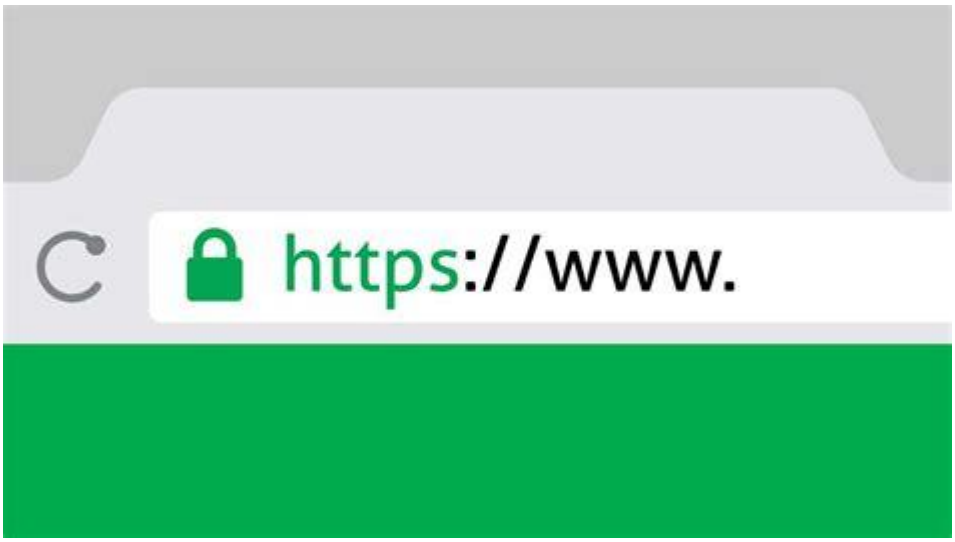
How does SSL/TLS work??



This blog is an attempt at explaining the concepts behind SSL (Secure Socket Layer) and TLS(Transport Socket Layer) and some other topics related to them.

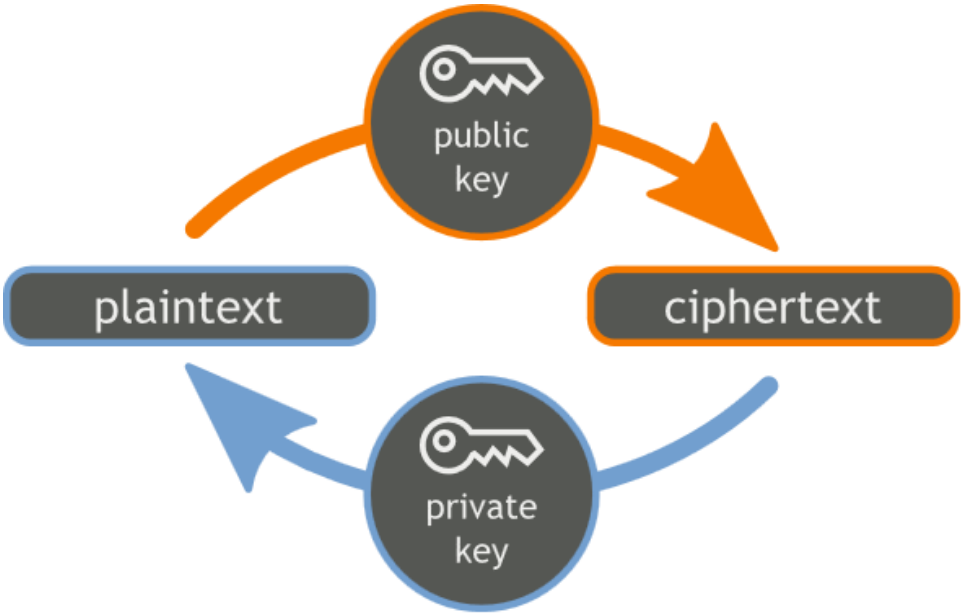
One common misunderstanding is that SSL and TLS are synonymous. This is certainly not the case.TLS 1.0 was released after SSL 3.0 to fix the security issues in the SSL 3.0. SSL 1.0 was developed but as it had significant faults, was never released. SSL 2.0 and 3.0 were successfully developed and deployed around the world. The current latest version in the TLS/SSL family is TLS 1.3

SSL and TLS both provide end-to-end encryption when implented for an application.SSL layer is implemented between the application layer and the TCP/IP protocol layer.

Before going straight to the SSL working , one must understand how public key cryptography works.

Public Key Cryptography

The main feature in this cryptography is an entity having both public and private keys. These two keys are defined mathematically and are related to each other. The main thing to remember is that a data encrypted with a person's private key can be decrypted by that person's public key only and vice versa.



To make it simple , let's take an example.
Suppose , I want to send my friend Tanmay a message without my mother not knowing what the message is.
For this ,
1)I encrypt the plaintext of my message by my private key first.
2) Then I encrypt this text by Tanmay's public key which is known throughout the network.
3) When Tanmay receives this doubly encrypted message , he will first decrypt using his own private key and then my (Shounak) public key which is known throughout the network.

Sounds secure right? Because now my mom has no idea of what my message is .
Well there is a loophole my mom found out.
Now she can read every single message of our conversation.
Turns out she published a fake public keys of me and Tanmay . Let's say fakeTanmay and fakeShounak.
So now , whenever I want to write to Tanmay , thinking that the fakeTanmay is Tanmay's public key, I am directly writing to my mother and she will get to read my message and perhaps change and give it Tanmay.
So both integrity and confidentiality of my message were broken.
Hmmmm,what can we do now?

Here comes the concept of **Certificate Authorities** (CA)

A CA issues digital certificates that identify a particular person and the public key used by that person. A digital certificate is the name (usually a domain name) and the associated public key

encrypted by the CA's private key. You can check the validity of a certificate by decrypting it with the CA's public key.
Wait, aren't we facing the same issue still? My mother can still replace the public key of CA with her own public key!! She still can read my messages. Luckily the public key of the CA is provided as a digital certificate with the OS or the browser.

Enlisted are some leading and trusted CA's that dominate the market right now.

Company	Price / Year	Unique Selling Points
Thawte	149$ US	Dominates 40% of the global SSL market
Symantec	349$ US	90% of the Fortune 500 company have contracts with Symantec
RapidSSL	17.95$ US	Cheapest on the market and can secure multiple servers without any additional cost.
GlobalSign	349$ US	Supports SHA-256 , ECC. Microsoft, Netflix are major customers of GlobalSign
DigiCert	198$ US	Provides security for the IoT devices and allows reissues of unlimited servers for lifetime.
GeoTrust	299$ US	Second most popular SSL certificate provider. Operates in 150 countries
IdenTrust	99$ US	Used by the US Government and US banks and a lot of e-commerce sites in The United States.

There are various formats of certificates in practice today. These formats differ in their encoding and the information they store. Different servers might require certificates in different formats and thus there is a way to convert one into another.

1) **DER Format**: stands for Distinguished Encoding Rules, a binary encoding format, which is mostly used in Windows OS. It is contained in .der or .cer files.
2) **PEM Format :**stands for Privacy Enhanced Email. PEM files are Base64(ASCII) encoded DER files . Being in ASCII, we can read such files in any text editor . Majority of CAs offer SSL Certificates in PEM format with different file extensions such as .pem, .crt, .cer, or .key.
   A single .pem file contains the server certificate, the intermediate certificate and the private key. Alternatively, the server and intermediate certificates come in a separate .crt or .cer file, while the private key is in a .key file.
3) **PKCS#7 Format:** stands for Public Key Cryptography Standards. It is a multi-purpose format for the distribution of encrypted data. It's mostly used on Windows platforms and Java Tomcat. The file extensions are .p7b or .p7c. Unlike PEM, PKCS#7 cannot store private keys , only the primary and intermediate certificates.
4) **PKCS#12 Format:** It includes the entire certificate chain and a key pair in a file extension like .pfx. This kind of file is password protected and can be opened only after successful password input.

How do we convert from one file extension to another ??

We can use the free OpenSSL software library to convert the SSL files. Most of the operating systems come with this library pre-installed. Run the following commands on your terminal to convert.

Convert X.509 to PEM
openssl x509 -in certificatename.cer -outform PEM -out certificatename.pem

Convert DER to PEM (Binary encoding to Base64 ASCII)
openssl x509 -inform der -in certificatename.der -out certificatename.pem

Convert PEM to DER (Base65 ASCII to binary encoding)
openssl x509 -inform der -in certificatename.der -out certificatename.pem

Convert PEM to PKCS#7 (the .p7b file does not include the private key)
openssl crl2pkcs7 -nocrl -certfile certificatename.pem -out certificatename.p7b -certfile CACert.cer

Convert PKCS#7 to PEM
openssl pkcs7 -print_certs -in certificatename.p7b -out certificatename.pem

Convert PKCS#12 to PEM (PKCS#12 file is password-protected)
openssl pkcs12 -in certificatename.pfx -out certificatename.pem

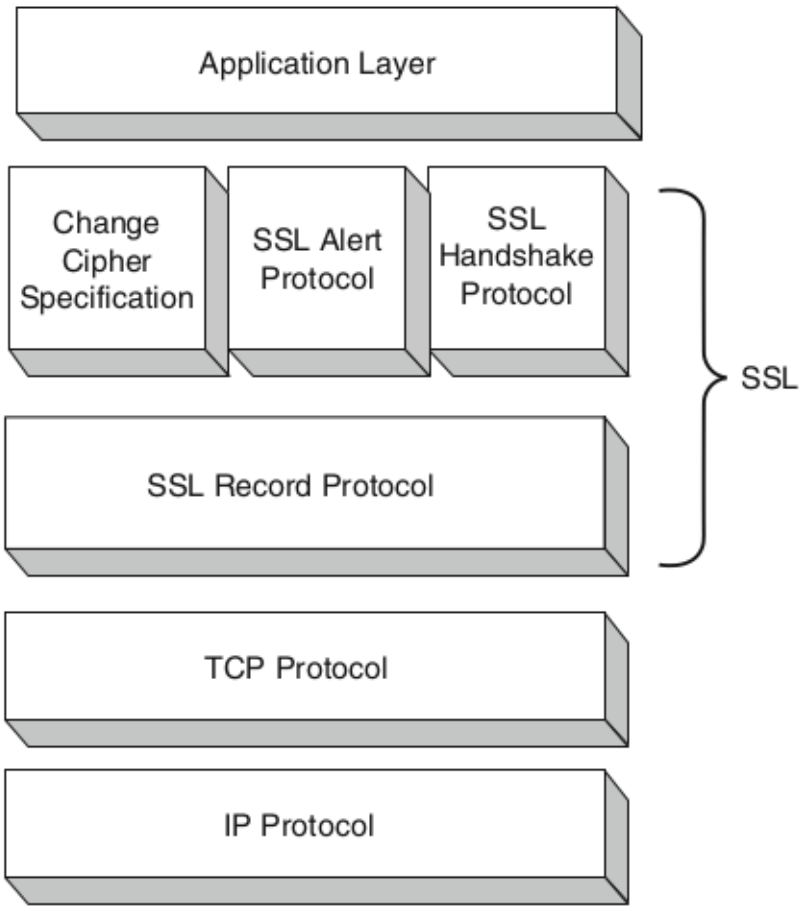
Convert PKCS7 to PKCS12
This requires two steps. Convert the P7B file to CER and then combine CER and Private Key into PFX.

1. *openssl pkcs7 -print_certs -in certificatename.p7b -out certificatename.cer*

2. *openssl pkcs12 -export -in certificatename.cer -inkey privateKey.key -out certificatename.pfx -certfile cacert.cer*

SSL objectives

- **Data encryption—**this ensures that no one can read the message when its travelling between the two communicating entities.
- **Server and client authentication—**this makes sure that the client is actually says who he is and the server really is who its says he is.
- **Message integrity**—this ensures no one tampered or changed the message when travelling through the medium between client and server.

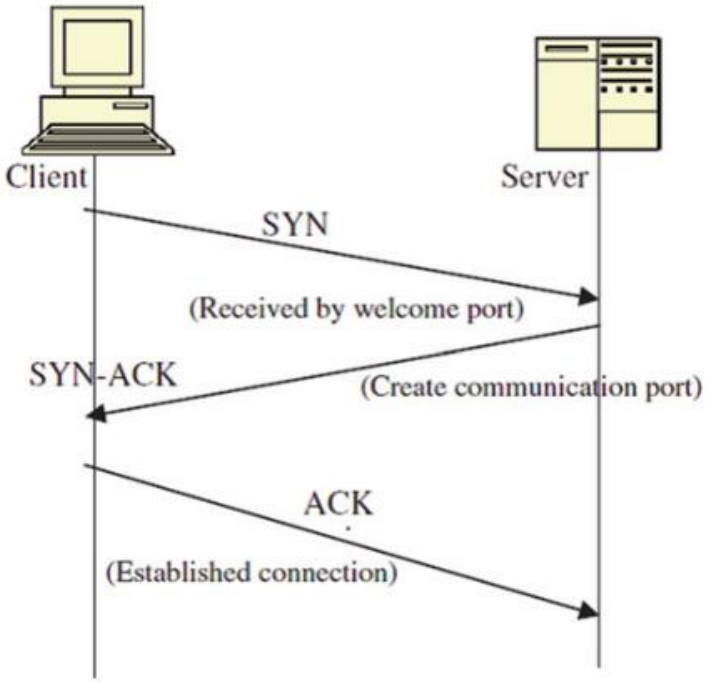


Above is the full SSL protocol stack

SSL Handshake Protocol

Before any TCP connection is established between the client and server, both running under SSL, The SSL Handshake is called to establish and authenticate connection. This is called the handshake.

1) The client sends a message to the server telling the server the available cryptographic algorithms with it. The server decides on a particular set of algorithms which both the server and client can work with. This set is called as the cypher suite.
2)The server has to mandatory send authentication by sending its certificate to the client to verify that the server's certificate was signed by a trusted CA.



3) The client can provide authentication, if required, through the client sending its own certificate to the server to verify that the client's certificate was signed by a trusted CA.
4) The client generates session key which is used in all subsequent encryption or decryption. The customer encrypts the session key using the public key of the merchant server (from the merchant's certificate). The server recovers the session key by decrypting it using its private key. This symmetric key, which now both parties have, is used in all subsequent communication.

Imp Note: The CA may not be the same CA who signed the client's certificate. CAs may come from a list of trusted CAs. The reason for making this step optional was a result of realization that since few customers are willing, know how to get digital certificates, requiring them to do this would mean exempting huge amounts of customer out of the system . This, definitely, leads to some security gaps to the system.

Symmetric key means a key which can both encrypt and decrypt the message.

The SSL Handshake protocol takes care of the authentication objective.

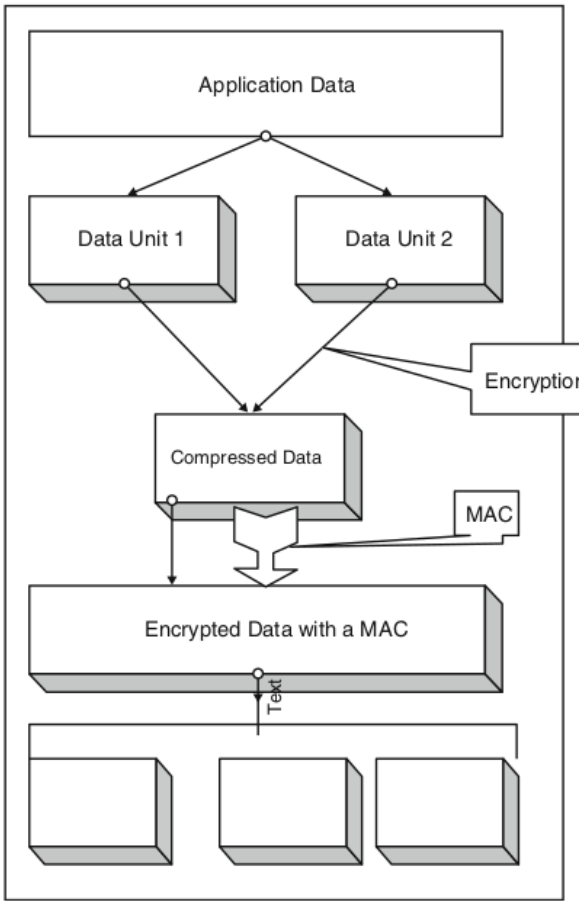
SSL Record Protocol

This protocol provides the other two objectives , Confidentiality and Message integrity

The confidentiality is maintained by using the agreed symmetric key in the handshake protocol.

1) SSL Record Protocol takes a message to be transmitted and fragments the data, compresses the fragments, adds a MAC(**Message authentication code**)**,** encrypts the block with both data and MAC by using the symmetric key, adds a SSL header, and transmits it under the TCP protocol.

SSL Record Protocol



2)The received data is decrypted, verified, and decompressed . The SSL header that is added to each data portion contains two key pieces of information, namely, the length of the record and the length of the data block added to the original data.
3)MAC is a hash function which looks like something Hash=fn(key, data, padding, sequence number). After decrypting the received message, The MAC is again calculated by the receiver and compared with the received MAC. If both the MAC's are the same , we can positively conclude that the message's integrity is maintained.

SSL Cipher Spec Protocol

This consists of an exchange of a single message in a byte with a value of 1 being exchanged, using the SSL record protocol between the server and client. The bit is exchanged to establish a pending session state to be copied into the working state, thus defining a new set of protocols as the new agreed on session state.
To put in blunt words , these message must be sent by the client to the server and vice versa. After exchange of messages, the session state is considered agreed.

SSL Alert Protocol

The SSL Alert protocol, which also runs over the SSL Record protocol, is used by the two parties to convey session warning/error messages associated with data exchange.

Each message in the alert protocol consists of 2 bytes, with the first byte taking a value of "1" for a warning and "2"for a fatal error.

The second byte of the message contains one of the defined error codes that may occur during an SSL communication session with some of them being unknown certificate, revoked certificate and expired certificate.

Analysis of different protocols employed in SSL cipher suites

Before we move to TLS , we need to analyse the cryptographic protocols used in SSL as most of the cipher suites are same in both TLS and SSL.

First and foremost , we need to understand a hash function and encryption algorithm are different. You use a hash function to digest a large document but you can't recover the large document from a given hash. Thus a hash function works one way only . Hash functions are used for Message Authentication purposes to see if we got the same message as we intended to.

Encryption algorithms are reversible , we can decrypt the encrypted message and vice versa.

First we will compare the running time between the MD5 and the SHA 256 hashing algorithms

MD5 vs SHA 256

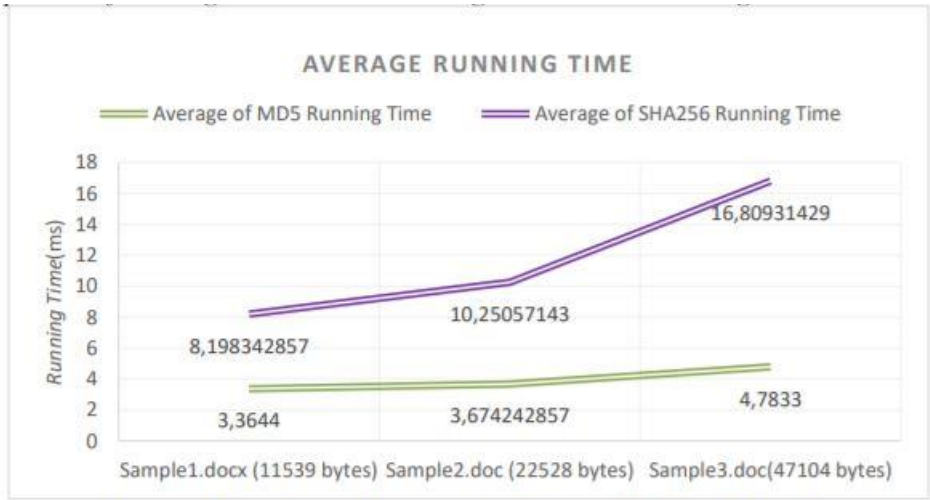


Figure 1. Graph about average running time MD5 and SHA256

We can clearly see that SHA-256 is slower when compared to MD5. But both the algorithms follow Θ(N) time complexity.

	SHA-256	MD5
Name	Secure Hash Algorithm -256	Message Direct -5
Output Size	256 bits	128bits
Collision prone	No	Yes
Attack prone	Less prone to attacks	More prone to attacks

We move on to compare encryption algorithms employed by the SSL Cipher suite.

3DES vs RC4

	3DES	RC4
Name	Triple Data Encryption Standard	Rivest Cipher 4
Technique	Ciphers blocks of 64 bits at a time and does it three times with three different keys	Ciphers unit by unit , either bit by bit or byte by byte
Output size	In multiples of cipher block defined	Variable length as the algorithm doesn't wait for a particular length to start encrypting
Ease of implementation	Harder	Easier
Strength	Tougher to hack, proved effective in industry	Easier to hack and there are ways to crack it effectively , thus is suggested not to use RC4

For a more detailed review of the two algorithms , we suggest you to read
The Simulation and Analysis of RC4 and 3DES Algorithm for Data Encryption in RFID Credit Card - Sharma, Rohit- Singh, Prabhat

Why did we shift to TLS??

From what you understand, everything seems super secure right??
Well , cyber security is much like the arms race , whenever a more advanced protocol is in place, hackers find its weaknesses eventually and sensitive information falls into the hackers hands.
So cryptographers and cyber security experts keep improving and fine tune algorithms and protocols to make it even more secure.

People kept improving on the initial SSL 2.0 deployed on computer networks. Thus SSL 3.0 came into picture . SSL 3.0 had some major flaws in its architecture so a revised SSL 3.1 was planned and was later renamed as TLS (transport layer socket). Although the name is completely different, the basic steps and concepts of the protocol remain the same.

The SSL 3.0 cipher suites had a weaker key derivation process. The key that is generated is based on the MD5 hash function, which is not resistant to collisions and thus is insecure. Under TLS 1.0, the key that is generated depends on both MD5 and SHA-1 so it is not that weak(SHA-1 doesn't allow collisions).

Some other major differences between TLS 1.0 and SSL 3.0 are :

1)Cipher Suite: FORTEZZA employed cipher suites were dropped in TLS 1.0
2)Certificate Management:
There are 2 important differences in the way the two protocols handle certificates

i) SSL 3.0 requires complete certificate chains, meaning that a certificate chain must include all certificates that are required to verify the chain. This means that all certificate must be present till the root CA. TLS 1.0 allows a intermediate trusted CA to be the endpoint in the certificate chain.
For e.g. A certificate can be verified by an intermediate CA which is trusted by the root CA(The CA which are approved by your OS or the browser). This can happen only in TLS but not in SSL 3.0.

ii) TLS 1.0 only supports certificates which are

a)RSA signed
b)DSA signed
c)DSA signed with fixed Diffie-Hellman key exchange
d)RSA signed with a fixed Diffie-Hellman key exchange

Below are the accepted Certificate type values in SSL 3.0

Value	Name	Description
1	rsa_sign	RSA signing and key exchange
2	dss_sign	DSA signing only
3	rsa_fixed_dh	RSA signing with fixed DH key exchange
4	dss_fixed_dh	DSA signing with fixed DH key exchange
5	rsa_ephemeral_dh	RSA signing with ephemeral DH key exchange
6	dss_ephemeral_dh	DSA signing with ephemeral DH key exchange
20	fortezza_kea	FORTEZZA signing and key exchange

3) Alert Messages :TLS 1.0 introduces new alert protocol messages such as

a) decryption_failed (a fatal error)
b)record_overflow :The cipherText is longer than 2^{14} +2048 bytes (a fatal error)
c)unknown_ca : At least one certificate in the certificate chain was associated with a untrusted CA (a fatal error)
d)access_denied :After receiving the valid certificate , when the client doesn't proceed with the session key , this alert is raised and is a fatal one.

Imp Note : FORTEZZA is a set of protocols designed by the National Security Agency . The main use of it is to encrypt emails.

This crypto-war between the hackers and cyber-security has led to significant changes to the original SSL 2.0 till the latest TLS 1.3. For details about each version and its specification visit https://www.ssl2buy.com/wiki/ssl-vs-tls

Why should you disable older versions of SSL and TLS ???

You should always configure your device to support the latest protocol versions to ensure you are using only the strongest algorithms and ciphers, but its super important to disable the older versions as well. Continuing to support old versions of the protocols can leave you vulnerable to downgrade attacks, where hackers force connections to your server to use older versions of the protocols that have known exploits. This can leave your encrypted connections vulnerable for attacks.
Such problems are of utmost important to servers of corporations as they handle sensitive information of thousands of their customers.

Sounds risky right?
Well , here's how you can disable the older versions of SSL/TLS. We will discuss how to do this on Apache web servers as these servers are used by the maximum websites.

1) Use vi (or vim) to edit the ssl.conf (the ssl.conf file pertaining to the Apache installation)

2. Look for the "SSL Protocol Support" section. It will probably read as follows:

```
# SSL Protocol support:# List the enable protocol levels with which clients will be able to# connect. Disable SSLv2 access by default:SSLProtocol all -SSLv2
```

3) Remove "SSLProtocol all -SSLv2" and add this line instead of it:

```
SSLProtocol all -SSLv2 -SSLv3 -TLSv1 -TLSv1.1
```

This section should now look as follows:

```
# SSL Protocol support:# List the enable protocol levels with which clients will be able to# connect. Disable SSLv2 access by default:

SSLProtocol all -SSLv2 -SSLv3 -TLSv1 -TLSv1.1
```

This setting turns off TLS 1.0/1.1 and SSL 2.0/3.0.

4) Look for the SSL Cipher Suite section. It will probably read as follows:

```
# SSL Cipher Suite:# List the ciphers that the client is permitted to negotiate.# See the mod_ssl documentation for a complete list.SSLCipherSuite HIGH:MEDIUM:!aNULL:!MD5:!SEED:!IDEA
```

5. Remove the "SSLCipherSuite HIGH:MEDIUM:!aNULL:!MD5:!SEED:!IDEA" and add this line instead of it:

```
SSLCipherSuite HIGH:!aNULL:!MD5:!3DES
```

This section should now look as follows:

```
# SSL Cipher Suite:# List the ciphers that the client is permitted to negotiate.# See the mod_ssl documentation for a complete list.SSLCipherSuite HIGH:!aNULL:!MD5:!3DES
```

This setting ensures only high-security SSL Ciphers will be used.

Save the file in the editor, then restart the Apache service for it to take effect: Use the following command to do that

```
service httpd restart
```

So, you have successfully made your Apache server secure. **Congratulations!**

The battle may be won , but the war is endless because no security protocol is a 100% secure. We need to always be on our toes and embrace the latest secure technologies to protect our sensitive information. Right now , TLS 1.3 is the global standard for providing end to end security. TLS 1.3

was released in 2018 and working well in the industry. The latest protocol coming up in the market is DTLS.

References

1) https://www.techrepublic.com/article/heres-how-to-disable-outdated-tls-and-ssl-versions-in-apache-and-why-you-should/
2)Joseph Migga Kizza -Guide to Computer Network Security(Springer publication)
3)https://w3techs.com/technologies/overview/web_server
4) https://www.znetlive.com/blog/top-10-reliable-ssl-certificate-providers/
5) https://www.ssldragon.com/blog/a-complete-guide-to-ssl-certificate-formats/
6) https://iopscience.iop.org/article/10.1088/1742-6596/978/1/012116/pdf
7) https://www.ukessays.com/essays/computer-science/comparison-of-triple-des-or-rc4.php
8) **SSL and TLS-Theory and Practice ,Second Edition-**Rolf Oppliger