Introduction

HTTP, also known as HyperText Transfer Protocol, serves as the fundamental protocol for data communication on the World Wide Web. HTTP was developed by Tim Berners-Lee and his colleagues over the period of 1989-1991. It permits data transmission between web servers and clients, which requests are initiated by the recipient that is typically a web browser. As HTTP allows for the transmission of numerous sorts of data, including text, graphics, and multimedia, so all elements can be seen on webpages.

Rapid growth of the Internet has led to an increase in web security concerns. In order to overcome the problems of HTTP, especially the protection of the privacy and web safety as traditional HTTP was unencrypted, making data easily accessible to monitoring and distorting. Thus, HTTPS (HyperText Transfer Protocol Secure) was established.

HTTPS was developed to enhance the security of HTTP by implementing encryption methods like SSL/TLS. This ensure that any data transmitted between the web server and the web browser is encrypted and protected. HTTPS also provides website authentication, preventing man-in-the-middle attacks and data breaches. Moreover, it protects the integrity of the data for preventing any modifications or corruption in the process of transmitting it.

The latest developments in encryption technology have improved the performance of HTTPS, with small extra expenses compared to using HTTP and without negatively affecting the user's overall experience. Search engines such as Google have begun using HTTPS as a ranking indication, motivating websites to use encrypted links in order to enhance their search exposure.

Moreover, it is strongly advised that websites use HTTPS in order to satisfy rules such as the Personal Data (Privacy) Ordinance (PDPO) in Hong Kong and the Payment Card Industry Data Security Standard (PCI DSS). This would enhance the secure transmission of sensitive data. These safety measures are also important for online transactions, personal data exchange and protecting privacy in the digital world. It has become essential for each website running on the internet nowadays.

Knowledge

The major difference between HTTP and HTTPS is how they handle the data sent out when HTTPS will perform encryption, but HTTP won't. Encryption is a security method that will scramble the data, and it will only be able to be viewed by authorized users. The original version of the data document from the sender before any encryption is called plaintext; the encryption method can be one or multi-algorithm only known by the sender and receiver. After the process of encryption, the encrypted document received is called ciphertext.

When performing various internet actions and events, data communication occurs, and the process can be monitored or reviewed by using the network analyser tool named a packet analyser, also known by several names, such as packet sniffer, protocol analyser, or network analyser. Wireshark is one of the popular packet analysers in ethical hacking and illegal attacks; in the following part, there will use Wireshark to review the packet while browsing the and see what HTTP look like.

An HTTP packet is usually divided into two parts: HTTP request and HTTP response. HTTP is performed under the distributed application structure called the client-server model. It is a model of two attributes of client and server: the client (a machine like a smartphone or laptop) will send a request through the internet, and the server will respond with the requested information back to the client.

Discussion

HTTPS uses the SSL/TLS protocol to encrypt data. Data is encrypted as it is sent from the client to the server, making it impossible for an attacker to easily access sensitive information even if it is intercepted. In contrast, HTTP transmits data in plaintext, making it vulnerable to eavesdropping and interception.

Example:

Suppose you connect to an HTTP site using public Wi-Fi and submit your login credentials. Since the data is transmitted in plaintext, an attacker can easily intercept your username and password. However, if you connect to the same website using HTTPS, your data will be encrypted, making it impossible for an attacker to easily access your sensitive information.

HTTPS verifies the integrity of data by using message authentication codes. This means that any tampering with the data during transmission will be detected.

If the data is tampered with, the receiver will sense it and can refuse to process the tampered data. HTTP, on the other hand, does not provide this mechanism,

resulting in data being easily tampered with during transmission without being detected.

Example:

Assuming there is an application programming interface, which records some data, and the data is public, all the data is not encrypted communication, HTTP may have the following problems:

Integrity damage: the data content may be changed by a third party resulting in the API caller obtaining false data.

Authenticity damage: The API provider has not yet returned the data; the data may have been forged by an intermediary to return a copy of the data. With MAC algorithm, the message sender calculates the MAC value based on and sends it together with the message to the receiver; after the receiver receives the message, it repeats the above steps to calculate the MAC value and compares it with the passed MAC value, if it is different, it means that the message has been tampered with, then it is immediately discarded.

HTTPS uses digital certificates to authenticate a server. Digital certificates are issued by trusted third-party organisations and are used to prove that the server is legitimate and associated with a specific domain name. This prevents man-in-the-middle attacks and forged websites. When an HTTPS connection is established,

the client verifies the server's digital certificate to ensure that the connection is destined for the intended legitimate server.

Example:

Suppose you are conducting an online banking transaction, and you want to make sure that you are connecting to your bank's real website and not a spoofed one. If you use an HTTP connection, an attacker could spoof a website that looks identical to your bank's and direct you to submit sensitive information. However, if you connect using HTTPS, the browser will verify the server's digital certificate, ensuring that you are connecting to a legitimate bank website. For example, the official website of the Bank of China is https://www.boc.cn/, while the fake site might be https://www.boc.cm/. The domain names of the two-look similar but are completely different.

Cracking of symmetric encryption algorithms is mainly based on trial-and-error methods such as exhaustive attacks or dictionary attacks. These attacks try to decrypt encrypted data using various possible keys. Asymmetric Encryption Cracking: Cracking of asymmetric encryption algorithms usually involves mathematically related attacks such as large number decomposition (for the RSA algorithm) or the discrete logarithm problem (for the Elliptic Curve Encryption Algorithm). These attacks are currently very time-consuming and complex, requiring significant computational resources and time. However, with advances in computational techniques and new mathematical discoveries, certain asymmetric encryption algorithms may be potentially at risk.

HTTPS relies on digital certificates to authenticate servers. If an attacker is able to forge a digital certificate or obtain a private key, they may be able to conduct a man-in-the-middle attack to steal sensitive information or tamper with data.