COMPSCI 755: Cryptography and Security Protocol

Final Exam –Tuesday March 14, 2023

(Please upload a pdf or word file to the Canvas "Final" dropbox by the deadline)

NAME:____Gunnar Yonker_____ **SCORE: _____**

I. (**10 points**) True or false questions.

1. Frequent key changes are usually desirable to limit the amount of data compromised if an attacker learns the key.
   True False

2. Any timestamp based procedure must allow for a window of time sufficiently large enough to accommodate network delays yet sufficiently small to minimize the opportunity for attack.
   True False

3. The challenge-response approach is unsuitable for a connectionless type of application because it requires the overhead of a handshake before any connectionless transmission, effectively negating the chief characteristic of a connectionless transaction.
   True False

4. X.509 defines the format for private-key certificates.
   True False

5. Each user must share a unique key with the key distribution center for purposes of key distribution
   True False

6. The Handshake Protocol is one of the three SSL-specific protocols that use the SSL Record Protocol.
   True False

7. The SSL Record Protocol is optional when https is used.
   True False

8. A digital signature can guarantee the integrity but not the source of the message.
   True False

9. Message authentication not only protects two parties who exchange messages from any third party but also protects the two parties against each other.
   True False

10. Message encryption does not provide a measure of authentication.
    True False

II. (**15 points**) Multiple choice questions.

1. An SSL session is an association between a client and a server and is created by the _____ .
   A spec Protocol B. user
   C. handshake Protocol D. administrator

2. If Alice wants to initiate a session key exchange with Bob, what kind of key can Alice use to encrypt the session key?.
   A. private key of Bob B. master key shared by Alice and Bob
   C. private key of Alice D. public key of Alice

3. A digital signature serves the function of _____.
   A. integrity check B. authentication
   C. non-repudiation D. All of the above

4. _____ is a procedure that allows communicating parties to verify that the contents of a received message have not been altered and that the source is authentic.
   A. Identification B. Message authentication
   C. Verification D. User authentication

5. The _____ is formed by taking the hash of the message and encrypting the message with the creator's private key.
   A. timestamp B. message digest
   C. hash code D. digital signature

6. In RSA algorithm, the public key value ***e*** is normally a not very large number. But it also should not be too small since that might provide clue for cryptanalysis. Among the following numbers, which one is the best?
   A. 5 B. 65537
   C. 11 D. 3245678

7. The Diffie-Helman protocol was designed for _____.
   A. authentication B. key exchange
   C. digital signarure D. All of the above

III. (20 points) Suppose that one has designed a block cipher operation mode as

$C_i = P_{i-1}$ XOR $E_K(P_i)$

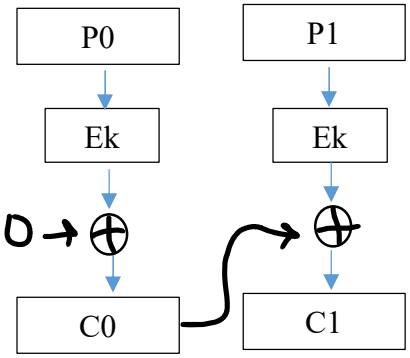
$P_0 = IV$

1. **Write the formula for decrypting in the receiver side. Also illustrate encryption and decryption diagram for this scheme.**

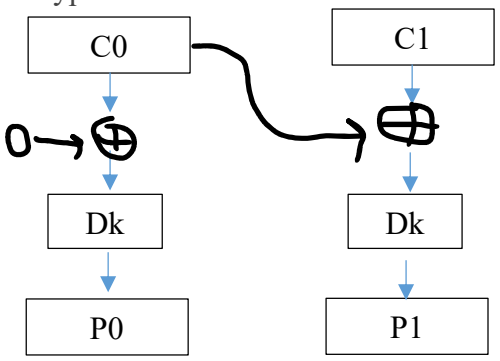
Ci = Ci – 1 XOR Ek(Pi) ➔ Ci XOR Ci-1 = Ek(Pi)

➔ Dk(Ci XOR Ci-1) = Pi

Encryption:



Decryption:



2. **If an error happened to C_3 during the transmission (all other blocks are transmitted correctly), which blocks of plaintext could not be recovered at the receiver? Please explain.**

If an error occurred at C3 during the transmission, the blocks of plaintext that would not be recovered would be P3, P4, P5… and all subsequent blocks. This is because if C3 was missing, then it would not be available to C4 to use to get P4, and then C4 would not be available to C5 to get P5 and so on. They are dependent on the encryption of C3 to compute the subsequent blocks.

3. **If an error happened to P_3 before encryption, actually all plaintext blocks except P_3 could be recovered correctly at the receiver Please prove this.**

If an error occurred to P3 before encryption, all plaintext blocks except for P3 could be recovered because each of the cipher blocks depends on the previous plaintext block and the encryption key, it does not rely on any of the subsequent plaintext blocks. P3 would continue over with the error that it had, but in the decryption as seen in the above diagram, the plaintext block is not used for further decryption. So P3 would have an error but would not be used for the decryption of P4 and P4 would be recovered successfully. The rest of the plaintext would be successfully recovered following that same principle.

4. **Is it possible to execute encryption in parallel? How about decryption? Justify your answer.**

It would be possible to execute encryption in parallel because each cipher block only depends on the previous plaintext block and the encryption key. So, it would be possible to have encryption in parallel. It would not be possible to have decryption done in parallel because each plaintext block depends on the previous cipher block and the decryption key which means they need to be computed sequentially.

5. **How do you think about the design of this block cipher operation design? Is it usable or secure? Justify your answer.**

I think that it is a simple design and that it could be usable, but that it is not secure because it has many vulnerabilities. The biggest vulnerability that this block cipher operation design has is that it is vulnerable to side-channel attacks which allows an attacker to use different types of analysis to gain information about the key that is being used. In the right situation I think that this design could be used, but I do not think that it is secure and should be used. Especially when you compare this design to the other cryptographic operations, we have learned about that are much more efficient and offer a higher level of security.

IV. (5 points) Compare the following two block operation designs. Which one is better at defending cryptanalysis? Justify your answer.

(1) $C_i = C_{i-1}$ XOR $E_K(P_i)$, $C_0 = IV$

(2) $C_i = E_k(C_{i-1})$XOR $E_K(P_i)$, $C_0 = IV$

Cipher block operation 2 is better at defending cryptanalysis because it uses the previous cipher block as Ci-1 encrypted with the key Ek as the input to the XOR operation with the current plaintext block as Pi encrypted with the same key. This is better at defending cryptanalysis because the input to the XOR operation is the encrypted previous cipher block so it does not directly relate to the plaintext if it were to be compromised. Encrypting the previous cipher block also adds security because the attacker would no longer be able to see that cipher block directly.

V. (10 points) Alice is sending an email to Bob. The email message (could include attachments of files up to 10 M Bytes) should be encrypted. Normally the email delivery do not employ mutual authentication. Instead it just uses one way communication.

Based on the above requirements, please

1. **Design a one message security protocol between the email sending server (Alice's side) and the email receiving server (Bob's side). Note that we assume that the two email servers have the public key of each other before hand.**

A handshake protocol message between Alice and Bob could be designed as such assuming that each side has the public key from the other beforehand:
Step 1. Alice generates a session key and signs the message digest of her email using her own private key. Alice's email server encrypts the message and any attachments and the message digest using the session key, and then encrypts the session key and the encrypted email (attachments etc.) to send to Bob's email server using Bob's public key.
Step 2. Bob's email server decrypts the session key using his private key and verifies the message digest signature using Alice's public key. If it is valid then Bob's email server uses the decrypted session key to decrypt the email message and any attachments.
Step 3. Once that is complete, Bob's email server can send a confirmation message to Alice's email if they choose to, to let Alice know that the message has been received and decrypted successfully.

2. **Explain how did the receiver authenticate the sender and obtain the plaintext email message.**

In the protocol above, Alice's email server is authenticated through the use of her digital signature using her private key which Bob can verify using her public key once receiving the digital signature. By doing this, the receiver (Bob) can authenticate that the sender was Alice and then continue on with decrypting the email message and attachments using the session key that was decrypted using Bob's private key. This session key is only known to Bob and Alice, so they are able to use that session key to carry out any secure communication between them.

VI. (15 points) Password file hashing.

1. **To make sure the passwords of users are securely stored, we normally need to hash each password in the password file. Is SHA-512 a good candidate for password hashing? Why or why not?**

SHA-512 is commonly used and is a one-way hashing function, producing a fixed output of 512 bits which helps to defend against rainbow table attacks. However, it is a relatively fast process which makes a dictionary attack more feasible so I do not think it is a good candidate for password hashing. Using a hashing method intended for passwords such as bcrypt where salt can be added in when hashing the passwords adds in additional security and makes attacks such as the dictionary attack much less feasible because it is a longer process.

2. **Somebody suggested a protocol like this: the client side hash the password before sending to the server side, but both username and password are not encrypted during the transmission. The server just need to compare the presented hashed password with the corresponding record in the hashed password file to find out whether the password presented was accurate or not. Is this a secure solution? Justify your conclusion.**

Sending the username and password not encrypted during the login process is not a secure solution. If there were any third parties eavesdropping they would be able to see the username and the hashed password which would allow that attacker to attempt to use the username to login since it can be seen in plaintext. If they also had access to the approved hashed password and that was all that was used to check on the server for the login they could potentially gain access to the server using the credentials they obtained. This also sets up the possibility of multiple compromised accounts if the server password file was compromised the attacker would easily be able to replay the hashed values sent by the client to obtain the plaintext passwords being used. To make this protocol secure, the communication between the client and server needs to be encrypted so that the credentials are secure. This solution is not a secure solution since the username and password are not encrypted during the transmission.

3. **Suppose a company has n users and n password records that are stored in the hashed format. Each password is <= 10 characters long and there are 128 possible choices of characters. A hacker has obtained this password file and plan to use the brute-force way to find out all n passwords.**
    i. **What is the lowest computing complexity for the case when salt was <u>not</u> used?**

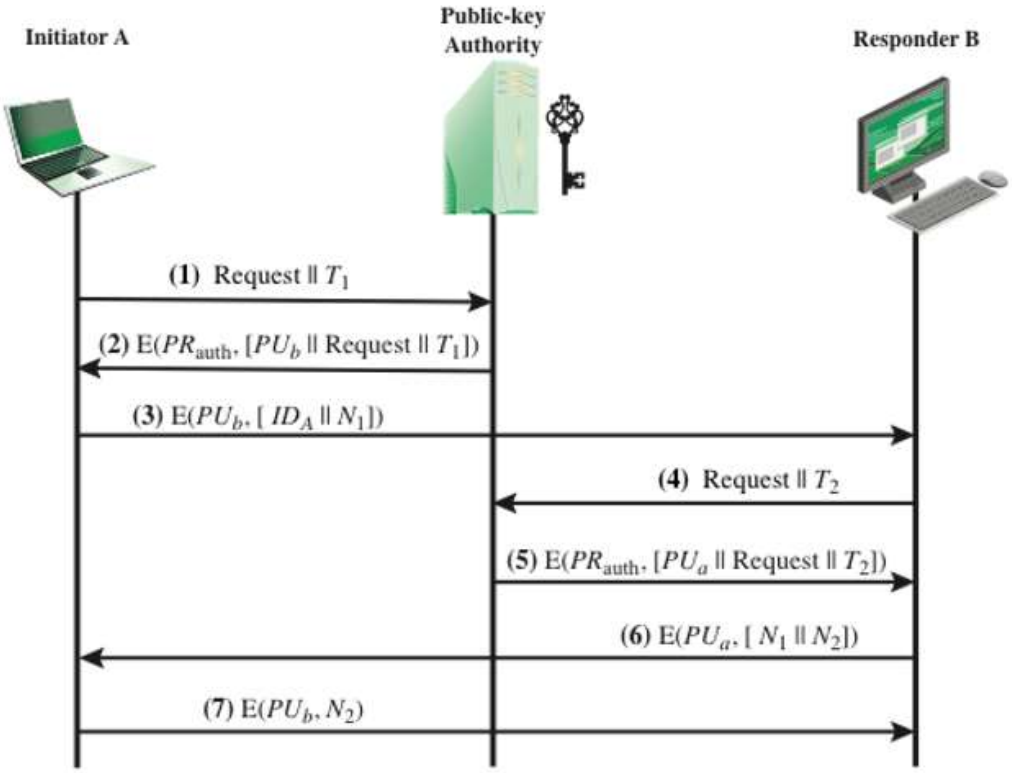
    (128^10) would be the maximum number of hash computations that would need to be completed, this would be feasible but could be time-consuming based on how many hashes the attacker was able to do per second. A brute-force attack would be used in this.

    ii. **What is the lowest computing complexity for the case when salt was used? Note that for both cases you need to consider the practicality of available memory.**

    When salt is introduced, it makes the brute-force attack or even a dictionary attack much more difficult and time consuming since each password will have a unique salt that needs to be checked for each

individual password. This would mean that there would be (128^10) * n hashes that would need to be computed, n would be the number of passwords because for each password every possibility would have to be tried with the unique salt identifier. For example if there were 2 passwords, password 1 would have 128^10 hashes to check with salt 1, and then the process would repeat for password 2 using salt 2 to check 128^10 new hashes. The length of the salt may be limited as well by the available memory because longer salt for each user requires more memory. This is resistant to an attacker using a dictionary attack because they cannot precompute the hashed passwords to guess because each guess needs to match the unique salt. The time needed to brute-force salted passwords becomes infeasible to being useful to the attacker.

VII. (15 points) Following graph shows one secure public key exchange protocol that goes through the public key authority.
1. Explain the operations and the purpose of important fields in the message for each step.
2. Give a brief summary on how does this protocol defend against the replay attack.
3. Is there any shortcoming about this public key exchange scheme? Can you propose any other scheme that could be more efficient?



1.
Step 1: A sends a message to the Public-key authority with A's request for B's public key and includes a timestamp as T1.
Step 2: The Public-key authority responds to A with a message that includes A's request, the timestamp T1, and B's public key which is all encrypted using the Public-key authority's private key. A is able to decrypt this using the public-key authority's public key.
Step 3: A sends a message to B that contains A's identity, the IDA, and a random nonce N1. This is all encrypted using B's public key PUb so that B can decrypt it using it's private key.
Step 4: B sends a request message with timestamp T2 to the public key authority to request A's public key.
Step 5: The public-key authority sends a message to B that contains A's public key, B's request, and the timestamp T2. This is all encrypted using the public-key authority's

private key so that then B can decrypt it using the public-key authority's public key and obtain the information they requested.
Step 6: B sends a message to A that contains the original nonce N1 that A sent to B, and a new nonce N2. They are encrypted using A's public key PUa so that when A receives it they can decrypt it using their private key to check if the original N1 nonce is the same which ensures there is not a replay attack taking place.
Step 7: A sends a message back to B the includes the new nonce N2 and is encrypted using PUb which is B's public key. B can then decrypt this message and check that the N2 matches.

2.
This public key exchange protocol defends against replay attacks by employing the use of the two nonces N1 and N2 that are exchanged between A and B. The nonces are random numbers that are used in the message to ensure that the messages are not being replayed and this ensures that if there was an attacker attempting a replay message they would not be able to use the same message and establish a connection with A and B to compromise their data.

3.
The only shortcoming about this public key exchange scheme that I can think of is that there is a lot of message exchanges that need to take place which can lead to a increased latency and leads to being less efficient. A protocol scheme that may be more efficient than this one could be the Elliptic Curve Diffie-Hellman (ECDH) protocol. This is a key exchange agreement that allows two parties to establish a shared secret key that is then used for further encryption and decryption of the communication. ECDH is based on the Diffie-Hellman key exchange algorithm but with the addition of using elliptic curves to achieve faster computations and using smaller key sizes. Nonces would also be used in this protocol to defend against replay attacks.

VIII. **(10 points)**

1. **TLS and IPSec are the two popular choices for implementing VPN. Which of these two protocols is better for VPN in terms of security, cost, and easiness of use?**

TLS and IPSec both offer a high level of security when implementing it into a VPN. They both have protocols that provide strong levels of encryption and authentication methods. TLS is a less expensive option than IPSec and is less expensive to implement, this is because TLS is more widely used so many web servers and applications already have support for it. IPSec could require specific specialized software or hardware to be deployed in order for it to work properly with a VPN. TLS is also generally easier to use than IPSec. This again is because TLS is more widely used so there is less work needed when trying to implement it into a VPN in terms of configuration and setup. IPSec on the other hand will potentially require more setup and configuration which can be more difficult for a user if they are not already familiar with the process. When picking one of these two protocols for a VPN in terms of security, cost and ease of use I would say that TLS is the better choice of the two options. It will most likely cost less, be much easier to setup and configure, and will provide a secure and strong encryption/authentication mechanisms when compared to IPSec.

2. **What is S/MIME? What are the four principle services provided by S/MIME?**

S/MIME stands for Secure/Multipurpose Internet Mail Extensions and is a standard for secure email messaging using encryption and digital signatures. It is widely used to provide secure email communications and most email clients use S/MIME.
The four principle services are:
1. Confidentiality – Email messages are encrypted using a symmetric encryption algorithm.
2. Integrity – A message digest algorithm is used to create a digital signature of the message, this helps to ensure that the message has not been tampered with.
3. Authentication – Digital certificates are used to verify the identity of the sender and the recipient.
4. Non-repudiation – The digital signature cannot be repudiated by the sender, this means that the sender cannot deny having sent the message and this allows the receiver to prove that the message was indeed sent by the sender.