T.Y.B.Tech (CSE)

Information Security

Lab Assignment No – B1

Name: Aniruddha Shende

Roll number: PE04

Batch: E1

Panel: E

PE04 Aniruddha Shende

Name :- Aniruddha Arun Shende
Roll no :- PE-04
Batch :- E1
Panel :- E

Subject :- Information Security

Lab Assignment B1

LAB Assignment B1 : To program asymmetric key cryptography such as RSA cryptography using JAVA API, Python, C++ API.

Aim: Implement a programming code for asymmetric key cryptography such as RSA cryptography using JAVA API, Python or C++ API

Objectives:

Public key cryptography is used as a method of ~~assumi~~ assuring the confidentiality, authenticity & non-repudiation of electronic communications & data storage.

Theory:-

The RSA algorithm is an asymmetric cryptography algorithm; this means that it uses a public key & private key.

As their names suggest, a public key is shared publicly, while a private key is secret & must not be shared with anyone.

PE04 Aniruddha Shende

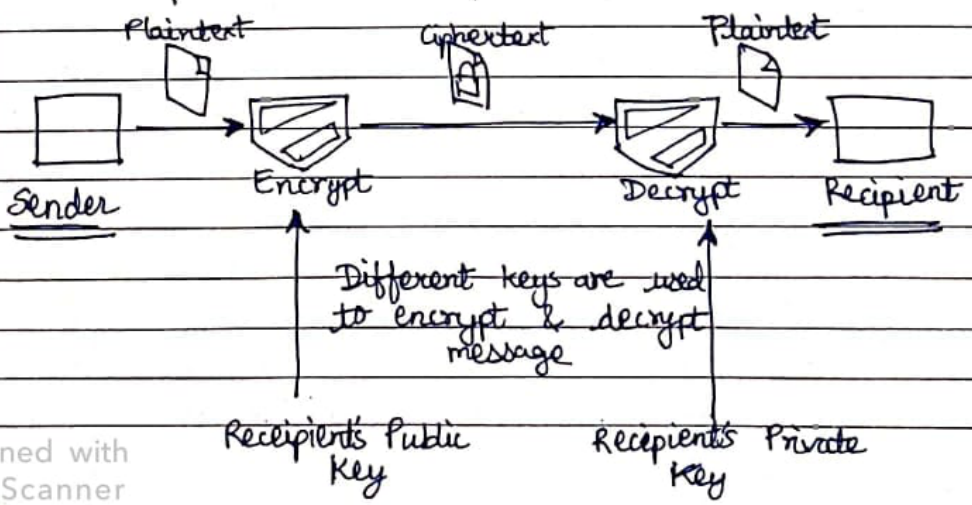
① Generating the keys

1. Select two large prime number, x & y. The prime numbers need to be large so that they will be difficult for someone to figure out.
2. Calculate $n = x \times y$
3. Calculate the $\phi(n) = (x-1)(y-1)$
4. Select an integer e, such that e is co-prime to $\phi(n)$ & $1 < e < \phi(n)$. The pair of numbers (n, e) makes up the public key.
5. Then calculate d such that $(d * e)[\text{mod } (p-1)(q-1)] = 1$

② ~~Encryption~~

~~1. Publish their~~

6. Publish their encryption key: $PU = \{e, n\}$
7. Keep secret private description key: $PR = \{d, n\}$
8. To encrypt a message M the sender:
   Computes ciphertext: $C = M^c \bmod n$, $0 \leq M < n$
9. To decrypt the ciphertext C the owner:
   Computes : $M = C^d \bmod n$



PE04 Aniruddha Shende

FAQ's:-

1. Explain Java support of library for RSA cryptography.

Ans:1. We can use the javax.crypto & java.security packages. It is in the Java standard platform. After using the two packages we can successfully run the RSA code.

2. Explain python support of library for RSA cryptography.

Ans:2. We can import the rsa library, it supports encryption & decryption, signing & verifying signatures & the key generation according to #PKCS#1 version 1.5.
We can use the functionalities by including the following code line
import rsa.

3. Explain various third party library support of security libraries.

Ans:3 We can use the following Python library for security.

① Nmap :- It is an open-source tool analyzer that is widely used in cyber security.

② Scapy :- Scapy is a sophisticated Python package that may be used for scanning, probing, unit testing, etc.

③ Cryptography :- It contains high-level recipies & reduced gateways to popular cryptographic methods, including ciphers, message digests & key derivation algorithms.

PE04 Aniruddha Shende

4. Explain cryptopp support for cryptography.

Ans:4. Crypto++ is a free & open-source C++ class library of cryptographic algorithms & schemes. It is also known as CryptoPP, libcrypto++ & libcryptopp.
It is widely used in academia, student projects, open source & non-commercial projects, etc.

5. What is base 64 coding & decoding?

Ans:5. Base 64 is an encoding & decoding technique used to convert binary data to an American Standard for Information Interchange (ASCII) text format. It is also known as Base 64 Content-Transfer-Encoding.

6. The reasons which specify why it is difficulty to hack RSA cipher are?

Ans:6. It is difficult to hack RSA cipher because:
(a) Brute force attack would not work as there are too many possible keys to work through.
(b) Dictionary attack will not work in RSA algorithm as the keys are numeric & does not include any characters in it.

Conclusion:- Hence we have successfully implemented a symmetric key cryptography program.

Program Code:

```
#Name : Aniruddha Shende
#Roll No : PE04
#Batch : E1
#Panel : E

from decimal import Decimal

def gcd(m,n):
    if n==0:
        return m
    else:
        return gcd(n,m%n)

#input variables
print("")
p=int(input("Enter 1st prime number : "))
if p > 1:
    for i in range(2, int(p/2)+1):
        if (p % i) == 0:
            print(p, "is not a prime number")
            quit()

q=int(input("Enter 2nd prime number : "))
if q > 1:
    for i in range(2, int(q/2)+1):
        if (q % i) == 0:
            print(q, "is not a prime number")
            quit()
no = int(input(("Input : ")))

#calculate n
n = (p*q)
```

```
#calculate phiN
phiN = (p-1)*(q-1)

#calculate K
for k in range(2,phiN):
    if gcd(k,phiN)== 1:
        break

for i in range(1,10):
    x = 1 + i*phiN
    if x % k == 0:
        d = int(x/k)
        break

local_cipher = Decimal(0)
local_cipher =pow(no,k)
ct = local_cipher % n

decrypt_t = Decimal(0)
decrypt_t= pow(ct,d)
dt = decrypt_t % n

print('e = '+str(k))
print('d = '+str(d))
print('Cipher text = '+str(ct))
print('Decrypted text = '+str(dt))
```

Output:

```
ani@Aniruddhas-MacBook-Pro RSA % python -u "/Users/ani/Desktop/Tri-8/IS LAbs/RSA/hehe2.py"

Enter 1st prime number : 11
Enter 2nd prime number : 13
Input : 54
e = 7
d = 103
Cipher text = 76
Decrypted text = 54
ani@Aniruddhas-MacBook-Pro RSA %
```