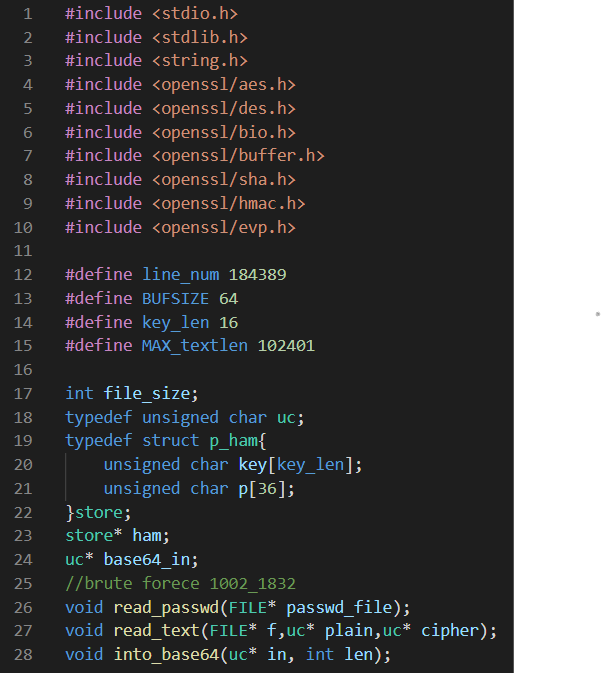
Assignment2

2016310932 배현웅

1. Overview
   1. To save hash value and key value, I made struct data structure. The code reads them from passwords.txt
   2. From PlaintextCiphertext.txt. read plain text and cipher text. (function : read\_text())
   3. First of all, from line 1st to line 184,389th, code encrypt by DES and AES, encrypted text would be converted to base64 encoding using into\_base64() function. Then, check whether it is same as cipher text given. I use Brute-force attack, Big O would be square of n
2. Code



Line 12 : Since passwords line length is 184,389, to save password & hash key value, line\_num is defined

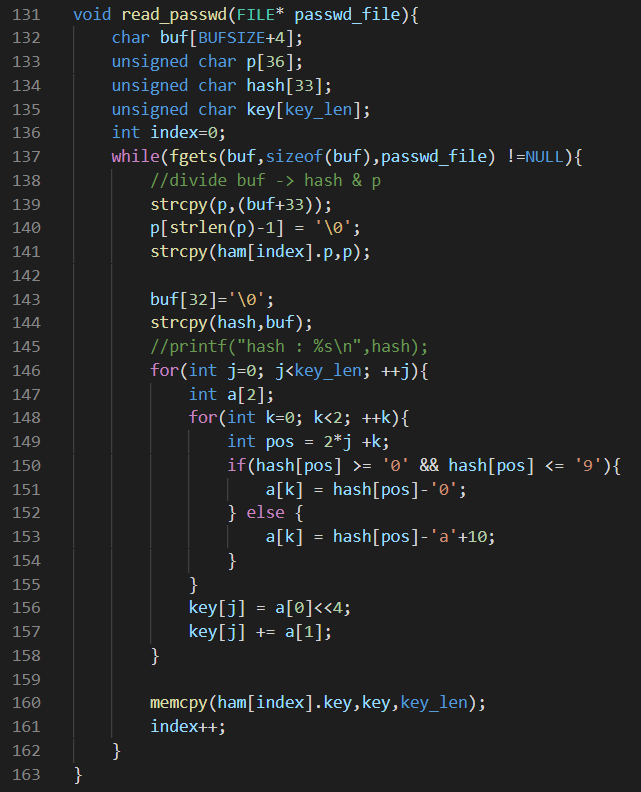
Line 13,14 to store key value& buffer length, they are defined

Line 15 : Maximum plaintext length is 100KB(100\*1024 = 102400), Maxtextlen is defined

Line 19-23 : To store password value, I made store struct data structure.

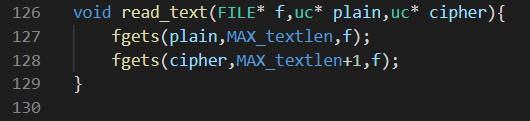
Line 24 : to store base64-encoded value, I use it.

* Read\_passwd()



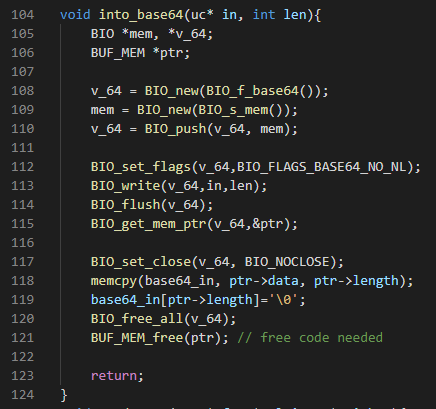
In this function, it reads every line of passwords.txt text which is consists of md5 hash value and password value. Then MD5 hash value which is saved as unsigned char converted into int value every 2 bytes(hash -> int a[2] -> key[] ).After divided into md5 value and password value, they are into ham struct which saved for password value and hash value.

* Read\_text



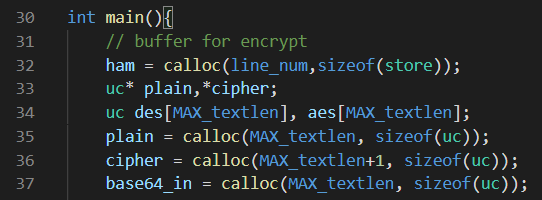
Read\_text function just read PlaintextCiphertext.txt, then divided into plaintext and cipher text and save into plain, cipher data which defined in main function

* Into\_base64

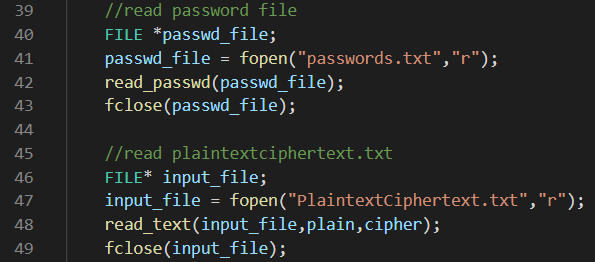


To convert AES-128(DES(m)) encrypted value, I made into\_base64 function with OPENSSL library bio.h, buffer.h, and etc. With BIO function, the encoding of base64 continues. Not to be divded into 64 bytes, I use BIO\_set\_close with BIO\_NOCLOSE.

* Main.c

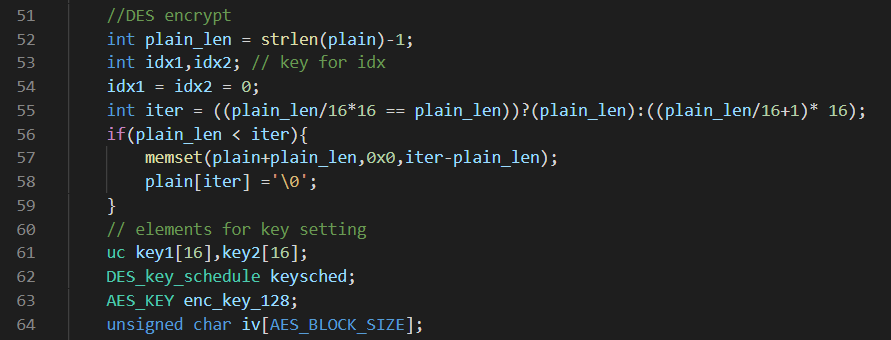


This section is for dynamic allocation text value for encryption and so on.

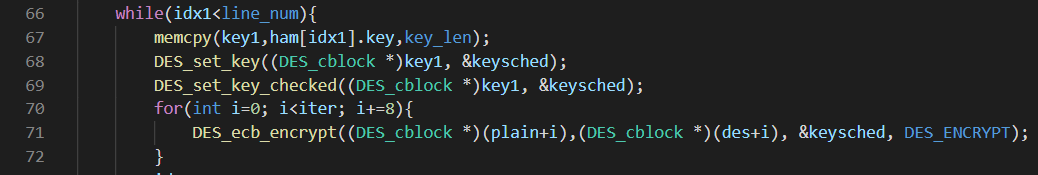


Line 40-43 : Read passwords.txt value to store password value into ham structure.

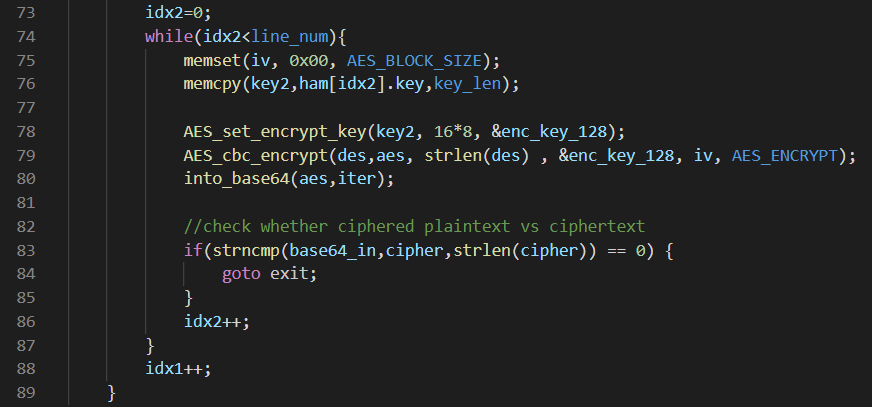
Line 46-49 : read plaintext and ciphertext.



Line 52-59 : to pad the 0 value for unaligned text with 16 bytes. Check plaintext length which is not included with newline. Then pad 0 value with memset.

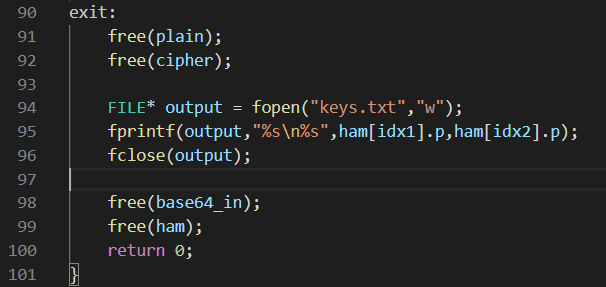


To break cipher text, the code try every values which is in passwords.txt. So, plain text is encrypted with DES.



Then encrypt DES encrypted text with AES-128 encryption. To AES encryption, I set iv to be 0, and copy possible key value from ham. Then encrypt (line 78-79). After that, this text would be encoded with base64. (line 80)

Finally, compare with ciphertext which is from PlaintextCiphertext.txt. If they are same, brute-force attack is finished (goto exit label)



To finish code, free all dynamic allocated value, and write key value into keys.txt(output file)