International Journal of Trend in Scientific Research and Development (IJTSRD)
Volume 3 Issue 5, August 2019 Available Online: www.ijtsrd.com e-ISSN: 2456 – 6470

A Study of RSA Algorithm in Cryptography

Soe Moe Myint[1], Moe Moe Myint[2], Aye Aye Cho[3]

[1,2]Lecturer, [3]Associate Professor
[1]Faculty of Computer Systems and Technologies, University of Computer Studies, Pathein, Myanmar
[2]Information Technology Support and Maintenance, University of Computer Studies, Pathein, Myanmar
[3]Faculty of Computer Science, University of Computer Studies, Hinthada, Myanmar

How to cite this paper: Soe Moe Myint | Moe Moe Myint | Aye Aye Cho "A Study of RSA Algorithm in Cryptography" Published in International Journal of Trend in Scientific Research and Development (ijtsrd), ISSN: 2456-6470, Volume-3 | Issue-5, August 2019, pp.1670-1674, **https://doi.org/10.31142/ijtsrd26749**

IJTSRD26749

ABSTRACT

RSA (Rivest–Shamir–Adleman) is an algorithm used by modern computers to encrypt and decrypt messages. The purpose of the paper is how to produce two different keys. This is also called public key cryptography, because one of the keys can be given to anyone. In this paper also represent how to separate unwanted character by using Linux command.

KEYWORDS: *RSA algorithm, private key, public key*

INTRODUCTION

Cryptography technique is one of the principal means to protect information security. Not only has it to ensure the information confidential, but also provides digital signature, authentication, secret sub-storage, system security and other functions. RSA is one of the first public-key cryptosystems and is widely used for secure data transmission.

BACKGROUND THEORY

RSA is an asymmetric cryptographic algorithm. Asymmetric means that there are two different keys. This is also called public key cryptography, because one of the keys can be given to anyone. The other key must be kept private. The algorithm is based on the fact that finding the factors of a large composite number is difficult: when the integers are prime numbers, the problem is called prime factorization. It is also a key pair (public and private key) generator. [1]

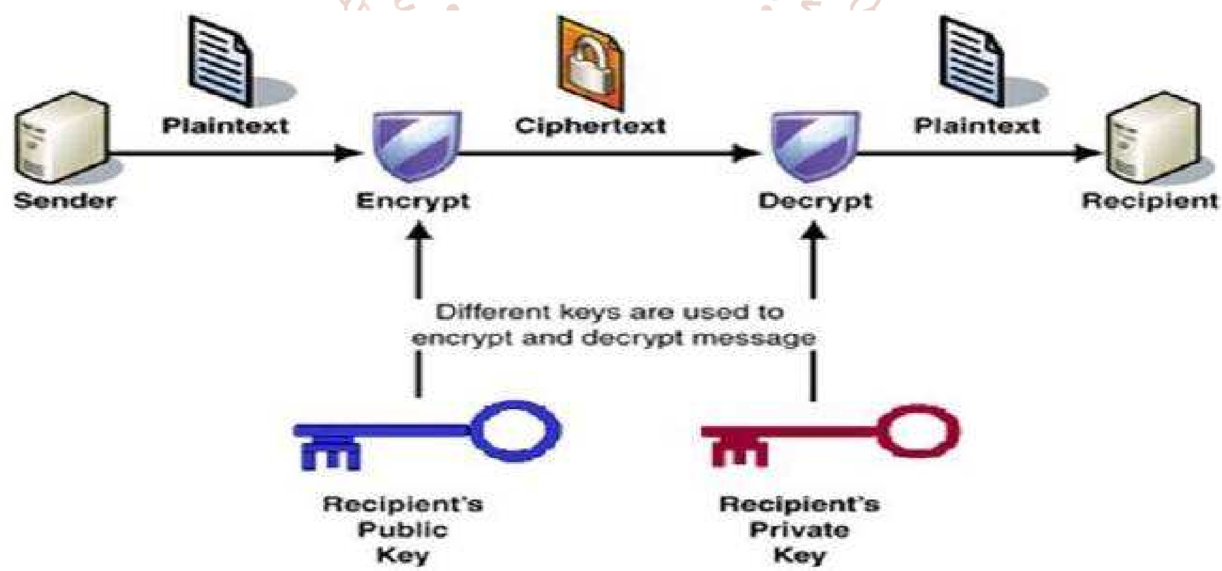


Fig 1 RSA algorithm

1. Getting Plain Text and Public Key

Step by step procedure by using public key

1.1 Using Openssl Command

Open Linux terminal under public.key folder and enter openssl command.

Command: openssl rsa -noout -text -inform PEM -in public.key

```
root@kali:~/Desktop/CT504/crypto100# openssl rsa -noout -text -inform PEM -in public.key -pubin
Public-Key: (2070 bit)
Modulus:
    25:b1:8b:f5:f3:89:09:7d:17:23:78:66:bb:51:cf:
    f8:de:92:24:53:74:9e:bc:40:3b:09:95:c9:7c:0e:
    38:6d:46:c1:61:ca:df:f7:7c:69:86:0d:ae:47:91:
    c2:14:cf:84:87:aa:aa:9f:26:e9:20:a9:77:83:49:
    06:03:8a:ef:b5:c3:08:27:df:cf:3f:c9:e9:76:95:
    44:f9:4e:07:cd:fe:08:72:03:9a:3a:62:62:11:66:
    78:b2:61:fb:2d:6b:9d:32:53:9e:92:a1:53:b3:67:
    56:29:ba:b3:94:2e:7d:35:e3:0f:7e:ef:5a:bf:1c:
    50:d7:97:d0:cc:88:e1:bd:cc:fd:1a:12:ea:6f:7e:
    f7:5c:37:27:db:df:2e:78:0f:34:28:ae:8f:7a:4f:
    b7:a8:9f:18:4a:36:50:32:b1:53:f8:42:5e:84:57:
    50:eb:2b:7a:bc:02:dc:15:ce:02:07:50:7a:a9:50:
    86:3b:b8:48:0a:78:02:8d:d6:29:79:94:4d:6c:63:
    3f:af:a1:03:e4:db:28:ce:87:f5:a0:c6:ed:4a:2f:
    26:64:42:7f:56:5c:77:81:ab:61:91:45:6d:97:1c:
    7f:fa:39:52:72:37:4c:ec:01:55:e5:f9:11:89:db:
    74:2e:4c:28:b0:3a:0f:a1:1c:ff:b0:31:73:d2:a4:
    cc:e6:ae:53
Exponent: 65537 (0x10001)
```

1.2 Removing unwanted characters

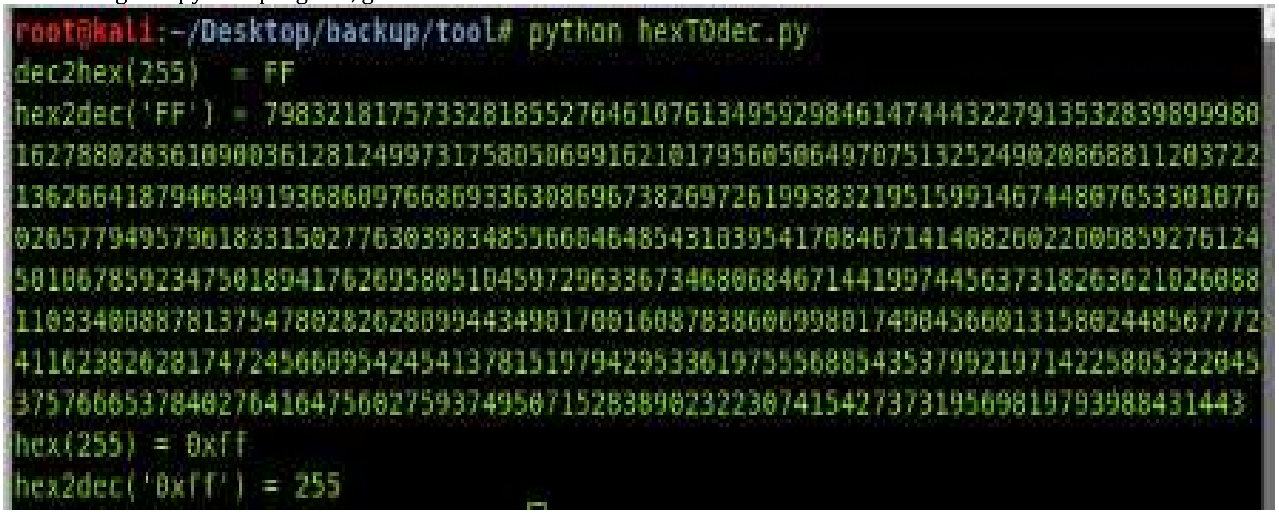
We get hexadecimal characters by using openssl command. But these characters are not real hexadecimal value. We remove unwanted characters and get the real hexadecimal value.

```
25b18bf5f389097d17237866bb51cf
f8de922453749ebc403b0995c97c0e
386d46c161cadff77c69860dae4791
c214cf8487aaaa9f26e920a9778349
06038aefb5c30827dfcf3fc9e97695
44f94e07cdfe0872039a3a62621166
78b261fb2d6b9d32539e92a153b367
5629bab3942e7d35e30f7eef5abf1c
50d797d0cc88e1bdccfd1a12ea6f7e
f75c3727dbdf2e780f3428ae8f7a4f
b7a89f184a365032b153f8425e8457
50eb2b7abc02dc15ce0207507aa950
863bb8480a78028dd62979944d6c63
3fafa103e4db28ce87f5a0c6ed4a2f
2664427f565c7781ab6191456d971c
7ffa395272374cec0155e5f91189db
742e4c28b03a0fa11cffb03173d2a4
cce6ae53
```

1.3 Converting Hexadecimal to Decimal by using Python Program

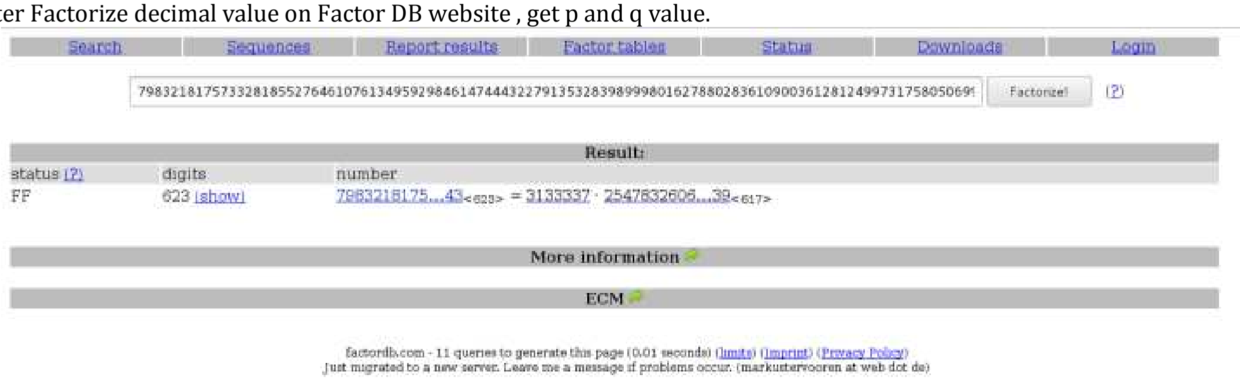
```python
def dec2hex(n):
    """return the hexadecimal string representation of integer n"""
    return "%X" % n

def hex2dec(s):
    """return the integer value of a hexadecimal string s"""
    return int(s, 16)

print "dec2hex(255)  =", dec2hex(255)    # FF
print "hex2dec('FF') =", hex2dec
('25b18bf5f389097d17237866bb51cff8de922453749ebc403b0995c97c0e386d46c161cadff77c69860dae4791c214cf8487aaaa9f26e920a977834906038aefb5c30827dfcf3fc9e9769

print "hex(255) =", hex(255)              # 0xff
print "hex2dec('0xff') =", hex2dec('0xff')  # 255
```

After running this python program, get decimal value



1.3 Factorization by using Factor DB Website

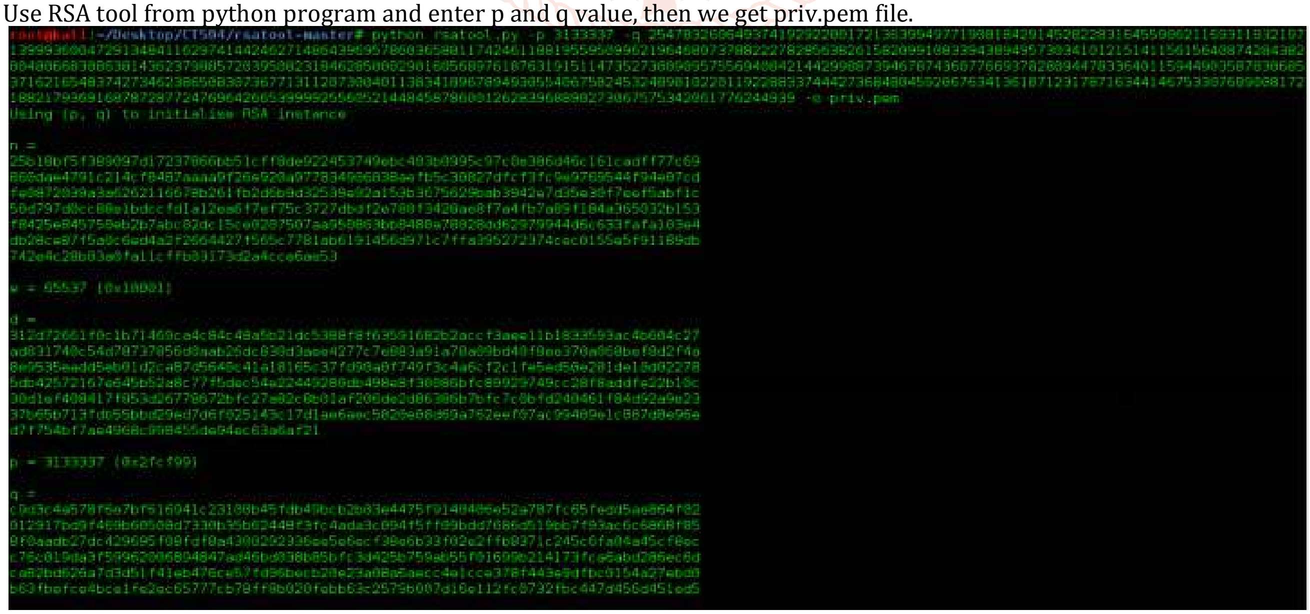
After Factorize decimal value on Factor DB website , get p and q value.



According to the result:
7983218175…43 = 3133337 x 2547832606…39
Here value p is 3133337 and q is 2547832606…39.

1.4 Using RSA tool Python Program

Use RSA tool from python program and enter p and q value, then we get priv.pem file.



Get private key file.


```
-----BEGIN RSA PRIVATE KEY-----
MIIELQIBAAKCAQMlsYv184kJfRcjeGa7Uc/43pIkU3SevEA7CZXJfA44bUbBYcrf93xphg2uR5HC
FM+Eh6qqnybpIKl3g0kGA4rvtcMIJ9/PP8npdpVE+U4Hzf4Icg0a0mJiEWZ4smH7LWudMl0ekqFT
s2dWKbqzlC59NeMPfu9avxxQ15fQzIjhvcz9GhLqb373XDcn298ueA80KK6Pek+3qJ8YSjZQMrFT
+EJehFdQ6yt6vALcFc4CB1B6qVCG07hICngCjdYpeZRNbGM/r6ED5Nsozof1oMbtSi8mZEJ/Vlx3
gathkUVtlxx/+jlScjdM7AFV5fkRidt0LkwosDoPoRz/sDFz0qTM5q5TAgMBAAECggECMS1yZh8M
G3FGnKTITEilsh3FOI+PY1kWgrKszzruEbGDNZOsS2BMJ62DF0DFTXhzeFbQqrJtyDDTruQnfH6I
OpGnigm9QPjuNwoGi++NL0qOlTXq3V6wHSyofVZAxBoYFlw3/ZCg90nzxKbPLB/l7VDigd4Q0CJ4
XbQlchZ+ZFtSqMd/XexU4iRJKA20mOjzAIa/yJkpdJzCj4rd/iKxDDDR70CEF/hT0md4Zyv8J6gs
iwGvIG3i2GOGt7/HwL/SQEYfhNkqniM3tltxP9tVu9Ke19bwJRQ8F9GuauxYIOCNaadi7vB6yZQJ
4cCH2Olu1/dUv3rkloyZhFXelOxjpq8hAgMvz5kCggEBAMnTxKV49ue/YWlBwjEAtF/bSbyysD5E
dfkUBAblKnh/xl/t1a6GTwIBKRe9n0abYFCNczCzW2JEjz/EraPAlPX/Cb3XaG1Rm7f50sbGho+F
jwqtsn3EKWlfCP34pDACkjNu5ebs845rM/AuL/uDccJFxvoEpFz47MdsAZ2j9ZliAGiUhHrUa9A4
uFv8PUJbdZq1XwFpmyFBc/ymq9KG7G3Kgr1ian09UfQetHbOV/2Wvssg4joIpq7MThzON49EPp37
wBVKJ+vQtj++/OS84f4uxld3y3j/iwIP67Y8JXmwB9FuES/Acy+8RH1FbUUe1ZNfQaxqjNouXTRd
ZYJPkMsCAwx6sQKCAQB5XE2y8roFQJ9im5gZv0K3ITWFsi0oRCJsVAzX2JVhP/QZWvpSp5B6tBfx
nqRX4LZZubS6ZB9fR7qbrbh77yGjimhhL1Yr5has2cDuJhJj2vvYf/oEhiAgrHTLwud3txQSuWyl
H3aU/QGOOze/FZsiJrMvQ/tRrJ00jU2rbRwRz0xPln7THUh3PKQfK93qOPTOwqEOSGJv7NvB4LcR
MPCaVFupZbSC+ox9Lrl1dz6RzkOMAYoHO4x/L3sI9zeRfofol6k5JA49TpNIYZ/QK4P5REcf8Xj4
mTENXGVwf1pJggAxfu32uNKKsbq9WTILji7/HxhuhOONjrOc+UxAv3dhAgMuK/k=
-----END RSA PRIVATE KEY-----
```


2. Reading Flag (Encrypted File)

2.1 Open flag.enc file, see base64 format characters that are not real base64 format characters.


```
CQGd9sC/h9lnLpua50/071knSsP4N8WdmRsjoNIdfclrBhMjp7NoM5xy2SlNLLC2
yh7wbRw08nwjo6UF4tmGKKfcjPcb4l4bFa5uvyMY1nJBvmqQylDbiCns0DjhpB1B
JfdpU1LUKtwsCxbc7fPL/zzUdWg0+of/R9WmM+QOBPagTANbJo0mpDYxvNKRjvac
9Bw4CQTTh87moqsNRSE/Ik5tV2pkFRZfQxAZWuVePsHp0RXVitHwvKzwmN9vMqGm
57Wb2Sto64db4gLJDh9GROQN+EQh3yLoSS8NNtBrZCDddzfKHa8wv6zN/5znvBst
sDBkGyi88NzQxw9k0GjCWtwpRw==
```


2.2 Remove unwanted characters

Remove unwanted characters from flag.enc file by using sed command.

Commands: sed -e ':a;N;$!ba;s/ //g;s/\n//g' flag.enc

```
root@kali:~/Desktop/CT504/rsatool-master# sed -e ':a;{N;s/ //g;s/\n//g};ba' flag.enc
CQGd9sC/h9lnLpua50/071knSsP4N8WdmRsjoNIdfclrBhMjp7NoM5xy2SlNLLC2yh7wbRw08nwjo6UF4tmGKKfcjPcb4l4bFa5uvyMY1nJBvmqQylDbiCns0DjhpB1BJfdpU1LUKtwsCxbc7fPL/z
zUdWg0+of/R9WmM+QOBPagTANbJo0mpDYxvNKRjvac9Bw4CQTTh87moqsNRSE/Ik5tV2pkFRZfQxAZWuVePsHp0RXVitHwvKzwmN9vMqGm57Wb2Sto64db4gLJDh9GROQN+EQh3yLoSS8NNtBrZCDd
dzfKHa8wv6zN/5znvBstsDBkGyi88NzQxw9k0GjCWtwpRw==root@kali:~/Desktop/CT504/rsatool-master#
```

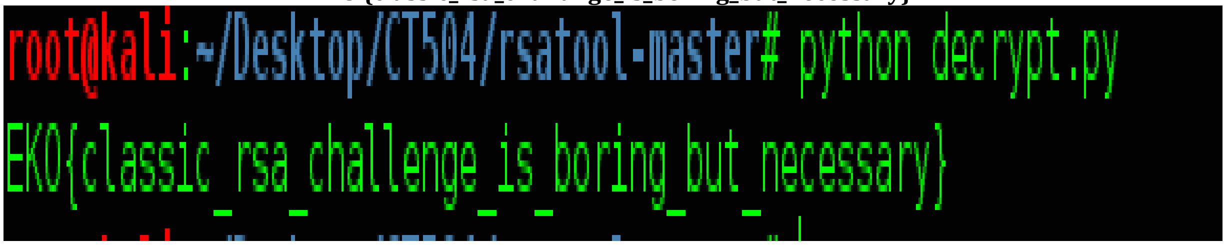
2.3 Decryption with python program

Use python program to decrypt base64 characters with private key file.

```
def decrypt_RSA(private_key_loc, package):
    from Crypto.PublicKey import RSA
    from Crypto.Cipher import PKCS1_OAEP
    from base64 import b64decode
    key = open(private_key_loc, "r").read()
    rsakey = RSA.importKey(key)
    rsakey = PKCS1_OAEP.new(rsakey)
    decrypted = rsakey.decrypt(b64decode(package))
    return decrypted
flag = 'CQGd9sC/
h9lnLpua50/071knSsP4N8WdmRsjoNIdfclrBhMjp7NoM5xy2SlNLLC2yh7wbRw08nwjo6UF4tmGKKfcjPcb4l4bFa5uvyMY1nJBvmqQylDbiCns0DjhpB1BJfdpU1LUKtwsCxbc7fPL/zzUdWg0
+of/R9WmM+QOBPagTANbJo0mpDYxvNKRjvac9Bw4CQTTh87moqsNRSE/Ik5tV2pkFRZfQxAZWuVePsHp0RXVitHwvKzwmN9vMqGm57Wb2Sto64db4gLJDh9GROQN
+EQh3yLoSS8NNtBrZCDddzfKHa8wv6zN/5znvBstsDBkGyi88NzQxw9k0GjCWtwpRw=='
print decrypt_RSA('priv.pem', flag)
```

After running Python program, we get the plain text.

EKO {classic_rsa_challange_is_boring_but_necessary}.



Conclusion

Today, public key encryption and is widely used to secure sensitive data, particularly when it is being sent over in secure network such as the internet. Therefore, this paper describes how to work key generation in cryptography. RSA key is a private key based on RSA algorithm. Encryption is used for a secure symmetric key exchange that is used for actual transmitted data encryption .

References

[1] https://simple.wikipedia.org/wiki/RSA_algorithm.

[2] Behrouz A. Forouzan, Cryptography and Network Security, McGraw-Hill International edition, 2008.

[3] Cryptography & Network Security (project_paper) , University of Computer Studies, Pathein, Myanmar,2018

[4] http://mathworld.wolfram.com/RSAEncryption.html

[5] https://www.tutorialspoint.com/cryptography/public_key_encryption.

[6] https://ieeexplore.ieee.org/document/6021216

[7] https://www.schneier.com/blog/archives/2005/08/new_cryptanalyt.html [Accessed : Oct. 7, 2014]