Lab Assignment - 2

Modern Crypto Assignment – Mayank

Fundamentals of Cryptography

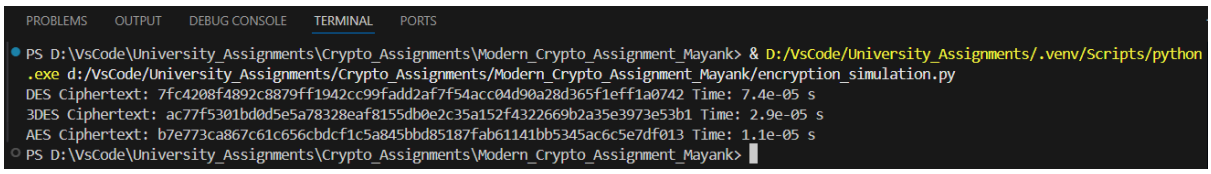
Course Code: ENSP259

By: Mayank Rawat Submitted to: Dr. Anshu

GitHub Repo - https://github.com/mayank24000/University_Assignments/tree/main/Crypto_Assignments/Modern_Crypto_Assignment_Mayank

Q-1. Symmetric Key Encryption Simulation Encrypt the message "Confidential Transaction" using all three algorithms: • DES • 3DES • AES

Solution:



Q-2. Modes of Operation for Block Ciphers Encrypt the same message using AES under the following modes: • ECB, CBC, CFB, OFB, CTR

Solution:

```
PS D:\VsCode\University_Assignments\Crypto_Assignments\Modern_Crypto_Assignment_Mayank> & D:/VsCode/University_Assignments/.venv/Scripts/python
.exe d:/VsCode/University_Assignments/Crypto_Assignments/Modern_Crypto_Assignment_Mayank/aes_modes.py
ECB ciphertext: a8a62ca6428ad3538eef76c0a97e90b291fdac8237f6818359...
CBC ciphertext: 20a52a77754c2339ed4774b3efdc88ac1946d80b8b80d048d5...
CFB ciphertext: ad1eda81fa6816f6b73920724c1657dbecd8dbe69d1d619dcf...
OFB ciphertext: 55e59462f778678aa615ccfb3af3b0c3ef4d5eca1493a11c63...
CTR ciphertext: 931dbdfb2bbef147063619a0a72331b1fa71c4c6f1b9f22714...
PS D:\VsCode\University_Assignments\Crypto_Assignments\Modern_Crypto_Assignment_Mayank>
```

Q-3 Cryptographic Hash Functions Hash the file sample_input.txt using the following algorithms: • MD5 • SHA-1 • SHA-256

Solution:

```
PS D:\VsCode\University_Assignments\Crypto_Assignments\Modern_Crypto_Assignment_Mayank> & D:/VsCode/University_Assignments/.venv/Scripts/python
.exe d:/VsCode/University_Assignments/Crypto_Assignments/Modern_Crypto_Assignment_Mayank/hashing_demo.py
MD5: d41d8cd98f00b204e9800998ecf8427e
SHA-1: da39a3ee5e6b4b0d3255bfef95601890afd80709
SHA-256: e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855
PS D:\VsCode\University_Assignments\Crypto_Assignments\Modern_Crypto_Assignment_Mayank>
```

Q-4 HMAC and Data Integrity Simulate a secure file transmission using HMAC.

Solution:

```
PS D:\VsCode\University_Assignments\Crypto_Assignments\Modern_Crypto_Assignment_Mayank> & D:/VsCode/University_Assignments/.venv/Scripts/python
.exe d:/VsCode/University_Assignments/Crypto_Assignments/Modern_Crypto_Assignment_Mayank/hmac_test.py
Original HMAC: f304c11274cabc93abe79f3abb848b94026d3e1c9a49071ea96fbece9b9f2bb0
Modified HMAC: 143ff9ae8d00a2d3674bd028b2cedb2e9edd31af8ddd26e01c889a360786458d
Data integrity compromised!
PS D:\VsCode\University_Assignments\Crypto_Assignments\Modern_Crypto_Assignment_Mayank>
```

Q-5 Replay Attack (Simulation) a) Simulate a simple authentication mechanism using tokens or timestamps. b) Show how replaying a valid token compromises security. c) Suggest prevention techniques (e.g., nonces, timestamps, OTPs).

Solution:

```
PS D:\VsCode\University_Assignments\Crypto_Assignments\Modern_Crypto_Assignment_Mayank> & D:/VsCode/University_Assignments/.venv/Scripts/python
.exe d:/VsCode/University_Assignments/Crypto_Assignments/Modern_Crypto_Assignment_Mayank/replay_attack.py
Generated Token: 1761320573.8812063
Replayed Token Valid? True
PS D:\VsCode\University_Assignments\Crypto_Assignments\Modern_Crypto_Assignment_Mayank>
```

Replay Attack Prevention Techniques

1. **Use Nonces (Random Unique Numbers)**
    - Each session or message includes a one-time random value.
    - The server tracks used nonces to ensure they can't be reused.
2. **Include Timestamps**
    - Every token or request carries the exact creation time.
    - The server only accepts requests within a short time window (e.g., 30 seconds).
3. **One-Time Passwords (OTPs)**
    - Tokens that expire after a single use or a few seconds (e.g., Google Authenticator).
    - Even if intercepted, an OTP cannot be reused.
4. **Session Tokens with Expiry**
    - Tokens or authentication sessions automatically expire after a fixed duration.
    - Forces users to re-authenticate, reducing replay risk.
5. **Digital Signatures or Message Authentication Codes (MACs)**
    - Attach cryptographic verification to each message, ensuring integrity and authenticity.
6. **TLS/SSL Encryption**
    - Encrypts the communication channel to prevent interception and replay of network packets.

7. **Server-side Tracking**
    - Maintain a log of previously used tokens or request IDs to detect duplicates.