CS433 Homework 1

1. Security Concepts (1.8 Exercise 12)

While all these harms and concerns are interrelated in the web of security concepts, you can dive deeper into their individual relations. The harm of interception can be closely related to the concept of confidentiality. Interception referring to unauthorized access or disclosure of confidential information. This directly concerns confidentiality as a threat since it can compromise sensitive information and disclose it to an unwanted party. The harm of modification is directly related to the concern of integrity, as you want to receive the data in an unaltered state without any compromises. If you allow any unauthorized tampering of the data then you are violating its integrity. Lastly, availability can be directly related to interruption. This is pretty straight forward as interruption can compromise a service or resource's availability, i.e. via a denial of service.

2. Secure Voting (1.8 Exercise 22)

Considering a program that accepts and tabulates votes during an election, parties that may want to attack the program include hacktivists or a terrorist/criminal organization, political candidates within the election. Types of harm include manipulating vote counts to skew the results or discredit the election. They could also tamper with votes to influence the election in such a way that would benefit them economically. They could also just disrupt the election process which would create internal chaos and mistrust in the democratic process. Leaking sensitive information could also influence voters' outcomes. They could exploit weak encryption algorithms or protocols at the critical points of communication within the different components of the program. They could also just flood the system with traffic, which could affect availability and integrity.

3. Smart lock (Design a "smart" lock for your home that does a 3-factor authentication that checks something the user knows AND something the user has AND something the user is.)

Your smart lock could include these 3 factors of authentication:

1. Something the user knows
   - A password or PIN
2. Something the user has
   - RFID or app verification (DUO style)
3. Something the user is

- Fingerprint scanner or facial recognition.

You could first require the RFID to be in range, which would prompt the user for a PIN and lastly verify it by a biometric such as a face ID.

4. Authentication That learns (2.5 Exercise 11)

The system would begin with a basic user information authenticaion such as a username and password. It would then gather information during the user's interactions with the system, such as :

- which programming language they use
- a heatmap of times they like to use the system and for how long
- what resources they access or utilize during their session
- frequently visited domains
- preferred file types
- when they make pulls/pushes to a remote repository
  Authentication challenges become more individualized as the system begins to learn more about the user's preferences. This could include:
- True/False questions based on the user's habits
- Multiple choice questions
- Requests for security questions such as their favorite novel or year they graduated
- Recognition tasks, where the user identifies the resources they commonly use.
  As the system continues to learn about the user, it could introduce additional factors such as Biometric authentication or a location-base one. i.e. Which machine they are using the system from. They could require the user to register and authorize only certain devices. Of course, the system would include robust encryption to protect all sensitive data about the user so not to compromise it. As well as regular security audits to identify and address any vulnerabilities.

5. Synchronous Password (2.5 Exercise 17)

Detecting and compensating for clock drift in a synchronous password token is crucial to maintain integrity and security. Some methods the receiver can employ include:

- Timestamp comparison. Compare each time stamp sent by the token and the difference is the estimated clock drift
- Accumulated drift tracking. Maintain a history of timestamp differences in a rolling window. Calculate the drift average over the window to smooth out any differences. You can use this average drift to adjust the timing of the next expected token.
- Artificial Intelligence/Machine learning. By keeping track of timestamp differences you can create a learning model that can accomodate and predict clock drift in your next token. However, this could

be extensive to train and computationally expensive.

6. Access Control Mechanisms (2.5 Exercise 1)

1. Per-Subject Access Control List:

- (a) ease of determining authorized access during execution:
  - Moderate. Checking per subject access control during execution involves looking up a single list for the specific subject.
- (b) Ease of Adding Access for a New Subject:
  - Easy. Adding a new subject involves creating a new access control list for that subject, specifying the objects the subject has access to.
- (c) Ease of Deleting Access by a Subject :
  - Easy. Removing access for a subject requires modifying or deleting the subject's access control list.
- (d) Ease of Creating a New Object with Default Access:
  - Moderate. Creating a new object with default access may require updating the access control lists for all existing subjects or implementing a default policy.

2. Per-Object Access Control List:

- (a) ease of determining authorized access during execution:
  - Moderate ease. Checking a per-object access control list involves looking up a single list for the specific object, which is relatively straightforward
- (b) Ease of Adding Access for a New Subject:
  - Moderate. Adding a new subject involves modifying the access control list for each object the subject needs access to.
- (c) Ease of Deleting Access by a Subject :
  - Moderate. Removing access for a subject requires modifying the access control list for each object the subject had access to
- (d) Ease of Creating a New Object with Default Access:
  - Easy. Creating a new object with default access involves setting up a new access control list for that object

3. Access Control Matrix:

- (a) ease of determining authorized access during execution:
  - Moderate ease. Checking the matrix involves finding the specific cell at the intersection of a subject and an object
- (b) Ease of Adding Access for a New Subject:

  - Easy. Adding a new subject involves adding a new row to the matrix.
- (c) Ease of Deleting Access by a Subject :
  - Easy. Removing access for a subject involves deleting the row corresponding to that subject.
- (d) Ease of Creating a New Object with Default Access:
  - Easy. Creating a new object involves adding a new column to the matrix, specifying default access for all subjects.

4. Capability:

- (a) ease of determining authorized access during execution:
  - High ease. Checking a capability involves verifying the possession of a specific token or key associated with the object.
- (b) Ease of Adding Access for a New Subject:
  - Easy. Granting access to a new subject involves creating a new capability or token and providing it to the subject.
- (c) Ease of Deleting Access by a Subject :
  - Easy. Revoking access for a subject involves invalidating or deleting the subject's capability.
- (d) Ease of Creating a New Object with Default Access:
  - Difficult. Capabilities are typically tied to specific objects, and creating a new object with default access might require establishing a new capability system-wide.

7. Traffic Access Control (You are charged to protect the machine of Alice from receiving unwanted traffic from machines on a blacklist.)

(1) If the access control is to be enforced by the machine of Alice, will the implementation of access control be an ACL, or a directory?

- In this scenario, the access control is enforced locally on Alice's machine. ACLs are commonly used for this purpose. An ACL is a set of rules that define what types of traffic are allowed or denied. Alice's machine would have a list specifying the IP addresses or other characteristics of machines on the blacklist, and it would block or allow traffic based on these rules.
  (2) If the access control is to be enforced by a firewall machine right in front of each sender, will the implementation of access control be an ACL, or a directory?
- In this case, the firewall machine in front of each sender would likely use ACLs to control access. The ACLs would be configured to block traffic from machines on the blacklist. Each firewall acts as a gatekeeper for the traffic leaving the sender and entering Alice's network, controlling which packets are allowed to pass through.
  (3) If the access control is to be enforced by routers en route from every sender to the machine of

Alice, which will inspect the traffic to either block or pass the traffic, what will be the access control implementation you would choose?

- Routers en route can enforce access control using NIPS or packet filtering rules. Network-Based Intrusion Prevention Systems inspect the traffic for malicious patterns and block or allow it accordingly. Alternatively, routers can be configured with packet filtering rules to drop packets from machines on the blacklist, effectively controlling access before the traffic reaches Alice's machine.

8. AES Lifetime (2.5 Exercise 21)

While it's difficult to predict the life of a cryptographic algorithm, due to the ever changing landscape of computing power, AES does have some advantages to DES. AES has larger key sizes of up to 256 bits, which provides a much higher level of security against brute force attacks. AES has undergone scrutiny and review before being selected by the NIST. This standardization enhances confidence in AES. AES is also designed to be more flexible than DES, allowing for easy adaptation in security requirements. There is still ongoing research contributing to the discovery of new algorithms as security threats evolve. Standards are constantly being updated to address these emerging threats. AES is also designed to be resistant to parallel processing attacks, making it more robust against certain types of attacks compared to DES. In theory, quantum computers have the potential to break AES encryptionm as they can perform large scale computations much faster than a classical machine. Shor's algorithm is designed for quantum computers, and could theoretically break many public-key encryption schemes including AES, by finding its decryption key significantly faster than classical computers. However, building such a machine obviously still poses an enormous technical challenge today. Experts at NIST estimate that building such a machine could take several decades. While quantum computing certainly poses a threat, it equally poses the potential for more advanced security algorithms that even the world's most powerful machines couldn't break. I believe this is the next logical step after AES-128 and we are introduced to quantum based encryption algorithms.

9. RSA (With n=10, show the procedure how you may choose a corresponding public and private key, and verify it works for a plain text P=2.)

Step 1:

- Choose two distinct prime numbers p and q, such that their product n = pq = 10. p =2, q = 5
  Step 2:
- Calculate the totient function. φ(n) = (p-1)(q-1).
- φ(10) = (2-1)(5-1) = 4
  Step 3:
- Choose an integer e such that 1 < e < φ(n) and e is co-prime with φ(n)

- Choosing e = 3 satisfies these conditions.
  Step 4:
- Calculate the private key, d, as the modular multiplicative inverse of e mod φ(n), meaning d*e ≡ 1 (mod φ(n))
- Using the Extended Euclidean Algorithm, we find d = 3 (since 3*3 ≡ 1 (mod 4)).
- Public key: (n, e) = (10, 3)
- Private key: (n, d) = (10, 3)

Encryption:

- $C = P^e \bmod n$
- $C = 2^3 \bmod 10$
- C = 8

Decryption:

- $P = C^d \bmod n$
- $P = 8^3 \bmod 10$
- P = 2

10. Diffie-Hellman key exchange. Describe how Diffie-Hellman key exchange protocol works, and why it is resilient against eavesdroppers.

Pretend Jose and Mert want to communicate using this protocol. So they will follow these steps.
Step 1:

- Jose has public key P, G
- Mert also has public key P, G

Step 2:

- Jose selected private key a
- Mert selected private key b

Step 3:

- Jose Generates key by $x = G^a \bmod P$
- Mert will generate key by $y = G^b \bmod P$

Step 4:

- Exchange of generated keys will take place
- Jose receives Y
- Mert receives X

Step 5:

- Jose will generate the key by K_a y^a mod P
- Mert will generate the key by K_b x^b mod P

Step 6:

- it will now be K_a = K $_b$

This algorithm is resilient against eavesdroppers since it uses hidden proceses, session specific keys, forward secrecy, and parameter sensitivity.

11. Alice and her friends. Alice has 50 friends, and she wishes to send all of them a common message secretly. That is, each friend will receive the same message, and only these 50 friends can decrypt it. Alice knows the public key of each of them. How would you design the message for Alice?

I would design the message to utilize hybrid encryption. This way, a symmetric key is generated to encrypt the common message, and then the symmetric key is encrypted using the receiver's public key. This way, only her friends with their corresponding private keys can decrypt the symmetric key and subsequently decrypt her message.