

Roll. No. A016	Name: Varun Khadayate
Class B.Tech CsBs	Batch: 1
Date of Experiment: 10-09-2022	Subject: Cryptology

## Aim

To implement RSA Algorithm.

## Theory

The pioneering paper by Diffie and Hellman [DIFF76b] introduced a new approach to cryptography and, in effect, challenged cryptologists to come up with a cryptographic algorithm that met the requirements for public-key systems. A number of algorithms have been proposed for public-key cryptography. Some of these, though initially promising, turned out to be breakable.<sup>4</sup>

One of the first successful responses to the challenge was developed in 1977 by Ron Rivest, Adi Shamir, and Len Adleman at MIT and first published in 1978 [RIVE78].<sup>5</sup> The Rivest-Shamir-Adleman (RSA) scheme has since that time reigned supreme as the most widely accepted and implemented general-purpose approach to public-key encryption.

The **RSA** scheme is a cipher in which the plaintext and ciphertext are integers between 0 and  $n - 1$  for some  $n$ . A typical size for  $n$  is 1024 bits, or 309 decimal digits. That is,  $n$  is less than  $2^{1024}$ . We examine RSA in this section in some detail, beginning with an explanation of the algorithm. Then we examine some of the computational and cryptanalytical implications of RSA.

RSA makes use of an expression with exponentials. Plaintext is encrypted in blocks, with each block having a binary value less than some number  $n$ . That is, the block

size must be less than or equal to  $\log_2(n) + 1$ ; in practice, the block size is  $i$  bits, where  $2^i \leq n < 2^{i+1}$ . Encryption and decryption are of the following form, for some plaintext block  $M$  and ciphertext block  $C$ .

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

$$M = C^d \bmod n = (M^e)^d \bmod n = M^{ed} \bmod n$$

Both sender and receiver must know the value of  $n$ . The sender knows the value of  $e$ , and only the receiver knows the value of  $d$ . Thus, this is a public-key encryption algorithm with a public key of  $PU = \{e, n\}$  and a private key of  $PR = \{d, n\}$ . For this algorithm to be satisfactory for public-key encryption, the following requirements must be met.

1. It is possible to find values of  $e$ ,  $d$ , and  $n$  such that  $M^{ed} \bmod n = M$  for all  $M \in \mathbb{Z}_n$ .
2. It is relatively easy to calculate  $M^e \bmod n$  and  $C^d \bmod n$  for all values of  $M \in \mathbb{Z}_n$ .
3. It is infeasible to determine  $d$  given  $e$  and  $n$ .

For now, we focus on the first requirement and consider the other questions later. We need to find a relationship of the form

$$M^{ed} \bmod n = M$$

The preceding relationship holds if  $e$  and  $d$  are multiplicative inverses modulo  $\phi(n)$ , where  $\phi(n)$  is the Euler totient function. It is shown in Chapter 8 that for  $p, q$  prime,  $\phi(pq) = (p - 1)(q - 1)$ . The relationship between  $e$  and  $d$  can be expressed as

$$ed \bmod \phi(n) = 1 \quad (9.1)$$

This is equivalent to saying

$$ed \equiv 1 \pmod{\phi(n)}$$

$$d \equiv e^{-1} \pmod{\phi(n)}$$

That is,  $e$  and  $d$  are multiplicative inverses mod  $\phi(n)$ . Note that, according to the rules of modular arithmetic, this is true only if  $d$  (and therefore  $e$ ) is relatively prime to  $\phi(n)$ . Equivalently,  $\gcd(\phi(n), d) = 1$ . See Appendix R for a proof that Equation (9.1) satisfies the requirement for RSA.

We are now ready to state the RSA scheme. The ingredients are the following:

$p, q$ , two prime numbers	(private, chosen)
$n = pq$	(public, calculated)
$e$ , with $\gcd(\phi(n), e) = 1$ ; $1 < e < \phi(n)$	(public, chosen)
$d \equiv e^{-1} \pmod{\phi(n)}$	(private, calculated)

The private key consists of  $\{d, n\}$  and the public key consists of  $\{e, n\}$ . Suppose that user A has published its public key and that user B wishes to send the message  $M$  to A. Then B calculates  $C = M^e \pmod{n}$  and transmits  $C$ . On receipt of this cipher-text, user A decrypts by calculating  $M = C^d \pmod{n}$ .

Figure 9.5 summarizes the RSA algorithm. It corresponds to Figure 9.1a: Alice generates a public/private key pair; Bob encrypts using Alice's public key; and Alice decrypts using her private key. An example from [SING99] is shown in Figure 9.6. For this example, the keys were generated as follows.

1. Select two prime numbers,  $p = 17$  and  $q = 11$ .
2. Calculate  $n = pq = 17 * 11 = 187$ .
3. Calculate  $\phi(n) = (p - 1)(q - 1) = 16 * 10 = 160$ .
4. Select  $e$  such that  $e$  is relatively prime to  $\phi(n) = 160$  and less than  $\phi(n)$ ; we choose  $e = 7$ .
5. Determine  $d$  such that  $de \equiv 1 \pmod{160}$  and  $d < 160$ . The correct value is  $d = 23$ , because  $23 * 7 = 161 = (1 * 160) + 1$ ;  $d$  can be calculated using the extended Euclid's algorithm.

## Code

```
p = int(input("Enter prime no. (p): "))
q = int(input("Enter prime no. (q): "))

n = p*q
print(f"n: {n}")

tot = (p-1)*(q-1)
print(f"Totient of n: {tot}")

print("Note: Value of E is  $1 < E < \text{totient of } n$  & is not a factor of totient of n")
E = int(input("Enter (E): "))

i = 1
while True:
    if(i*E % tot == 1):
        D = i
        break
    i += 1
print(f"D: {D}")

pt = 10

ct = pow(pt, E) % n
print(f"Cipher Text: {ct}")

pt = pow(ct, D) % n
print(f"Plain Text: {pt}")
```

## Output

```
Enter prime no. (p): 3
Enter prime no. (q): 7
n: 21
Totient of n: 12
Note: Value of E is  $1 < E < \text{totient of } n$  & is not a factor of totient of n
Enter (E): 5
D: 5
Cipher Text: 19
Plain Text: 10
```

## Conclusion

Hence, we were able to perform RSA Algorithm.