HỌ VÀ TÊN: LÊ THỊ PHƯƠNG LINH
LỚP : DHKL16A1HN
MSV : 22174600057

```
# bài thực hành 1
```

markdown

```
pip install pycryptodome
```

✓ 6.8s Python

```
Requirement already satisfied: pycryptodome in c:\users\admin\appdata\local\packages\pythonsoftwarefoundation.python.3.11_qbz5n2kfra8p0\localcache\local-packages\python311\site-packages (3.23.0)
Note: you may need to restart the kernel to use updated packages.

[notice] A new release of pip is available: 25.0.1 -> 25.1.1
[notice] To update, run: C:\Users\admin\AppData\Local\Microsoft\WindowsApps\PythonSoftwareFoundation.Python.3.11_qbz5n2kfra8p0\python.exe -m pip install --upgrade pip
```

BƯỚC 1 : MÃ HÓA VÀ GIẢI MÃ AES

```
from Crypto.Cipher import AES
from Crypto.Random import get_random_bytes
from Crypto.Util.Padding import pad, unpad
import time

# Tạo khóa mã hóa 128-bit (16 byte)
key = get_random_bytes(16)

# Khởi tạo đối tượng AES ở chế độ CBC
cipher = AES.new(key, AES.MODE_CBC)

# Văn bản gốc cần mã hóa (dạng bytes)
plaintext = b"Hello, this is a test message for AES encryption!"

# Đo thời gian mã hóa AES
start_time = time.time()
ciphertext = cipher.encrypt(pad(plaintext, AES.block_size))
end_time = time.time()
aes_encryption_time = end_time - start_time

# In kết quả mã hóa
print("Văn bản mã hóa (AES):", ciphertext)
print("Thời gian mã hóa AES:", aes_encryption_time, "giây")

# Giải mã và đo thời gian giải mã AES
start_time = time.time()
decipher = AES.new(key, AES.MODE_CBC, cipher.iv)  # Sử dụng lại IV
decrypted_text = unpad(decipher.decrypt(ciphertext), AES.block_size)
end_time = time.time()
aes_decryption_time = end_time - start_time

# In kết quả giải mã
print("Văn bản giải mã (AES):", decrypted_text.decode())
print("Thời gian giải mã AES:", aes_decryption_time, "giây")
```

✓ 0.7s Python

```
Văn bản mã hóa (AES): b"\x1fK\x91\x87\xec\x00f\xd6\xe3\x0cgcY\xffx\xba\xc4\x1fcE\xa8\xb1\xf6\xe6\xd5\x0e\x14\xe6\xd2\x1a\xfb\x89\xb7.)\r\xac\xe0\xa4v\xa9TF\xaf\xdc\x90\\\xf4\x15^\xa9\xdf\xc63\x05\xf0\xc
Thời gian mã hóa AES: 0.0 giây
Văn bản giải mã (AES): Hello, this is a test message for AES encryption!
Thời gian giải mã AES: 0.0020158290863037 giây
```

BƯỚC 2: MÃ HÓA VÀ GIẢI MÃ RSA

```
from Crypto.PublicKey import RSA
from Crypto.Cipher import PKCS1_OAEP
from Crypto.Random import get_random_bytes
import time

# Tạo cặp khóa RSA 2048-bit
key = RSA.generate(2048)
private_key = key.export_key()
public_key = key.publickey().export_key()

# Sinh khóa AES ngẫu nhiên (16 byte)
aes_key = get_random_bytes(16)

# Mã hóa khóa AES bằng khóa công khai RSA và đo thời gian
cipher_rsa = PKCS1_OAEP.new(RSA.import_key(public_key))
start_time = time.time()
encrypted_aes_key = cipher_rsa.encrypt(aes_key)
end_time = time.time()
rsa_encryption_time = end_time - start_time

print("Khóa AES sau khi mã hóa bằng RSA:", encrypted_aes_key)
print("Thời gian mã hóa RSA:", rsa_encryption_time, "giây")

# Giải mã khóa AES bằng khóa bí mật RSA và đo thời gian
decipher_rsa = PKCS1_OAEP.new(RSA.import_key(private_key))
start_time = time.time()
decrypted_aes_key = decipher_rsa.decrypt(encrypted_aes_key)
end_time = time.time()
rsa_decryption_time = end_time - start_time

print("Khóa AES sau khi giải mã:", decrypted_aes_key)
print("Thời gian giải mã RSA:", rsa_decryption_time, "giây")
```

✓ 4.0s Python

```
Khóa AES sau khi mã hóa bằng RSA: b';\x1f\xb7\xc8K\xe7\x0c<\xba_z\x02Sn\xf6\x15\xc4\xa5\xa0\x0c\n\x81\x86i\xf3\x9f>\x137\x06p0\x8b\xec\x0c\x84\xf1\xd5\xa9\x11\xf8\xd4\xa4\xbe\xa2\xd9+\x9cp\xaf\xac3\xd64
Thời gian mã hóa RSA: 0.0 giây
Khóa AES sau khi giải mã: b'\xf6\xb95\xda\x86\xe2\xdf|\xcf\xc0\xe9\xaf$\xb0,\xd2'
Thời gian giải mã RSA: 0.010101795196533203 giây
```

```python
from Crypto.PublicKey import RSA
from Crypto.Cipher import PKCS1_OAEP
# Tạo cặp khóa RSA
key = RSA.generate(2048)
private_key = key.export_key()
public_key = key.publickey().export_key()
# Mã hóa khóa AES bằng khóa công khai RSA và đo thời gian
aes_key = get_random_bytes(16)
cipher_rsa = PKCS1_OAEP.new(RSA.import_key(public_key))
start_time = time.time()
print("Khóa AES sau khi giải mã:", decrypted_aes_key)
print("Thời gian giải mã RSA:", rsa_decryption_time, "giây")
```

✓ 0.5s Python

```
Khóa AES sau khi giải mã: b'\xf6\xb95\xda\x86\xe2\xdf|\xcf\xc0\xe9\xaf$\xb0,\xd2'
Thời gian giải mã RSA: 0.010101795196533203 giây
```

1. Tại sao mã hóa AES có tốc độ nhanh hơn đáng kể so với RSA?
Mã hóa AES có tốc độ nhanh hơn RSA vì AES là thuật toán mã hóa đối xứng, sử dụng cùng một khóa để mã hóa và giải mã, với quy trình tính toán đơn giản và đã được tối ưu tốt trên cả phần mềm lẫn phần cứng. Trong khi đó, RSA là mã hóa bất đối xứng, dựa trên các phép toán phức tạp như lũy thừa modulo với số nguyên lớn, khiến quá trình xử lý chậm hơn rất nhiều.

2. Trong thực tế, tại sao người ta thường kết hợp cả AES và RSA trong một hệ thống bảo mật?
Trong thực tế, người ta thường kết hợp AES và RSA để tận dụng ưu điểm của cả hai thuật toán: AES được dùng để mã hóa dữ liệu vì tốc độ nhanh và hiệu quả, còn RSA được dùng để mã hóa khóa AES nhằm đảm bảo an toàn khi chia sẻ khóa giữa các bên. Cách kết hợp này được gọi là mã hóa lai (hybrid encryption), giúp hệ thống vừa nhanh vừa bảo mật.

3. Dựa trên kết quả đo thời gian, loại mã hóa nào phù hợp hơn cho việc mã hóa dữ liệu dung lượng lớn?
Dựa trên kết quả đo thời gian, AES là loại mã hóa phù hợp hơn cho việc mã hóa dữ liệu dung lượng lớn vì có tốc độ xử lý nhanh, khả năng mã hóa dữ liệu theo khối hiệu quả và tiêu tốn ít tài nguyên hơn. Ngược lại, RSA chỉ phù hợp để mã hóa các dữ liệu nhỏ như khóa, vì tốc độ chậm và giới hạn độ dài dữ liệu được xử lý.

1. Tại sao mã hóa AES có tốc độ nhanh hơn đáng kể so với RSA?
Mã hóa AES có tốc độ nhanh hơn RSA vì AES là thuật toán mã hóa đối xứng, sử dụng cùng một khóa để mã hóa và giải mã, với quy trình tính toán đơn giản và đã được tối ưu tốt trên cả phần mềm lẫn phần cứng. Trong khi đó, RSA là mã hóa bất đối xứng, dựa trên các phép toán phức tạp như lũy thừa modulo với số nguyên lớn, khiến quá trình xử lý chậm hơn rất nhiều.

2. Trong thực tế, tại sao người ta thường kết hợp cả AES và RSA trong một hệ thống bảo mật?
Trong thực tế, người ta thường kết hợp AES và RSA để tận dụng ưu điểm của cả hai thuật toán: AES được dùng để mã hóa dữ liệu vì tốc độ nhanh và hiệu quả, còn RSA được dùng để mã hóa khóa AES nhằm đảm bảo an toàn khi chia sẻ khóa giữa các bên. Cách kết hợp này được gọi là mã hóa lai (hybrid encryption), giúp hệ thống vừa nhanh vừa bảo mật.

3. Dựa trên kết quả đo thời gian, loại mã hóa nào phù hợp hơn cho việc mã hóa dữ liệu dung lượng lớn?
Dựa trên kết quả đo thời gian, AES là loại mã hóa phù hợp hơn cho việc mã hóa dữ liệu dung lượng lớn vì có tốc độ xử lý nhanh, khả năng mã hóa dữ liệu theo khối hiệu quả và tiêu tốn ít tài nguyên hơn. Ngược lại, RSA chỉ phù hợp để mã hóa các dữ liệu nhỏ như khóa, vì tốc độ chậm và giới hạn độ dài dữ liệu được xử lý.

markdown

so sánh thời gia thực thi aes và rsa
Mã hóa AES là gần như tức thời: Với thời gian 0.0 giây, điều này có nghĩa là quá trình mã hóa diễn ra cực kỳ nhanh, đến mức không thể đo lường được bằng đơn vị nhỏ nhất mà hệ thống ghi nhận trong lần chạy này.
Giải mã AES cũng rất nhanh: Mặc dù mất một khoảng thời gian nhỏ hơn 0.003 giây (khoảng 0.002016 giây), quá trình giải mã vẫn diễn ra cực kỳ nhanh chóng.
Nhận xét
So sánh trực tiếp, quá trình mã hóa AES diễn ra nhanh hơn hoặc ngang bằng với quá trình giải mã AES trong trường hợp cụ thể này, với dữ liệu và môi trường chạy mà bạn đã thử nghiệm. Việc mã hóa mất "0.0 giây" thường ngụ ý rằng thời gian thực tế là một con số rất nhỏ, dưới ngưỡng đo lường của phép đo hiện tại, nhưng vẫn là một giá trị dương.

Điều này củng cố thêm khẳng định về hiệu suất vượt trội của thuật toán mã hóa đối xứng như AES, khiến nó trở thành lựa chọn lý tưởng cho việc bảo mật dữ liệu với khối lượng lớn, nơi mà tốc độ là yếu tố then chốt.

markdown