Riley Heike
Lab 8: Encryption

Secret Key Encryption

- Generated shift key: “wolxjthgmincdypfrkvsqebuza”
- `“This is a secret file that has important information which we do not want to reveal" -> “Tgmv mv w vjlkjs tmcj sgws gwv mdfpkswys mytpkdwsmpy bgmlg bj xp yps bwys sp kjejwc”`

Frequency Analysis

UZQSOVUOHXMOPVGPOZPEVSGZWSZOPFPESXUDBMETSXAIZVUEPHZHMDZSHZOWS
FPAPPDTSVPQUZWYMXUZUHSXEPYEPOPDZSZUFPOMBZWPFUPZHMDJUDTMOHMQ

```
[rheike@linux22414 frequency analysis]$ cat ciphertext | fold -w1 | sort | uniq -c | sort -nr
        16 P
        14 Z
        10 U
        10 S
        9 O
        8 M
        7 H
        6 E
        6 D
        5 X
        5 V
        4 W
        4 F
        3 T
        3 Q
        2 Y
        2 G
        2 B
        2 A
        1 J
        1 I
```

Cipher Breaking:
Final substitution mapping

Cipher	Plain
A	B
B	F
C	C
D	N
E	R
F	V
G	Y
H	C
I	U
J	G
K	K
L	L
M	O
N	N
O	S
P	E
Q	W
R	R
S	A
T	M
U	I
V	D
W	H
X	L
Y	P
Z	T

ITWASDISCLOSEDYESTERDAYTHATSEVERALINFORMALBUTDIRECTCONTACTSHAVEBE
ENMADEWITHPOLITICALREPRESENTATIVESOFTHEVIETCONGINMOSCOW

Advanced Encryption Standard

```
[rheike@linux22414 AES]$ openssl enc -aes-128-ecb -e -in plaintext -out ciphertext -k
00112233445566778899AABBCCDDEEFF
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.

[rheike@linux22414 AES]$ cat ciphertext
Salted__�+��_3^��"{��mk}�9q+��g_Y�8ɤ�(ha��HU�s*�N+���#�7�xk

                                        �15��Ɔ0��{�+��Ċ�Ư1]�{(([[[[[[[[[[[[[r[[[[[
[rheike@linux22414 AES]$ openssl enc -aes-128-ecb -d -in ciphertext -out plaintxt -k
00112233445566778899AABBCCDDEEFF
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.

[rheike@linux22414 AES]$ cat plaintxt
This is a secret file that has important information which we do not want to reveal
```

Public Key Encryption

```
[rheike@linux22414 RSA]$ openssl genrsa -aes128 -out privatekey 1024
Enter PEM pass phrase:
Verifying - Enter PEM pass phrase:
[rheike@linux22414 RSA]$ openssl rsa -in privatekey -noout -text
Enter pass phrase for privatekey:
Private-Key: (1024 bit, 2 primes)
modulus:
    00:9f:ac:17:52:3f:19:80:7d:c4:21:cd:47:1c:27:
    fa:ea:c3:3c:11:1e:af:1d:d3:02:40:fc:1a:b9:3f:
    1a:76:36:c3:56:26:d0:a9:15:37:47:cf:e7:ff:92:
    61:c9:20:d5:02:d6:3c:6b:74:3f:c3:68:c6:09:4b:
    c0:7e:1f:3b:a6:85:55:08:08:f0:cd:2a:fd:ee:66:
    35:e8:fb:32:37:f5:ac:a8:b6:37:24:91:eb:7d:42:
    eb:65:28:ce:76:8a:a5:71:37:5c:a4:13:3e:86:29:
    61:a6:db:47:af:f6:3e:c5:68:b7:cd:7a:53:a4:a0:
    74:fd:35:12:96:bc:06:2e:cd
publicExponent: 65537 (0x10001)
privateExponent:
    68:41:cd:d8:7e:2b:00:a3:1d:f5:94:3b:e2:3e:98:
    af:c1:5a:ef:32:c1:d5:0f:7a:61:44:3b:8e:c9:8d:
    55:b2:dc:48:dc:7f:52:67:ef:f8:8b:e0:48:18:24:
    91:57:46:be:db:74:08:15:97:ac:d8:34:b6:cd:27:
    9b:32:79:97:71:3b:57:a1:82:78:f3:58:cc:58:6b:
    6e:30:d7:ba:74:2b:b4:72:e1:d9:d9:1a:73:c4:ba:
    45:8f:82:0b:0d:e3:50:c0:e1:db:8a:45:e2:ee:a5:
    a0:b7:34:62:d8:6f:2a:b0:2e:3a:d2:13:d9:e1:9c:
    66:c8:8a:a1:78:d8:40:61
prime1:
    00:d1:0c:f8:12:2f:77:f9:1d:16:52:cb:3b:95:8b:
    97:1a:53:cb:50:14:dd:e2:8f:71:c6:06:e9:f3:33:
    f2:58:75:68:47:be:f8:b6:5b:c6:bb:63:0e:fb:25:
    32:e4:d9:44:ae:7f:b2:8c:90:e1:7f:49:0f:28:25:
    83:d2:f9:36:1f
```

```
prime2:
    00:c3:88:30:55:ef:5b:dc:3f:52:03:ac:1c:05:c0:
    c9:a8:57:0a:df:b4:fc:1a:34:32:5a:c7:cc:c0:4d:
    72:89:4d:30:7d:99:62:29:3c:a4:fb:f1:32:49:f8:
    1e:e6:4f:9c:c6:e5:7b:5d:16:f8:9a:8d:a8:0c:ea:
    e8:7c:72:85:93
exponent1:
    44:bd:8c:fc:fd:da:e7:71:67:1b:c6:74:4b:52:61:
    57:68:e2:5b:ec:e0:a1:55:25:c6:46:13:bb:c3:03:
    17:8f:53:c0:f3:cc:f8:b9:e8:f9:49:33:6d:e5:e7:
    7c:54:ed:3e:ac:02:dc:31:ef:d4:59:03:c0:e1:c5:
    1d:24:91:65
exponent2:
    70:32:24:32:1b:2f:65:98:bb:d1:b9:9f:36:b9:e1:
    bc:83:7d:8c:d1:c7:da:ad:5a:bb:76:6c:09:68:27:
    31:9b:a6:18:5b:bb:d4:97:a4:bf:a0:2d:cf:fd:dc:
    95:20:d7:7f:d5:4b:cd:25:92:2e:f4:db:99:d5:ec:
    e3:03:bf:9b
coefficient:
    00:a5:18:d3:53:c5:b7:ed:13:05:a9:f9:07:fb:09:
    82:e7:d5:8c:5a:ea:d7:88:a7:e4:17:b8:50:d6:4c:
    63:e4:fa:99:f4:13:8b:42:71:75:3d:93:ec:95:65:
    67:4b:cb:2b:02:b2:01:09:b9:ea:2e:88:94:fe:2b:
    fb:74:4f:a7:fc
```

```
[rheike@linux22414 RSA]$ openssl rsautl -encrypt -inkey publickey -pubin -in plaintext -out ciphertext
The command rsautl was deprecated in version 3.0. Use 'pkeyutl' instead.

[rheike@linux22414 RSA]$ cat ciphertext
6�f����$��|qH�Dc&�ɷ2|#�hָ��,�,d�j��Zs���c���[�c��%'QH=��ݑ[ǭn7��[W
n�L/�mLA�$�C{�>Z���\�[���x�J��,�Z�B)��,
```

Digital Signature

```
[rheike@linux22414 RSA]$ openssl sha256 -binary plaintext > plaintext.sha256
[rheike@linux22414 RSA]$ xxd plaintext.sha256
00000000: 0c53 f8ec c05c 0792 f3e2 ee23 82b5 27c6  .S...\.....#..'.
00000010: 4eff b3bf 69f8 ff74 2d22 2003 6245 90b6  N...i..t-" .bE..

[rheike@linux22414 RSA]$ openssl rsautl -sign -inkey privatekey -in plaintext.sha256 -out plaintext.sig
```

```
The command rsautl was deprecated in version 3.0. Use 'pkeyutl' instead.
Enter pass phrase for privatekey:
```

[rheike@linux22414 RSA]$ openssl rsautl -verify -inkey publickey -in plaintext.sig -pubin -raw | xxd

```
The command rsautl was deprecated in version 3.0. Use 'pkeyutl' instead.
00000000: 0001 ffff ffff ffff ffff ffff ffff ffff  ................
00000010: ffff ffff ffff ffff ffff ffff ffff ffff  ................
00000020: ffff ffff ffff ffff ffff ffff ffff ffff  ................
00000030: ffff ffff ffff ffff ffff ffff ffff ffff  ................
00000040: ffff ffff ffff ffff ffff ffff ffff ffff  ................
00000050: ffff ffff ffff ffff ffff ffff ffff ff00  ................
00000060: 0c53 f8ec c05c 0792 f3e2 ee23 82b5 27c6  .S...\.....#..'.
00000070: 4eff b3bf 69f8 ff74 2d22 2003 6245 90b6  N...i..t-" .bE..
```

Hash Function

[rheike@linux22414 RSA]$ openssl dgst -sha256 plaintext

```
SHA2-256(plaintext)=
0c53f8ecc05c0792f3e2ee2382b527c64effb3bf69f8ff742d222003624590b6
```

[rheike@linux22414 RSA]$ openssl dgst -md5 plaintext

```
MD5(plaintext)= 39cd5fd7d44fb9df0ca0bca082a7f8ae
```

Description of Steps

1. Check OpenSSL installation:
    - Run `openssl version -a` to confirm OpenSSL is installed.

2. Monoalphabetic encryption (substitution cipher):
    - Create a plaintext file:
        - `echo "This is a secret file that has important information which we do not want to reveal" > plaintext`
    - Encrypt using `tr` with a substitution key:
        - `tr 'a-z' 'qgvmftzyceolhsuwbjaxdnikpr' < plaintext > ciphertext`
    - Decrypt by reversing substitution:
        - `tr 'qgvmftzyceolhsuwbjaxdnikpr' 'a-z' < ciphertext > plaintext`
    - Analyze letter frequency to break cipher:
        - `cat ciphertext | fold -w1 | sort | uniq -c | sort -nr`

3. AES encryption/decryption with OpenSSL:
   - Encrypt using AES-128-ECB:
     - `openssl enc -aes-128-ecb -e -in plaintext -out ciphertext -k 00112233445566778899AABBCCDDEEFF`
   - Decrypt ciphertext:
     - `openssl enc -aes-128-ecb -d -in ciphertext -out plaintxt -k 00112233445566778899AABBCCDDEEFF`

4. RSA key generation and encryption:
   - Generate 1024-bit RSA private key:
     - `openssl genrsa -aes128 -out privatekey 1024`
   - Extract public key:
     - `openssl rsa -in privatekey -pubout -out publickey`
   - Encrypt plaintext with public key:
     - `openssl rsautl -encrypt -inkey publickey -pubin -in plaintext -out ciphertext`
   - Decrypt ciphertext with private key:
     - `openssl rsautl -decrypt -inkey privatekey -in ciphertext`

5. Digital signature with RSA:
   - Create SHA-256 hash of plaintext:
     - `openssl sha256 -binary plaintext > plaintext.sha256`
   - Sign the hash with private key:
     - `openssl rsautl -sign -inkey privatekey -in plaintext.sha256 -out plaintext.sig`
   - Verify signature with public key:
     - `openssl rsautl -verify -inkey publickey -in plaintext.sig -pubin -raw | xxd`

6. Hashing with OpenSSL:
   - Compute SHA-256 hash:
     - `openssl dgst -sha256 plaintext`
   - Compute MD5 hash:
     - `openssl dgst -md5 plaintext`