This assignment is done by mohammed Ammaruddin

Q1

```
ammaruddin@unt-sec:~$ cat plaintext.txt
this is my secret message
ammaruddin@unt-sec:~$
```

Q2

```
ammaruddin@unt-sec:~$ openssl enc -aes-256-ctr -pass pass:am3033 -
pbkdf2 -in plaintext.txt -out ciphertext.bin
ammaruddin@unt-sec:~$ hexdump -C ciphertext.bin
00000000  53 61 6c 74 65 64 5f 5f  a0 a1 e4 56 8f e6 67 a3  |Salte
d__...V..g.|
00000010  b4 11 37 e9 f5 5e 8f 96  f5 95 43 7e 8f 42 97 10  |..7..
^....C~.B..|
00000020  38 1b b5 a8 65 b6 3f 52  ef e0                    |8...e
.?R..|
0000002a
```

Q3

```
ammaruddin@unt-sec:~$ openssl enc -aes-256-ctr -pass pass:am3033 -
pbkdf2 -d -in ciphertext.bin -out plaintext_dec.txt
ammaruddin@unt-sec:~$ cat plaintext_dec.txt
this is my secret message
ammaruddin@unt-sec:~$
```

Q4

```
ammaruddin@unt-sec:~$ openssl enc -aes-256-ctr -a -pass pass:am303
3 -pbkdf2 -in plaintext.txt -out ciphertext.txt
ammaruddin@unt-sec:~$ cat ciphertext.txt
U2FsdGVkX18etw2IiofnXA/sjSzd3X03pNdfjNqQ8WLXE4aDBa/BZbEY
ammaruddin@unt-sec:~$ openssl enc -aes-256-ctr -d -a -pass pass:am
3033 -pbkdf2 -in ciphertext.txt -out plaintext_dec2.txt
ammaruddin@unt-sec:~$ openssl enc -aes-256-ctr -d -a -pass pass:am
3033 -pbkdf2 -in ciphertext.txt -out plaintext_dec2.txt
ammaruddin@unt-sec:~$
```

```
ammaruddin@unt-sec:~$ openssl enc -aes-256-ctr -d -a -pass pass:am
3033 -pbkdf2 -in ciphertext.txt -out plaintext_dec2.txt
ammaruddin@unt-sec:~$ cat plaintext.txt
this is my secret message
```

Q5

Due to the fact that OpenSSL creates a separate initialization vector (IV) for each encryption operation, the ciphertexts in ciphertext.txt and ciphertext2.txt are distinct from one another. The encryption key that is used to encrypt the plaintext is generated using the IV and the password in conjunction with each other. Even with the same password, a new IV is created each time the encryption is executed, therefore the ciphertexts that are produced will vary.

The salt, key, and IV used in the encryption procedure can be printed out with the "-p" option in the decryption command. We can verify that ciphertext.txt and ciphertext2.txt were encrypted with various IVs and produce various ciphertexts by including this option in the decryption command for both files.

```
ammaruddin@unt-sec:~$ openssl enc -aes-256-ctr -a -pass pass:am303
3 -pbkdf2 -in plaintext.txt -out ciphertext2.txt
ammaruddin@unt-sec:~$ openssl enc -aes-256-ctr -d -a -pass pass:am
3033 -pbkdf2 -in ciphertext2.txt
this is my secret message
ammaruddin@unt-sec:~$ cat ciphertext.txt ciphertext2.txt
U2FsdGVkX18KDN2ktm7VZacVhB3Z5LehZEsZYNQkCVgTRTHGjDApADng
U2FsdGVkX1//9rGb1ShyAVAyPRY9vIWG6kGTjjC+U39Pv9yKsVdMpQGz
ammaruddin@unt-sec:~$
```

Q6

This RSA public key has a 3072-bit length. A public-key encryption algorithm that is frequently used is RSA, and the strength of an RSA key depends on the length of the key. In general, stronger encryption results from longer keys.

A very high level of security is offered by a 3072-bit RSA key against brute-force attacks and other attempts to factorize the key. It is advised for the majority of applications that need for secure encryption because it is thought to be a strong key size.

A 3072-bit RSA key, in conclusion, provides a high level of security and is appropriate for the majority of applications that demand secure encryption.

```
39bzgYikyXqFkyPsqafSG+LmNcliFbG5MbuKpcWYZQtisFQi10w9i7/80XJL+fPp
kLrzvCsFXS7T/1vx0Y+ydP89WivS0v5XcMwGeyDdhq4n3LkCgcBIN3FC+4spb0Ri
IICOylyDQUnpUysmBedRG45T+bltSmj8DblLV+w8QxDJ3ywvT1heOMJI5la31pGy
v8cWRh/Ok7C+L74AEni4DfQYxcTxDUepj4uGSibeqR2Vup5IiQETBLdiHkv/NVBF
lj4ytesaKDeja7QlM9ADO2YCTaG6Klhm7Khum+k3CkhYKFcyOCS8tDjffA2ILray
An5OERE1EcR0QNMTfBwPC/3+BIsjPTJiL8aPv35P6j7wzVLbP7M=
-----END RSA PRIVATE KEY-----
ammaruddin@unt-sec:~$ openssl rsa -in am3033.key -pubout -out am30
33_pk.key
writing RSA key
ammaruddin@unt-sec:~$ cat am3033_pk.key
-----BEGIN PUBLIC KEY-----
MIIBojANBgkqhkiG9w0BAQEFAAOCAY8AMIIBigKCAYEA1kYFhJZscy+XEA1Udv1n
d5xbOXnsvnXe5dpzhSuqY0nEpKg9uBer7fcV8msid2/APCGJvmVijpG+nJOzgj47
Du6rp02/dQ09SxArQ5xMnYNDS3xPQ92IKyYN8l9mXd2BbnwStsgJt1DxNZcMJ4Vi
WmL6AAjC+nmOUMb44541zSzAyG/TbasXULdY/Vq9ByXXy3lGkkItMGFXjXl2e4Al
xUajdFUbN25MjeLmzYh+/Lke8uCLt6EILBIgGv+CxUhSAXeBdrN1OeUealvhyJMq
gwS+Oil8ACNhg75P3+CN9gh7SNDo3Rr3BVuu883Ug0hNnt7vxq6LeDCWhXVtz5TM
xk6IgSTzTd4TOxa17NQ6Dj6bgdPx0p+mg3mGA03feU/2ZACCMTwUChKHGoJFJMOV
eMr/A3rFy2D8QRcuFwrC/NAZzL5oGokeOh4xmap/YqH8uBup218faN6TE0byMm4z
R9fXQyW/iGVsI+kq/dlej2Cj/5eLvozm24oXld7AUjAtAgMBAAE=
-----END PUBLIC KEY-----
ammaruddin@unt-sec:~$
```

Q7

```
-----END PUBLIC KEY-----
ammaruddin@unt-sec:~$ openssl rsautl -encrypt -pubin -inkey am3033
_pk.key -in plaintext.txt -out rsa_ciphertext.bin
ammaruddin@unt-sec:~$ openssl rsautl -decrypt -inkey am3033.key -i
n rsa_ciphertext.bin -out rsa_plaintext_dec.txt
ammaruddin@unt-sec:~$ cat rsa_plaintext_dec.txt
this is my secret message
ammaruddin@unt-sec:~$
```

Q8

```
0000060 bfb2 b01d 784d 210e 19b4 cf9a fdc3 25fb
0000070 3b16 f247 fb7f bb05 9aac fd24 9cf5 e324
0000080 7eae 348e 60db bf5e a867 32b5 5b69 f0c0
0000090 3c68 b2a4 eee4 888a 374d 4332 45f9 6644
00000a0 3576 40e1 0433 b66f c2c6 feaa e701 0a45
00000b0 a0a6 0a7f 53d7 0ac5 c89b f784 54b4 f225
00000c0 2bd3 427d f32a 5351 7fc9 8511 f229 58af
00000d0 7f97 4046 72d7 8a53 db6a 068c d568 2a1c
00000e0 dcea 45f4 c9f4 6299 3ca9 bbe0 4a5f 9bef
00000f0 bcaf 8158 e938 719f 164b 10bd ee43 433b
0000100 e4e4 a5f7 f69d 7764 3b77 9b58 5f7e e476
0000110 ef49 54ce 1141 49af 8474 ce35 6d37 a41d
0000120 2650 0898 1d59 652a 298b 4c88 b857 0e98
0000130 573b 61d0 c6dc b11b 688f 38eb ba16 1a2c
0000140 d19b 6be1 7c62 c6a6 393e 86cc 5bf3 6ac9
0000150 b745 f0bf 872e d5e3 9a78 2f27 72d7 fddb
0000160 41d5 9150 38c8 7fb6 b528 9651 f94d 3012
0000170 2f14 a401 2b58 543a ac87 ca2f 5c00 20af
0000180
ammaruddin@unt-sec:~$ openssl dgst -verify am3033_pk.key -signatur
e sig.bin plaintext.txt
Verified OK
ammaruddin@unt-sec:~$
```

Q9

Verification failed

```
ammaruddin@unt-sec:~$ cat plaintext.txt
this is my secret message
ammaruddin@unt-sec:~$ echo "that is my secret message" > plaintext
.txt
ammaruddin@unt-sec:~$ cat plaintext.txt
that is my secret message
ammaruddin@unt-sec:~$ openssl dgst -verify am3033_pk.key -signatur
e sig.bin plaintext.txt
Verification Failure
ammaruddin@unt-sec:~$
```

Q10

```
ammaruddin@unt-sec:~$ openssl req -text -in am3033_domain.csr -noo
ut -verify
verify OK
Certificate Request:
    Data:
        Version: 1 (0x0)
        Subject: C = US, ST = Texas, L = Denton, O = UNT, OU = CSE
, CN = www.am3033.edu
        Subject Public Key Info:
            Public Key Algorithm: rsaEncryption
                RSA Public-Key: (3072 bit)
                Modulus:
                    00:d6:46:05:84:96:6c:73:2f:97:10:0d:54:76:fd:
                    67:77:9c:5b:39:79:ec:be:75:de:e5:da:73:85:2b:
                    aa:63:49:c4:a4:a8:3d:b8:17:ab:ed:f7:15:f2:6b:
                    22:77:6f:c0:3c:21:89:be:65:62:8e:91:be:9c:93:
                    b3:82:3e:3b:0e:ee:ab:a7:4d:bf:75:0d:3d:4b:10:
                    2b:43:9c:4c:9d:83:43:4b:7c:4f:43:dd:88:2b:26:
                    0d:f2:5f:66:5d:dd:81:6e:7c:12:b6:c8:09:b7:50:
                    f1:35:97:0c:27:85:62:5a:62:fa:00:08:c2:fa:79:
```

Q11

```
ammaruddin@unt-sec:~$ openssl s_client -connect google.com:443 -sh
owcerts </dev/null | more
depth=2 C = US, O = Google Trust Services LLC, CN = GTS Root R1
verify return:1
depth=1 C = US, O = Google Trust Services LLC, CN = GTS CA 1C3
verify return:1
depth=0 CN = *.google.com
verify return:1
CONNECTED(00000003)
---
Certificate chain
 0 s:CN = *.google.com
   i:C = US, O = Google Trust Services LLC, CN = GTS CA 1C3
-----BEGIN CERTIFICATE-----
MIIOOTCCDSGgAwIBAgIQBggoxxmFbYEKYeHW3ZLfgzANBgkqhkiG9w0BAQsFADBG
MQswCQYDVQQGEwJVUzEiMCAGA1UEChMZR29vZ2xlIFRydXN0IFNlcnZpY2VzIExM
QzETMBEGA1UEAxMKR1RTIENBIDFDMzAeFw0yMzAzMjgxNjQ3MzNaFw0yMzA2MjAx
NjQ3MzJaMBcxFTATBgNVBAMMDCouZ29vZ2xlLmNvbTBZMBMGByqGSM49AgEGCCqG
SM49AwEHA0IABOeBTg0lojDOW27zDLgwpQpCewXWx2dXRQhP0sYHmkUKMYHcBICe
DONE
LVuRpC2xAZcJEMPlqnRCduNk4t4LzsPXs4CjggwbMIIMFzAOBgNVHQ8BAf8EBAMC
```

```
---
Certificate chain
 0 s:CN = *.google.com
   i:C = US, O = Google Trust Services LLC, CN = GTS CA 1C3
-----BEGIN CERTIFICATE-----
MIIOOTCCDSGgAwIBAgIQBggoxxmFbYEKYeHW3ZLfgzANBgkqhkiG9w0BAQsFADBG
MQswCQYDVQQGEwJVUzEiMCAGA1UEChMZR29vZ2xlIFRydXN0IFNlcnZpY2VzIExM
QzETMBEGA1UEAxMKR1RTIENBIDFDMzAeFw0yMzAzMjgxNjQ3MzNaFw0yMzA2MjAx
NjQ3MzJaMBcxFTATBgNVBAMMDCouZ29vZ2xlLmNvbTBZMBMGByqGSM49AgEGCCqG
SM49AwEHA0IABOeBTg0lojDOW27zDLgwpQpCewXWx2dXRQhP0sYHmkUKMYHcBICe
DONE
LVuRpC2xAZcJEMPlqnRCduNk4t4LzsPXs4CjggwbMIIMFzAOBgNVHQ8BAf8EBAMC
B4AwEwYDVR0lBAwwCgYIKwYBBQUHAwEwDAYDVR0TAQH/BAIwADAdBgNVHQ4EFgQU
JPAQqwmJhQbj9DoSN6gyPa295TMwHwYDVR0jBBgwFoAUinR/r4XN7pXNPZzQ4kYU
83E1HScwagYIKwYBBQUHAQEEXjBcMCcGCCsGAQUFBzABhhtodHRwOi8vb2NzcC5w
a2kuZ29vZy9ndHMxYzMwMQYIKwYBBQUHMAKGJWh0dHA6Ly9wa2kuZ29vZy9yZXBv
L2NlcnRzL2d0czFjMy5kZXIwggnNBgNVHREEggnEMIIJwIIMKi5nb29nbGUuY29t
ghYqLmFwcGVuZ2luZS5nb29nbGUuY29tggkqLmJkbi5kZXaCFSoub3JpZ2luLXRl
c3QuYmRuLmRldoISKi5jbG91ZC5nb29nbGUuY29tghgqLmNyb3dkc291cmNlLmdv
b2dsZS5jb22CGCouZGF0YWNvbXB1dGUuZ29vZ2xlLmNvbYILKi5nb29nbGUuY2GC
--More--
```

```
issuer=C = US, O = Google Trust Services LLC, CN = GTS CA 1C3

---
No client certificate CA names sent
Peer signing digest: SHA256
Peer signature type: ECDSA
Server Temp Key: X25519, 253 bits
---
SSL handshake has read 6776 bytes and written 382 bytes
Verification: OK
---
New, TLSv1.3, Cipher is TLS_AES_256_GCM_SHA384
Server public key is 256 bit
Secure Renegotiation IS NOT supported
Compression: NONE
Expansion: NONE
No ALPN negotiated
Early data was not sent
Verify return code: 0 (ok)
---
ammaruddin@unt-sec:~$
```

Q12

```
The key fingerprint is:
SHA256:E94pZSXHphCNqKIkC/2dWzCf/3gpQ9E7xaKmmNZkOMA ammaruddin@unt-
sec
The key's randomart image is:
+---[RSA 3072]----+
|       ..+..o    |
|      . o .+o    |
| . . .  ..o+ .   |
|o.o E o. =o.o o  |
|+o o o *S.oo +   |
|o   . = *o+ o    |
|       X =   o   |
|      = o +.o    |
|     .    .=.    |
+----[SHA256]-----+
ammaruddin@unt-sec:~$
```

Q13

```
Welcome to Ubuntu 20.04.5 LTS (GNU/Linux 5.15.0-69-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

 * Introducing Expanded Security Maintenance for Applications.
   Receive updates to over 25,000 software packages with your
   Ubuntu Pro subscription. Free for personal use.

     https://ubuntu.com/pro

0 updates can be applied immediately.


The list of available updates is more than a week old.
To check for new updates run: sudo apt update
Your Hardware Enablement Stack (HWE) is supported until April 2025
.
Last login: Fri Mar 10 10:54:34 2023 from 10.0.2.2
ammaruddin@unt-sec:~$
```