**Assignment 3**

**Simple attacks on TLS**

Nisha M

CS19BTECH11012

# Question 1

**Ans :** In the given scenario, Alice and Bob are using TLS without any nounces for securely communicating between each other.

Alice encrypts a PMF(named P1) to Bob over TLS and the key that will be used to encrypt or decrypt the application data that would be transmitted between them will be dependent only on this P1. So the master secret (named m1) can be obtained by applying a pseudorandom function on P1.

| m1= PRF(P1) K1 = PRF(m1) = PRF(PRF(P1)) |
| --- |

There are two cases now :

1. Trudy replaying Alice’s previous messages(SUCCESSFUL) - Trudy starts replaying Alice after P1 is leaked and hence she has the messages of the whole of the session between Alice and Bob till then. Next, Trudy launches a replay attack (replaying Alice) by sending P1 to Bob (it will be approved as the certificates sent to Bob to approve the identity will be the same as of Alice) and hence the key that Bob generates will be the same as that of the previous one.

| m2=PRF(P1)  Here m2=m1  K2= PRF(m2) =PRF(m1)=K1 |
| --- |

We can infer from the above decryption that the newly generated key (K2) is the same as the old one (K1). This concludes that Trudy was successful in launching the attack with Bob, replaying Alice.

1. Trudy replaying Bob’s previous messages(UNSUCCESSFUL) - When Trudy sends Alice the certificates of Bob for validation, Alice sends back a newly generated PMF (named P2) with the help of Bob’s private key to Bob. Even if Trudy interferes in between Alice and Bob , she won’t be able to decrypt P2 and hence cannot generate the key. Even if she sends messages to Alice replaying Bob, Alice won’t be able to decrypt them and hence Trudy is unsuccessful in launching the attack with Alice replaying Bob.

| m3= PRF(P2) K3 = PRF(m3)!=K2 |
| --- |

# Question 2

**Ans :** The nonces employed in TLS help in preventing Session replay attacks. For example, let us consider the scenario where Alice encrypts a PMF (named P1) and sends it to Bob. Let the nonces used by Alice and Bob be N\_a and N\_b respectively.

| m1= PRF(P1) K1 = PRF(m1,N\_a,N\_b) |
| --- |

Suppose if Trudy launches an attack by send the above generated N\_a to Bob, Bob can cross check the integrity by the following steps:

* The nonce sent by Bob (N\_b’) will be different(i.e N\_b’ != N\_b)
* The newly generated key will also be different from the previous one

| m2=PRF(P1) = m1 K2=PRF(m1,N\_a,N\_b')  K2 != K1 |
| --- |

Due to this, Bob won’t be able to decrypt messages sent by Trudy replaying Alice and thus the attack is prevented. In the same manner, in the opposite case where Trudy replays as Bob, the nonce generated by Alice (N\_a’) won’t be same the previous one (N\_a) and thus the key generated will be different making it impossible for Alive to decrypt messages sent by Trudy (replaying Bob). Hence the attack is prevented.

# Question 3

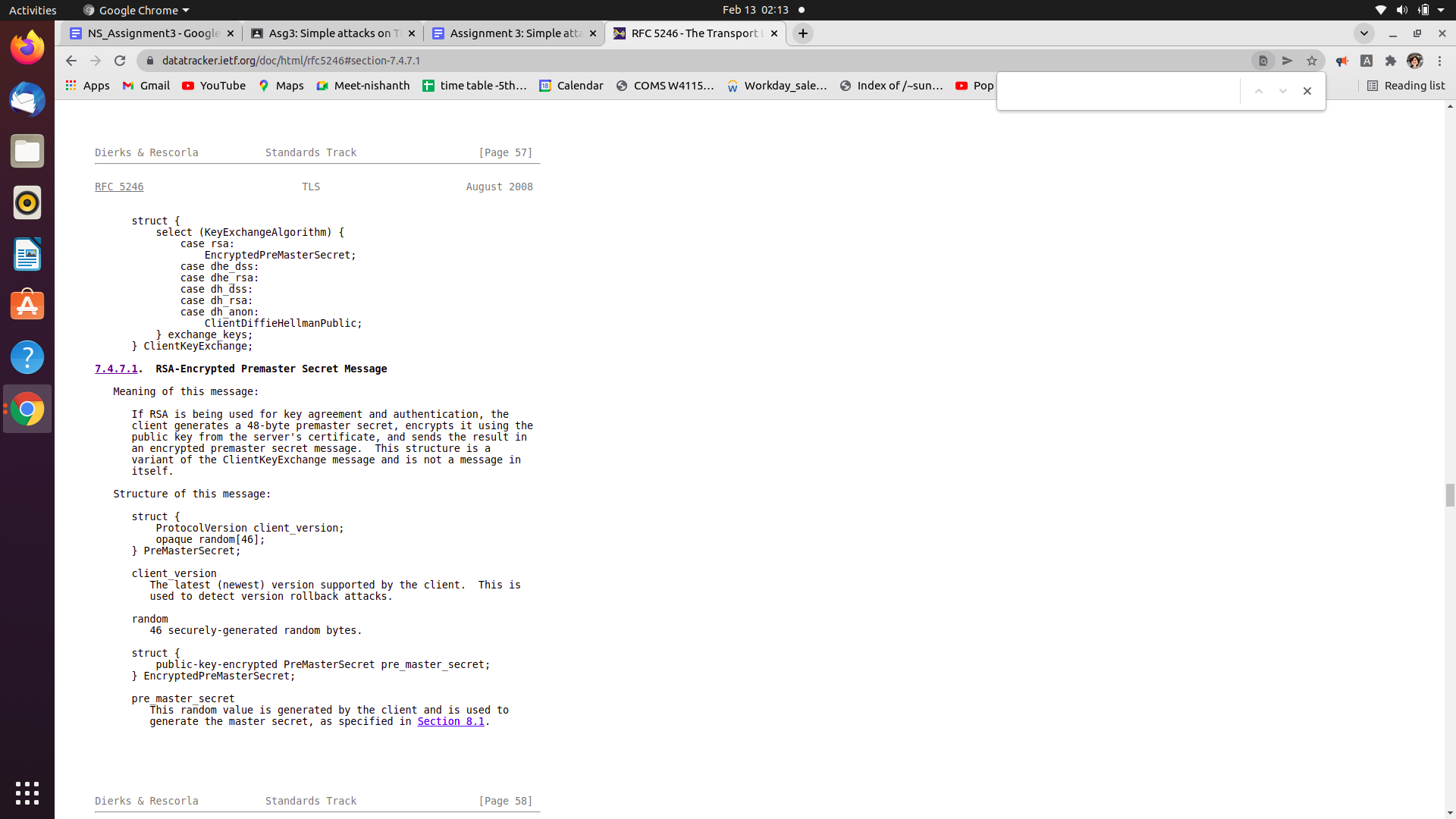
**Ans :** RSA authentication

* Client\_version - The latest (newest) version supported by the client. This is used to detect version rollback attacks.
* Random - 46 securely-generated random bytes.

struct {

public-key-encrypted PreMasterSecret pre\_master\_secret;

} EncryptedPreMasterSecret;

* Pre\_master\_secret - This random value is generated by the client and is used to generate the master secret . 

# Question 4

**Ans :** The procedures of verifying each other's certificates in the current TLS handshake is done in different phases. In the second phase, Bob sends his certificates to Alice for verification and then Alice derives the pre master secret , encrypts it with the public ke received from Bob alongwith his certificates and hence in the third phase of this handshake, Alice sends her certificates to Bob for verification.

In the other case, if Bob wants to derive PMS, it has to have the public key of Alice so that he encrypts the generated PMS and sends it to Alice. Since this step is not mandatory in the current TLS handshake protocol because of the lack of surety of Bob receiving Alice’s certificates for verification, he cannot derive the PMS.

# Question 5

**Ans :** If Bob has to derive the PMS and share it with Alice, he has to have the certificates of Alice to verify her identity and as well as her public key so that he can encrypt the message with that key and send it back to Alice. In this case there is an addition of one more phase which will delay the handshake time .

# Question 6

**Ans :** It is given that Alice is generating the MS from PMS and directly shares the MS with Bob by encrypting it with his public key in the following manner :

| MS = PRF(PMS, N\_a,N\_b) , {N\_a-nonce of alice, N\_b-nonce of Bob} Key\_matl = PRF(MS, N\_a, N\_b, key\_bs) , {key\_bs- key block size} |
| --- |

Assuming that Trudy captures the messages exchanged between Alice and Bob, we can declare that Trudy has N\_a (nonce of Alice), N\_b (nonce of Bob) , MS that is encrypted with Bob’s public key and the application data exchanged in that session as well. In this replay attack, Trudy is replaying Alice and sends the N\_a to Bob and Bob in return sends a nonce N\_b’ which is not equal to N\_b that was produced in the previous session. This is because of the involvement of nonces in manufacturing the key material for every session. So Trudy won’t be successful in launching replay attacks in this scenario.

# Question 7

**Ans :** In this scenario, Alice doesn’t just generate MS from PMS and nonces . She instead generates the key material from MS and nonces and then shares it to Bob by encrypting it with Bob’s public key. In this case, if Trudy launches a replay attack on BOB by replaying Alice, she sends the same key material to Bob encrypted with his public key and Bob will also be able to decrypt the messages sent by Trudy in that and as well as the next sessions unlike the previous question. Thus Trudy succeeds in launching session/connection replay attacks on Bob .

# Question 8

**Ans :** When Alice is sending 10 TLS records which are numbered in sequence starting from 1 to 10 carrying application data to Bob, Trudy deletes the 7th record and the integrity is lost. But if Trudy wants to fool the TCP's insequence delivery mechanism so that the TCP receiver at Bob end thinks everything is perfect , the only way would be to renumber the records by one number less after the 7th record which is being deleted . That is the next set of records which were numbered 8 ,9 and 10 which now be numbered 7,8 and 9 respectively and hence there is no flaw detected. Since the ordering of packets is right, it is assumed that there is no data loss.

# Question 9

**Ans :** Having successfully fooled the TCP receiver of Bob , Trudy will still fail in fooling the TLS receiver at his end. The MAC digest which is generated by Alice with the help of HMAC algorithm (message, MAC key, sequence number) is added to the message and then sent to Bob. So this after decrypting contains the original sequence number that was associated with the record before Trudy manipulates it . When Bob generates the MAC digest using the same algorithm mentioned above (after Trudy renumbers the records ), it won’t match with the decrypted MAC of the original message . Thus Trudy fails in fooling the TLS receiver of Bob.

# Question 10

**Ans :** The application data messages exchanged between Alice and Bob over TLS are captured by Trudy and alongwith that she hacks and gets to know the private key. As we know that Bob’s private key is the one that is being used in decrypting preparing pre master secret that was encrypted with Bob’s public key, Trudy would be able to decrypt PMS sent by Alice and can successfully generate key material (info about nonces is leaked in the application data exchanged before). Hence Trudy will be able to decrypt all the application data exchanged between them concluding that there is no PFS (perfect forward secrecy).

# Question 11

**Ans :** Unlike the previous question if we have to maintain perfect forward secrecy, We have to generate keys dynamically i.e every session being connected between Alice and Bob should have new keys generated that are specific to only that session.

The pair of keys (public and private) will be generated for a session , and will be used for encrypting the PMS generated and hence decrypting it on the other side. Once the session is complete, the private keypair will be deleted and even if Trudy gets hold of this private key, she will be able to decrypt messages only for that session as new key pairs will be generated for the next session to maintain perfect forward secrecy.

# Question 12

**Ans :** ECDH stands for elliptical curve Diffie-Hellman algorithm which is static. In this case, Alice and Bob both derive the keys independently. They agree upon two numbers and randomly generate private keys and with the help of these, they generate corresponding public keys which are exchanged between each other. In this way they end up with the shared secret value which will be the exact same for every session henceforth in their connection. If an attacker succeeds in obtaining Bob’s private key , he may recompute the shared secret and so decode all previous and future Alice-Bob communications. To provide PFS, the ECDH has to be ephemeral. In this way the above cipher suite that uses ECDH\_RSA doesn’t maintain perfect forward secrecy.

# Question 13

**Ans :** All the cipher suites that use ephemeral Diffie-Hellman(DHE) key exchange are meant to provide perfect forward secrecy. Below is the list :

| **Cipher Suite** | **Key exchange** | **Cipher** | **Mac** |
| --- | --- | --- | --- |
| TLS\_DHE\_DSS\_WITH\_3DES\_EDE\_CBC\_SHA | DHE\_DSS | 3DES\_EDE\_CBC | SHA |
| TLS\_DHE\_RSA\_WITH\_3DES\_EDE\_CBC\_SHA | DHE\_RSA | 3DES\_EDE\_CBC | SHA |
| TLS\_DHE\_DSS\_WITH\_AES\_256\_CBC\_SHA | DHE\_DSS | AES\_256\_CBC | SHA |
| TLS\_DHE\_RSA\_WITH\_AES\_256\_CBC\_SHA | DHE\_RSA | AES\_256\_CBC | SHA |
| TLS\_DHE\_DSS\_WITH\_AES\_128\_CBC\_SHA256 | DHE\_DSS | AES\_128\_CBC | SHA256 |
| TLS\_DHE\_RSA\_WITH\_AES\_128\_CBC\_SHA256 | DHE\_RSA | AES\_128\_CBC | SHA256 |
| TLS\_DHE\_DSS\_WITH\_AES\_256\_CBC\_SHA256 | DHE\_DSS | AES\_256\_CBC | SHA256 |
| TLS\_DHE\_RSA\_WITH\_AES\_256\_CBC\_SHA256 | DHE\_RSA | AES\_256\_CBC | SHA256 |

# Question 14

**Ans :** The key exchanges supported by TLS 1.3 are DHE(Diffie-Hellman ephemeral) and ECDHE (Elliptical curve Diffie-Hellman ephemeral) only.

* TLS\_AES\_256\_GCM\_SHA384
* TLS\_CHACHA20\_POLY1305\_SHA256
* TLS\_AES\_128\_GCM\_SHA256
* TLS\_AES\_128\_CCM\_8\_SHA256
* TLS\_AES\_128\_CCM\_SHA256

# Question 15

**Ans :** Privacy breach is still an issue with the TLS 1.2 even though the application data that gets transferred in the connection handshake will be encrypted. Studying the user’s activities without actually decrypting the data is called profiling. There are many attempts which study the user patterns and behavior by using no more information than packet size, timing and direction . The traffic features adopted in profiling include just the packet-level statistical values, such as average packet size and average interarrival time etc., in the MAC layer. This is one way profiling is being carried out and hence the privacy is breached.

**PLAGIARISM STATEMENT**

I certify that this assignment/report is my own work, based on my personal

study and/or research and that I have acknowledged all material and sources

used in its preparation, whether they be books, articles, reports, lecture notes,

and any other kind of document, electronic or personal communication. I also

certify that this assignment/report has not previously been submitted for

assessment in any other course, except where specific permission has been granted

from all course instructors involved, or at any other time in this course, and that I

have not copied in part or whole or otherwise plagiarized the work of other

students and/or persons. I pledge to uphold the principles of honesty and responsibility at

CSE@IITH. In addition, I understand my responsibility to report honor violations by other students if I become aware of it.

Name: Nisha M

Date: 13/2/2022

Signature: Nisha M