Applied Cryptography (UE20CS314)

Padding Oracle Lab

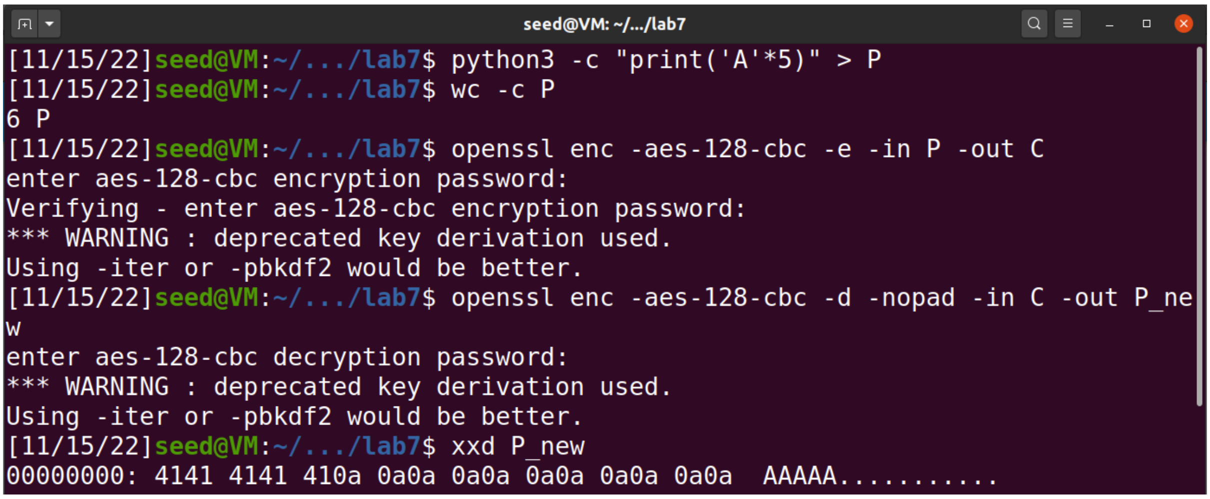
Name: Vishwa Mehul Mehta

SRN: PES2UG20CS389

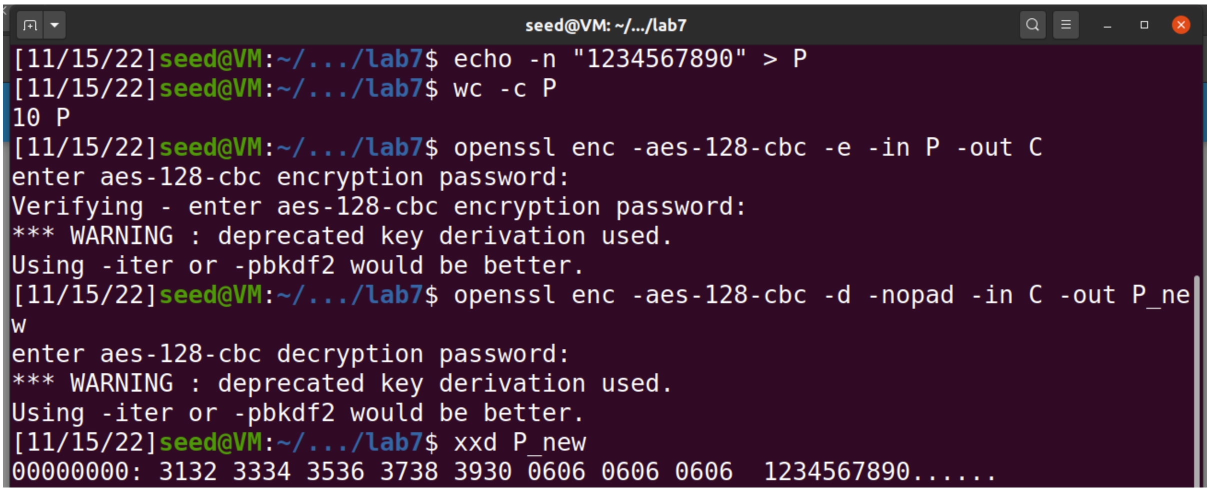
Section: F

Task 1:

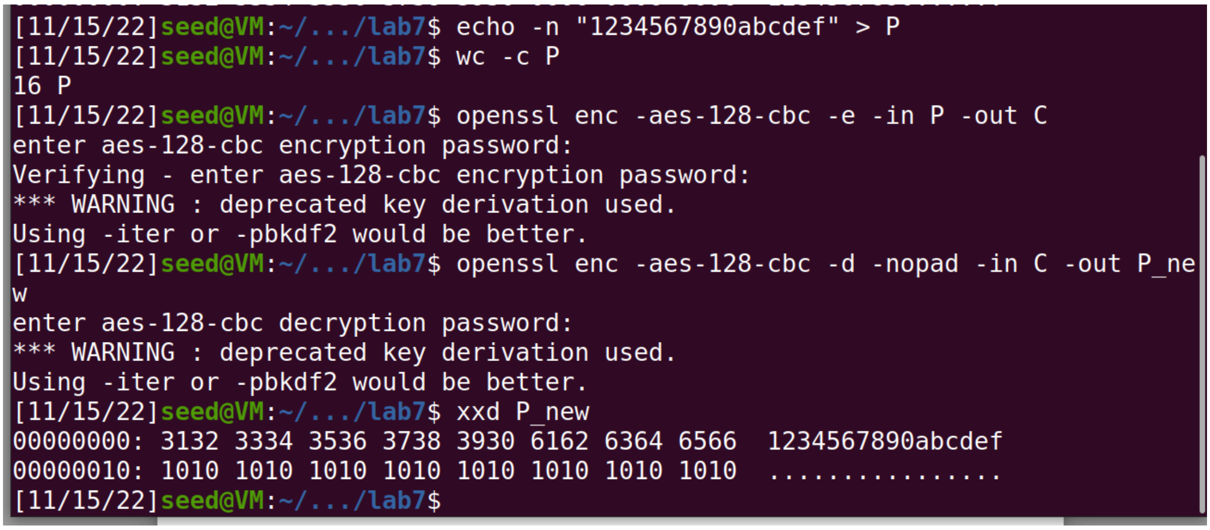
Screenshot:



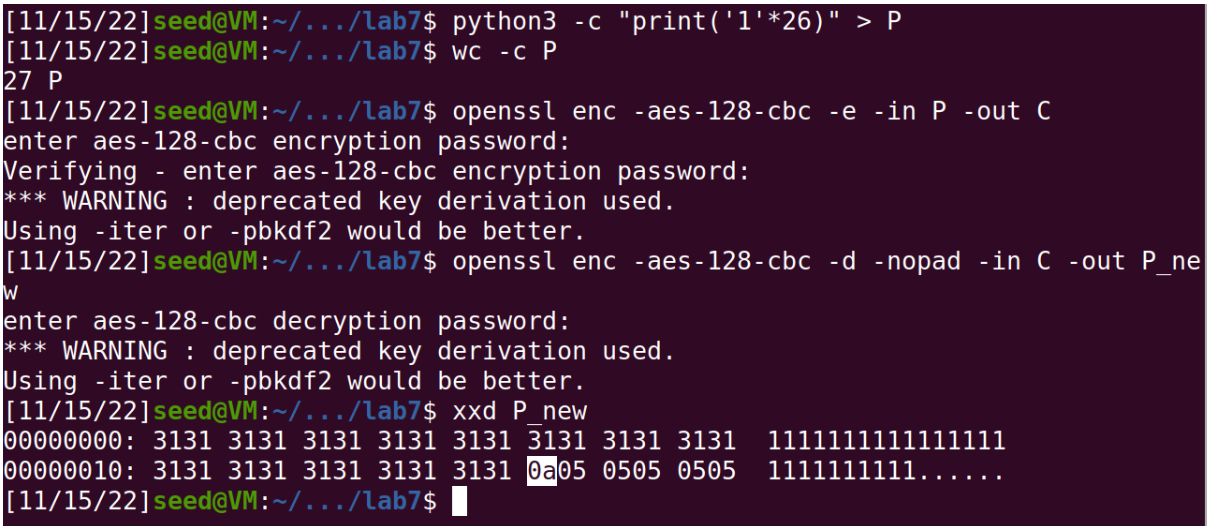
```
[11/15/22]seed@VM:~/.../lab7$ python3 -c "print('A'*5)" > P
[11/15/22]seed@VM:~/.../lab7$ wc -c P
6 P
[11/15/22]seed@VM:~/.../lab7$ openssl enc -aes-128-cbc -e -in P -out C
enter aes-128-cbc encryption password:
Verifying - enter aes-128-cbc encryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
[11/15/22]seed@VM:~/.../lab7$ openssl enc -aes-128-cbc -d -nopad -in C -out P_ne
w
enter aes-128-cbc decryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
[11/15/22]seed@VM:~/.../lab7$ xxd P_new
00000000: 4141 4141 410a 0a0a 0a0a 0a0a 0a0a 0a0a  AAAAA...........
```



```
[11/15/22]seed@VM:~/.../lab7$ echo -n "1234567890" > P
[11/15/22]seed@VM:~/.../lab7$ wc -c P
10 P
[11/15/22]seed@VM:~/.../lab7$ openssl enc -aes-128-cbc -e -in P -out C
enter aes-128-cbc encryption password:
Verifying - enter aes-128-cbc encryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
[11/15/22]seed@VM:~/.../lab7$ openssl enc -aes-128-cbc -d -nopad -in C -out P_ne
w
enter aes-128-cbc decryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
[11/15/22]seed@VM:~/.../lab7$ xxd P_new
00000000: 3132 3334 3536 3738 3930 0606 0606 0606  1234567890......
```



```
[11/15/22]seed@VM:~/.../lab7$ echo -n "1234567890abcdef" > P
[11/15/22]seed@VM:~/.../lab7$ wc -c P
16 P
[11/15/22]seed@VM:~/.../lab7$ openssl enc -aes-128-cbc -e -in P -out C
enter aes-128-cbc encryption password:
Verifying - enter aes-128-cbc encryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
[11/15/22]seed@VM:~/.../lab7$ openssl enc -aes-128-cbc -d -nopad -in C -out P_ne
w
enter aes-128-cbc decryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
[11/15/22]seed@VM:~/.../lab7$ xxd P_new
00000000: 3132 3334 3536 3738 3930 6162 6364 6566  1234567890abcdef
00000010: 1010 1010 1010 1010 1010 1010 1010 1010  ................
[11/15/22]seed@VM:~/.../lab7$
```



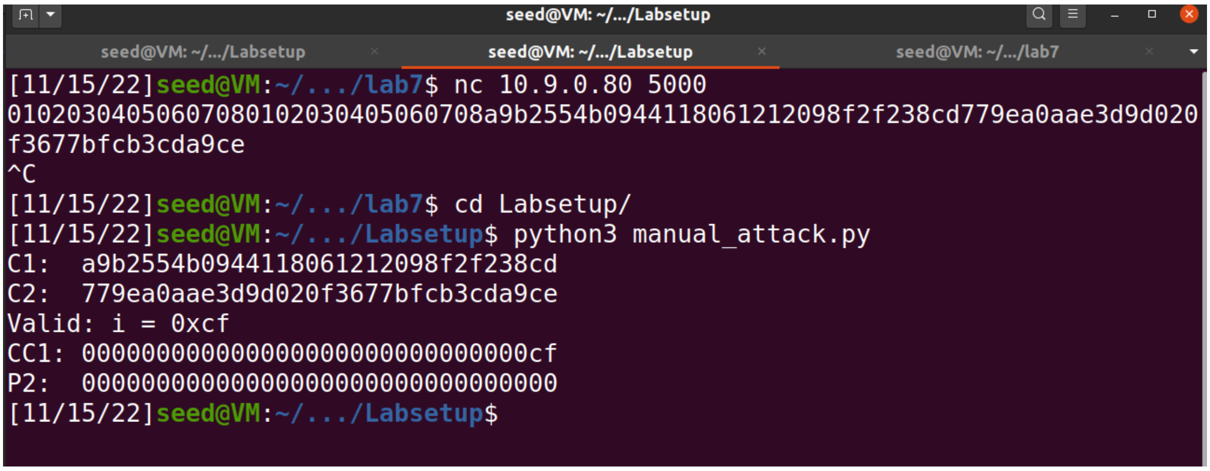
```
[11/15/22]seed@VM:~/.../lab7$ python3 -c "print('1'*26)" > P
[11/15/22]seed@VM:~/.../lab7$ wc -c P
27 P
[11/15/22]seed@VM:~/.../lab7$ openssl enc -aes-128-cbc -e -in P -out C
enter aes-128-cbc encryption password:
Verifying - enter aes-128-cbc encryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
[11/15/22]seed@VM:~/.../lab7$ openssl enc -aes-128-cbc -d -nopad -in C -out P_ne
w
enter aes-128-cbc decryption password:
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
[11/15/22]seed@VM:~/.../lab7$ xxd P_new
00000000: 3131 3131 3131 3131 3131 3131 3131 3131  1111111111111111
00000010: 3131 3131 3131 3131 3131 0a05 0505 0505  1111111111......
[11/15/22]seed@VM:~/.../lab7$
```

Observation:

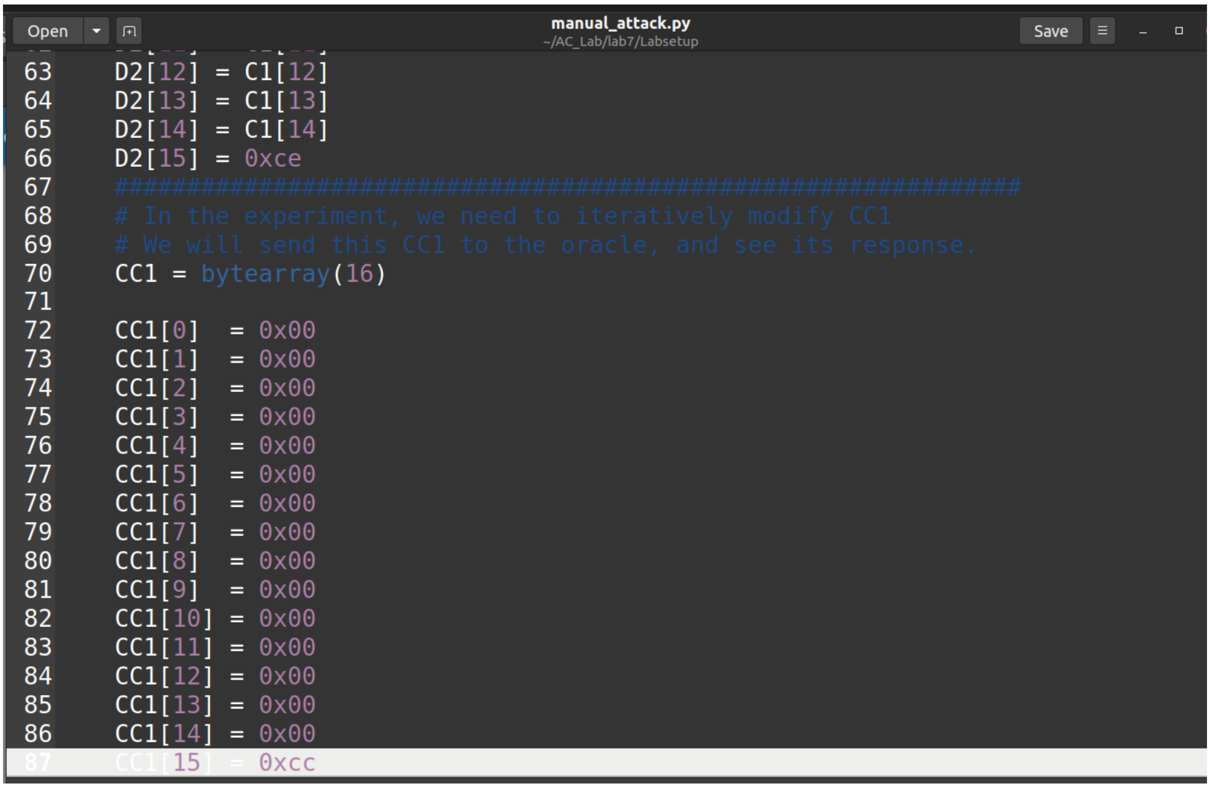
The padding for 5, 10, 16 and 27 bit is “0a”, “06”, “10” and “05” respectively. This shows that the padding value is equal to length of the characters mod 16.

Task 2:

Screenshot:

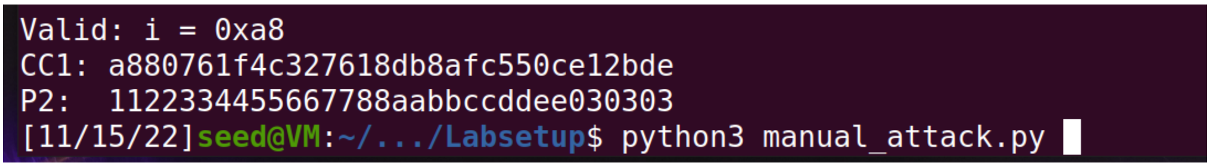


```
[11/15/22]seed@VM:~/.../lab7$ nc 10.9.0.80 5000
0102030405060708010203040506070 8a9b2554b0944118061212098f2f238cd779ea0aae3d9d020
f3677bfcb3cda9ce
^C
[11/15/22]seed@VM:~/.../lab7$ cd Labsetup/
[11/15/22]seed@VM:~/.../Labsetup$ python3 manual_attack.py
C1:  a9b2554b0944118061212098f2f238cd
C2:  779ea0aae3d9d020f3677bfcb3cda9ce
Valid: i = 0xcf
CC1: 000000000000000000000000000000cf
P2:  00000000000000000000000000000000
[11/15/22]seed@VM:~/.../Labsetup$
```



```
    D2[12] = C1[12]
    D2[13] = C1[13]
    D2[14] = C1[14]
    D2[15] = 0xce
    ##############################################################
    # In the experiment, we need to iteratively modify CC1
    # We will send this CC1 to the oracle, and see its response.
    CC1 = bytearray(16)

    CC1[0]  = 0x00
    CC1[1]  = 0x00
    CC1[2]  = 0x00
    CC1[3]  = 0x00
    CC1[4]  = 0x00
    CC1[5]  = 0x00
    CC1[6]  = 0x00
    CC1[7]  = 0x00
    CC1[8]  = 0x00
    CC1[9]  = 0x00
    CC1[10] = 0x00
    CC1[11] = 0x00
    CC1[12] = 0x00
    CC1[13] = 0x00
    CC1[14] = 0x00
    CC1[15] = 0xcc
```



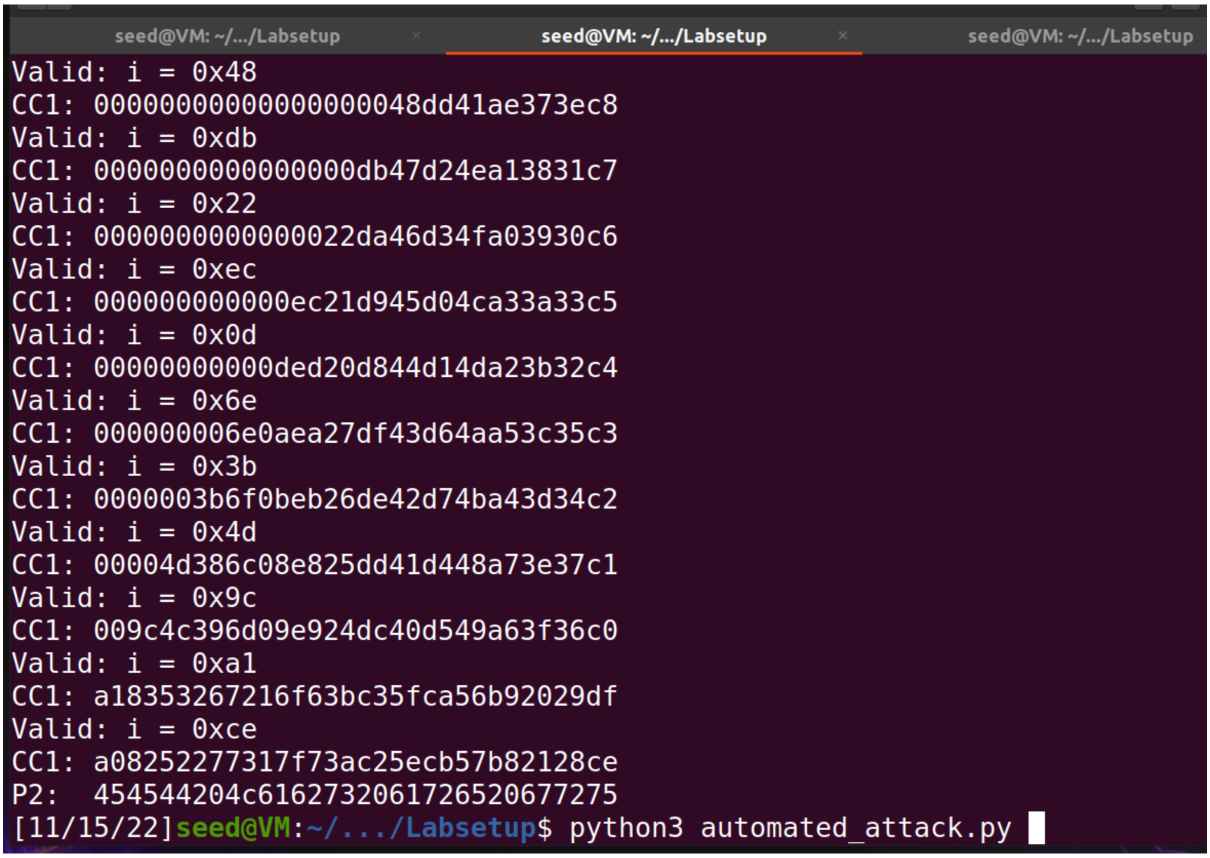
```
Valid: i = 0xa8
CC1: a880761f4c327618db8afc550ce12bde
P2:  1122334455667788aabbccddee030303
[11/15/22]seed@VM:~/.../Labsetup$ python3 manual_attack.py
```

Observation:

After performing the changes manually we have got the value of the plaintext =
1223344556677881122334455667788112233445566778 8aabbccddee030303

Task 3:

Screenshot:



```
Valid: i = 0x48
CC1: 00000000000000000048dd41ae373ec8
Valid: i = 0xdb
CC1: 0000000000000000db47d24ea13831c7
Valid: i = 0x22
CC1: 0000000000000022da46d34fa03930c6
Valid: i = 0xec
CC1: 000000000000ec21d945d04ca33a33c5
Valid: i = 0x0d
CC1: 00000000000ded20d844d14da23b32c4
Valid: i = 0x6e
CC1: 000000006e0aea27df43d64aa53c35c3
Valid: i = 0x3b
CC1: 0000003b6f0beb26de42d74ba43d34c2
Valid: i = 0x4d
CC1: 00004d386c08e825dd41d448a73e37c1
Valid: i = 0x9c
CC1: 009c4c396d09e924dc40d549a63f36c0
Valid: i = 0xa1
CC1: a18353267216f63bc35fca56b92029df
Valid: i = 0xce
CC1: a08252277317f73ac25ecb57b82128ce
P2:  454544204c616273206172652067727275
[11/15/22]seed@VM:~/.../Labsetup$ python3 automated_attack.py
```

Observation:

We now do the same process with port number 6000 and run the automated attack to find the final plaintext.