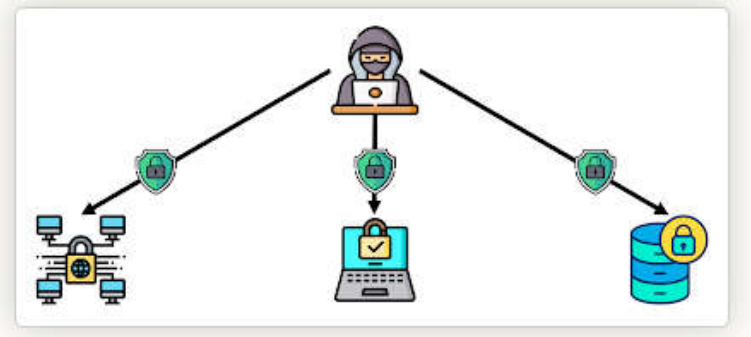
What is Web Security?

Web security refers to networks, computer system and data are protected from unauthorized person or group.



Purpose of Web Security

The purpose of web security is to prevent security attack like Passive attack and Active Attack. Web security maintains the smooth operation of any business that uses computers and prevents hackers and malware from manipulating your systems, software, or network.

How can achieve Web Security?

Various tools & technologies are available to achieve web security:

Web & Network Firewall: Web Application firewall sets between your website server and the data Connection. The purpose is to read every bit of data that passes through it and to protect your site.

Keep your software & plugins up to date: If your website's s/w or applications are not up-to-date, your site is not secure. Updates are vital to the

health and security of your website. Take all software and plugins update request seriously. Also use https and SSL Certificate to secure your website.

Backup your data: Back up your site regularly. You should maintain backups of all your website files in case your site becomes inaccessible or your data is lost.

Keep your website clean: Every database, application or plugins on your website is another possible point of entry for hackers. You should delete any files, databases or applications from your website that are no longer in use.

Strong password policy: It is important to use strong passwords to protect against brute force, password should be complex, containing uppercase and lowercase letters, numbers and special characters. Your password should be at least 10 characters long.

Password cracking tools: Password cracking tools help restore lost password, whether you have forgotten a password of your password has been hacked, a password Cracking tools can help you recover it.

Scan your website for vulnerabilities: It is important to regularly perform web security scans to check for website and server vulnerabilities. web security scans should be performed on a schedule and after any change or addition to your web Components.

Use of Antivirus: Antivirus software helps protect your computer against malwares and other incoming threats. Antivirus software looks at data - like webpage, files, software applications – which are travelling over the network to your device. It searches for known threats and monitors the behaviour of all programs and flagging suspicious behaviour.

What are Web Security Threats?

Web security threats are vulnerabilities within website and applications or attacks launched by malicious users. Web security threats are designed to breach security of website or applications. Web security threats involve malicious people and organizations, as well as the tools they use to leverage the internet in an attempt to infiltrate your network or devices. The most common security threats are malware, phishing, denial of services, SQL injection, and stolen data.

Modification of Message: Message should not be altered during transmission it is also called as data breach. It means some confidential and sensitive information gets exposed. It is one kind of threat.

Denial of Services: It is known as DDOS (Distributed Denial of Services). It is a web security threat that involves attackers flooding servers with large volumes of internet traffic to disrupt service and take websites offline. The sheer volume of fake traffic results in the target network or server being overwhelmed, which leaves them inaccessible.

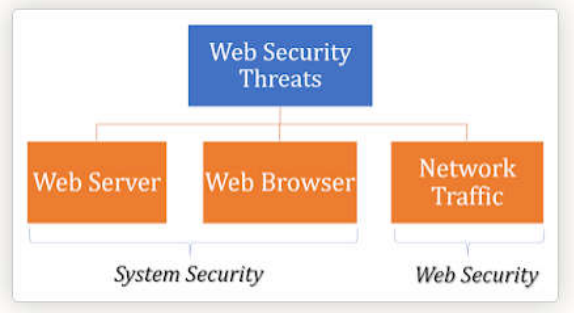
Phishing: Phishing attack targeting users through email, text message or social media messaging sites. Attackers impersonate of real user or website, users can trust that link and click on given link and provide sensitive information like account number, credit/debit card data and login credentials. User Can lost their money, sensitive information etc.....

SQL Injection: SQL stands for structured query language. SQL is used to search and query database. SQL Injection is a website security threats. SQL injection is the placement of malicious code in SQL statement, via webpage input. Using SQL injection hacker can retrieve credential and some sensitive information.

Malware: Malware stands for "Malicious Software". It is a file or code, typically delivered over a network, that infects, explores, steals or conduct virtually any behaviour an attacker wants. Malware comes in so many variants, there are number of methods to infect computer systems.

Classification of Web Security Threats

Web security threats are classify based on security attack: Passive and Active attacks. Another way to classify Web security threats is in terms of the location of the threat:

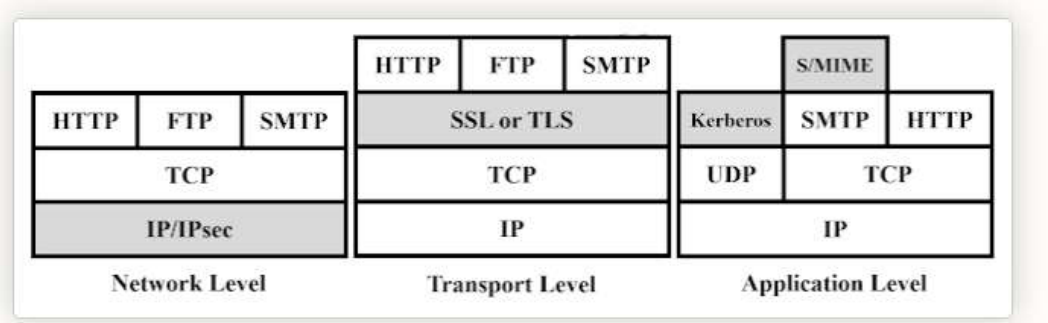


Web Security Threats

	Threats	Consequences	Countermeasures
Integrity	• Modification of user data • Modification of memory • Modification of message traffic in transit	• Loss of information • Compromise of machine • Vulnerability to all other threats	• Cryptographic checksums
Confidentiality	• Eavesdropping on the net • Theft of info from server • Theft of data from client • Info about network configuration • Info about which client talks to server	• Loss of information • Loss of privacy	• Encryption • Web proxies
Denial of Service	• Killing of user threads • Flooding machine with bogus requests • Filling up disk or memory • Isolating machine by DNS attacks	• Disruptive • Annoying • Prevent user from getting work done	• Difficult to prevent
Authentication	• Impersonation of legitimate users • Data forgery	• Misrepresentation of user • Belief that false information is valid	• Cryptographic techniques

Web Security Approaches

A number of approaches to providing Web security are possible. The various approaches that have been considered are similar in the services and the mechanisms that they are used. But it may be differed with respect to their scope of applicability and their relative location within the TCP/IP protocol stack.



Network Level: One way to provide Web security is to use IP security (IPsec). The advantage of using IPsec is that it is transparent to end users and applications and provides a general-purpose solution. Furthermore, IPsec includes a filtering capability so that only selected traffic need incur the overhead of IPsec processing.

Transport Level: Another relatively general-purpose solution is to implement security just above TCP. The foremost example of this approach is the Secure Sockets Layer (SSL) and the follow-on Internet standard known as Transport Layer Security (TLS). At this level, there are two implementation choices. For full generality, SSL (or TLS) could be provided as part of the underlying protocol suite and therefore be transparent to applications. Alternatively, SSL can be embedded in specific packages. For example, Netscape and Microsoft Explorer browsers come equipped with SSL, and most Web servers have implemented the protocol.

Application Level: Application-specific security services are embedded within the particular application. The advantage of this approach is that the service can be tailored to the specific needs of a given application.

SSL Architecture & Protocol

SSL is developed by Netscape Communication. SSL stands for "Secure Socket Layer".

SSL stands for Secure Socket Layer.

What is SSL?

SSL is a protocol for establishing secure links between networked computers.

Purpose of SSL

SSL provides confidentiality, authentication and data integrity in internet communication. SSL is the predecessor to the modern TLS encryption used today.

Figure: Connection using HTTP

In above figure user interact with server but server uses only normal http, it shows unsecured connection between server and client. Hacker or attacker can capture the message from unsecured connection. Username and password are also sent in plain text form. It means hacker or attacker get user's sensitive information.

Figure: Connection using HTTPS (use of SSL Certificate)

In above figure user interact with server and server user secure https, it shows secure connection between client and server. ***SSL certificate*** is added at server side. So, http use SSL, it converts into https. Connection is secured, it means all

the data is in encrypted form during transmission. Hacker or attacker cannot get any information from that connection.

What is an SSL Certificate?

An SSL certificate is a bit of code on your web server that provides security for online communications. When a web browser contacts your secured website, the SSL certificate enables an encrypted connection. It's kind of like sealing a letter in an envelope before sending it through the mail. Websites need SSL certificates to keep user data secure, verify ownership of the website, prevent attackers from creating a fake version of the site, and convey trust to users.

SSL Architecture

The current version of SSL is 3.0. SSL is works in between application layer and transport layer the reason SSL is also called TLS (Transport Layer Security).

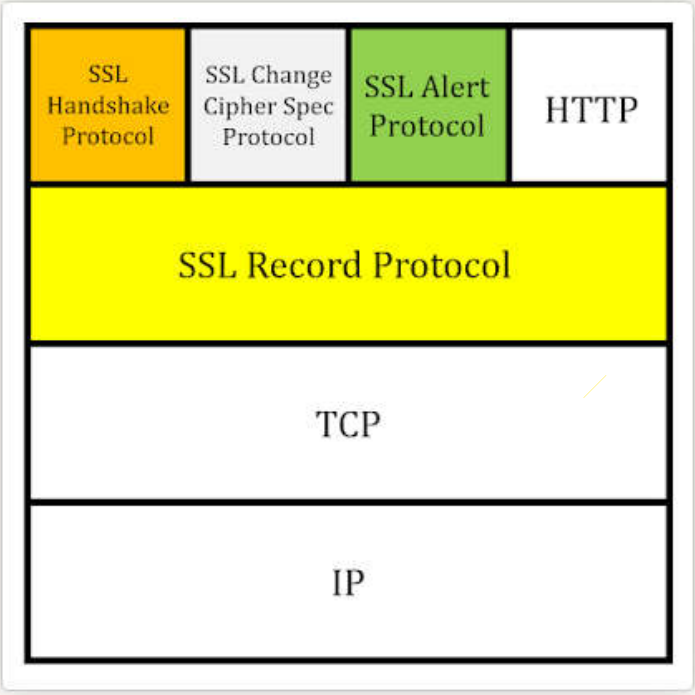


Figure: SSL Protocol Stack

SSL encrypt the data received from application layer of client machine and add its own header (SSL header) into the encrypted data and send encrypted data

to the server side. SSL is not a single protocol to perform security tasks there are two layers of sub-protocols which supports SSL there are the SSL handshake protocol, SSL change cipher specification, SSL alert protocol and the SSL record protocol shown in architecture.

SSL Handshake Protocol: Connection establishment.

SSL Change Spec Protocol: Use of required cipher techniques for data encryption.

SSL Alert Protocol: Alert (warning, error if any) generation.

SSL Record Protocol: Encrypted data transmission and encapsulation of the data sent by the higher layer protocols. Two important SSL concepts are the SSL Connection and the SSL Session.

SSL Connection: It is a transport that provides a suitable type of service. Each connection is associated with one SSL session.

SSL Session: It is a set of cryptographic security parameters which can be shared among multiple SSL connections. An SSL session is an association between a client and a server.

SSL Protocols

SSL Handshake Protocol

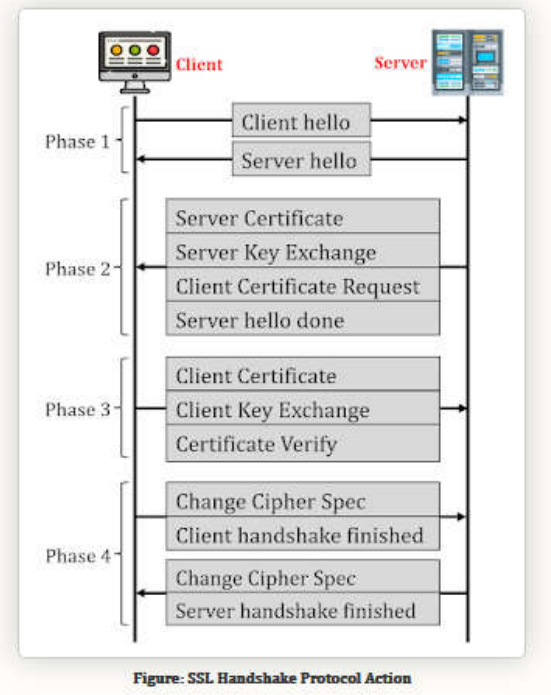


Figure: SSL Handshake Protocol Action

Phase 1: Establishing security capabilities

Client Hello:

The highest SSL version number which the client can support.

A session ID that defines the session.

There is a cipher suite parameter that contains the entire cryptographic algorithm which supports client's system.

A list of compression methods that can be supported by client system.

Server Hello:

The highest SSL version number which the server can support.

A session ID that defines the session.

A cipher suite contains the list of all cryptographic algorithms that is sent by the client which the server will select the algorithm.

A list of compression method sent by the client from which the server will select the method.

Phase 2: Server Authentication and Key Exchange

Certificate: The server sends a certificate message to authentication itself to the client. If the key exchange algorithm is Diffie-Hellman than no need of authentication.

Server key exchange: This is optional. It is used only if the server doesn't sends its digital certificate to client.

Certificate Request: The server can request for the digital certificate of client. The client's authentication is optional.

Server Hello done: The server message hello done is the last message in phase 2, this indicates to the client that the client can now verify all the certificates received by the server. After this hello message done, the server waits for the client-side response in phase 3.

Phase 3: Client Authentication and Key Exchange

Client Certificate: It is optional, it is only required if the server had requested for the client's digital certificate. If client doesn't have certificate, it can be sending no certificate message. Then it is up to server's decision whether to continue with the session or to abort the session.

Client key exchange: The client sends a client key exchange, the contents in this message are based on key exchange algorithms between both the parties.

Certificate Verify: It is necessary only if the server had asked for client authentication. The client has already sent its certificate to the server. But additionally if server wants then the client has to prove that it is authorized holder of the private key. The server can verify the message with its public key already sent to ensure that the certificate belongs to client.

Phase 4: Finish

Change cipher spec: It is a client-side messages telling about the current status of cipher protocols and parameters which has been made active from pending state.

Finished: This message announces the finish of the handshaking protocol from client side.

Change cipher spec: This message is sent by server to show that it has made all the pending state of cipher protocols and parameters to active state.

Finished: This message announces the finish of the handshaking protocol from server and finally handshaking is totally completed.

SSL Change Cipher Spec Protocol

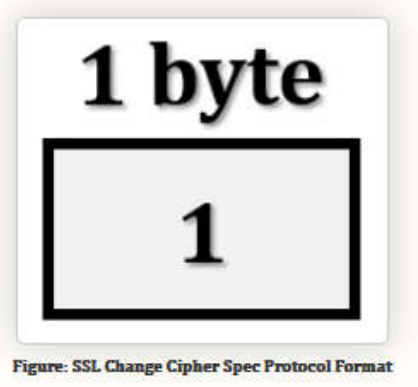


Figure: SSL Change Cipher Spec Protocol Format

SSL Change Cipher Spec Protocol is upper layer protocol. It is the simplest protocol. This protocol consists of only single byte with value "1", as shown in figure. It consists of single message only. It copies pending state to current state, which updates the cipher suite to be used to this connection.

SSL Alert Protocol

Figure Shows Alert Protocol Format

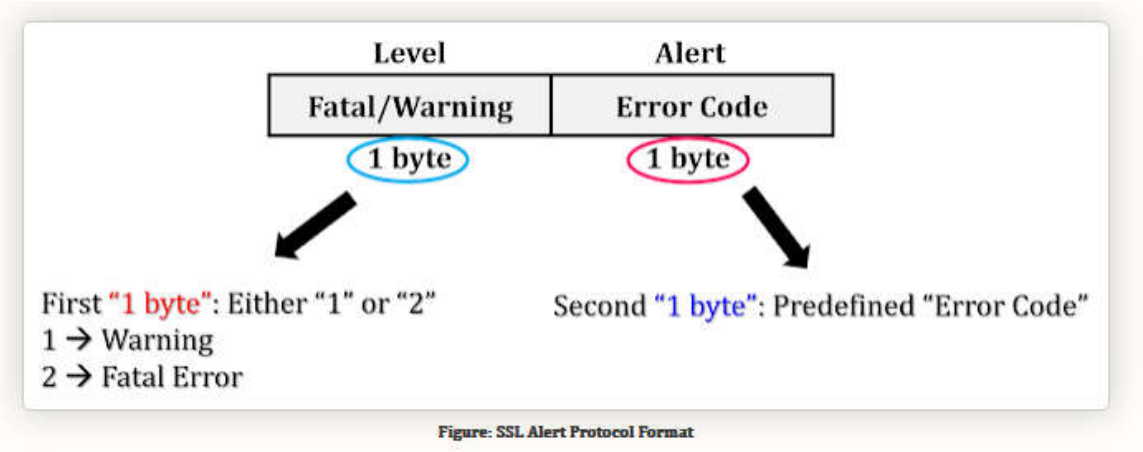


Figure: SSL Alert Protocol Format

SSL uses the Alert protocol for reporting error that is detected by client or server, the party which detects error sends an alert message to other party. If error is serious than both parties terminate the session.

Table shows the types of alert messages. SSL alert protocol is the last protocol of SSL used transmit alerts, if any via SSL record protocol to the client or server.

The SSL alert protocol format is shown in figure. Alert protocol uses two bytes to generate alert. First 1 byte indicates two values either 1 or 2. "1" value indicate warning and "2" indicate a fatal error.

Whereas second 1 byte indicates predefined error code either the server or client detects any error it sends an alert containing the error.

Alert Code	Alert Message	Description
0	close_notify	No more message from sender.
10	unexpected_message	An incorrect message received.
20	bad_record_mac	A wrong MAC received.
30	decompression_failure	Unable to decompress.
40	handshake_failure	Unable to finalize handshake by the sender.
42	bad_certificate	Received a corrupted certificate.
42	No_certificate	Client has no certificate to send to server.
42	certificate_expired	Certificate has expired.

SSL Record Protocol

After completion of successful SSL handshaking the keen role of SSL record protocol starts now. SSL record protocol is second sub-protocol of SSL also called lower-level protocol. As defined earlier the SSL record protocol is responsible for encrypted data transmission and encapsulation of the data sent by the higher layer protocols also to provide basic security services to higher layer protocols. SSL records protocol provides different service like data authentication; data confidentiality through encryption algorithm and data integrity through message authentication to SSL enabled connections.

The record protocol provides two services in SSL connection:

Confidentiality: This can be achieved by using secret key, which is already defined by handshake protocol.

Integrity: The handshake protocol defines a shared secret key that is used to assure the message integrity.

Following are the operation performed in Record protocol after connection is established and authentication is done of both client and server.

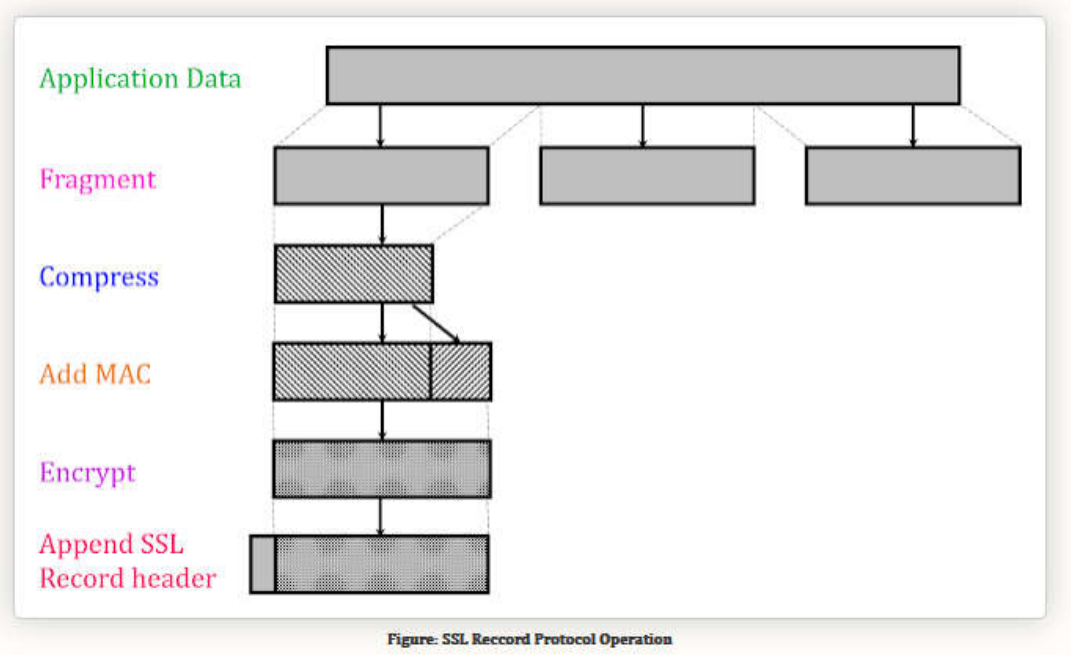


Figure: SSL Reccord Protocol Operation

Fragmentation: The original message that is to be sent it broken into blocks. The size of each block is less than or equal to 214 bytes.

Compression: The fragmented blocks are compressed which is optional. It should be noted that the compression process must not result into loss of original data.

Addition of MAC: A short piece of information used to authenticate a message for integrity and assurance of message.

Encryption: The overall steps including message is encrypted using symmetric key but the encryption should not increase the overall block size.

Append Header: After all the above operation, header is added in the encrypted block which contains following fields.

Content type: It specifies which protocol is used for processing.

Major version: It specifies the major version of SSL used, for example if SSL version 3.1 is in use than this field contains 3.

Minor version: It specifies minor version of SSL used, for example version 3.0 is in use than field contains 0.

Compressed length: It specifies the length in bytes of the original plain text block.

Figure shows SSL record protocol header format.

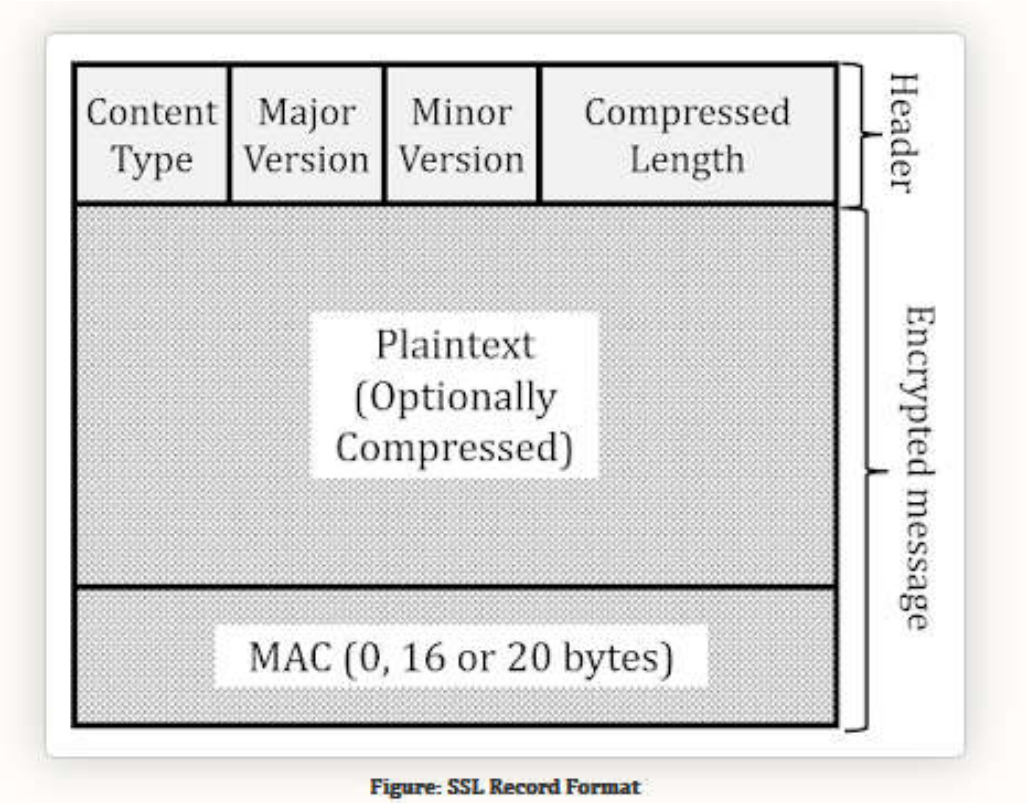


Figure: SSL Record Format

HTTPS

HTTPS stands for “Hyper Text Transfer Protocol Secure”.

What is HTTPS?

HTTPS is a protocol, which is used for communication between web browser and web server. HTTPS is secure version of HTTP.

Purpose of HTTPS

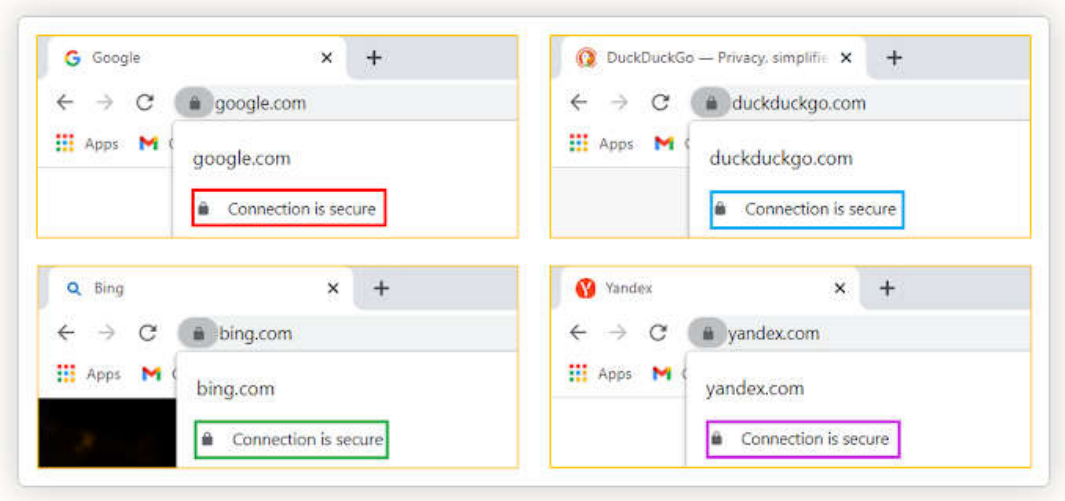
HTTPS provides the confidentiality and integrity of data between the user's computer and the website. HTTPS encrypt URL, username, password and sensitive information of user.

What is the default port number of HTTPS?

443 is default port number of HTTPS.

Is search engine uses HTTP or HTTPS?

Search engine uses HTTPS.



Working of HTTPS

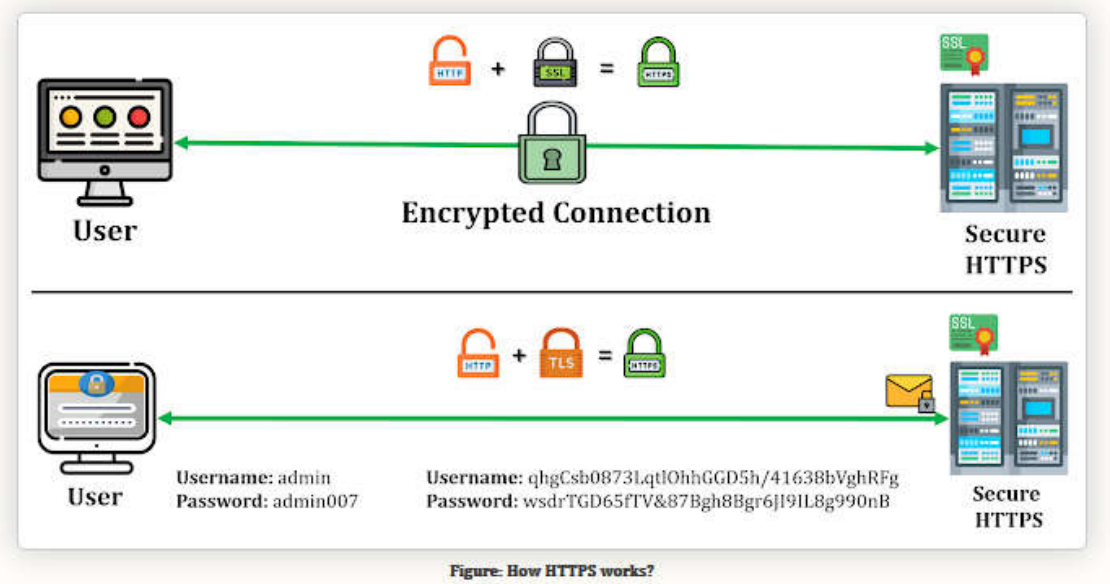


Figure: How HTTPS works?

In above figure user interact with server and server user secure https, it shows secure connection between client and server. ***SSL certificate*** is added at server side. So, http use SSL, it converts into https. Connection is secured, it means all the data in encrypted from during transmission. Hacker or attacker cannot get any information from that connection.

When HTTPS is used, the following elements are encrypted during communication:

URL of the requested document

Contents of the document

Contents of browser forms (filled in by browser user)

Cookies sent from browser to server and from server to browser

Contents of HTTP header

HTTPS Connection

HTTPS connection execute in three phases: Connection Initiation, Data Transfer, Connection Closure.

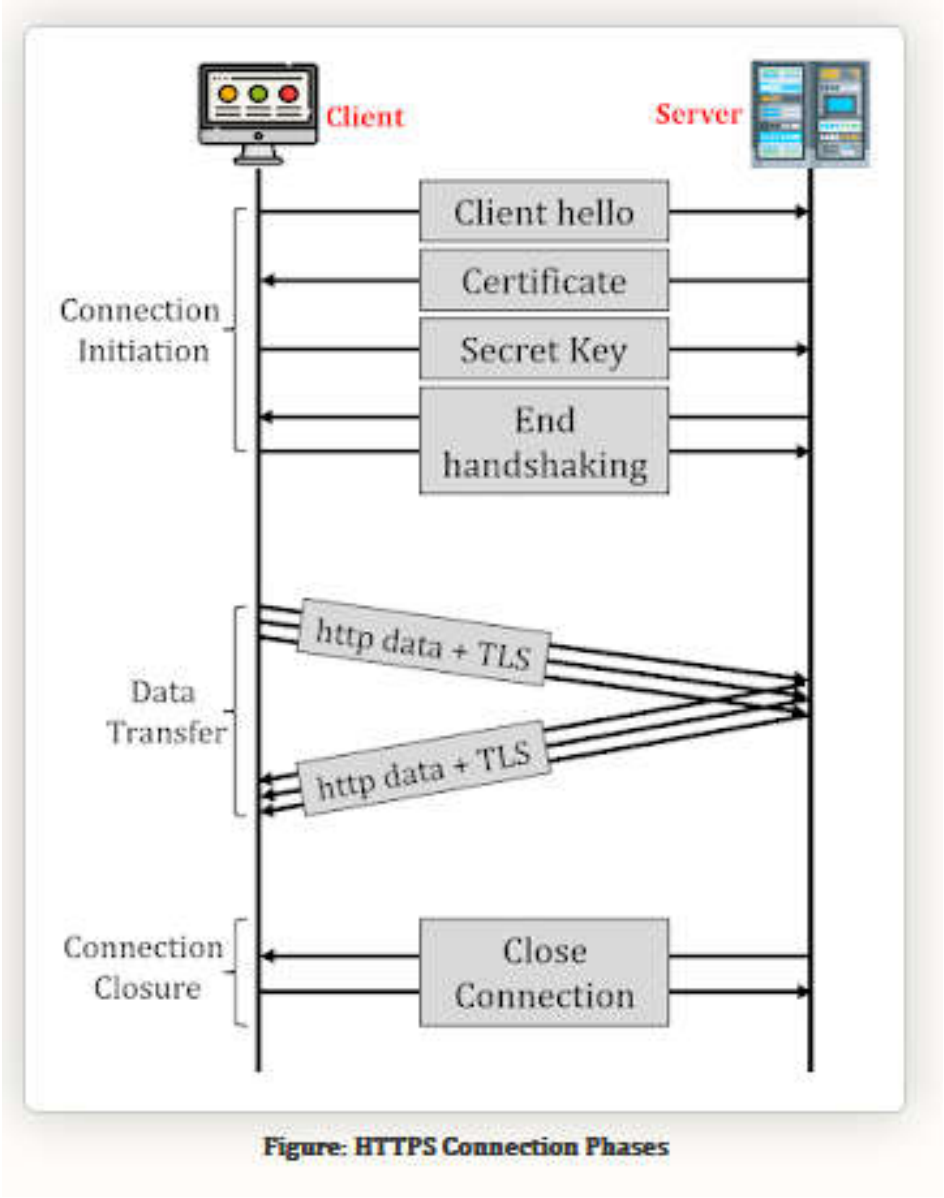


Figure: HTTPS Connection Phases

Connection Initiation

HTTPS uses TLS handshake protocol to establish a connection between client and server.

Client Hello to server: Client sends hello request to server to start connection initiation.

Digital Certificate shared by server: Server shares its digital certificate with client for the purpose to share a public key of server.

Secret Key share with server: Client generates secret key and share with server. This secret key is encrypted using server's public key. It is decrypt using only server's private key.

End handshaking: When connection is established TLS handshake end.

We need to be clear that there are three levels of awareness of a connection in HTTPS: At the HTTP level, At the level of TLS, At the level of TCP.

At the HTTP level, an HTTP client requests a connection to an HTTP server by sending a connection request to the next lowest layer. Typically, the next lowest layer is TCP, but it also may be TLS/SSL. At the level of TLS, a session is established between a TLS client and a TLS server. This session can support one or more connections at any time. As we have seen, a TLS request to establish a connection begins with the establishment of a TCP connection between the TCP entity on the client side and the TCP entity on the server side.

Data Transfer

Data Transfer should be done by HTTP Request with TLS application data. All HTTP data is to be sent as TLS application data. Normal HTTP behaviour, including retained connections, should be followed.

Connection Closure

Connection closure should be done by three levels: HTTP Level, TLS Level, TCP Level.

An HTTP client or server can indicate the closing of a connection by including the following line in an HTTP record: Connection: close. This indicates that the connection will be closed after this record is delivered. The closure of an HTTPS connection requires that TLS close the connection with the peer TLS entity on the remote side, which will involve closing the underlying TCP connection. At the TLS level, the proper way to close a connection is for each side to use the TLS alert protocol to send a close_notify alert. TLS implementations must initiate an exchange of closure alerts before closing a connection.

A TLS implementation may, after sending a closure alert, close the connection without waiting for the peer to send its closure alert, generating an "incomplete close". Note that an implementation that does this may choose to reuse the session. This should only be done when the application knows (typically through detecting HTTP message boundaries) that it has received all the message data that it cares about. HTTP clients also must be able to cope with a situation in which the underlying TCP connection is terminated without a prior close_notify alert and without a Connection: close indicator. Such a situation could be due to a programming error on the server or a communication error that causes the TCP connection to drop. However, the unannounced TCP closure could be evidence of some sort of attack. So the HTTPS client should issue some sort of security warning when this occurs.

Difference between HTTP and HTTPS (HTTP vs HTTPS)

Parameter	HTTP	HTTPS
Acronyms	Hyper Text Transfer Protocol	Secure Hyper Text Transfer Protocol
Port Number	Default port 80.	Default port 443
URL	URL is start with http://	URL is start with https://
Security	Unsecure	Enhanced Security.
Encryption	No	Yes
SSL Certificate	No	Yes
Speed	Faster	Slower
Protocol layer	Application Layer	Application & Transport Layer