***HTTPS PROTOCOL-***

HTTP Secure (HTTPS) is an adaptation of the Hypertext Transfer Protocol (HTTP) for secure communication over a computer network, and is widely used on the Internet. In HTTPS, the communication protocol is encrypted by Transport Layer Security (TLS), or formerly, its predecessor, Secure Sockets Layer (SSL). The protocol is therefore also often referred to as HTTP over TLS, or HTTP over SSL.

The principal motivation for HTTPS is authentication of the accessed website and protection of the privacy and integrity of the exchanged data. It protects against man-in-the-middle attacks. The bidirectional encryption of communications between a client and server protects against eavesdropping and tampering of the communication.[5] In practice, this provides a reasonable assurance that one is communicating without interference by attackers with the website that one intended to communicate with, as opposed to an impostor.

Historically, HTTPS connections were primarily used for payment transactions on the World Wide Web, e-mail and for sensitive transactions in corporate information systems.[citation needed] In the late 2000s and early 2010s, HTTPS was increasingly used for protecting page authenticity on all types of websites, securing accounts and keeping user communications, identity and web browsing private.

Difference from HTTP

HTTPS URLs begin with "https://" and use port 443 by default, whereas HTTP URLs begin with "http://" and use port 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 or advertisements. HTTPS is designed to withstand such attacks and is considered secure against them (with the exception of older, deprecated versions of SSL).

HTTPS (HTTP over SSL or HTTP Secure)

HTTPS (HTTP over SSL or HTTP Secure) is the use of Secure Socket Layer (SSL) or Transport Layer Security (TLS) as a sublayer under regular HTTP application layering. HTTPS encrypts and decrypts user page requests as well as the pages that are returned by the Web server. The use of HTTPS protects against eavesdropping and man-in-the-middle attacks. HTTPS was developed by Netscape.

HTTPS and SSL support the use of X.509 digital certificates from the server so that, if necessary, a user can authenticate the sender. Unless a different port is specified, HTTPS uses port 443 instead of HTTP port 80 in its interactions with the lower layer, TCP/IP.

Suppose you visit a Web site to view their online catalog. When you're ready to order, you will be given a Web page order form with a Uniform Resource Locator (URL) that starts with https://. When you click "Send," to send the page back to the catalog retailer, your browser's HTTPS layer will encrypt it. The acknowledgement you receive from the server will also travel in encrypted form, arrive with an https:// URL, and be decrypted for you by your browser's HTTPS sublayer.

***HASHING ALGORITHM – SHA 1 AND 2-***

As I said earlier, SHA stands for Secure Hashing Algorithm. SHA-1 and SHA-2 are two different versions of that algorithm. They differ in both construction (how the resulting hash is created from the original data) and in the bit-length of the signature. You should think of SHA-2 as the successor to SHA-1, as it is an overall improvement.

Primarily, people focus on the bit-length as the important distinction. SHA-1 is a 160-bit hash. SHA-2 is actually a “family” of hashes and comes in a variety of lengths, the most popular being 256-bit.

The variety of SHA-2 hashes can lead to a bit of confusion, as websites and authors express them differently. If you see “SHA-2,” “SHA-256” or “SHA-256 bit,” those names are referring to the same thing. If you see “SHA-224,” “SHA-384,” or “SHA-512,” those are referring to the alternate bit-lengths of SHA-2. You may also see some sites being more explicit and writing out both the algorithm and bit-length, such as “SHA-2 384.”

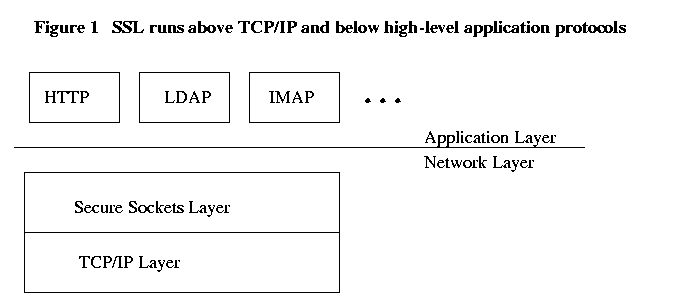
The SSL industry has picked SHA as their hashing algorithm for digital signatures. From 2011 to 2015, SHA-1 was the primary algorithm. A [growing body of research showing the weaknesses of SHA-1](https://en.wikipedia.org/wiki/SHA-1" \l "Attacks)prompted a revaluation. From 2016 onward, SHA-2 is the new standard. If you are receiving a certificate today it **must** be using that signature at a minimum.

Occasionally you will see certificates using SHA-2 384-bit. You will rarely see the 224-bit variety, which is not approved for use with publicly trusted certificates, or the 512-bit variety which is less widely supported by software.

SHA-2 will likely remain in use for at least five years. However, some unexpected attack against the algorithm could be discovered which would prompt an earlier transition.

***SSL(SECURE SOCKET LAYER)-***

* A protocol developed by Netscape.
* It is a whole new layer of protocol which operates above the Internet TCP protocol and below high-level application protocols.
* SSL uses TCP/IP on behalf of the higher-level protocols.
* Allows an SSL-enabled server to authenticate itself to an SSL-enabled client;
* Allows the client to authenticate itself to the server;
* Allows both machines to establish an encrypted connection.
* SSL server authentication.
* SSL client authentication.



***TSL(TRANSPORT SECURE LAYER)-***

Transport Layer Security (TLS) is a protocol that provides privacy and [data integrity](http://searchdatacenter.techtarget.com/definition/integrity)between two communicating applications. It's the most widely deployed security [protocol](http://searchnetworking.techtarget.com/definition/protocol) used today, and is used for Web browsers and other applications that require data to be securely exchanged over a network, such as [file transfers](http://searchnetworking.techtarget.com/definition/file-transfer), [VPN](http://searchenterprisewan.techtarget.com/definition/virtual-private-network) connections, [instant messaging](http://searchunifiedcommunications.techtarget.com/definition/instant-messaging) and [voice over IP](http://searchunifiedcommunications.techtarget.com/definition/VoIP).

According to the protocol specification, TLS is composed of two layers: the TLS Record Protocol and the TLS Handshake Protocol. The Record Protocol provides connection security, while the Handshake Protocol allows the [server](http://whatis.techtarget.com/definition/server)and [client](http://searchenterprisedesktop.techtarget.com/definition/client) to authenticate each other and to negotiate [encryption](http://searchsecurity.techtarget.com/definition/encryption) [algorithms](http://whatis.techtarget.com/definition/algorithm) and [cryptographic](http://searchsoftwarequality.techtarget.com/definition/cryptography) keys before any data is exchanged.

***IP SECURITY-***

Internet Protocol Security (IPSec) is a framework of open standards for ensuring private, secure communications over Internet Protocol (IP) networks, through the use of cryptographic security services. IPSec supports network-level peer authentication, data origin authentication, data integrity, data confidentiality (encryption), and replay protection. The Microsoft implementation of IPSec is based on standards developed by the Internet Engineering Task Force (IETF) IPSec working group.

IPSec is supported by the Windows Server 2003, Windows XP, and Windows 2000 operating systems and is integrated with the Active Directory service. IPSec policies can be assigned through Group Policy configuration of Active Directory domains and organizational units. This allows the IPSec policy to be assigned at the domain, site, or organizational unit level, simplifying IPSec deployment.

You can also assign IPSec policies using the IP Security Policies MMC snap-in on the local computer. You must be an administrator on every computer on which you want to assign IPSec policies.

***RSA ALGORITM-***

**RSA** (**Rivest–Shamir–Adleman**) is one of the first practical [public-key cryptosystems](https://en.wikipedia.org/wiki/Public-key_cryptography) and is widely used for secure data transmission. In such a [cryptosystem](https://en.wikipedia.org/wiki/Cryptosystem), the [encryption key](https://en.wikipedia.org/wiki/Encryption_key) is public and it is different from the [decryption key](https://en.wikipedia.org/wiki/Decryption_key) which is kept secret (private). In RSA, this asymmetry is based on the practical difficulty of the [factorization](https://en.wikipedia.org/wiki/Factorization) of the product of two large [prime numbers](https://en.wikipedia.org/wiki/Prime_number), the "[factoring problem](https://en.wikipedia.org/wiki/Factoring_problem)". The [acronym](https://en.wikipedia.org/wiki/Acronym) RSA is made of the initial letters of the surnames of [Ron Rivest](https://en.wikipedia.org/wiki/Ron_Rivest), [Adi Shamir](https://en.wikipedia.org/wiki/Adi_Shamir), and [Leonard Adleman](https://en.wikipedia.org/wiki/Leonard_Adleman), who first publicly described the algorithm in 1978. [Clifford Cocks](https://en.wikipedia.org/wiki/Clifford_Cocks), an English mathematician working for the British intelligence agency· [Government Communications Headquarters](https://en.wikipedia.org/wiki/Government_Communications_Headquarters) (GCHQ), had developed an equivalent system in 1973, but this was not [declassified](https://en.wikipedia.org/wiki/Classified_information) until 1997.[[1]](https://en.wikipedia.org/wiki/RSA_(cryptosystem)" \l "cite_note-1)

A user of RSA creates and then publishes a public key based on two large [prime numbers](https://en.wikipedia.org/wiki/Prime_number), along with an auxiliary value. The prime numbers must be kept secret. Anyone can use the public key to encrypt a message, but with currently published methods, and if the public key is large enough, only someone with knowledge of the prime numbers can decode the message feasibly.[[2]](https://en.wikipedia.org/wiki/RSA_(cryptosystem)" \l "cite_note-rsa-2) Breaking RSA [encryption](https://en.wikipedia.org/wiki/Encryption) is known as the [RSA problem](https://en.wikipedia.org/wiki/RSA_problem). Whether it is as difficult as the factoring problem remains an open question.

RSA is a relatively slow algorithm, and because of this, it is less commonly used to directly encrypt user data. More often, RSA passes encrypted shared keys for [symmetric key](https://en.wikipedia.org/wiki/Symmetric-key_algorithm) cryptography which in turn can perform bulk encryption-decryption operations at much higher speed.

***DIGITAL SIGNATURE-***

is called the "digital signatureis a type of **asymmetric cryptography** used to simulate the security properties of a **signature** in digital, rather than written, form. Digital signature schemes normally give two algorithms, one for signing which involves the user's secret or **private key**, and one for verifying signatures which involves the user's **public key**. The output of the signature process.

***How it works-***

* The use of digital signatures usually involves two processes, one performed by the signer and the other by the receiver of the digital signature:
* **Digital signature creation** uses a hash result derived from and unique to both the signed message and a given private key. For the hash result to be secure, there must be only a negligible possibility that the same digital signature could be created by the combination of any other message or private key.
* **Digital signature verification** is the process of checking the digital signature by reference to the original message and a given public key, thereby determining whether the digital signature was created for that same message using the private key that corresponds to the referenced public key.

**Benefits of digital signatures-**

* **Authentication**

Although messages may often include information about the entity sending a message, that information may not be accurate. Digital signatures can be used to authenticate the source of messages. When ownership of a digital signature secret key is bound to a specific user, a valid signature shows that the message was sent by that user. The importance of high confidence in sender authenticity is especially obvious in a financial context. For example, suppose a bank's branch office sends instructions to the central office requesting a change in the balance of an account. If the central office is not convinced that such a message is truly sent from an authorized source, acting on such a request could be a grave mistake.

* **Integrity**

In many scenarios, the sender and receiver of a message may have a need for confidence that the message has not been altered during transmission. Although encryption hides the contents of a message, it may be possible to *change* an encrypted message without understanding it. (Some encryption algorithms, known as nonmalleable ones, prevent this, but others do not.) However, if a message is digitally signed, any change in the message will invalidate the signature. Furthermore, there is no efficient way to modify a message and its signature to produce a new message with a valid signature, because this is still considered to be computationally infeasible by most cryptographic hash functions.

Despite their usefulness, digital signatures do not alone solve all the problems we might wish them to.

**Non-repudiation**

In a cryptographic context, the word *repudiation* refers to the act of disclaiming responsibility for a message. A message's recipient may insist the sender attach a signature in order to make later repudiation more difficult, since the recipient can show the signed message to a third party (eg, a court) to reinforce a claim as to its signatories and integrity. However, loss of control over a user's private key will mean that all digital signatures using that key, and so ostensibly 'from' that user, are suspect. Nonetheless, a user cannot repudiate a signed message without repudiating their signature key.