**DETAILED REPORT ON HTTPS**

**DCF255: DATA COMMUNICATIONS FUNDAMENTALS (ZDD)**

**PROFESSOR AMIR MOFIDI**

**17th April 2022**

**Created by:**

**Yuvraj Singh**

**HTTPS PROTOCOL**

**Introduction**

In this report, we will look at the HTTPS protocol and its similarities and differences with the HTTP protocol. The report will begin with an overview of HTTP and HTTPS, followed by browser integration and usage. The technical concepts of HTTPS are discussed after the basic overview and high-level information. The technical concepts are further subdivided into sections such as the network layer, server configuration, vulnerabilities, and differences between the standard and secured versions of the protocol. Finally, detailed writing on TLS is discussed, as well as the various steps that occur in a TLS handshake.

**Overview**

The Hypertext Transfer Protocol Secure (HTTPS) is an application-level protocol. It is mainly used for distributed, collaborative, hypermedia information systems. It plays a major role in data communication as it is the foundation for the World Wide Web (i.e., the internet) since 1990. It can be used for other purposes well like request methods, error codes, and headers. It is a generic and stateless protocol.

HTTP comes under the communication protocol TCP/IP which helps to deliver data in various formats like HTML files, image files, query results, etc On the World Wide Web The default port is TCP 80, but other ports can be used as well. It sets a standard for the computer to communicate with one other. HTTP specification specifies how clients' request data will be constructed and sent to the server, and how the servers respond to these requests. The main features of HTTP would be, that it is connectionless, media-independent, and stateless.

**HTTPS Trusted resource**

It is used to protect insecure networks and networks that may be subject to tampering. The WI-FI access points have access to insecure networks which allows anyone on the same local network to [packet](https://en.wikipedia.org/wiki/Packet_analyzer) sniff and discover sensitive information not protected by HTTPS. Additionally, some free-to-use and paid [WLAN](https://en.wikipedia.org/wiki/Wireless_LAN) networks have been observed tampering with webpages by engaging in [packet injection](https://en.wikipedia.org/wiki/Packet_injection) in order to serve their own ads on other websites. This realization can be attacked This practice can be malicious in many ways, such as by injecting [malware](https://en.wikipedia.org/wiki/Malware) onto webpages and stealing users' private information. Malicious Tor Nodes have the highest possibility of damaging the data passing through them using malware injection. It can be prevented or secured using HTTPS over the Tor network. It is the major reason for the Electronic Frontier Foundation and the Tor project to the development of HTTPS everywhere in the Tor Browser. In Today’s world and with increasing awareness of global surveillance and stealing of personal data, the usage of HTTPS websites is increasing for security purposes.

**Browser integration of HTTPS**

The web browsers will display a warning if it finds something suspicious and receives an invalid certificate. In older versions of web browsers, it would ask the user whether they want to continue in form of a dialog box if it senses an invalid certificate. The new browsers display a warning from the entire window. It also displays the website’s information on security in the address bar. Extended validation certificates show the legal entity on the certificate information. Also, the newer browser shows various warnings if the user browses a website that contains a mixture of encrypted and unencrypted content. It also displays the site's security information in the address bar.

The [Electronic Frontier Foundation](https://en.wikipedia.org/wiki/Electronic_Frontier_Foundation), opining that "In an ideal world, every web request could default to “HTTPS", has enabled an add-on called HTTPS for [Mozilla Firefox](https://en.wikipedia.org/wiki/Mozilla_Firefox), [Google Chrome](https://en.wikipedia.org/wiki/Google_Chrome), [Chromium](https://en.wikipedia.org/wiki/Chromium_(web_browser)), and [Android](https://en.wikipedia.org/wiki/Android_(operating_system)), which helps to enable HTTPS by default for hundreds of frequently used websites

Forcing a web browser to load HTTPS content only has been supported in Firefox starting in version 83. Starting in version 94, Google Chrome can "always use secure connections" if toggled in the browser's settings.

**Usage of HTTPS**

The usage of HTTP is vast in today’s world. With today’s scope and its authenticity of security, it is used in every nook. Till, April 2018, 33.2% of Alexa top 1,000,000 websites use HTTPS as default, 57.1% of the Internet's 137,971 most popular websites have a secure implementation of HTTPS and 70% of page loads (measured by Firefox Telemetry) use HTTPS.

**Technicals of the HTTPS**

* Difference between HTTPS & HTTPS

Let’s have a look at the technicals of HTTPS. To start with, we must know the difference between HTTP & HTTPS. Firstly, the HTTP URL begins with “http://“ which we see in our browser’s top tab. Also, HTTP used port 80 by default. On the other hand, HTTPS has an additional “s” in the URL meaning a secure version of HTTP. Unlike HTTP the secure version used port for 443 by default.  
  
The HTTP normal version is not encrypted Making it vulnerable to attacks such as man in the middle. The attackers can easily gain access to the sensitive information of the websites and also can inject malware or advertisements easily. whereas HTTPS is designed to withstand such attacks and is far more secure than HTTP.

* The Network Layers

Talking about the layer on HTTPS works, it operates on the highest layer of the TCP/IP suite, which is the application layer. HTTPS is not a separate protocol, it is basically an ordinary HTTP over a secure connection of SSL/TLS. In HTTPS, the TLS security protocol, which is the sublayer of the application layer, encrypts the simple HTTP message before the transmission and also decrypts the data upon arrival *(Refer to Diagram 2)*. This protocol is responsible for the encryption of all of the content, Including the HTTP Request/Response data as well as headers.

* The Server setup

Moving forward, If someone wants to set up the webserver to accept an HTTPS connection, They must Create a public key certificate. The web browsers are provided with the list of Major certificate authorities to check and accept without any warning prompts. This certificate is generally signed by a trusted certificate authority, which makes sure that the holder of the certificate is the operator of the Web server.

* Vulnerabilities of HTTPS

Everything has its pros and cons, let’s discuss the cons of HTTPS. There are two Modes in which the configuration of the SSL and the TLS layer is performed. These are simple and Mutual modes. In the simple mode, authentication is only performed on one side which is the server-side, Whereas the mutual mode requires the user to Provide a personal certificate to the browser for authentication. But in either of these modes, the protection depends on how it is implemented, and it is not always 100% secure.  
  
A complicated attack, SSL stripping Which is a man-in-the-middle type of attack can outweigh the security of HTTPS by changing the URL from https: to http:, making the users think that it is a secure webpage which in actuality is not. This vulnerability can be avoided using HTTP strict transport security.

**How does it actually work?**

Now that we have covered all the concepts of HTTPS, let’s dive straight into how HTTPS actually works. To start with, as we know from the above paragraphs, HTTPS requires a secure certificate from a third-party vendor to secure a connection and verify that the site is legitimate or not. That secure certificate is defined as an SSL Certificate.  
SSL(secure sockets layer) is responsible to establish a secure ad encryption connection between a browser and the server. Now, this SSL certificate does not act in the same way for all the webpages, rather the level of protection or encryption depends on when the certificate was bought with what level of security. This SSL certificate is extremely important for that additional security over other software as it protects sensitive data from third-party attackers, especially when it comes to running an e-commerce website.

For instance, when we buy something from an online e-commerce store, we must initiate a payment to get it delivered to us, whichever payment method it be such as credit card, debit card, or internet banking. Also, the address, phone number, and other sensitive information are required in order to place an order. Now, this sensitive data is what someone might want to keep away from the attackers and this is where the HTTPS plays an important role, even though its cost is slightly higher than its counterpart, it is definitely worth it.

**TLS and its application in HTTPS**

We discussed what is TLS in the previous paragraphs, but how does it actually go hand in hand with HTTPS?  
As we know that it helps encrypt the data, it uses cryptographic techniques that ensure that the data which is being sent over has not been tampered, the data is being sent to the right person where it was supposed to be sent and prevents the data from being visible.

This is where the TLS handshake comes into account. TLS handshake is basically a process that gets carried out when a communication session is established which uses a TLS encryption. During this handshake, the two communicating devices exchange messages to verify each other, admit the session keys, acknowledge, and establish the algorithms that they will use till the end of that connection. Thus, we can say that the TLS handshake plays a fundamental role in the working of HTTPS.

* TLS vs SSL handshake, what’s the difference

Now, why TLS handshakes when we already have SSL handshakes. This is because the SSL, or Secure Sockets Layer was an encryption protocol that was originally developed for HTTP. Now, TLS has replaced the SSL and the SSL handshake can be referred to as the TLS handshake. This TLS handshake takes place when a user navigates to a website that uses HTTPS and the browser begins to query the website’s origin server. But, before this handshake, a TCP connection and handshake must be established.

**Steps of the TLS handshake**

A TLS handshake involves multiple steps, as the client and server communicate the information that is necessary for a handshake. Although, the exact steps within a TLS handshake depend upon the kind of algorithm used and also the session keys exchanged, the RSA key exchange algorithm is used most of the time. It goes as follows:

1) The client’s hello message – The client initiates the handshake by sending a hello message to the server

2) The server’s hello message – In response to the client’s hello, the server sends a message which contains the server’s SSL certificate, the cipher suite, and the server random.

3) Authentication – The client verifies the SSL certificate that the server provided with the authority that issued it.

4) The premaster secret – The client sends a random string of bytes one more time called the “premaster secret” which is encrypted with a public key and can only be decrypted by the server using a private key.

5) Private key used – The server decrypts the premaster secret.

6) Session key created – Both the server and the client generate session keys using the above information.

7) Client is ready – The client sends a session key encrypted message that states “finished”.

8) Server is ready – The server sends a session key encrypted message that states “finished”.

9) Secure symmetric encryption achieved – The handshake is now complete, and the communication can be continued using the session keys.

*Refer to diagram 1 for the TLS handshake.*

**Conclusion**

In a nutshell, the Hypertext Transfer Protocol Secure (HTTPS) is a protocol used at the application level. HTTP is widely used in today's world. Unlike HTTP, the secure version uses port 443 by default, whereas HTTP normal is not encrypted, making it vulnerable to man-in-the-middle attacks. Attackers can easily gain access to websites' sensitive information and inject malware or advertisements. The TLS security protocol, which is a sublayer of the application layer in HTTPS, encrypts the simple HTTP message prior to transmission and decrypts the data upon arrival. This protocol is in charge of encrypting all content, including HTTP Request/Response data and headers. The SSL certificate is critical for that extra layer of security over other software because it protects sensitive data from third-party attackers, which is especially important when running an e-commerce website. TLS handshake is essentially a process that occurs when a communication session using TLS encryption is established. TLS handshake consists of several steps in which the client and server communicate the information required for a handshake.

**Diagram

Description automatically generated**

**DIAGRAM 1**

Source- <https://www.semrush.com/blog/what-is-https/>

Diagram

Description automatically generated

**DIAGRAM 2**

Source: <https://tiptopsecurity.com/how-does-https-work-rsa-encryption-explained/>

**REFERENCES:**

Harnish, Brian. “What Is HTTPS: The Definitive Guide to How HTTPS Works.” Semrush Blog, Dec. 2020, <https://www.semrush.com/blog/what-is-https/>.

“What Happens in TLS Handshake.” Www.Cloudflare.Com, 2022, <https://www.cloudflare.com/learning/ssl/what-happens-in-a-tls-handshake/>.

Contributors to Wikimedia projects. “HTTPS - Wikipedia.” Wikipedia, the Free Encyclopedia, Wikimedia Foundation, Inc., 26 July 2001, <https://en.wikipedia.org/wiki/HTTPS>.