***Nidanur Gunay 24231 CS411 HW4***

1)We know that PIN can be between 999-10000 since it is given tat it contains 4 digit. And R can be 2^7 and 2^8-1 since k0 determined as 8 and we know that R is unsigned int. So we need to look for it in brute force. In order the search the all possible numbers I’ve iterated 2 for loops. One is for R that in range 2^7 and 2^8-1 and one is for PIN that in range 999 and 10000. When I run the program, I’ve get the result as R is 157 and PIN is 5377.

2)We know that k is random number that between 1 and 2^16-1. In order to find k, we can apply brute force since g, p and r values are given. After the brute force I found the k values as 17106. Then I calculate the modular inverse of h^k in mod p and my result is 21544490089832216808308331319858540081496426686327159697934566494456896122678783934289942511159118535387215740734123568454049466127172977648991549952804928046015294301116475551022590359054101472806177602513365484586670572061046115461644803844036767259566663632888024277264102550943997457333378166275287252966.

Since our plaintext m equals to inverse(h^k x t) in mod p, when we calculate it by the python code we found that m is 3356366513591938179202454688524890539966506364615741171676962160942064295810867763650735855139356337022790316030665447642707682606 and when we convert it to bytes we get the message as

b'My favorite machine at the gym is the vending machine.'

3) Ticket Granting Server supposed to include the identity of the server when responding to client. Response is (*EKCG (S, KCS),EKSG (C, TS3, LT, KCS) )* and *EKSG (C, TS3, LT, KCS)* is the ticket for the client to give it to server and by this way server would get that client has an authorization. *EKCG (S, KCS*) is used for the server’s identity. If it would not included ticket would not be specified for a service. It could be used for other services either. In that case client would be able to access any service with the ticket that he granted from the ticket granting service. So any server would be able to attacked by the client.

4)By the station and station protocol, they can achieve forward secrecy.

In that protocol Alice and Bob have their private key pairs that includes their session private keys let call that their private session keys as a and b respectively. And their public keys that can be reachable to everyone would be and When Alice encrypt Bob’s public key with her private key she would get and when Bob has done the same thing he would get and call them as T. Then both would have same key by hashing (T||param). After that, bob signs and encrypts the generated signs by using the key that he got by hashing. Then Alice decrypt the sign with the same key that she got by hashing. After she got the sign, she verifies the sign with their private keys and generate a new sign with the same key that generated by hashing. After bob receives the encrypted sign, he decrypts it with his K and after he verifies it, forward secrecy would be achieved.

Session private keys chosen randomly in every session and If the eve would get the session private keys, eve could not receive the plaintext due to the frequent alteration of the session private keys.

5) We know how to find alpha from the ch10\_handout.pdf and we can see that alpha is (si \* hj – sj \* hi \* x) \* modinv((sj\*ri\*x – si\*rj),q). And as you can see from my DSA.py we found alhpa as 482061878283805054797834433118192109249147480995. In order to find hi and hj values I calculate the g^si in mod p.

With the help of the Sig\_Ver function, I made verification of alpha and it verified.