CS-3610: Information Security (Spring 2025)

Assignment Type: Individual

Assignment - 1 **Marks: 100**
Assign Date: Feb 3, 2025 *Submit Date: 23:55, Feb 9, 2025*

Setup

This question ensures that you have the necessary tools for this assignment and the course.

Pre-requisites:

- A text editor (e.g., VSCode)
- Python 3.7 or higher
- Familiarity with basic terminal commands
- OpenSSL installed (check with: `openssl version`)
- If OpenSSL is not installed, install it using:

```
sudo apt-get install openssl
```

Task 1: Generating Public and Private Keys

Using OpenSSL, generate an asymmetric key pair.

Steps:

1. Generate a 2048-bit RSA private key (`private_key.pem`).
2. Extract the corresponding public key (`public_key.pem`).
3. Encrypt a message with the public key and decrypt it with the private key.
4. Explain how asymmetric encryption ensures confidentiality.

Deliverables:

- Commands used for key generation.
- Encrypted and decrypted messages.
- A brief explanation.

Task 2: Secure Symmetric Key Exchange

Establish a secure symmetric key exchange using OpenSSL.

Steps:

1. Generate a random symmetric key.
2. Encrypt the symmetric key using the recipient's public key.
3. The recipient decrypts it using their private key.
4. Use the shared key to encrypt and decrypt a message.

Deliverables:

- Commands and scripts for key exchange.
- Encrypted and decrypted messages.
- Explanation of security measures.

Task 3: Setting Up a Certificate Authority

Create a self-signed Certificate Authority (CA) and use it to sign a public key.

Steps:

1. Generate a CA private key and self-signed certificate.
2. Generate a Certificate Signing Request (CSR).
3. Use the CA to sign the CSR, issuing a certificate.
4. Verify the certificate using the CA's certificate.

Deliverables:

- Commands for CA setup and signing.
- The CA certificate, CSR, and signed certificate.
- Explanation of the CA's role in PKI.

Task 4: Signing a Public Key with the CA

Use the previously created CA to sign and validate a public key.

Steps:

1. Submit a public key to the CA.
2. CA signs and returns a certificate.
3. Validate the signed certificate using OpenSSL.

Deliverables:

- Commands for signing and validation.
- The signed certificate.
- Explanation of digital certificates.

Task 5: Man-in-the-Middle (MITM) Attack

Implement a MITM attack on a key exchange protocol.

Steps:

1. Simulate Alice and Bob's encrypted communication.
2. Introduce Mallory as an adversary.
3. Show how Mallory intercepts and modifies messages.
4. Suggest countermeasures.

Deliverables:

- Code demonstrating the attack.
- Screenshots of intercepted messages.
- Countermeasures against MITM attacks.

Task 6: Needham-Schroeder Protocol and Denning-Sacco Attack

Implement the Needham-Schroeder protocol and demonstrate the Denning-Sacco attack (refer to lecture slides for this).

Steps:

1. Implement the Needham-Schroeder protocol in Python.
2. Simulate communication between two parties.
3. Introduce an adversary who reuses an old key exchange message.
4. Demonstrate the attack and suggest mitigation techniques.

Deliverables:

- Python code implementing the protocol.
- Output showing the attack.
- Explanation of the attack and mitigation techniques.