Why is HTTP not secure? | HTTP vs. HTTPS

HTTP requests and responses are sent in plaintext, which means that anyone can read them. HTTPS corrects this problem by using TLS/SSL encryption.

Learning Objectives

After reading this article you will be able to:

- Understand how HTTP works
- Learn why HTTP is not secure
- Explain the differences between HTTP and HTTPS
- Learn how to get an SSL certificate for HTTPS

RELATED CONTENT

What is HTTPS?

Why Use HTTPS?

SSL Handshake

HTTP vs. HTTPS: What are the differences?

HTTPS is HTTP with encryption. The only difference between the two protocols is that HTTPS uses TLS (SSL) to encrypt normal HTTP requests and responses. As a result, HTTPS is far more secure than HTTP. A website that uses HTTP has http:// in its URL, while a website that uses HTTPS has https://.

What is HTTP?

HTTP stands for Hypertext Transfer Protocol, and it is a protocol – or a prescribed order and syntax for presenting information – used for transferring data over a network. Most information that is sent over the Internet, including website content and API calls, uses the HTTP protocol. There are two main kinds of HTTP messages: requests and responses.

In the OSI model (see What is the OSI model?), HTTP is a layer 7 protocol.

What is an HTTP request? What is an HTTP response?

HTTP requests are generated by a user's browser as the user interacts with web properties. For example, if a user clicks on a hyperlink, the browser will send a series of "HTTP GET" requests for the content that appears on that page. If someone Googles "What is HTTP?" and this article shows up in the search results, when they click on the link, their browser will create and send a series of HTTP requests in order to get the information necessary to render the page.

These HTTP requests all go to either an origin server or a proxy caching server, and that server will generate an HTTP response. HTTP responses are answers to HTTP requests.

What does a typical HTTP request look like?

An HTTP request is just a series of lines of text that follow the HTTP protocol. A GET request might look like this:

```
GET /hello.txt HTTP/1.1
User-Agent: curl/7.63.0 libcurl/7.63.0 OpenSSL/1.1.1 zlib/1.2.11
Host: www.example.com
Accept-Language: en
```

This section of text, generated by the user's browser, gets sent across the Internet. The problem is, it's sent just like this, in plaintext that anyone monitoring the connection can read. (Those who are unfamiliar with the HTTP protocol may find this text hard to understand, but anyone with a baseline knowledge of the protocol's commands and syntax can read it easily.)

This is especially an issue when users submit sensitive data via a website or a web application. This could be a password, a credit card number, or any other data entered into a form, and in HTTP all this data is sent in plaintext for anyone to read. (When a user submits a form, the browser translates this into an HTTP POST request instead of an HTTP GET request.)

When an origin server receives an HTTP request, it sends an HTTP response, which is similar:

```
HTTP/1.1 200 OK
Date: Wed, 30 Jan 2019 12:14:39 GMT
Server: Apache
Last-Modified: Mon, 28 Jan 2019 11:17:01 GMT
Accept-Ranges: bytes
Content-Length: 12
Vary: Accept-Encoding
Content-Type: text/plain

Hello World!
```

If a website uses HTTP instead of HTTPS, all requests and responses can be read by anyone who is monitoring the session. Essentially, a malicious actor can just read the text in the request or the response and know exactly what information someone is asking for, sending, or receiving.

What is HTTPS?

The S in HTTPS stands for "secure." HTTPS uses TLS (or SSL) to encrypt HTTP requests and responses, so in the example above, instead of the text, an attacker would see a bunch of seemingly random characters.

Instead of:

```
GET /hello.txt HTTP/1.1
User-Agent: curl/7.63.0 libcurl/7.63.0 OpenSSL/1.1.1 zlib/1.2.11
Host: www.example.com
Accept-Language: en
```

The attacker sees something like:

```
t8Fw6T8UV81pQfyhDkhebbz7+oiwldr1j2gHBB3L3RFTRsQCpaSnSBZ78Vme+DpDVJPvZdZUZHpzbbcqmSW1+3xXGsERHg9YDmpYk0VVDiRvw1H5miNieJeJ/FNUjgH0BmVRWII6+T4MnDwmCMZUI/orxP3HGwYCSIvyzS3Mpmm
Se4iaWKCOHQ==
```

In HTTPS, how does TLS/SSL encrypt HTTP requests and responses?

TLS uses a technology called public key encryption: there are two keys, a public key and a private key, and the public key is shared with client devices via the server's SSL certificate. When a client opens a connection with a server, the two devices use the public and private key to agree on new keys, called session keys, to encrypt further communications between them.

All HTTP requests and responses are then encrypted with these session keys, so that anyone who intercepts communications can only see a random string of characters, not the plaintext.

For more on how encryption and keys work, see What is encryption?

How does HTTPS help authenticate web servers?

Authentication means verifying that a person or machine is who they claim to be. In HTTP, there is no verification of identity – it's based on a principle of trust. The architects of HTTP didn't necessarily make a decision to implicitly trust all web servers; they simply had priorities other than security at the time. But on the modern Internet, authentication is essential.

Just like an ID card confirms a person's identity, a private key confirms server identity. When a client opens a channel with an origin server (e.g. when a user navigates to a website), possession of the private key that matches with the public key in a website's SSL certificate proves that the server is actually the legitimate host of the website. This prevents or helps block a number of attacks that are possible when there is no authentication, such as:

- On-path attacks
- DNS hijacking
- BGP hijacking

- Domain spoofing

How does Cloudflare enable websites to adopt HTTPS?

Cloudflare released Universal SSL in 2014 and was the first company to make SSL certificates free. Any website that is signed up for Cloudflare services can enable HTTPS and move away from HTTP with one click. This makes TLS encryption widely available, to protect users and user data all over the Internet.

To learn more about HTTP vs. HTTPS, see What is mixed content? To test that a website offers HTTPS, visit the Cloudflare Diagnostic Center.