Project 2 Lab Note _ Security Fundamentals

IV - **The Initialisation Vector (IV)** is a random value that is transmitted in the clear that ensures the same plaintext and key does not produce the same ciphertext. Getting proper IV is important to ensure the encryption is secure.

AES-128-CBC : Advanced Encryption Standard operates on 128 bit blocks. The blocks are chained together using XOR operation.

CBC - In **Cipher Block Chaining (CBC)** blocks are chained together using XOR.

File I/O
Useful for reading and writing to external files.

Uses different modes:

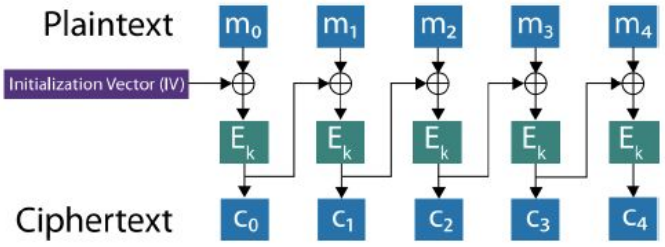
- r : open for reading (file should exist)
- w : open for writing (file need not exist)
- a : open for appending (file need not exist)
- r+ : open for reading and writing (start at beginning)
- w+ : open for reading and writing (overwrite file)
- a+ : open for reading and writing (append if file exists)

following command to encrypt/decrypt a file.
> openssl enc <cipher-type> -e / -d -in <> -out <> -k <> -iv <>

- -e encrypt
- -d decrypt
- -in input file
- -out output file
- -k secret key (in hexadecimal)
- -iv initialization vector (in hexadecimal)

> **<cipher-type> some types include:**

- **Cipher Block Chaining (CBC)**



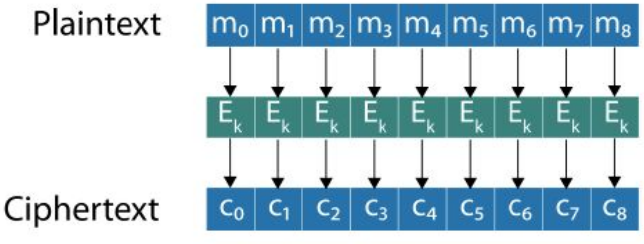
In **Cipher Block Chaining (CBC)** blocks are chained together using XOR.

The **Initialization Vector (IV)** is a random value that is transmitted in the clear that ensures the same plaintext and key does not produce the same ciphertext.

Cipher Block Chaining (CBC)

- **Cipher Feedback (CFB)**

- Electronic Codebook (ECB) : **ECB encrypts each block separately.**



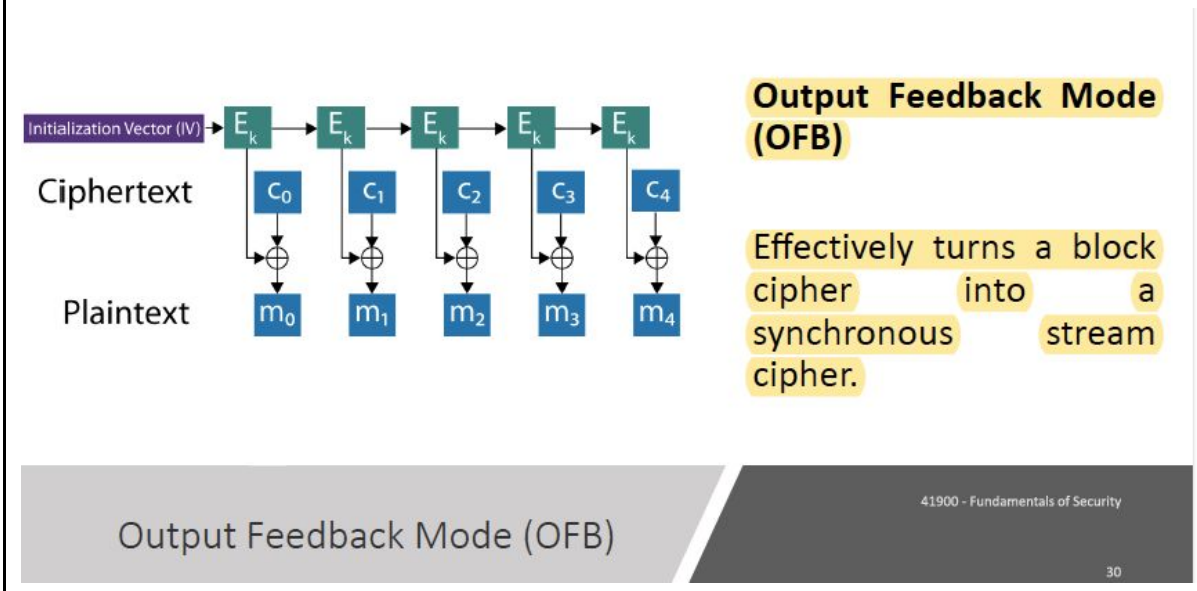
Electronic Code Book (ECB) encrypts each block separately.

ECB is generally an insecure and naïve implementation.

It is vulnerable to a range of attacks including dictionary and frequency attacks.

Electronic Code Book (ECB)

- **Output Feedback (OFB)**



EVP_MD_CTX_new()
Allocates and returns a digest context.

EVP_MD_CTX_init() initialises digest context ctx.

EVP_DigestInit_ex() sets up digest context ctx to use a digest type from ENGINE impl. ctx must be initialised before calling this function. type will typically be supplied by a function such as EVP_sha1(). If impl is NULL then the default implementation of digest type is used.

EVP_DigestUpdate() hashes cnt bytes of data at d into the digest context ctx. This function can be called several times on the same ctx to hash additional data.

EVP_DigestFinal_ex() retrieves the digest value from ctx and places it in md. If the s parameter is not NULL then the number of bytes of data written (i.e. the length of the digest) will be written to the integer at s, at most E

VP_MAX_MD_SIZE bytes will be written. After calling EVP_DigestFinal_ex() no additional calls to EVP_DigestUpdate() can be made, but EVP_DigestInit_ex() can be called to initialise a new digest operation.

EVP_MD_CTX_cleanup() cleans up digest context ctx, it should be called after a digest context is no longer needed.

-std=c90 makes gcc accept a superset of C90 (for example the more flexible C99 variable declarations anywhere in the program)

Len in C
C - **strlen() function counts the number of characters in a give string and returns the integer value**. It stops counting when null character is found.

void gen_random(char *s, const int len)
The above expression has two elements, ***s is a pointe**r (memory address) to a character, the second **variable named "len" is type of integer.**

mallock()
Dynamic memory allocation in C. In C, the library function malloc is used to allocate a block of memory on the heap. The **program** access this block of memory via a pointer that **malloc** returns. When the memory is no longer needed, the pointer is passed to free which deallocates the memory so that it can be used for other purposes.

How can we generate real random number?
Ans: We can use **computer input to generate** random number.

Entropy
Entropy is a measure of randomness. How many bytes do we have in a random poll.

Linux uses 2 devices to generate random number.
/dev/**random**
/dev/**urandom**