CRYPTOGRAPHY

Mini project

On

An experimental study on Performance Evaluation of RSA and Elgamal Algorithms

By:

DIVYANSHU GOEL 13103501

NEELAKSH CHAUHAN 13103513

YASH GUPTA 13103485

Introduction

Information security is one of the key challenges in data communication. For secure information communication over public network, different cryptographic methods are applied. The cryptographic methods are widely classified as symmetric and asymmetric. In symmetric methods, encryption and decryption keys are same or decryption key is easily calculated from the encryption key. The problem with symmetric method is that participants must share a secret key in a secure way which is difficult. Asymmetric methods solve the problem of key distribution by using a pair of keys. It is computationally infeasible to determine the decryption key given only the knowledge of cryptographic algorithm and the encryption key.

RSA and Elgamal encryption scheme belongs to asymmetric algorithms. RSA is one of the oldest and most widely used encryption algorithm. In RSA, the key pair is derived from the product of two prime numbers chosen according to special rules.Elgamal is fundamental, efficient, and simple asymmetric algorithm and widely known as alternative to RSA.

Evaluation Parameters

We selected following parameters for evaluation of RSA, ElGamal & Pallier asymmetric encryption algorithms for both encryption and decryption schemes.

• **Encryption time (Computation Time/Response Time)**

The encryption time is considered the time that an encryption algorithm takes to produces a ciphertext from a plain text.

• **Decryption time (Computation Time/Response Time)**

The decryption time is considered the time that an encryption algorithm takes to reproduces a plain text from a ciphertext.

• **Throughput**

Throughput is equal to total plaintext in bytes encrypted divided by the encryption time. Higher the throughput, higher will be the performance.

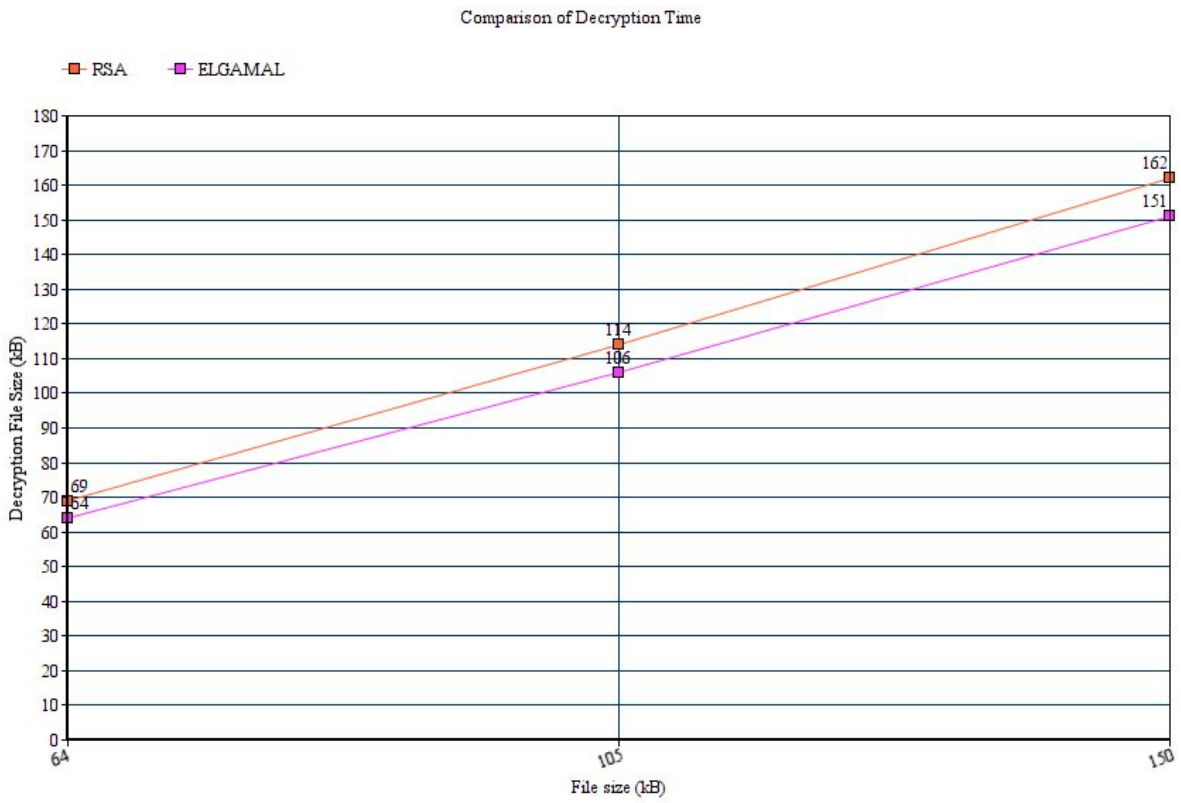
• **Encrypted File Size**

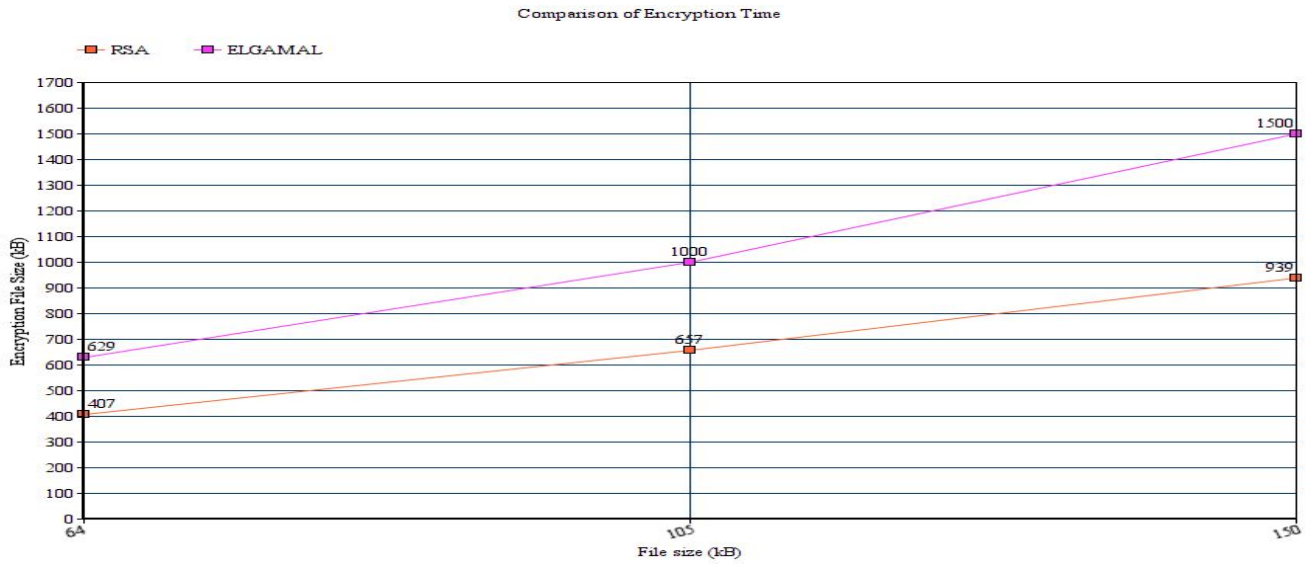
The size of encrypted file is called encrypted file size.

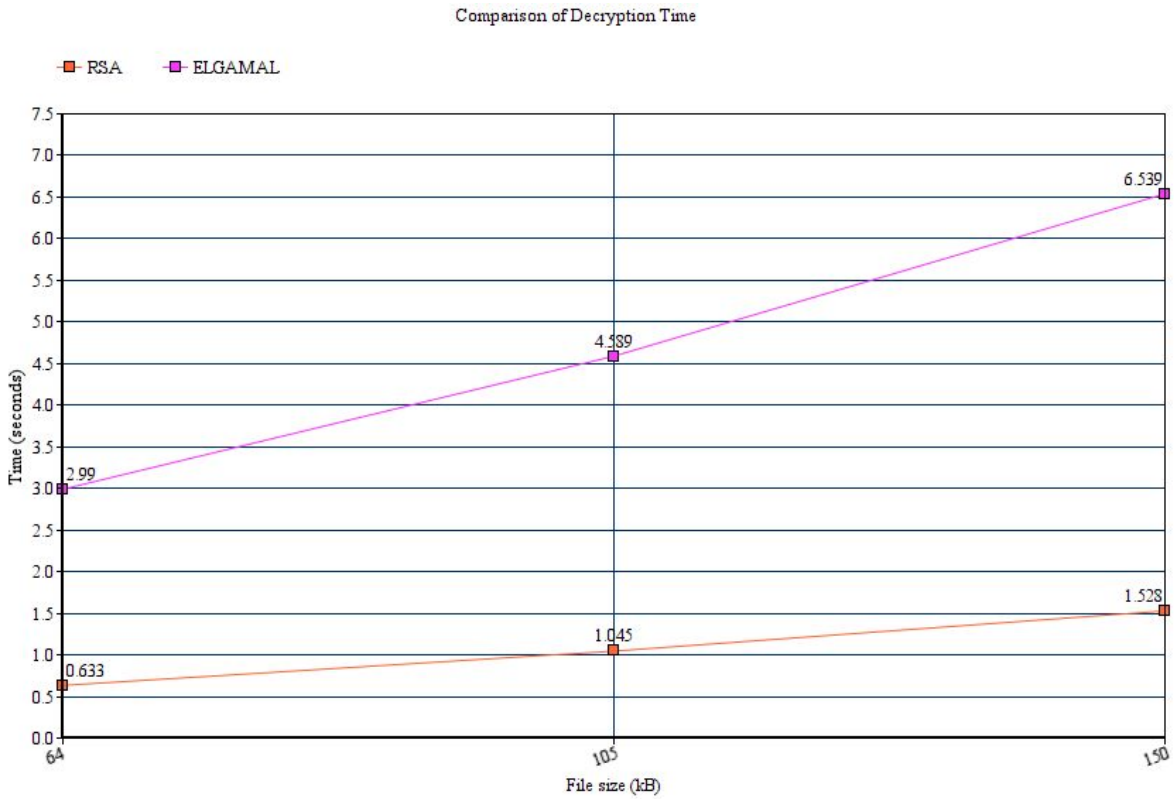
• **Decrypted File Size**

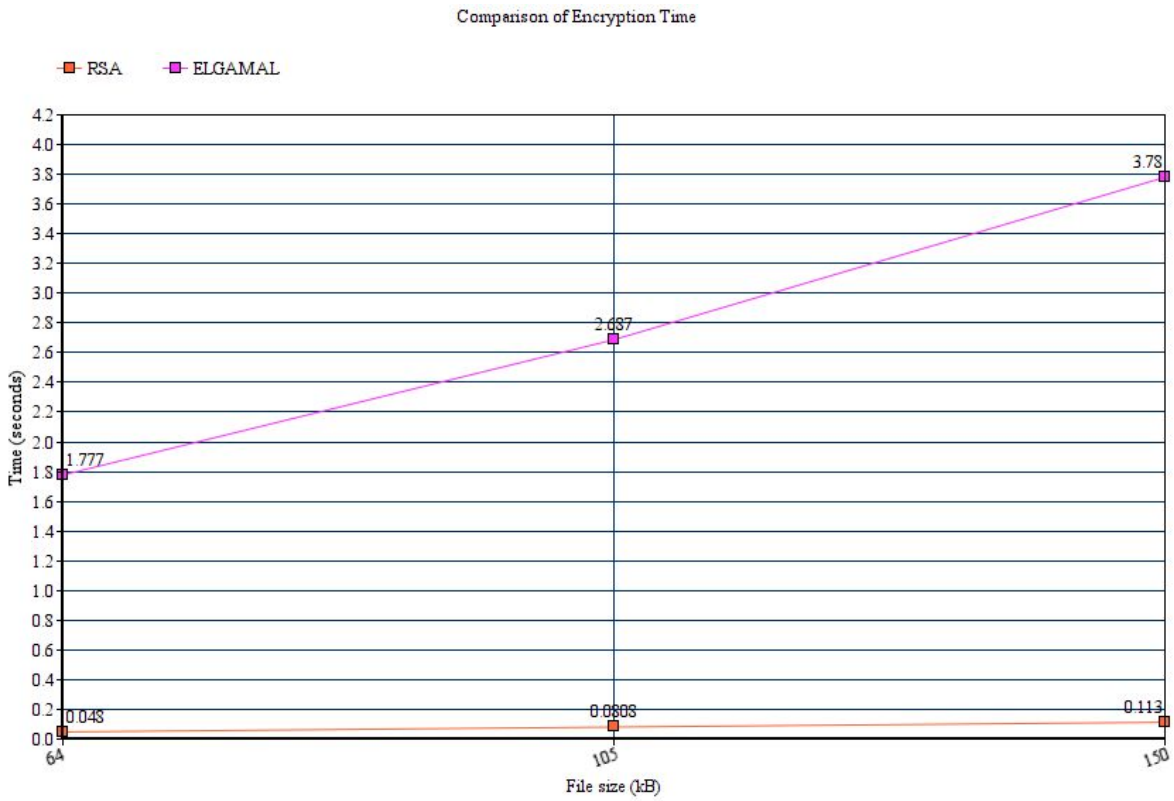
The size of decrypted file is called decrypted file size.

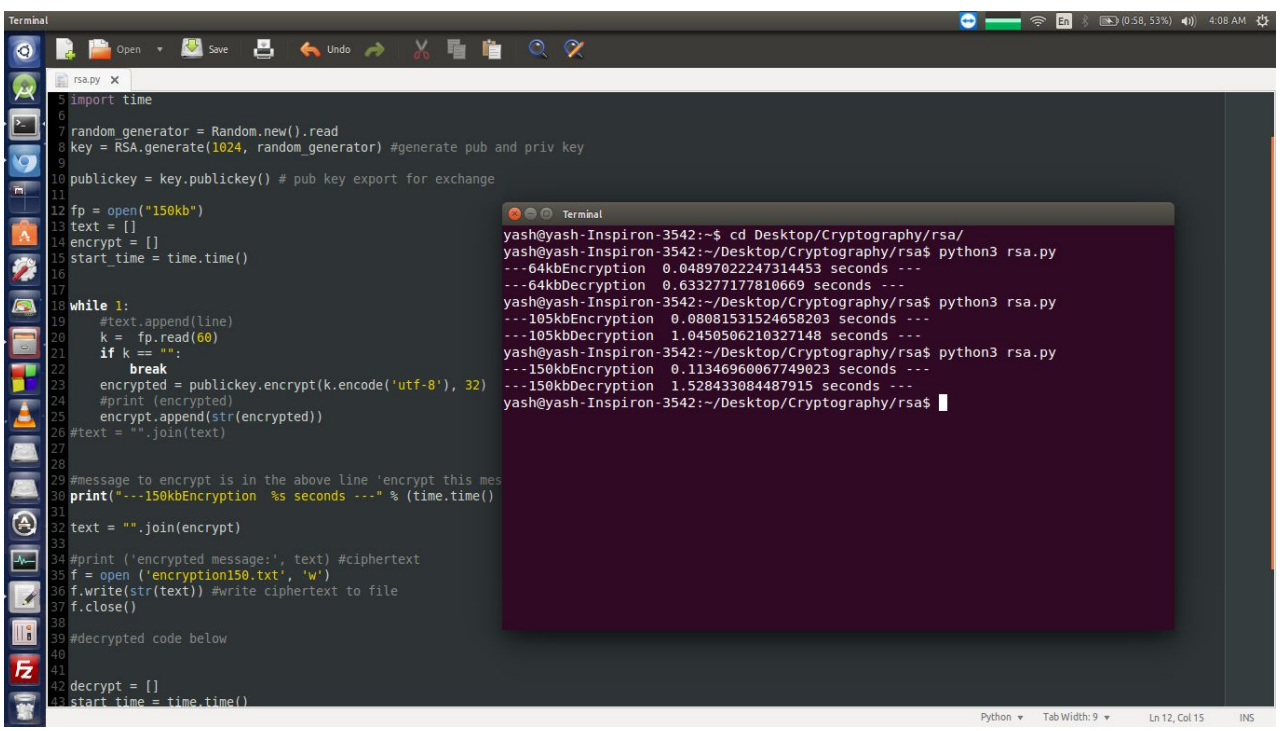
Results

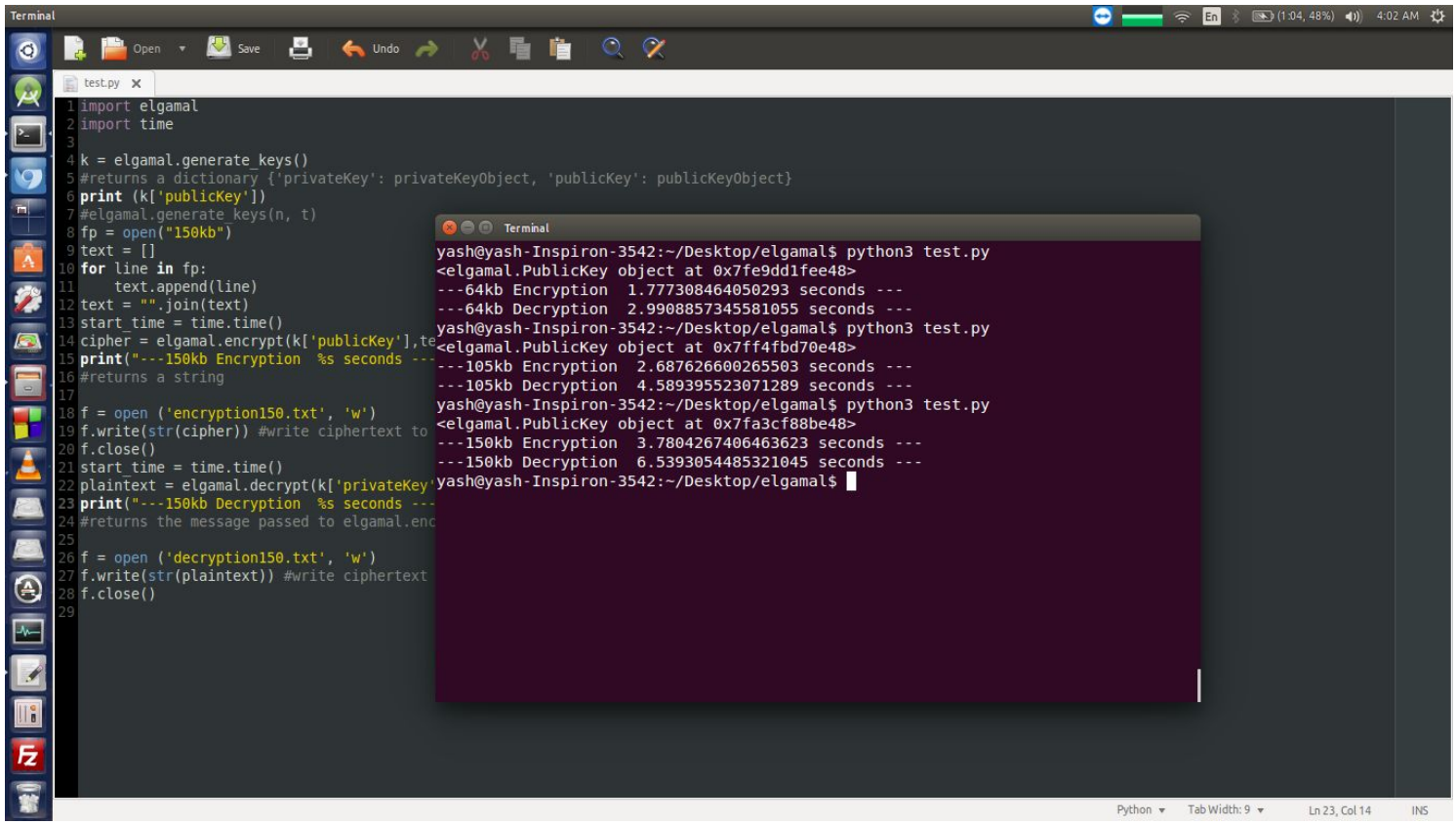



Comparison of Encryption Time
RSA
ELGAMAL
Encryption File Size (kB)
File size (kB)
1700
1600
1500
1400
1300
1200
1100
1000
900
800
700
600
500
400
300
200
100
0
64
105
150
629
1000
1500
407
657
939


Comparison of Decryption Time
RSA
ELGAMAL
Time (seconds)
File size (kB)
7.5
7.0
6.5
6.0
5.5
5.0
4.5
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
64
105
150
2.99
4.589
6.539
0.633
1.045
1.528


Comparison of Encryption Time
RSA
ELGAMAL
Time (seconds)
File size (kB)
1.777
2.637
3.78
0.048
0.0803
0.113
64
105
150


Terminal
rsa.py
import time
random_generator = Random.new().read
key = RSA.generate(1024, random_generator) #generate pub and priv key
publickey = key.publickey() # pub key export for exchange
fp = open("150kb")
text = []
encrypt = []
start_time = time.time()
while 1:
#text.append(line)
k = fp.read(60)
if k == "":
break
encrypted = publickey.encrypt(k.encode('utf-8'), 32)
#print (encrypted)
encrypt.append(str(encrypted))
#text = "".join(text)
#message to encrypt is in the above line 'encrypt this mes
print("---150kbEncryption %s seconds ---" % (time.time()
text = "".join(encrypt)
#print ('encrypted message:', text) #ciphertext
f = open ('encryption150.txt', 'w')
f.write(str(text)) #write ciphertext to file
f.close()
#decrypted code below
decrypt = []
start_time = time.time()
Terminal
yash@yash-Inspiron-3542:~$ cd Desktop/Cryptography/rsa/
yash@yash-Inspiron-3542:~/Desktop/Cryptography/rsa$ python3 rsa.py
---64kbEncryption 0.04897022247314453 seconds ---
---64kbDecryption 0.633277177810669 seconds ---
yash@yash-Inspiron-3542:~/Desktop/Cryptography/rsa$ python3 rsa.py
---105kbEncryption 0.08081531524658203 seconds ---
---105kbDecryption 1.0450506210327148 seconds ---
yash@yash-Inspiron-3542:~/Desktop/Cryptography/rsa$ python3 rsa.py
---150kbEncryption 0.11346960067749023 seconds ---
---150kbDecryption 1.528433084487915 seconds ---
yash@yash-Inspiron-3542:~/Desktop/Cryptography/rsa$


Terminal
Open
Save
Undo
test.py
import elgamal
import time
k = elgamal.generate_keys()
#returns a dictionary {'privateKey': privateKeyObject, 'publicKey': publicKeyObject}
print (k['publicKey'])
#elgamal.generate_keys(n, t)
fp = open("150kb")
text = []
for line in fp:
text.append(line)
text = "".join(text)
start_time = time.time()
cipher = elgamal.encrypt(k['publicKey'],te
print("---150kb Encryption %s seconds ---
#returns a string
f = open ('encryption150.txt', 'w')
f.write(str(cipher)) #write ciphertext to
f.close()
start_time = time.time()
plaintext = elgamal.decrypt(k['privateKey'
print("---150kb Decryption %s seconds ---
#returns the message passed to elgamal.enc
f = open ('decryption150.txt', 'w')
f.write(str(plaintext)) #write ciphertext
f.close()
Terminal
yash@yash-Inspiron-3542:~/Desktop/elgamal$ python3 test.py
<elgamal.PublicKey object at 0x7fe9dd1fee48>
---64kb Encryption 1.777308464050293 seconds ---
---64kb Decryption 2.9908857345581055 seconds ---
yash@yash-Inspiron-3542:~/Desktop/elgamal$ python3 test.py
<elgamal.PublicKey object at 0x7ff4fbd70e48>
---105kb Encryption 2.687626600265503 seconds ---
---105kb Decryption 4.589395523071289 seconds ---
yash@yash-Inspiron-3542:~/Desktop/elgamal$ python3 test.py
<elgamal.PublicKey object at 0x7fa3cf88be48>
---150kb Encryption 3.7804267406463623 seconds ---
---150kb Decryption 6.5393054485321045 seconds ---
yash@yash-Inspiron-3542:~/Desktop/elgamal$
Python
Tab Width: 9
Ln 23, Col 14
INS

Thanks