Batch: A3 Roll No: 16010121045

Experiment No.

Title: Implementation of Symmetric key cryptography algorithm (DES/AES)

Objective:

To write a program implementing RSA Digital Signature showing the application of RSA Algorithm for various security services like confidentiality, authentication, signature, non-repudiation and integrity

Expected Outcome of Experiment:

CO	Outcome
CO3	Comprehend cryptographic hash functions, Message Authentication Codes and Digital Signatures for Authentication

Books/ Journals/ Websites referred:

Abstract:-

(*Asymmetric key cryptography*)

Related theory:
Digital signatures, RSA algorithm, RSA signing and verifying process

Program:

```
import math
keyspace="ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz0123456789!@#$%^&*()-=_+"
def publicKey(p,q):
    temp=(p-1)*(q-1)
    for i in range(temp-1,2,-1):
        if(math.gcd(temp,i)==1):
            return i

def privateKey(e,phin):
    for d in range(1,phin):
        if((d*e)%phin==1):
            return d

p=int(input("Enter value of p (Prime number): "))
q=int(input("Enter value of q (Prime number): "))
n=p*q
phin=(p-1)*(q-1)
e=publicKey(p,q)
print("Public key <e,n> = <%d,%d> "%(e,n))
d=privateKey(e,phin)
print("Private key <d,n> = <%d,%d> "%(d,n))
plaintext=input("Enter plain text: ")
print("Encrypted text: ",end="")
encTxt=[]
for i in plaintext:
    x=int(keyspace.find(i))
    encTxt.append(((x+2)**e)%n)
```

```
print(encTxt)
decTxt=""
for i in encTxt:
    decTxt+=keyspace[((i**d)%n)-2]
print(decTxt)
```

Output Screenshots:

```
Enter value of p (Prime number): 53
Enter value of q (Prime number): 59
Public key <e,n> = <3015,3127>
Private key <d,n> = <3015,3127>
Enter plain text: Hey There this is Pargat!
Encrypted text: [695, 2748, 2173, 2792, 1638, 2519, 2748, 68, 2748, 2792, 2671, 2519, 2958, 998, 2792, 2958, 998, 2792, 184, 647, 68, 2859, 6
47, 2671, 1780]
Hey There this is Pargat!
pargat@Router AC %
```

```
Enter value of p (Prime number): 13
Enter value of q (Prime number): 11
Public key <e,n> = <119,143>
Private key <d,n> = <119,143>
Enter plain text: This is a test
Encrypted text: [109, 4, 58, 70, 46, 58, 70, 46, 74, 46, 3, 132, 70, 3]
This is a test
pargat@Router AC %
```

```
Enter value of p (Prime number): 1009
Enter value of q (Prime number): 1013
Public key <e,n> = <1020095,1022117>
Private key <d,n> = <1020095,1022117>
Enter plain text: Hello Madam!
Encrypted text: [794980, 123893, 689929, 689929, 142621, 401546, 803092, 422945, 606882, 422945, 473664, 896318]
Hello Madam!
pargat@Router AC %
```

Conclusion:-

In this experiment, we learnt about the RSA algorithm and how to implement it in the real world. Successfully implemented the given task with detailed steps shown in the output.

Postlab questions:

1. **Comment on strengths and weaknesses of asymmetric key cryptosystem.**

   Strengths:

   • In asymmetric or public key, cryptography there is no need for exchanging keys, thus eliminating the key distribution problem.

   • The primary advantage of public-key cryptography is increased security: the private keys do not ever need to be transmitted or revealed to anyone.

   • Can provide digital signatures that can be repudiated.

   Weakness:

   - A disadvantage of using public-key cryptography for encryption is speed there are popular secret-key encryption methods which are significantly faster than any currently available public-key encryption method.
   - If a key is leaked, the protocol is destroyed.
   - It needs a whole separate secure protocol for sharing the final key.

2. **Discuss strengths and weaknesses of RSA, comment on solutions over the lacunas.**

Strengths

- RSA is stronger than any other symmetric key algorithm.
- If initial prime number values are large, then near to impossible to preform brute force attacks.
- RSA has overcome the weakness of symmetric algorithm i.e. authenticity and confidentiality.

Weakness

- RSA has too much computation.
- If a key is leaked, the protocol is destroyed.
- It needs a whole separate secure protocol for sharing the final key.
- Is very resource and hardware intensive.