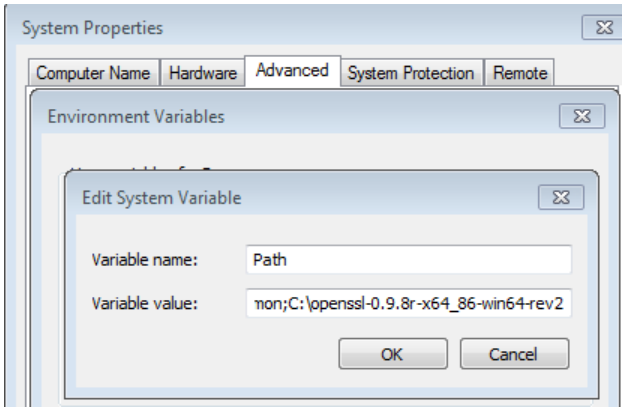
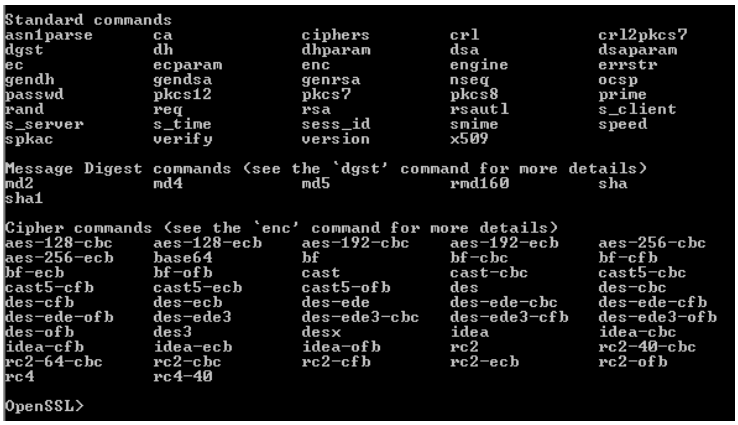
# CS-253-Cryptography-Exercises-using-OpenSSL

Josuel Racca
Electrical and Electronics Engineering Institute
University of the Philippines, Diliman

1. Environment of the set up
    a. Operating System: Windows 7 Ultimate 64-bit (6.1, Build 7601)
    b. OpenSSL for Windows from https://indy.fulgan.com/SSL/
        i. Extracted file added on path




        ii. Input OpenSSL call from CMD prompt,




2. Encryption of 512x512 Color (24-bit) Lena image http://www.ece.rice.edu/~wakin/images/lena512color.tiff)
    a. Unencrypted hex file of the image

```
Offset(h)  00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
00000000   4D 4D 00 2A 00 0C 00 08 E2 89 7D E2 89 7D DF 89
00000010   85 DF 88 80 E2 8A 78 E2 81 74 E4 8A 7B E3 86 7C
00000020   E3 8C 7F E1 88 77 E4 87 7E E1 86 79 DF 82 6C E2
00000030   8B 77 DF 87 78 DD 81 72 DD 86 6C DD 83 71 DE 8A
00000040   79 DE 8B 72 DF 7F 6D DF 84 69 E0 81 66 DD 86 6D
00000050   DA 83 6E DD 85 71 DF 82 6C E1 7D 62 DD 82 79 DD
00000060   81 6E DC 7E 79 DE 83 6D E1 7E 67 DE 86 6D E2 80
```

b. AES-128 ECB with key = josuel

```
OpenSSL> aes-128-ecb -e -in C:\cipher\lena512color.tiff -out C:\cipher\lenaAES128ECB.tiff -k josuel
```

```
OpenSSL> aes-128-ecb -e -in C:\cipher\lena512color.tiff -out C:\cipher\lenaAES128ECB.tiff -k josuel
OpenSSL>
```

| Offset(h) | 00 | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 0A | 0B | 0C | 0D | 0E | 0F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 00000000 | 53 | 61 | 6C | 74 | 65 | 64 | 5F | 5F | 4C | 38 | 7C | 7B | E1 | B2 | 70 | 80 |
| 00000010 | F3 | 70 | A6 | 76 | 68 | 2E | 18 | 09 | CC | 3B | 6C | E8 | 90 | 68 | 63 | 96 |
| 00000020 | 91 | E8 | 59 | 50 | 24 | 2F | 8F | 52 | 85 | 22 | 28 | 6D | A5 | 9F | BC | D1 |
| 00000030 | 18 | D7 | B7 | 48 | C0 | 17 | E0 | 17 | 72 | 86 | 87 | 96 | 2D | E9 | E7 | 68 |
| 00000040 | 3F | AD | A7 | DE | 86 | 2F | B7 | 6B | 53 | FA | 38 | 60 | 96 | A2 | D5 | FD |
| 00000050 | 79 | FD | 4D | A5 | A7 | 57 | A8 | 44 | 0A | 9E | B3 | B7 | 99 | FD | EC | 88 |

c. AES-128 CBC with key = josuel

```
OpenSSL> aes-128-cbc -e -in C:\cipher\lena512color.tiff -out C:\cipher\lenaAES128CBC.tiff -k josuel
```

```
OpenSSL> aes-128-cbc -e -in C:\cipher\lena512color.tiff -out C:\cipher\lenaAES128cbc.tiff -k josuel
OpenSSL>
```

| Offset(h) | 00 | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 0A | 0B | 0C | 0D | 0E | 0F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 00000000 | 53 | 61 | 6C | 74 | 65 | 64 | 5F | 5F | C8 | 80 | 32 | FE | A3 | 21 | 30 | 95 |
| 00000010 | 0E | 79 | 9E | 73 | EA | A5 | 8F | D0 | 34 | D1 | FA | 87 | 40 | AA | D6 | 5E |
| 00000020 | 85 | DC | 92 | 6A | D9 | 83 | EC | 05 | 51 | F3 | 4D | E2 | F1 | 4B | 2C | 22 |
| 00000030 | 0F | BF | 4F | D6 | 00 | 55 | 5C | 86 | C7 | AE | 75 | 44 | FF | E6 | 1F | 14 |
| 00000040 | 91 | F1 | 36 | 0E | 9D | 22 | CF | 97 | BF | 73 | 6C | E2 | 54 | 95 | 8A | 90 |
| 00000050 | 09 | 2F | 59 | 4C | 0C | 85 | 9B | 55 | 6D | 53 | 61 | 35 | 72 | 09 | D2 | A0 |

3. HASHING the same Lena 512x512 image using the following hash functions:

a. SHA-1

```
OpenSSL> dgst -sha1 C:\Cipher\lena512color.tiff
```

Output

```
SHA1(C:\Cipher\lena512color.tiff)=
e647d0f6736f82e498de8398eccc48cf0a7d53b9
```

b. SHA-256

```
OpenSSL> dgst -sha256 C:\Cipher\lena512color.tiff
```

Output

```
SHA256(C:\Cipher\lena512color.tiff)=
c056da23302d2fb0d946e7ffa11e0d94618224193ff6e2f78ef8097bb8a3569b
```

c. SHA-512

```
OpenSSL> dgst -sha512 C:\Cipher\lena512color.tiff
```

Output

```
SHA512(C:\Cipher\lena512color.tiff)=
2cb9d7df53eb8640dc48d736974f472a98d9c7186de7a972490455f5f3ed29dfc5b7
5c95ccb3ed4596bc2bfc4b1e52cf4d76bcee27d334dd155bb426617392dc
```

4. Public key Encryption
    a. RSA encryption on the Lena 512x512 image, using RSA-2048
        i. Generate Private Key with passphrase = josuel

```
OpenSSL> genrsa -aes128 -out C:\Cipher\private.pem 2048
Loading 'screen' into random state - done
Generating RSA private key, 2048 bit long modulus
.....................................................
.....................
.....................................................++
+
.....................................+++
unable to write 'random state'
e is 65537 (0x10001)
Enter pass phrase for C:\Cipher\private.pem:
Verifying - Enter pass phrase for C:\Cipher\private.pem:
```

```
Offset(h)  00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
00000000   2D 2D 2D 2D 2D 42 45 47 49 4E 20 52 53 41 20 50  -----BEGIN RSA P
00000010   52 49 56 41 54 45 20 4B 45 59 2D 2D 2D 2D 2D 0A  RIVATE KEY-----.
00000020   50 72 6F 63 2D 54 79 70 65 3A 20 34 2C 45 4E 43  Proc-Type: 4,ENC
00000030   52 59 50 54 45 44 0A 44 45 4B 2D 49 6E 66 6F 3A  RYPTED.DEK-Info:
00000040   20 41 45 53 2D 31 32 38 2D 43 42 43 2C 35 44 32   AES-128-CBC,5D2
00000050   37 39 41 46 43 45 45 35 45 37 33 36 45 33 39 38  79AFCEE5E736E398
00000060   34 44 46 34 43 39 45 32 39 39 38 34 43 0A 0A 77  4DF4C9E29984C  w
```

        ii. Generate Public Key with input private key generated above.

```
OpenSSL> rsa -in C:\Cipher\private.pem -out
C:\Cipher\public.pem -outform PEM -pubout
Enter pass phrase for C:\Cipher\private.pem:
writing RSA key
```

```
Offset(h)  00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
00000000   2D 2D 2D 2D 2D 42 45 47 49 4E 20 50 55 42 4C 49  -----BEGIN PUBLI
00000010   43 20 4B 45 59 2D 2D 2D 2D 2D 0A 4D 49 49 42 49  C KEY-----.MIIBI
00000020   6A 41 4E 42 67 6B 71 68 6B 69 47 39 77 30 42 41  jANBgkqhkiG9w0BA
00000030   51 45 46 41 41 4F 43 41 51 38 41 4D 49 49 42 43  QEFAAOCAQ8AMIIBC
00000040   67 4B 43 41 51 45 41 77 77 73 32 67 7A 6D 58 61  gKCAQEAwws2gzmXa
00000050   73 55 45 6A 6F 76 38 4D 45 67 7A 0A 46 59 71 7A  sUEjov8MEgz.FYqz
00000060   48 38 58 6F 66 77 55 74 34 30 54 73 32 54 52 69  H8XofwUt40Ts2TRi
```

        iii. Since direct encryption will result to “data too large for key size”

```
OpenSSL> rsautl -encrypt -inkey C:\Cipher\public.pem -pubin -in C:\Cipher\lena51
2color.tiff -out C:\Cipher\lenaRSAenc.tiff
Loading 'screen' into random state - done
RSA operation error
180:error:0406D06E:rsa routines:RSA_padding_add_PKCS1_type_2:data too large for
key size:.\crypto\rsa\rsa_pk1.c:151:
error in rsautl
```

Lets encrypt the image with AES using SHA-256 as a key, then use RSA on the key.

SHA256 of Lena image to be use as key.

```
Offset(h)  00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
00000000   63 30 35 36 64 61 32 33 33 30 32 64 32 66 62 30  c056da23302d2fb0
00000010   64 39 34 36 65 37 66 66 61 31 31 65 30 64 39 34  d946e7ffa11e0d94
00000020   36 31 38 32 32 34 31 39 33 66 66 36 65 32 66 37  618224193ff6e2f7
00000030   38 65 66 38 30 39 37 62 62 38 61 33 35 36 39 62  8ef8097bb8a3569b
```

```
OpenSSL> aes-128-cbc -e -in C:\Cipher\lena512color.tiff -out C:\Cipher\lenaSHA25
6enc.tiff -kfile C:\Cipher\sha256key.txt
```

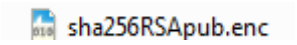
```
OpenSSL> aes-128-cbc -e -in C:\Cipher\lena512color.tiff -
out C:\Cipher\lenaSHA256enc.tiff -kfile
C:\Cipher\sha256key.txt
```

iv. Encrypt the SHA-256 key with Public key generated on 4.a.ii “public.pem”

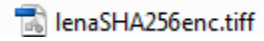
```
OpenSSL> rsautl -encrypt -inkey C:\Cipher\public.pem -pubin -in C:\Cipher\sha256
key.txt -out C:\Cipher\sha256RSApub.enc
Loading 'screen' into random state - done
```

```
OpenSSL> rsautl -encrypt -inkey C:\Cipher\public.pem -
pubin -in C:\Cipher\sha256
key.txt -out C:\Cipher\sha256RSApub.enc
Loading 'screen' into random state - done
```

the Two output files are:



--> the RSA encrypted key using public key



--> the lena encrypted file by AES-126-CBC using SHA-256 as key.

b. Generate an ECDSA signature on the same Lena image

i. First choose a curve by “ecparam –list_curves”

```
OpenSSL> ecparam -list_curves
  secp112r1 : SECG/WTLS curve over a 112 bit prime field
  secp112r2 : SECG curve over a 112 bit prime field
  secp128r1 : SECG curve over a 128 bit prime field
  secp128r2 : SECG curve over a 128 bit prime field
  secp160k1 : SECG curve over a 160 bit prime field
  secp160r1 : SECG curve over a 160 bit prime field
  secp160r2 : SECG/WTLS curve over a 160 bit prime field
  secp192k1 : SECG curve over a 192 bit prime field
  secp224k1 : SECG curve over a 224 bit prime field
  secp224r1 : NIST/SECG curve over a 224 bit prime field
  secp256k1 : SECG curve over a 256 bit prime field
  secp384r1 : NIST/SECG curve over a 384 bit prime field
  secp521r1 : NIST/SECG curve over a 521 bit prime field
  prime192v1: NIST/X9.62/SECG curve over a 192 bit prime field
  prime192v2: X9.62 curve over a 192 bit prime field
  prime192v3: X9.62 curve over a 192 bit prime field
  prime239v1: X9.62 curve over a 239 bit prime field
  prime239v2: X9.62 curve over a 239 bit prime field
  prime239v3: X9.62 curve over a 239 bit prime field
  prime256v1: X9.62/SECG curve over a 256 bit prime field
  sect113r1 : SECG curve over a 113 bit binary field
  sect113r2 : SECG curve over a 113 bit binary field
  sect131r1 : SECG/WTLS curve over a 131 bit binary field
  sect131r2 : SECG curve over a 131 bit binary field
  sect163k1 : NIST/SECG/WTLS curve over a 163 bit binary field
  sect163r1 : SECG curve over a 163 bit binary field
  sect163r2 : NIST/SECG curve over a 163 bit binary field
  sect193r1 : SECG curve over a 193 bit binary field
  sect193r2 : SECG curve over a 193 bit binary field
  sect233k1 : NIST/SECG/WTLS curve over a 233 bit binary field
  sect233r1 : NIST/SECG/WTLS curve over a 233 bit binary field
  sect239k1 : SECG curve over a 239 bit binary field
  sect283k1 : NIST/SECG curve over a 283 bit binary field
  sect283r1 : NIST/SECG curve over a 283 bit binary field
  sect409k1 : NIST/SECG curve over a 409 bit binary field
  sect409r1 : NIST/SECG curve over a 409 bit binary field
  sect571k1 : NIST/SECG curve over a 571 bit binary field
  sect571r1 : NIST/SECG curve over a 571 bit binary field
  c2pnb163v1: X9.62 curve over a 163 bit binary field
  c2pnb163v2: X9.62 curve over a 163 bit binary field
  c2pnb163v3: X9.62 curve over a 163 bit binary field
```

ii. Generate a private key (I choose secp128r1)

```
OpenSSL> ecparam -genkey -name secp128r1 -noout -out C:\Cipher\priECDSA.pem
Loading 'screen' into random state - done
unable to write 'random state'
```

```
OpenSSL> ecparam -genkey -name secp128r1 -noout -out
C:\Cipher\priECDSA.pem
Loading 'screen' into random state - done
unable to write 'random state'
```

```
Offset(h)  00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F

00000000  2D 2D 2D 2D 2D 42 45 47 49 4E 20 45 43 20 50 52  -----BEGIN EC PR
00000010  49 56 41 54 45 20 4B 45 59 2D 2D 2D 2D 2D 0A 4D  IVATE KEY-----.M
00000020  45 51 43 41 51 45 45 45 4D 2B 70 76 65 54 47 78  EQCAQEEEM+pveTGx
00000030  41 59 45 54 55 6D 58 6D 75 32 68 35 4F 47 67 42  AYETUmXmu2h5OGgB
00000040  77 59 46 4B 34 45 45 41 42 79 68 4A 41 4D 69 41  wYFK4EEAByhJAMiA
00000050  41 51 78 47 44 68 71 67 48 61 64 51 77 55 6F 0A  AQxGDhqgHadQwUo.
00000060  6C 51 47 4A 70 6D 50 71 2F 6F 35 46 75 48 7A 45  lQGJpmPq/o5FuHzE
00000070  64 46 49 72 46 67 32 33 71 6D 31 7A 50 41 3D 3D  dFIrFg23qm1zPA==
00000080  0A 2D 2D 2D 2D 2D 45 4E 44 20 45 43 20 50 52 49  .-----END EC PRI
00000090  56 41 54 45 20 4B 45 59 2D 2D 2D 2D 2D 0A        VATE KEY-----.
```

iii. Generate a public key

```
OpenSSL> ec -in C:\Cipher\priECDSA.pem -pubout -out C:\Cipher\pubECDSA.pem
read EC key
writing EC key
```

```
OpenSSL> ec -in C:\Cipher\priECDSA.pem -pubout -out
C:\Cipher\pubECDSA.pem
read EC key
writing EC key
```

```
Offset(h)  00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F

00000000  2D 2D 2D 2D 2D 42 45 47 49 4E 20 50 55 42 4C 49  -----BEGIN PUBLI
00000010  43 20 4B 45 59 2D 2D 2D 2D 2D 0A 4D 44 59 77 45  C KEY-----.MDYwE
00000020  41 59 48 4B 6F 5A 49 7A 6A 30 43 41 51 59 46 4B  AYHKoZIzj0CAQYFK
00000030  34 45 45 41 42 77 44 49 67 41 45 4D 52 67 34 61  4EEABwDIgAEMRg4a
00000040  6F 42 32 6E 55 4D 46 4B 4A 55 42 69 61 5A 6A 36  oB2nUMFKJUBiaZj6
00000050  76 36 4F 52 62 68 38 78 48 52 53 0A 4B 78 59 4E  v6ORbh8xHRS.KxYN
00000060  74 36 70 74 63 7A 77 3D 0A 2D 2D 2D 2D 2D 45 4E  t6ptczw=.-----EN
00000070  44 20 50 55 42 4C 49 43 20 4B 45 59 2D 2D 2D 2D  D PUBLIC KEY----
00000080  2D 0A                                            -.
```

5. Decrypting the images:

a. Decrypting AES-128-ECB

```
OpenSSL> aes-128-ecb -d -in C:\cipher\lenaAES128ECB.tiff -out
C:\cipher\DeclenaAES128ECB.tiff -k josuel
```

b. Decrypting AES-128-CBC

```
OpenSSL> aes-128-cbc -d -in C:\cipher\lenaAES128CBC.tiff -out
C:\cipher\DeclenaAES128CBC.tiff -k josuel
```

c. Decrypting RSA for the SHA-256 Key

```
OpenSSL> rsautl -decrypt -inkey C:\Cipher\private.pem -in
C:\Cipher\sha256RSApub.enc -out C:\Cipher\decSHA265key.txt
Loading 'screen' into random state - done
Enter pass phrase for C:\Cipher\private.pem:
```

Decrypting the Image file using the decrypted SHA-256 key and the public key

```
OpenSSL> aes-128-cbc -d -in C:\Cipher\lenaSHA256enc.tiff -out
C:\Cipher\DeclenaSHA256enc.tiff -kfile C:\Cipher\decSHA265key.txt
```

6. Output Images:
   Upper left = Original
   Upper right = AES-128-ECB
   Lower left = AES-128-CBC
   Lower Right = RSA on the key and AES-128-CBC on the Image File