

# Assignment 3: Secret Key Crypto

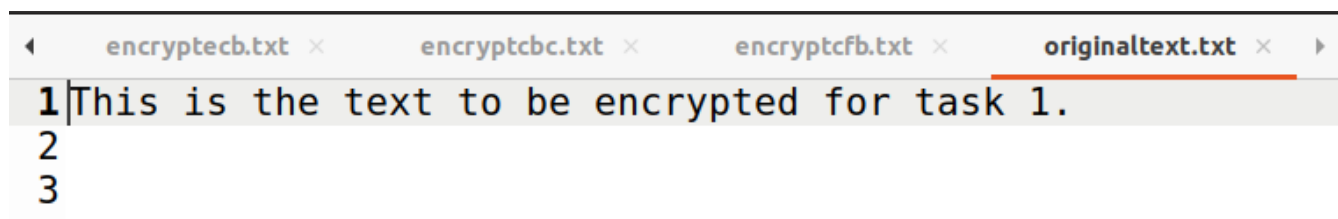
## Marco Alanis

Assignment 3 involved using several algorithms to encrypt and decrypt text and pictures. Additionally, we corrupted encrypted algorithms to understand how different algorithms handle decryption when the data has been changed.

### Task 1

On this task, we used the ECB, CBC, and CFB to encrypt and decrypt algorithms. On decryption, all of the encrypted files returned to the original file which is shown below.

Original File:



The screenshot shows a text editor with four tabs: `encryptecb.txt`, `encryptcbc.txt`, `encryptcfb.txt`, and `originaltext.txt`. The `originaltext.txt` tab is active and contains the following text:

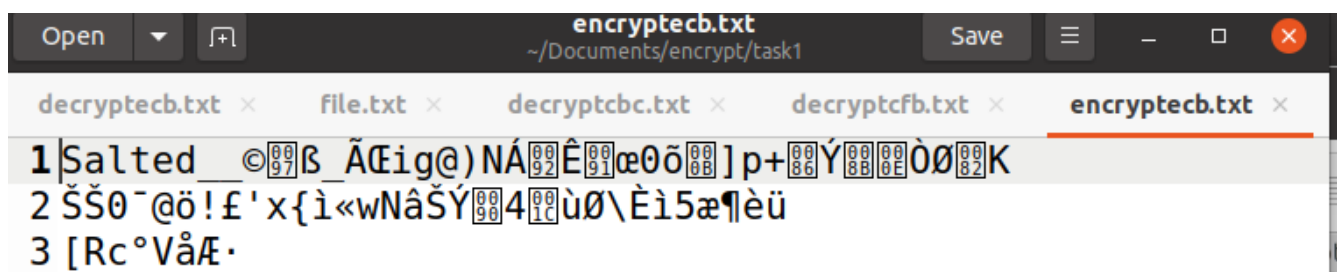
```
1|This is the text to be encrypted for task 1.
2
3
```

### Encrypt & Decrypt ECB

```
[10/29/23]seed@VM:~/../task1$ openssl enc -aes-128-ecb -e -in originaltext.txt -out encryptecb.txt -k password -iv 0102030405060708
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
warning: iv not used by this cipher
```

```
[10/29/23]seed@VM:~/../task1$ openssl enc -aes-128-ecb -d -in encryptecb.txt -out decryptecb.txt -k password -iv 0102030405060708
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
warning: iv not used by this cipher
```

Encrypted File:



The screenshot shows a text editor with five tabs: `decryptecb.txt`, `file.txt`, `decryptcbc.txt`, `decryptcfb.txt`, and `encryptecb.txt`. The `encryptecb.txt` tab is active and contains the following text:

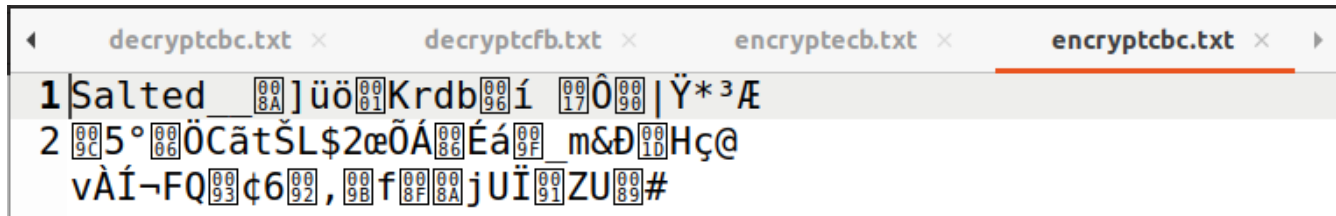
```
1|Salted __©[97]ß ÃÆig@)NÁ[92]Ê[91]æ0õ[00] p+[86]Ý[88][88]Ò0[82]K
2 ŠŠ0~@ö!£'x{ì«wNâŠÝ[90]4[10]ù0\Èì5æ¶èü
3 [Rc°VâÆ·
```

## Encrypt & decrypt CBC

```
[10/29/23]seed@VM:~/.../task1$ openssl enc -aes-128-cbc -e -in originaltext.txt -out encryptcbc.txt -k password -iv 0102030405060708
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
hex string is too short, padding with zero bytes to length
```

```
[10/29/23]seed@VM:~/.../task1$ openssl enc -aes-128-cbc -d -in encryptcbc.txt -out decryptcbc.txt -k password -iv 0102030405060708
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
hex string is too short, padding with zero bytes to length
```

Encrypted File:

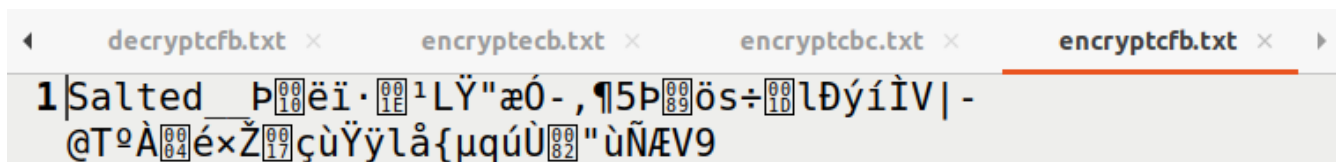


## Encrypt & Decrypt CFB

```
[10/29/23]seed@VM:~/.../task1$ openssl enc -aes-128-cfb -e -in originaltext.txt -out encryptcfb.txt -k password -iv 0102030405060708
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
hex string is too short, padding with zero bytes to length
```

```
[10/29/23]seed@VM:~/.../task1$ openssl enc -aes-128-cfb -d -in encryptcfb.txt -out decryptcfb.txt -k password -iv 0102030405060708
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
hex string is too short, padding with zero bytes to length
```

Encrypted file:



Things to note for task 1: Every file starts with Salted

## Task 2

For this task, we encrypted a picture with two different algorithms the ECB and CBS.

ECB observations: This algorithm changed colors but not shapes, the encrypted file had similarities with the original picture.

CBS observations: This algorithm changed shapes and colors, the encrypted file was irrerecognizable.

When decrypting files, both ECB and CBS encrypted files returned to the original picture.

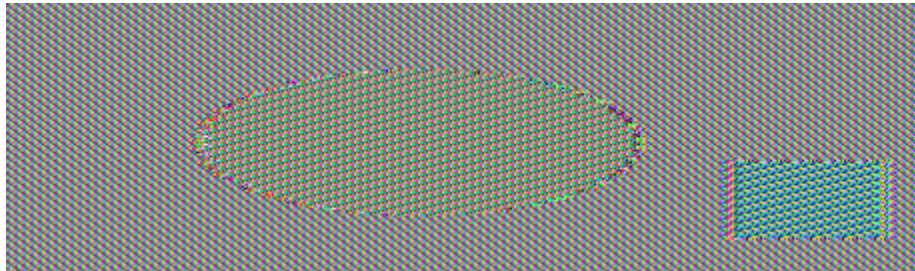
Original Picture:



Encrypt ECB:

```
[10/29/23]seed@VM:~/.../task2$ openssl enc -aes-128-ecb -e -in pic_original.bmp -out encryptecb.bmp -k password -iv 0102030405060708
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
warning: iv not used by this cipher
[10/29/23]seed@VM:~/.../task2$ head -c 54 pic_original.bmp > header
[10/29/23]seed@VM:~/.../task2$ tail -c +55 encryptecb.bmp > body
[10/29/23]seed@VM:~/.../task2$ cat header body > new.bmp
```

ECB File Result:



ECB file Decrypt

```
[10/29/23]seed@VM:~/.../task2$ openssl enc -aes-128-ecb -d -in encryptecb.bmp -out decryptecb.bmp -k password -iv 0102030405060708
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
warning: iv not used by this cipher
```

Encrypt CBC

```
[10/29/23]seed@VM:~/.../task2$ openssl enc -aes-128-cbc -e -in pic_original.bmp -out encryptcbc.bmp -k password -iv 0102030405060708
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
hex string is too short, padding with zero bytes to length
[10/29/23]seed@VM:~/.../task2$ head -c 54 pic_original.bmp > header
[10/29/23]seed@VM:~/.../task2$ tail -c +55 encryptcbc.bmp > body
[10/29/23]seed@VM:~/.../task2$ cat header body > new1.bmp
```

## CBC Encrypt Result



## Decrypt CBS

```
[10/29/23]seed@VM:~/../task2$ openssl enc -aes-128-cbc -d -in encryptcbc.bmp -out decryptcbc.bmp -k password -iv 0102030405060708
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
hex string is too short, padding with zero bytes to length
```

## Task 3

For this task, we used ECB CBS and CFB to encrypt a text file and then cause errors when changing bytes inside the Bless Hex Editor. Once inside the Bless Hex Editor, you went to offset 55 to increase the value by one. After that you decrypt the file and see results.

### Observations:

ECB and CBC decrypted files were not recognizable on the section where the change in byte was implemented; however, the CFS decrypted file had a different section where the encryption happened. Instead of encrypting the words “encrypted for task “ like in ECB and CBC. The CFB algorithm changed “is to encrypt and “. This change happened a little further down the file.

## Encrypt ecb

```
[10/29/23]seed@VM:~/../task3$ openssl enc -aes-128-ecb -e -in file.txt -out encryptecb.bin -k password -iv 010203040506070809
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
warning: iv not used by this cipher
```

## Decrypt ecb

```
[10/29/23]seed@VM:~/../task3$ openssl enc -aes-128-ecb -d -in encryptecb.bin -out decryptecb.txt -k password -iv 010203040506070809
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
warning: iv not used by this cipher
```

## Decrypted File:

```
decryptcb.txt × file.txt × decryptcbc.txt × decryptcfb.txt ×
1|This is the file that will be en€"øÛš»úł±+øøø1Aï· 3.
2First step is to encript and then open the bless hex
  editor to change a byte
```

## Encrypt cbc

```
[10/29/23]seed@VM:~/../task3$ openssl enc -aes-128-cbc -e -in file.txt -out encryptcbc.bin -k password -iv 010203040506070809
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
hex string is too short, padding with zero bytes to length
```

## Decrypt cbc

```
[10/29/23]seed@VM:~/../task3$ openssl enc -aes-128-cbc -d -in encryptcbc.bin -out decryptcbc.txt -k password -iv 010203040506070809
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
hex string is too short, padding with zero bytes to length
```

## Decrypted File:

```
decryptcb.txt × file.txt × decryptcbc.txt × decryptcfb.txt ×
1|This is the file that will be en1IÖ2«öÄnê_z^i÷ 3.
2Fhrst step is to encript and then open the bless hex
  editor to change a byte
```

## Encrypt cfb

```
[10/29/23]seed@VM:~/../task3$ openssl enc -aes-128-cfb -e -in file.txt -out encryptcfb.bin -k password -iv 010203040506070809
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
hex string is too short, padding with zero bytes to length
```

## Decrypt cfb

