Teammates: Ron Gassner, Lucas Yao

Course name: **CSCI 360: Cryptography and Cryptanalysis**

Professor: Matluba Khodjaeva

Project 2 Report

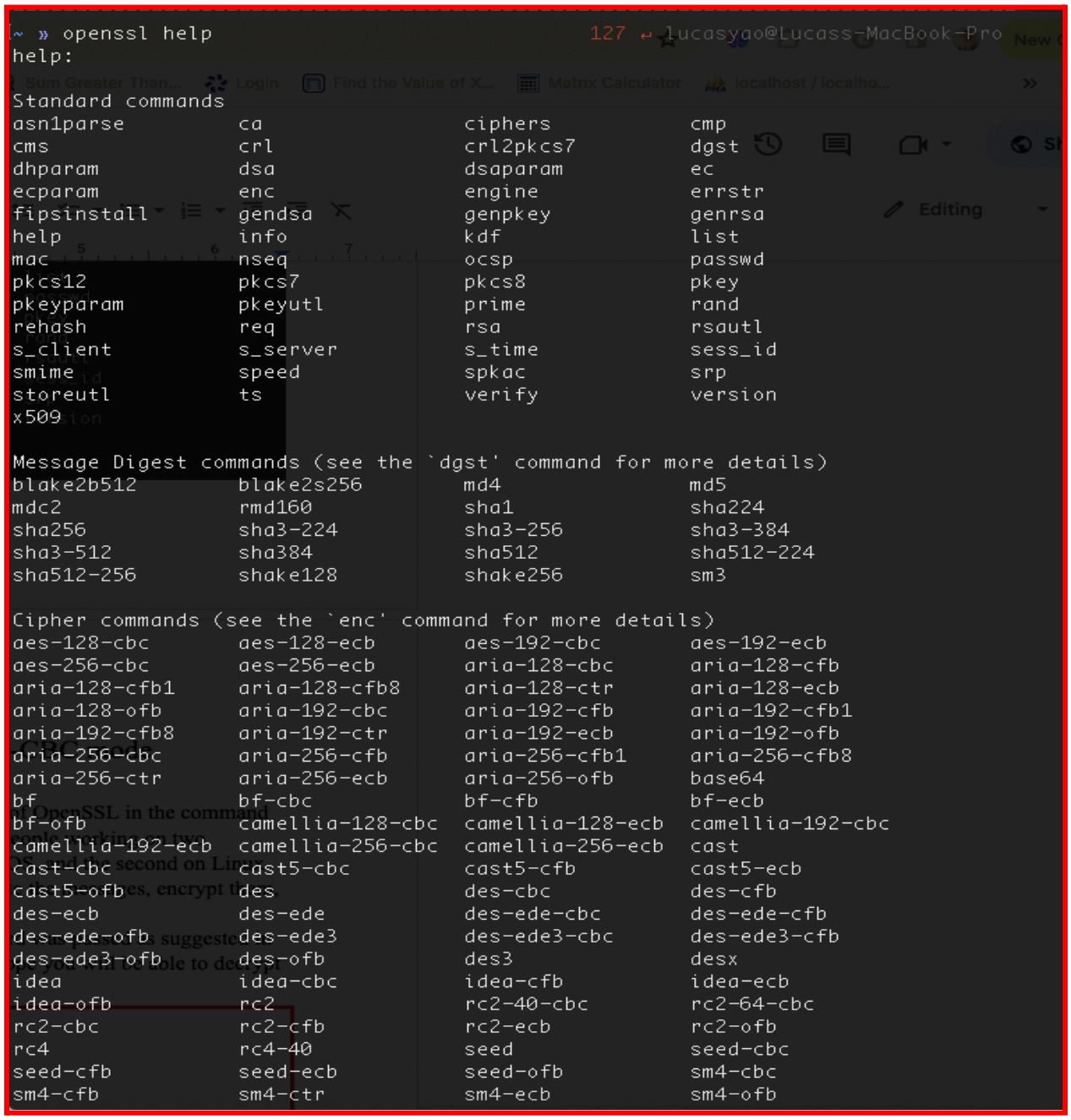
Youtube Video summary: The video provides an explanation of how to use OpenSSL for encryption and decryption in a Linux system. The tutorial covers various aspects, such as symmetric encryption, creating public and private keys, sharing public keys, encrypting and decrypting messages, and signing and verifying signatures. It also emphasizes the significance of encrypting the private key for security purposes. OpenSSL is an excellent tool that assists with encryption functions in Linux systems. You can check the OpenSSL version and view available ciphers. Using OpenSSL to encrypt and decrypt messages is straightforward. OpenSSL enables you to specify keys for encryption/decryption in Excel decimal format. It can also manage symmetric encryption and RSA encryption with public and private keys. You can generate RSA public and private keys using OpenSSL and perform operations with the key files. To generate and share public and private keys for encryption, exponentiate the two primes and the public. Then use OpenSSL RSA to create public and private keys. Finally, share public keys between users by linking files in folders. The encryption process involves using the recipient's public key to encrypt the message, while the decryption process requires the recipient's private key to decrypt the encrypted message. OpenSSL RSA utility enables you to encrypt and decrypt messages using private keys. You can also sign and verify signatures in files with OpenSSL. Use the public key to verify signatures created with the private key. It is essential to encrypt private keys using methods like SSL for security. Encrypting private keys enhances security when accessing them for operations like signing.

Task 1

```
Standard commands
asn1parse         ca                crl2pkcs7         ciphers
cms               crl               dgst              dh
dhparam           dsa               dsaparam          ec
ecparam           enc               errstr            gendh
gendsa            genpkey           genrsa            nseq
ocsp              passwd            pkcs12            pkcs7
pkcs8             pkey              pkeyparam         pkeyutl
prime             rand              req               rsa
rsautl            s_client          s_server          s_time
sess_id           smime             speed             spkac
ts                verify            version           x509
```

```
Cipher commands (see the `enc' command for more details)
aes-128-cbc       aes-128-ecb       aes-192-cbc       aes-192-ecb
aes-256-cbc       aes-256-ecb       base64            bf
bf-cbc            bf-cfb            bf-ecb            bf-ofb
camellia-128-cbc  camellia-128-ecb  camellia-192-cbc  camellia-192-ecb
camellia-256-cbc  camellia-256-ecb  cast              cast-cbc
cast5-cbc         cast5-cfb         cast5-ecb         cast5-ofb
chacha            des               des-cbc           des-cfb
des-ecb           des-ede           des-ede-cbc       des-ede-cfb
des-ede-ofb       des-ede3          des-ede3-cbc      des-ede3-cfb
des-ede3-ofb      des-ofb           des3              desx
rc2               rc2-40-cbc        rc2-64-cbc        rc2-cbc
rc2-cfb           rc2-ecb           rc2-ofb           rc4
rc4-40            sm4               sm4-cbc           sm4-cfb
sm4-ecb           sm4-ofb
```

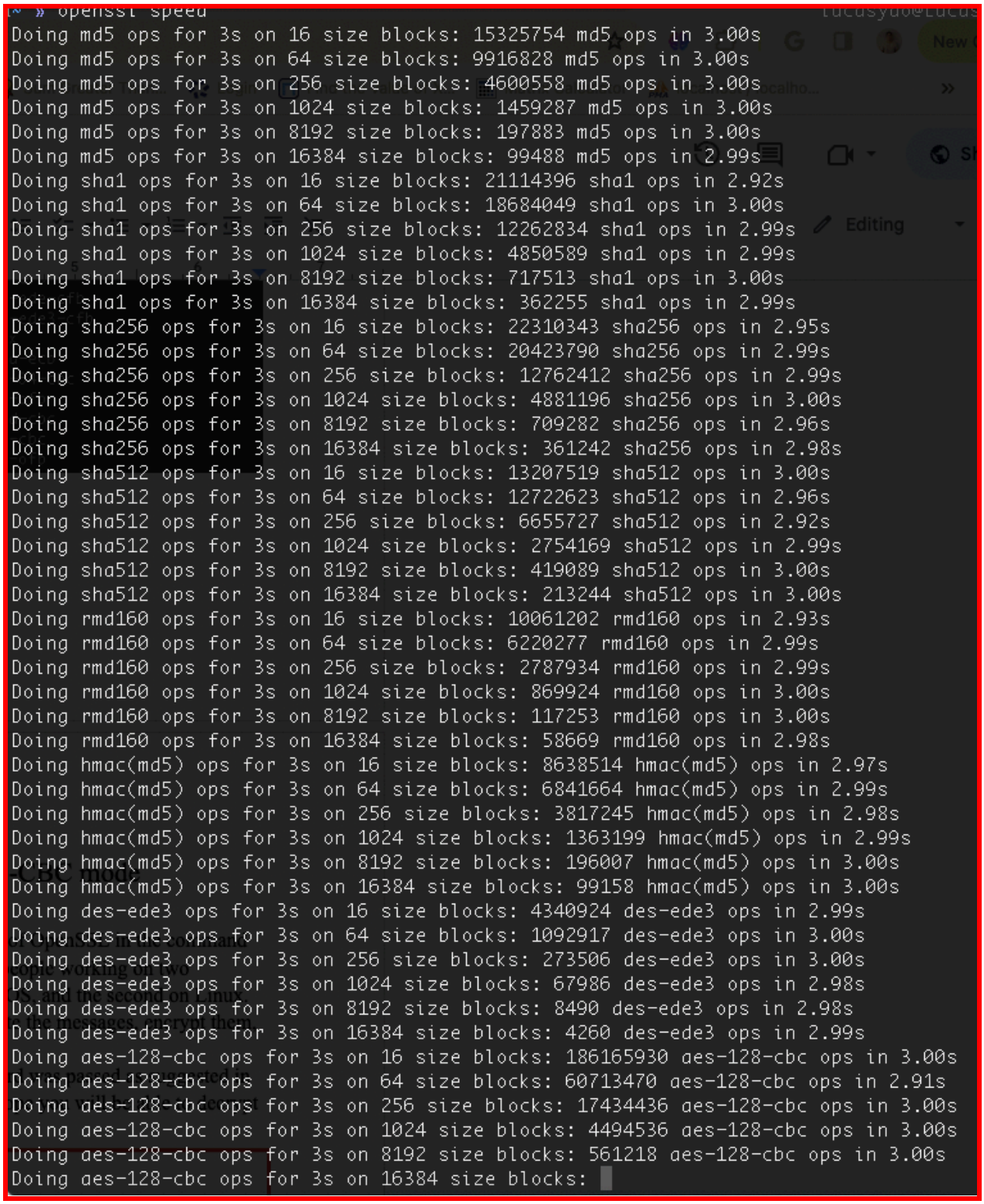
```
Message Digest commands (see the `dgst' command for more details)
gost-mac          md4               md5               md_gost94
ripemd160         sha1              sha224            sha256
sha384            sha512            sm3               sm3WithRSAEncryption
streebog256       streebog512       whirlpool
```



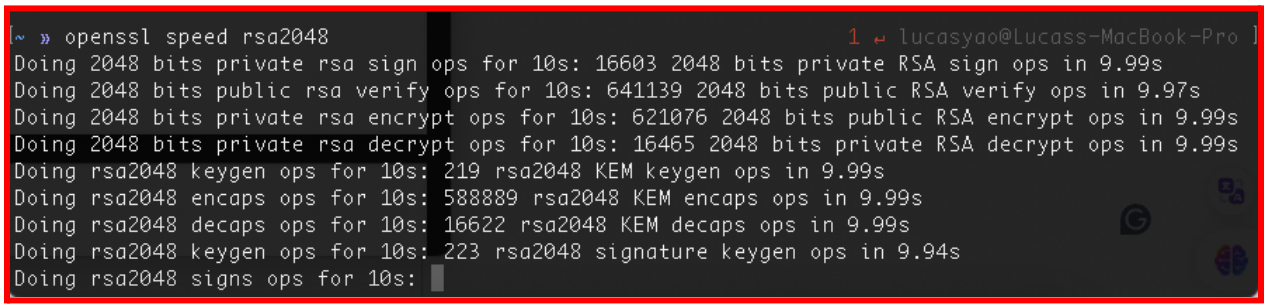
```
~ » openssl help                                    127 ↵ lucasyao@Lucass-MacBook-Pro
help:

Standard commands
asn1parse         ca                ciphers           cmp
cms               crl               crl2pkcs7         dgst
dhparam           dsa               dsaparam          ec
ecparam           enc               engine            errstr
fipsinstall       gendsa            genpkey           genrsa
help              info              kdf               list
mac               nseq              ocsp              passwd
pkcs12            pkcs7             pkcs8             pkey
pkeyparam         pkeyutl           prime             rand
rehash            req               rsa               rsautl
s_client          s_server          s_time            sess_id
smime             speed             spkac             srp
storeutl          ts                verify            version
x509

Message Digest commands (see the `dgst' command for more details)
blake2b512        blake2s256        md4               md5
mdc2              rmd160            sha1              sha224
sha256            sha3-224          sha3-256          sha3-384
sha3-512          sha384            sha512            sha512-224
sha512-256        shake128          shake256          sm3

Cipher commands (see the `enc' command for more details)
aes-128-cbc       aes-128-ecb       aes-192-cbc       aes-192-ecb
aes-256-cbc       aes-256-ecb       aria-128-cbc      aria-128-cfb
aria-128-cfb1     aria-128-cfb8     aria-128-ctr      aria-128-ecb
aria-128-ofb      aria-192-cbc      aria-192-cfb      aria-192-cfb1
aria-192-cfb8     aria-192-ctr      aria-192-ecb      aria-192-ofb
aria-256-cbc      aria-256-cfb      aria-256-cfb1     aria-256-cfb8
aria-256-ctr      aria-256-ecb      aria-256-ofb      base64
bf                bf-cbc            bf-cfb            bf-ecb
bf-ofb            camellia-128-cbc  camellia-128-ecb  camellia-192-cbc
camellia-192-ecb  camellia-256-cbc  camellia-256-ecb  cast
cast-cbc          cast5-cbc         cast5-cfb         cast5-ecb
cast5-ofb         des               des-cbc           des-cfb
des-ecb           des-ede           des-ede-cbc       des-ede-cfb
des-ede-ofb       des-ede3          des-ede3-cbc      des-ede3-cfb
des-ede3-ofb      des-ofb           des3              desx
idea              idea-cbc          idea-cfb          idea-ecb
idea-ofb          rc2               rc2-40-cbc        rc2-64-cbc
rc2-cbc           rc2-cfb           rc2-ecb           rc2-ofb
rc4               rc4-40            seed              seed-cbc
seed-cfb          seed-ecb          seed-ofb          sm4-cbc
sm4-cfb           sm4-ctr           sm4-ecb           sm4-ofb
```

Task 2

Make a speed test on your PC-platform with the speed command

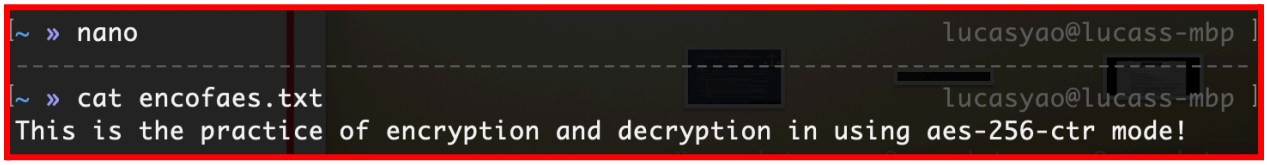

» openssl speed
Doing md5 ops for 3s on 16 size blocks: 15325754 md5 ops in 3.00s
Doing md5 ops for 3s on 64 size blocks: 9916828 md5 ops in 3.00s
Doing md5 ops for 3s on 256 size blocks: 4600558 md5 ops in 3.00s
Doing md5 ops for 3s on 1024 size blocks: 1459287 md5 ops in 3.00s
Doing md5 ops for 3s on 8192 size blocks: 197883 md5 ops in 3.00s
Doing md5 ops for 3s on 16384 size blocks: 99488 md5 ops in 2.99s
Doing sha1 ops for 3s on 16 size blocks: 21114396 sha1 ops in 2.92s
Doing sha1 ops for 3s on 64 size blocks: 18684049 sha1 ops in 3.00s
Doing sha1 ops for 3s on 256 size blocks: 12262834 sha1 ops in 2.99s
Doing sha1 ops for 3s on 1024 size blocks: 4850589 sha1 ops in 2.99s
Doing sha1 ops for 3s on 8192 size blocks: 717513 sha1 ops in 3.00s
Doing sha1 ops for 3s on 16384 size blocks: 362255 sha1 ops in 2.99s
Doing sha256 ops for 3s on 16 size blocks: 22310343 sha256 ops in 2.95s
Doing sha256 ops for 3s on 64 size blocks: 20423790 sha256 ops in 2.99s
Doing sha256 ops for 3s on 256 size blocks: 12762412 sha256 ops in 2.99s
Doing sha256 ops for 3s on 1024 size blocks: 4881196 sha256 ops in 3.00s
Doing sha256 ops for 3s on 8192 size blocks: 709282 sha256 ops in 2.96s
Doing sha256 ops for 3s on 16384 size blocks: 361242 sha256 ops in 2.98s
Doing sha512 ops for 3s on 16 size blocks: 13207519 sha512 ops in 3.00s
Doing sha512 ops for 3s on 64 size blocks: 12722623 sha512 ops in 2.96s
Doing sha512 ops for 3s on 256 size blocks: 6655727 sha512 ops in 2.92s
Doing sha512 ops for 3s on 1024 size blocks: 2754169 sha512 ops in 2.99s
Doing sha512 ops for 3s on 8192 size blocks: 419089 sha512 ops in 3.00s
Doing sha512 ops for 3s on 16384 size blocks: 213244 sha512 ops in 3.00s
Doing rmd160 ops for 3s on 16 size blocks: 10061202 rmd160 ops in 2.93s
Doing rmd160 ops for 3s on 64 size blocks: 6220277 rmd160 ops in 2.99s
Doing rmd160 ops for 3s on 256 size blocks: 2787934 rmd160 ops in 2.99s
Doing rmd160 ops for 3s on 1024 size blocks: 869924 rmd160 ops in 3.00s
Doing rmd160 ops for 3s on 8192 size blocks: 117253 rmd160 ops in 3.00s
Doing rmd160 ops for 3s on 16384 size blocks: 58669 rmd160 ops in 2.98s
Doing hmac(md5) ops for 3s on 16 size blocks: 8638514 hmac(md5) ops in 2.97s
Doing hmac(md5) ops for 3s on 64 size blocks: 6841664 hmac(md5) ops in 2.99s
Doing hmac(md5) ops for 3s on 256 size blocks: 3817245 hmac(md5) ops in 2.98s
Doing hmac(md5) ops for 3s on 1024 size blocks: 1363199 hmac(md5) ops in 2.99s
Doing hmac(md5) ops for 3s on 8192 size blocks: 196007 hmac(md5) ops in 3.00s
Doing hmac(md5) ops for 3s on 16384 size blocks: 99158 hmac(md5) ops in 3.00s
Doing des-ede3 ops for 3s on 16 size blocks: 4340924 des-ede3 ops in 2.99s
Doing des-ede3 ops for 3s on 64 size blocks: 1092917 des-ede3 ops in 3.00s
Doing des-ede3 ops for 3s on 256 size blocks: 273506 des-ede3 ops in 3.00s
Doing des-ede3 ops for 3s on 1024 size blocks: 67986 des-ede3 ops in 2.98s
Doing des-ede3 ops for 3s on 8192 size blocks: 8490 des-ede3 ops in 2.98s
Doing des-ede3 ops for 3s on 16384 size blocks: 4260 des-ede3 ops in 2.99s
Doing aes-128-cbc ops for 3s on 16 size blocks: 186165930 aes-128-cbc ops in 3.00s
Doing aes-128-cbc ops for 3s on 64 size blocks: 60713470 aes-128-cbc ops in 2.91s
Doing aes-128-cbc ops for 3s on 256 size blocks: 17434436 aes-128-cbc ops in 3.00s
Doing aes-128-cbc ops for 3s on 1024 size blocks: 4494536 aes-128-cbc ops in 3.00s
Doing aes-128-cbc ops for 3s on 8192 size blocks: 561218 aes-128-cbc ops in 3.00s
Doing aes-128-cbc ops for 3s on 16384 size blocks:

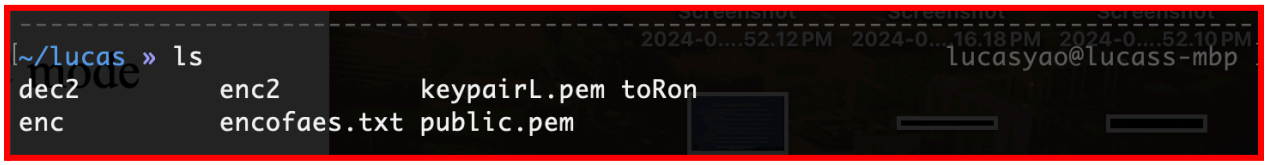
- Compare the results for symmetric encryption

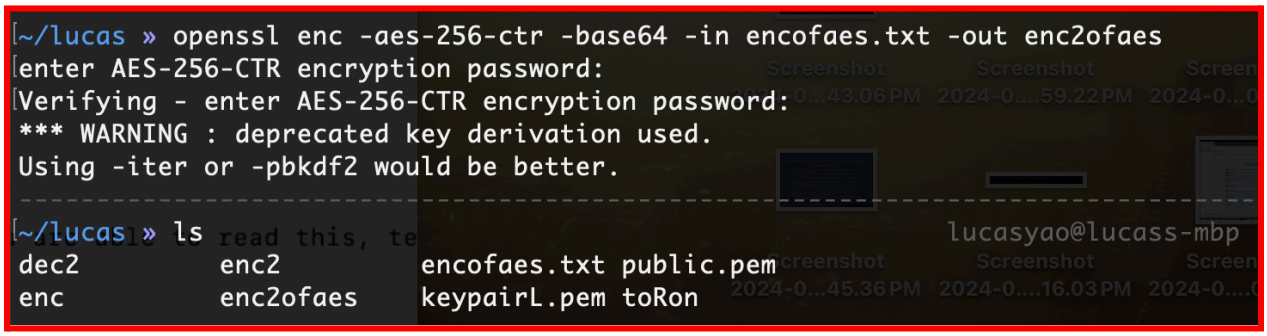


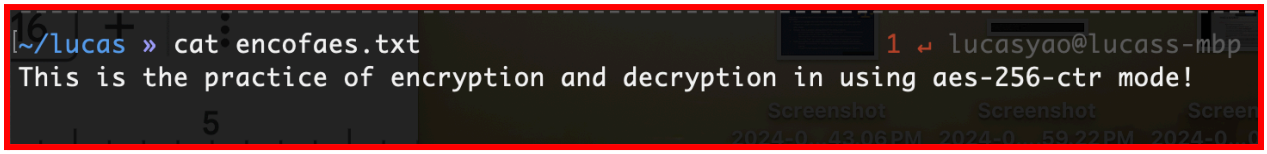
Task 3

i. Encryption and Decryption with AES-256-CTR mode









```
~/lucas » cat enc2ofaes                                    1 ↵ lucasyao@lucass-mbp
U2FsdGVkX1/vVll2ohHAtZWJu4+l1aN65QAEwQKedBeL6d+bdjPf7XL+idGswgIv
laOuUhG4I3pnjJdYD24PyZc7ElcHZYDJ8zFzLIudkMPNzeDjHFNI8ErVu0Rj1Ks=

~/lucas » openssl enc -aes-256-ctr -d -base64 -in enc2ofaes
enter AES-256-CTR decryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
This is the practice of encryption and decryption in using aes-256-ctr mode!
```

ii. Encryption with DES algorithm

```
~/lucas » nano                                             1 ↵ lucasyao@lucass-mbp

~/lucas » cat despra                                       lucasyao@lucass-mbp
this is a practice of DES algorithm for enc and dec!

~/lucas » openssl enc -des -in despra -out despra.enc      lucasyao@lucass-mbp
enter DES-CBC encryption password:
Verifying - enter DES-CBC encryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
Error setting cipher DES-CBC
009043D701000000:error:0308010C:digital envelope routines:inner_evp_generic_fetc
h:unsupported:crypto/evp/evp_fetch.c:355:Global default library context, Algorit
hm (DES-CBC : 15), Properties ()

~/lucas » ls                                               1 ↵ lucasyao@lucass-mbp
dec2          despra.enc    enc2          encofaes.txt encofdes.enc public.pem
despra        enc           enc2ofaes     encofdes     keypairL.pem toRon

~/lucas » cat despra.enc                                   lucasyao@lucass-mbp
Salted__???D??%
```

iii. Encryption and Decryption with AES-128-CBC mode algorithm:

Ron Gassner: On this project, we were asked to practice the usage of OpenSSL in the command line for basis functions. We watched two YouTube videos by two people working on two different operating systems. The first seems to be working on MacOS, and the second on Linux. The two videos provided guidance and instructions on how to create the messages, encrypt them, and decrypt them.

I decided to use AES-128-CBC for the encryption, and the password was passed as suggested in the video. My message was: “This is a message for Lucas Yao; I hope you will be able to decrypt it! If you are able to read this, text me!

```
 -camellia-256-cfb          -camellia-256-cfb1         -camellia-256-cfb8
 -camellia-256-ecb          -camellia-256-ofb          -camellia128
 -camellia192               -camellia256               -cast
 -cast-cbc                  -cast5-cbc                 -cast5-cfb
 -cast5-ecb                 -cast5-ofb                 -chacha
 -des                       -des-cbc                   -des-cfb
 -des-cfb1                  -des-cfb8                  -des-ecb
 -des-ede                   -des-ede-cbc               -des-ede-cfb
 -des-ede-ofb               -des-ede3                  -des-ede3-cbc
 -des-ede3-cfb              -des-ede3-cfb1             -des-ede3-cfb8
 -des-ede3-ofb              -des-ofb                   -des3
 -desx                      -desx-cbc                  -gost89
 -gost89-cnt                -gost89-ecb                -id-aes128-CCM
 -id-aes128-GCM             -id-aes128-wrap            -id-aes192-CCM
 -id-aes192-GCM             -id-aes192-wrap            -id-aes256-CCM
 -id-aes256-GCM             -id-aes256-wrap            -rc2
 -rc2-40-cbc                -rc2-64-cbc                -rc2-cbc
 -rc2-cfb                   -rc2-ecb                   -rc2-ofb
 -rc4                       -rc4-40                    -rc4-hmac-md5
 -sm4                       -sm4-cbc                   -sm4-cfb
 -sm4-ctr                   -sm4-ecb                   -sm4-ofb

ron@rons-mbp ~ % clear

ron@rons-mbp ~ % nano
ron@rons-mbp ~ % cat Message.txt
This is a message for Lucas Yao, I hope you will be able to decrypt it! if you are able to read this, text me!
ron@rons-mbp ~ % ls
Applications    Downloads       Movies          Public
Desktop         Library         Music           public.pem
Documents       Message.txt     Pictures
ron@rons-mbp ~ % openssl enc -aes-128-cbc -base64 -in Message.txt -out enc -K public.pem
iv undefined
ron@rons-mbp ~ % openssl enc -aes-128-cbc -base64 -in Message.txt -out enc
enter aes-128-cbc encryption password:
Verifying - enter aes-128-cbc encryption password:
ron@rons-mbp ~ % ls
Applications    Downloads       Movies          Public
Desktop         Library         Music           enc
Documents       Message.txt     Pictures        public.pem
ron@rons-mbp ~ % cat Message.txt
This is a message for Lucas Yao, I hope you will be able to decrypt it! if you are able to read this, text me!
ron@rons-mbp ~ % cat enc
U2FsdGVkX18l9bkRrGtk5wlHu7Wv9w0aN0vdhpAYjUeRi0dCo7SdLkuloePQrDsi
NXJ1d1YyDF/84gbePbDRCEyPhO6doIbnEtWjojArXdNKsGwZwwK9gIyrukU5f6lS
G0djZfDb/ZypNDvAAGg25TjrnSjWZoJ6iJhf74zrOw0=
ron@rons-mbp ~ %
```

Lucas Yao: After receiving Ron Gassner's encrypted file through email, I saved it in the directory folder named Lucas.

```
~ » mkdir Lucas                                          lucasyao@lucass-mbp

~ » cd Lucas                                             lucasyao@lucass-mbp
```

```
~ » ls                                                   lucasyao@lucass-mbp
Applications  Documents  Library  Movies  Pictures  dotnet
Desktop       Downloads  Lucas    Music   Public    version

~ » cd Lucas                                             lucasyao@lucass-mbp
```

```
~ » cd lucas                                                    lucasyao@Lucass-MBP
~/lucas » ls                                                    lucasyao@Lucass-MBP
dec2          enc           enc2          keypairL.pem public.pem    toRon
```

```
~/Lucas » openssl enc -aes-128-cbc -d -base64 -in enc
enter AES-128-CBC decryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
This is a message for Lucas Yao, I hope you will be able to decrypt it! if you are able to read this, text me!
```

Lucas Yao: I used a terminal tool on my computer to locate the file and decrypted it with the encryption algorithm(AES-128-CBC) and the password when he created the encrypted file named enc.

Task 4

Using RSA with size 2048-bit and public key and private key to encrypt and decrypt the message.

```
~/Lucas » openssl genrsa -out keypairL.pem 2048          1 ↵ lucasyao@lucass-mbp
~/Lucas » ls                                                    lucasyao@lucass-mbp
keypairL.pem
~/Lucas » openssl rsa -in keypairL.pem -pubout -out public.pem
writing RSA key
```

Lucas Yao: I generated the RSA keypair called keypairL.pem file that contains private and public keys;
Then, I extracted the public key to a file called public.pem from the keypair and sent it to Ron so he can use it to encrypt his message.
Here is the public key:

```
~/Lucas » cat public.pem                                        lucasyao@lucass-mbp
-----BEGIN PUBLIC KEY-----
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEAtjo0f4pvX5EpNfERHIGG
R0m9mkKr0ihn+Sbx1fnAX4+/EtHN/WR35j/NXaNGiFxYYitjRwwKEJffC0tW7b9x
JKVVfFgQnk0F7UmS+bpLOihnSSjR8Mcgb36VH7F9fEaK8LOCJyw2H1mH2QUTQkn8
ReL+d+JriOUClgL5HPvnOLT+XluGTjBvkILRI0XQvCiFVCrxJkbusSdr/lfNLYCh
OIUurn37qo41amEEwXLiZtgtKVvbmodNQCTYrrSOKYgrjIymGzVJ0IDTXYGEdWR/
DkX4M8IBMIGWIfkRF99jRpNWQz7rY2RiClJATSjsRhs3MBIAlzW6zd32rktodZ9c
EQIDAQAB
-----END PUBLIC KEY-----
```

```
ron@rons-mbp ~ % nano
ron@rons-mbp ~ % ls
Applications    Documents       Library         Message2.txt    Music           Public          enc2
Desktop         Downloads       Message.txt     Movies          Pictures        enc             public.pem
ron@rons-mbp ~ % cat Message2.txt
Lucas I happy you were able to decrypt Message.txt and texted me. If you decrypt this one, call me!
ron@rons-mbp ~ % openssl rsautl -encrypt -in Message2.txt -out enc2 -inkey public.pem -pubin
ron@rons-mbp ~ % ls
Applications    Documents       Library         Message2.txt    Music           Public          enc2
Desktop         Downloads       Message.txt     Movies          Pictures        enc             public.pem
ron@rons-mbp ~ % cat enc2
zL?y?h?+?[???ш'l?G???L?]V{S??5???????5?Ń'Md?WG??V?=????????;?wG?V甇?B??2
?F$[??'???3? {??、hk??}??~?=u?{Y6??O1
                                   ??Ilp´yV₂??V3??5Wf???v?1]??+?y???   W?_?f}?,??i?
                                                                                  ???_R??[|&)J乾?
                                                                                        ck;y???`.? Ow???q??CD?X
????????B?;yY??P?s??6K:?e??%
ron@rons-mbp ~ %
```

Ron Gassner: After receiving Lucas's public key, I used it to encrypt the Message2.txt file that contains the words: Lucas, I happy you were able to decrypt Messege2.txt and texted me. If you decrypt this one, call me!

I generated the encrypted file called enc2

```
~/Lucas » openssl rsautl -decrypt -in enc2 -out dec2 -inkey keypairL.pem          lucasyao@lucass-mbp
The command rsautl was deprecated in version 3.0. Use 'pkeyutl' instead.
------------------------------------------------------------------------------------------------
~/Lucas » openssl pkeyutl -decrypt -in enc2 -out dec2 -inkey keypairL.pem         lucasyao@lucass-mbp
```

Lucas Yao: After receiving Ron Gassner's encrypted file enc2, I used my keypairL.pem to decrypt his message and save it as dec2 file in my directory.

```
~/Lucas » ls                                                                      lucasyao@lucass-mbp
dec2        enc         enc2        keypairL.pem public.pem   toRon
------------------------------------------------------------------------------------------------
```

Here is his decrypted message:

```
------------------------------------------------------------------------------------------------
~/Lucas » cat dec2                                                          127 ↵ lucasyao@lucass-mbp
Lucas I happy you were able to decrypt Message.txt and texted me. If you decrypt this one, call me!
```