CS 253 Project
Danah Veronica P. Torres
2008-16422

Environment:
MAC OS

Openssl is built in but you can install it via Homebrew using the command:

```
brew install openssl
```

For this project, I preferred to use the command line and my openssl version is OpenSSL 1.0.2k.

Symmetric Encryption

To perform symmetric AES-128 encryption CBC mode on the Lena image, I ran the following command via shell:

```
openssl aes-128-cbc -K 1 -iv 1 -e -in lena512color.tiff -out lena_encrypted_aes_cbc
```

I chose a random key and IV here which are both 1 (written in hex). The output file is saved in the same directory. To decrypt this, I ran this command via shell:

```
openssl aes-128-cbc -K 1 -iv 1 -d -in lena_encrypted_aes_cbc -out lena_decrypted_aes_cbc
```

For the ECB mode:

```
openssl aes-128-ecb -K 1 -iv 1 -e -in lena512color.tiff -out lena_encrypted_aes_ecb
```

To decrypt:

```
openssl aes-128-ecb -K 1 -iv 1 -d -in lena_encrypted_aes_ecb -out lena_decrypted_aes_ecb
```

Hashing

To hash the Lena image, I use the openssl dgst. Here are the commands and outputs for hashing using sha-1, sha-256, and sha-512.

```
Danahs-MacBook-Pro:CS253FinalProject danahveronicatorres$ openssl dgst -sha1 lena512color.tiff
SHA1(lena512color.tiff)= e647d0f6736f82e498de8398eccc48cf0a7d53b9
```

```
Danahs-MacBook-Pro:CS253FinalProject danahveronicatorres$ openssl dgst -sha256 lena512color.tiff
SHA256(lena512color.tiff)= c056da23302d2fb0d946e7ffa11e0d94618224193ff6e2f78ef8097bb8a3569b
```

```
Danahs-MacBook-Pro:CS253FinalProject danahveronicatorres$ openssl dgst -sha512 lena512color.tiff
SHA512(lena512color.tiff)= 2cb9d7df53eb8640dc48d736974f472a98d9c7186de7a972490455f5f3ed29dfc5b75c95ccb3ed4596bc2bfc4b1e52cf4d76bcee27d334dd155bb426617392dc
```

Public Key Encryption

For RSA Encryption, we must first generate the RSA key pair using the command:

```
openssl genrsa -des3 -out rsa_key.pem 2048
```

```
Danahs-MacBook-Pro:temp danahveronicatorres$ openssl genrsa -des3 -out rsa_key.pem 2048
Generating RSA private key, 2048 bit long modulus
..............+++
............+++
e is 65537 (0x10001)
Enter pass phrase for rsa_key.pem:
Verifying - Enter pass phrase for rsa_key.pem:
```

I chose -des3 because it's one of the encryptions we learned in class. I was asked to input a pass phrase — I chose the word 'lenaimage'. Now that we have a private key, we can then use this to export the public key through this command:

```
openssl rsa -in rsa_key.pem -outform PEM -pubout -out public.pem
```

```
Danahs-MacBook-Pro:temp danahveronicatorres$ openssl rsa -in rsa_key.pem -outform PEM -pubout -out public.pem
Enter pass phrase for rsa_key.pem:
writing RSA key
```

Using these keys, I tried to encrypt the Lena Image but got the following error:

```
openssl rsautl -encrypt -pubin -inkey public.pem -in
lena512color.tiff -out cipher.txt
```

```
Danahs-MacBook-Pro:temp danahveronicatorres$ openssl rsautl -encrypt -pubin -inkey public.pem -in lena512color.tiff -out cipher.txt
RSA operation error
12412:error:0406D06E:rsa routines:RSA_padding_add_PKCS1_type_2:data too large for key size:/BuildRoot/Library/Caches/com.apple.xbs/Sources/OpenSSL098/OpenSSL098-59.60.1/src/crypto/rsa/rsa_pk1.c:153:
```

I searched for a way to encrypt large files through RSA and I saw that it can be done with the help of AES. Again, start with the generation of the RSA key pairs (private and public):

```
openssl genrsa -out private_key.pem 2048
```

```
openssl rsa -in private_key.pem -out public_key.pem -outform PEM
-pubout
```

Then generate an AES key:

```
openssl rand -base64 32 | cut -c1-31 > aesKey.txt
```

Next encrypt the file using the generated AES key:

```
openssl enc -aes-256-cbc -salt -in lena512color.tiff -out lena.enc -pass file:./aesKey.txt
```

Lastly, encrypt the AES key with the public RSA key and save the outcome in an encrypted file.

```
openssl rsautl -encrypt -inkey public_key.pem -pubin -in aesKey.txt -out aesKey.txt.crypted
```

ECDSA Signature

To generate an ECDSA signature on the Lena image, we again need a private and public key pair. To generate the private key:

```
openssl ecparam -genkey -name secp384r1 -noout -out private_ecdsa.pem
```

And the public key:

```
openssl ec -in private_ecdsa.pem -pubout -out public_ecdsa.pem
```

Lastly, we use the key to sign the Lena image:

```
openssl dgst -sha256 -sign private_ecdsa.pem < lena512color.tiff > signature.bin
```

My references:

https://blog.engelke.com/2012/09/17/aes-encryption-with-openssl-command-line/

https://www.madboa.com/geek/openssl/

https://www.openssl.org/docs/man1.0.2/apps/

https://rietta.com/blog/2012/01/27/openssl-generating-rsa-key-from-command/

http://certificate.fyicenter.com/2032_OpenSSL_rsautl_data_too_large_for_key_size_Error.html

http://stackoverflow.com/questions/22856059/openssl-ecdsa-sign-and-verify-file

https://gist.github.com/carlj/650982