CIPHER BLOCK CHAINING(CBC)
AND
OUTPUT FEEDBACK(OFB)

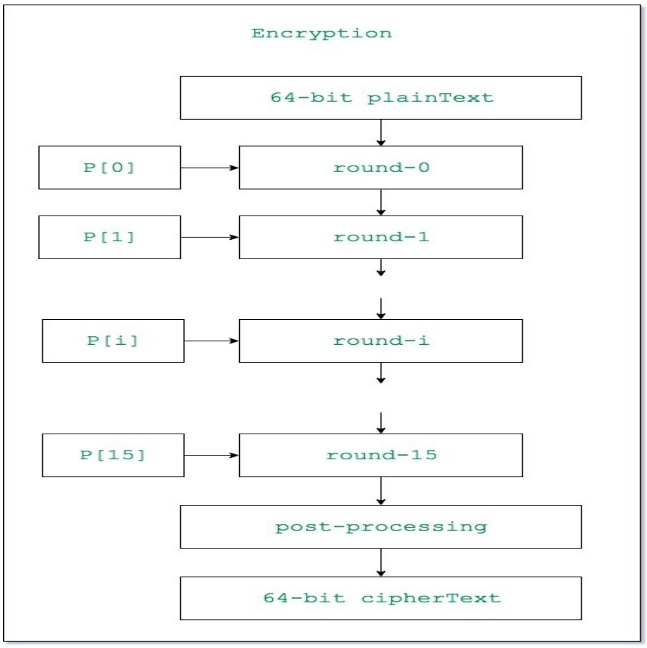
Problem Statement - Using a standard Blowfish implementation as basic encryption, write code for the CBC and OFB modes.

Theory - Blowfish is an encryption technique designed by Bruce Schneier in 1993 as an alternative to DES Encryption Technique. It is significantly faster than DES and provides a good encryption rate with no effective cryptanalysis technique found to date. It is one of the first, secure block cyphers not subject to any patents and hence freely available for anyone to use.

1. blockSize: 64-bits
2. keySize: 32-bits to 448-bits variable size
3. number of subkeys: 18 [P-array]
4. number of rounds: 16
5. number of substitution boxes: 4 [each having 512 entries of 32-bits each]

Blowfish Encryption Algorithm

The entire encryption process can be elaborated as:

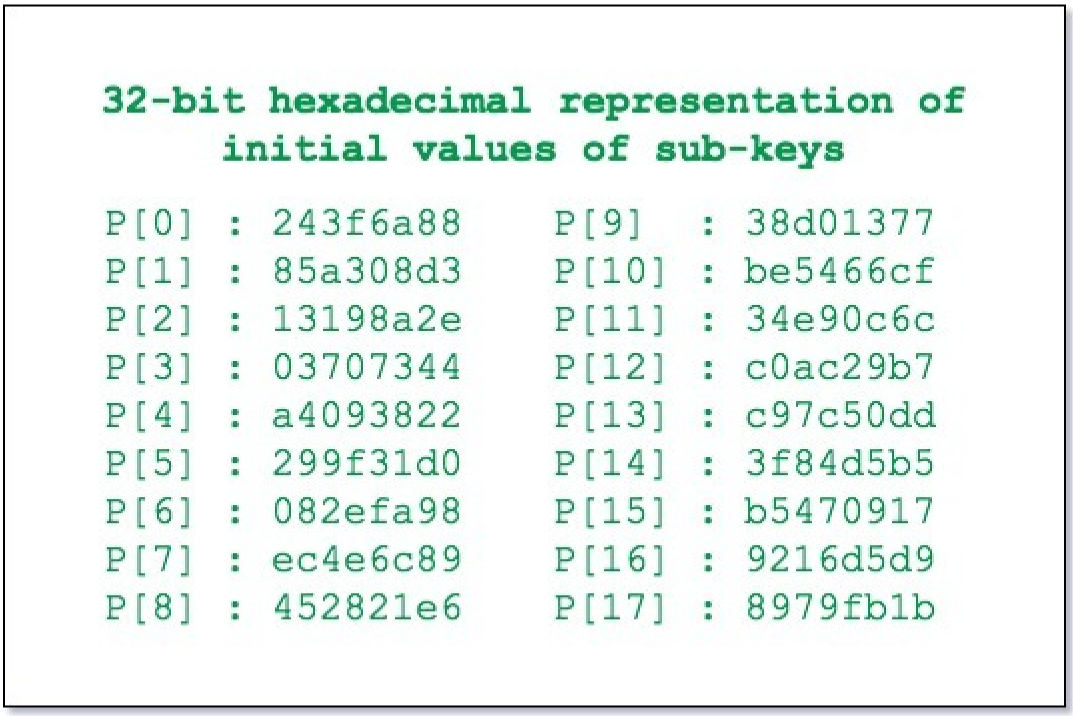


Lets see each step one by one:

Step1: Generation of subkeys:

- **18 subkeys{P[0]…P[17]} are needed in both encryption as well as the decryption process and the same subkeys are used for both the processes.**
- **These 18 subkeys are stored in a P-array with each array element being a 32-bit entry.**
- **It is initialised with the digits of pi(?).**

The hexadecimal representation of each of the subkeys is given by



32-bit hexadecimal representation of initial values of sub-keys

P[0]	243f6a88	P[9]	38d01377
P[1]	85a308d3	P[10]	be5466cf
P[2]	13198a2e	P[11]	34e90c6c
P[3]	03707344	P[12]	c0ac29b7
P[4]	a4093822	P[13]	c97c50dd
P[5]	299f31d0	P[14]	3f84d5b5
P[6]	082efa98	P[15]	b5470917
P[7]	ec4e6c89	P[16]	9216d5d9
P[8]	452821e6	P[17]	8979fb1b

Now each of the subkey is changed with respect to the input key as:
P[0] = P[0] xor 1st 32-bits of input key
P[1] = P[1] xor 2nd 32-bits of input key
.
.
.
P[i] = P[i] xor (i+1)th 32-bits of input key
(roll over to 1st 32-bits depending on the key length)
.
.
.
P[17] = P[17] xor 18th 32-bits of input key
(roll over to 1st 32-bits depending on key length)
The resultant P-array holds 18 subkeys that are used during the entire encryption process.

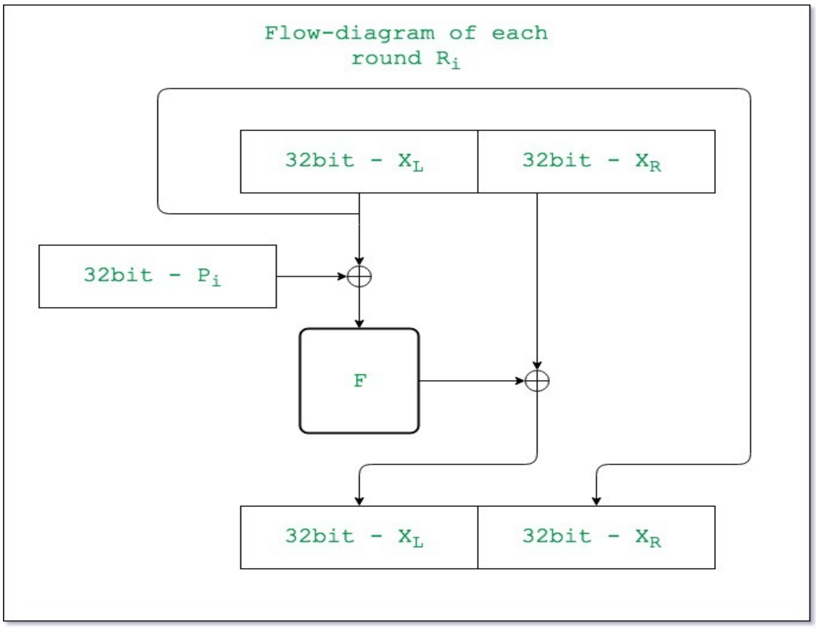
Step2: initialise Substitution Boxes:

- **4 Substitution boxes(S-boxes) are needed{S[0]…S[4]} in both encryption as well as decryption process with each S-box having 256 entries{S[i][0]…S[i][255], 0&lei&le4} where each entry is 32-bit.**
- **It is initialised with the digits of pi(?) after initialising the P-array.**
- **You can find S-BOXES here https://github.com/Ray784/Blowfish-S-boxes**

Step3: Encryption:

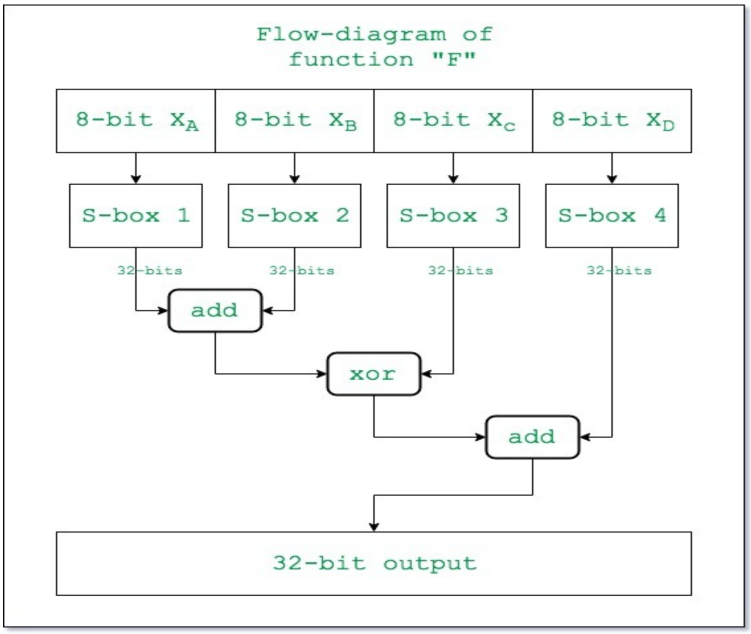
- **The encryption function consists of two parts:**

1. **Rounds: The encryption consists of 16 rounds with each round(Ri) taking inputs the plainText(P.T.) from the previous round and corresponding subkey(Pi). The description of each round is as follows:**



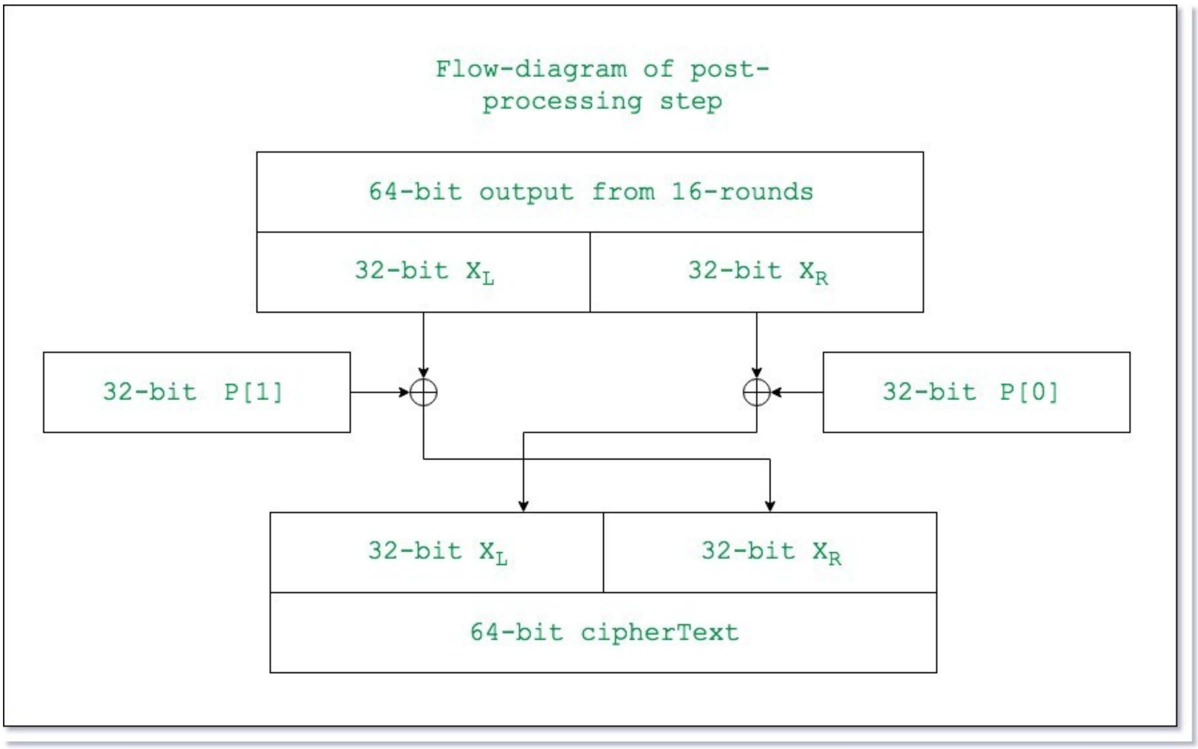
fo

The description of the function " F " is as follows:



Here the function “add” is addition modulo 2^32.

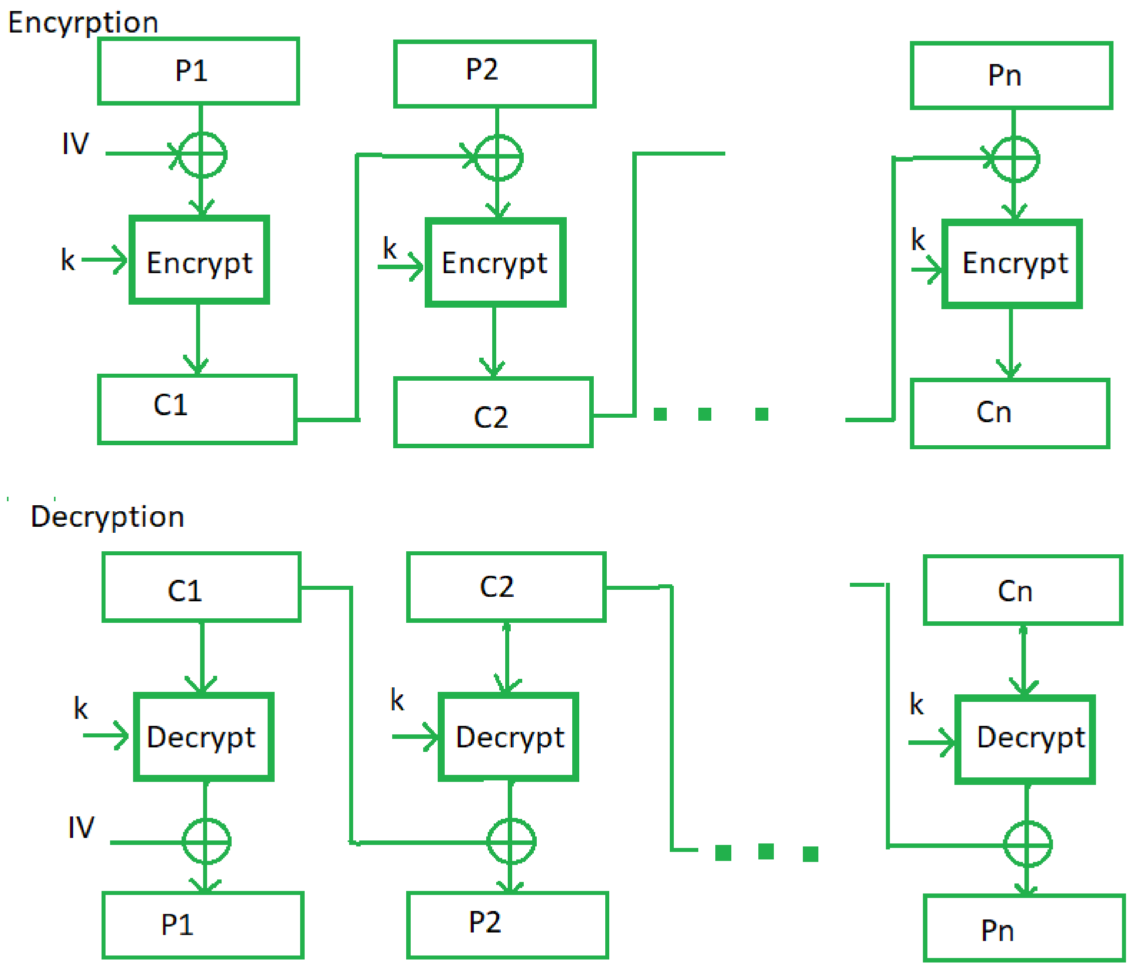
2. **<u>Post-processing:</u> The output after the 16 rounds is processed as follows:**



Cipher Block Chaining –

Cipher block chaining or CBC is an advancement made on ECB since ECB compromises some security requirements. In CBC, the previous cipher block is given as input to the next encryption algorithm after XOR with the original plaintext block. In a nutshell here, a cipher block is produced by encrypting a XOR output of the previous cipher block and present plaintext block.

The process is illustrated here:

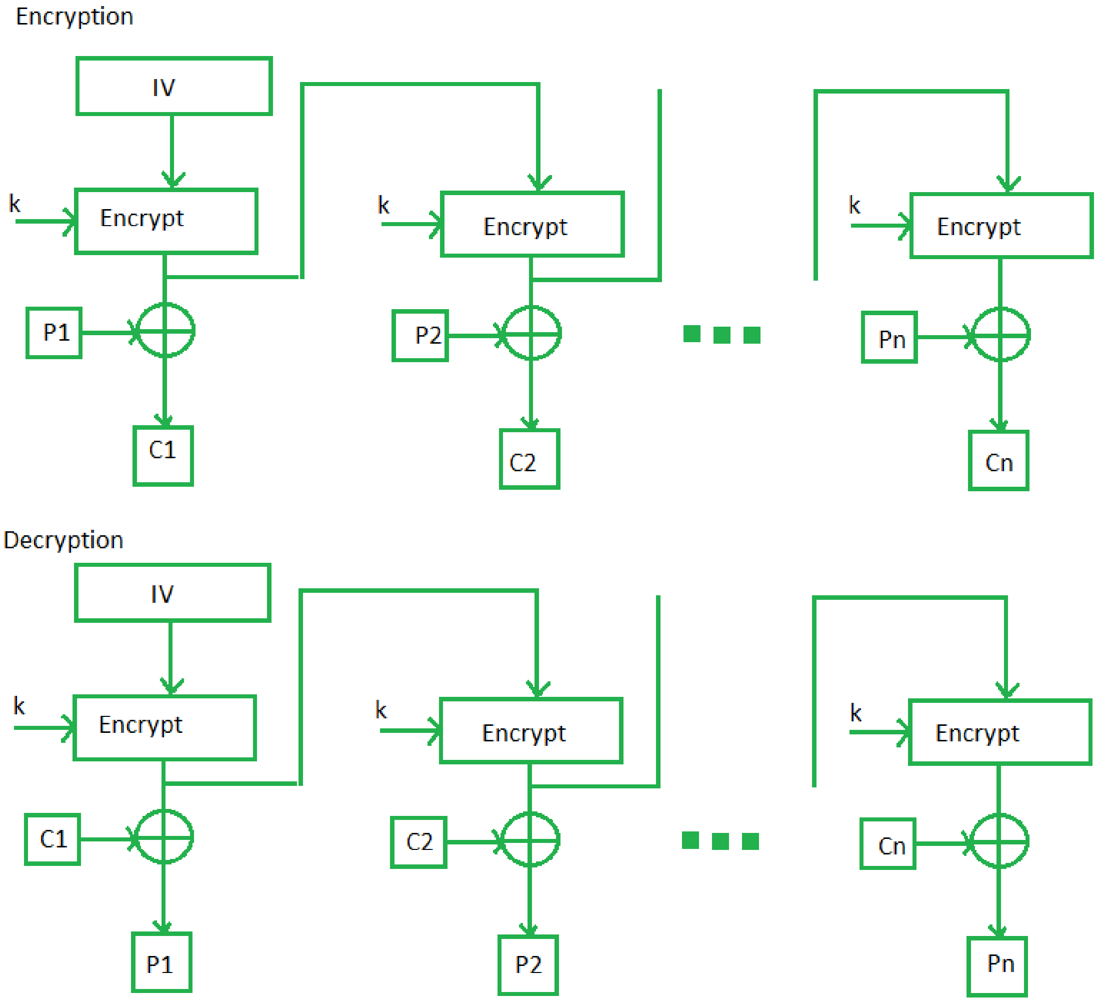


Advantages of CBC –

- CBC works well for input greater than *b* bits.
- CBC is a good authentication mechanism.
- Better resistive nature towards cryptanalysis than ECB.

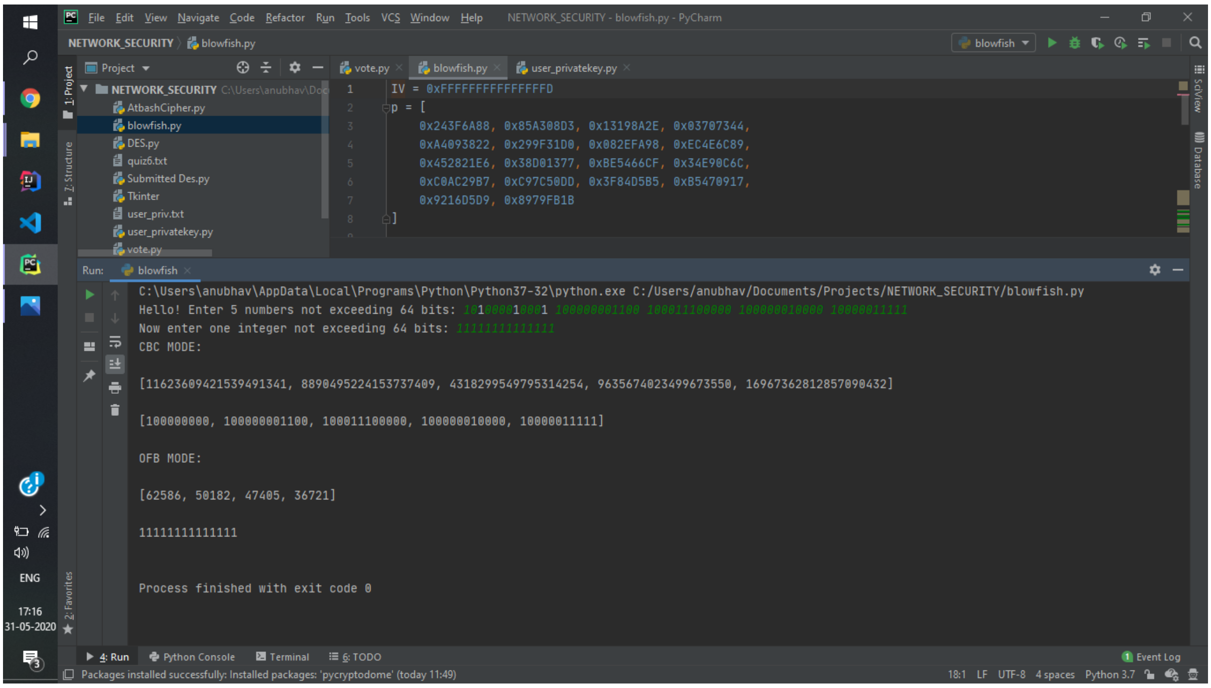
Output Feedback Mode –

The output feedback mode follows nearly the same process as the Cipher Feedback mode except that it sends the encrypted output as feedback instead of the actual cipher which is XOR output. In this output feedback mode, all bits of the block are sent instead of sending selected *s* bits. The Output Feedback mode of block cipher holds great resistance towards bit transmission errors. It also decreases dependency or relationship of cipher on plaintext.



Code Link -
https://drive.google.com/file/d/1POWCZYhyjbr5WOIGGu0Uaf8IWQIx_3Fe/view?usp=sharing

Screenshots-



```
IV = 0xFFFFFFFFFFFFFFFD
p = [
    0x243F6A88, 0x85A308D3, 0x13198A2E, 0x03707344,
    0xA4093822, 0x299F31D0, 0x082EFA98, 0xEC4E6C89,
    0x452821E6, 0x38D01377, 0xBE5466CF, 0x34E90C6C,
    0xC0AC29B7, 0xC97C50DD, 0x3F84D5B5, 0xB5470917,
    0x9216D5D9, 0x8979FB1B
]
```

```
C:\Users\anubhav\AppData\Local\Programs\Python\Python37-32\python.exe C:/Users/anubhav/Documents/Projects/NETWORK_SECURITY/blowfish.py
Hello! Enter 5 numbers not exceeding 64 bits: 101000010001 100000001100 100011100000 100000010000 10000011111
Now enter one integer not exceeding 64 bits: 11111111111111
CBC MODE:

[11623609421539491341, 8890495224153737409, 4318299549795314254, 9635674023499673550, 16967362812857090432]

[100000000, 100000001100, 100011100000, 100000010000, 10000011111]

OFB MODE:

[62586, 50182, 47405, 36721]

11111111111111


Process finished with exit code 0
```

Conclusion - With CFB, there is some data loss due to use of shift register, thus it is difficult for applying cryptanalysis.

With OFB, The biggest drawback is that the repetition of encrypting the initialization vector may produce the same state that has occurred before. It is an unlikely situation but in such a case the plaintext will start to be encrypted by the same data as previously.