Tutorial 3

Exercise 1

* To extend the exercise 2 from tutorial 2 to be able to use digital signatures, we need to add one function for creating hash of the message and then sign it using the private key of the sender, alice.

  
The function takes in private\_key of alice and the plaintext message. It uses hashes to create a SHA256 hash of the plaintest and stores it in variables digest. Then signature is made using private\_key and signed with the digest, some padding, this time with PSS – probabilistic signature scheme.

* Similarly a function for message verification is made.  
  A computer screen shot of text

  Description automatically generated  
  This function takes in the public\_key of the sender(alice), the plaintext message and the signature. Verification works by computing the sha256 hash of the received plaintext message and then using public\_key to verify if the signature received is same as the one now created in variable digest. If both are same True is returned if not then False.

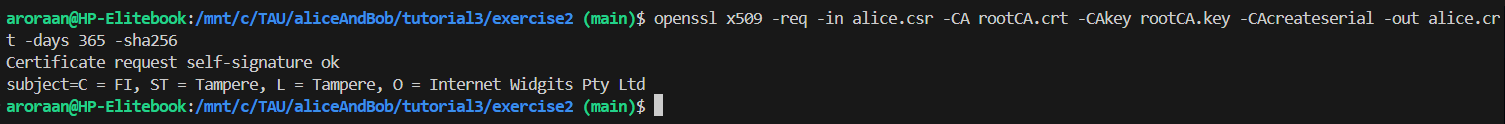
Exercise 2

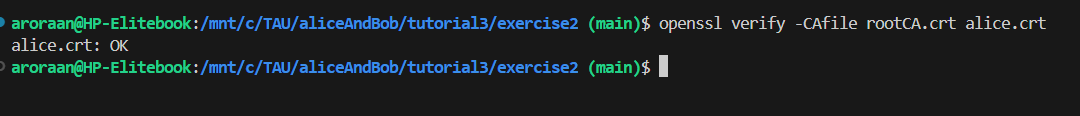
* The commands to set up a rootCA and self sign it are  
    
  A screen shot of a computer program

  Description automatically generated  
  Created rootCA.key with rsa and 2048 bits then using that key to sign a crt for rootCA.crt with validity of 3650 days.

Now create key for user, alice and certificate signing request to be sent to rootCA  
  
A screen shot of a computer

Description automatically generated  
Password is kept empty.

Sign the CSR with rootCA for 365 days validitiy  
  
  
 Verification

  
I found that if i put the same values for rootCA.crt and alice.crt in CountryName, Locality etc it assumed that alice.crt is self signed because Issuer and Subject field had the same entries.

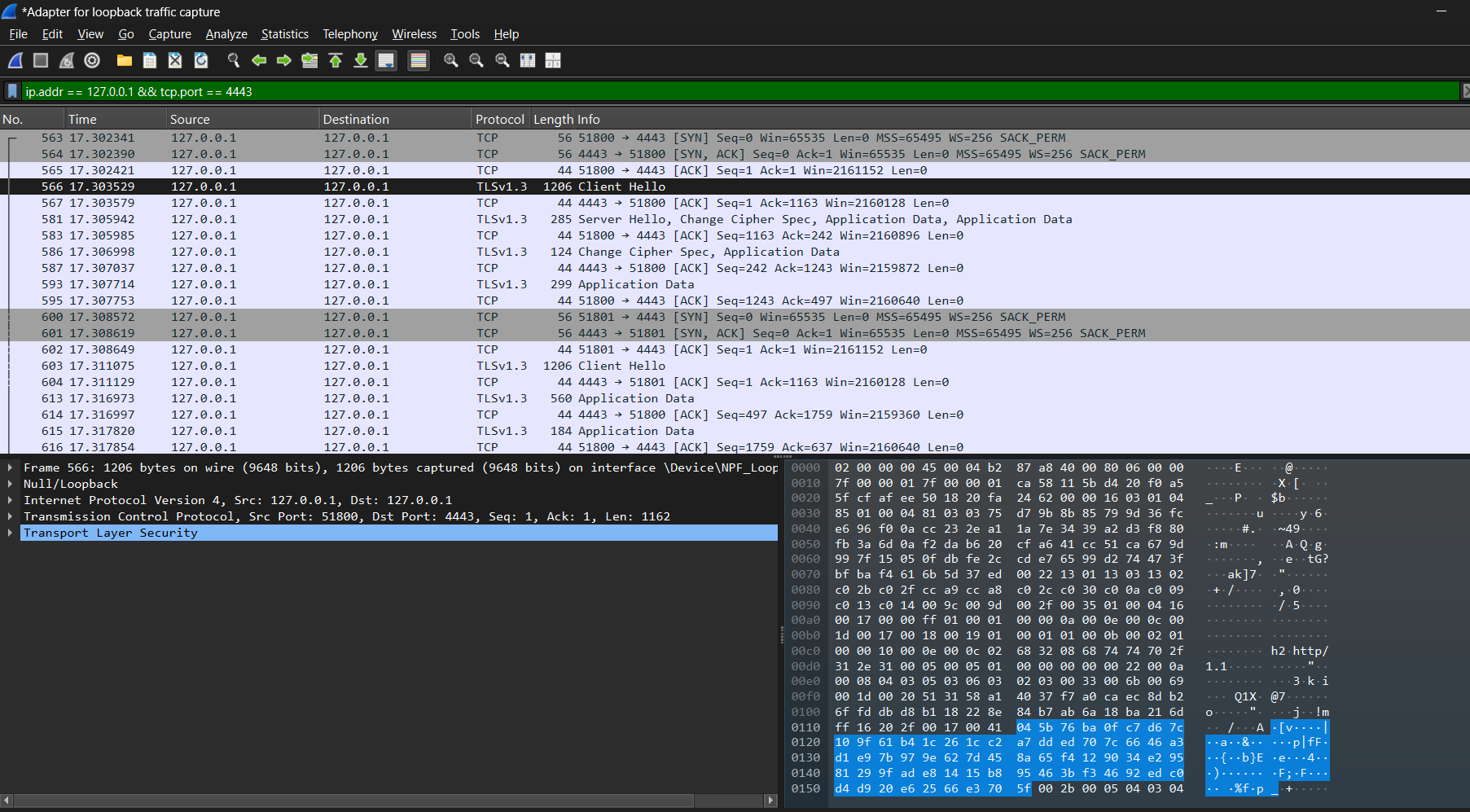
Exercise 3

* First create new server key and certificate using openssl  
    
  A screenshot of a computer program

  Description automatically generated  
    
  For domain name i used localhost or 127.0.0.1 also works.
* The python server looks like this  
  A computer screen shot of a program

  Description automatically generated  
  It just displays Hello, world! When accessed.
* To configure my computer to associate domain name to localhost I didn’t have to do anything since i kept domain name as localhost. But if it was anything else we could edit the /etc/hosts file via WSL2.
* To allow firefox to trust this certificate we need to add it in settings. That is done by going to Settings -> Search Certificate -> View Certificates -> Authorities -> Import -> server.crt  
   A screenshot of a computer

  Description automatically generated
* Then go to <https://127.0.0.1:4443>  
  A screenshot of a computer

  Description automatically generated  
    
  Firefox still says the connection is not secure because the website doesn’t provide any information about the ownership but https works just fine.
* Output in wireshark  
    
    
  In wireshark we can see the TLS 3way handshake between the client and the server. And because of TLS we also cannot see the websites content ie ‘Hello, world!’ in cleartext which would’ve been visible if it was HTTP.

Exercise 4

I tried to do exercise 4 and created self signed certificate but getting that certificate into outlook was/is a challenge. I tried numerous things like adding that certificate via mmc console to my device and settings in Outlook trust centre but outlook would always complain that there is some issue with the certificate and didn’t let me send the email.

Getting the trust centre settings in outlook was a whole other ordeal, the new outlook doesn’t allow users to set those permissions and i had to force outlook to use older version to be able to find those settings.

I think this exercise needs some updating or atleast I have no ideas how to do it.