem. If a middleman reports the mistake, ICMP helps in notifying the sender about the issue. For example, if a message can't reach its destination, if there's network congestion, or if packets are lost, ICMP sends back feedback about these issues. This feedback is essential for diagnosing and fixing network problems, making sure that communication can be adjusted or rerouted to keep everything running smoothly.

Uses of ICMP

ICMP is used for error reporting if two devices connect over the internet and some error occurs, So, the router sends an ICMP error message to the source informing about the error. For Example, whenever a device sends any message which is large enough for the receiver, in that case, the receiver will drop the message and reply to the ICMP message to the source.

How Does ICMP Work?

ICMP is the primary and important protocol of the IP suite, but ICMP isn't associated with any transport layer protocol (TCP or UDP) as it doesn't need to establish a connection with the destination device before sending any message as it is a connectionless protocol.

The working of ICMP is just contrasting with TCP, as TCP is a connection-oriented protocol whereas ICMP is a connectionless protocol. Whenever a connection is established before the message sending, both devices must be ready through a TCP Handshake.

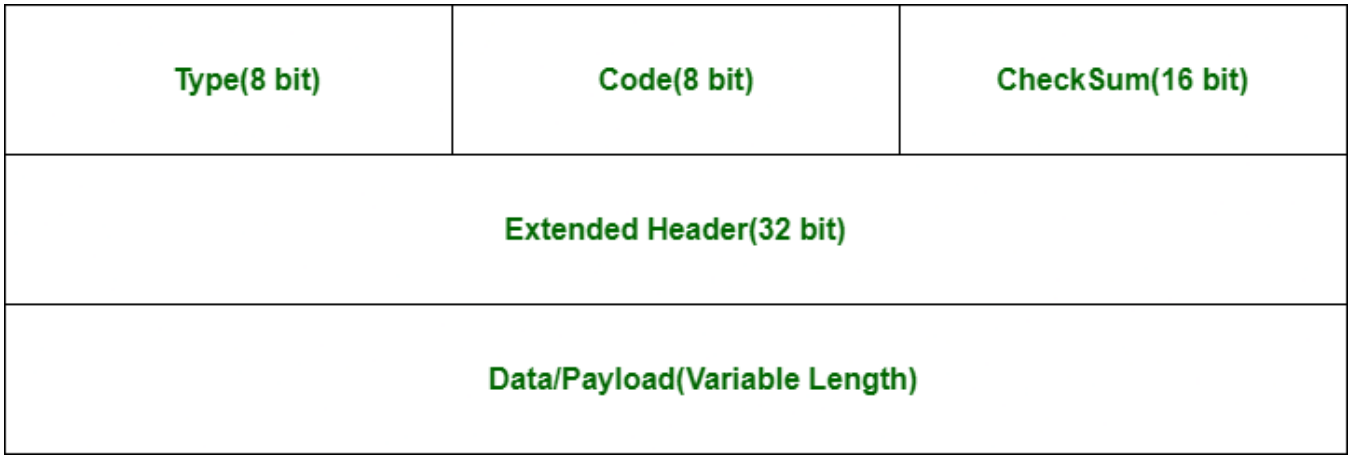
ICMP packets are transmitted in the form of datagrams that contain an IP header with ICMP data. ICMP datagram is similar to a packet, which is an independent data entity.

ICMP Packet Format

ICMP header comes after IPv4 and IPv6 packet header.

In the ICMP packet format, the first 32 bits of the packet contain three fields:

Type (8-bit): The initial 8-bit of the packet is for message type, it provides a brief description of the message so that receiving network would know what kind of message it is receiving and how to respond to it.



Some common message types are as follows:

- Type 0 - Echo reply
- Type 3 - Destination unreachable
- Type 5 - Redirect Message
- Type 8 - Echo Request
- Type 11 - Time Exceeded
- Type 12 - Parameter problem
- **Code (8-bit):** Code is the next 8 bits of the ICMP packet format, this field carries some additional information about the error message and type.
- **Checksum (16-bit):** Last 16 bits are for the checksum field in the ICMP packet header. The checksum is used to check the number of bits of the complete message and enable the ICMP tool to ensure that complete data is delivered.
- The next 32 bits of the ICMP Header are Extended Header which has the work of pointing out the problem in IP Message. Byte locations are identified by the pointer which causes the problem message and receiving device looks here for pointing to the problem.
- The last part of the ICMP packet is Data or Payload of variable length. The bytes included in IPv4 are 576 bytes and in IPv6, 1280 bytes.