

Task 1

```
CryptoRSA1.c      x
1 #include <stdio.h>
2 #include <openssl/bn.h>
3 #define NBITS 256
4 void printBN(char *msg, BIGNUM * a)
5 {
6     char * number_str = BN_bn2hex(a);
7     printf("%s %s\n", msg, number_str);
8     OPENSSL_free(number_str);
9 }
10 void computer_privatekey(){
11     BN_CTX *ctx = BN_CTX_new();
12     BIGNUM *p = BN_new();
13     BIGNUM *q = BN_new();
14     BIGNUM *e = BN_new();
15     BN_hex2bn(&p, "F7E75FDC469067FFDC4E847C51F452DF");
16     BN_hex2bn(&q, "E85CED54AF57E53E092113E62F436F4F");
17     BN_hex2bn(&e, "0D88C3");
18     BIGNUM *n = BN_new();
19
20     BN_mul(n,p,q, ctx);
21     printBN("n = p * q = ", n);
22
23     BIGNUM *totient = BN_new();
24     BIGNUM *p_minus_one = BN_new();
25     BIGNUM *q_minus_one = BN_new();
26
27     BIGNUM *one = BN_new();
28     BN_hex2bn(&one, "1");
29
30     BN_sub(p_minus_one, p, one);
31     BN_sub(q_minus_one, q, one);
32
33     BN_mul(totient,p_minus_one,q_minus_one,ctx);
34     printBN("totient = (p-1)*(q-1) = ", totient);
35
36     BIGNUM *d = BN_new();
37
38     BN_mod_inverse(d,e,totient, ctx);
39     printBN("private key d = ", d);
40 }
41 int main ()
42 {
43     computer_privatekey();
44 }
```

```
(UWLGroup2㉿kali)-[~/Desktop/CryptoRSA Lab UWL]
$ gcc CryptoRSA1.c -o Task1 -lcrypto
(UWLGroup2㉿kali)-[~/Desktop/CryptoRSA Lab UWL]
$ ./Task1
n = p * q = E103ABD94892E3E74AFD724BF28E78366D9676BCCC70118BD0AA1968DBB143D1
totient = (p-1)*(q-1) = E103ABD94892E3E74AFD724BF28E78348D52298BD687C44DEB3A81065A7981A4
private key d = 3587A24598E5F2A21DB007D89D18CC50ABA5075BA19A33890FE7C28A9B496AEB
```

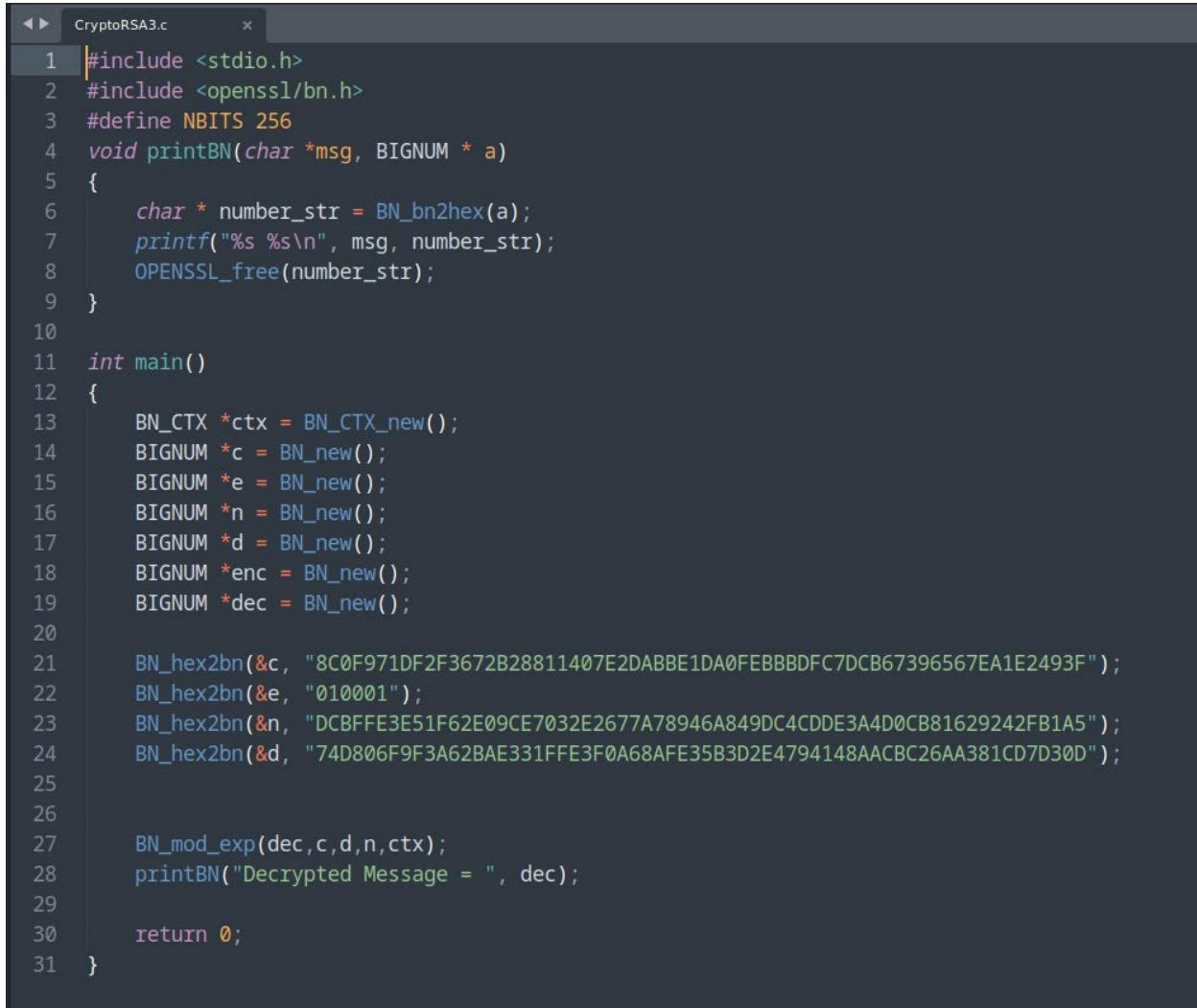
Task 2

```
1 #include <stdio.h>
2 #include <openssl/bn.h>
3 #define NBITS 256
4 void printBN(char *msg, BIGNUM *a)
5 {
6     char * number_str = BN_bn2hex(a);
7     printf("%s %s\n", msg, number_str);
8     OPENSSL_free(number_str);
9 }
10
11 int main()
12 {
13     BN_CTX *ctx = BN_CTX_new();
14     BIGNUM *m = BN_new();
15     BIGNUM *e = BN_new();
16     BIGNUM *n = BN_new();
17     BIGNUM *d = BN_new();
18     BIGNUM *enc = BN_new();
19     BIGNUM *dec = BN_new();
20
21     BN_hex2bn(&m, "4120746f702073656372657421");
22     BN_hex2bn(&e, "010001");
23     BN_hex2bn(&n, "DCBFFE3E51F62E09CE7032E2677A78946A849DC4CDDE3A4D0CB81629242FB1A5");
24     BN_hex2bn(&d, "74D806F9F3A62BAE331FFE3F0A68AFE35B3D2E4794148AACBC26AA381CD7D30D");
25
26     BN_mod_exp(enc,m,e,n,ctx);
27     printBN("Encrypted Message = ", enc);
28
29     BN_mod_exp(dec,enc,d,n,ctx);
30     printBN("Decrypted Message = ", dec);
31
32     return 0;
33 }
```

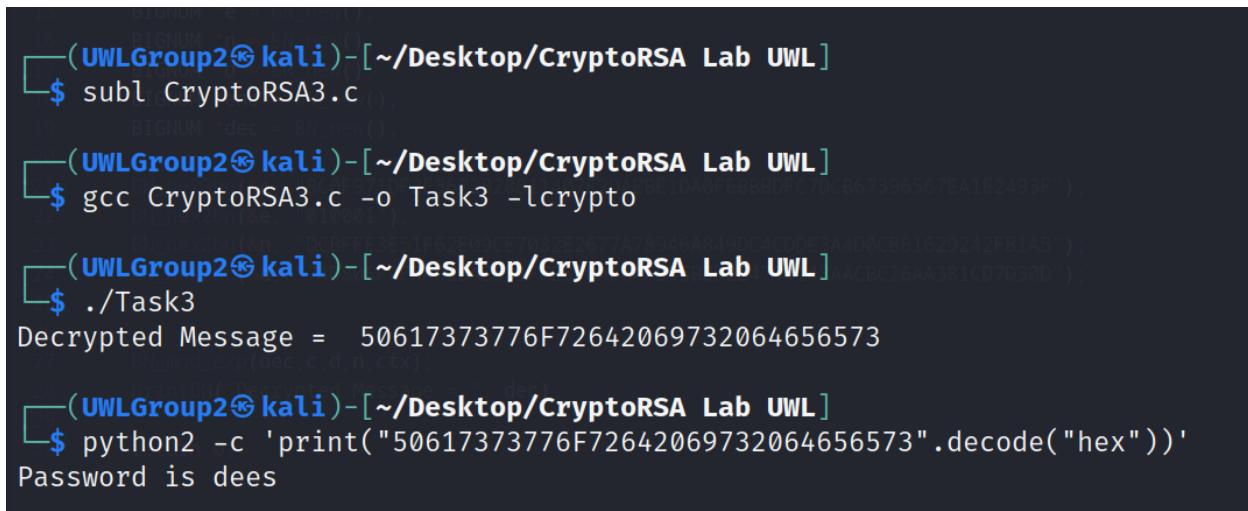
```
(UWLGroup2㉿kali)-[~/Desktop/CryptoRSA Lab UWL] more you are able to hear'
$ python2 -c 'print("A top secret!".encode("hex"))'
4120746f702073656372657421

(UWLGroup2㉿kali)-[~/Desktop/CryptoRSA Lab UWL]
$ python2 -c 'print("4120746f702073656372657421".decode("hex"))'
A top secret!
```

Task 3



```
1 #include <stdio.h>
2 #include <openssl/bn.h>
3 #define NBITS 256
4 void printBN(char *msg, BIGNUM *a)
5 {
6     char * number_str = BN_bn2hex(a);
7     printf("%s %s\n", msg, number_str);
8     OPENSSL_free(number_str);
9 }
10
11 int main()
12 {
13     BN_CTX *ctx = BN_CTX_new();
14     BIGNUM *c = BN_new();
15     BIGNUM *e = BN_new();
16     BIGNUM *n = BN_new();
17     BIGNUM *d = BN_new();
18     BIGNUM *enc = BN_new();
19     BIGNUM *dec = BN_new();
20
21     BN_hex2bn(&c, "8C0F971DF2F3672B28811407E2DABBE1DA0FEBBBDFC7DCB67396567EA1E2493F");
22     BN_hex2bn(&e, "010001");
23     BN_hex2bn(&n, "DCBFFE3E51F62E09CE7032E2677A78946A849DC4CDDE3A4D0CB81629242FB1A5");
24     BN_hex2bn(&d, "74D806F9F3A62BAE331FFE3F0A68AFE35B3D2E4794148AACBC26AA381CD7D30D");
25
26
27     BN_mod_exp(dec, c, d, n, ctx);
28     printBN("Decrypted Message = ", dec);
29
30     return 0;
31 }
```



```
(UWLGroup2㉿kali)-[~/Desktop/CryptoRSA Lab UWL]
$ subl CryptoRSA3.c
$ g++ CryptoRSA3.c -o Task3 -lcrypto
$ ./Task3
Decrypted Message = 50617373776F72642069732064656573
$ python2 -c 'print("50617373776F72642069732064656573".decode("hex"))'
Password is dees
```

Task 4

```
(UWLGroup2㉿kali)-[~/Desktop/CryptoRSA Lab UWL]
$ python2 -c 'print("I owe you $2000.".encode("hex"))'
49206f776520796f752024323030302e
```

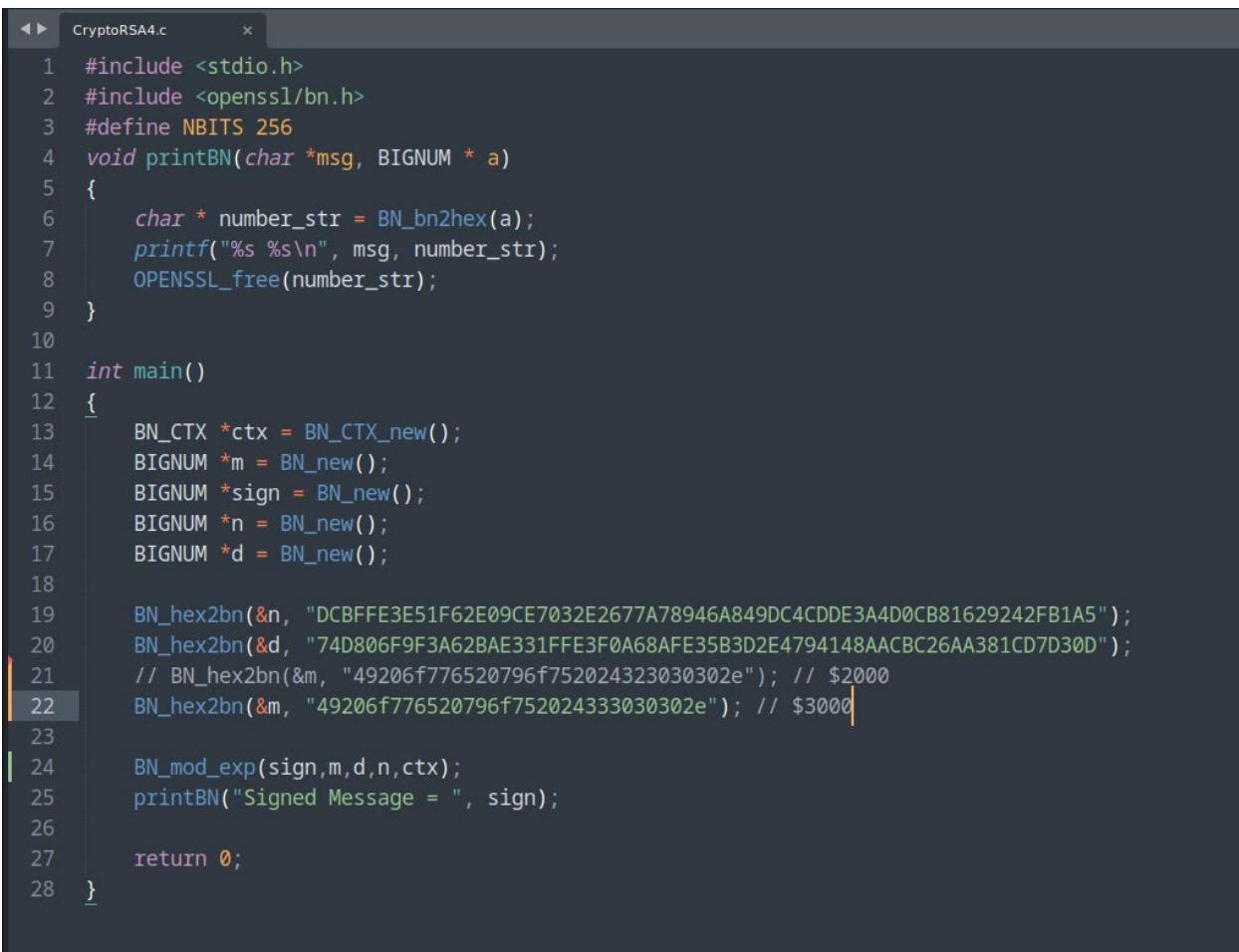
```
CryptoRSA4.c
1 #include <stdio.h>
2 #include <openssl/bn.h>
3 #define NBITS 256
4 void printBN(char *msg, BIGNUM * a)
5 {
6     char * number_str = BN_bn2hex(a);
7     printf("%s %s\n", msg, number_str);
8     OPENSSL_free(number_str);
9 }
10
11 int main()
12 {
13     BN_CTX *ctx = BN_CTX_new();
14     BIGNUM *m = BN_new();
15     BIGNUM *sign = BN_new();
16     BIGNUM *n = BN_new();
17     BIGNUM *d = BN_new();
18
19     BN_hex2bn(&n, "DCBFFE3E51F62E09CE7032E2677A78946A849DC4CDDE3A4D0CB81629242FB1A5");
20     BN_hex2bn(&d, "74D806F9F3A62BAE331FFE3F0A68AFE35B3D2E4794148AACBC26AA381CD7D30D");
21     BN_hex2bn(&m, "49206f776520796f752024323030302e");
22
23     BN_mod_exp(sign,m,d,n,ctx);
24     printBN("Signed Message = ", sign);
25
26     return 0;
27 }
```

```
(UWLGroup2㉿kali)-[~/Desktop/CryptoRSA Lab UWL]
$ gcc CryptoRSA4.c -o Task4 -lcrypto
(UWLGroup2㉿kali)-[~/Desktop/CryptoRSA Lab UWL]
$ ./Task4 ("Signed Message = ", sign)
Signed Message = 55A4E7F17F04CCFE2766E1EB32ADDBA890BBE92A6FBE2D785ED6E73CCB35E4CB
```

Changing from \$2000 to \$3000

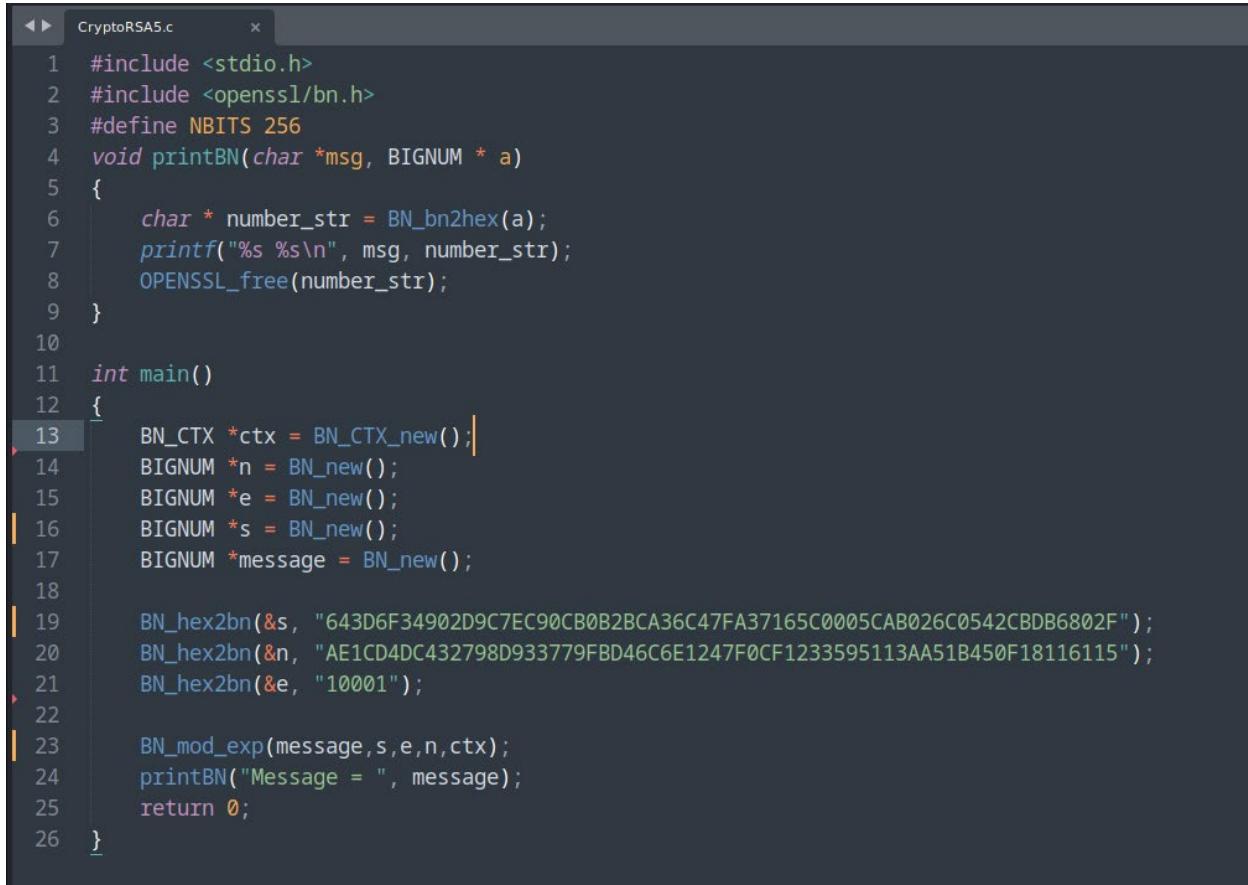
```
(UWLGroup2㉿kali)-[~/Desktop/CryptoRSA Lab UWL] $ python2 -c 'print("I owe you $3000.".encode("hex"))' | hexdump -v -e /%02x/ | tail -n 1
49206f776520796f752024333030302e // $3000

(UWLGroup2㉿kali)-[~/Desktop/CryptoRSA Lab UWL] $ gcc CryptoRSA4.c -o Task4 -lcrypto
(UWLGroup2㉿kali)-[~/Desktop/CryptoRSA Lab UWL] $ ./Task4
Signed Message = BCC20FB7568E5D48E434C387C06A6025E90D29D848AF9C3EBAC0135D99305822
```

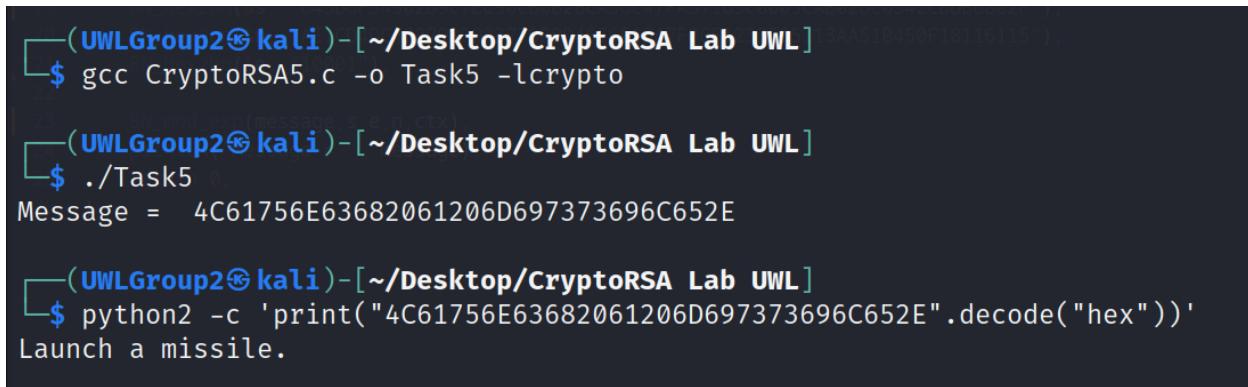


```
CryptoRSA4.c
1 #include <stdio.h>
2 #include <openssl/bn.h>
3 #define NBITS 256
4 void printBN(char *msg, BIGNUM * a)
5 {
6     char * number_str = BN_bn2hex(a);
7     printf("%s %s\n", msg, number_str);
8     OPENSSL_free(number_str);
9 }
10
11 int main()
12 {
13     BN_CTX *ctx = BN_CTX_new();
14     BIGNUM *m = BN_new();
15     BIGNUM *sign = BN_new();
16     BIGNUM *n = BN_new();
17     BIGNUM *d = BN_new();
18
19     BN_hex2bn(&n, "DCBFFE3E51F62E09CE7032E2677A78946A849DC4CDDE3A4D0CB81629242FB1A5");
20     BN_hex2bn(&d, "74D806F9F3A62BAE331FFE3F0A68AFE35B3D2E4794148AACBC26AA381CD7D30D");
21     // BN_hex2bn(&m, "49206f776520796f752024323030302e"); // $2000
22     BN_hex2bn(&m, "49206f776520796f752024333030302e"); // $3000
23
24     BN_mod_exp(sign,m,d,n,ctx);
25     printBN("Signed Message = ", sign);
26
27     return 0;
28 }
```

Task 5



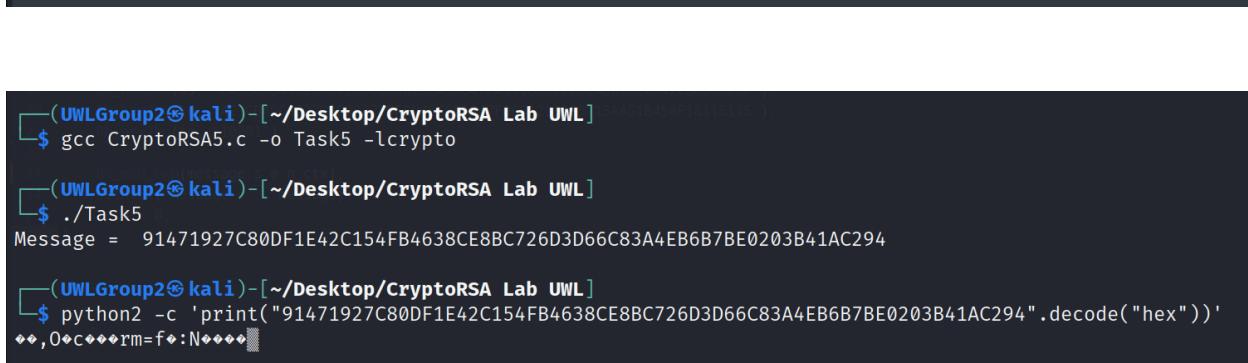
```
CryptoRSA5.c
1 #include <stdio.h>
2 #include <openssl/bn.h>
3 #define NBITS 256
4 void printBN(char *msg, BIGNUM * a)
5 {
6     char * number_str = BN_bn2hex(a);
7     printf("%s %s\n", msg, number_str);
8     OPENSSL_free(number_str);
9 }
10
11 int main()
12 {
13     BN_CTX *ctx = BN_CTX_new();
14     BIGNUM *n = BN_new();
15     BIGNUM *e = BN_new();
16     BIGNUM *s = BN_new();
17     BIGNUM *message = BN_new();
18
19     BN_hex2bn(&s, "643D6F34902D9C7EC90CB0B2BCA36C47FA37165C0005CAB026C0542CBDB6802F");
20     BN_hex2bn(&n, "AE1CD4DC432798D933779FBD46C6E1247F0CF1233595113AA51B450F18116115");
21     BN_hex2bn(&e, "10001");
22
23     BN_mod_exp(message,s,e,n,ctx);
24     printBN("Message = ", message);
25     return 0;
26 }
```



```
(UWLGroup2㉿kali)-[~/Desktop/CryptoRSA Lab UWL]$ gcc CryptoRSA5.c -o Task5 -lcrypto
(UWLGroup2㉿kali)-[~/Desktop/CryptoRSA Lab UWL]$ ./Task5
Message = 4C61756E63682061206D697373696C652E

(UWLGroup2㉿kali)-[~/Desktop/CryptoRSA Lab UWL]$ python2 -c 'print("4C61756E63682061206D697373696C652E".decode("hex"))'
Launch a missile.
```

Changing from 2F to 3F



```
CryptoRSA5.c      *
1 #include <stdio.h>
2 #include <openssl/bn.h>
3 #define NBITS 256
4 void printBN(char *msg, BIGNUM *a)
5 {
6     char * number_str = BN_bn2hex(a);
7     printf("%s %s\n", msg, number_str);
8     OPENSSL_free(number_str);
9 }
10
11 int main()
12 {
13     BN_CTX *ctx = BN_CTX_new();
14     BIGNUM *n = BN_new();
15     BIGNUM *e = BN_new();
16     BIGNUM *s = BN_new();
17     BIGNUM *message = BN_new();
18
19     BN_hex2bn(&s, "643D6F34902D9C7EC90CB0B2BCA36C47FA37165C0005CAB026C0542CBDB6803F");
20     BN_hex2bn(&n, "AE1CD4DC432798D933779FB46C6E1247F0CF1233595113AA51B450F18116115");
21     BN_hex2bn(&e, "10001");
22
23     BN_mod_exp(message,s,e,n,ctx);
24     printBN("Message = ", message);
25     return 0;
26 }
```

```
(UWLGroup2㉿kali)-[~/Desktop/CryptoRSA Lab UWL]
$ gcc CryptoRSA5.c -o Task5 -lcrypto

(UWLGroup2㉿kali)-[~/Desktop/CryptoRSA Lab UWL]
$ ./Task5
Message = 91471927C80DF1E42C154FB4638CE8BC726D3D66C83A4EB6B7BE0203B41AC294

(UWLGroup2㉿kali)-[~/Desktop/CryptoRSA Lab UWL]
$ python2 -c 'print("91471927C80DF1E42C154FB4638CE8BC726D3D66C83A4EB6B7BE0203B41AC294".decode("hex"))'
♦♦,0♦c♦♦♦rm=f♦:N♦♦♦♦
```

Task 6

```
└─(UWLGroup2㉿kali)-[~/Desktop/CryptoRSA Lab UWL]
$ openssl s_client -connect www.tryhackme.com:443 -showcerts
CONNECTED(00000003)
depth=2 C = US, O = Internet Security Research Group, CN = ISRG Root X2
verify return:1
depth=1 C = US, O = Let's Encrypt, CN = E1
verify return:1
depth=0 CN = tryhackme.com
verify return:1
_____
Certificate chain
0 s:CN = tryhackme.com
    i:C = US, O = Let's Encrypt, CN = E1
    a:PKEY: id-ecPublicKey, 256 (bit); sigalg: ecdsa-with-SHA384
    v:NotBefore: Sep 21 02:09:33 2023 GMT; NotAfter: Dec 20 02:09:32 2023 GMT
    BEGIN CERTIFICATE
MIIDjDCCAXKgAwIBAgISAwg749s8VcXcTsesY0RAdCNjMAoGCCqGSM49BAMMDI
CzAJBgNVBAYTAlVTMRYwFAYDVQQKEw1MZXQncyBFbmNyeXB0MQswCQYDVQQDEwJF
MTAeFw0yMzA5MjEwMjA5MzNaFw0yMzEyMjAwMjA5MzJaMBgxFjAUBgNVBAMTDXR
eWhhY2ttZS5jb20wWTATBgcqhkjOPQIBBggqhkjOPQM
BBwNCAARdNFNKlRwgkXfP
11P3xmp
rt18VPWq29eZs6llCrd
sN7Ixw0kAUQXYzrxjjLk
k+lg9epwmrK1XiUfpQ
i+aWJSYWo4ICIDCAhwwDgYDVR0PAQH/BAQDAgeAMB0GA1UDJQQWMBQGCCsGAQUF
BwMBBgrB
gEFBQcDAjAMBgNVHRMBAf8EAjAAMB0GA1UdDgQWB
BT+mkTSBJh305oi
OaIn2AvY3dR7kzAfB
gNVHSMEGDAwBRa8+0r/DbCN3m5UjDqVG/PVcsurDBVBgr
BgEFBQcBAQRJMEcwIQYIKwYBBQUHMAGGFWh0dHA6Ly9lMS5vLmxlbmNyLm9yZzAi
Bgg
rB
gEFBQcAoYWaHR0cDovL2UxLmkubGVuY3Iub3JnLzApB
gNVHREEIjAggg8q
LnRyeWhhY2ttZS5jb22CDXR
yeWhhY2ttZS5jb20wEwYDVR0gBAwwCjAIBgZngQwB
AgEwggEEBgorB
gEEAdZ5AgQCB
IH1BIHyAPAA
dQB6MoxU2LcttiDqOOBSHumEFnAy
E4VN09IrwTpXo1LrUgAAAYq1ttcMAAAE
AwBGMEQCIB72SVnMWiiLUXCfcoatddkG
EDgVn/A+ASC5ZpsqpytDAiBC4aTOYUUFI/21loxf/ZtNmzK4u4MICQvwqDskge1Z
VwB3AOg+0No+9QY1MudXKL
yJa8kD08vREWvs62nhd31tBr1uAAABirW21vAAAAQD
AEgwRgIhAOF8+NTH+xApb8Dtq8XSX1PgTDiN7+16CpYTTy
rpjhXFAiEA1p/j70xJ
```

```
CryptoRSA6.c | Signature | Certificate1.pem | Certificate2.pem | Certificate3.pem | Certificate4.pem
1 -----BEGIN CERTIFICATE-----
2 MIIDjDCCAxKgAwIBAgISAwg749s8VcXcTsesY0RAdCNjMAoGCCqGSM49BAMMDMIX
3 CzAJBgNVBAYTA1VTMRVwFAYDVQQKEw1MZXQnCYBFbmNyeXB0MQswCQYDVQQDEwJF
4 MTAeFw0yMzA5MjEwMjA5MzNaFw0yMzEyMjAwMjA5MzJaMBgxFjAUBgNVBAMTDXRy
5 eWhhY2ttZS5jb20wWTATBgcqhkJOPQIBBggqhkJOPQMBlwNCARDNFNK1RwgkXfP
6 11P3xmprt18VPWq29eZs6llCrdsN7Ixw0kAUQXYzxjjLkk+lg9epwmrK1X1ufpQ
7 i+aWJSYWo4ICIDCCAhwwDgYDVR0PAQH/BAQDAgeAMB0GA1UdJQQWMBQGCCsGAQUF
8 BwMBBgrBgEFBQcDajAMBvNHRMBAf8EAjAAMB0GA1UdDgQWBTT+mkTSBjh305oi
9 OaIn2AvY3dR7kzAfBgNVHSMEGDAWgBra8+0r/DbCN3m5UjDqVG/PVcsurDBVBgr
10 BgEFBQcBAQRJMEcwIqYIKwYBBQUHMAggFwh0dHA6ly91MS5vLmx1bmNyLm9yZzAi
11 BggxBgEFBQcwAoYWaHR0cDovL2UxLmkubGVuY3Iub3JnLzApBgNVHREEIjAggg8q
12 LnRyeWhhY2ttZS5jb22CDXRyeWhhY2ttZS5jb20wEwYDVR0gBAwwCjAIBgZngQwB
13 AgEwgEEBgorBgEEAdZ5AgQCBIH1BIHyAPAAQDQ86MoxU2LcttiDqO0BSHunEFnAy
14 E4VN09IrwTpXo1LrUgAAAYq1ttcMAAAEawBGMEQCIB72SVnMwiILUXfcotddkG
15 EDgVn/A+ASC5ZpsqpytDAiBC4aTOUUFI/21loxf/ZtNmzK4u4MICQvwqDskge1Z
16 VwB3A0g+0No+9QY1MudXKLjJa8kD08vREWvs62nhd31tBr1uAAABirW21vAAAAQD
17 AEgwRgIhAOF8+NTH+xApb8Dtq8XSX1PgTDiN7+16CpYTtyrpjHXFAiEA1p/j70xJ
18 thW0cs9H2ETjwqFnls4SC2y0lVs06Jbj1owCgYIKoZIzj0EAwMDaAAwZQIwQJrg
19 MSDaHAFKRIqMFmTf0veIy8z90bGR20jFa0+7rvLo70dJar/fvGwA4h++tShdAjEA
20 t8i+Q5B5XtUVZoFtEzKeJEBerp4TMW1sDHBYU1IFpkxg35oTNkL5uJe/PHe39NID
21 -----END CERTIFICATE-----
22
```

```
CryptoRSA6.c | Signature | Certificate1.pem | Certificate2.pem | Certificate3.pem | Certificate4.pem
1 -----BEGIN CERTIFICATE-----
2 MIICxjCCAk2gAwIBAgIRAL093/inhFu86Q0gQTWzSkUwCgYIKoZIzj0EAwMwTzEL
3 MAkGA1UEBhMCVVmxKTAnBgNVBAoTIEludGVybmV0IFNlY3VyaXR5IFJlc2VhcmNo
4 IEddyb3VwMRUwEwYDVQQDEwxJu1JHIFJvb3QgWDIwHhcNMjAwOTA0MDAwMDAwWhcN
5 MjUwOTE1MTYwMDAwjAyMQswCQYDVQQGEwJVUzEwMBQGA1UEChMNTGV0J3MgRW5j
6 cn1wdDELMAkGA1UEAxMCRTEdwjaQBgCqhkJOPQIBBgrUrgQQAIgNiAAQkXC2iKv0c
7 S6Zdl3MnMayyoGli72XopzDwxEuf/xwLcA/TmC9N/A8AmzfwdAVXMpcuBe8qQyWj
8 +240JxP2T35p0wKZUskR5LBJJvmsSGPwSSB/GjMH2m6WPUZIvd0xhajggEIMIIB
9 BDAOBgnVHQ8BaF8EBAMCAYwHQYDVR01BBYwFAYIKwYBBQUHawIGCCsGAQUFBwMB
10 MBIGA1UdEwEB/wQIMAYBAf8CAQAwHQYDVR0OBByEFFrz7Sv8NsI3eb1SM0pUb89V
11 yy6sMB8GA1UdIwQYMbaAFHxC1q7eS0g7+pL4nozPbYupcjeVMDIGCCsGAQUFBwEB
12 BCYwJDAlBggxBgEFBQcwAoYwaHR0cDovL3gyLmkubGVuY3Iub3JnLzAnBgNVHR8E
13 IDAeMBygGqAYhhZodHRw0i8veDIuYy5sZW5jc15vcmcvMCIGA1UdIAQbMBkwCAYG
14 Z4EMAQIBMA0GCysGAQQBgt8TAQEBAoGCCqGSM49BAMDA2cAMGQCMHt01VITjWH+
15 Dbo/AwCd89eYhNlXlr3pD5xcSAQh8suzYHK019YST8pE9kLJ03uGqQIwRgxt03q
16 YJkgsTgDyj2gJr jubilK9sZmHz0a25JK1fUpE8ZwYi16I4zPPS/Lgul/
17 -----END CERTIFICATE-----
18
```

```

CryptoRSA6.c | Signature | Certificate1.pem | Certificate2.pem | Certificate3.pem | Certificate4.pem |
1 -----BEGIN CERTIFICATE-----
2 MIIEYCCAkigAwIBAgIB55JKIY3b9QISMI/xjHkYzANBgkqhkiG9w0BAQsFADBP
3 MQswCQYDVQGEwJVUzEpMCCGA1UEChMgSW50ZXJuZXQgU2VjdXJpdHkgUmVzzWFy
4 Y2ggR3JvdXAxFATBqNVBAMTDE1TUkcgUm9vdCBYMTAeFw0yMDA5MDQwMDAwMDBa
5 Fw0yNTA5MTUxNjAwMDBaME8xCzAJBgNVBAYTA1VTMSkwJwYDVQQKEyBjbnR1cm5l
6 dCBTzWN1cm10eSBSZXN1YXJjaCBHcm91cDEVMBMGAA1UEAxMMSSVNSRyBsb290IFgy
7 MHwEAYHKoZIzj0CAQYFK4EEACIDYgAEzvVn4CDCuwJSvMWSj5cz3es3mcFDR0H
8 ttwWt1qLFNvicwDEuklwVEYm06gbf9yoWHKS5xcUy4APgHoIYOivXRdgKam7mAHF7
9 A1F9ItgKbpb9d/w+kHsOdx1ymgHDB/qo4H1MIHiMA4GA1UdDwEB/wQEawIBBjAP
10 BgNVHRMBAf8EBTADAQH/MB0GA1UdDgQWBBR8Qpau3ktIO/qS+J6Mz22LqXI3lTaf
11 BgNVHSMEGDAwBGR5tFnme7b15AFzgAiYBpY9umbbjAyBgrBgfEFBQcBAQQmMCQw
12 IgYIKwYBBQUHMAKGFmh0dHA6Ly94MS5pLmx1bmNyLm9yZy8wJwYDVR0fBCAwHjAc
13 oBqgGIYWh0rDovL3gxLmPubGVuY31ub3JnlzA1bgnVHSAEGzAZMAqGBmeBDAEC
14 ATANBgsrBgeEAYfFwEBATANBgkqhkiG9w0BAQsFAAOCAgEAG381K5B6CHAdxjh
15 wy6KNkxBfr8XS+Mw1sMfpwWmG97sGjAJETM4vL80eirb0p8B+RdNDJ1V/aWtbdIV
16 P0tywC6uc8c1f1fCPht4DHRCosEBGJ4QjEiRhrtekC/lxaBRHfKbHtdIVwH8hGR
17 Ib/hL8Lvbv0FIOS93nZbs3KvDgsaysUfuFs1oeZs5YBxg4f3GpPI0617yCnpp2
18 D56wKf3L84kHSBv+q5MuFCENX6+0t1SrXQ7UW0xxJLqPam2m3wf4DtVudhTU8yD
19 ZrtK3IEGABiL9LPXSLTQbnEtP7PLHeOQiALgH6fxatI27xvBI1sRikCDXCKhfES
20 c7ZGJEKeKhcY46zMjyZG0tdm3dLCsmllqXPIQgb5dovy++fc50u+DzFr4+XKM6r
21 4pgmmIV97igyiintTJUJxCD6B+GGLET2gUfA5GiY7R3YPEiIlslNekbave1mk7uOG
22 nMeIMookmZVm4WAUR3YQcvJHM8qvemciWQPb1pk4qJwxsuseETLQyu/w4XKAM
23 YxtX7hd0UM+v8qIPN4zhDtTLxq9nHE+z0H40aijvQT2GcD5hq/1DhqqlWvvykdx
24 S2McTZbbVSMKrn+Q-BdaDmQPVkRgNuzvpqfqBspDQGdNpT2Lm4xiN9qfgqLaScpi4t
25 EcrmzTFYeYXmchynn9NM0GbQp7s=
26 -----END CERTIFICATE-----
27

```

```

CryptoRSA6.c | Signature | Certificate1.pem | Certificate2.pem | Certificate3.pem | Certificate4.pem |
1 -----BEGIN CERTIFICATE-----
2 MIIFYCCBeigAwIBAgIQQAF3ITfu6UK47naqPGKtzANBgkqhkiG9w0BAQsFADA/
3 MSQwIgYDVQKExTeaWdpdGfsIFNpz25hdhvYZSBUcnVzdCBuby4FxZAVBqNVBAMT
4 DKRTVCB290IEENBFgzMB4XDTIxMDEyMDE5MTQwM1oXTDI0MDkzMDE4MTQwM1ow
5 TzELMAKGa1UEBhMCVVMxKTAnBgNVBAoTIElduGVybV0IFNlY3VyaXR5IFJlc2Vh
6 cmNoIEdyb3VmRUJwEwYDVQDewxJU1JHTFJvb3QgWDewggIIma0GCSgS1b3DQEB
7 AQUAA4ICDwAwggIKAoICAQCT6CrzBQ385ueK1coHie+3Lff0JCMbjzmVB493XC
8 ov71am72AE8o295ohmxEk7axY/0UEmu/H9LqmZshftEzPLpI9d153704/xLxIZpL
9 wYqGcwlKZmZsj348cL+tKSiG8+TA5oCu4kuPt51+1Ao0f00exFJlII1Po0K5PCm+D
10 LtFJv4yAdlba94AjxsDcCEbdFtwpPqPrta3Y6vrFk/CjhFlfs8L6P+1dy70sntK
11 4EwSJQxwjQmpo0FTJ0wT2e4VvxCzSow/iaNhUd6shweU9GNx7C7ib1uYgeGJXDR5
12 bHbv05BieebbpJovJsXQE0E03tkQjhB7t/eo98flAgeYjzYIlefIn5YNNnWe+w5y
13 sR2bvAP5SQYXg0FtCwQemsAxAvCg/Y39W9Eh81LygXbnKYwagJZhduRze6zqxZ
14 Xnidf3LwiCUGQSk+WT7dJvUkyRGrwqNMQB9GogZm1pzbRboY7nn1px1fEfntP1F4
15 FQsDj43QLwWpntKHeTzBRl8xurqUBN8Q5N0s8p0544fAQjQMNRbcTa0B7rBMDbc
16 SLeC05imfwCKoqMptsy6vYMEG6KDA0Gh1gXg8K28Kh8hjtGqEgqiNx2mna/H2q1
17 PRmP6zjzZN7IKw0KKP/32+IVQtQi0Cdd4Xn+G0dwik105tmLsbdJ1fu/7xk9TND
18 TwIDAQAB4IBRjCCAUtwDwYDVR0TAQH/BAUwAEB/ZA0BgNVHQ8Af8EBAMCAQYw
19 SwYIKwYBBQUHAAQEPzA9MDsGCCsGAQUFBzAchi9odHrw0i8VYXbwcy5pZGVudHJ1
20 c3QyZ9tl3Jvb3RzL2RzdJvB3RjYXgZlnA3YzafBgnVHSMEDAwgBTep7Gkeyxx
21 +tvh5SB1/8QVYIWJEDBuBgNVHSAETTBLMAGBmeBDAECATA/BgsrBgeEAYLfEwEB
22 ATAwMC4GCCsGAQUFBwIBF1JodHrw0i8VY3bzLnJvb3QteDeubGV0c2VuY3J5ChQu
23 b3JnMDwGA1UdHwQ1MDMwMaAvoc2GK2h0dHA6Ly9jcmwuaWR1bnRydXN0LmNvbS9E
24 U1RST09UQ0FYM0NSTC5jcmwwHqYDVR0BByEFeM0WeZ7tuXkAX0ACIjIG1j26Ztu
25 MA0GCSqGSIb3DQEBCwUA4IBAQAKcwBs1m7/DlQrt2M51oGrS+o44+/yQoDFVDC
26 5WxCu2+b9LRPwkSICHXM6webFGJueN7sJ7o5XPWiow5WlHAQU7G75K/QosMrAdSw
27 9MUgNTP52GE24HGNTl1qoJF1cDyqSMo59ahy2cI2qBDLkobkx/J3vWraV0T9VuG
28 WCLKTVXkcGdtw1fFRj1Bz4pYg1htmf5X6DY08A4jqv2I19DjXA6Usbw1FzXSlr90
29 he8Y4IwS6wY7bCkjCWDcRQJMEhg76fs03txE+F1Yruq9RUWhiF1myv4Q6w+CyBFC
30 Dfvp700GAN6dE0M4+qR9sdjoSYKEBpsr6GtPAQw4dy753ec5
31 -----END CERTIFICATE-----

```

n value to be copied in CryptoRSA6.c

```
(UWLGrouP2@kali)-[~/Desktop/CryptorSA Lab UWL]
$ openssl x509 -in Certificate1.pem -noout -modulus
Modulus=No modulus for this public key type

(UWLGrouP2@kali)-[~/Desktop/CryptorSA Lab UWL]
$ openssl x509 -in Certificate2.pem -noout -modulus
Modulus=No modulus for this public key type

(UWLGrouP2@kali)-[~/Desktop/CryptorSA Lab UWL]
$ openssl x509 -in Certificate3.pem -noout -modulus
Modulus=No modulus for this public key type

(UWLGrouP2@kali)-[~/Desktop/CryptorSA Lab UWL]
$ openssl x509 -in Certificate4.pem -noout -modulus
Modulus=ADE82473F41437F39B9E2B57C87EBD7F38908C63CE657A078F775C2A2FEF56A6F6004F28DBDE68866C4493B6B163FD14126BBF1FD2EA319B217ED1333CBA4F5D
79DFB38BF1219A4ABC18A8671694A66666C8F7E3C70BFAD292206F3E4C0680AAE248BF87997E94039FD347977C99482353E838AE4F0A6F832ED149578C8074B6DA2FD0388D7B
03701121B7F2303CAFA8F4EADD6A3ABE164FC28E114B7E653E8F8FB5772F4B27B4AE04C12250C708D0329A0E15324E13D9E19B8F10B3A48C3F89A36151DEAC870794F46371CE2E
265F5B9881E1895C34796C76E3F8906279E6DBA9492F26CS0D10E0FED9108E16FB7F7A8F7C7E50207988F360895E7E237960D36759EF80E72811D9BBC03F94905D881D05B42AD6
41E9AC0176950A0FD8FD5B121F352F28176CD298C1A80964776E4737BACECA595E689D7F72D689C50641293E593EDD26F524C911A75AA34C401F46A199B5A73A516E863B9E7D72A
712057859D1517815B0838F8BD05F5B23E78A1C4B730512FC6EA050137C49374B3CA74E78E1F0108D03045B7136B407BAC130305C4887823898A67D608AA232982CCBABB
83041BA2830341A1D605F11BC2B6FOA87C863B48A482A88D0C79A76BF1F6AA53D198FEB38F364DEC82B0D0A28FF7D8E2154D242D0275DE179F1E877088AD4EE6G98B3AC6DD275
16EFFCB64F533434F
```

```
(UWLGroup2@kali)-[~/Desktop/CryptoRSA Lab UWL]
$ openssl x509 -in Certificate4.pem -noout -text
Certificate:
Data:
Version: 3 (0x2)
Serial Number:
        40:01:77:21:37:d4:e9:42:b8:ee:76:aa:3c:64:0a:b7
Signature Algorithm: sha256WithRSAEncryption
Issuer: O = Digital Signature Trust Co., CN = DST Root CA X3
Validity
    Not Before: Jan 20 19:14:03 2021 GMT
    Not After : Sep 30 18:14:03 2024 GMT
Subject: C = US, O = Internet Security Research Group, CN = ISRG Root X1
Subject Public Key Info:
Public Key Algorithm: rsaEncryption
Public-Key: (4096 bit)
Modulus:
        00:ad:e8:24:73:f4:14:37:f3:9b:9e:2b:57:28:1c:
        87:be:dc:b7:df:38:90:8c:6e:3c:e6:57:a0:78:f7:
        75:c2:a2:fe:f5:6a:6e:f6:00:4f:28:db:de:68:86:
        6c:44:93:b6:b1:63:fd:14:12:6b:bf:1f:d2:ea:31:
        b3:jn:MDwGA1uDHQ1MD:
        9b:21:7e:d1:33:3c:ba:48:f5:dd:79:df:b3:b8:ff:
        12:f1:21:9a:4b:c1:8a:86:71:69:4a:66:66:6c:8f:
        7e:3c:70:bf:ad:29:22:06:f3:e4:c0:e6:80:ae:e2:
        4b:8f:b7:99:7e:94:03:9f:d3:47:97:7c:99:48:23:
        53:e8:38:ae:4f:0a:6f:83:2e:d1:49:57:8c:80:74:
        b6:da:2f:d0:38:8d:7b:03:70:21:1b:75:f2:30:3c:
        fa:8f:ae:dd:da:63:ab:eb:16:4f:c2:8e:11:4b:7e:
        cf:0b:e8:ff:b5:77:2e:f4:b2:7b:4a:e0:4c:12:25:
        0c:70:8d:03:29:a0:e1:53:24:ec:13:d9:ee:19:bf:
        10:b3:4a:8c:3f:89:a3:61:51:de:ac:87:07:94:f4:
        63:71:ec:2e:e2:6f:5b:98:81:e1:89:5c:34:79:6c:
-----END CERTIFICATE-----
```

e value to be copied in CryptoRSA6.c

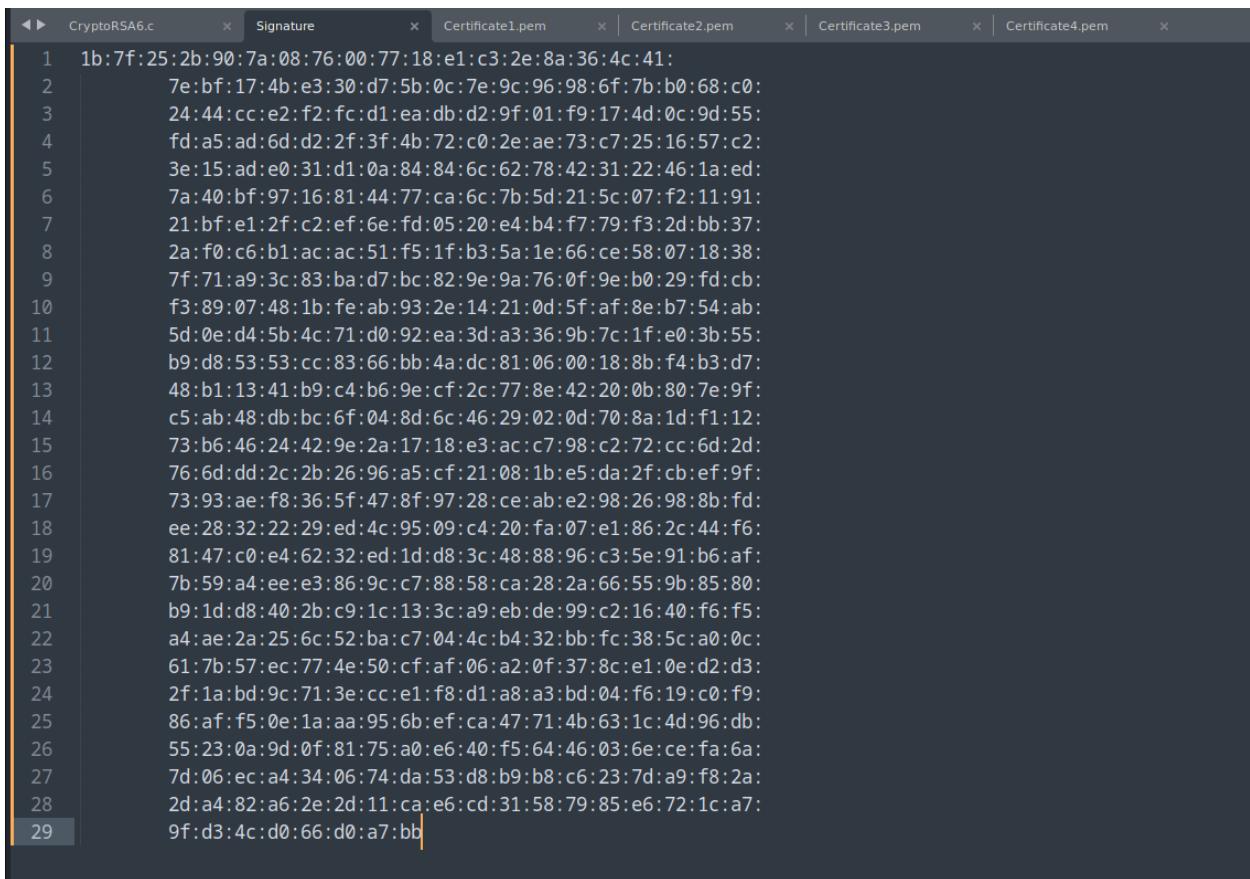
```
37:ba:ce:ac:59:5e:68:9d:7f:72:d6:89:c5:06:41: certificated.pem
1: ---- BEGIN CERTIFICATE ----
2: MIIFYDCBEBigAwIBAg...
3: MSQwIgYDVQQKExtEaW...
4: DKRTVCBsb290IENBIF...
5: TzELMAkGATUEBhMCVv...
6: cmNoIEdyb3VwMRUwEw...
7: AQUAA4ICDwAwggIKAd...
8: ov71am72AE8o295ohmu...
9: wYqGcWIKzmZsj348CL...
10: LtFJV4yAdLbaL9A4jX...
11: 4Ew5JQxwjOMpdOFTJov...
12: bHbv05B1eebbpJovJ...
13: sR2bvAP5SQXYgd0FtC...
14: Xm1df3LWicUG0SK4wT...
15: FQsdT430LwvPn...
16: SLecD X509v3 extensions:
17: PRmPg2] Z X509v3 Basic Constraints: critical
18: TwIDAQABg4IBR] CA:TRUE
19: SWYIKwYF X509v3 Key Usage: critical
20: c3QuY29tL3VJ... Certificate Sign, CRL Sign
21: +tvh5SB1/ Authority Information Access:
22: ATAwMC4GCCSAQ...
23: b3JnMDwGA... CA Issuers - URI:http://apps.identrust.com/roots/dstroottax3.p7c
24: U1RST09UQ... X509v3 Authority Key Identifier:
25: MA0GCSqGSIb3DQE... C4:A7:B1:A4:7B:2C:71:FA:DB:E1:4B:90:75:FF:C4:15:60:85:89:10
26: SWxCu2+b9... X509v3 Certificate Policies:
27: 9MuqNTP52GE24H... Policy: 2.23.140.1.2.1
28: WCLKTVXkcgdtw... Policy: 1.3.6.1.4.1.44947.1.1.1
29: he8Y4IW56w7bCKjC... CPS: http://cps.root-x1.letsencrypt.org
30: Dfvp700GA X509v3 CRL Distribution Points:
31: ----END CERTIFICATE---- Full Name:
32: URI:http://crl.identrust.com/DSTROOTCAX3CRL.crl
33: X509v3 Subject Key Identifier:
34: 79:B4:59:E6:7B:B6:E5:E4:01:73:80:08:88:C8:1A:58:F6:E9:9B:6E
```

```
(UWLGroup2㉿kali)-[~/Desktop/CryptoRSA Lab UWL]
$ openssl x509 -in Certificate3.pem -text -noout
Certificate: RSA 256
Data:
Version: 3 (0x2)
Serial Number:
    07:9e:49:28:86:37:f4:08:48:c2:3f:c6:31:e4:63
Signature Algorithm: sha256WithRSAEncryption
Issuer: C = US, O = Internet Security Research Group, CN = ISRG Root X1
Validity
    Not Before: Sep 4 00:00:00 2020 GMT
    Not After : Sep 15 16:00:00 2025 GMT
Subject: C = US, O = Internet Security Research Group, CN = ISRG Root X2
Subject Public Key Info:
    Public Key Algorithm: id-ecPublicKey
    Public-Key: (384 bit)
        pub:
        // Certificate4: 04:cd:9b:d5:9f:80:83:0a:ec:09:4a:f3:16:4a:3e:
        BN_hex2bn(&n: 5c:cf:77:ac:de:67:05:0d:1d:07:b6:dc:16:fb:5a:0C2A2FEEF56A6EF6004F28DBD568
        // Certificate4: 8b:14:db:e2:71:60:c4:ba:45:95:11:89:8e:ea:06:
        BN_hex2bn(&e: df:f7:2a:16:1c:a4:b9:c5:c5:32:e0:03:e0:1e:82:
        BN_hex2bn(&s: 18:38:8b:d7:45:d8:0a:6a:6e:e6:00:77:fb:02:51:
        BN_mod_exp(message: 7d:22:d8:0a:6e:9a:5b:77:df:f0:fa:41:ec:39:dc:
        75:ca:68:07:0c:1f:ea
        printBN(Certificate4: ASN1 OID: secp384r1
        return 0
        NIST CURVE: P-384
X509v3 extensions:
    X509v3 Key Usage: critical
        Certificate Sign, CRL Sign
    X509v3 Basic Constraints: critical
        CA:TRUE
    X509v3 Subject Key Identifier:
```

Signature value taken from Certificate3.pem

```
Policy: 2.23.140.1.2.1
Policy: 1.3.6.1.4.1.44947.1.1.1
Signature Algorithm: sha256WithRSAEncryption
Signature Value:
1b:7f:25:2b:90:7a:08:76:00:77:18:e1:c3:2e:8a:36:4c:41:
7e:bf:17:4b:e3:30:d7:5b:0c:7e:9c:96:98:6f:7b:b0:68:c0:
24:44:cc:e2:f2:fc:d1:ea:db:d2:9f:01:f9:17:4d:0c:9d:55:
fd:a5:ad:6d:d2:2f:3f:4b:72:c0:2e:ae:73:c7:25:16:57:c2:
3e:15:ad:e0:31:d1:0a:84:84:6c:62:78:42:31:22:46:1a:ed:
7a:40:bf:97:16:81:44:77:ca:6c:7b:5d:21:5c:07:f2:11:91:
21:bf:e1:2f:c2:ef:6e:fd:05:20:e4:b4:f7:79:f3:2d:bb:37:
2a:f0:c6:b1:ac:ac:51:f5:1f:b3:5a:1e:66:ce:58:07:18:38:
7f:71:a9:3c:83:ba:d7:bc:82:9e:9a:76:0f:9e:b0:29:fd:cb:
f3:89:07:48:1b:fe:ab:93:2e:14:21:0d:5f:af:8e:b7:54:ab:
5d:0e:d4:5b:4c:71:d0:92:ea:3d:a3:36:9b:7c:1f:e0:3b:55:
b9:d8:53:53:cc:83:66:bb:4a:dc:81:06:00:18:8b:f4:b3:d7:
48:b1:13:41:b9:c4:b6:9e:cf:2c:77:8e:42:20:0b:80:7e:9f:
c5:ab:48:db:bc:6f:04:8d:6c:46:29:02:0d:70:8a:1d:f1:12:
73:b6:46:24:42:9e:2a:17:18:e3:ac:c7:98:c2:72:cc:6d:2d:
76:6d:dd:2c:2b:26:96:a5:cf:21:08:1b:e5:da:2f:cb:ef:9f:
73:93:ae:f8:36:5f:47:8f:97:28:ce:ab:e2:98:26:98:8b:fd:
ee:28:32:22:29:ed:4c:95:09:c4:20:fa:07:e1:86:2c:44:f6:
81:47:c0:e4:62:32:ed:1d:d8:3c:48:88:96:c3:5e:91:b6:af:
7b:59:a4:ee:e3:86:9c:c7:88:58:ca:28:2a:66:55:9b:85:80:
b9:1d:d8:40:2b:c9:1c:13:3c:a9:eb:de:99:c2:16:40:f6:f5:
a4:ae:2a:25:6c:52:ba:c7:04:4c:b4:32:bb:fc:38:5c:a0:0c:
61:7b:57:ec:77:4e:50:cf:af:06:a2:0f:37:8c:e1:0e:d2:d3:
2f:1a:bd:9c:71:3e:cc:e1:f8:d1:a8:a3:bd:04:f6:19:c0:f9:
86:af:f5:0e:1a:aa:95:6b:ef:ca:47:71:4b:63:1c:4d:96:db:
55:23:0a:9d:0f:81:75:a0:e6:40:f5:64:46:03:6e:ce:fa:6a:
7d:06:ec:a4:34:06:74:da:53:d8:b9:b8:c6:23:7d:a9:f8:2a:
2d:a4:82:a6:2e:2d:11:ca:e6:cd:31:58:79:85:e6:72:1c:a7:
9f:d3:4c:d0:66:d0:a7:bb
```

Copy in Sublime text and save the file as Signature



The screenshot shows a Sublime Text interface with several tabs open. The active tab is titled "Signature" and contains a large amount of hex data. The data is a continuous sequence of 16-digit hex values, starting with 1b:7f:25:2b:90:7a:08:76:00:77:18:e1:c3:2e:8a:36:4c:41: and ending with 9f:d3:4c:d0:66:d0:a7:bb. The code editor has a dark theme with orange highlights for the line numbers (1 through 29) and the current line. Other tabs visible include "CryptoRSA6.c", "Certificate1.pem", "Certificate2.pem", "Certificate3.pem", and "Certificate4.pem".

```
1 1b:7f:25:2b:90:7a:08:76:00:77:18:e1:c3:2e:8a:36:4c:41:  
2 7e:bf:17:4b:e3:30:d7:5b:0c:7e:9c:96:98:6f:7b:b0:68:c0:  
3 24:44:cc:e2:f2:fc:d1:ea:db:d2:9f:01:f9:17:4d:0c:9d:55:  
4 fd:a5:ad:6d:d2:2f:3f:4b:72:c0:2e:ae:73:c7:25:16:57:c2:  
5 3e:15:ad:e0:31:d1:0a:84:84:6c:62:78:42:31:22:46:1a:ed:  
6 7a:40:bf:97:16:81:44:77:ca:6c:7b:5d:21:5c:07:f2:11:91:  
7 21:bf:e1:2f:c2:ef:6e:fd:05:20:e4:b4:f7:79:f3:2d:bb:37:  
8 2a:f0:c6:b1:ac:ac:51:f5:1f:b3:5a:1e:66:ce:58:07:18:38:  
9 7f:71:a9:3c:83:ba:d7:bc:82:9e:9a:76:0f:9e:b0:29:fd:cb:  
10 f3:89:07:48:1b:fe:ab:93:2e:14:21:0d:5f:af:8e:b7:54:ab:  
11 5d:0e:d4:5b:4c:71:d0:92:ea:3d:a3:36:9b:7c:1f:e0:3b:55:  
12 b9:d8:53:53:cc:83:66:bb:4a:dc:81:06:00:18:8b:f4:b3:d7:  
13 48:b1:13:41:b9:c4:b6:9e:cf:2c:77:8e:42:20:0b:80:7e:9f:  
14 c5:ab:48:db:bc:6f:04:8d:6c:46:29:02:0d:70:8a:1d:f1:12:  
15 73:b6:46:24:42:9e:2a:17:18:e3:ac:c7:98:c2:72:cc:6d:2d:  
16 76:6d:dd:2c:2b:26:96:a5:cf:21:08:1b:e5:da:2f:cb:ef:9f:  
17 73:93:ae:f8:36:5f:47:8f:97:28:ce:ab:e2:98:26:98:8b:fd:  
18 ee:28:32:22:29:ed:4c:95:09:c4:20:fa:07:e1:86:2c:44:f6:  
19 81:47:c0:e4:62:32:ed:1d:d8:3c:48:88:96:c3:5e:91:b6:af:  
20 7b:59:a4:ee:e3:86:9c:c7:88:58:ca:28:2a:66:55:9b:85:80:  
21 b9:1d:d8:40:2b:c9:1c:13:3c:a9:eb:de:99:c2:16:40:f6:f5:  
22 a4:ae:2a:25:6c:52:ba:c7:04:4c:b4:32:bb:fc:38:5c:a0:0c:  
23 61:7b:57:ec:77:4e:50:cf:af:06:a2:0f:37:8c:e1:0e:d2:d3:  
24 2f:1a:bd:9c:71:3e:cc:e1:f8:d1:a8:a3:bd:04:f6:19:c0:f9:  
25 86:af:f5:0e:1a:aa:95:6b:ef:ca:47:71:4b:63:1c:4d:96:db:  
26 55:23:0a:9d:0f:81:75:a0:e6:40:f5:64:46:03:6e:ce:fa:6a:  
27 7d:06:ec:a4:34:06:74:da:53:d8:b9:b8:c6:23:7d:a9:f8:2a:  
28 2d:a4:82:a6:2e:2d:11:ca:e6:cd:31:58:79:85:e6:72:1c:a7:  
29 9f:d3:4c:d0:66:d0:a7:bb
```

Sha256Sum to Certificate3_body.bin, found that is same as compiling Task6

n copied from openssl x509 -in Certificate4.pem -noout -modulus

e copied from Exponent of openssl x509 -in Certificate4.pem -noout -text

s copied from cat Signature | tr -d '[:space:]' of Certificate3.pem

```
 1 #include <stdio.h>
 2 #include <openssl/bn.h>
 3 #define BNITS 256
 4 void printBN(char *msg, BIGNUM *a)
 5 {
 6     char *number_str = BN_bn2hex(a);
 7     printf("%s %s\n", msg, number_str);
 8     OPENSSL_free(number_str);
 9 }
10 int main()
11 {
12     BN_CTX *ctx = BN_CTX_new();
13     BIGNUM *s = BN_new();
14     BIGNUM *n = BN_new();
15     BIGNUM *e = BN_new();
16     BIGNUM *message = BN_new();
17
18     // Certificate4.pem Modulus
19     BN_hex2bn(&n, "0E52475FA14373989E2857281C878EF0C87DF38980C63CE657A078F775C2A2EF56A6FF6004F2808D68866C449386B163FD141268BF1F02EA3198217ED133CBA48F5D0790FB3886F12F1219A48C18A867169
20     4A66666C8F7E73C708FD392206F3EAC0E680AE2488FB7997E594039F0347977C9482353E838AE4F0A6F832E0149578C8074860A2FD03880780370211875F2303CF48F8E0D6A63A8E8164FC28E11487C9F88E8F85772F4A82784AE
21     04C1259E708D329A0E15324EC13D9E198F1083448C3F89A361510EAC870794F46371EC2EE26F5B9881E1895C34796C76EF3B99062796E08A49A2F26C5D010E18EDED9108E16FB87F7A8F7C7E50207988F3600895E7E23796003675
22     9EF80E72B11D98BC03F94905D8810D05842AD641E9AC0176930A0F8DFD05B0121F352F28176CD298C1AB0964776E4737BACEAC595E68907720689C50641293E593E0D26F524C911A75AA34C401F46A19985A73A5168E863B9E7072A
23     712057839ED3E517815008038F80D002F59823F784A1C467308512FC66EAE059137C43937483CA74E78E1F01880030045871369407BA1C130305C4887823896A70D608AA2A32982CCBADB03041BA2830341A1D605F11BC286F0A87C863B
24     46A8482A880DC769A768F1F6AA53D198FEB30F3640EC82800A28FFF708E215420422D02750E179FE18E77888AD4E609883AC60027516EFFBC64F533434F";
25     // Certificate4.pem Exponent
26     BN_hex2bn(&e, "10001");
27     // Signature from certificate3.pem
28     BN_hex2bn(&s, "1b7f252b907a0876007718e1c32e8a364c417ebf174be338d75b0c7e9c96986f7bb068c02444cce2f2fc1eadbd29f01f9174d0c9d55fda5ad6dd22f3f4b72c02eae73c7251657c23e15ade031d10a84846c6278
29     423122401aed7a40ff716814477cac67b5d215c07f2119121bf1e12f2cefe0fd0520e4ab4f79f32dbb32fa0cc0b1acac51f51fb35a1e60ce580718387f71a93c83bad7bc829ea760f9eb029fcbf38907481bf0a932e14210d5fa
30     f8eb754ab5d0ed4504c71d092ea3d3a33697c1fe03b55b9d85353c83660ba4adc180600188fb45d3748b11349c4b69ecf2778e42200b807e9fc5ab48dbbcf0484dc4629020708a1df11273b64624429e2a1718e3acc798c272
31     c662d2766dd2c2b2696a5cf21081be5da2fcbe9f7393aef365f478f9728ceabe29826988bfde2e28322229ed4c9509420f7a0e71862c44f68147c0e46232ed1dd83c488896c35e91b6a77b5944eee3869cc78858ca282a66559b8
32     5800b91d8402bc91c133c9ebde99c21640f0fs4aae2a250c52bac7044c432bbfc385ca00c617057ec774e50cfaf06a20f378ce10ed2d32f1abd9c713ecc1f8d18a3b0d4f619c0f86aff50e1aaa956befca47714b631c4d96db
33     55230a9d0ff8175a0e640f56446036ceccfa6a7d06eca4340674da53d8b98c6237da0ff2a2d4a62a6ze2d1cae6c0315879856e6721ca79fd34cd066d0a7b8");
34     BN_mod_exp(message,s,e,n,ctx);
35
36     printBN("Message = ", message);
37
38     return 0;
39 }
40 }
```