COMSATS UNIVERSITY ISLAMABAD, ATTOCK CAMPUS

IS LAB TERMINAL

Submitted by:-

M ABDULLAH AZAM

Registration No:

SP24-BSE-034

Submitted to:

Ms. Ambreen GUL

Subject:

Information Security LAB

QUESTION NO 1

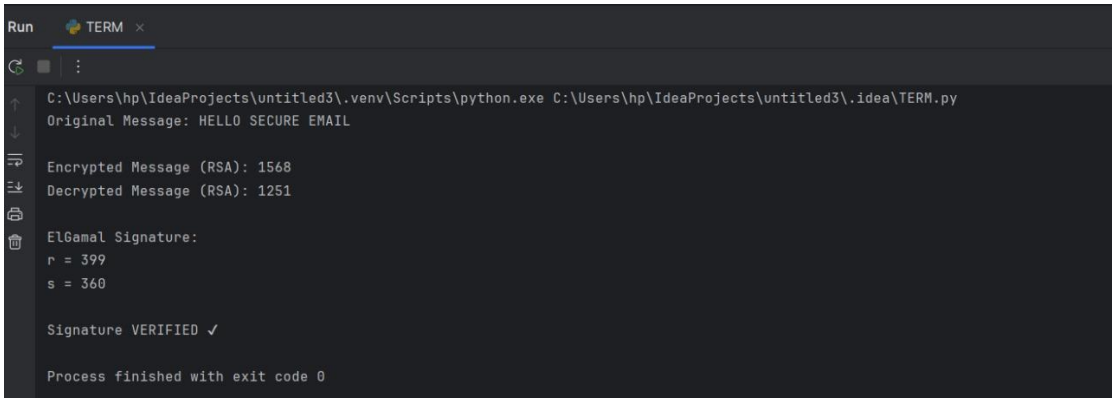
CODE :

```
from cryptography.hazmat.primitives.asymmetric import rsa, padding
from cryptography.hazmat.primitives import hashes
import random

message = b"HELLO THIS IS A SECURE EMAIL"
print("\nOriginal Message:", message.decode())


# 1. RSA KEY GENERATION (Receiver)

rsa_private_key = rsa.generate_private_key(
    public_exponent=65537,
    key_size=2048
)

rsa_public_key = rsa_private_key.public_key()


# 2. RSA ENCRYPTION (Sender)

ciphertext = rsa_public_key.encrypt(
    message,
    padding.OAEP(
        mgf=padding.MGF1(algorithm=hashes.SHA256()),
        algorithm=hashes.SHA256(),
        label=None
    )
)

print("\nEncrypted Message (RSA):", ciphertext)


# 3. RSA DECRYPTION (Receiver)

decrypted_message = rsa_private_key.decrypt(
```

```
    ciphertext,
    padding.OAEP(
        mgf=padding.MGF1(algorithm=hashes.SHA256()),
        algorithm=hashes.SHA256(),
        label=None
    )
)

print("\nDecrypted Message (RSA):", decrypted_message.decode())


# 4. ELGAMAL KEY GENERATION (Sender)

# Public parameters (demo values)
p = 467        # large prime (small for demo)
g = 2          # generator

# Private key
x = random.randint(1, p-2)

# Public key
y = pow(g, x, p)


# 5. ELGAMAL DIGITAL SIGNATURE (Sender)

# Hash message (simple integer hash)
h = int.from_bytes(message, 'big') % (p - 1)

k = random.randint(1, p-2)
r = pow(g, k, p)

k_inv = pow(k, -1, p-1)
s = (k_inv * (h - x * r)) % (p - 1)

print("\nElGamal Signature Generated:")
print("r =", r)
print("s =", s)


# 6. ELGAMAL SIGNATURE VERIFICATION (Receiver)
```

```
left = (pow(y, r, p) * pow(r, s, p)) % p
right = pow(g, h, p)

if left == right:
    print("\nSignature VERIFIED ✓ (Authentic & Intact)")
else:
    print("\nSignature INVALID ")
```

OUTPUT



QUESTION NO 2

1. Justification of the Security Method (Salsa20)

My project uses Salsa20, a stream cipher, as the main security method for protecting data. Salsa20 is suitable for this project because it is fast,

lightweight, and secure, making it ideal for applications that require efficient encryption without heavy computational overhead.

Compared to other encryption methods such as AES, Salsa20 performs especially well in software-based environments and on systems with limited processing power. It does not rely on complex hardware acceleration and is resistant to many common cryptographic attacks. Additionally, Salsa20 has a simple design and has been widely analyzed by the cryptographic community, which increases confidence in its security. For my project's requirements—secure and efficient data encryption—Salsa20 provides a better balance of performance and security than many alternative methods.

---

2. Possible Vulnerability or Weakness

One possible vulnerability in my current system is poor key or nonce management. Salsa20 requires that the same key and nonce combination must never be reused. If an attacker obtains two encrypted messages that were encrypted using the same key and nonce, they could perform a keystream reuse attack.

In such a case, an attacker could XOR the ciphertexts together to reveal patterns or potentially recover parts of the original plaintext messages. This could lead to sensitive information being exposed even though a strong encryption algorithm is being used.

---

3. Suggested Improvement to Enhance Security

A realistic improvement to enhance the security of my project would be to implement secure nonce generation and validation, such as using a cryptographically secure random number generator or a counter-based nonce system.

This improvement would ensure that each encryption operation uses a unique nonce, preventing keystream reuse attacks. Additionally, storing or tracking used nonces would help avoid accidental reuse. By strengthening nonce management, the project would significantly reduce the risk of cryptographic attacks and improve the overall security of the system

---