HTTPS: Securing Web Communication

7 min to complete · By Ryan Desmond, Jared Larsen

Contents

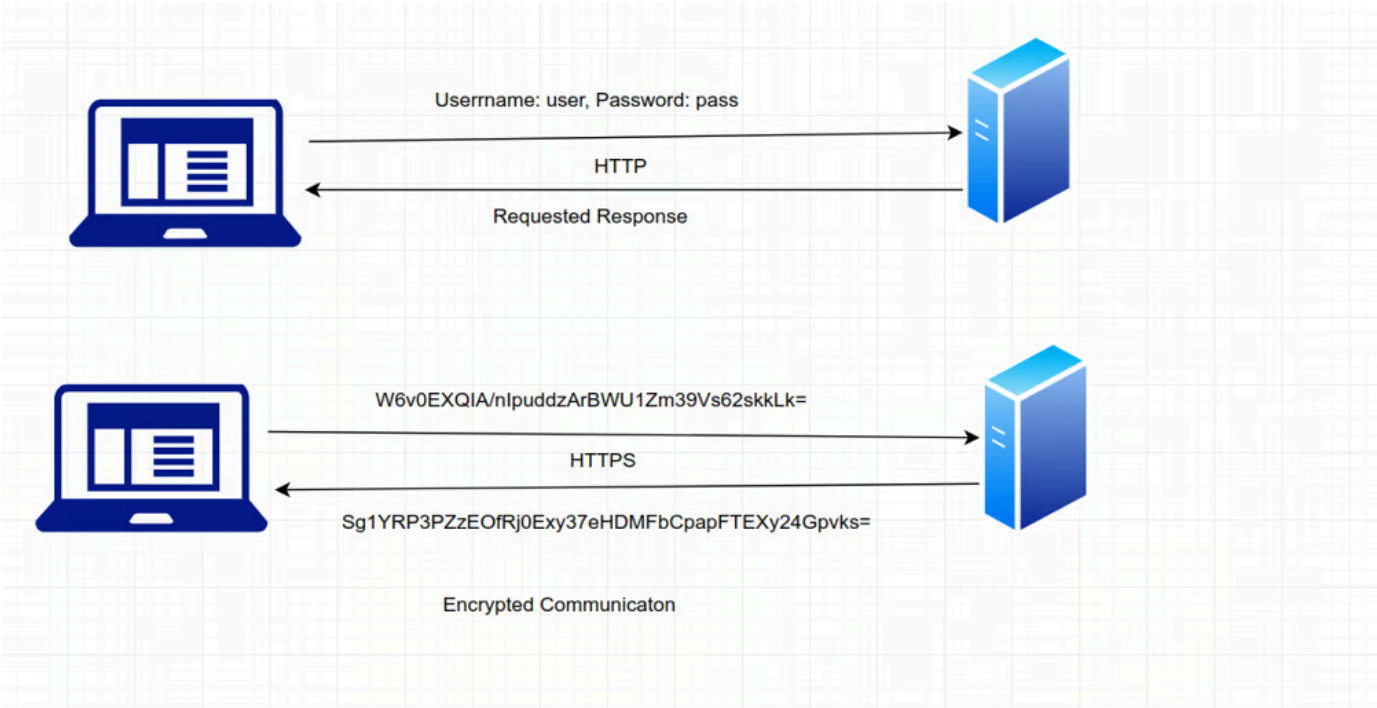
Each day, the amount of valuable information about us that is being stored and transferred over the internet continues to grow exponentially. From personal details to financial transactions, data is continuously sent back and forth from system to system. This makes internet communications a prime target for theft and interception. In a world where digital data is everywhere, security is everything.

In order to combat this, HTTPS was created. This lesson introduces the HTTPs protocol, and the difference between HTTP vs. HTTPs.

What is HTTPS?

HTTPS stands for Hyper Text Transfer Protocol **Secure**. HTTPs is an extension of HTTP designed to provide a secure channel for communication over the internet. By encrypting the data transferred between two systems, HTTPS ensures that the information remains confidential and protected from eavesdropping and tampering.

Like its name suggests, the HTTPs adapted protocol secures requests and responses transferred between two systems. In other words, if HTTPS messages are intercepted by a nefarious actor, all they see is gibberish as is shown in the following image.



How does HTTPS Work?

Creating the HTTPs protocol requires some fancy encryption footwork. Here's a quick overview.

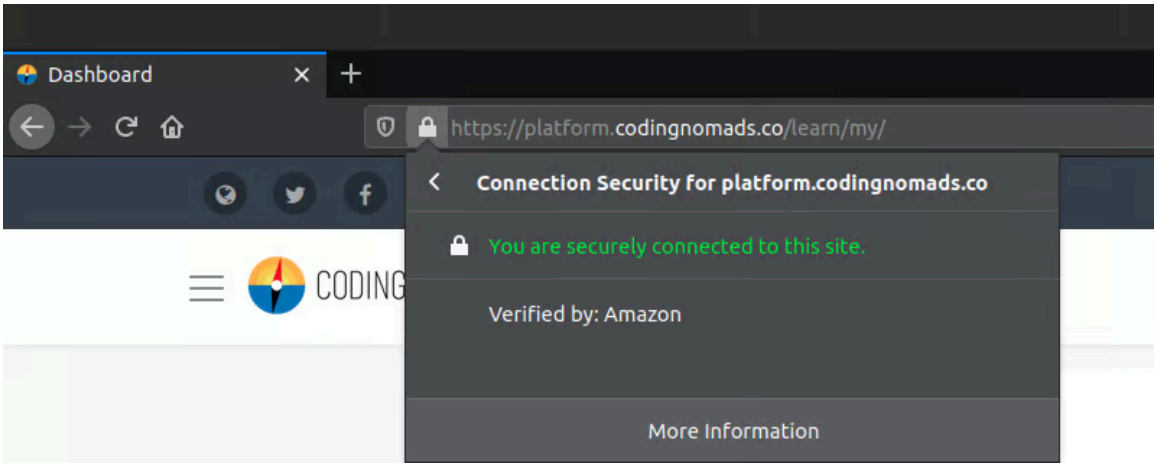
HTTPS adds a new SSL/TSL encryption layer when establishing a connection between client and server, then uses the established SSL/TSL protocol to secure the information. This extra step fits in between the DNS/NAT operations and the actual exchange of resources. This extra step is usually referred to as a handshake. The two systems introduce themselves, and they agree on methods to communicate.

The handshake is split up into three sections:

1. **Introductions**: Having found the server's IP address, the client sends a "hello" message. This message includes all the public information needed to establish a secure connection.
2. **Exchange of Certificates**: This part of the handshake is critical. It allows both systems to confirm that the entity they are talking to is actually the intended recipient. If the certificates are confirmed to be authentic, the handshake proceeds. Otherwise, the client is warned that the certificate provided by the server is invalid.

3. **Exchange of Encryption Keys**: Keys to encrypt and decrypt messages are exchanged, and allow sensitive information to be communicated between client and server.

Here's how a browser lets you know if the connection is secure:



Advantages of the HTTPS Protocol

- **Security** It becomes very hard to access and tamper with information while it is in transit between client and server.
- **Trust**:Certificates and the verification of these certificates allows us to ensure we are talking to the right server and not an imposter.
- **Search Engine Optimization**: Google and other search engines prioritize HTTPS sites over HTTP sites when displaying results. Hence, if you implement HTTPS you'll be much more visible.

Tip: Implementing HTTPS over HTTP has very much become standard practice these days for any data communications that occur on the internet.

Difference Between HTTP vs. HTTPS

Take a look at this table. It lays out some key differences in how HTTP and HTTPS function:

Parameter	HTTP	HTTPS
Name	Hyper Text Transfer Protocol	Hyper Text Transfer Protocol Secure
Port	port: 80 by default	port: 443 by default
URL Naming Convention	http://	https://
SSL/TLS?	HTTP does not need SSL/TLS	HTTPS requires SSL/TLS certificate
Search Engine Optimization (SEO)	HTTP does not improve search ranking	HTTPS improves site ranking
Vulnerability	Not secured. Very vulnerable to intrusion	Highly Secured. All data is encrypted

Note: It is important to mention that while HTTPS does make communication over the internet more secure, it does not protect you from *all* attacks.

There are many security concerns to address when deploying web applications, do not think you are safe by just implementing HTTPS!

Summary: What is HTTPS?

HTTPS stands for Hyper Text Transfer Protocol **Secure**. This lesson covered a lot of technical procedures for establishing encrypted connections with HTTPS. The most important points to remember are:

- HTTP vs. HTTPS: HTTPs is an extension of HTTP that provides a more secure channel for communication over the internet by encrypting the data transferred.
- HTTPS secures the information sent from client to server.
- You should use HTTPS instead of HTTP in all possible situations.
- HTTPS does not mean your web app is fully secure. There are many other vulnerabilities to look out for.

Previous Next → RESTful APIs & Endpoints

Git & GitHub
SQL + Databases

Django Web Development
Flask Web Development

Career Tracks

Java Engineering Career Track
Python Web Dev Career Track
Data Science / ML Career Track
Career Services

Resources

About CodingNomads
Corporate Partnerships
Contact us
Blog
Discord