УДК 004.738.1.056:811.111

Kosyakova D., Molchan O.

What Is HTTPS and What Does It Do?

Belarusian National Technical University
Minsk, Belarus

HTTP was originally proposed by Tim Berners-Lee, who designed the application protocol in mind to perform high-level data communication functions between Web-servers and clients. It takes the well-known and understood HTTP protocol, and simply layers a SSL/TLS encryption layer on top of it. Servers and clients still speak exactly the same HTTP to each other, but over a secure SSL connection that encrypts and decrypts their requests and responses. The SSL layer has 2 main purposes: verifying that you are talking directly to the server that you think you are talking to; ensuring that only the server can read what you send it and only you can read what it sends back.

The really clever part is that anyone can intercept every single one of the messages you exchange with a server, including the ones where you are agreeing on the key and encryption strategy to use, and still not be able to read any of the actual data you send.

Let’s have a closer look at how an SSL connection is established. An SSL connection between a client and server is set up by a handshake, the goals of which is to agree on a *cipher suite*; to agree on any necessary keys for this algorithm.

Once the connection is established, both parties can use the agreed algorithm and keys to securely send messages to each other. We will break the handshake up into 3 main phases – Hello, Certificate Exchange and Key Exchange:

1. *Hello* – The handshake begins with the client sending a ClientHello message. This contains all the information the server needs in order to connect to the client via SSL, including the various cipher suites and maximum SSL version that it supports. The server responds with a ServerHello, which contains similar information required by the client, including a decision based on the client's preferences about which cipher suite and version of SSL will be used [1].

2. *Certificate Exchange* – When you request a HTTPS connection to a webpage, the website will initially send its SSL certificate to your browser. This certificate contains the public key needed to begin the secure session. Based on this initial exchange, your browser and the website then initiate the *SSL handshake*. The SSL handshake involves the generation of shared secrets to establish a uniquely secure connection between yourself and the website.

When a trusted SSL Digital Certificate is used during a HTTPS connection, users will see a padlock icon in the browser address bar. When an Extended Validation Certificate is installed on a web site, the address bar will turn green.

All communications sent over regular HTTP connections are in *plain text* and can be read by any hacker that manages to break into the connection between your browser and the website. This presents a clear danger if the *communication* is on an order form and includes your credit card details or social security number. With a HTTPS connection, all communications are securely encrypted. This means that even if somebody managed to break into the connection, they would not be able decrypt any of the data which passes between you and the website [2].

3. *Key Exchange* – The encryption of the actual message data exchanged by the client and server will be done using a symmetric algorithm, the exact nature of which was already

agreed during the Hello phase. A symmetric algorithm uses a single key for both encryption and decryption, in contrast to asymmetric algorithms that require a public/private key pair. Both parties need to agree on this single, symmetric key, a process that is accomplished securely using asymmetric encryption and the server's public/private keys.

The client generates a random key to be used for the main, symmetric algorithm. It encrypts it using an algorithm also agreed upon during the Hello phase, and the server's public key (found on its SSL certificate). It sends this encrypted key to the server, where it is decrypted using the server's private key, and the interesting parts of the handshake are complete. The parties are happy that they are talking to the right person, and have secretly agreed on a key to symmetrically encrypt the data that they are about to send each other. HTTP requests can now be sent by forming a plaintext message and then encrypting and sending it. The other party is the only one who knows how to decrypt this message, and so cybercriminals are unable to read or modify any requests that they may intercept [1].

References:

1. How does HTTPS actually work? [Electronic resource]. – Mode of access: https://robertheaton.com/2014/03/27/how-does-https-actually-work/. – Date of access: 19.03.2018.
2. What is HTTPS? [Electronic resource]. – Mode of access: https://www.instantssl.com/ssl-certificate-products/https.html. – Date of access: 19.03.2018.