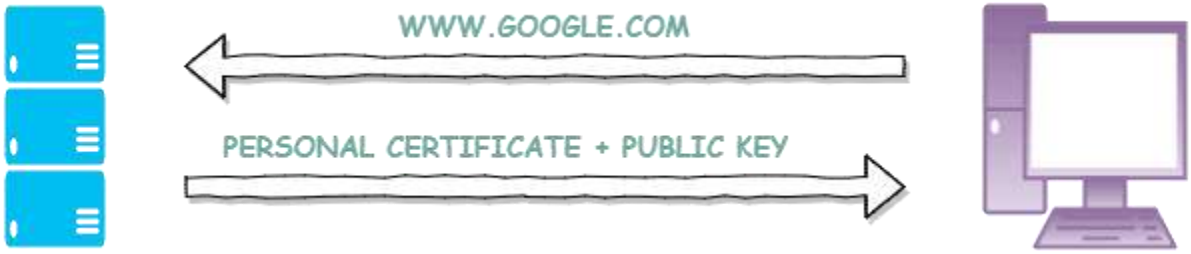
HTTPS security can be split into 2 parts (Handshakes):

1. To validate the certificate of a website:



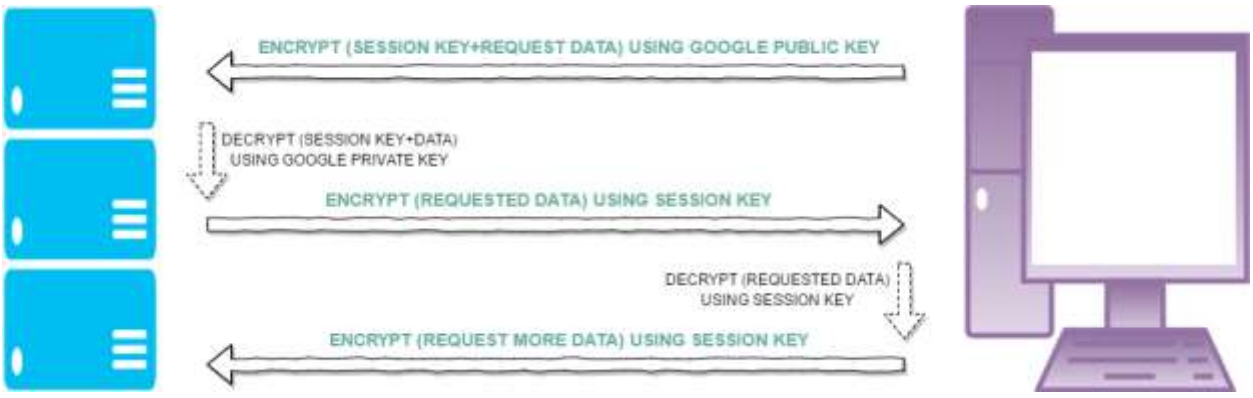
1) When you enter the URL www.Google.com, Google's server gives its public key and certificate (which was signed by GeoTrust) to the Browser.

2) Now browser has to verify the authenticity of the certificate i.e. it's actually signed from GeoTrust or not. As browsers come with a pre-installed list of public keys from all the major CA's, it picks the public key of the GeoTrust and tries to decrypt the digital signature of the certificate which was encrypted by the private key of GeoTrust.

3) If it's able to decrypt the signature (which means it's a trustworthy website) then it proceeds to the next step else it stops and shows a red cross before the URL.

2. To create a secure connection (encrypts outgoing and incoming data) :

So that no one else can read it:



1) As I mentioned, Google sends its public key when you enter www.Google.com . Any data encrypted with this public key can only be decrypted by Google's private key which Google doesn't share with anyone.

2) After validating the certificate, browser creates a new key let's call it Session Key and make 2 copies of it. These keys can encrypt as well as decrypt the data.

3) The browser then encrypts (1 copy of session key + other request data) with the Google's public key . Then it sends it back to the Google server.

4) Google's server decrypts the encrypted data using its private key and gets the session key , and other request data.

Note : Now, see, server and browser both have got the same copies of session key of the browser. No one else has this key, therefore, only server and browser can encrypt and decrypt the data. This key will now be used for both to decrypt and to encrypt the data.

5) When Google sends the data like requested HTML document and other HTTP data to the browser it first encrypts the data with this session key and browser decrypts the data with the other copy of the session key.

6) Similarly, when browser sends the data to the Google server it encrypts it with the session key which server decrypts on the other side.

Note: This session key is only used for that session only. If the user closes the website and opens again, a new session key would be created.