

SRI JAYACHAMARAJENDRA COLLEGE OF ENGINEERING

**A Constituent College of
JSS SCIENCE & TECHNOLOGY UNIVERSITY**



Additive Cipher , Multiplicative Cipher & RSA Algorithm

Mini project report submitted in partial fulfillment of curriculum prescribed for the :
Cryptography and Network Security (20CS552) course for the award of the degree of

**BACHELOR OF ENGINEERING
IN
COMPUTER SCIENCE AND ENGINEERING**

PROJECT REPORT

Submitted by

S.No.	NAME	ROLLNO.	USN
1	Adithya Deepthi Kumar	28	01JST21CS005
2	E Shreyas Herale	35	01JST21CS037
3	Eshwar J	8	01JCE21CS033
4	Harsha N P	10	01JCE21CS038

D-SECTION

Under the Guidance of
Shwethashree G C
Assistant Professor
Dept.of CS & E,
SJCE, JSSSTU, Mysuru

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
November 2023**

SRI JAYACHAMARAJENDRA COLLEGE OF ENGINEERING
A Constituent College of
JSS SCIENCE & TECHNOLOGY UNIVERSITY



CERTIFICATE

This is to certify that the work entitled“ **Additive Cipher, Multiplicative Cipher,RSA Algorithm** ” is a bonafied work carried out by **Adithya Deepthi Kumar, E Shreyas Herale, Eshwar J, Harsha N P** in partial fulfillment of the award of the degree of **Bachelor of Engineering in Computer Science and Engineering of JSS Science and Technology University, Mysuru during the year 2023**. It is certified that all corrections / suggestions indicated during CIE have been incorporated in the report. The mini project report has been approved as it satisfies the academic requirements in respect of mini project work for event 1 prescribed for the Cryptography and Network Security (20CS552) course.

Course in Charge and Guide

Shwethashree G C
Assistant Professor
Dept.of CS & E,
SJCE, JSSSTU, Mysuru

Place: Mysuru

Date : 28/11/2023

ABSTRACT

This report provides a comprehensive exploration of three fundamental cryptographic algorithms: the Additive Cipher, Multiplicative Cipher, and RSA Algorithm. The Additive Cipher involves shifting characters by a fixed amount, while the Multiplicative Cipher employs modular multiplication to achieve encryption. Both ciphers are classical symmetric key algorithms. In contrast, the RSA Algorithm, a widely used asymmetric key algorithm, relies on the mathematical complexity of factoring large prime numbers for secure communication. The report delves into the theoretical foundations, operational mechanisms, and security considerations of each algorithm, practical applications and historical perspectives, offering a comparative analysis of their strengths and weaknesses and their role in modern information security.

TABLE OF CONTENTS

S.No.	Content	Page No.
1.	Introduction	1
1.1	Intoduction to Cryptography	1
1.2	Cipher	2
1.3	Historical Significance	3
1.4	Symmetric key Cipher	5
2	Additive Cipher	6
2.1	Additive Cipher Implementation	7
2.2	Advantages and Disadvantages of Additive Cipher	10
2.3	Applications of Additive Cipher	11
3	Multiplicative Cipher	12
3.1	Multiplicative Cipher Implementation	13
3.2	Advantages and Disadvantages of Multiplicative Cipher	16
3.3	Applications of Multiplicative Cipher	17
4	RSA Algorithm	18
4.1	RSA Algorithm Implementation	21
4.2	Pros, Cons and Application of RSA	24
5	Conclusion	26
6	References	27

RSA ALGORITHM:

RSA (Rivest–Shamir–Adleman) is a public-key cryptosystem, one of the oldest that is widely used for secure data transmission. The initialism "RSA" comes from the surnames of Ron Rivest, Adi Shamir and Leonard Adleman, who publicly described the algorithm in 1977.

In a public-key cryptosystem, the encryption key is public and distinct from the decryption key, which is kept secret (private). An RSA user creates and publishes a public key based on two large prime numbers, along with an auxiliary value. The prime numbers are kept secret. Messages can be encrypted by anyone, via the public key, but can only be decoded by someone who knows the prime numbers.

The security of RSA relies on the practical difficulty of factoring the product of two large prime numbers, the "factoring problem". Breaking RSA encryption is known as the RSA problem.

RSA ALGORITHM IMPLEMENTATION:

RSA algorithm is an asymmetric cryptography algorithm. Asymmetric actually means that it works on two different keys i.e. **Public Key** and **Private Key**. As the name describes that the Public Key is given to everyone and the Private key is kept private.

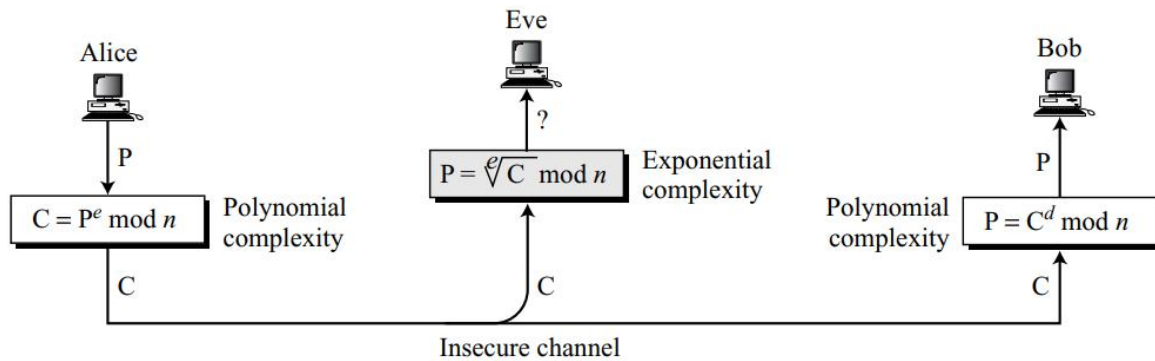
Public key encryption algorithm:

Public Key encryption algorithm is also called the Asymmetric algorithm. Asymmetric algorithms are those algorithms in which sender and receiver use different keys for encryption and decryption. Each sender is assigned a pair of keys:

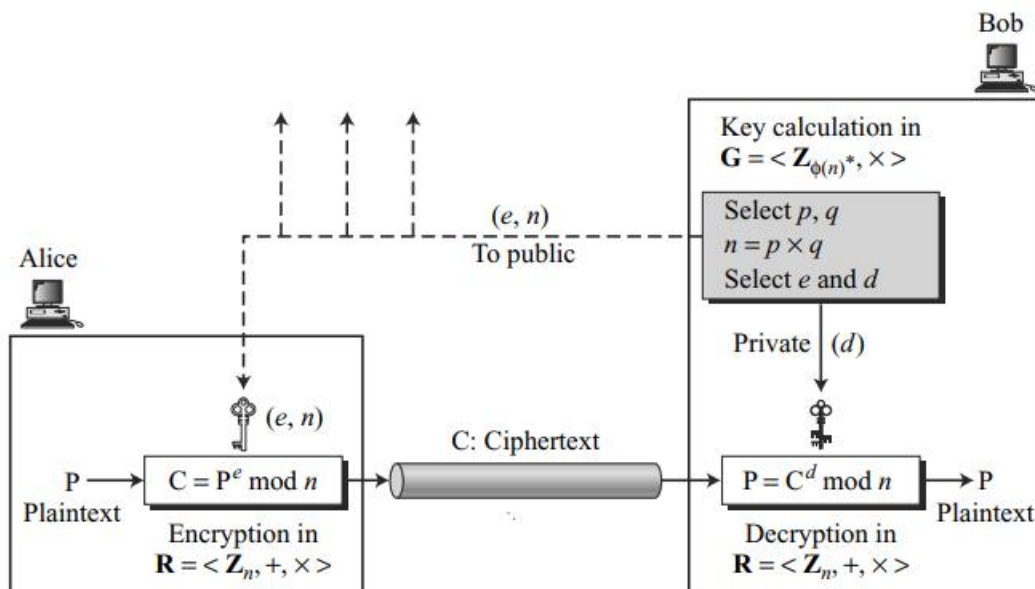
- **Public key**
- **Private key**

The **Public key** is used for encryption, and the **Private Key** is used for decryption. Decryption cannot be done using a public key. The two keys are linked, but the private key cannot be derived from the public key. The public key is well known, but the private key is secret and it is known only to the user who owns the key. It means that everybody can send a message to the user using user's public key. But only the user can decrypt the message using his private key

Complexity of RSA Algorithm



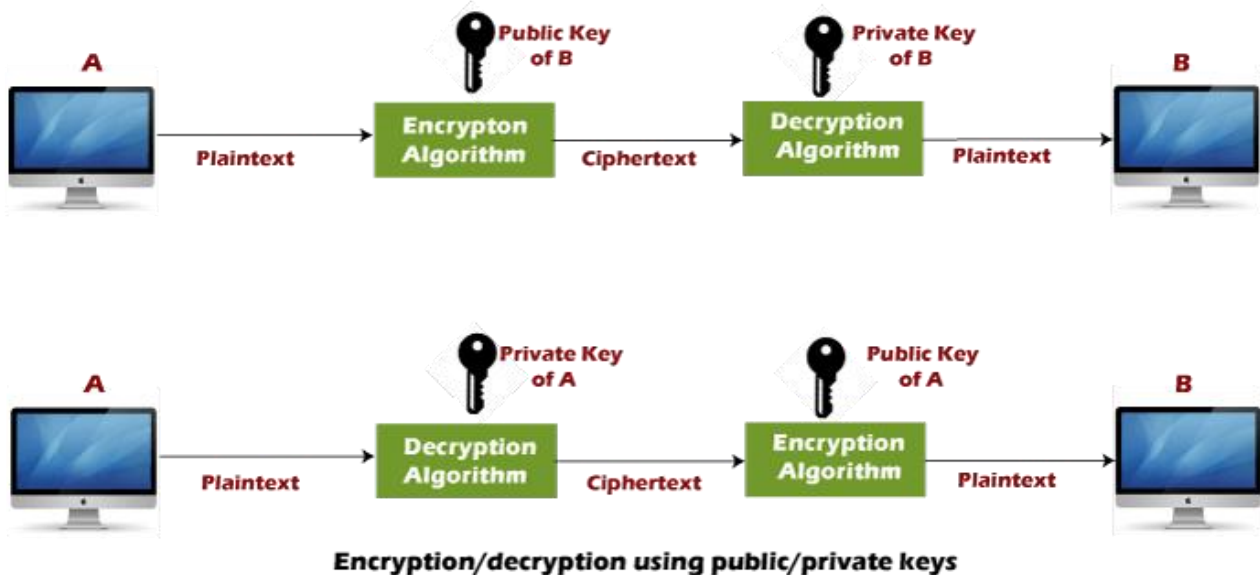
Key Generation, Encryption, Decryption in RSA Algorithm



Attacks on RSA Algorithm

The **RSA algorithm**, while widely utilized for secure communication and encryption, is susceptible to various attacks. **Brute force attacks** aim to systematically test all possible keys and are thwarted by employing large key sizes. **Factorization attacks** exploit the difficulty of factoring the product of two large primes, underscoring the importance of using sufficiently large prime numbers and regularly updating keys. **Timing attacks** and **side-channel attacks** involve analyzing the time taken for cryptographic operations and exploiting information leaked during these operations, respectively. Implementing constant-time algorithms and countermeasures, such as adding noise to side-channel information, helps mitigate these threats. **Chosen plaintext attacks**, **ciphertext-only attacks**, and **common modulus attacks** target gaining information about the plaintext or key, emphasizing the need for robust padding schemes, randomized encryption, and distinct moduli. Additionally, **fault attacks** involve inducing errors in the cryptographic process, mitigated through error-detection mechanisms and redundancy. Staying informed about vulnerabilities, updating cryptographic implementations, and adhering to best practices in key management are crucial for maintaining the security of RSA-based systems.

The Public key algorithm operates in the following manner:



- The data to be sent is encrypted by sender A using the public key of the intended receiver
- B decrypts the received ciphertext using its private key, which is known only to B. B replies to A encrypting its message using A's public key.
- A decrypts the received ciphertext using its private key, which is known only to him.

RSA algorithm uses the following procedure to generate public and private keys:

- Select two large prime numbers, **p** and **q**.
- Multiply these numbers to find **n = p x q**, where **n** is called the modulus for encryption and decryption.
- Choose a number **e** less than **n**, such that **n** is relatively prime to **(p - 1) x (q - 1)**. It means that **e** and **(p - 1) x (q - 1)** have no common factor except 1. Choose "e" such that $1 < e < \phi(n)$, **e** is prime to $\phi(n)$, **gcd (e, $\phi(n)$) = 1**.
- If **n = p x q**, then the public key is **<e, n>**. A plaintext message **M** is encrypted using public key **<e, n>**. To find ciphertext from the plain text following formula is used to get ciphertext C.

$$C = M^e \bmod n$$
 Here, **M** must be less than **n**. A larger message (**>n**) is treated as a concatenation of messages, each of which is encrypted separately.
- To determine the private key, we use the following formula to calculate the **d** such that: **(d x e) mod $\phi(n)$ = 1**
- The private key is **<d, n>**. A ciphertext message **C** is decrypted using private key **<d, n>**. To calculate plain text **m** from the ciphertext **c** following formula is used to get plain text M.

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

RSA ALGORITHM IMPLEMENTATION:

Private and Public Key Generation:



```
1 def is_prime(n):
2     if n < 2:
3         return False
4     for i in range(2, int(n**0.5) + 1):
5         if n % i == 0:
6             return False
7     return True
8
9 def gcd(a, b):
10    while b:
11        a, b = b, a % b
12    return a
13
14 def mod_inverse(a, m):
15    m0, x0, x1 = m, 0, 1
16    while a > 1:
17        q = a // m
18        m, a = a % m, m
19        x0, x1 = x1 - q * x0, x0
20    return x1 + m0 if x1 < 0 else x1
21
22 def generate_keypair(p, q):
23     if not (is_prime(p) and is_prime(q)):
24         raise ValueError("Both numbers must be prime.")
25     elif p == q:
26         raise ValueError("p and q cannot be equal.")
27
28     n = p * q
29     phi = (p - 1) * (q - 1)
30
31     # Choose e such that 1 < e < phi and e is coprime with phi
32     e = random.randrange(2, phi)
33     while gcd(e, phi) != 1:
34         e = random.randrange(2, phi)
35
36     # Compute d, the modular multiplicative inverse of e (mod phi)
37     d = mod_inverse(e, phi)
38
39     return ((e, n), (d, n))
```



Encryption and Decryption Process:

```
1 def encrypt(message, public_key):
2     e, n = public_key
3     cipher = pow(message, e, n)
4     return cipher
5
6 def decrypt(ciphertext, private_key):
7     d, n = private_key
8     plain = pow(ciphertext, d, n)
9     return plain
```

Main Function:

```
1 def main():
2     print("RSA Encryption and Decryption Program")
3
4     while True:
5         print("\nMenu:")
6         print("1. Encrypt")
7         print("2. Decrypt")
8         print("3. Exit")
9
10        choice = input("Enter your choice (1, 2, or 3): ")
11
12        if choice == '1':
13            # Step 1: Input prime numbers p and q
14            p = int(input("Enter a large prime number (p): "))
15            q = int(input("Enter another large prime number (q): "))
16
17            # Generate public and private keys
18            public_key, private_key = generate_keypair(p, q)
19            print(f"\nPublic Key (e, n): {public_key}")
20            print(f"Private Key (d, n): {private_key}")
21
22            # Step 3: Input plaintext
23            plaintext = int(input("\nEnter the plaintext to encrypt: "))
24
25            # Step 4: Encrypt plaintext message using public key <e, n>
26            ciphertext = encrypt(plaintext, public_key)
27            print(f"\nEncrypted Ciphertext: {ciphertext}")
28
29        elif choice == '2':
30            # Step 5: Decrypt ciphertext message using private key <d, n>
31            ciphertext = int(input("Enter the ciphertext to decrypt: "))
32            decrypted_text = decrypt(ciphertext, private_key)
33            print(f"\nDecrypted Plaintext: {decrypted_text}")
34
35        elif choice == '3':
36            print("Exiting the program. Goodbye!")
37            break
38
39        else:
40            print("Invalid choice. Please enter 1, 2, or 3.")
41
42    if __name__ == "__main__":
43        main()
```


Snapshots of Implementation



```
1  RSA Encryption and Decryption Program
2
3  Menu:
4  1. Encrypt
5  2. Decrypt
6  3. Exit
7  Enter your choice (1, 2, or 3): 1
8  Enter a large prime number (p): 7
9  Enter another large prime number (q): 11
10
11 Public Key (e, n): (7, 77)
12 Private Key (d, n): (43, 77)
13
14 Enter the plaintext to encrypt: 9
15
16 Encrypted Ciphertext: 37
17
18 Menu:
19 1. Encrypt
20 2. Decrypt
21 3. Exit
22 Enter your choice (1, 2, or 3): 2
23 Enter the ciphertext to decrypt: 37
24
25 Decrypted Plaintext: 9
26
27 Menu:
28 1. Encrypt
29 2. Decrypt
30 3. Exit
31 Enter your choice (1, 2, or 3): 3
32 Exiting the program. Goodbye!
```

Advantages and Disadvantages of RSA

Advantages of RSA:

1. Security Strength:

- RSA is widely considered secure when used with sufficiently large key sizes. The difficulty of factoring large semiprime numbers forms the basis of its security.

2. Public and Private Keys:

- RSA uses a pair of keys (public and private), allowing for secure communication and digital signatures. The public key can be freely distributed, while the private key remains secret.

3. Versatility:

- RSA is versatile and can be used for various cryptographic applications, including encryption, digital signatures, and key exchange.

4. Standardization:

- RSA has been standardized and widely adopted, making it interoperable across different systems and applications.

5. Mathematical Underpinnings:

- The security of RSA is based on the difficulty of factoring the product of two large prime numbers, a problem believed to be computationally infeasible.

Disadvantages of RSA:

1. Key Size Requirements:

- To maintain security, RSA requires larger key sizes compared to some other encryption algorithms. As computing power increases, longer key lengths may be needed to withstand attacks.

2. Computational Complexity:

- RSA operations, especially key generation and decryption, are computationally intensive. This can be a disadvantage in resource-constrained environments or for devices with limited processing power.

3. Padding Overhead:

- RSA encryption requires padding schemes to avoid certain vulnerabilities, adding overhead to the size of the encrypted message.

4. Vulnerability to Quantum Computers:

- While not an immediate threat, the development of quantum computers could potentially undermine RSA's security, as they could efficiently solve the underlying mathematical problem of factoring large numbers.

5. Key Management:

- The management of keys, especially the secure distribution and storage of private keys, can be challenging in certain scenarios.

6. Performance:

- In some situations, RSA may not be as performant as other encryption algorithms, particularly in scenarios where speed is a critical factor.

In practice, RSA is often used in combination with other cryptographic techniques to address its limitations and provide a more comprehensive security solution.

Applications of RSA (Rivest-Shamir-Adleman):

1. Secure Communication:

- RSA is widely used for securing communication over the internet, including email encryption and securing data transmitted between parties.

2. Digital Signatures:

- RSA is employed in the creation of digital signatures, allowing for the verification of the authenticity and integrity of digital documents and messages.

3. Secure Web Browsing (SSL/TLS):

- RSA is a fundamental component of secure web browsing through protocols like SSL (Secure Sockets Layer) and its successor TLS (Transport Layer Security). It is used for key exchange and ensuring the confidentiality and integrity of data exchanged between web servers and browsers.

4. Secure File Transfer:

- RSA is applied in secure file transfer protocols, ensuring the confidentiality and integrity of files during transmission.

5. Cryptographic Key Exchange:

- RSA is used in key exchange protocols, allowing parties to securely exchange cryptographic keys for use in symmetric key algorithms.

6. VPN (Virtual Private Network):

- RSA is utilized in VPN technologies to establish secure communication channels over public networks.

7. Authentication Tokens:

- RSA is often employed in the generation of cryptographic tokens for user authentication, providing an additional layer of security in systems like two-factor authentication (2FA).

8. Secure Email Communication:

- RSA is used to secure email communication through the implementation of encryption and digital signatures, ensuring the privacy and authenticity of email messages.

9. Secure Shell (SSH) Protocol:

- RSA is a key component in the SSH protocol, providing secure remote access to computer systems.

10. Smart Cards and Secure Hardware Tokens:

- RSA is utilized in smart card technologies and secure hardware tokens, providing a secure means of authentication and authorization.

11. Secure Messaging Apps:

- Some messaging applications use RSA for end-to-end encryption, ensuring that only the intended recipients can decrypt and read the messages.

12. Secure Online Transactions:

- RSA is applied in securing online transactions, including e-commerce and online banking, to protect sensitive information such as credit card details.

The versatility and security of RSA make it a crucial component in various cryptographic applications, contributing to the secure functioning of numerous systems and services in the digital landscape.

CONCLUSION :

In conclusion, this report has provided a comprehensive exploration of three cryptographic algorithms: Additive Cipher, Multiplicative Cipher, and the RSA algorithm. The Additive Cipher demonstrated a simple yet effective method of encryption through the shifting of characters, with its vulnerability lying in susceptibility to frequency analysis. The Multiplicative Cipher, building upon the Additive Cipher, introduced a new layer of complexity by incorporating multiplication, offering enhanced security but still vulnerable to certain attacks.

The RSA algorithm, a cornerstone of modern cryptography, emerged as a robust asymmetric encryption technique. Its strength lies in the difficulty of factoring large semiprime numbers, contributing to its security in secure communication, digital signatures, and various applications. While RSA exhibits versatility and wide adoption, it demands careful consideration of key sizes, computational complexity, and potential future threats from quantum computing.

The comparative analysis of these algorithms underscores the evolution in cryptographic techniques, with each method addressing specific challenges and vulnerabilities. Additive and Multiplicative Ciphers offer historical insights into early

encryption methods, while RSA represents a contemporary and widely implemented solution. As cryptographic landscapes evolve, understanding the strengths and weaknesses of these algorithms is crucial for maintaining secure communication in an ever-changing digital environment.

REFERENCES :

- Cryptography and Network Security By Behrouz A Forouzan
- <https://www.geeksforgeeks.org>
- <https://en.wikipedia.org/wiki/Cryptography>
- <https://www.techtarget.com/searchsecurity/definition/cryptography>
- <https://www.khanacademy.org/computing/computer-science/cryptography>
- <https://brilliant.org/wiki/caesar-cipher/>
- <https://www.javatpoint.com/rsa-encryption-algorithm>
- <https://www.geeksforgeeks.org/rsa-algorithm-cryptography/>

Steps Involved in Encryption and Decryption process :

Additive Cipher:

Plain text: "encrypt this message", Key=15.

Encryption :

Plain Text	Plain Text No.	Addition	Cipher Text No.	Cipher Text
e	4	4+15=19	19	T
n	13	13+15=28	2	C
c	2	2+15=17	17	R
r	17	17+15=32	6	G
y	24	24+15=39	13	N
p	15	15+15=30	4	E
t	19	19+15=34	8	I
t	19	19+15=34	8	I
h	7	7+15=22	22	W
i	8	8+15=23	23	X
s	18	18+15=33	7	H
m	12	12+15=27	1	B
e	4	4+15=19	19	T
s	18	18+15=33	7	H
s	18	18+15=33	7	H
a	0	0+15=15	15	P
g	6	6+15=21	21	V
e	4	4+15=19	19	T

Decryption:

Cipher Text	Cipher Text No.	Subtraction	Plain Text No.	Plain Text
T	19	$19-15=4$	4	e
C	2	$2-15=-13$	13	n
R	17	$17-15=2$	2	c
G	6	$6-15=-9$	17	r
N	13	$13-15=-2$	24	y
E	4	$4-15=-11$	15	p
I	8	$8-15=-7$	19	t
I	8	$8-15=-7$	19	t
W	22	$22-15=7$	7	h
X	23	$23-15=8$	8	i
H	7	$7-15=-8$	18	s
B	1	$1-15=-14$	12	m
T	19	$19-15=4$	4	e
H	7	$7-15=-8$	18	s
H	7	$7-15=-8$	18	s
P	15	$15-15=0$	0	a
V	21	$21-15=6$	6	g
T	19	$19-15=4$	4	e

Plain text : “encrypt this message”

Cipher text : “TCRGNEIIWXHBTTHHPVT”

Multiplicative Cipher:

Plain Text : ”top secret message”, key=17

Encryption :

Plain Text	Plain text No	Multiplication	Cipher Text No	Cipher Text
t	19	323	11	L
o	14	238	4	E
p	15	255	21	V
s	18	306	20	U
e	4	68	16	Q
c	2	34	8	I
r	17	289	3	D
e	4	68	16	Q
t	19	323	11	L
m	12	204	22	W
e	4	68	16	Q
s	18	306	20	U
s	18	306	20	U
a	0	0	0	A
g	6	102	24	Y
e	4	68	16	Q

Decryption:

K^{-1} in Z_{26} , that is 17^{-1} in 26

K^{-1} obtained by Euclidean Algorithm

q	r1	r2	r	t1	t2	t
1	26	17	9	0	1	-1
1	17	9	8	1	-1	2
1	9	8	1	-1	2	-3
8	8	1	0	2	-3	26
	$\Rightarrow 1$			$\Rightarrow -3$		

17^{-1} in $Z_{26} = -3 + 26 = \mathbf{23}$

Cipher Text	Cipher Text No	Multiplication	Plain Text No	Plain Text
L	11	253	19	t
E	4	92	14	o
V	21	483	15	p
U	20	460	18	s
Q	16	368	4	e
I	8	184	2	c
D	3	69	7	r
Q	16	368	4	e
L	11	253	19	t
W	22	506	12	m
Q	16	368	4	e
U	20	460	18	s
U	20	460	18	s
A	0	0	0	a
Y	24	552	6	g
Q	16	368	4	e

Plain text : “top secret message”

Cipher text : “LEVUQIDQLWQUUAYQ”

RSA Encryption and Decryption:

This example shows how we can encrypt plaintext '9' using the RSA public-key encryption algorithm. This example uses prime numbers 7 and 11 to generate the public and private keys.

Explanation:

Step 1: Select two large prime numbers, **p**, and **q**. $p = 7$ $q = 11$

Step 2: Multiply these numbers to find $n = p \times q$, where **n** is called the modulus for encryption and decryption.

First, we calculate $n = p \times q$; $n = 7 \times 11$; $n = 77$

Step 3: Choose a number **e** less than **n**, such that **n** is relatively prime to

$(p - 1) \times (q - 1)$. It means that **e** and $(p - 1) \times (q - 1)$ have no common factor except 1. Choose "e" such that $1 < e < \phi(n)$, e is prime to $\phi(n)$, $\gcd(e, \phi(n)) = 1$.

Second, we calculate : $\phi(n) = (p - 1) \times (q - 1)$

$\phi(n) = (7 - 1) \times (11 - 1)$; $\phi(n) = 6 \times 10$; $\phi(n) = 60$

Let us now choose relative prime e of 60 as 7.

Thus the public key is $\langle e, n \rangle = (7, 77)$

Step 4: A plaintext message **m** is encrypted using public key $\langle e, n \rangle$. To find ciphertext from the plain text following formula is used to get ciphertext C.

To find ciphertext from the plain text following formula is used to get ciphertext C.

$C = M^e \bmod n$; $C = 9^7 \bmod 77$; $C = 37$

Step 5: The private key is $\langle d, n \rangle$. To determine the private key, we use the following formula d such that:

$d \times e \bmod \{(p - 1) \times (q - 1)\} = 1$; $7 \times d \bmod 60 = 1$, which gives $d = 43$

The private key is $\langle d, n \rangle = (43, 77)$

Step 6: A ciphertext message **c** is decrypted using private key $\langle d, n \rangle$. To calculate plain text **m** from the ciphertext c following formula is used to get plain text m.

$M = C^d \bmod n$; $m = 37^{43} \bmod 77$; $m = 9$

In this example, Plain text = 9 and the ciphertext = 37.