Practical 05: Implementation of RSA Algorithm in Cryptography

Objective:
To understand and implement the RSA (Rivest–Shamir–Adleman) asymmetric encryption algorithm and solve a numerical example to demonstrate encryption and decryption.

---

1. Theory

RSA is a **public-key cryptosystem** used for secure data transmission. It is asymmetric, meaning it uses two keys: a **public key** for encryption and a **private key** for decryption. RSA relies on the mathematical difficulty of factoring large prime numbers.

Key Features: - Asymmetric encryption (public/private keys) - Security depends on the difficulty of factoring large numbers - Widely used in secure communications (emails, digital signatures, etc.)

Applications: - Secure web communication (HTTPS) - Digital signatures - Data encryption

---

2. RSA Algorithm

Key Generation:

1. Select two large prime numbers, p and q .
2. Compute $n = p \times q$.
3. Compute Euler's totient: $\phi(n) = (p-1)(q-1)$
4. Choose integer e such that $1 < e < \phi(n)$ and gcd($e, \phi(n)$) = 1.
5. Determine d such that $d \times e \equiv 1 \pmod{\phi(n)}$
6. Public key: (e, n) , Private key: (d, n)

Encryption:

$$C = M^e \mod n$$

Where M = plaintext, C = ciphertext

Decryption:

$$M = C^d \mod n$$

Where M = decrypted plaintext

---

3. Program (Python Example)

```python
def gcd(a, b):
    while b != 0:
        a, b = b, a % b
    return a

def mod_inverse(e, phi):
    for d in range(1, phi):
        if (d * e) % phi == 1:
            return d
    return None

# Input primes
p = int(input("Enter prime p: "))
q = int(input("Enter prime q: "))

n = p * q
phi = (p-1)*(q-1)

# Choose e
e = 2
while e < phi:
    if gcd(e, phi) == 1:
        break
    e += 1

d = mod_inverse(e, phi)

print(f"Public Key: ({e}, {n})")
print(f"Private Key: ({d}, {n})")

# Encrypt message
M = int(input("Enter plaintext as integer: "))
C = pow(M, e, n)
print("Ciphertext:", C)

# Decrypt message
decrypted = pow(C, d, n)
print("Decrypted plaintext:", decrypted)
```

---

4. Numerical Example

Given:
$p = 7, q = 11, M = 8$

Step 1:

$$n = p \times q = 7 \times 11 = 77$$

$$\phi(n) = (p - 1)(q - 1) = 6 \times 10 = 60$$

Step 2: Choose e such that gcd(e, 60) = 1 → $e = 7$

Step 3: Find d such that $d \times 7 \equiv 1 \mod 60 \rightarrow d = 43$

Step 4: Encrypt $M = 8$

$$C = 8^7 \mod 77 = 2$$

Step 5: Decrypt $C = 2$

$$M = 2^{43} \mod 77 = 8$$

Result matches original plaintext.

---

5. Conclusion

- RSA provides a secure method for encrypting and decrypting messages using asymmetric keys.
- It ensures confidentiality and authentication through public-private key pairs.
- Numerical example demonstrates the encryption and decryption process accurately.
- Although RSA is computationally intensive for very large keys, it remains a cornerstone of modern cryptography.