# Foundations Network Security

# RSA LAB

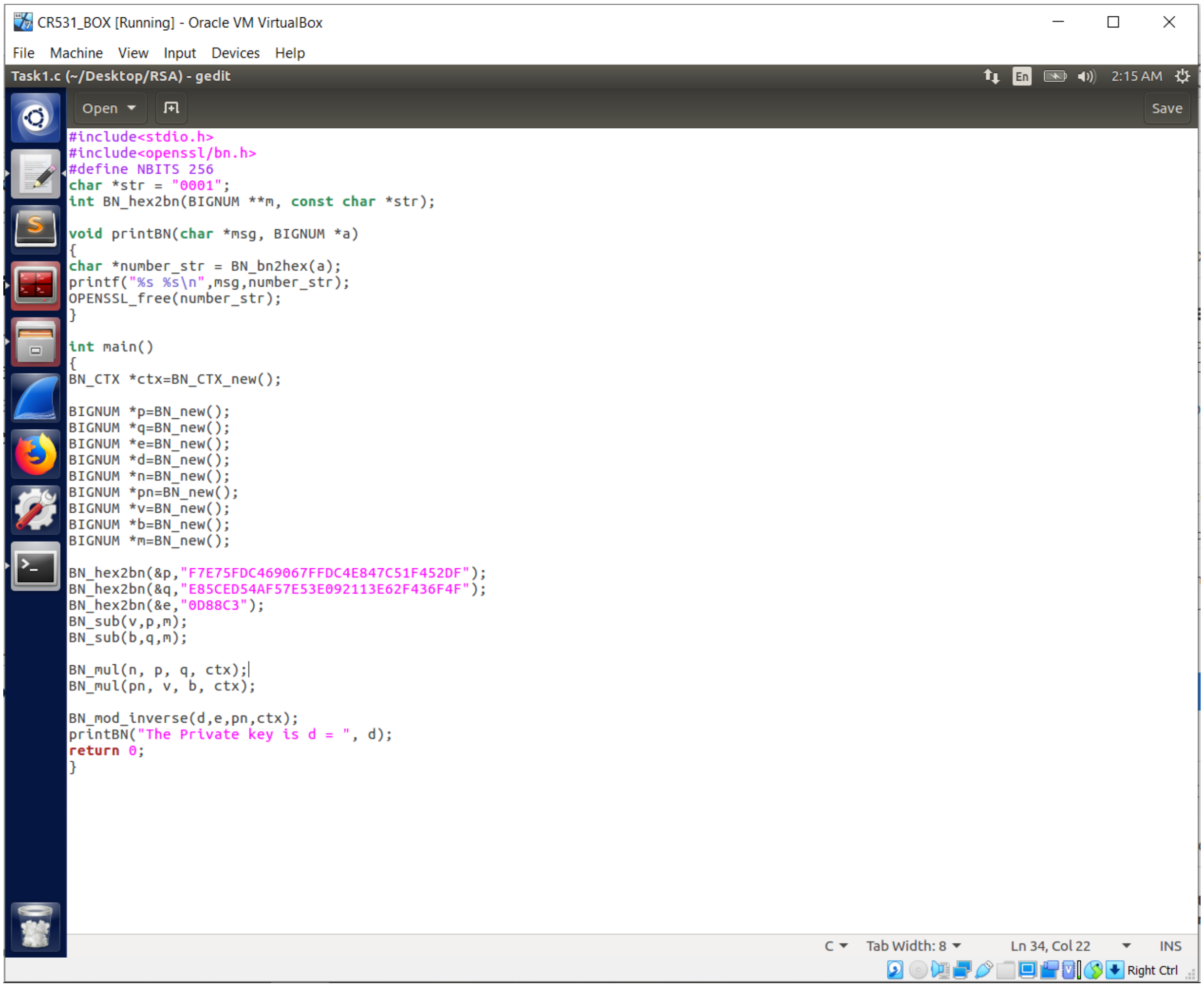
*Chakradhar Reddy Donuri*

*E949F496*

### Task 1: Deriving the Private Key

**Deliverable**. Your code should print out “The private key is d=...”, where the dots should be replaced with the actual value of d that you calculated

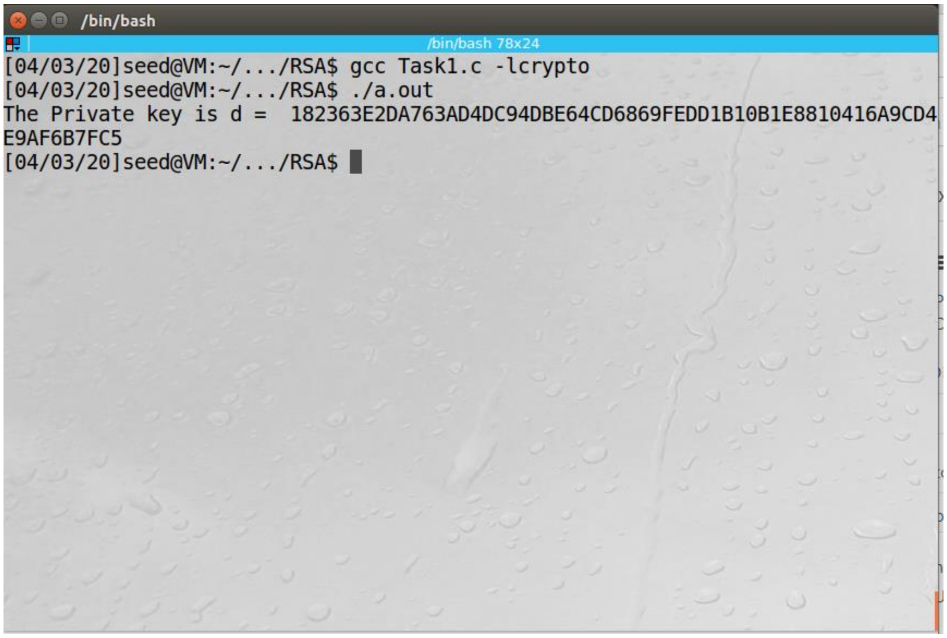
**Code**:



CR531_BOX [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Task1.c (~/Desktop/RSA) - gedit
2:15 AM
Open
Save
C
Tab Width: 8
Ln 34, Col 22
INS
Right Ctrl

```
#include<stdio.h>
#include<openssl/bn.h>
#define NBITS 256
char *str = "0001";
int BN_hex2bn(BIGNUM **m, const char *str);

void printBN(char *msg, BIGNUM *a)
{
char *number_str = BN_bn2hex(a);
printf("%s %s\n",msg,number_str);
OPENSSL_free(number_str);
}

int main()
{
BN_CTX *ctx=BN_CTX_new();

BIGNUM *p=BN_new();
BIGNUM *q=BN_new();
BIGNUM *e=BN_new();
BIGNUM *d=BN_new();
BIGNUM *n=BN_new();
BIGNUM *pn=BN_new();
BIGNUM *v=BN_new();
BIGNUM *b=BN_new();
BIGNUM *m=BN_new();

BN_hex2bn(&p,"F7E75FDC469067FFDC4E847C51F452DF");
BN_hex2bn(&q,"E85CED54AF57E53E092113E62F436F4F");
BN_hex2bn(&e,"0D88C3");
BN_sub(v,p,m);
BN_sub(b,q,m);

BN_mul(n, p, q, ctx);
BN_mul(pn, v, b, ctx);

BN_mod_inverse(d,e,pn,ctx);
printBN("The Private key is d = ", d);
return 0;
}
```

**Output:**

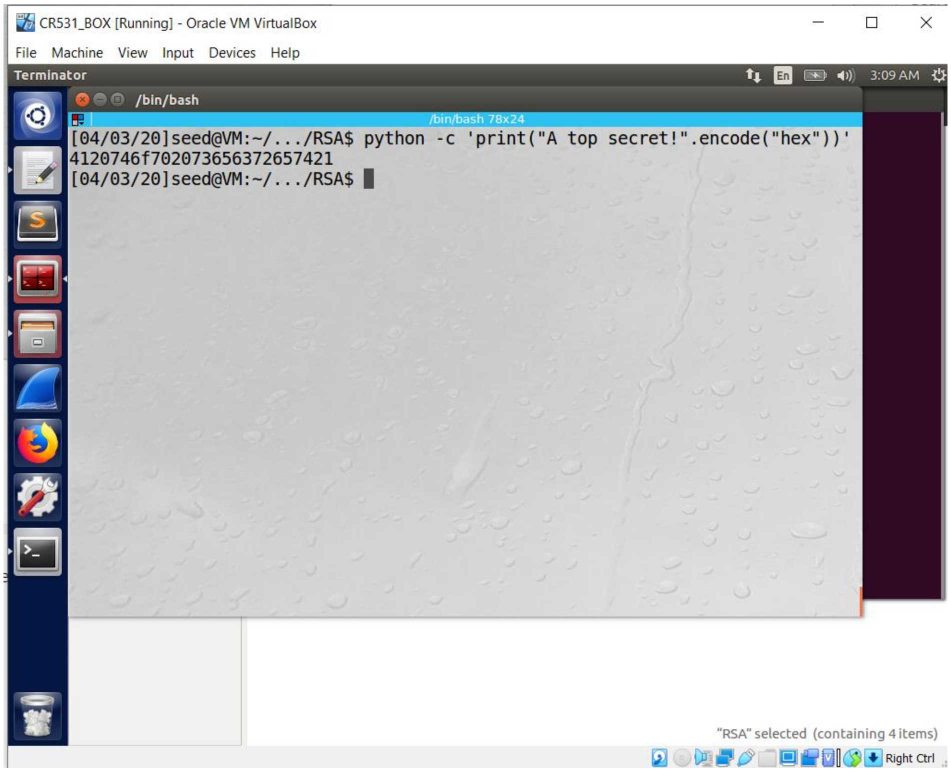


```
[04/03/20]seed@VM:~/.../RSA$ gcc Task1.c -lcrypto
[04/03/20]seed@VM:~/.../RSA$ ./a.out
The Private key is d =  182363E2DA763AD4DC94DBE64CD6869FEDD1B10B1E8810416A9CD4
E9AF6B7FC5
[04/03/20]seed@VM:~/.../RSA$
```

## Task 2: Encrypting a Message

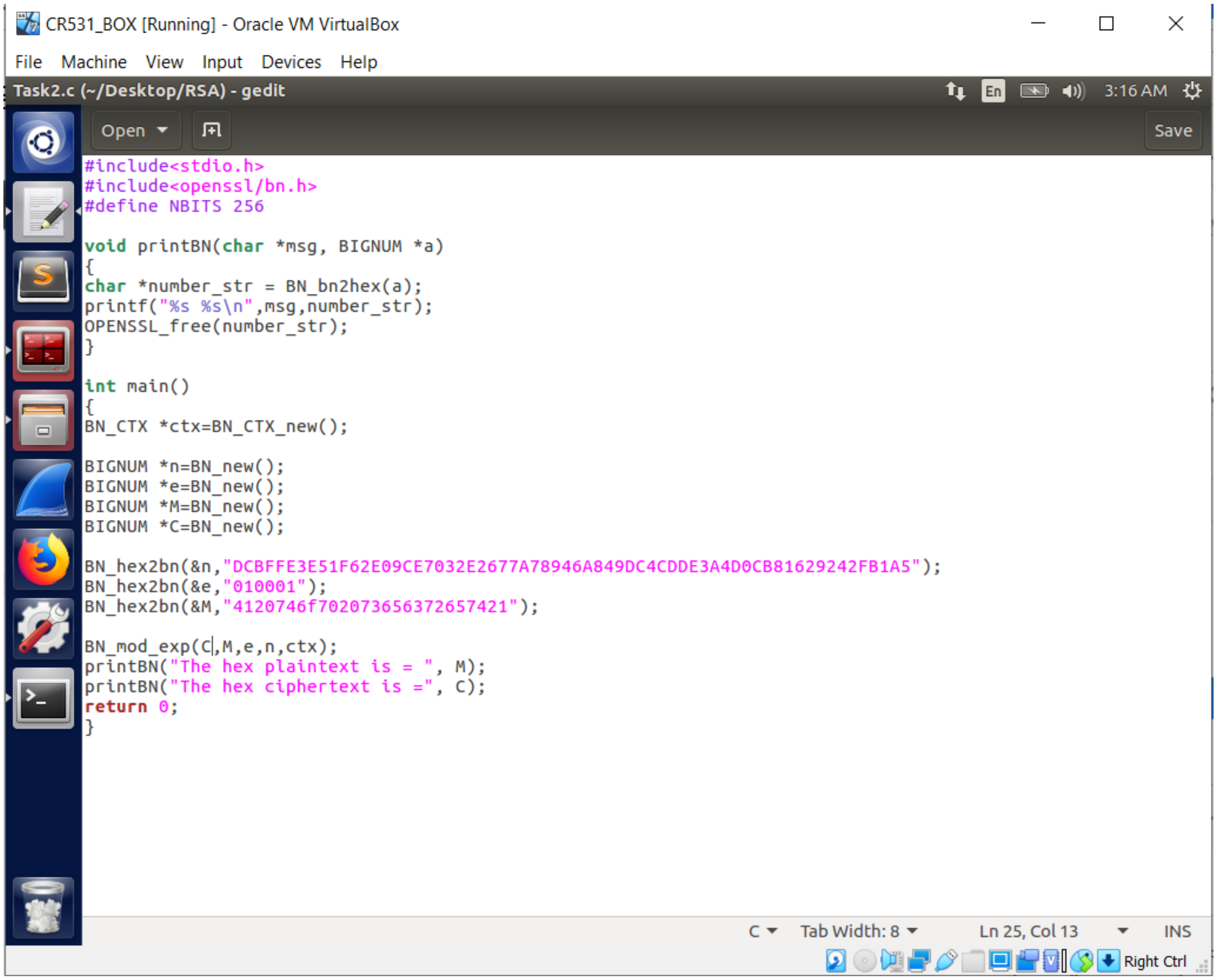
**Deliverable**. Your code should print out ”The hex plaintext is ... and the hex ciphertext is ...”, where the dots should be replaced with the actual hex values.

$ python -c ‘print("A top secret!".encode("hex"))’



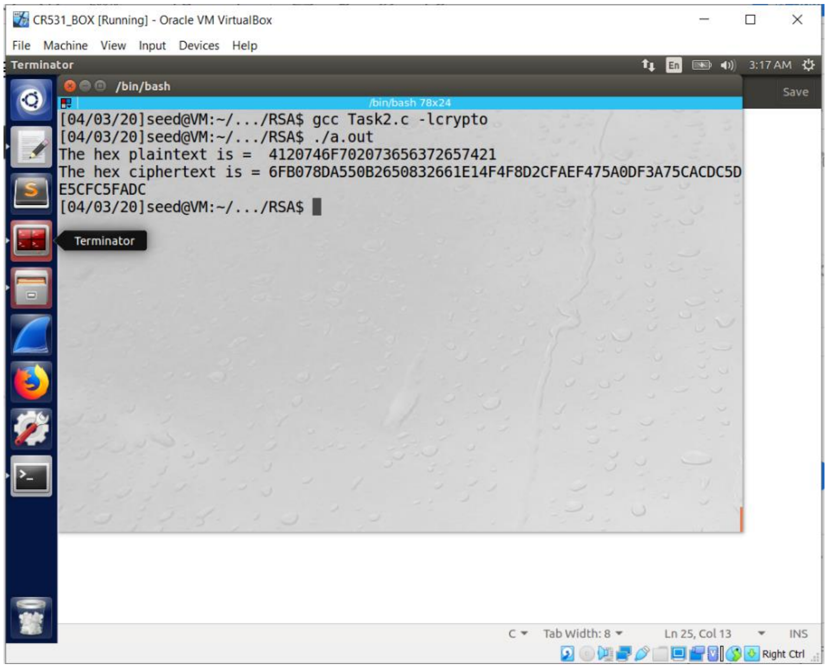
```
[04/03/20]seed@VM:~/.../RSA$ python -c 'print("A top secret!".encode("hex"))'
4120746f702073656372657421
[04/03/20]seed@VM:~/.../RSA$
```

**Code**:


CR531_BOX [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Task2.c (~/Desktop/RSA) - gedit
En
3:16 AM
Open
Save
C
Tab Width: 8
Ln 25, Col 13
INS
Right Ctrl

```c
#include<stdio.h>
#include<openssl/bn.h>
#define NBITS 256

void printBN(char *msg, BIGNUM *a)
{
char *number_str = BN_bn2hex(a);
printf("%s %s\n",msg,number_str);
OPENSSL_free(number_str);
}

int main()
{
BN_CTX *ctx=BN_CTX_new();

BIGNUM *n=BN_new();
BIGNUM *e=BN_new();
BIGNUM *M=BN_new();
BIGNUM *C=BN_new();

BN_hex2bn(&n,"DCBFFE3E51F62E09CE7032E2677A78946A849DC4CDDE3A4D0CB81629242FB1A5");
BN_hex2bn(&e,"010001");
BN_hex2bn(&M,"4120746f702073656372657421");

BN_mod_exp(C,M,e,n,ctx);
printBN("The hex plaintext is = ", M);
printBN("The hex ciphertext is =", C);
return 0;
}
```

**Output**:

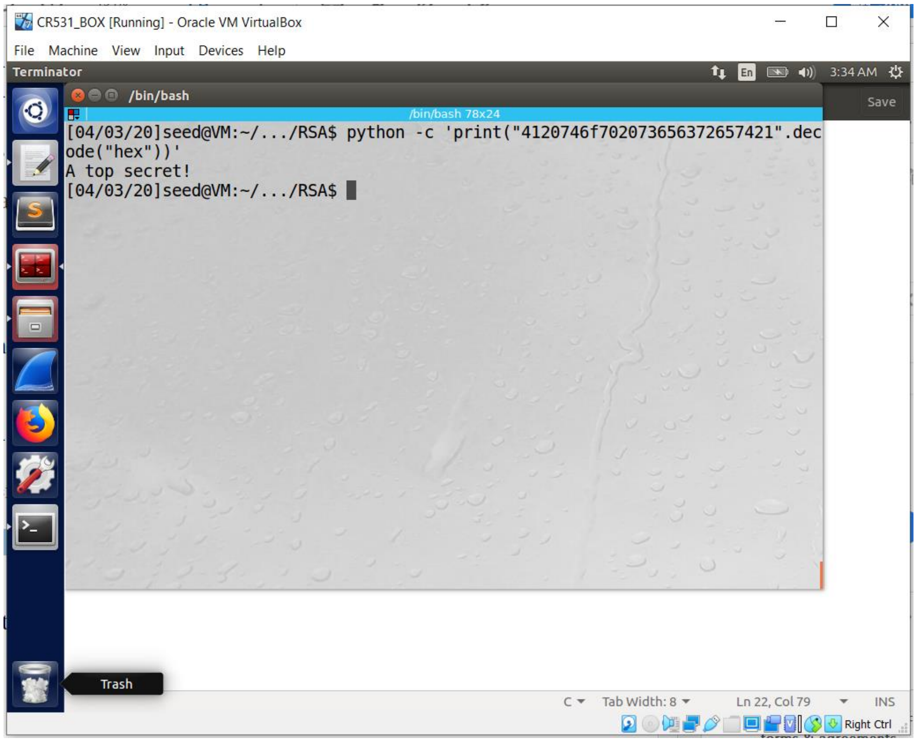


```
[04/03/20]seed@VM:~/.../RSA$ gcc Task2.c -lcrypto
[04/03/20]seed@VM:~/.../RSA$ ./a.out
The hex plaintext is =  4120746F702073656372657421
The hex ciphertext is = 6FB078DA550B2650832661E14F4F8D2CFAEF475A0DF3A75CACDC5D
E5CFC5FADC
[04/03/20]seed@VM:~/.../RSA$ 
```

## Task 3: Decrypting a Message

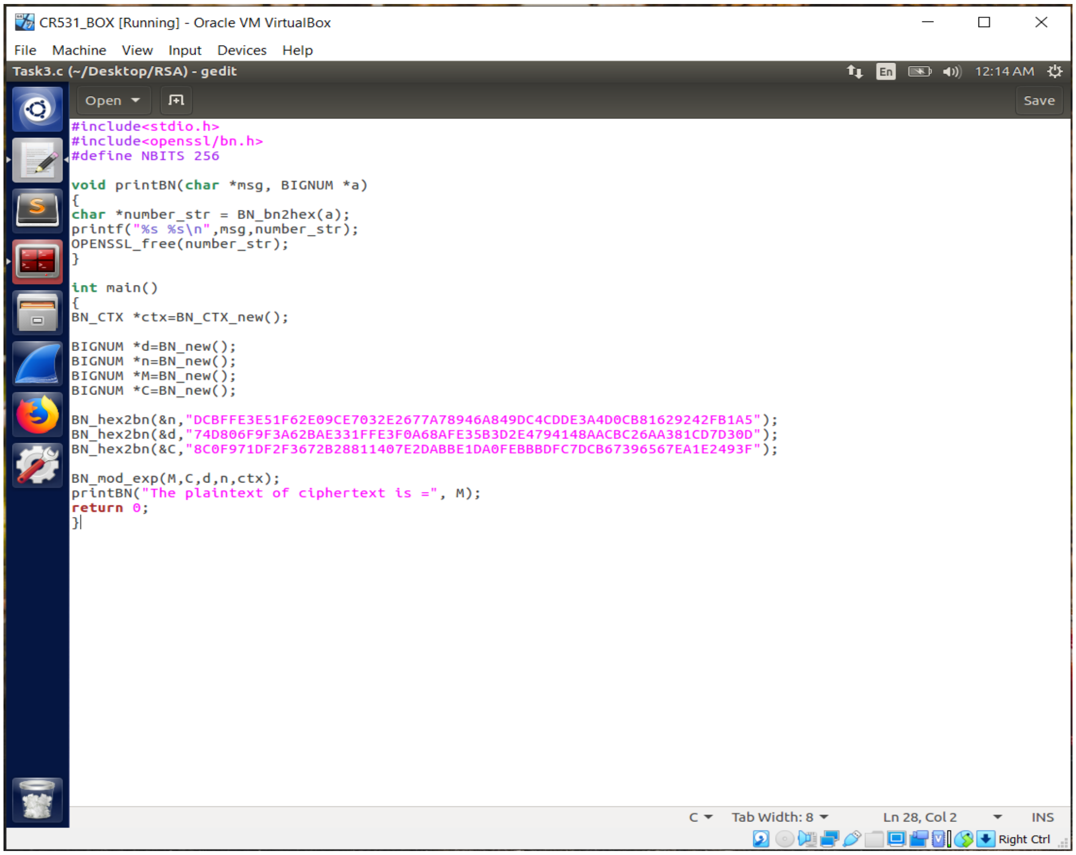
**Deliverable**. Your code should print out ”The plaintext of ciphertext ... is ...” where the dots should be replaced with the actual values.

```
$ python -c ‘print("4120746f702073656372657421".decode("hex"))’
```


CR531_BOX [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Terminator
/bin/bash
/bin/bash 78x24
Save
Trash
C Tab Width: 8 Ln 22, Col 79 INS
Right Ctrl

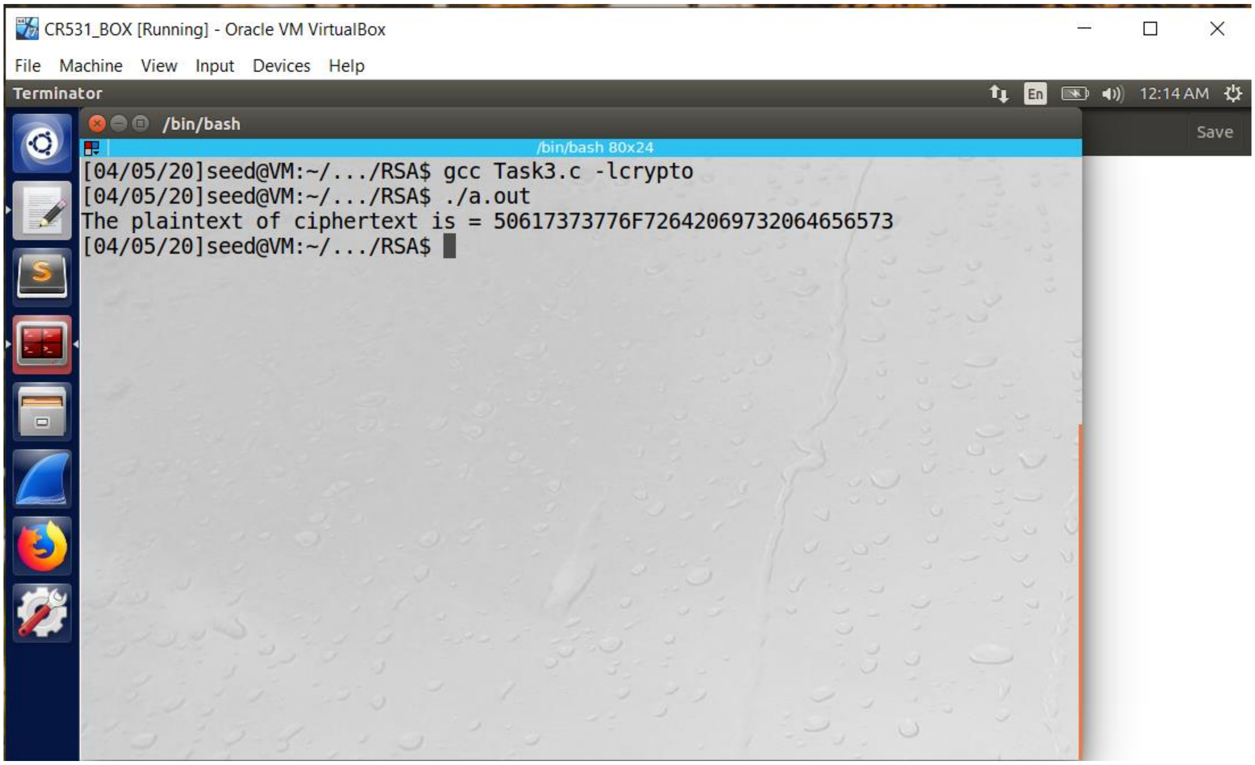
```
[04/03/20]seed@VM:~/.../RSA$ python -c 'print("4120746f702073656372657421".dec
ode("hex"))'
A top secret!
[04/03/20]seed@VM:~/.../RSA$ 
```

**Code**:

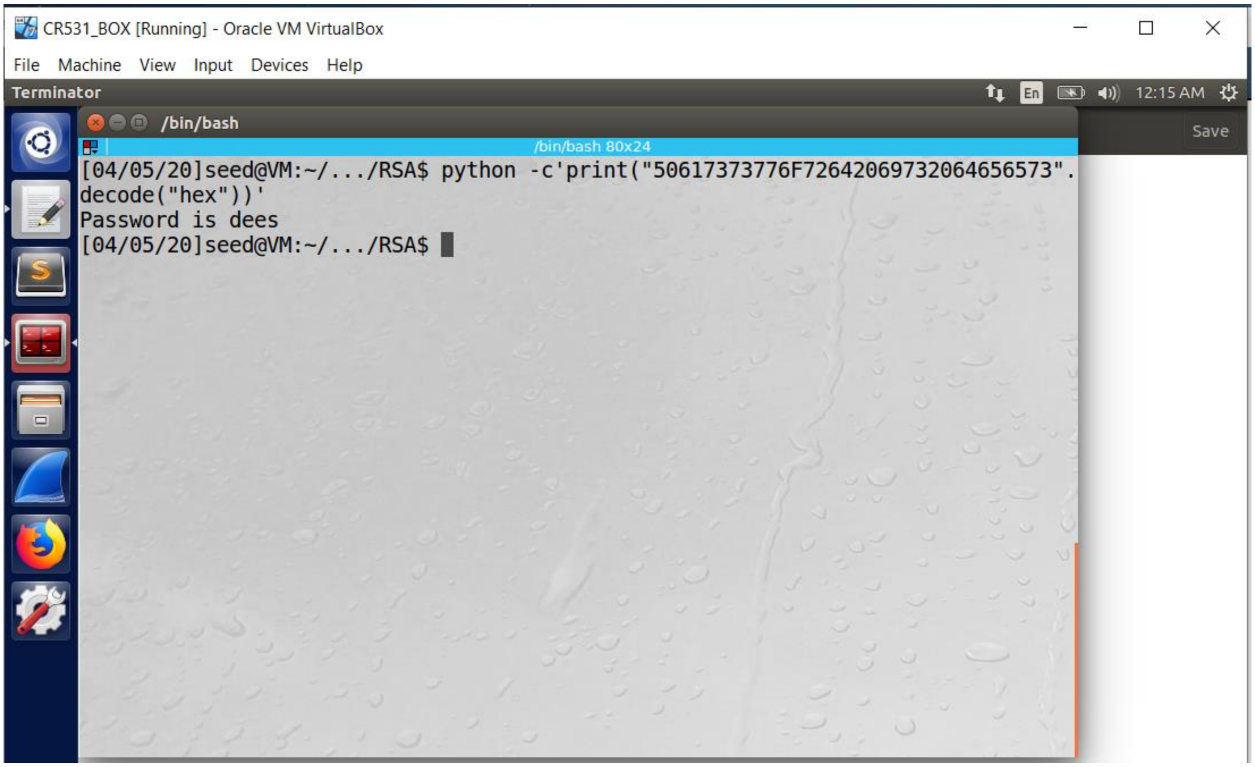

CR531_BOX [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Task3.c (~/Desktop/RSA) - gedit
Open
Save
C Tab Width: 8 Ln 28, Col 2 INS
Right Ctrl

```
#include<stdio.h>
#include<openssl/bn.h>
#define NBITS 256

void printBN(char *msg, BIGNUM *a)
{
char *number_str = BN_bn2hex(a);
printf("%s %s\n",msg,number_str);
OPENSSL_free(number_str);
}

int main()
{
BN_CTX *ctx=BN_CTX_new();

BIGNUM *d=BN_new();
BIGNUM *n=BN_new();
BIGNUM *M=BN_new();
BIGNUM *C=BN_new();

BN_hex2bn(&n,"DCBFFE3E51F62E09CE7032E2677A78946A849DC4CDDE3A4D0CB81629242FB1A5");
BN_hex2bn(&d,"74D806F9F3A62BAE331FFE3F0A68AFE35B3D2E4794148AACBC26AA381CD7D30D");
BN_hex2bn(&C,"8C0F971DF2F3672B28811407E2DABBE1DA0FEBBBDFC7DCB67396567EA1E2493F");

BN_mod_exp(M,C,d,n,ctx);
printBN("The plaintext of ciphertext is =", M);
return 0;
}
```

**Output**:


CR531_BOX [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Terminator
12:14 AM
/bin/bash
/bin/bash 80x24
Save

```
[04/05/20]seed@VM:~/.../RSA$ gcc Task3.c -lcrypto
[04/05/20]seed@VM:~/.../RSA$ ./a.out
The plaintext of ciphertext is = 50617373776F72642069732064656573
[04/05/20]seed@VM:~/.../RSA$
```

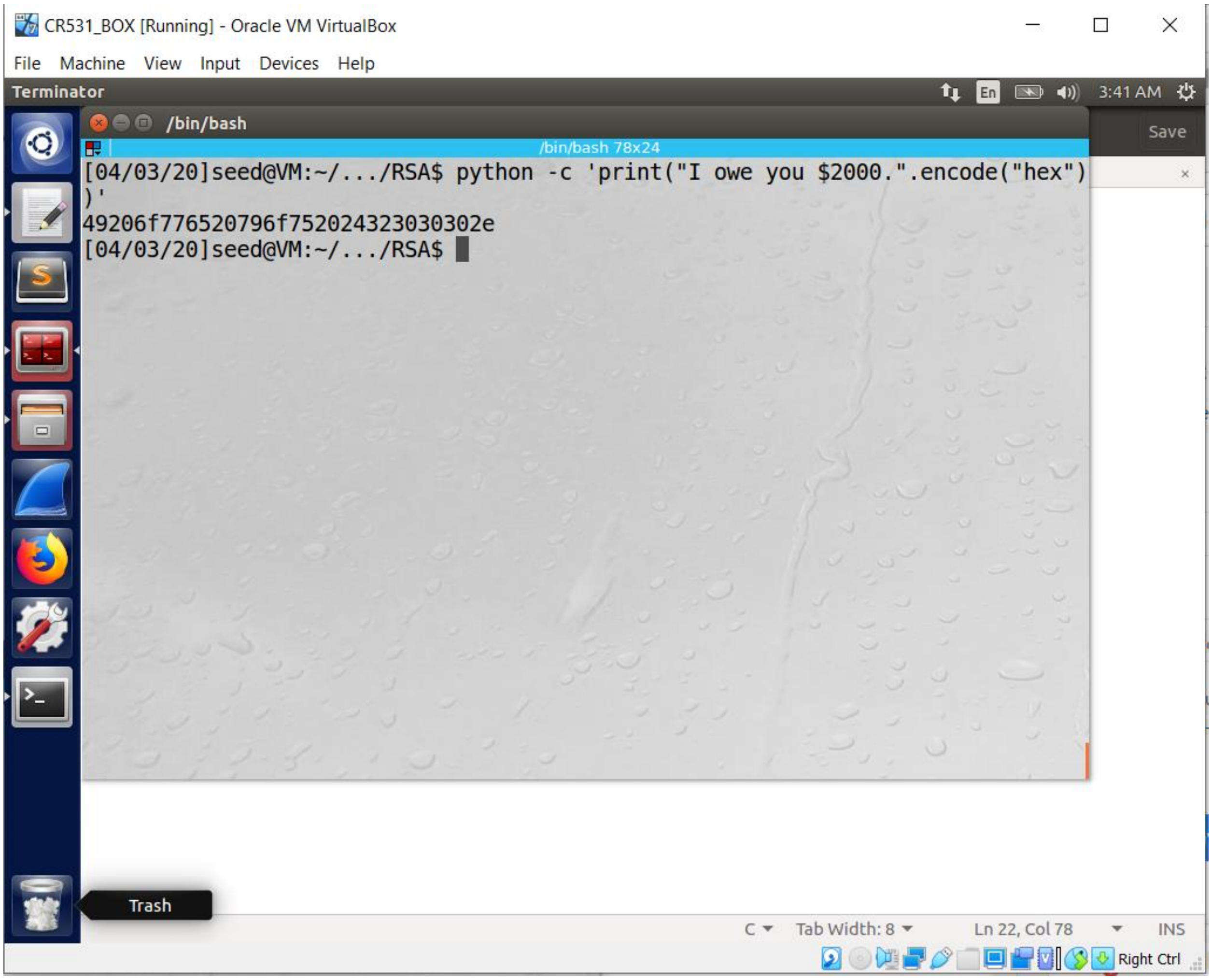

CR531_BOX [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Terminator
12:15 AM
/bin/bash
/bin/bash 80x24
Save

```
[04/05/20]seed@VM:~/.../RSA$ python -c'print("50617373776F72642069732064656573".
decode("hex"))'
Password is dees
[04/05/20]seed@VM:~/.../RSA$
```

**Task 4: Signing a Message**

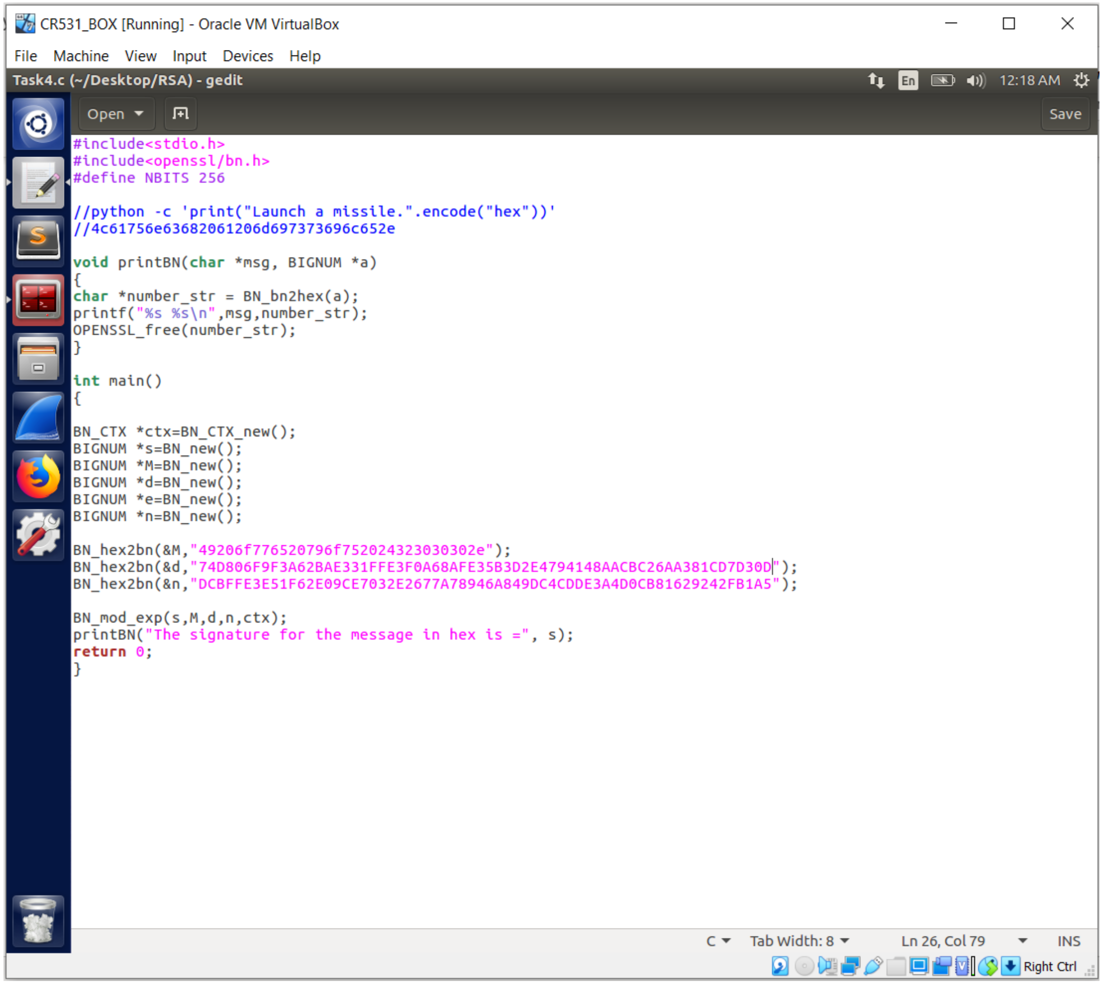
M = I owe you $2000.
**Deliverable**. Your code should print out “The signature for the message in hex is ...”, where the dots should be replaced with the actual values.

```
$ python -c ‘print("I owe you $2000.".decode("hex"))’
```



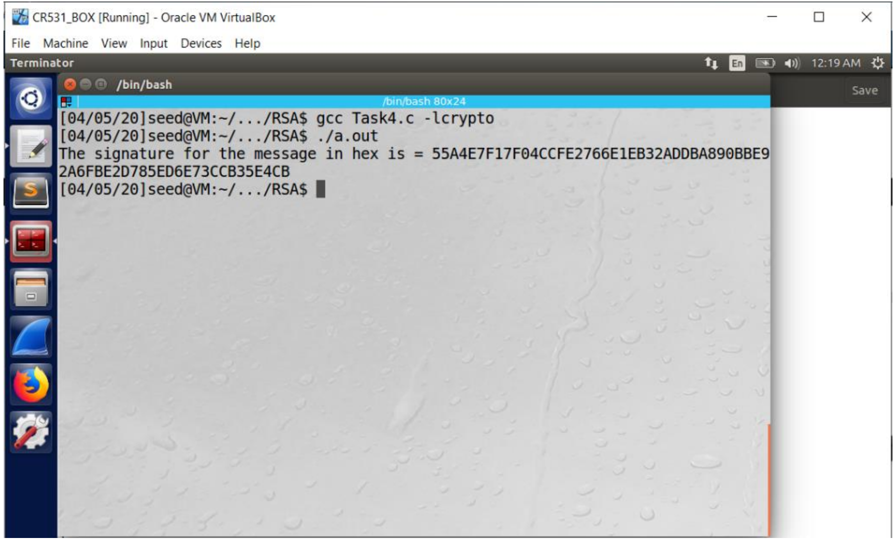
```
[04/03/20]seed@VM:~/.../RSA$ python -c 'print("I owe you $2000.".encode("hex")
)'
49206f776520796f752024323030302e
[04/03/20]seed@VM:~/.../RSA$ 
```

**Code**:


CR531_BOX [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Task4.c (~/Desktop/RSA) - gedit
12:18 AM
Open
Save
C
Tab Width: 8
Ln 26, Col 79
INS
Right Ctrl

```
#include<stdio.h>
#include<openssl/bn.h>
#define NBITS 256

//python -c 'print("Launch a missile.".encode("hex"))'
//4c61756e63682061206d697373696c652e

void printBN(char *msg, BIGNUM *a)
{
char *number_str = BN_bn2hex(a);
printf("%s %s\n",msg,number_str);
OPENSSL_free(number_str);
}

int main()
{

BN_CTX *ctx=BN_CTX_new();
BIGNUM *s=BN_new();
BIGNUM *M=BN_new();
BIGNUM *d=BN_new();
BIGNUM *e=BN_new();
BIGNUM *n=BN_new();

BN_hex2bn(&M,"49206f776520796f752024323030302e");
BN_hex2bn(&d,"74D806F9F3A62BAE331FFE3F0A68AFE35B3D2E4794148AACBC26AA381CD7D30D");
BN_hex2bn(&n,"DCBFFE3E51F62E09CE7032E2677A78946A849DC4CDDE3A4D0CB81629242FB1A5");

BN_mod_exp(s,M,d,n,ctx);
printBN("The signature for the message in hex is =", s);
return 0;
}
```

Output:

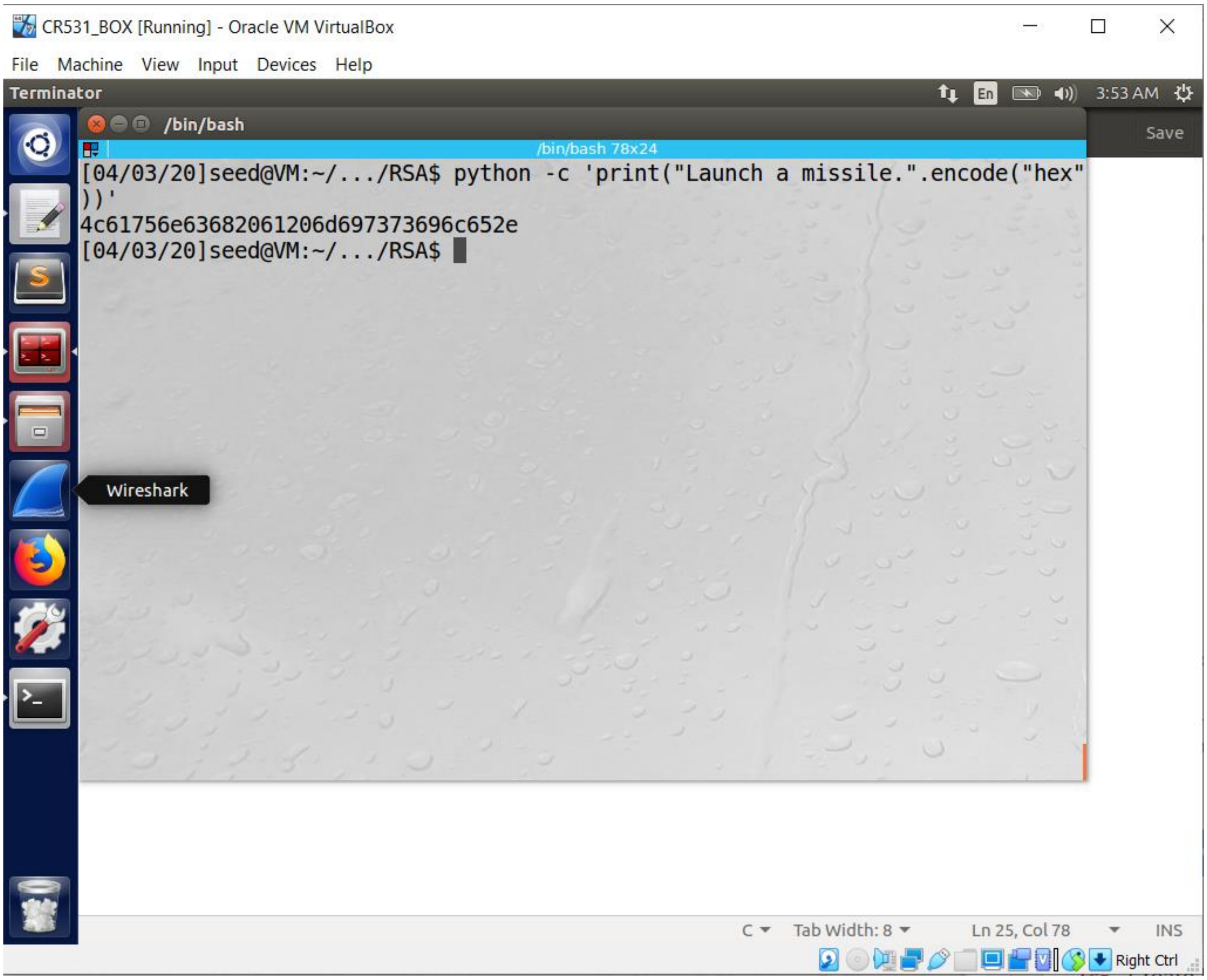

CR531_BOX [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Terminator
12:19 AM
/bin/bash
/bin/bash 80x24
Save

```
[04/05/20]seed@VM:~/.../RSA$ gcc Task4.c -lcrypto
[04/05/20]seed@VM:~/.../RSA$ ./a.out
The signature for the message in hex is = 55A4E7F17F04CCFE2766E1EB32ADDBA890BBE9
2A6FBE2D785ED6E73CCB35E4CB
[04/05/20]seed@VM:~/.../RSA$
```

### Task 5: Verifying a Signature

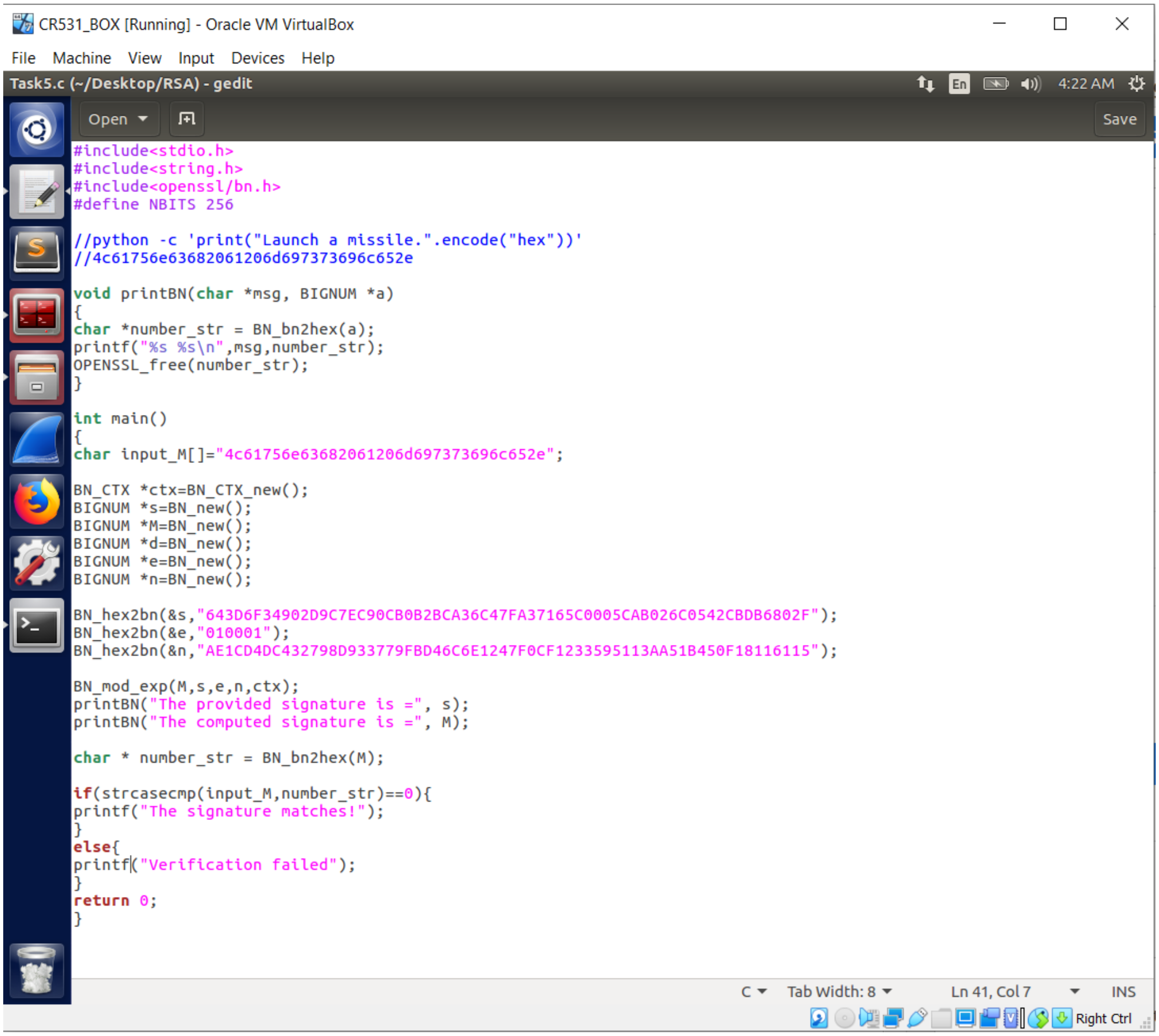
M = Launch a missile.

**Deliverable**. Your code should output the message ”The provided signature is ... and the computed signature is...”. Replace the dots with the actual values. If S and the computed signature are equal, output ” The signature matches!”, Otherwise, print ”Verification failed”.



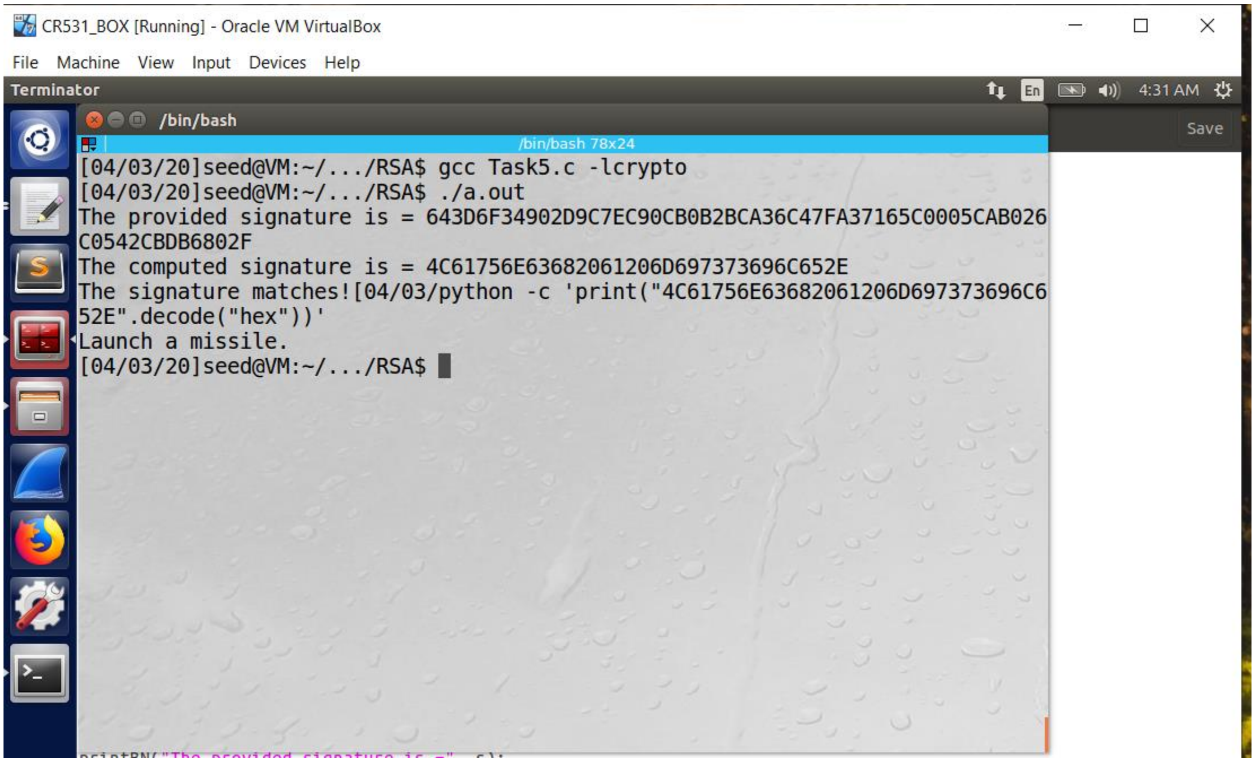
```
[04/03/20]seed@VM:~/.../RSA$ python -c 'print("Launch a missile.".encode("hex"
))'
4c61756e63682061206d697373696c652e
[04/03/20]seed@VM:~/.../RSA$ 
```

**Code**:



```c
#include<stdio.h>
#include<string.h>
#include<openssl/bn.h>
#define NBITS 256

//python -c 'print("Launch a missile.".encode("hex"))'
//4c61756e63682061206d697373696c652e

void printBN(char *msg, BIGNUM *a)
{
char *number_str = BN_bn2hex(a);
printf("%s %s\n",msg,number_str);
OPENSSL_free(number_str);
}

int main()
{
char input_M[]="4c61756e63682061206d697373696c652e";

BN_CTX *ctx=BN_CTX_new();
BIGNUM *s=BN_new();
BIGNUM *M=BN_new();
BIGNUM *d=BN_new();
BIGNUM *e=BN_new();
BIGNUM *n=BN_new();

BN_hex2bn(&s,"643D6F34902D9C7EC90CB0B2BCA36C47FA37165C0005CAB026C0542CBDB6802F");
BN_hex2bn(&e,"010001");
BN_hex2bn(&n,"AE1CD4DC432798D933779FBD46C6E1247F0CF1233595113AA51B450F18116115");

BN_mod_exp(M,s,e,n,ctx);
printBN("The provided signature is =", s);
printBN("The computed signature is =", M);

char * number_str = BN_bn2hex(M);

if(strcasecmp(input_M,number_str)==0){
printf("The signature matches!");
}
else{
printf("Verification failed");
}
return 0;
}
```

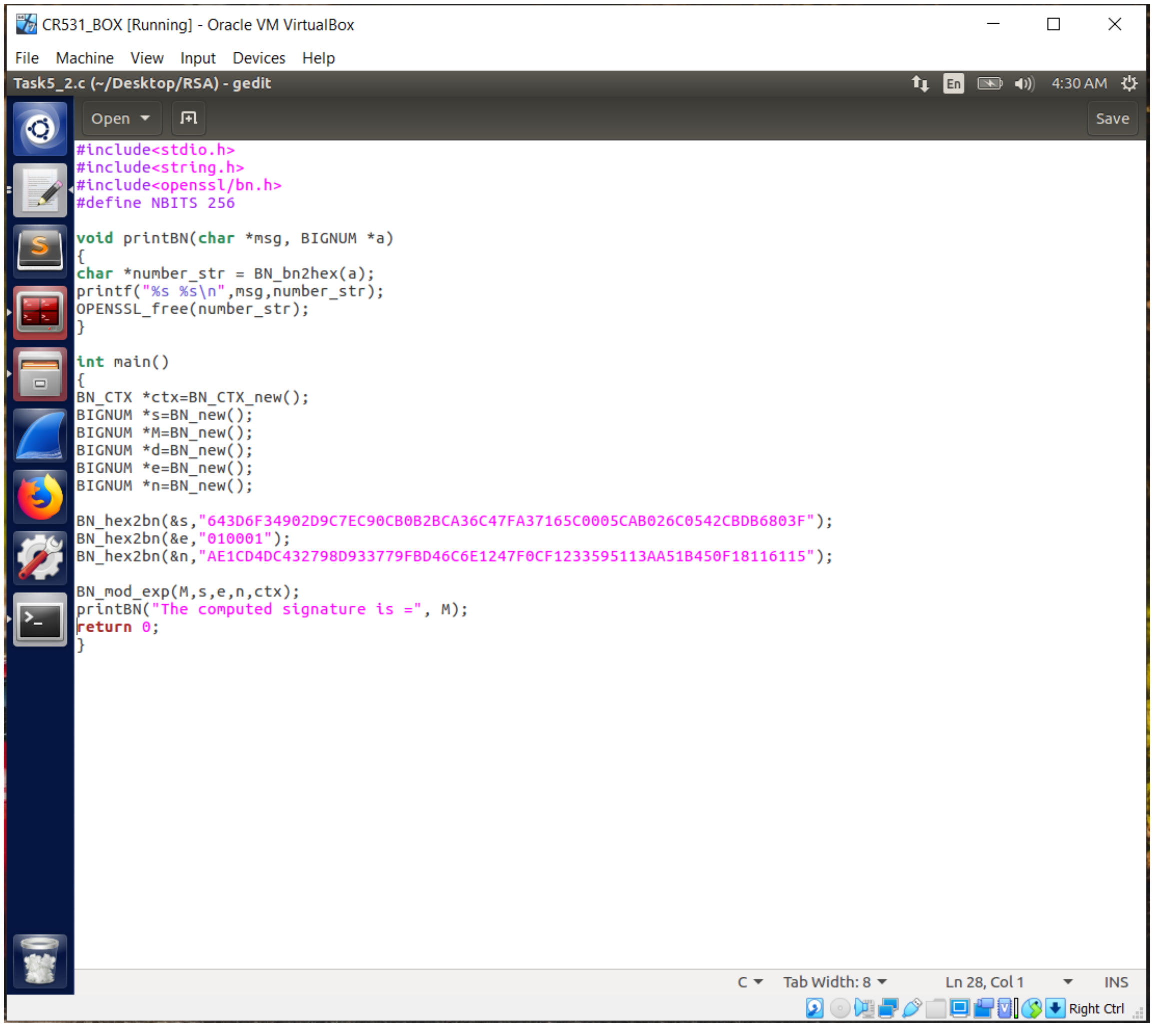
**Output**:


CR531_BOX [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Terminator
4:31 AM
/bin/bash
/bin/bash 78x24
Save

```
[04/03/20]seed@VM:~/.../RSA$ gcc Task5.c -lcrypto
[04/03/20]seed@VM:~/.../RSA$ ./a.out
The provided signature is = 643D6F34902D9C7EC90CB0B2BCA36C47FA37165C0005CAB026
C0542CBDB6802F
The computed signature is = 4C61756E63682061206D697373696C652E
The signature matches![04/03/python -c 'print("4C61756E63682061206D697373696C6
52E".decode("hex"))'
Launch a missile.
[04/03/20]seed@VM:~/.../RSA$
```

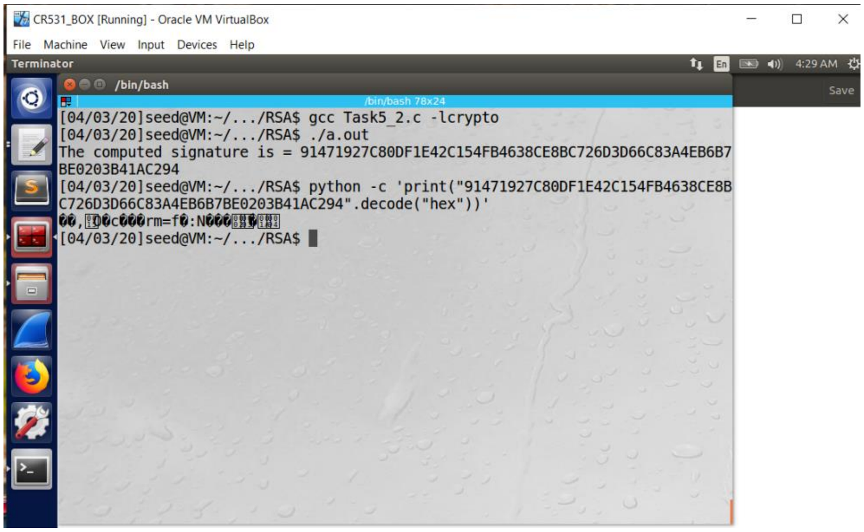
Then, add another section to your code where you repeat this task but for a corrupted signature such that the last byte of the provided signature S changes from 2F to 3F, i.e, there is only one bit of change.

Code:


CR531_BOX [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Task5_2.c (~/Desktop/RSA) - gedit
4:30 AM
Open
Save
C
Tab Width: 8
Ln 28, Col 1
INS
Right Ctrl

```
#include<stdio.h>
#include<string.h>
#include<openssl/bn.h>
#define NBITS 256

void printBN(char *msg, BIGNUM *a)
{
char *number_str = BN_bn2hex(a);
printf("%s %s\n",msg,number_str);
OPENSSL_free(number_str);
}

int main()
{
BN_CTX *ctx=BN_CTX_new();
BIGNUM *s=BN_new();
BIGNUM *M=BN_new();
BIGNUM *d=BN_new();
BIGNUM *e=BN_new();
BIGNUM *n=BN_new();

BN_hex2bn(&s,"643D6F34902D9C7EC90CB0B2BCA36C47FA37165C0005CAB026C0542CBDB6803F");
BN_hex2bn(&e,"010001");
BN_hex2bn(&n,"AE1CD4DC432798D933779FBD46C6E1247F0CF1233595113AA51B450F18116115");

BN_mod_exp(M,s,e,n,ctx);
printBN("The computed signature is =", M);
return 0;
}
```

**Output**:


CR531_BOX [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Terminator
4:29 AM
/bin/bash
/bin/bash 78x24
Save

```
[04/03/20]seed@VM:~/.../RSA$ gcc Task5_2.c -lcrypto
[04/03/20]seed@VM:~/.../RSA$ ./a.out
The computed signature is = 91471927C80DF1E42C154FB4638CE8BC726D3D66C83A4EB6B7
BE0203B41AC294
[04/03/20]seed@VM:~/.../RSA$ python -c 'print("91471927C80DF1E42C154FB4638CE8B
C726D3D66C83A4EB6B7BE0203B41AC294".decode("hex"))'
��,�D�c���rm=f�:N���������
[04/03/20]seed@VM:~/.../RSA$ █
```