HTTPS AND SECURE COMMUNICATION

Http communication is not
secure .anybody can intercept the communication and see the data that are passing by .so we need use the HTTPS protocol where the communication is done in a secure manner where the data that is sent in encrypted between the client ans the server side

there are lot of way of encryption

to understand this two you need to understand the two different types of encryption

1) Symmetric key Cryptography
2) public private key Cryptography

Symmetric Key Cryptography

[pic]

the main properties of the Symmetric key encryption is that the client and the server share a common key that used to encrypt and the messages and send it and then server takes the encrypted messages and then decrypt the messages with the exactly same key and then original messages is going to be extracted.this gives you assurance that this communication is secured.

Asymmetric Key Encryption:
[public and private key Encryption]

→ it uses two different key to encrypt and decrypt the messages

→ public key that can be widely distributed

→ private key that will only known by the receiver

when somebody wants to send a message to you they will use your public key that you gave it to them .then they will encrypt the message with with the public key and send the encrypted message to you and the recever will decrypt the message with the private key and the extract the original message.public key can be widely distributed without any concern but not the private key.remember they use two different key.

[pic]

now there is a question ?

If this is so good then why we dont use public and private key for everything? Well it is expensive you need to give everyone your public key that you want to communiate.and for each host you need a different key pair.thats not a easy task.thats why even this public key encryption is very good way of encryption it does not use in all the communication

COMMON SECRET KEY

is another method that is widely used .the most important properties is that it uses both the symmetric and Asymmetric encryption process to established a secure connection

when we talking about the SSL (secure socket layer) AND the TLS (Transport

layer security) protocol that use the common secret key.

This protocol use the encrypted messages and they make it using the combination to the public key encryption and the Symmetric key encryption

[how TLS] work.

This process is done by a handshaking process.

This process is going to be with this process

1) the client send a server a request that “i want to make a secure communication with you”

2)then server send then a certificate of the server that contains the public key.[the certificate is authorized by some international company]

3) then the client use this public key to make another secret key called the “premaster secret” which is encrypted by the public key that is given by the server.

4) now the client send the “pre master secret” to the server.

5) now the both server and client has the pre master key

6) using this pre master key they will make a same pair of another key called “session key”. So now the client and server has a same key called session key. [so now the communication is Symmetric]

7) now the client and server communicate with the messgae that is encrypted with the session key.

This is the TLS protocol use

now in HTTP communication there are four layer

http → tcp → ip → Underlying network

in Https we add another layer after the http

http → SSL/TLS → tcp → ip → Underlying network

now how to make a certificate.for a public server there needs to make certificate with the help of a international organization like ‘verysign’.they will ensure the server and give them a certificate.

But if you want to make a development environment with https you dont need to do that .you can use self signed certificate to your development environment to test your application. Remember this is only for development and testing purpose not for production purpose.to make a self signed certificate we use a program called OPENSSL .

To make a public key cetificate in open ssl

=>openssl genrsa 1024 > private.key
=>openssl req -new -key -out cert.csr
=>openssl x509 -req -in cert.csr -signkey private.key -out certificate.pem