



computer
emergency
response
team

CERT-EU
for the EU institutions, bodies
and agencies

CERT-EU Security Whitepaper 2016-001

Improved Security with HTTPS

Krzysztof SOCHA, Georgios PSYKAKOS

ver. **1.0**

April 26, 2016

TLP: WHITE

Scope

The scope of this whitepaper is to present in a simple way the advantages of using HTTPS instead of HTTP. This document aims at providing high-level information and advice to website owners and everyday users of the Internet services. The purpose is to raise the security awareness of the target audience, and not to provide a complete technical discussion of the implementation of HTTPS. Details related to more technical aspects can be found instead in [1, 2, 3, 4].

Summary

With the increasing popularity and availability of web-based applications, it becomes very important to ensure a secure way for accessing them. Security could be significantly improved by moving from using HTTP to HTTPS protocol. HTTPS addresses several key vulnerabilities of HTTP protocol by ensuring confidentiality and integrity of data being exchanged, as well as allowing the verification of the web server identity [5].

HTTPS needs to be supported by both client (a web browser) and a web server. Most modern web browsers support it out-of-the-box. To be supported by the server, it is the responsibility of the website owners to enable and configure it properly. While HTTPS does not solve all potential problems related to browsing, it significantly improves the security posture of both advanced web applications as well as casual browsing. Nevertheless, as HTTPS does not solve all security problems, it should not be considered independently, but rather as part of a holistic security approach. Any system is only as strong as its weakest component, so the general guidelines for securing and hardening computer environments apply, as well as general rules and recommendations related to user training and awareness.

HTTPS Overview

HTTPS – or HTTP over TLS (formerly also SSL) – is a protocol identical to standard HTTP (Hyper-Text Transfer Protocol) with an additional layer of TLS (Transport Layer Security) [1]. The basic idea consists of preceding any transfer of data with a secure handshake, which allows creation and secure exchange of session key. The secure handshake is possible thanks to the use of X.509 certificate [2] that is issued to the server from one of the generally recognized Certificate Authorities (CAs).

A session key is then subsequently used to secure the HTTP data transferred between the client and the server. While the specific details of the secure layer remain outside of scope of this document, it is important to note that TLS (or formerly also SSL) support multiple TLS/SSL protocol versions and internally multiple encryption and hashing algorithms. Not all combinations are considered as secure as others, and future research in cryptography will most likely cause further evolution and changes. More information about current up-to-date level of TLS/SSL ratings can be found in [4].

Advantages of HTTPS

HTTPS aims at fixing a few significant vulnerabilities of HTTP protocol. HTTP-S *for security* means:

- **Identification** The client can verify the identity of the server providing the Internet services.
- **Confidentiality** The content of the data exchanged as part of a web transaction cannot be accessed by unauthorized party that may have intercepted the exchange.
- **Data integrity** Any change in the data exchanged as part of a web transaction can be detected and therefore rejected.

Server Identity

Server identity verification is accomplished through certificates. A public key certificate is a file that uses a digital signature to bind a machine's public key with its identity. This digital signature is provided by Certification Authority who guarantees that a particular key belongs to a specific organisation or individual. The process of issuing this guarantee is called *signing the certificate*.

A common form of attack is an attempt to trick a user to visit a site that looks like a legitimate site (e.g., of a bank, institution, or company), but which in fact is malicious and could capture sensitive data provided by the user. A way of executing this attack is to redirect the traffic from the user to the malicious site in a way that is transparent to the user.

In order to protect the end-user to some extent, HTTPS protocol requires the client browser to verify the certificate of the server before any data is transmitted. In the hypothetical case, when the certificate provided by the server does not match the site to which the user wished to connect, the connection will be blocked and the user warned.

HTTPS is however unable to help in some cases. If a user decides to ignore the warning, or if a user enters incorrect URL (e.g., by copy-pasting from suspicious sources), or fails to notice a redirection to an incorrect URL or to a site that does not use HTTPS, the user may still be vulnerable to an attack.

Data Confidentiality

Today, with wireless networks being increasingly popular, a large number of people can have very easy access to the data in transit. For open, unencrypted networks - any person in the vicinity can potentially have access to the data. For encrypted wireless networks this limits the access only to the people having access to this network, i.e., those who know the password. Often this may be a surprisingly large group.

Without HTTPS all data exchanged between client and server are sent in clear text. It means that anybody having access to the transmission has also access to all this data. The data could contain usernames and passwords, financial data, and other sensitive

data depending on the site or web application being accessed. This data could be potentially available to anyone sniffing on the network.

Contrary, when HTTPS is used, all data exchanged is encrypted with a special session key, which prevents eavesdropping on the data being exchanged. It is then only accessible for the client and the server involved in the communication.

Certainly, the security of the data exchanged using HTTPS depends also on the security of the end-points. Hence, the security of both the client and the server should be also taken into consideration.

Data Integrity

Without HTTPS, a persistent attacker who is able to sniff data in transit may be also able to modify data. This could result in fake information appearing on downloaded content, but also – potentially more dangerous – wrong data being sent to the server. In the latter case, one could imagine changing for instance bank account numbers, when initiating money transfer.

On the other hand, if data flowing across an HTTPS session is altered, the receiving node will detect this condition and not pass the corrupted data to the application. When HTTPS is used, the data is protected and cannot be modified or replayed without detection. Hence, the risk of maliciously modified data being sent either to the server or to the client is significantly decreased.

What to Do?

Server Side

The owner of the website has to obtain a certificate for their server from one of the generally recognized CAs. Once the certificate is installed, the server needs to be configured to serve pages using HTTPS protocol.

Special care should be taken to properly handle requests that come using HTTP. While it is possible to serve pages using alternatively HTTP and HTTPS, such configuration is not recommended. A recommended configuration is a permanent redirection from HTTP version of the site to HTTPS version of the site.

It should be regularly ensured that a latest version of TLS is installed on the server, and any vulnerabilities discovered are timely patched. Also, the server configuration should be regularly reviewed to ensure that older cryptographic algorithms are removed. Finally, the server certificate needs to be regularly renewed.

It is important to note that the use of HTTPS creates some additional load on the server due to the need of encryption and decryption of the data transmitted. This additional load is usually not very significant, but should be evaluated before moving to HTTPS - especially for very busy servers.

The key management is critical to deliver the secure services of authentication, integrity and non-repudiation by the HTTPS architecture [7]. The integrity and confidentiality of the keys are essential to be protected. Their protection must be covered by business procedures and appropriate technical means.

Client Side

On the client side, it is important to use recent versions of the browsers, which support recent versions of TLS, patch any discovered vulnerabilities, and use regularly updated list of Root Certificates of CAs. A browser can be easily checked for its TLS/SSL support using for instance on-line service available in [6].

Furthermore, the user must be alerted[8]. The danger of being exploited exists even when using HTTPS. A careless user can undermine the use of HTTPS. The use of HTTPS must be verified whenever transactions of sensitive and financial data occur on the Internet. This can be easily done by observing the browsers URL bar and look for the HTTPS on the address or a symbol of a lock.

References

- [1] <https://tools.ietf.org/html/rfc2818> - RFC2818 defining HTTP over TLS
- [2] <http://www.ietf.org/rfc/rfc2459.txt> - Internet X.509 Public Key Infrastructure Certificate and CRL Profile
- [3] DIGIT.C INFOSEC Technical Standards SSL/TLS
- [4] <http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-52r1.pdf> - NIST Special Publication 800 - 52
- [5] <https://en.wikipedia.org/wiki/HTTPS> - good description of HTTPS
- [6] <https://www.howssmyssl.com> - a website that gives current ranking of SSL(TLS) clients.
- [7] http://csrc.nist.gov/publications/nistpubs/800-57/sp800-57_part1_rev3_general.pdf - NIST Recommendation for Key Management Part 1: General
- [8] https://www.enisa.europa.eu/publications/archive/copy_of_new-users-guide/at_download/fullReport - ENISA The new users' guide: How to raise information security awareness