Devoir I – CEG 4399

CEG 4399- Design of Secure Computer Sys.

Université d'Ottawa

Professeur : Amir H. Razavi

Noms et numéros des étudiants :
Gbegbe Decaho Jacques 300094197

Date de soumission: 11 October 2024

Assignment 1

Introduction

UNIVERSITY OF Ottawa
Faculty of Engineering and Computer Science
CSI 4139-CEG 4399 - SEC 5100 – Fall 2024
Assignment 1

Note

Some questions might be open-ended and require your own opinions and creativity, as the answers can vary considerably.

Question 1 (15%)

Based on the definitions we had for “Threats”, “Vulnerabilities”, and “Controls”, bring two different examples or scenarios, and indicate each of these three aspects in each scenario.

Les scénarios impliquant les concepts de “threats”, “Vulnerabilities”, “Controls” sont :

scenario 1: **A social platform (social networks)**

- **thread**: unauthorized access attempt to a user account.
- **Vulnerability**: User using the same password on all platforms or having a weak password
- **Control:** offer a password manager, favor the use of complex passwords (mixture of numbers, letters, special characters and capital letters)

scenario 2: **A banking platform (online banking)**

- **thread:** attempt to steal a user account login ID.
- **Vulnerability**: User using a weak password, or unencrypted communication between the bank server and the user's application
- **Control:** implement dual authentication and encryption protocols to secure communication.

Question 2 (10%)

Using some sentences or examples, show how the four kinds of threats, "Interception", "Interruption", "Fabrication", and "Modifications" relate to the three concepts, preserving "Confidentiality", "Integrity", and "Availability".

The "**Interception**" threat type is related to the concept of "**Confidentiality**". We can illustrate it by taking as an example the interception of an unencrypted data communication between an online bank and a user.

The "**Interruption**" threat type is related to the concept of "**Availability**". We can illustrate it by taking as an example the unavailability of the user's data when he needs it.

The "**Modification**" threat type is related to the concept of "**Integrity**". We can illustrate it by taking as an example the change without consent of legitimate data, thus compromising its reliability and accuracy.

The "**Fabrication**" threat type is related to the concept of "**Integrity**". We can illustrate it by taking as an example the addition of false data in a system, thus causing the processing of the operation with unauthorized or inaccurate data.

Question 3 (10%)

Do you believe attempting to break into a computing system without authorization should be illegal? Why or why not? Bring at least two examples/scenarios to support your answer.

I believe that attempting to penetrate a computer system without authorization should be illegal. Because it endangers the privacy of individuals, public safety, and the stability of the affected structure. That is why these actions must be illegal to protect the interests of users and affected organizations.
To support this, we can cite as examples:

Scenario 1: **Personal Data Theft**
A hacker attempts to gain access to a financial company's (bank) customers' personal data and credit card numbers. This exposes this data to abuse such as fraud or identity theft, compromising users' privacy and security.

Scenario 2: **Hardware Sabotage**
A hacker attempts to break into a hospital's computer system to disrupt the operation of connected medical equipment. This could endanger patients' lives by interrupting vital treatments.

Question 4 (15%)

For each of the following two programs, answer the three questions followed:

1. A program that accepts and tabulates votes in an election.
2. A program that allows consumers to order products from the web.

- Who might want to attack the program?
- What type of harms might they want to cause?
- What kinds of vulnerabilities might they exploit to cause harm?

1. A program that accepts and tabulates votes in an election
    - Who might want to attack the program?
      Competing political parties, hackers or spies wanting to destabilize the country.

    - What type of harms might they want to cause?
      Invalidate the votes of certain voters, alter voting results and create errors in the vote count.

    - What kinds of vulnerabilities might they exploit to cause harm?
      Transmitting data without encryption, exploiting database security vulnerabilities (managing permissions).

2. A program that allows consumers to order products from the web.
    - Who might want to attack the program?
      Hackers, a competing store or scammers wanting to obtain bank card numbers

    - What type of harms might they want to cause?
      Causing financial loss and loss of trust to customers, stealing sensitive or personal information, disrupting customer service

    - What kinds of vulnerabilities might they exploit to cause harm?
      Exploiting flaws in the order payment system or in the storage of sensitive data, vulnerabilities in the verification of user identities.

Question 5 (10%)

One-Time Pad is the only cryptosystem that provides ***Perfect Secrecy***.

- Describe the advantages and disadvantages of this cryptosystem.
- Bring one example in real-world applications, in which One-Time Pad is suitable to be used, and one example that is not. Justify your answers.

1. The advantages and disadvantages of this cryptosystem
    - **Advantages :**
        - No information leakage: each encrypted message can match any message of the same length, which guarantees absolute confidentiality even if the message is intercepted.
        - No dependence on security assumptions: Its security does not depend on any mathematical assumptions. It is therefore secure by design, not by computational complexity.
    - **Disadvantages :**
        - Message length limitation: the generated key must be exactly equal to the message length. If we want to encrypt a very long message, we must also provide an equally long key.
        - Impracticality in modern networks: Each communication requires a secure key distribution between the two parties, which is impossible to manage in infrastructures with millions of users

2. The example in real-world applications
    - **Communications during armed conflict:** During armed conflicts for encrypted messages, keys can be exchanged physically in complete security, for example via trusted agents, and messages can be encrypted without risk of decryption by the enemy because key management is easier to organize and is limited to pre-planned situations.
    - **Cloud storage:** If an organization wants to encrypt its data on the cloud it also requires storing a key as large as the data to be protected because, The volume of data in the cloud is often huge, and the need to reuse or share keys would be inevitable, which would compromise the security of the system.

Question 6 (10%)

Rotor machines were used by Germany (Enigma) and Japan (Purple) in World War II. Watch this short clip on the Enigma rotor machine:

https://www.khanacademy.org/computing/computer-science/cryptography/crypt/v/case-study-ww2-encryption-machines

It consists of a set of independently rotating cylinders, each of which has 26 input pins and 26 output pins. Each input pin is connected to a unique output pin using internal wiring. You can see a related diagram in the following link, under the title "Rotor Machine":

http://sjsu.rudyrucker.com/~haile.eyob/paper/#3.%20Classic%20Cryptography

- A single-cylinder defines a mono-alphabetic substitution. Considering a 5-rotor machine, what would be the equivalent key length of a Vigenere cipher for this machine? Explain your answer.
- Humans are said to be the weakest link in any security system. Give two examples of human failure that could lead to compromise of encrypted data.

- The equivalent key length of a Vigenere cipher for this machine will be :
  A key of length 5. Because each rotor has 26 different positions and when it turns it produces a polyalphabetic cipher similar to that of the cipher, for a machine with 5 rotors we just do five times the 26 possible positions (26^5 = 11 881 376).

- The examples of human failure that could lead to compromise of encrypted data :
  - **Insecure Key Sharing:** If an encryption key is shared over an insecure channel, such as an email or unencrypted text message, it would allow an attacker intercepting this communication to compromise the security of the data.
  - **Misuse of Encryption Software:** Users sending sensitive files without encryption software into an organization without enabling certificate authentication or using an older version of the software opens up vulnerabilities for an attacker.

Question 7 (5%)

Based on the convention we use to represent the English alphabet using numbers 0 to 25, formulate Atbash Cipher by showing two mathematical expressions, one for encryption and one for decryption. Show the correctness of your expressions with one example.

- **Encryption :**
  - Expression : C = 25 - x
  - Example : let b (the 2nd letter of the alphabet) ⇒ C = 25 - 1 = 24 ⇐⇒ C is equivalent to Y
- **Decryption :**
  - Expression : D = 25 - C
  - Example : 25 - 24 ⇒ 1⇒ B (correct) so the decipherment leads back to the original letter

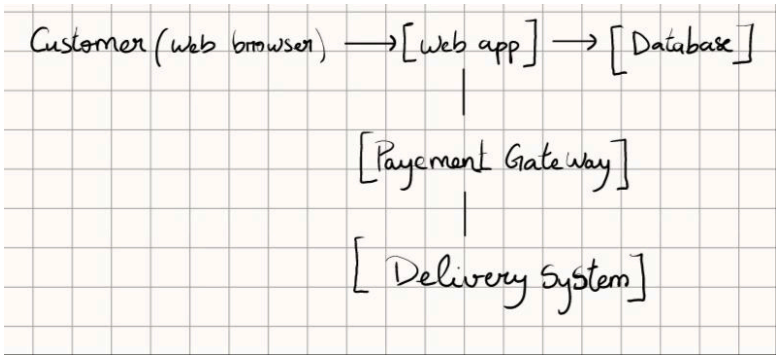
Question 8 (5%)

Apply threat modeling to identify potential security flaws in a system:

- Choose a simple ecommerce web application and create a Data Flow Diagram (DFD) of the system.
- Using the STRIDE model, identify at least five threats that exist within the application.
- Propose specific mitigations for each threat and justify how these mitigations align with security design principles.
- Submit your DFD, a threat model table (Threats, Vulnerabilities, Mitigations), and a brief report explaining how the mitigations improve the security of the system.

The e-commerce choose is an mailbox delivery

1. **Data Flow Diagram (DFD)**



2. **Threat model table**

Stride Category	Threat	Vulnerability	Mitigation
Spoofing	An attacker poses as a legitimate customer to place fraudulent orders.	Inadequate or missing customer authentication mechanisms.	Implement multi-factor authentication (MFA) to verify that customers are legitimate users.
Repudiation	A customer denies making a purchase, leading to a dispute.	Absence of non-repudiation controls, such as digital receipts.	Maintain digital signatures for transaction receipts and audit trails to ensure non-repudiation.
Information Disclosure	An attacker intercepts or gains access to sensitive customer information.	Sensitive data, such as customer personal information or payment details, is not encrypted.	Encrypt sensitive data at rest using AES encryption for databases, and in transit using SSL/TLS.

Denial of Service	Attackers overwhelm the system, rendering the service unavailable to customers.	Lack of protection against high traffic volumes or automated requests.	Implement rate limiting and deploy DDoS protection services, such as Cloudflare, to mitigate attacks.
Elevation of Privilege	An attacker gains administrative privileges, granting full control over the system.	Insufficient access control mechanisms and poor privilege management.	Implement role-based access control (RBAC) and limit administrative actions to authorized personnel only.
Tampering	An attacker modifies product details or delivery information during transit.	Absence of encryption for data transmitted between the client and server.	Implement SSL/TLS encryption to secure data in transit, preventing unauthorized modification by attackers.

3. **Mitigation and Alignment with Security Design Principles**
    - **Spoofing Mitigation:** Implementing Multi-Factor Authentication (MFA) verifies customer identities with an extra factor, strengthening authentication and reducing unauthorized access.

    - **Tampering Mitigation:** Encrypting data in transit using SSL/TLS ensures data integrity and confidentiality, preventing attackers from altering information during transmission.

    - **Repudiation Mitigation:** Logging transactions with digital signatures and audit trails ensures accountability and provides proof of transaction origin, reducing disputes.

    - **Information Disclosure Mitigation:** Encrypting sensitive data both at rest and in transit protects customer information, maintaining confidentiality even if data is compromised.

- **Denial of Service Mitigation:** Using rate limiting and DDoS protection prevents system overload, ensuring service availability even during high traffic or attacks.

- **Elevation of Privilege Mitigation:** Role-Based Access Control (RBAC) restricts administrative actions to authorized users, minimizing the risk of privilege escalation and enhancing security.

4. **Brief report**

   In this mailbox delivery e-commerce application, several key threats were identified using the STRIDE model, and specific mitigations were proposed to improve security. Implementing multi-factor authentication significantly reduces unauthorized access by adding a second layer of protection, thereby improving authentication. SSL/TLS encryption ensures data integrity and confidentiality, preventing tampering of sensitive information during transmission.

   Digital signatures and transaction logs ensure non-repudiation, empower users, and reduce the risk of fraud.

   Encrypting sensitive data at rest and in transit secures customer information, preventing unauthorized disclosure.

   Rate limiting and DDoS protection maintain service availability during attacks, ensuring continued access to the system.

   Finally, RBAC limits user capabilities, reducing the risk of privilege escalation and improving access control through the principle of least privilege.

Question 9 (10%)

Apply Secure SDLC principles to a practical example. Imagine you are part of a team developing an online payment system:

- Outline how you would integrate security practices into each phase of the SDLC:
  - Requirements
  - Design
  - Implementation
  - Testing
  - Deployment
- Submit a detailed report describing your security plan for each SDLC phase, along with a few real-world examples of how security failures could be avoided with proper SDLC practices.

Secure SDLC Plan for an Online Payment System :

1. Security practices integration :
   - **Requirements**
     - **Regulatory needs**: Ensure compliance with standards like PCI-DSS for handling payment data.
     - **Sensitive data handling**: Define secure processing and storage of sensitive information (e.g., credit card data).

    - **User authentication:** Implement strong authentication mechanisms, including MFA.
    - **Encryption**: Enforce encryption for data in transit and at rest.
    - Fraud protection: Include mechanisms to detect and prevent fraud and data breaches.
- **Design**
    - **System architecture:** Incorporate encryption, SSL/TLS for secure communication, and RBAC for access control.
    - **Threat modeling:** Identify risks like SQL injection and XSS, and plan corresponding defenses.
    - **Error handling and logging:** Ensure errors and logs do not expose sensitive information
- **Implementation**
    - **Secure coding practices:** Implement input validation and sanitization to prevent injection attacks.
    - **Secure libraries:** Use trusted libraries and frameworks for encryption, authentication, and session management.
    - **Code reviews:** Regularly review code to identify and address security vulnerabilities.
- **Testing**
    - **Security testing:** Conduct penetration testing, vulnerability scanning, and code audits to find weaknesses.
    - **Common vulnerabilities:** Test for issues like XSS, SQL injection, and buffer overflows.
    - **Security feature validation:** Ensure encryption, authentication, and other security measures function as intended.
- **Deployment**
    - **Secure environment:** Configure firewalls and intrusion detection systems for deployment security.
    - **Regular updates:** Apply patches and updates to address new vulnerabilities.
    - **System monitoring:** Continuously monitor for suspicious activities and implement backup/recovery plans.

2. **Detailed report**

   Requirements : Identify security needs based on regulations (e.g., PCI-DSS) and business goals, defining data handling, authentication, and encryption. Example: Requiring complex passwords and implementing CAPTCHA helps prevent automated brute-force attacks.

Design : Ensure encryption (SSL/TLS), RBAC, and secure communication. Use threat modeling to identify risks like SQL injection and XSS. Example: Designing role-based access control (RBAC) ensures that only authorized users access sensitive functions.
Implementation : Follow secure coding practices (input validation, sanitization), use secure libraries, and conduct code reviews. Example: Using parameterized queries in database interactions prevents SQL injection attacks.
Testing : Perform penetration tests and scan for vulnerabilities (e.g., XSS, SQL injection). Example: Vulnerability scanning helps detect and mitigate cross-site scripting (XSS) attacks before deployment.
Deployment : Secure configurations, apply patches, monitor for suspicious activity, and have backup/recovery plans. Example: Configuring firewalls and intrusion detection systems reduces the risk of unauthorized access during deployment.

Question 10 (10%)

- How would you test a ciphertext to quickly determine if it was likely the result of a simple **substitution**?
- How would you test a ciphertext to quickly determine if it was likely the result of a **transposition**?

- To test a ciphertext to quickly determine if it was likely the result of a simple substitution, we will :
  - make a analysis of the frequency
  - make a analysis of the Mono/Bigram

- To test a ciphertext to quickly determine if it was likely the result of a simple transposition, we will :
  - make a analysis of the frequency
  - make a analysis of the pattern
  - make a simple reversing of the anagrams