

Lab Report: Secret Key Encryption Lab

- Task 1: Deriving the Private Key
- Task 2: Encrypting a Message
- Task 3: Decrypting a Message
- Task 4: Signing a Message
- Task 5: Verifying a Signature
 - 5.1:
- Task 6:
- Task 7:

Robert D. Hernandez rherna70@uic.edu

Task 1: Deriving the Private Key

My solution is in `taskone.c`

Let p, q and e be three prime numbers. Let $n = p * q$. We will use (e, n) as the public key.

Given:

$p = \text{F7E75FDC469067FFDC4E847C51F452DF}$ $q = \text{E85CED54AF57E53E092113E62F436F4F}$ $e = \text{0D88C3}$

Calculate the private key d .

First n was calculated to be

" $n = \text{E103ABD94892E3E74AFD724BF28E78366D9676BCCC70118BD0AA1968DBB143D1}$ "

$\phi(n) = (p - 1) * (q - 1)$ $\phi(n) =$

$\text{D2E88FE4EEB33F597D5F1B0D2D96EDD5156C94EC27065CFB268768CA1738BBC4}$ $d =$

$\text{A90C278677B53385F22D2792D6E3A0FC1D1E537FAF4325779577A6BCA11BE65B}$

```

• vscode → /workspaces/CS487/hw6/6_part2 (hw6/rsa-pub-key-ecryption-and-sig-lab) $ task build
task: [build] gcc taskone.c -o taskone -lcrypto
• vscode → /workspaces/CS487/hw6/6_part2 (hw6/rsa-pub-key-ecryption-and-sig-lab) $ task run
task: [run] ./taskone
n= E103ABD94892E3E74AFD724BF28E78366D9676BCCC70118BD0AA1968DBB143D1
phi(n)= D2E88FE4EEB33F597D5F1B0D2D96EDD5156C94EC27065CFB268768CA1738BBC4
d= A90C278677B53385F22D2792D6E3A0FC1D1E537FAF4325779577A6BCA11BE65B

```

Note: Using Wolfram Alpha, the Decimal value of p was found to be

329520679814142392965336341297134588639 Note: Using Wolfram Alpha, the Decimal value of q was

found to be 308863399973593539130925275387286220623 Note: The decimal value of e was small

enough to fit in a programmer's calculator and was found to be 886979 Note: n , $\phi(n)$, and d were

computed using the `<openssl/bn.h>`

Task 2: Encrypting a Message

My solution is in `tasktwo.c`

I found the encrypted version of the message was

"6FB078DA550B2650832661E14F4F8D2CFAEF475A0DF3A75CACDC5DE5CFC5FADC" in hex. I checked that the encryption was correct by decrypting the message.

```
vscode → /workspaces/CS487/hw6/6_part2 (hw6/rsa-pub-key-ecryption-and-sig-
lab) $ task run
task: [run] ./tasktwo
encrypted M=
6FB078DA550B2650832661E14F4F8D2CFAEF475A0DF3A75CACDC5DE5CFC5FADC
decrypted c= 4120746F702073656372657421
BigInteger as string: Decrypted string: A top secret!
```

```
● vscode → /workspaces/CS487/hw6/6_part2 (hw6/rsa-pub-key-ecryption-and-sig-lab) $ task run
task: [run] ./tasktwo
encrypted M= 6FB078DA550B2650832661E14F4F8D2CFAEF475A0DF3A75CACDC5DE5CFC5FADC
decrypted c= 4120746F702073656372657421
BigInteger as string: Decrypted string: A top secret!
○ vscode → /workspaces/CS487/hw6/6_part2 (hw6/rsa-pub-key-ecryption-and-sig-lab) $ █
```

Task 3: Decrypting a Message

Decrypt: C = 8C0F971DF2F3672B28811407E2DABBE1DA0FEBBBDFC7DCB67396567EA1E2493F

My solution is in `taskthree.c` I found the ciphertext to decrypt to "Password is dees"

```
vscode → /workspaces/CS487/hw6/6_part2 (hw6/rsa-pub-key-ecryption-and-sig-
lab) $ task run
task: [run] ./taskthree
decrypted c= 50617373776F72642069732064656573
Decrypted string: Password is dees
```

```
● vscode → /workspaces/CS487/hw6/6_part2 (hw6/rsa-pub-key-ecryption-and-sig-lab) $ task run
task: [run] ./taskthree
decrypted c= 50617373776F72642069732064656573
Decrypted string: Password is dees
○ vscode → /workspaces/CS487/hw6/6_part2 (hw6/rsa-pub-key-ecryption-and-sig-lab) $ █
rsa-pub-key-ecryption-and-sig-lab* ↻ 0 0 0 0 0.37% 0.00 GHz ... 1.31/7.65 GB -- NORM
```

Task 4: Signing a Message

M = I owe you \$2000

We know that: Digital Signature = $m^d \bmod n$

```
● vscode → /workspaces/CS487/hw6/6_part2 (hw6/rsa-pub-key-ecryption-and-sig-lab) $ task run
task: [run] ./taskfour
Hex value of digital signature = D218BC69F10BB671F5D716CC820CE633064F69CFF16878F6EF7BC4DFC963680F
String representation of Digital Signature: 00i0
0q00^
030i00hx00{000ch
○ vscode → /workspaces/CS487/hw6/6_part2 (hw6/rsa-pub-key-ecryption-and-sig-lab) $ █
rsa-pub-key-ecryption-and-sig-lab* ↻ 0 0 0 0 0.62% 0.00 GHz ... 1.30/7.65 GB -- NORMAL --
```

M = I owe you \$3000

```

vscode → /workspaces/CS487/hw6/6_part2 (hw6/rsa-pub-key-ecryption-and-sig-lab) $ task run
task: [run] ./taskfour
Hex value of digital signature = D95E9354A95BD87A74D5FB10C6490F2FF999DD57E4AA84095E88E21A888D3115
String representation of Digital Signature: 0^0T0[0zt000I/000W鞞^00001
vscode → /workspaces/CS487/hw6/6_part2 (hw6/rsa-pub-key-ecryption-and-sig-lab) $ █

```

Comparing the two digital signatures we can see there is a large difference in the hex value even though we only changed one byte of the input string

Task 5: Verifying a Signature

My solution is in taskfive.c

M = "Launch a missile."

Let pubic key: (e, n)

M = Launch a missile. S =

643D6F34902D9C7EC90CB0B2BCA36C47FA37165C0005CAB026C0542CBDB6802F e = 010001 (this hex value equals to decimal 65537) n =

AE1CD4DC432798D933779FBD46C6E1247F0CF1233595113AA51B450F18116115

try retrieve M using

$res = S^e \bmod n$

if res = M => verified

```

task: [build] gcc taskfive.c -o taskfive -lcrypt
vscode → /workspaces/CS487/hw6/6_part2 (hw6/rsa-pub-key-ecryption-and-sig-lab) $ task run
task: [run] ./taskfive
Hex value of m= 4C61756E63682061206D697373696C652E
String representation of m derived from S: Launch a missile.
string1 and string2 are the same.
vscode → /workspaces/CS487/hw6/6_part2 (hw6/rsa-pub-key-ecryption-and-sig-lab) $ █

```

5.1:

Task 6:

Task 7: