Q1

md5um

```
[wen.dong@ALMORG901P-W01 ~]$ echo -n "Harry Porter" | md5sum
bd5475f99084b8c7e7721e80f0030c85  -
[wen.dong@ALMORG901P-W01 ~]$
```

Hash_comp.py

```
dong23@charlie: ~/uwinds3/networking_mon/lab6/crypto
dong23@charlie:~/uwinds3/networking_mon/lab6/crypto$ python3 hash_comp.py "Harry
 Porter"
SHA512(Harry Porter)=1799cc40a787be112d4e891adcae9f2a56e2f8a436fcece6311ddc46688
cdf214b663a2d7a1ab28c094029422a18ce0b3b3490df0f7a0dc0b448a2e1f9ab14ca
SHA224(Harry Porter)=e50e7bf49b769291f5ab1698adcd13eeb283aaef5f54d355e03c7d8b
MD5(Harry Porter)=caccbc643153b5dab69ab59b868540f5
MD5(Harry PorterAlice in Wonderland)=e447be44fee0dbfadf5df9be95bcd340
dong23@charlie:~/uwinds3/networking_mon/lab6/crypto$
```

2.

Generate private RSA key

```
dong23@charlie:~/uwinds3/networking_mon/lab6/crypto$ openssl genrsa -aes128 -out
 private.pem 1024
Generating RSA private key, 1024 bit long modulus (2 primes)
....+++++
.......+++++
e is 65537 (0x010001)
Enter pass phrase for private.pem:
Verifying - Enter pass phrase for private.pem:
dong23@charlie:~/uwinds3/networking_mon/lab6/crypto$
```

Extract public key

```
dong23@charlie:~/uwinds3/networking_mon/lab6/crypto$ openssl rsa -in private.pem -pubout >public.pem
Enter pass phrase for private.pem:
writing RSA key
dong23@charlie:~/uwinds3/networking_mon/lab6/crypto$ ls -l
total 10
-rwxr-xr-x 1 dong23 temp 312 Jul  8 05:22 decrypt_RSA.py
-rwxr-xr-x 1 dong23 temp 297 Jul  8 05:22 encrypt_RSA.py
-rwxr-xr-x 1 dong23 temp 662 Jul  8 05:22 endec_AES.py
-rwxr-xr-x 1 dong23 temp 665 Jul  8 05:22 hash_comp.py
-rw------- 1 dong23 temp 986 Jul 10 21:52 private.pem
-rw-r--r-- 1 dong23 temp 272 Jul 10 21:56 public.pem
-rwxr-xr-x 1 dong23 temp 350 Jul  8 05:22 sign_RSA.py
-rwxr-xr-x 1 dong23 temp 464 Jul  8 05:22 verify_RSA.py
dong23@charlie:~/uwinds3/networking_mon/lab6/crypto$
```

Display private key

```
dong23@charlie:~/uwinds3/networking_mon/lab6/crypto$ openssl rsa -in private.pem -text -noout
Enter pass phrase for private.pem:
RSA Private-Key: (1024 bit, 2 primes)
modulus:
    00:b0:ad:51:7e:a7:9b:60:fe:0d:0d:1a:87:19:af:
    01:c1:27:2c:2d:91:2b:10:f7:86:da:cd:0e:cc:df:
    79:e7:5f:c2:32:3f:06:fb:00:5d:0e:4f:3e:d8:70:
    ba:d8:5e:b7:69:47:4a:bb:e7:41:fd:b2:1c:b5:31:
    f4:26:36:5d:c5:bb:b5:2d:be:3e:b1:55:1e:d9:4c:
    ad:f4:fd:3e:c7:7d:be:6f:1c:f8:ed:a2:b8:f4:e7:
    d6:78:82:e2:a6:63:2f:dc:19:a0:be:96:a2:5f:dc:
    82:ae:3e:32:88:46:9f:9e:22:03:f8:15:33:d9:78:
    be:99:38:6a:0e:9e:86:b7:45
publicExponent: 65537 (0x10001)
privateExponent:
    00:91:46:a2:e9:53:66:97:75:af:43:a4:19:8d:0b:
    f3:94:60:e2:99:c1:71:b9:2c:e0:2d:90:49:e8:3a:
    a6:61:93:c2:fa:50:0b:2c:5f:9b:25:06:12:76:25:
    13:ee:90:cb:9d:fc:ea:3d:e2:25:c8:37:2a:15:f0:
    32:83:2d:57:22:bf:51:38:f1:d1:18:42:91:35:98:
    f5:ed:80:88:fb:27:de:fa:6f:81:79:3d:70:b5:86:
    a7:b2:ab:14:87:f4:e4:0b:ea:8d:b4:ec:a7:e5:b6:
    8f:81:b8:75:33:71:0f:9b:51:f6:5b:23:03:53:85:
    d4:c8:b9:2b:8e:2f:6e:a2:81
prime1:
    00:df:42:e0:b3:41:bc:f8:0f:8b:4b:63:2a:5d:d6:
    62:79:ed:d1:fa:53:79:35:66:53:7b:ef:b6:cf:43:
    f6:20:04:a1:94:69:15:46:cf:ce:7a:cf:be:cf:08:
    f8:90:ba:cd:09:cf:6c:28:03:12:5e:c6:d9:15:d4:
```

```
    24:3f:c8:81:25
prime2:
    00:ca:95:b0:e3:8c:99:97:ab:56:ca:ec:c4:6d:03:
    1b:14:27:ef:a7:71:3f:79:3b:87:6c:cc:55:32:46:
    e3:b6:8d:cb:02:47:f1:9f:4d:d8:b5:53:69:29:4e:
    79:28:9b:da:19:4b:6f:1e:21:ff:12:36:a7:72:6a:
    a1:c8:4c:d3:a1
exponent1:
    48:b0:e3:ac:39:a9:27:33:18:6d:51:3a:48:17:37:
    34:ef:c3:c0:37:51:6d:9b:85:6f:02:db:88:9c:5e:
    14:8a:ad:79:3e:c5:98:aa:ac:55:bc:32:2a:02:87:
    bb:c7:b6:5e:8d:54:6f:aa:a0:5e:8f:6b:ba:f9:b9:
    5b:b3:12:a9
exponent2:
    60:c3:ef:a2:41:5e:7c:1b:d3:71:4a:76:e8:bb:3b:
    0b:fd:a7:73:8b:9c:8e:03:e9:44:06:b6:0b:35:e1:
    3f:29:ab:76:83:76:de:69:08:c2:53:fd:3f:45:c4:
    89:a9:28:1c:3c:f3:ee:a8:be:75:ac:b5:7e:e9:80:
    2d:74:c5:01
coefficient:
    18:ff:c3:0e:aa:9e:c6:85:21:32:18:cc:34:74:e2:
    a3:47:50:f0:a3:43:0f:d1:b4:3b:d4:4b:bb:6b:cc:
    01:d0:c2:25:e8:25:45:62:7d:a7:74:ec:95:3b:f1:
    b7:54:43:a6:02:a3:60:43:e7:9a:e2:9a:ea:53:8a:
    80:6d:7c:d7
```

Display public key

```
dong23@charlie:~/uwinds3/networking_mon/lab6/crypto$ openssl rsa -in public.pem
-pubin -text -noout
RSA Public-Key: (1024 bit)
Modulus:
    00:b0:ad:51:7e:a7:9b:60:fe:0d:0d:1a:87:19:af:
    01:c1:27:2c:2d:91:2b:10:f7:86:da:cd:0e:cc:df:
    79:e7:5f:c2:32:3f:06:fb:00:5d:0e:4f:3e:d8:70:
    ba:d8:5e:b7:69:47:4a:bb:e7:41:fd:b2:1c:b5:31:
    f4:26:36:5d:c5:bb:b5:2d:be:3e:b1:55:1e:d9:4c:
    ad:f4:fd:3e:c7:7d:be:6f:1c:f8:ed:a2:b8:f4:e7:
    d6:78:82:e2:a6:63:2f:dc:19:a0:be:96:a2:5f:dc:
    82:ae:3e:32:88:46:9f:9e:22:03:f8:15:33:d9:78:
    be:99:38:6a:0e:9e:86:b7:45
Exponent: 65537 (0x10001)
dong23@charlie:~/uwinds3/networking_mon/lab6/crypto$
```

3. encrypt & decrypt with RSA

Encrypt

```
dong23@charlie:~/uwinds3/networking_mon/lab6$ python3 encrypt_RSA.py
input the message you would like to encrypt:Wen Dong #110057395
<_RSAobj @0x7f69969bb550 n(1024),e>
dong23@charlie:~/uwinds3/networking_mon/lab6$ ls -l
total 3412
-rw-r--r-- 1 dong23 temp     128 Jul 10 22:36 ciphertext.bin
drwxr-xr-x 2 dong23 temp       8 Jul 10 22:28 crypto
-rw-r--r-- 1 dong23 temp     316 Jul 10 22:32 decrypt_RSA.py
-rw-r--r-- 1 dong23 temp     351 Jul 10 22:31 encrypt_RSA.py
-rw-r--r-- 1 dong23 temp   71594 Jul 10 22:19 Lab-6-anwsers.docx
-rw-r--r-- 1 dong23 temp  962222 Jul  8 05:22 Lab-6.pdf
-rw-r--r-- 1 dong23 temp 2296171 Jul  8 05:22 L_CRYPTO.pptx
-rw------- 1 dong23 temp     986 Jul 10 21:52 private.pem
-rw-r--r-- 1 dong23 temp     272 Jul 10 21:56 public.pem
dong23@charlie:~/uwinds3/networking_mon/lab6$ python decrypt_RSA.py
Wen Dong #110057395
dong23@charlie:~/uwinds3/networking_mon/lab6$ hexdump -C ciphertext.bin
00000000  0d df 96 0b 42 2f ed 5f  c9 20 3a a9 4b 4f 19 97  |....B/._. :.KO..|
00000010  a9 13 eb f8 aa a4 f3 0e  84 2b eb 85 f9 e0 26 e3  |.........+....&.|
00000020  04 2b 14 7c 6b 1d 11 86  2a 8e 9b 99 44 70 78 59  |.+.|k...*...DpxY|
00000030  56 b1 13 c0 7f c8 54 06  44 96 a5 12 ee 94 b8 eb  |V.....T.D.......|
00000040  59 19 b8 bd 52 15 0f 2c  69 15 94 ea 93 5b b4 96  |Y...R..,i....[..|
00000050  72 40 84 e7 8d e7 9d 25  67 04 4b 5d 0b f4 36 4b  |r@.....%g.K]..6K|
00000060  cc 8f 22 2b bd 26 d8 0d  64 c0 55 45 9a af 2f d3  |..."+.&..d.UE../.|
00000070  a6 1d b6 03 9f f5 4f bb  44 8a 70 31 0e d5 a7 5b  |......O.D.p1...[|
00000080
```

Decrypt

```
dong23@charlie:~/uwinds3/networking_mon/lab6$ python3 encrypt_RSA.py
input the message you would like to encrypt:Wen Dong #110057395
<_RSAobj @0x7f69969bb550 n(1024),e>
dong23@charlie:~/uwinds3/networking_mon/lab6$ ls -l
total 3412
-rw-r--r-- 1 dong23 temp     128 Jul 10 22:36 ciphertext.bin
drwxr-xr-x 2 dong23 temp       8 Jul 10 22:28 crypto
-rw-r--r-- 1 dong23 temp     316 Jul 10 22:32 decrypt_RSA.py
-rw-r--r-- 1 dong23 temp     351 Jul 10 22:31 encrypt_RSA.py
-rw-r--r-- 1 dong23 temp   71594 Jul 10 22:19 Lab-6-anwsers.docx
-rw-r--r-- 1 dong23 temp  962222 Jul  8 05:22 Lab-6.pdf
-rw-r--r-- 1 dong23 temp 2296171 Jul  8 05:22 L_CRYPTO.pptx
-rw------- 1 dong23 temp     986 Jul 10 21:52 private.pem
-rw-r--r-- 1 dong23 temp     272 Jul 10 21:56 public.pem
dong23@charlie:~/uwinds3/networking_mon/lab6$ python decrypt_RSA.py
Wen Dong #110057395
dong23@charlie:~/uwinds3/networking_mon/lab6$
```

4.

RSA signature sign - $2000

```
dong23@charlie:~/uwinds3/networking_mon/lab6$ python sign_RSA.py
("hexdigest - 'I owe you $2000'", '43211516ffb74dd8150d4f6ea4901bba22bd90b0c7074
6105bfb62fc11155127')
dong23@charlie:~/uwinds3/networking_mon/lab6$ ls -l
total 4245
-rw-r--r-- 1 dong23 temp     128 Jul 10 22:36 ciphertext.bin
drwxr-xr-x 2 dong23 temp       8 Jul 10 22:28 crypto
-rw-r--r-- 1 dong23 temp     316 Jul 10 22:32 decrypt_RSA.py
-rw-r--r-- 1 dong23 temp     351 Jul 10 22:31 encrypt_RSA.py
-rw-r--r-- 1 dong23 temp  411656 Jul 10 22:46 Lab-6-anwsers.docx
-rw-r--r-- 1 dong23 temp  287780 Jul 10 22:46 Lab-6-anwsers.pdf
-rw-r--r-- 1 dong23 temp  962222 Jul  8 05:22 Lab-6.pdf
-rw-r--r-- 1 dong23 temp 2296171 Jul  8 05:22 L_CRYPTO.pptx
-rw------- 1 dong23 temp     986 Jul 10 21:52 private.pem
-rw-r--r-- 1 dong23 temp     272 Jul 10 21:56 public.pem
-rw-r--r-- 1 dong23 temp     128 Jul 10 22:47 signature.bin
-rw-r--r-- 1 dong23 temp     387 Jul 10 22:46 sign_RSA.py
-rw-r--r-- 1 dong23 temp     464 Jul 10 22:46 verify_RSA.py
dong23@charlie:~/uwinds3/networking_mon/lab6$ hexdump -C signature
hexdump: signature: No such file or directory
dong23@charlie:~/uwinds3/networking_mon/lab6$ hexdump -C signature.bin
00000000  48 70 f8 8c 3a f8 34 79  6c d4 34 31 76 08 63 b4  |Hp..:.4yl.41v.c.|
00000010  f3 1e db 5d 64 86 3e 4b  78 e0 ab 33 13 03 95 48  |...]d.>Kx..3...H|
00000020  c1 df 80 bc c7 6c e7 c8  51 3c 33 f6 cc a3 27 7b  |.....l..Q<3...'{|
00000030  81 dc 1d 1a b2 15 fa 62  e9 b6 ef 5c c3 6e 28 99  |.......b...\.n(.|
00000040  9f 71 9a 8b 20 18 fe b8  53 63 0f 94 ac 77 a0 e0  |.q.. ...Sc...w..|
00000050  34 48 2c c3 31 f7 4a 2c  e7 b8 5b e9 e0 96 6c f4  |4H,.1.J,..[...l.|
00000060  72 48 2b 21 46 f1 a8 e3  0b d0 89 2b 17 37 89 6d  |rH+!F......+.7.m|
00000070  11 fe 99 ff 5e dc b8 c0  23 a8 bd b8 c6 ef 4d f0  |....^...#.....M.|
00000080
dong23@charlie:~/uwinds3/networking_mon/lab6$
```

RSA signature sign - $3000

```
dong23@charlie:~/uwinds3/networking_mon/lab6$ python sign_RSA.py
("hexdigest - 'I owe you $3000'", '4db055d6ddb7a26c117d5baac32b0ada4cbc3601848b54da42095300c5f0e
e9f')
dong23@charlie:~/uwinds3/networking_mon/lab6$ hexdump -C signature3000.bin
00000000  0c ad 6d 67 64 ba 82 a6  13 26 5d 76 6b ef 6e 34  |..mgd....&]vk.n4|
00000010  52 ac b9 b4 8c 03 db ff  62 ab 9a 10 a9 0b c0 7b  |R.......b......{|
00000020  94 04 69 1d b8 08 0c 79  46 6a 83 ee 36 24 01 54  |..i....yFj..6$.T|
00000030  1d a5 0f 36 0b 08 d2 b6  43 9b 76 50 de e5 a1 82  |...6....C.vP....|
00000040  db 65 64 01 c7 93 86 22  e3 dd 14 72 75 b2 1c cd  |.ed...."...ru...|
00000050  b7 68 8e 90 19 13 1c 09  ff 69 62 dc 69 1d b6 7a  |.h.......ib.i..z|
00000060  29 bb d7 fa 87 c9 3b 2a  0e 9c 4a 4a 54 f4 aa 9e  |).....;*..JJT...|
00000070  bc c9 2a c8 3f ae 63 6f  bb 08 42 40 03 e5 58 86  |..*.?.co..B@..X.|
00000080
dong23@charlie:~/uwinds3/networking_mon/lab6$
```

The experiment shows that signatures are different for messages of slightly different.

Verify the signature

```
dong23@charlie:~/uwinds3/networking_mon/lab6$ python verify_RSA.py
4db055d6ddb7a26c117d5baac32b0ada4cbc3601848b54da42095300c5f0ee9f
The signature is valid.
dong23@charlie:~/uwinds3/networking_mon/lab6$
```

5.

```
dong23@charlie:~/uwinds3/networking_mon/lab6$ python endec_AES.py
sk - ec19ba016bb8d44154590f5ee6a2a436
c1 - 42deb5760fedf4b6ad26867eecad5ad38f2528c73989a533402f06ab0445636a59636c64fc8
8336a1da100163bca82df53193658fdc21ad82fb2c2a47cbd2d5672679187763201073000500a43a
68e59426accc7e2085264a13753420a9b7bdbacbc207f7d9a4fe9b9478ada12c8dc77d149a49ee4f
7831f0badd067db42bf91
c2: 78e9c4ff9d36a1fb868d77f0ddfe8085447309434197582935519d5556e21f11c3b9dac79035
2a3f7b444003fb185c95

decrypted c1 (sk) - ec19ba016bb8d44154590f5ee6a2a436
decrypted msg - I find the solution for P not equal NP
dong23@charlie:~/uwinds3/networking_mon/lab6$
```

Entire source code

```
#!/usr/bin/python3

from Crypto.Cipher import AES
from Crypto.PublicKey import RSA
from Crypto.Cipher import PKCS1_OAEP
from Crypto.Util import Padding
from Crypto.Random import get_random_bytes

import binascii

def encryptRSA(message, keyfile):
    key=RSA.importKey(open(keyfile).read())
    cipher=PKCS1_OAEP.new(key)
    ciphertext=cipher.encrypt(message)
    return ciphertext

def decryptRSA(ciphertext, privatekeyfile):
    key_str = open(privatekeyfile).read()
    prikey = RSA.importKey(key_str, passphrase='111111')
    cipher = PKCS1_OAEP.new(prikey)
    message = cipher.decrypt(ciphertext)
    return message
```

```
key_hex_string = '00112233445566778899AABBCCDDEEFF00112233445566778899AABBCCDDEEF
F'
key = bytearray.fromhex(key_hex_string)

# a.
iv = get_random_bytes(16)
print("sk - {}".format(binascii.hexlify(bytearray(iv))))

# b.
c1 = encryptRSA(iv, "public.pem")
print("c1 - {}".format(binascii.hexlify(bytearray(c1))))

data = b'I find the solution for P not equal NP'

# c.
cipher = AES.new(key, AES.MODE_CBC, iv)
c2 = cipher.encrypt(Padding.pad(data, 16))
print("c2: {0}\n".format(binascii.hexlify(bytearray(c2))))


# d.
decrpted_message = decryptRSA(c1, "private.pem")
print("decrypted c1 (sk) - {}".format(binascii.hexlify(bytearray(decrpted_message
))))

# Decrypt the ciphertext
cipher = AES.new(key, AES.MODE_CBC, iv)
plaintext = cipher.decrypt(c2)
print("decrypted msg - {0}".format(Padding.unpad(plaintext, 16)))
```