Experiment #	<TO BE FILLED BY STUDENT>	**Student ID**	<TO BE FILLED BY STUDENT>
Date	<TO BE FILLED BY STUDENT>	**Student Name**	THANOS

Lab 13: Write a python program to implement of RSA Algorithm

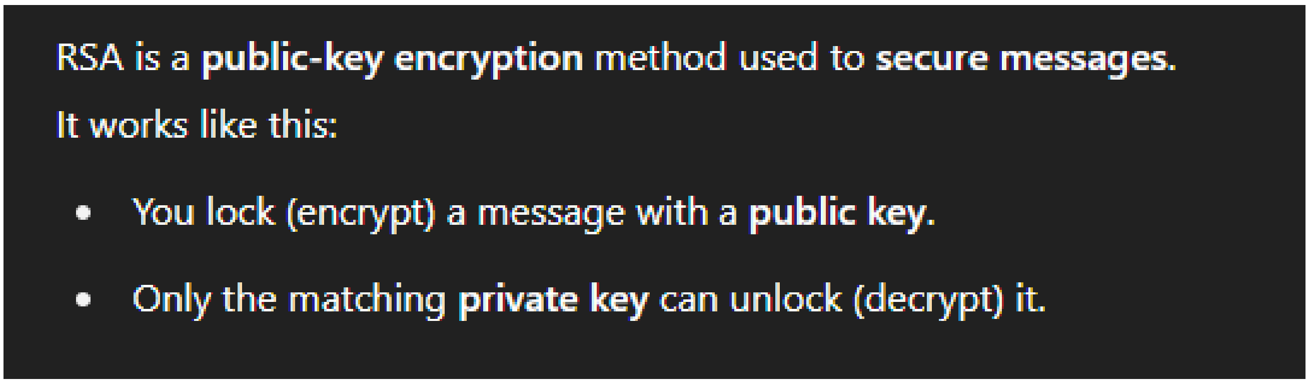
Date of the Session:_______/___/_____ **Session Time:_____to________**

Learning outcome:

- Learners will gain a comprehensive understanding of public key cryptography and the principles behind it.
- To understand and implement the concept of RSA Algorithm.

Pre-Lab Task:

1. Mention the RSA algorithm in simple terms. What is its purpose and how does it work?

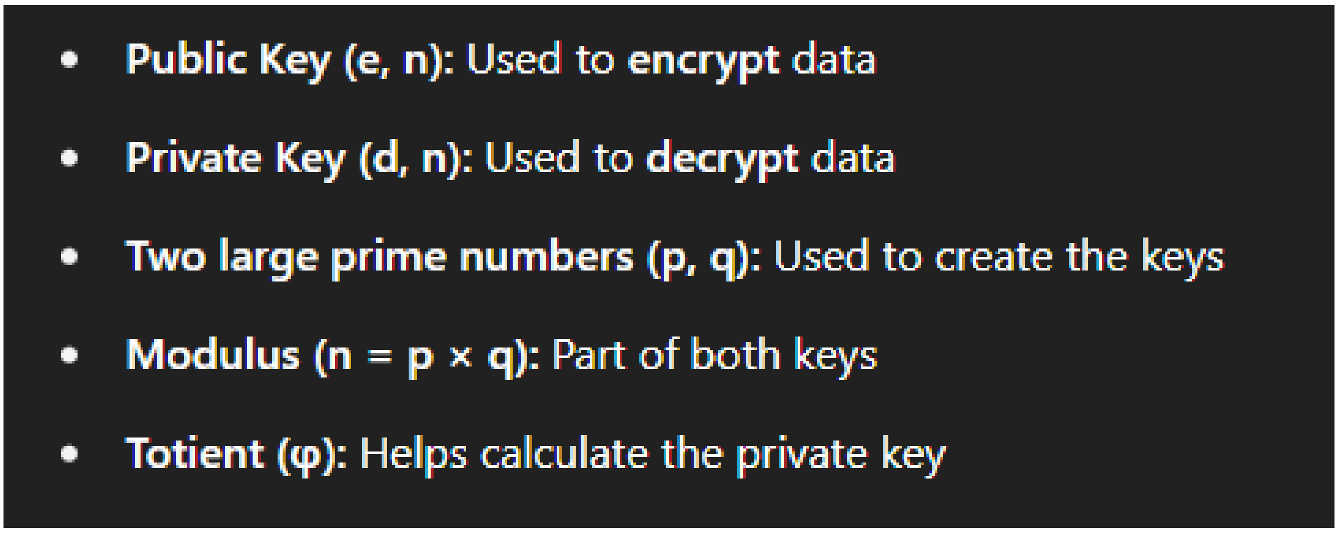

RSA is a **public-key encryption** method used to **secure messages.**

It works like this:

- You lock (encrypt) a message with a **public key.**
- Only the matching **private key** can unlock (decrypt) it.

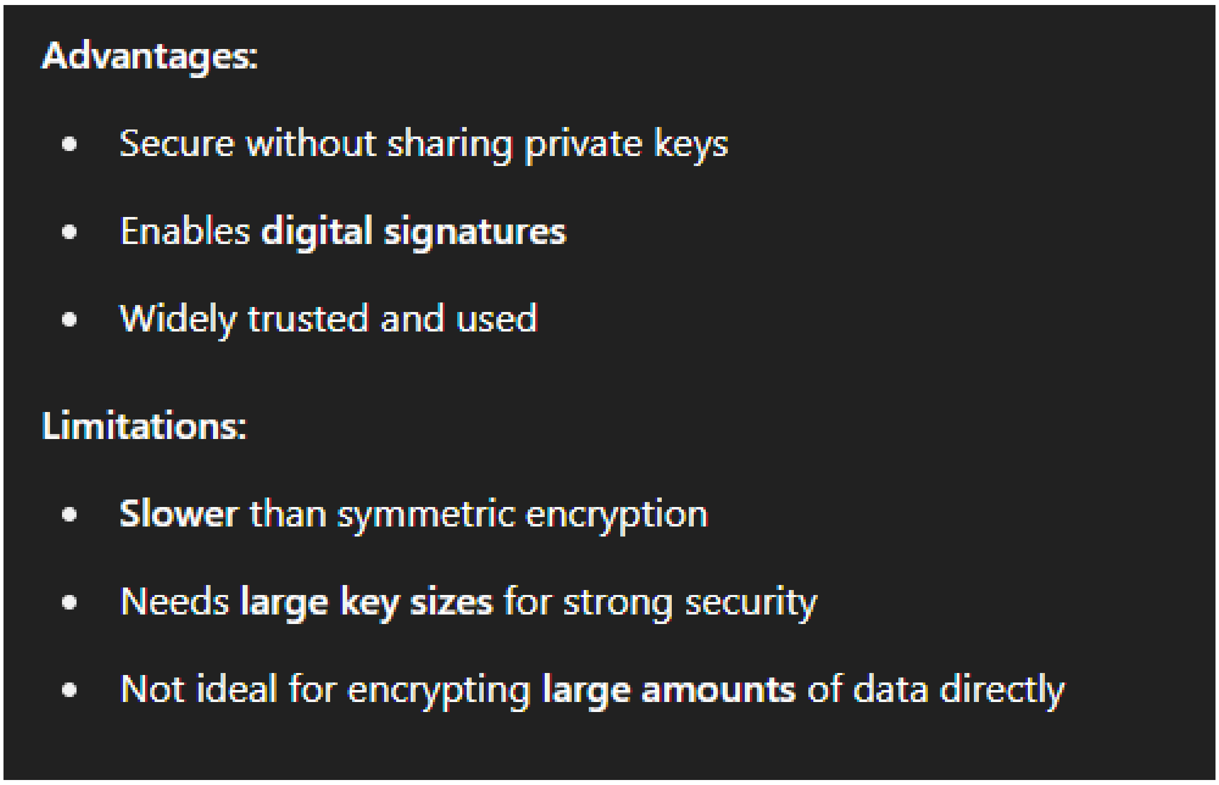
Experiment #	<TO BE FILLED BY STUDENT>	Student ID	<TO BE FILLED BY STUDENT>
Date	<TO BE FILLED BY STUDENT>	Student Name	THANOS

2. What are the key components of the RSA algorithm, and what roles do they play in the encryption and decryption processes?



- **Public Key (e, n):** Used to **encrypt** data
- **Private Key (d, n):** Used to **decrypt** data
- **Two large prime numbers (p, q):** Used to create the keys
- **Modulus (n = p × q):** Part of both keys
- **Totient (φ):** Helps calculate the private key

3. Discuss the advantages and limitations of the RSA algorithm.



Advantages:

- Secure without sharing private keys
- Enables **digital signatures**
- Widely trusted and used

Limitations:

- **Slower** than symmetric encryption
- Needs **large key sizes** for strong security
- Not ideal for encrypting **large amounts** of data directly

Course Title	NETWORK PROTOCOLS & SECURITY	ACADEMIC YEAR: 2024-25
Course Code(s)	23EC2210R	

Experiment #	<TO BE FILLED BY STUDENT>	**Student ID**	<TO BE FILLED BY STUDENT>
Date	<TO BE FILLED BY STUDENT>	**Student Name**	THANOS

In Lab Task: Write a python program to implement of RSA Algorithm

Implementing the RSA (Rivest-Shamir-Adleman) algorithm involves several steps for generating keys and performing encryption and decryption.

Here are the steps for implementing the RSA algorithm:

Key Generation:

Step 1: Select Two Large Prime Numbers (p and q):

Choose two distinct prime numbers, p and q. These should be large enough to provide security. The security of RSA relies on the difficulty of factoring the product of these two prime numbers (n = p * q).

Step 2: Calculate n and φ(n):

Calculate n = p * q, the modulus used for both the public and private keys.

Calculate φ(n) (Euler's totient function), which is the number of positive integers less than n that are coprime to n. φ(n) = (p - 1) * (q - 1).

Step 3: Select an Encryption Key (e):

Choose an encryption key (e) such that 1 < e < φ(n) and e is coprime to φ(n). Common choices include small prime numbers like 3 or 65537 (2^16 + 1). The value of e becomes part of the public key.

Step 4: Calculate the Decryption Key (d):

Calculate the modular multiplicative inverse of e modulo φ(n). In other words, find d such that (d * e) % φ(n) = 1. The value of d becomes part of the private key.

Encryption:

Step 5: Message Encoding:

Convert the plaintext message into an integer M. This can be done using various encoding schemes (e.g., ASCII, Unicode).

Step 6: Encryption:

Compute the ciphertext C by raising M to the power of e modulo n: C = (M^e) % n.

Experiment #	<TO BE FILLED BY STUDENT>	**Student ID**	<TO BE FILLED BY STUDENT>
Date	<TO BE FILLED BY STUDENT>	**Student Name**	THANOS

Decryption:

Step 7: Decryption:

Compute the plaintext message M by raising C to the power of d modulo n: M = (C^d) % n.

Step 8: Message Decoding:

Convert the integer M back to the original plaintext message using the same encoding scheme used for encryption.

Key Management:

Step 9: Key Storage and Protection:

Safeguard the private key, as it should be kept secret. The public key can be openly shared.

Cryptographic Strength and Security Considerations:

Step 10: Key Length Selection:

The security of RSA depends on the key length. Ensure that you use sufficiently long keys to resist attacks. For modern applications, key lengths of 2048 bits or higher are recommended.

Step 11: Periodic Key Renewal:

Consider periodically generating new key pairs and transitioning to them to enhance security.

Step 12: Protect Against Attacks:

Be aware of potential attacks on RSA, such as factoring attacks and timing attacks. Implement countermeasures and best practices to protect against these threats.

Step 13: Test and Validate:

Test the RSA implementation thoroughly to ensure that it functions correctly and securely.

Writing space for the Program: (For Student's use only)

Write a Python program to Implementation of RSA Algorithm

```
import random
from math import gcd

def is_prime(num):
    if num <= 1:
        return False
    for i in range(2, int(num**0.5)+1):
        if num % i == 0:
            return False
    return True

def generate_large_prime(keysize=8):
    while True:
        num = random.getrandbits(keysize)
        if is_prime(num):
            return num

def mod_inverse(e, phi):
    def extended_gcd(a, b):
        if a == 0:
            return b, 0, 1
        gcd, x1, y1 = extended_gcd(b % a, a)
        x = y1 - (b // a) * x1
        y = x1
        return gcd, x, y
    gcd_val, x, _ = extended_gcd(e, phi)
    if gcd_val != 1:
        raise Exception('Modular inverse does not exist')
    return x % phi

def generate_keys(keysize=8):
    p = generate_large_prime(keysize)
    q = generate_large_prime(keysize)
    while q == p:
        q = generate_large_prime(keysize)
    n = p * q
    phi = (p - 1) * (q - 1)
    e = 65537
```

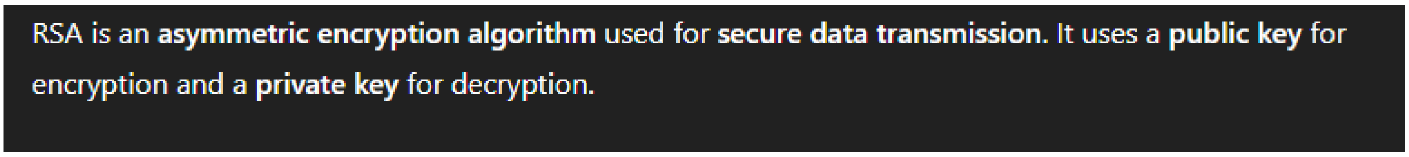
```
  if gcd(e, phi) != 1:
      e = 3
      while gcd(e, phi) != 1:
          e += 2
  d = mod_inverse(e, phi)
  return (e, n), (d, n)

def encrypt(message, public_key):
  e, n = public_key
  return [pow(ord(char), e, n) for char in message]

def decrypt(cipher, private_key):
  d, n = private_key
  return ''.join([chr(pow(char, d, n)) for char in cipher])

if __name__ == '__main__':
  public_key, private_key = generate_keys(keysize=8)
  print("Public Key:", public_key)
  print("Private Key:", private_key)
  message = "HELLO"
  print("\nOriginal Message:", message)
  cipher = encrypt(message, public_key)
  print("Encrypted Message (as integers):", cipher)
  decrypted_message = decrypt(cipher, private_key)
  print("Decrypted Message:", decrypted_message)
```

Experiment #	<TO BE FILLED BY STUDENT>	**Student ID**	<TO BE FILLED BY STUDENT>
Date	<TO BE FILLED BY STUDENT>	**Student Name**	THANOS

```
OutputPublic Key: (65537, 259)

Private Key: (17, 259)


Original Message: HELLO

Encrypted Message (as integers): [130, 237, 55, 55, 207]

Decrypted Message: HELLO
```

VIVA-VOCE Questions (In-Lab):

1. What is the RSA algorithm, and what is its primary purpose in cryptography?

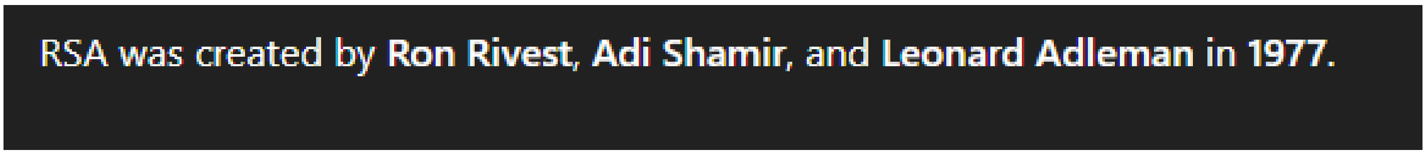


RSA is an **asymmetric encryption algorithm** used for **secure data transmission**. It uses a **public key** for encryption and a **private key** for decryption.

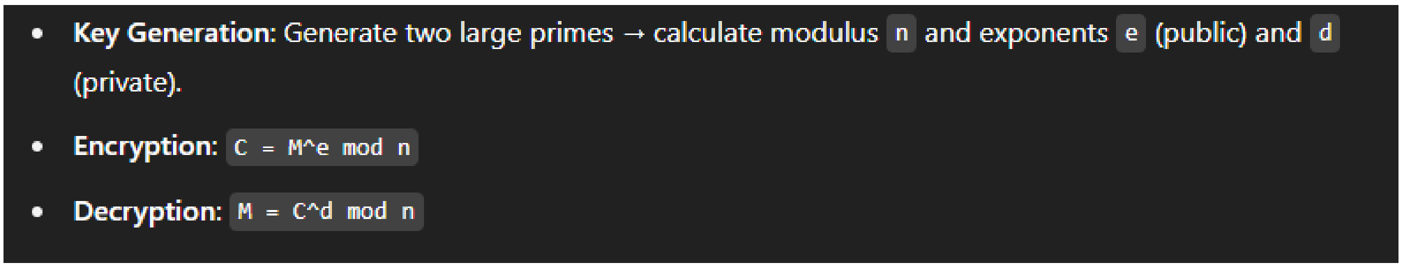
Course Title	**NETWORK PROTOCOLS & SECURITY**	**ACADEMIC YEAR: 2024-25**
Course Code(s)	**23EC2210R**	

Experiment #	<TO BE FILLED BY STUDENT>	**Student ID**	<TO BE FILLED BY STUDENT>
Date	<TO BE FILLED BY STUDENT>	**Student Name**	THANOS

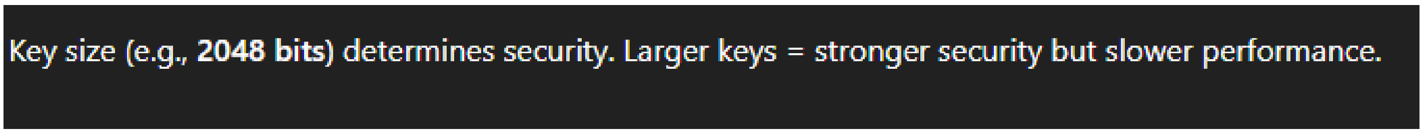
2. Who are the creators of the RSA algorithm, and when was it introduced?


RSA was created by **Ron Rivest**, **Adi Shamir**, and **Leonard Adleman in 1977.**

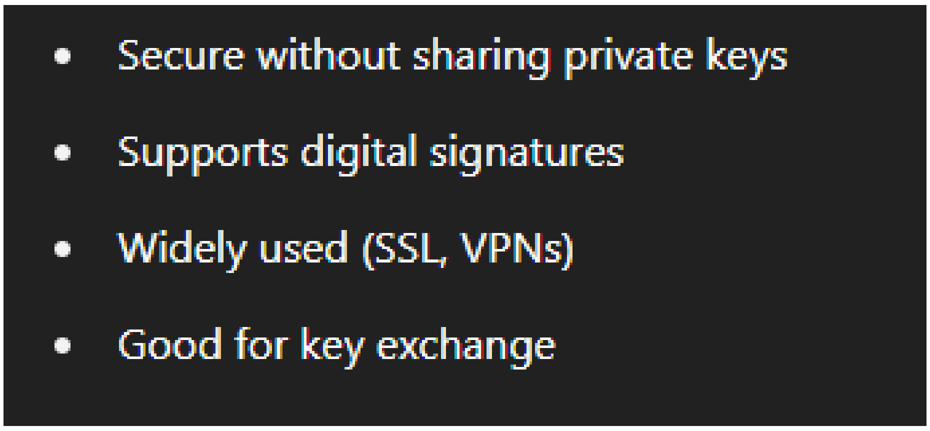
3. Explain the basic principles behind the RSA algorithm's encryption and decryption processes.


- **Key Generation:** Generate two large primes → calculate modulus `n` and exponents `e` (public) and `d` (private).
- **Encryption:** `C = M^e mod n`
- **Decryption:** `M = C^d mod n`

4. What is the key size in RSA, and why is it important for security?


Key size (e.g., **2048 bits**) determines security. Larger keys = stronger security but slower performance.

5. What are the advantages of using the RSA algorithm for secure data transmission and encryption?

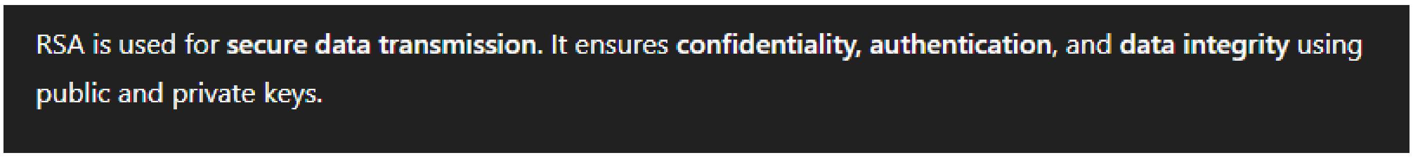

- Secure without sharing private keys
- Supports digital signatures
- Widely used (SSL, VPNs)
- Good for key exchange

Course Title	**NETWORK PROTOCOLS & SECURITY**	**ACADEMIC YEAR: 2024-25**
Course Code(s)	**23EC2210R**	

Experiment #	<TO BE FILLED BY STUDENT>	**Student ID**	<TO BE FILLED BY STUDENT>
Date	<TO BE FILLED BY STUDENT>	**Student Name**	THANOS

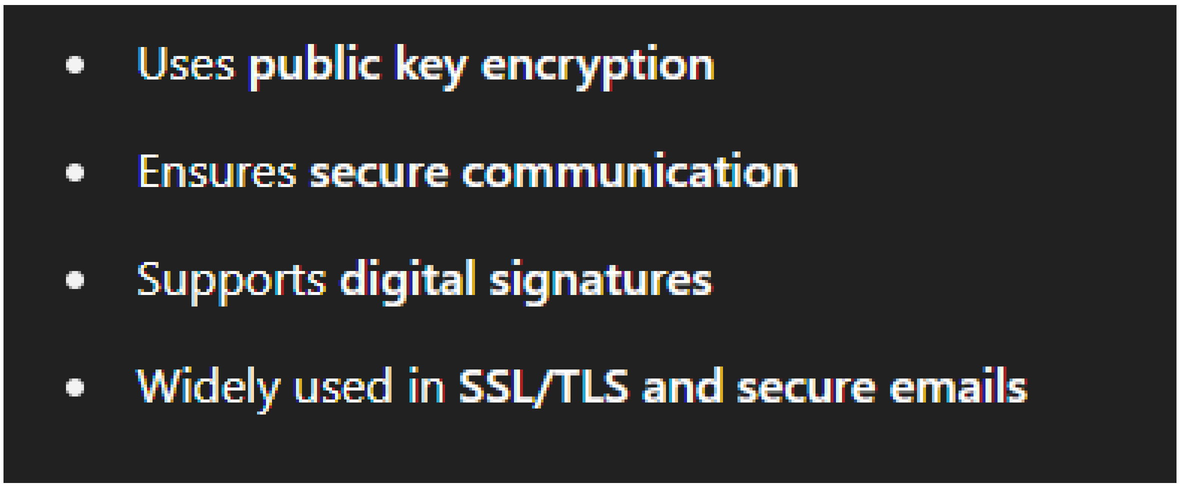
Post Lab Task:

1. Explain the purpose and significance of the RSA algorithm in cryptography.



RSA is used for **secure data transmission.** It ensures **confidentiality, authentication**, and **data integrity** using public and private keys.

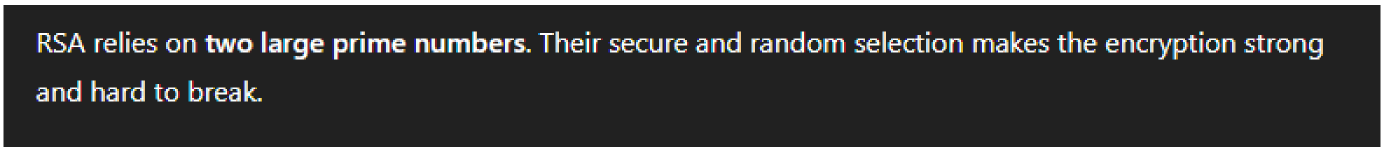
2. What advantages does it offer in terms of secure communication and data encryption?



- Uses **public key encryption**
- Ensures **secure communication**
- Supports **digital signatures**
- Widely used in **SSL/TLS and secure emails**

Experiment #	<TO BE FILLED BY STUDENT>	**Student ID**	<TO BE FILLED BY STUDENT>
Date	<TO BE FILLED BY STUDENT>	**Student Name**	THANOS

3. Discuss the importance of prime number generation and selection in the RSA algorithm.



RSA relies on **two large prime numbers.** Their secure and random selection makes the encryption strong and hard to break.

Evaluator Remark (if Any):	**Marks Secured _____ out of 50**
	Signature of the Evaluator with Date

Course Title	**NETWORK PROTOCOLS & SECURITY**	**ACADEMIC YEAR: 2024-25**
Course Code(s)	**23EC2210R**	