[Template:Selfref](/wiki/Template:Selfref" \o "Template:Selfref) [Template:HTTP](/wiki/Template:HTTP) **HTTPS** (also called **HTTP over** [**TLS**](/wiki/Transport_Layer_Security),[[1]](#cite_note-1)[[2]](#cite_note-2) **HTTP over SSL**,[[3]](#cite_note-3) and **HTTP Secure**[[4]](#cite_note-4)[[5]](#cite_note-5)) is a [protocol](/wiki/Communications_protocol) for [secure](/wiki/Network_security) communication over a [computer network](/wiki/Computer_network) which is widely used on the [Internet](/wiki/Internet). HTTPS consists of communication over [Hypertext Transfer Protocol](/wiki/Hypertext_Transfer_Protocol) (HTTP) within a connection encrypted by [Transport Layer Security or its predecessor, Secure Sockets Layer](/wiki/Transport_Layer_Security). The main motivation for HTTPS is [authentication](/wiki/Authentication) of the visited [website](/wiki/Website) and protection of the [privacy](/wiki/Information_privacy) and [integrity](/wiki/Data_integrity) of the exchanged data.

In its popular deployment on the internet, HTTPS provides authentication of the website and associated [web server](/wiki/Web_server) with which one is communicating, which protects against [man-in-the-middle attacks](/wiki/Man-in-the-middle_attack). Additionally, it provides bidirectional [encryption](/wiki/Encryption) of communications between a client and server, which protects against [eavesdropping](/wiki/Eavesdropping) and [tampering](/wiki/Tamper-evident#Tampering) with or forging the contents of the communication.<ref name=httpse>[Template:Cite web](/wiki/Template:Cite_web)</ref> In practice, this provides a reasonable guarantee that one is communicating with precisely the website that one intended to communicate with (as opposed to an impostor), as well as ensuring that the contents of communications between the user and site cannot be read or forged by any third party.

Historically, HTTPS connections were primarily used for payment transactions on the [World Wide Web](/wiki/World_Wide_Web), e-mail and for sensitive transactions in corporate information systems. In the late 2000s and early 2010s, HTTPS began to see widespread use for protecting page authenticity on all types of websites, securing accounts and keeping user communications, identity and [web browsing](/wiki/Web_browsing) private.

## Contents

* 1 Overview[[edit](/index.php?title=(none)&action=edit&section=1)]
  + 1.1 Usage in websites[[edit](/index.php?title=(none)&action=edit&section=2)]
  + 1.2 Browser integration[[edit](/index.php?title=(none)&action=edit&section=3)]
* 2 Security[[edit](/index.php?title=(none)&action=edit&section=4)]
* 3 Technical[[edit](/index.php?title=(none)&action=edit&section=5)]
  + 3.1 Difference from HTTP[[edit](/index.php?title=(none)&action=edit&section=6)]
  + 3.2 Network layers[[edit](/index.php?title=(none)&action=edit&section=7)]
  + 3.3 Server setup[[edit](/index.php?title=(none)&action=edit&section=8)]
    - 3.3.1 Acquiring certificates[[edit](/index.php?title=(none)&action=edit&section=9)]
    - 3.3.2 Use as access control[[edit](/index.php?title=(none)&action=edit&section=10)]
    - 3.3.3 In case of compromised secret (private) key[[edit](/index.php?title=(none)&action=edit&section=11)]
  + 3.4 Limitations[[edit](/index.php?title=(none)&action=edit&section=12)]
* 4 History[[edit](/index.php?title=(none)&action=edit&section=13)]
* 5 See also[[edit](/index.php?title=(none)&action=edit&section=14)]
* 6 References[[edit](/index.php?title=(none)&action=edit&section=15)]
* 7 External links[[edit](/index.php?title=(none)&action=edit&section=16)]

## Overview[[edit](/index.php?title=(none)&action=edit&section=1)]

[Template:Details](/wiki/Template:Details) [thumb|Logo of the networking protocol https and the www letters](/wiki/File:Internet2.jpg)

The *HTTPS* [uniform resource identifier](/wiki/Uniform_resource_identifier) (URI) scheme has identical syntax to the standard HTTP scheme, aside from its scheme token. However, HTTPS signals the browser to use an added encryption layer of SSL/TLS to protect the traffic. SSL/TLS is especially suited for HTTP since it can provide some protection even if only one side of the communication is [authenticated](/wiki/Authentication). This is the case with HTTP transactions over the Internet, where typically only the [server](/wiki/Webserver) is authenticated (by the client examining the server's [certificate](/wiki/Public_key_certificate)).

HTTPS creates a secure channel over an insecure network. This ensures reasonable protection from [eavesdroppers](/wiki/Eavesdropping) and [man-in-the-middle attacks](/wiki/Man-in-the-middle_attack), provided that adequate [cipher suites](/wiki/Cipher_suite) are used and that the server certificate is verified and trusted.

Because HTTPS piggybacks HTTP entirely on top of TLS, the entirety of the underlying HTTP protocol can be encrypted. This includes the request URL (which particular web page was requested), query parameters, headers, and cookies (which often contain identity information about the user). However, because host (website) addresses and port numbers are necessarily part of the underlying [TCP/IP](/wiki/TCP/IP) protocols, HTTPS cannot protect their disclosure. In practice this means that even on a correctly configured web server, eavesdroppers can infer the IP address and port number of the web server (sometimes even the domain name e.g. www.example.org, but not the rest of the URL) that one is communicating with as well as the amount (data transferred) and duration (length of session) of the communication, though not the content of the communication.[[6]](#cite_note-6) Web browsers know how to trust HTTPS websites based on [certificate authorities](/wiki/Certificate_authorities) that come pre-installed in their software. Certificate authorities (such as [Symantec](/wiki/Symantec), [Comodo](/wiki/Comodo_Group), [GoDaddy](/wiki/GoDaddy) and [GlobalSign](/wiki/GlobalSign)) are in this way being trusted by web browser creators to provide valid certificates. Therefore, a user should trust an HTTPS connection to a website [if and only if](/wiki/If_and_only_if) all of the following are true:

* The user trusts that the browser software correctly implements HTTPS with correctly pre-installed certificate authorities.
* The user trusts the certificate authority to vouch only for legitimate websites.
* The website provides a valid certificate, which means it was signed by a trusted authority.
* The certificate correctly identifies the website (e.g., when the browser visits "https://example.com", the received certificate is properly for "example.com" and not some other entity).
* The user trusts that the protocol's encryption layer (SSL/TLS) is sufficiently secure against eavesdroppers.

HTTPS is especially important over insecure networks (such as public WiFi access points), as anyone on the same local network can [packet sniff](/wiki/Packet_analyzer) and discover sensitive information not protected by HTTPS. Additionally, many free to use and even paid for WLAN networks engage in [packet injection](/wiki/Packet_injection) in order to serve their own ads on webpages. However, this can be exploited maliciously in many ways, such as injecting malware onto webpages and stealing users' private information.[[7]](#cite_note-7) HTTPS is also very important for connections over the [Tor anonymity network](/wiki/Tor_(anonymity_network)), as malicious Tor nodes can damage or alter the contents passing through them in an insecure fashion and inject malware into the connection. This is one reason why the [Electronic Frontier Foundation](/wiki/Electronic_Frontier_Foundation) and the Tor project started the development of [HTTPS Everywhere](/wiki/HTTPS_Everywhere),<ref name=httpse/> which is included in the Tor Browser Bundle.[[8]](#cite_note-8) As more information is revealed about global [mass surveillance](/wiki/Mass_surveillance) and criminals stealing personal information, the use of HTTPS security on all websites is becoming increasingly important regardless of the type of Internet connection being used.[[9]](#cite_note-9)[[10]](#cite_note-10) While [metadata](/wiki/Metadata) about individual pages that a user visits is not sensitive, when combined together, they can reveal a lot about the user and compromise the user's privacy.[[2]](#cite_note-2)[[11]](#cite_note-11)<ref name=deployhttpscorrectly/>

Deploying HTTPS also allows the use of [SPDY](/wiki/SPDY)/[HTTP/2](/wiki/HTTP/2), that are new generations of HTTP, designed to reduce page load times and latency.

It is recommended to use [HTTP Strict Transport Security](/wiki/HTTP_Strict_Transport_Security) (HSTS) with HTTPS to protect users from man-in-the-middle attacks, especially [SSL stripping](/wiki/Moxie_Marlinspike#SSL_stripping).<ref name=deployhttpscorrectly>[Template:Cite web](/wiki/Template:Cite_web)</ref>[[12]](#cite_note-12) HTTPS should not be confused with the little-used [Secure HTTP](/wiki/Secure_Hypertext_Transfer_Protocol) (S-HTTP) specified in RFC 2660.

### Usage in websites[[edit](/index.php?title=(none)&action=edit&section=2)]

As of June 28, 2016, 10.2% of Alexa top 1,000,000 websites use HTTPS as default.<ref name=StatOperator>[Template:Cite web](/wiki/Template:Cite_web)</ref>

As of June 1, 2016, 43.1% of the Internet's 141,387 most popular websites have a secure implementation of HTTPS.<ref name=sslpulse>[Template:Cite web](/wiki/Template:Cite_web)</ref>

### Browser integration[[edit](/index.php?title=(none)&action=edit&section=3)]

Most browsers display a warning if they receive an invalid certificate. Older browsers, when connecting to a site with an invalid certificate, would present the user with a dialog box asking if they wanted to continue. Newer browsers display a warning across the entire window. Newer browsers also prominently display the site's security information in the [address bar](/wiki/Address_bar). [Extended validation certificates](/wiki/Extended_validation_certificate) turn the address bar green in newer browsers. Most browsers also display a warning to the user when visiting a site that contains a mixture of encrypted and unencrypted content. [Template:Gallery](/wiki/Template:Gallery)

Firefox uses HTTPS for Google searches as of version 14,[[13]](#cite_note-13) to "shield our users from network infrastructure that may be gathering data about the users or modifying/censoring their search results".[[14]](#cite_note-14) The [Electronic Frontier Foundation](/wiki/Electronic_Frontier_Foundation), opining that "In an ideal world, every web request could be defaulted to HTTPS", has provided an add-on called [HTTPS Everywhere](/wiki/HTTPS_Everywhere) for [Mozilla Firefox](/wiki/Mozilla_Firefox) that enables HTTPS by default for hundreds of frequently used websites. A beta version of this plugin is also available for [Google Chrome](/wiki/Google_Chrome) and Chromium.[[15]](#cite_note-15)[[16]](#cite_note-16)

## Security[[edit](/index.php?title=(none)&action=edit&section=4)]

[Template:Main article](/wiki/Template:Main_article) The security of HTTPS is that of the underlying TLS, which typically uses long-term public and private keys to generate a short term session key which is then used to encrypt the data flow between client and server. [X.509](/wiki/X.509) certificates are used to authenticate the server (and sometimes the client as well). As a consequence, [certificate authorities](/wiki/Certificate_authority) and [public key certificates](/wiki/Public_key_certificate) are necessary to verify the relation between the certificate and its owner, as well as to generate, sign, and administer the validity of certificates. While this can be more beneficial than verifying the identities via a [web of trust](/wiki/Web_of_trust), the [2013 mass surveillance disclosures](/wiki/2013_mass_surveillance_disclosures) drew attention to certificate authorities as a potential weak point allowing [man-in-the-middle attacks](/wiki/Man-in-the-middle_attack).[[17]](#cite_note-17)[[18]](#cite_note-18) An important property in this context is [forward secrecy](/wiki/Forward_secrecy), which ensures that encrypted communications recorded in the past cannot be retrieved and decrypted should long-term secret keys or passwords be compromised in the future. Not all web servers provide forward secrecy.<ref name=ecdhe>[SSL: Intercepted today, decrypted tomorrow](http://news.netcraft.com/archives/2013/06/25/ssl-intercepted-today-decrypted-tomorrow.html), Netcraft, 2013-06-25.</ref>[Template:Update inline](/wiki/Template:Update_inline)

A site must be completely hosted over HTTPS, without having part of its contents loaded over HTTP - for example, having scripts loaded insecurely - or the user will be vulnerable to some attacks and surveillance. Also having only a certain page that contains sensitive information (such as a log-in page) of a website loaded over HTTPS, while having the rest of the website loaded over plain HTTP, will expose the user to attacks. On a site that has sensitive information somewhere on it, every time that site is accessed with HTTP instead of HTTPS, the user and the session will get exposed. Similarly, [cookies](/wiki/HTTP_cookie) on a site served through HTTPS have to have the [secure attribute](/wiki/Secure_cookie) enabled.[[19]](#cite_note-19)

## Technical[[edit](/index.php?title=(none)&action=edit&section=5)]

### Difference from HTTP[[edit](/index.php?title=(none)&action=edit&section=6)]

HTTPS URLs begin with "https://" and use [port](/wiki/List_of_TCP_and_UDP_port_numbers) 443 by default, whereas [HTTP](/wiki/HTTP) [URLs](/wiki/URL) begin with "http://" and use [port](/wiki/List_of_TCP_and_UDP_port_numbers) 80 by default.

HTTP is not encrypted and is vulnerable to man-in-the-middle and eavesdropping attacks, which can let attackers gain access to website accounts and sensitive information, and modify webpages to inject [malware](/wiki/Malware) or advertisements. HTTPS is designed to withstand such attacks and is considered secure against them (with the exception of older, deprecated versions of SSL).

### Network layers[[edit](/index.php?title=(none)&action=edit&section=7)]

HTTP operates at the highest layer of the [TCP/IP model](/wiki/TCP/IP_model), the Application layer; as does the TLS security protocol (operating as a lower sublayer of the same layer), which encrypts an HTTP message prior to transmission and decrypts a message upon arrival. Strictly speaking, HTTPS is not a separate protocol, but refers to use of ordinary [HTTP](/wiki/HTTP) over an [encrypted](/wiki/Encryption) SSL/TLS connection.

Everything in the HTTPS message is encrypted, including the headers, and the request/response load. With the exception of the possible [CCA](/wiki/Chosen-ciphertext_attack) cryptographic attack described in the [limitations](/wiki/#Limitations) section below, the attacker can only know that a connection is taking place between the two parties and their domain names and IP addresses.

### Server setup[[edit](/index.php?title=(none)&action=edit&section=8)]

To prepare a web server to accept HTTPS connections, the administrator must create a [public key certificate](/wiki/Public_key_certificate) for the web server. This certificate must be signed by a trusted [certificate authority](/wiki/Certificate_authority) for the web browser to accept it without warning. The authority certifies that the certificate holder is the operator of the web server that presents it. Web browsers are generally distributed with a list of [signing certificates of major certificate authorities](/wiki/Root_certificate) so that they can verify certificates signed by them.

#### Acquiring certificates[[edit](/index.php?title=(none)&action=edit&section=9)]

Authoritatively signed certificates may be free[[20]](#cite_note-20)[[21]](#cite_note-21)or cost between [Template:Nowrap](/wiki/Template:Nowrap)[[22]](#cite_note-22) and [Template:Nowrap](/wiki/Template:Nowrap)[[23]](#cite_note-23) per year (in 2012–2014).

Organizations may also run their own certificate authority, particularly if they are responsible for setting up browsers to access their own sites (for example, sites on a company [intranet](/wiki/Intranet), or major universities). They can easily add copies of their own signing certificate to the trusted certificates distributed with the browser.

There also exists a peer-to-peer certificate authority, [CACert](/wiki/CACert). However, it is not included in the trusted root certificates of many popular browsers (e.g. Firefox, Chrome, Internet Explorer), which may cause warning messages to be displayed to end users.

[Let's Encrypt](/wiki/Let's_Encrypt), launched in April 2016,[[24]](#cite_note-24) provides free and automated SSL/TLS certificates to websites.[[25]](#cite_note-25) According to the [Electronic Frontier Foundation](/wiki/Electronic_Frontier_Foundation), "Let's Encrypt" will make switching from HTTP to HTTPS "as easy as issuing one command, or clicking one button."[[26]](#cite_note-26)

#### Use as access control[[edit](/index.php?title=(none)&action=edit&section=10)]

The system can also be used for client [authentication](/wiki/Authentication) in order to limit access to a web server to authorized users. To do this, the site administrator typically creates a certificate for each user, a certificate that is loaded into their browser. Normally, that contains the name and e-mail address of the authorized user and is automatically checked by the server on each reconnect to verify the user's identity, potentially without even entering a password.

#### In case of compromised secret (private) key[[edit](/index.php?title=(none)&action=edit&section=11)]

An important property in this context is [perfect forward secrecy](/wiki/Perfect_forward_secrecy) (PFS). Possessing one of the long-term asymmetric secret keys used to establish an HTTPS session should not make it easier to derive the short-term session key to then decrypt the conversation, even at a later time. [Diffie–Hellman key exchange](/wiki/Diffie–Hellman_key_exchange) (DHE) and [Elliptic curve Diffie–Hellman](/wiki/Elliptic_curve_Diffie–Hellman) key exchange (ECDHE) are in 2013 the only ones known to have that property. Only 30% of Firefox, Opera, and Chromium Browser sessions use it, and nearly 0% of Apple's Safari and Microsoft Internet Explorer sessions.<ref name=ecdhe/> Among the larger internet providers, only Google supports PFS since 2011 (State of September 2013).[Template:Citation needed](/wiki/Template:Citation_needed)

A certificate may be revoked before it expires, for example because the secrecy of the private key has been compromised. Newer versions of popular browsers such as [Firefox](/wiki/Firefox),[[27]](#cite_note-27) [Opera](/wiki/Opera_(web_browser)),[[28]](#cite_note-28) and [Internet Explorer](/wiki/Internet_Explorer) on [Windows Vista](/wiki/Windows_Vista)[[29]](#cite_note-29) implement the [Online Certificate Status Protocol](/wiki/Online_Certificate_Status_Protocol) (OCSP) to verify that this is not the case. The browser sends the certificate's serial number to the certificate authority or its delegate via OCSP and the authority responds, telling the browser whether or not the certificate is still valid.[[30]](#cite_note-30)

### Limitations[[edit](/index.php?title=(none)&action=edit&section=12)]

SSL/TLS comes in two options, simple and mutual. The mutual version is more secure, but requires the user to install a personal [client certificate](/wiki/Client_certificate) into their web browser in order to authenticate themselves.[Template:Citation needed](/wiki/Template:Citation_needed)

Whatever strategy is used (simple or mutual), the level of protection strongly depends on the correctness of the [implementation](/wiki/Implementation) of the [web browser](/wiki/Web_browser) and the server software and the actual [cryptographic algorithms](/wiki/Cipher) supported.

SSL/TLS does not prevent the entire site from being indexed using a [web crawler](/wiki/Web_crawler), and in some cases the [URI](/wiki/Uniform_resource_identifier) of the encrypted resource can be inferred by knowing only the intercepted request/response size.[[31]](#cite_note-31) This allows an attacker to have access to the [plaintext](/wiki/Plaintext) (the publicly available static content), and the [encrypted text](/wiki/Ciphertext) (the encrypted version of the static content), permitting a [cryptographic attack](/wiki/Chosen-ciphertext_attack).

Because [TLS](/wiki/Transport_Layer_Security) operates below HTTP and has no knowledge of higher-level protocols, TLS servers can only strictly present one certificate for a particular IP/port combination.[[32]](#cite_note-32) This means that, in most cases, it is not feasible to use [name-based virtual hosting](/wiki/Virtual_hosting#Name-based) with HTTPS. A solution called [Server Name Indication](/wiki/Server_Name_Indication) (SNI) exists, which sends the hostname to the server before encrypting the connection, although many older browsers do not support this extension. Support for SNI is available since [Firefox](/wiki/Firefox) 2, [Opera](/wiki/Opera_(web_browser)) 8, Safari 2.1, Google Chrome 6, and [Internet Explorer 7](/wiki/Internet_Explorer_7) on [Windows Vista](/wiki/Windows_Vista).[[33]](#cite_note-33)[[34]](#cite_note-34)[[35]](#cite_note-35) From an architectural point of view:

1. An SSL/TLS connection is managed by the first front machine that initiates the TLS connection. If, for any reasons (routing, traffic optimization, etc.), this front machine is not the application server and it has to decipher data, solutions have to be found to propagate user authentication information or certificate to the application server, which needs to know who is going to be connected.
2. For SSL/TLS with mutual authentication, the SSL/TLS session is managed by the first server that initiates the connection. In situations where encryption has to be propagated along chained servers, session timeOut management becomes extremely tricky to implement.
3. With mutual SSL/TLS, security is maximal, but on the client-side, there is no way to properly end the SSL/TLS connection and disconnect the user except by waiting for the server session to expire or closing all related client applications.

A sophisticated type of [man-in-the-middle attack](/wiki/Man-in-the-middle_attack) called SSL stripping was presented at the Blackhat Conference 2009. This type of attack defeats the security provided by HTTPS by changing the https: link into an http: link, taking advantage of the fact that few Internet users actually type "https" into their browser interface: they get to a secure site by clicking on a link, and thus are fooled into thinking that they are using HTTPS when in fact they are using HTTP. The attacker then communicates in clear with the client.[[36]](#cite_note-36) This prompted the development of a countermeasure in HTTP called [HTTP Strict Transport Security](/wiki/HTTP_Strict_Transport_Security).

HTTPS has been shown vulnerable to a range of [traffic analysis](/wiki/Traffic_analysis) attacks. Traffic analysis attacks are a type of [side-channel attack](/wiki/Side-channel_attack) that relies on variations in the timing and size of traffic in order to infer properties about the encrypted traffic itself. Traffic analysis is possible because SSL/TLS encryption changes the contents of traffic, but has minimal impact on the size and timing of traffic. In May 2010, a research paper by researchers from [Microsoft Research](/wiki/Microsoft_Research) and [Indiana University](/wiki/Indiana_University_Bloomington) discovered that detailed sensitive user data can be inferred from side channels such as packet sizes. More specifically, the researchers found that an eavesdropper can infer the illnesses/medications/surgeries of the user, his/her family income and investment secrets, despite HTTPS protection in several high-profile, top-of-the-line web applications in healthcare, taxation, investment and web search.[[37]](#cite_note-37) Although this work demonstrated vulnerability of HTTPS to traffic analysis, the approach presented by the authors required manual analysis and focused specifically on web applications protected by HTTPS.

In June 2014, a team of researchers at [UC Berkeley](/wiki/University_of_California,_Berkeley) and [Intel](/wiki/Intel) lead by [Brad Miller](http://bradmiller.io) demonstrated a generalized approach to HTTPS traffic analysis based on [machine learning](/wiki/Machine_learning).[[38]](#cite_note-38)[[39]](#cite_note-39) The researchers demonstrated that the attack applied to a range of websites, including [Mayo Clinic](/wiki/Mayo_Clinic), [Planned Parenthood](/wiki/Planned_Parenthood) and [Youtube](/wiki/Youtube).[[40]](#cite_note-40) The attack assumes that the attacker is able to visit the same webpages as the victim to gather network traffic which serves as training data. The attacker is then able to identify similarities in the packet sizes and orderings between the victim traffic and the training data traffic which frequently allow the attacker to infer the exact page the victim is visiting. For example, this attack could be used to determine whether a user browsing the Planned Parenthood website is looking for information about preventative health screening or an abortion. Note that the attack can not be used to discover user specific values embedded in a webpage. For example, many banks offer web interfaces which allow users to view account balances. While the attacker would be able to discover that the user was viewing an account balance page, he would be unable to learn the users exact account balance or account number.

## History[[edit](/index.php?title=(none)&action=edit&section=13)]

[Netscape Communications](/wiki/Netscape_Communications) created HTTPS in 1994 for its [Netscape Navigator](/wiki/Netscape_Navigator) web browser.[[41]](#cite_note-41) Originally, HTTPS was used with the [SSL](/wiki/Secure_Sockets_Layer) protocol. As SSL evolved into [Transport Layer Security](/wiki/Transport_Layer_Security) (TLS), the current version of HTTPS was formally specified by RFC 2818 in May 2000.

## See also[[edit](/index.php?title=(none)&action=edit&section=14)]

* [Bullrun (decryption program)](/wiki/Bullrun_(decryption_program)) — a secret anti-encryption program run by the U.S. [National Security Agency](/wiki/National_Security_Agency)
* [Computer security](/wiki/Computer_security)
* [curl-loader](/wiki/Curl-loader)
* [Diameter protocol](/wiki/Diameter_protocol)
* [HTTPsec](/wiki/HTTPsec)
* [Moxie Marlinspike](/wiki/Moxie_Marlinspike)
* [Opportunistic encryption](/wiki/Opportunistic_encryption)
* [Stunnel](/wiki/Stunnel)

## References[[edit](/index.php?title=(none)&action=edit&section=15)]

[Template:Reflist](/wiki/Template:Reflist)

## External links[[edit](/index.php?title=(none)&action=edit&section=16)]

[Template:Commons category](/wiki/Template:Commons_category)

* [RFC 2818: HTTP Over TLS](https://tools.ietf.org/html/rfc2818)
* [Hacking HTTPS by man-in-the-middle attack](http://greyhatsspeak.blogspot.in/2013/10/mitm-against-https-sites.html)
* [RFC 5246: The Transport Layer Security Protocol 1.2](https://tools.ietf.org/html/rfc5246)
* [SSL 3.0 Specification](https://tools.ietf.org/html/draft-ietf-tls-ssl-version3-00) (IETF)
* [HTTPS Error: Your connection is not private](https://usefulpcguide.com/16666/your-connection-is-not-private/)

[Template:URI scheme](/wiki/Template:URI_scheme) [Template:Web browsers](/wiki/Template:Web_browsers) [Template:SSL/TLS](/wiki/Template:SSL/TLS)

[Template:DEFAULTSORT:Https](/wiki/Template:DEFAULTSORT:Https) [Category:Hypertext Transfer Protocol](/wiki/Category:Hypertext_Transfer_Protocol) [Category:Cryptographic protocols](/wiki/Category:Cryptographic_protocols) [Category:Secure communication](/wiki/Category:Secure_communication) [Category:URI schemes](/wiki/Category:URI_schemes) [Category:Transport Layer Security](/wiki/Category:Transport_Layer_Security)