**信息安全技术实验报告**

**姓名: 梁莉莉**

**学号: 2017011581**

**专业：软件工程**

日期:2019年04月01日

**Lab1: 基于Openssl的密钥加解密实验**

**一、实验目的（简要写，不必全部照抄实验要求文档中的内容）**

1.熟悉加密的概念，熟悉和了解加密算法(cipher)、加密模式(encryption mode)、以及初始向量(IV)的定义与作用。

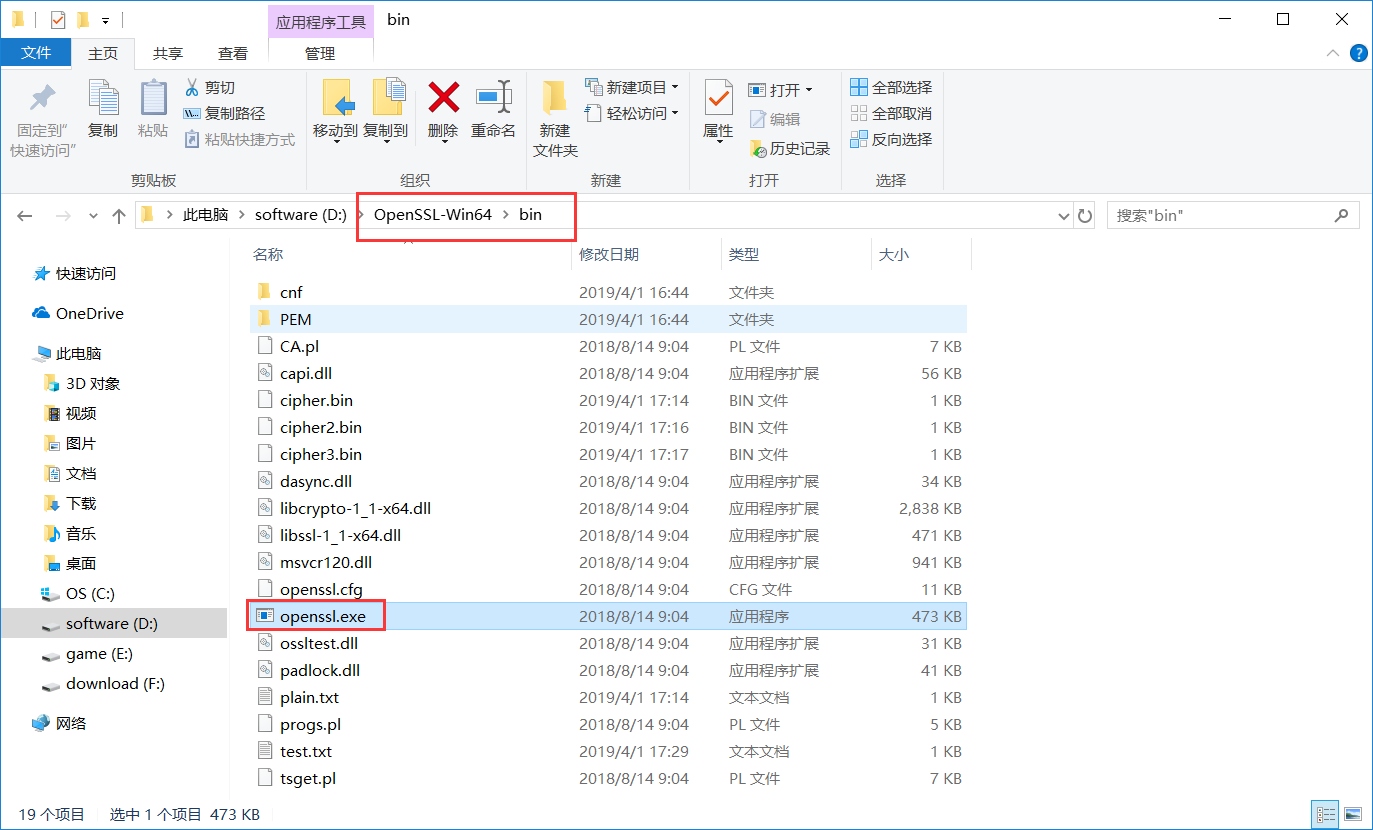
2.通过使用工具来编写基于Openssl库的C语言程序来实现消息的加密/解密。

**二、实验环境（如实给出实验平台）**

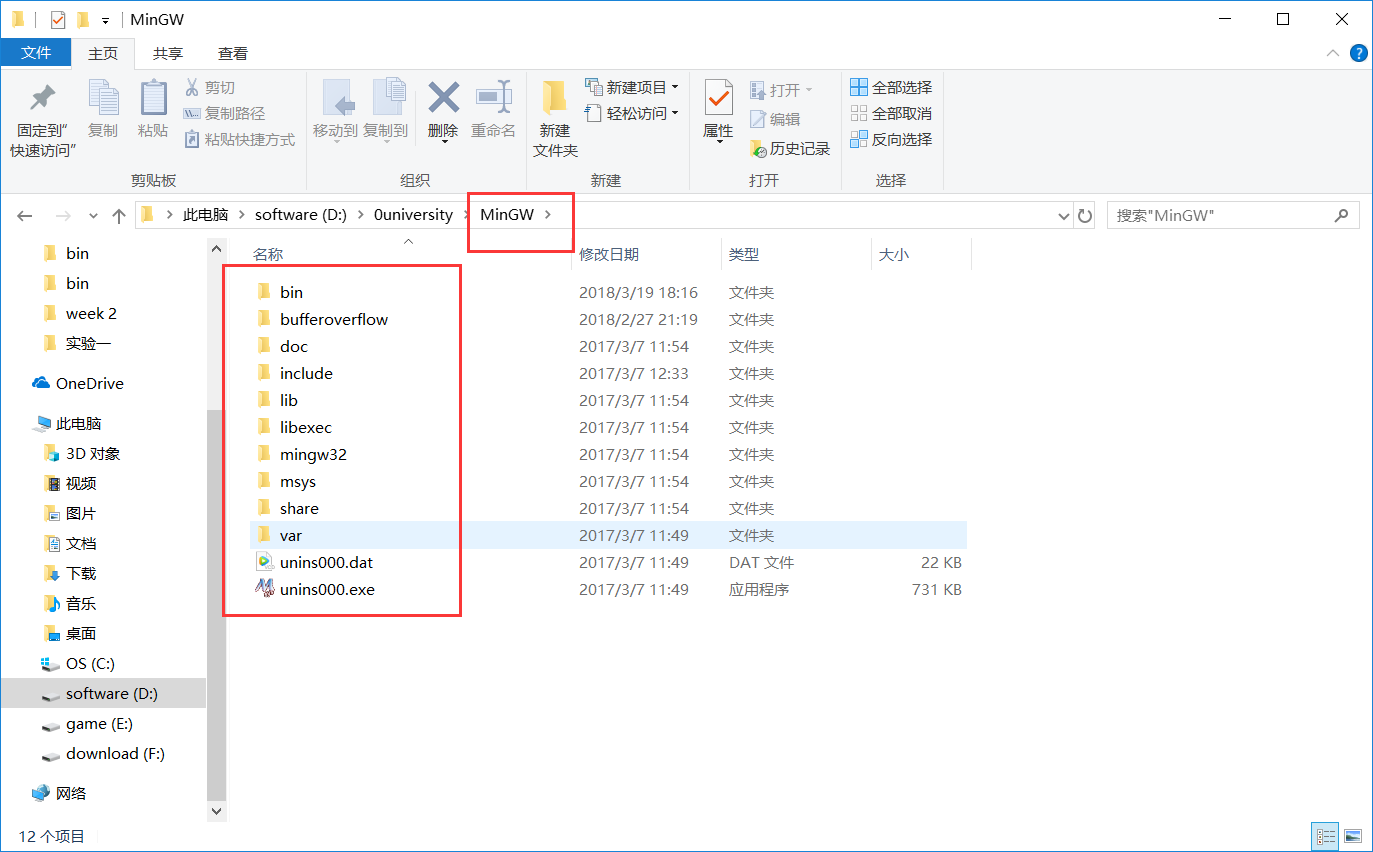
1.Windows10平台 + MinGW + OpenSSL库 + WinHex编辑器

2.环境搭建：

①下载安装OpenSSL库（版本：Win64OpenSSL-1\_1\_0i）

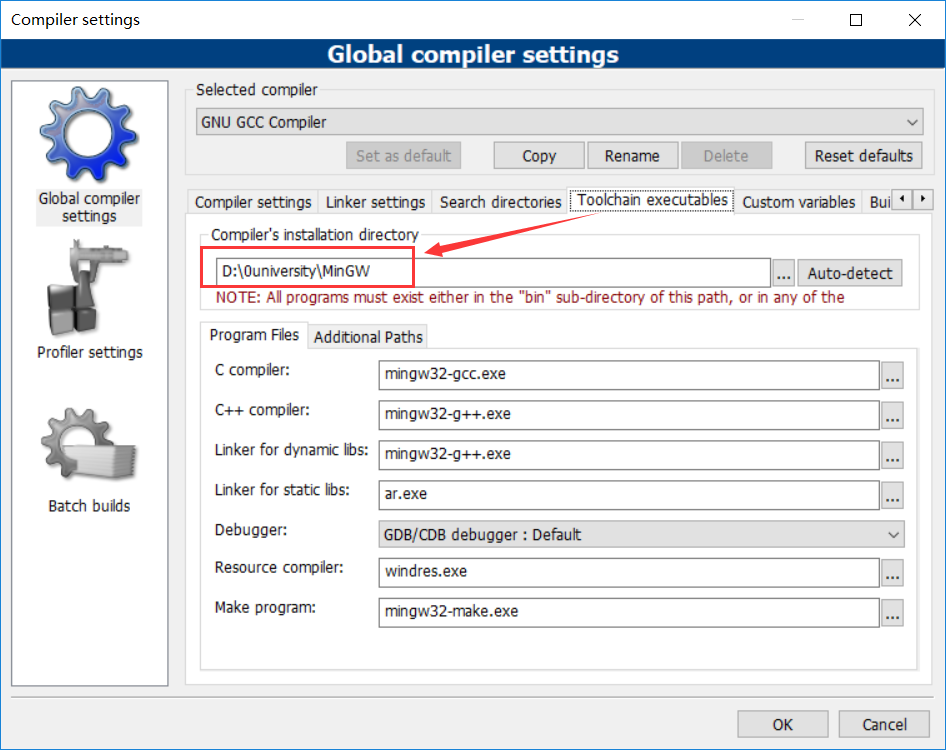


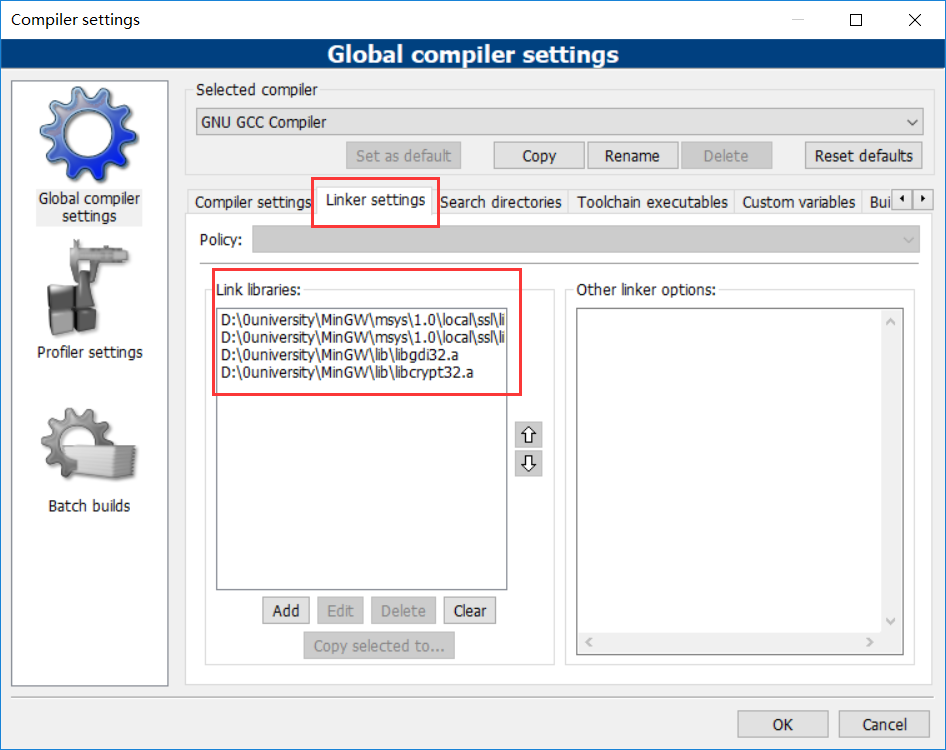
②搭建MinGW编译器（版本：使用的是Bb平台上的版本）

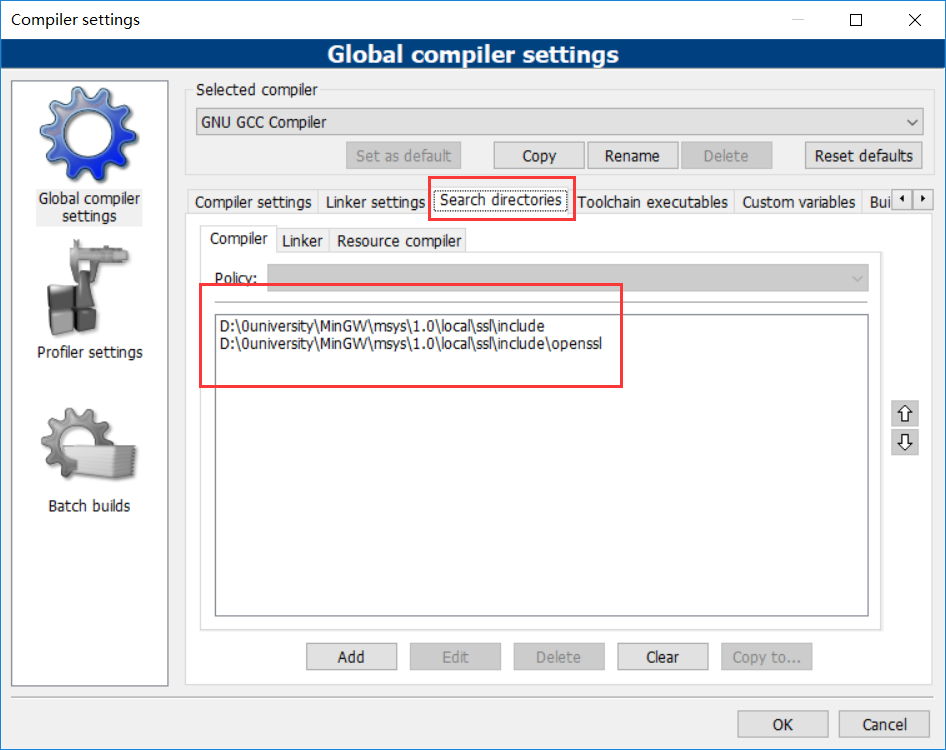


③Codeblocks基于Openssl库（版本：codeblocks-17.12-setup，此版本Codeblocks的编译器需要安装的是Bb平台上的MinGw）

A．选择编译器：

B. 加载Openssl库：

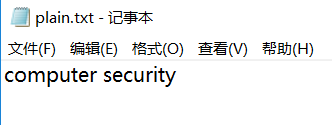




**三、实验内容（主要是实验结果，不必全部照抄实验要求文档中的内容。实验结果截图需按照顺序标号并给出标题，且给出简单文字说明。）**

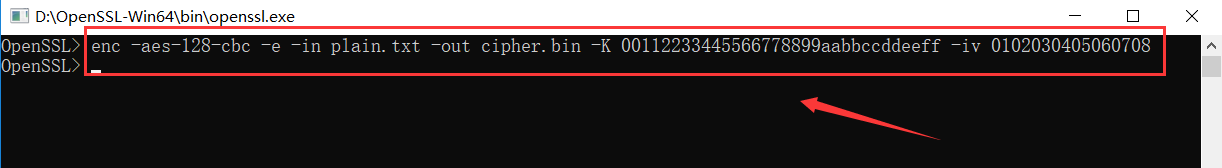
1.使用openssl enc命令来加密／解密一个文件（使用D:\OpenSSL-Win64\bin的openssl.exe执行命令）

Plain.txt为待加密文件，内容为：computer security

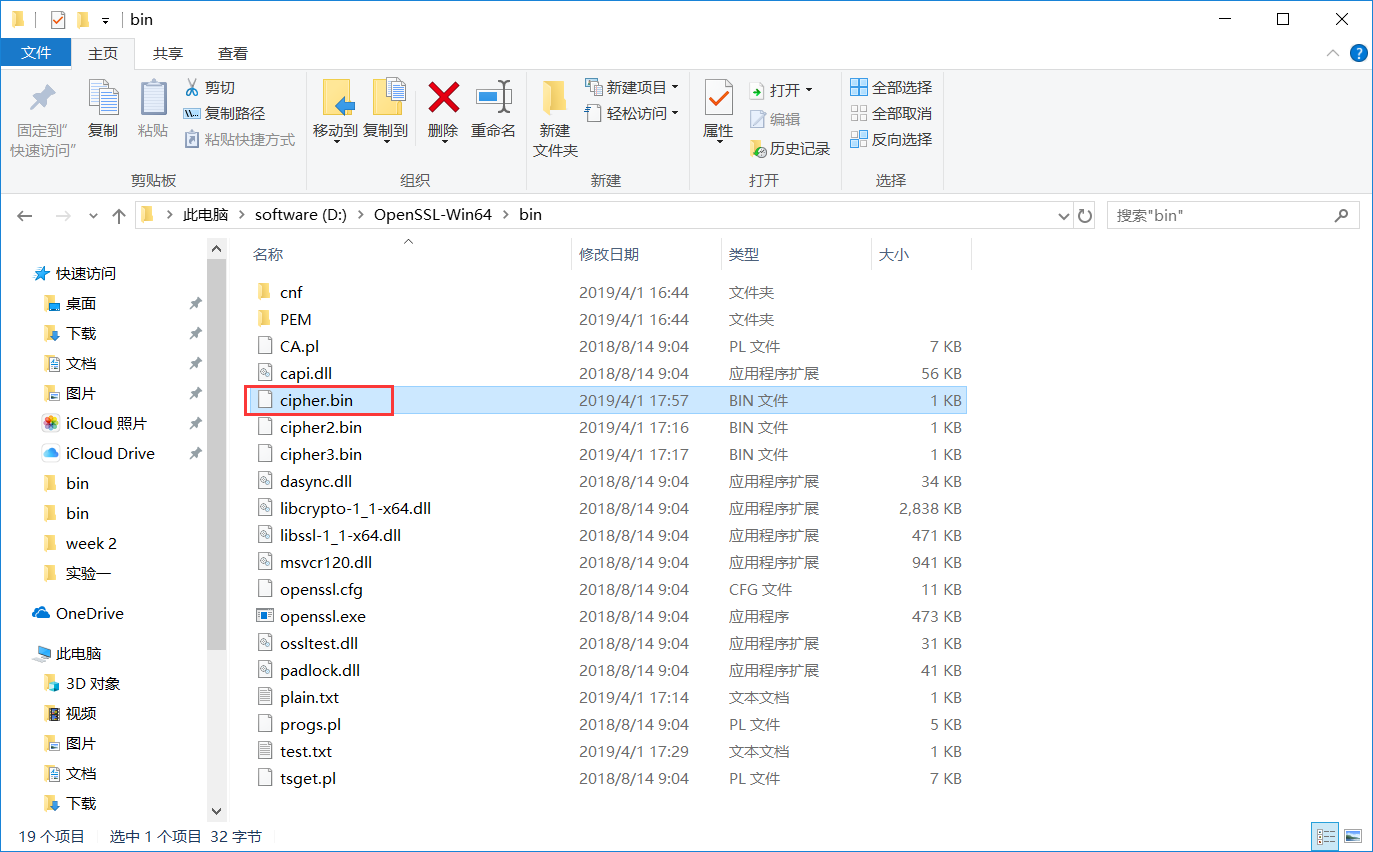


1. 加密类型（Cipher type）：-aes-128-cbc

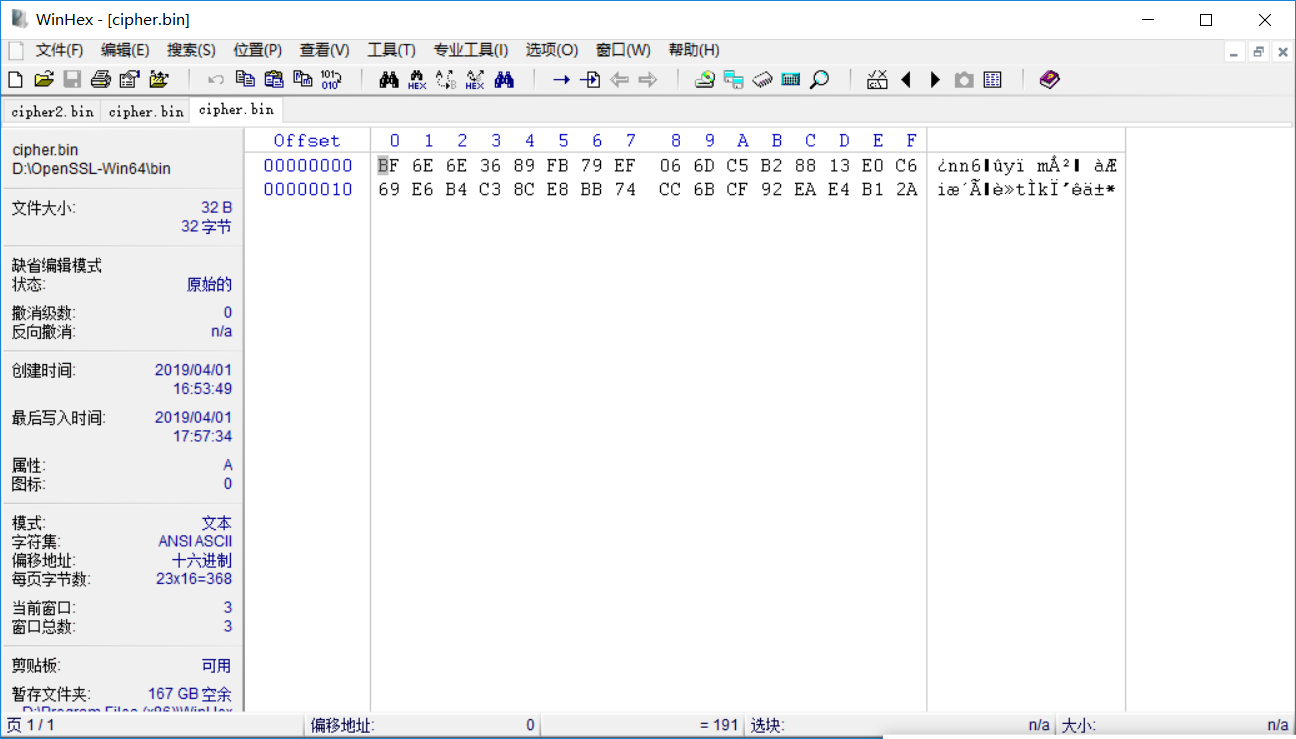
A.加密命令：enc -aes-128-cbc -e -in plain.txt -out cipher.bin -K 00112233445566778899aabbccddeeff -iv 0102030405060708



B.产生文件：cipher.bin

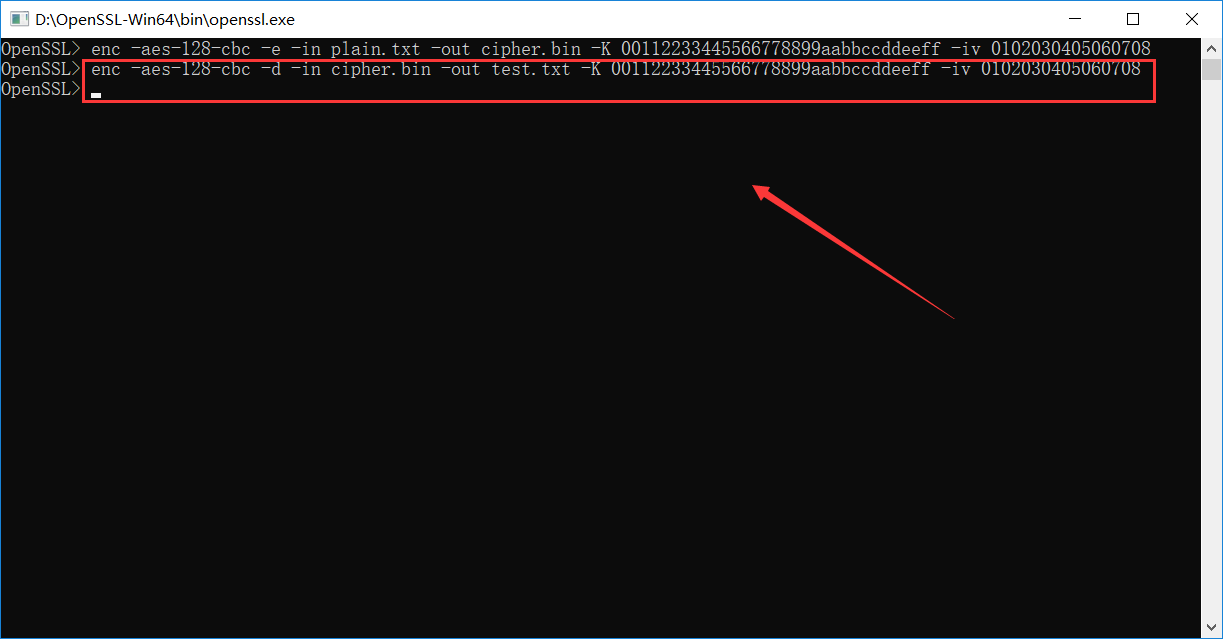


C.将cipher.bin在WinHex中打开：

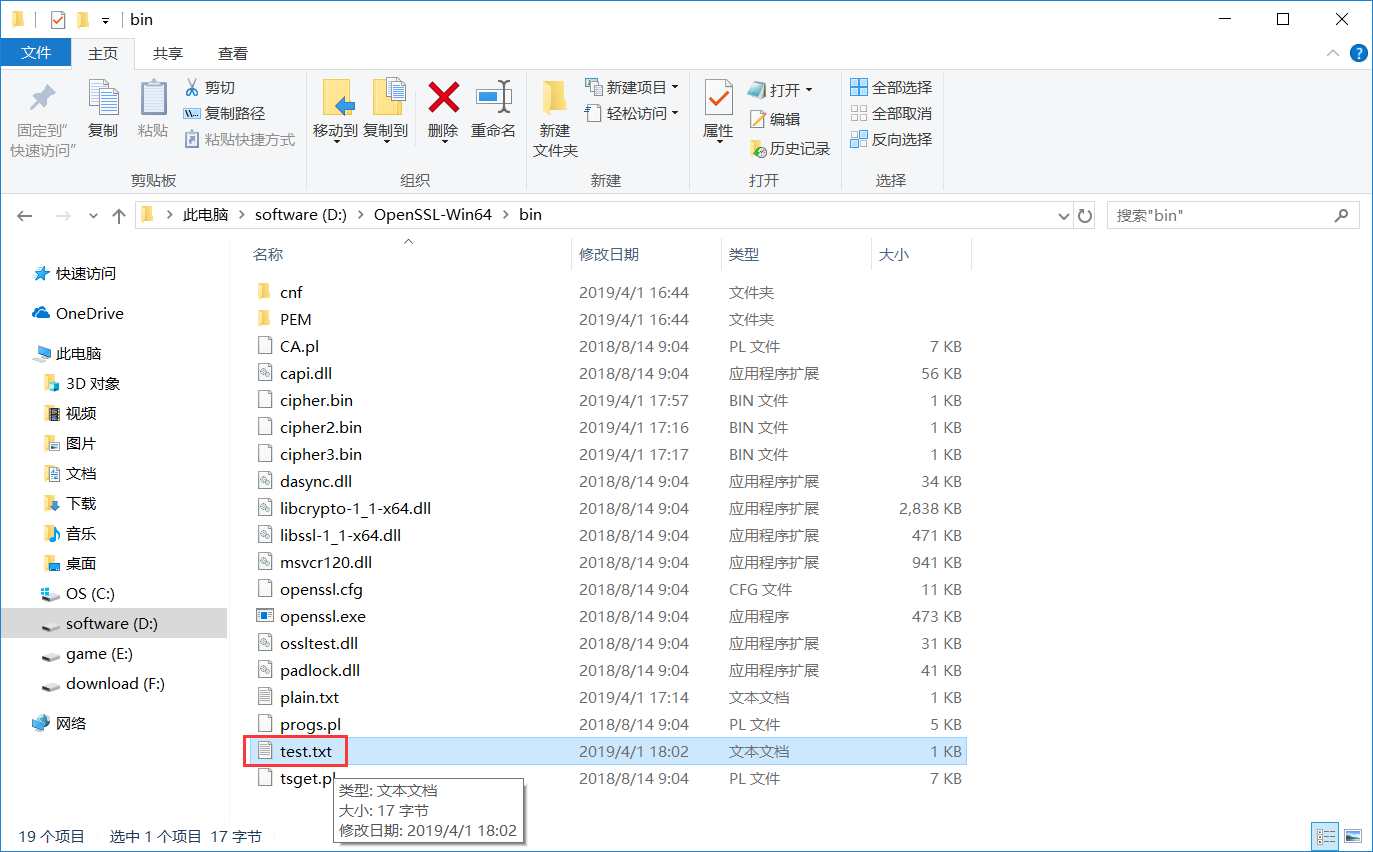


1. 解密类型：-aes-128-cbc

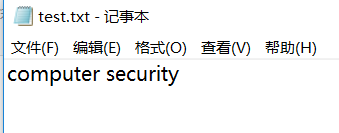
A.解密命令：enc -aes-128-cbc -d -in cipher.bin -out test.txt -K 00112233445566778899aabbccddeeff -iv 0102030405060708



B.产生文件：test.txt



C.解密内容：



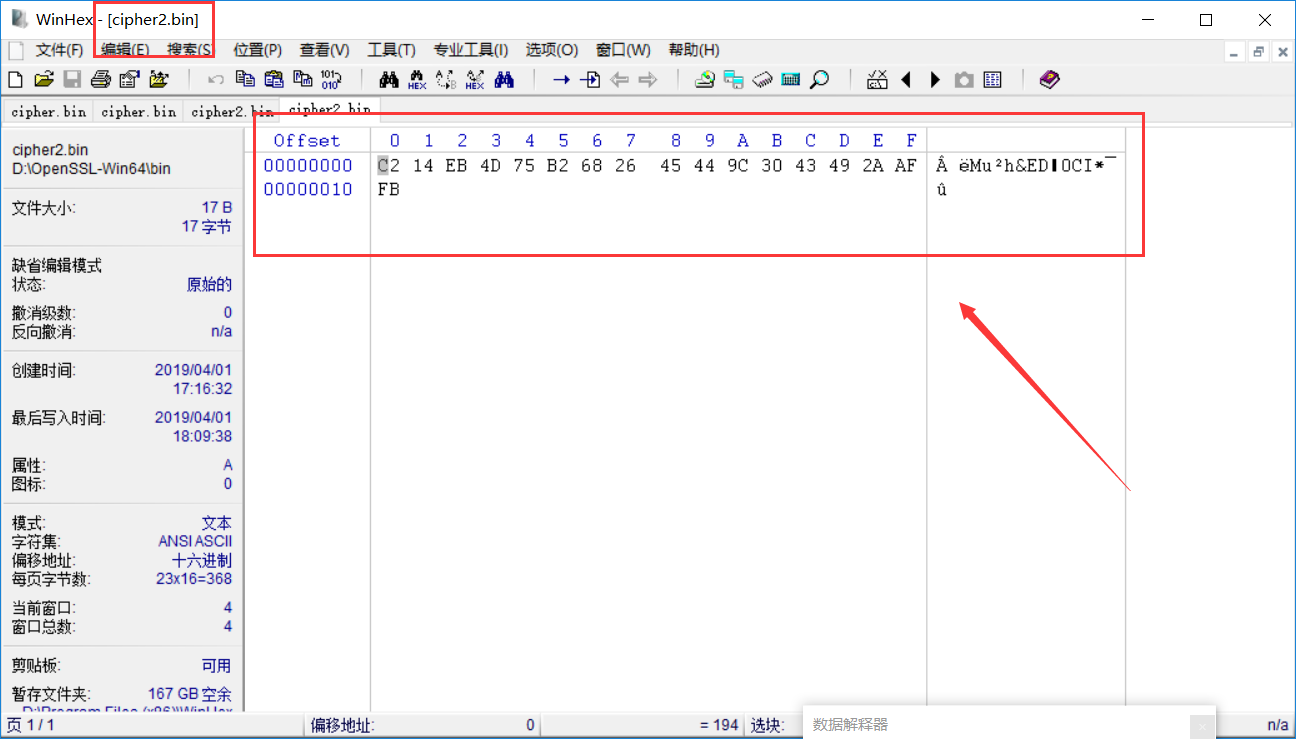
内容仍为：computer security

1. 加密类型（Cipher type）：-aes-128-cfb

A.加密命令：enc -aes-128-cfb -e -in plain.txt -out cipher2.bin -K 00112233445566778899aabbccddeeff -iv 0102030405060708

B.产生文件：cipher2.bin

C.将cipher2.bin在Winhex中打开：

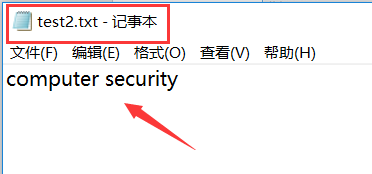


1. 解密类型：-aes-128-cfb

A.解密命令：enc -aes-128-cfb -d -in cipher2.bin -out test2.txt -K 00112233445566778899aabbccddeeff -iv 0102030405060708

B.产生文件：test2.txt

C.解密内容：



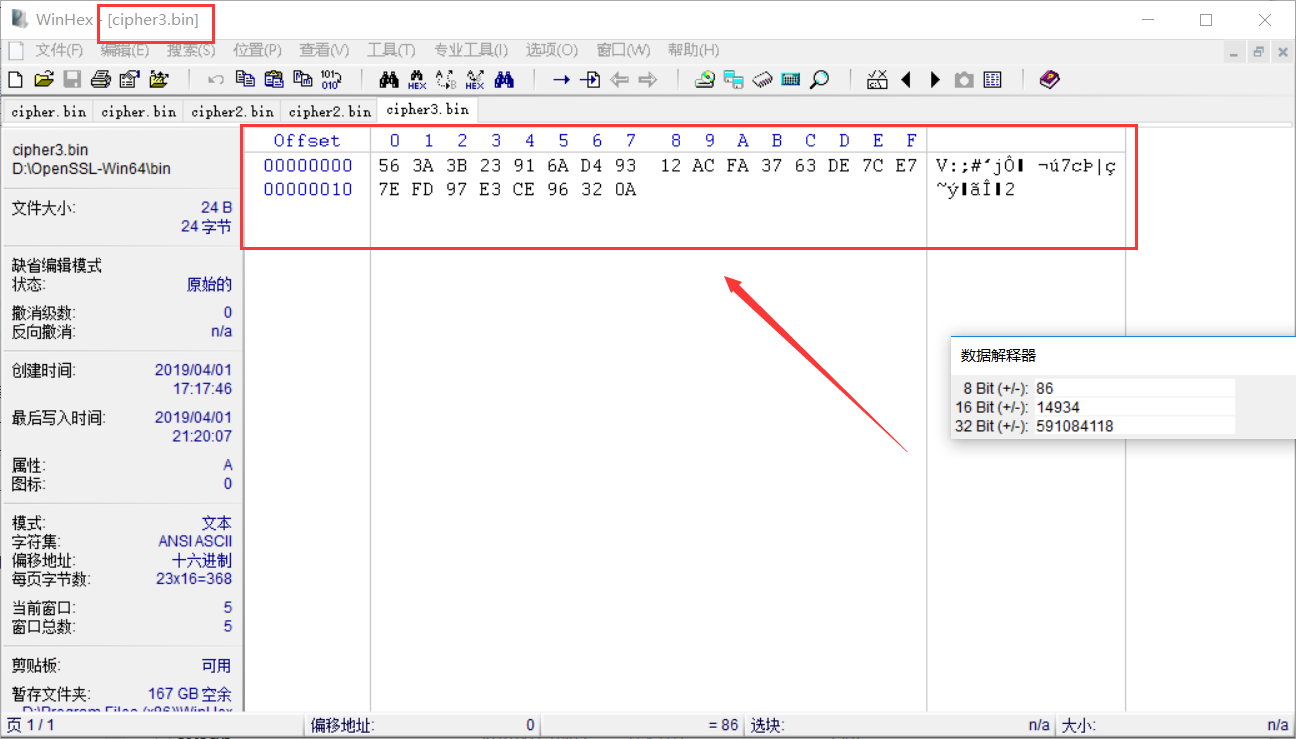
内容仍为：computer security

1. 加密类型（Cipher type）：-bf-cbc

A.加密命令：enc -bf-cbc -e -in plain.txt -out cipher3.bin -K 00112233445566778899aabbccddeeff -iv 0102030405060708

B.产生文件：cipher3.bin

C.将cipher3.bin在WinHex中打开：

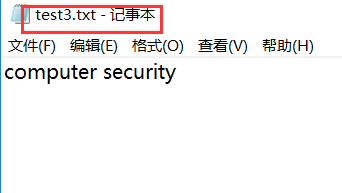


1. 解密类型：-bf-cbc

A.解密命令：enc -bf-cbc -d -in cipher3.bin -out test3.txt -K 00112233445566778899aabbccddeeff -iv 0102030405060708

B.产生文件：test3.txt

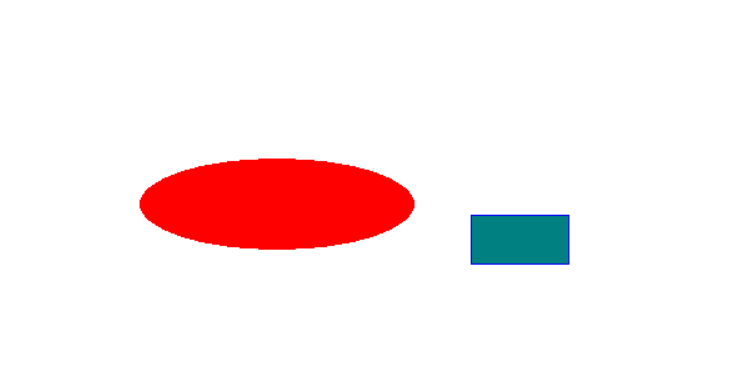
C.解密内容：



内容仍为：computer security

2. 使用openssl enc命令来加密／解密一张图片（使用D:\OpenSSL-Win64\bin的openssl.exe执行命令）

pic\_original.bmp为待加密图片文件：

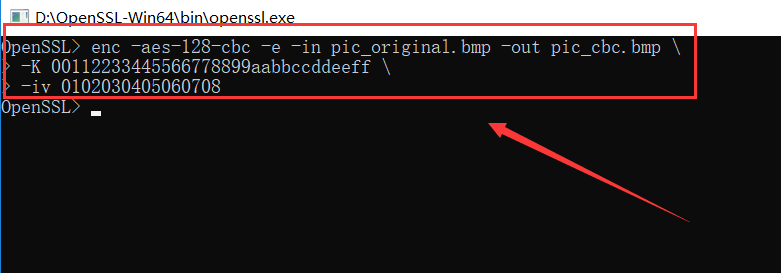


①加密类型（Cipher type）：aes-128-ecb

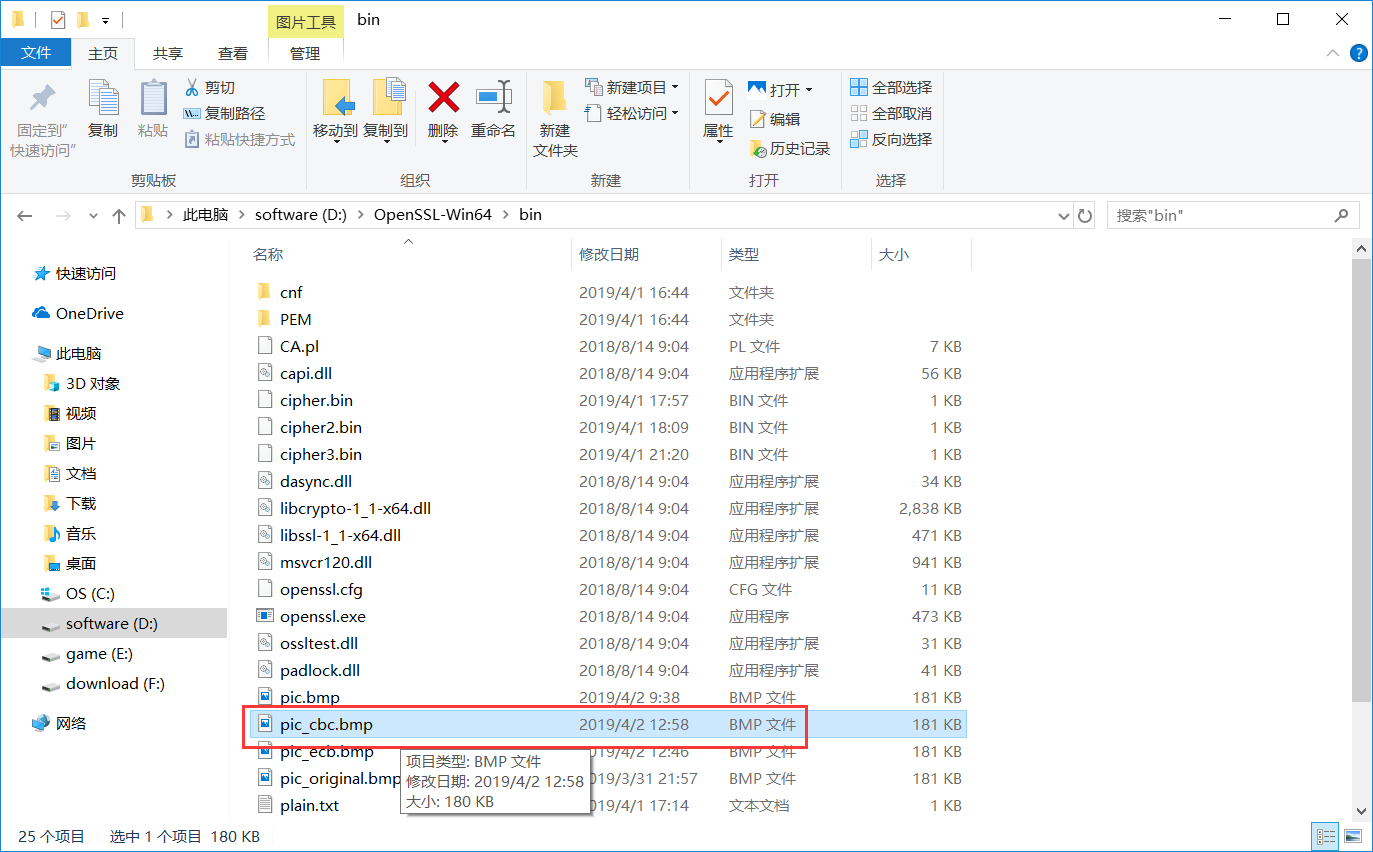
A.加密命令：enc -aes-128-cbc -e -in pic\_original.bmp -out pic\_cbc.bmp \

-K 00112233445566778899aabbccddeeff \

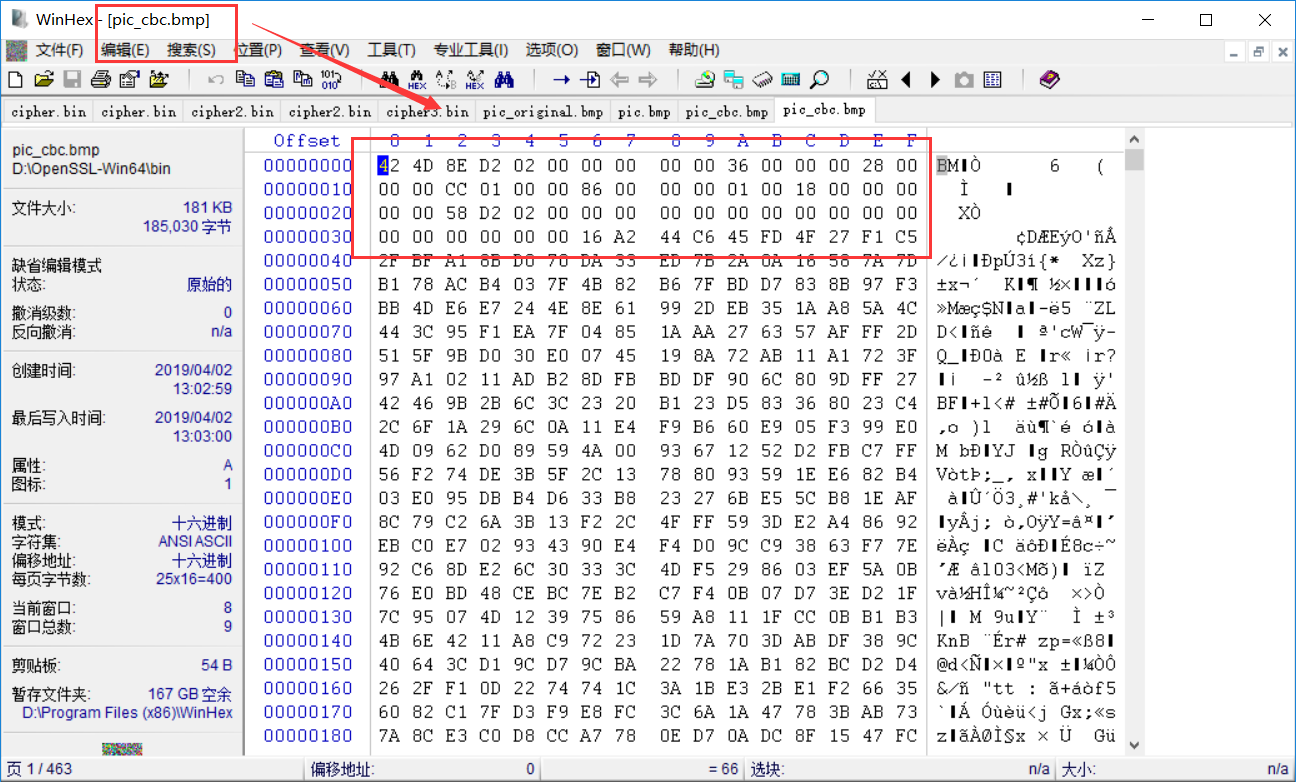
-iv 0102030405060708



B.产生文件：pic\_cbc.bmp



C.使用WinHex修改其前缀为标准的图像文件前缀:



D.加密后的图片文件：



②加密类型（Cipher type）：-aes-128-ecb

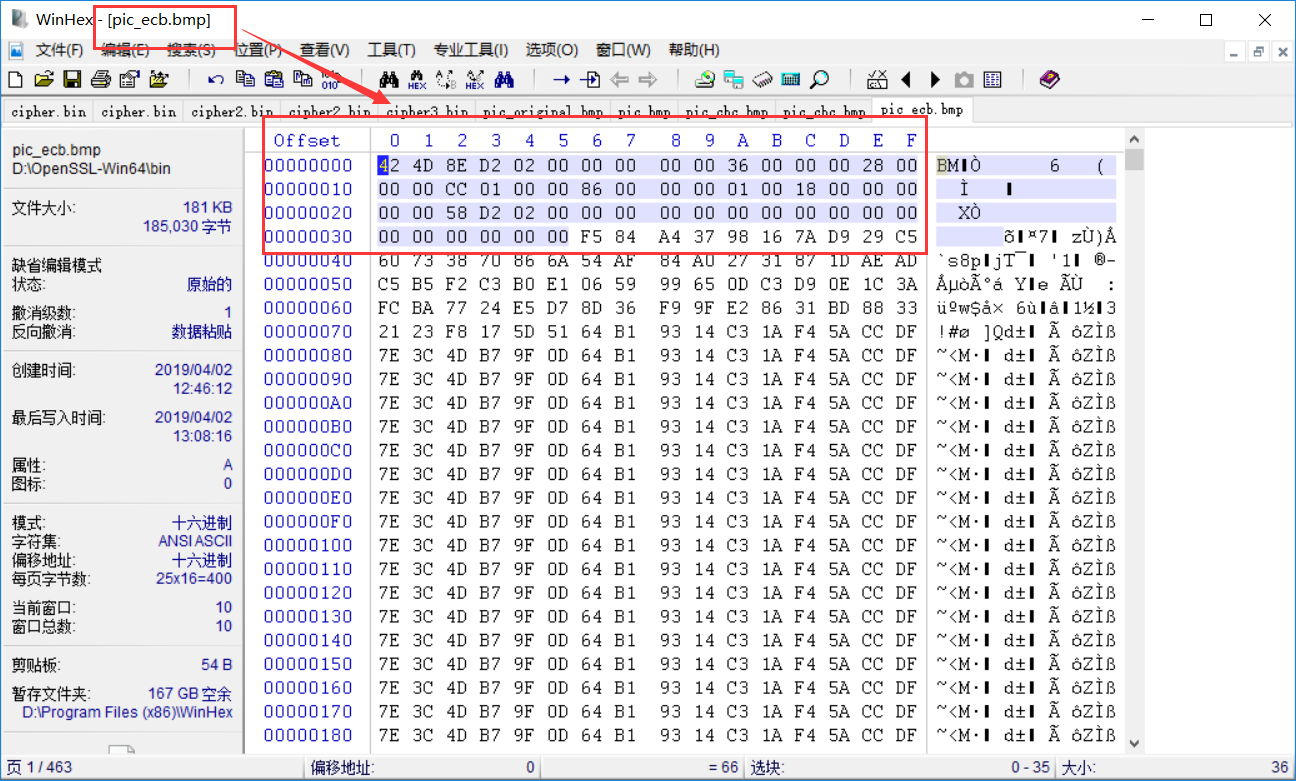
A.加密命令：enc -aes-128-ecb -e -in pic\_original.bmp -out pic\_ecb.bmp \

-K 00112233445566778899aabbccddeeff \

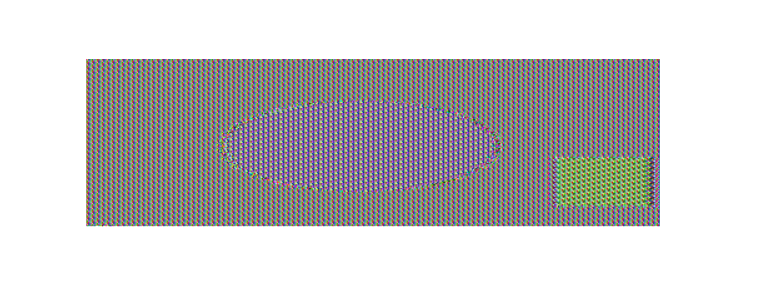
-iv 0102030405060708

B.产生文件：pic\_ecb.bmp

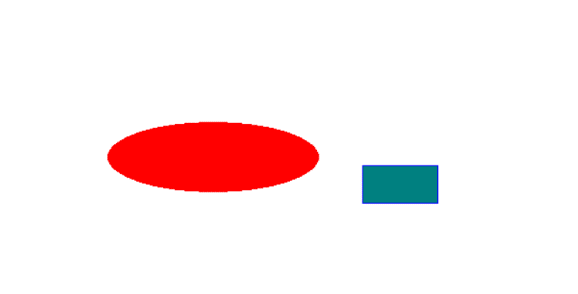
C.使用WinHex修改其前缀为标准的图像文件前缀:



D. 加密后的图片文件：

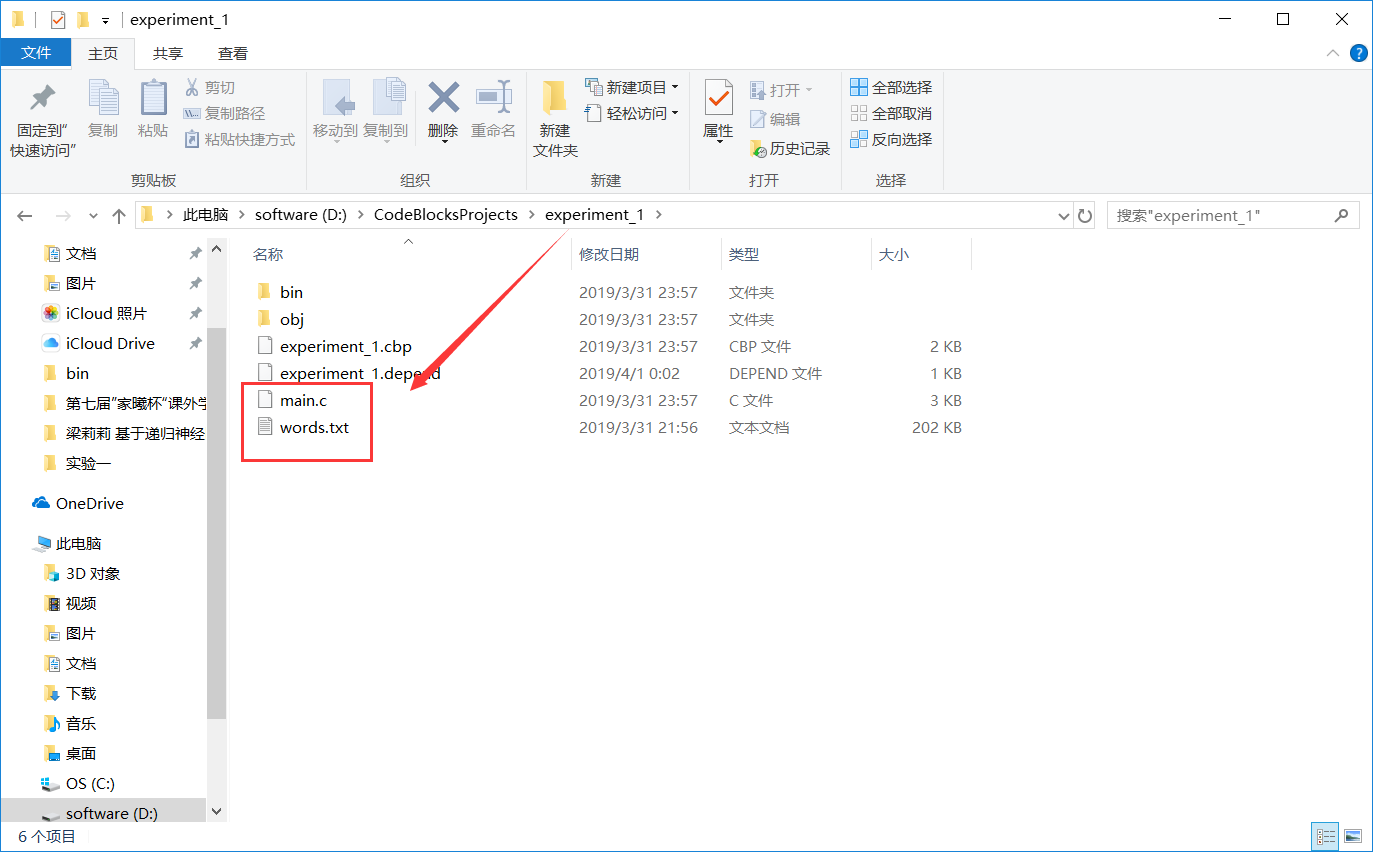


③观察结果(左:cbc 中:ecb 右:原图)

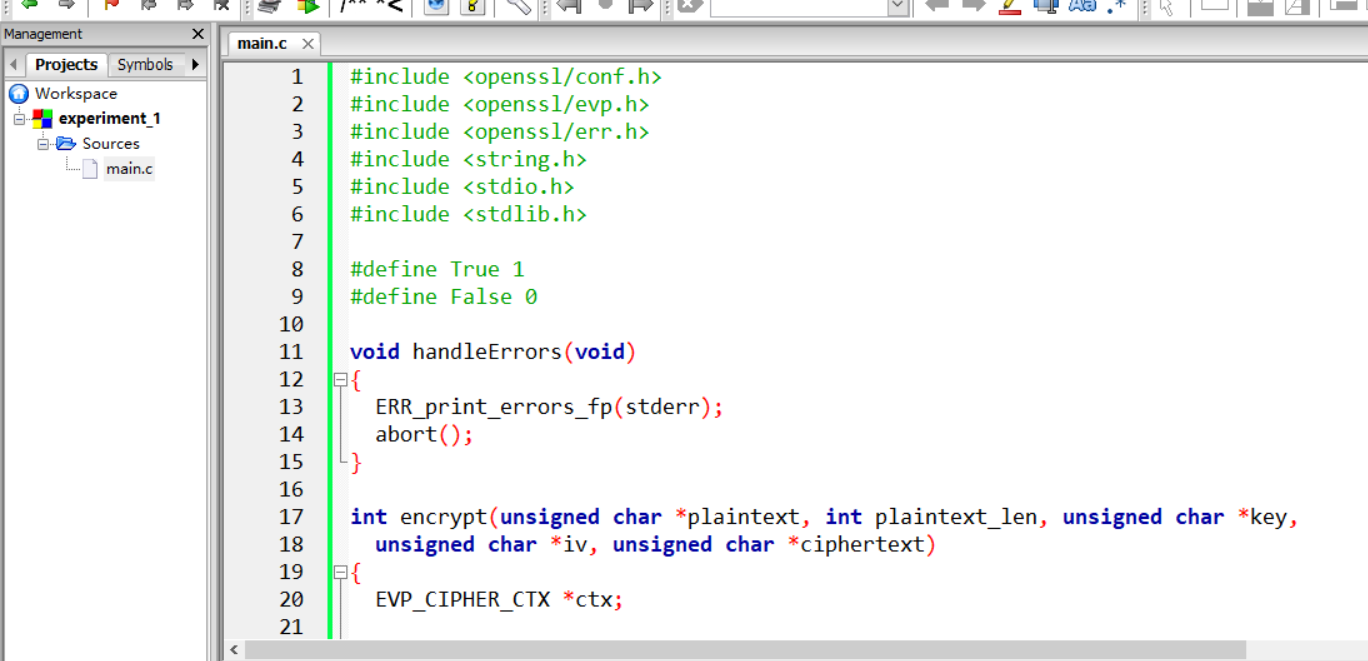


3. 使用 Openssl 加密库进行编程

A.文件目录：



B.运行程序：





C. 给程序中添加注释：

#include <openssl/conf.h>

#include <openssl/evp.h>

#include <openssl/err.h>

#include <string.h>

#include <stdio.h>

#include <stdlib.h>

#define True 1

#define False 0

//抛出错误函数

void handleErrors(void)

{

ERR\_print\_errors\_fp(stderr);

abort();

}

//使用EVP\_Encrypt接口进行加密处理

int encrypt(unsigned char \*plaintext, int plaintext\_len, unsigned char \*key,

unsigned char \*iv, unsigned char \*ciphertext)

{

//开辟EVP\_CIPHER\_CTX

EVP\_CIPHER\_CTX \*ctx;

int len;

int ciphertext\_len;

//如果无法开辟空间，则抛出错误

if(!(ctx = EVP\_CIPHER\_CTX\_new())) handleErrors();

//EVP\_CIPHER\_CTX\_set\_padding(ctx, 0);

//初始化ctx

if(1 != EVP\_EncryptInit\_ex(ctx, EVP\_aes\_128\_cbc(), NULL, key, iv))

handleErrors();

//根据加密的数据大小去处理数据

if(1 != EVP\_EncryptUpdate(ctx, ciphertext, &len, plaintext, plaintext\_len))

handleErrors();

ciphertext\_len = len;

//处理没加密完的数据

if(1 != EVP\_EncryptFinal\_ex(ctx, ciphertext + len, &len)) handleErrors();

ciphertext\_len += len;

//释放空间

EVP\_CIPHER\_CTX\_free(ctx);

return ciphertext\_len;

}

int append(char\* buffer){

int length = (int)strlen(buffer);

if (length > 16)

return False;

memset(buffer+strlen(buffer),' ', 16-length);

buffer[16] = '\0';

return True;

}

//调用加密函数测试

int main(int argc, char const \*argv[])

{

char buffer[50];

int i = 0;

char iv[17];

memset(iv, 0, 17);

//明文

unsigned char \*plaintext = "This is a top secret.";

unsigned char ciphertext[100];

//密文

unsigned char \*cryptotext="8d20e5056a8d24d0462ce74e4904c1b513e10d1df4a2ef2ad4540fae1ca0aaf9";

ERR\_load\_crypto\_strings();

OpenSSL\_add\_all\_algorithms();

OPENSSL\_config(NULL);

int ciphertext\_len;

int k=0;

//读入英文字典文件

FILE \*fp = fopen("words.txt", "r");

//开始读文件

while (fscanf(fp, "%s\n", buffer) != EOF){

k++;

if (!append(buffer))

continue;

//开始加密

ciphertext\_len = encrypt(plaintext, strlen(plaintext), buffer, iv, ciphertext);

unsigned char cryptohex[50];

for (i = 0; i < ciphertext\_len; i++)

{

sprintf(cryptohex+i\*2,"%02x", ciphertext[i]);

}

cryptohex[ciphertext\_len\*2] = '\0';

//找到密钥

if (0 == strcmp(cryptohex, cryptotext)){

printf("The key is: %s, cryptohex is %s\n", buffer, cryptohex);

break;

}

}

//printf("k is %d\n", k);

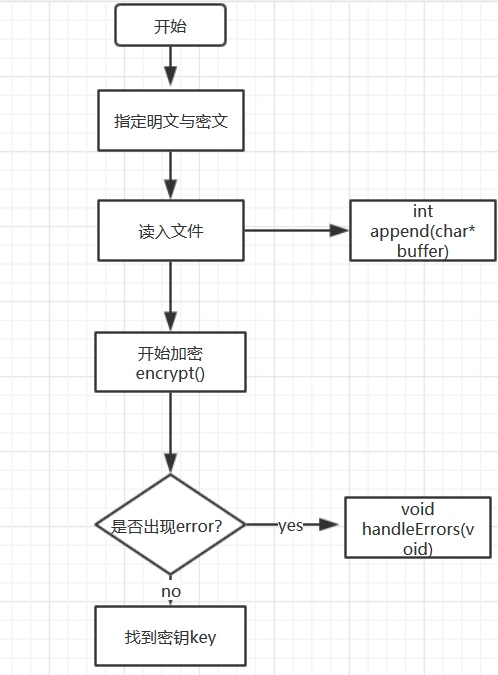
EVP\_cleanup();

ERR\_free\_strings();

return 0;

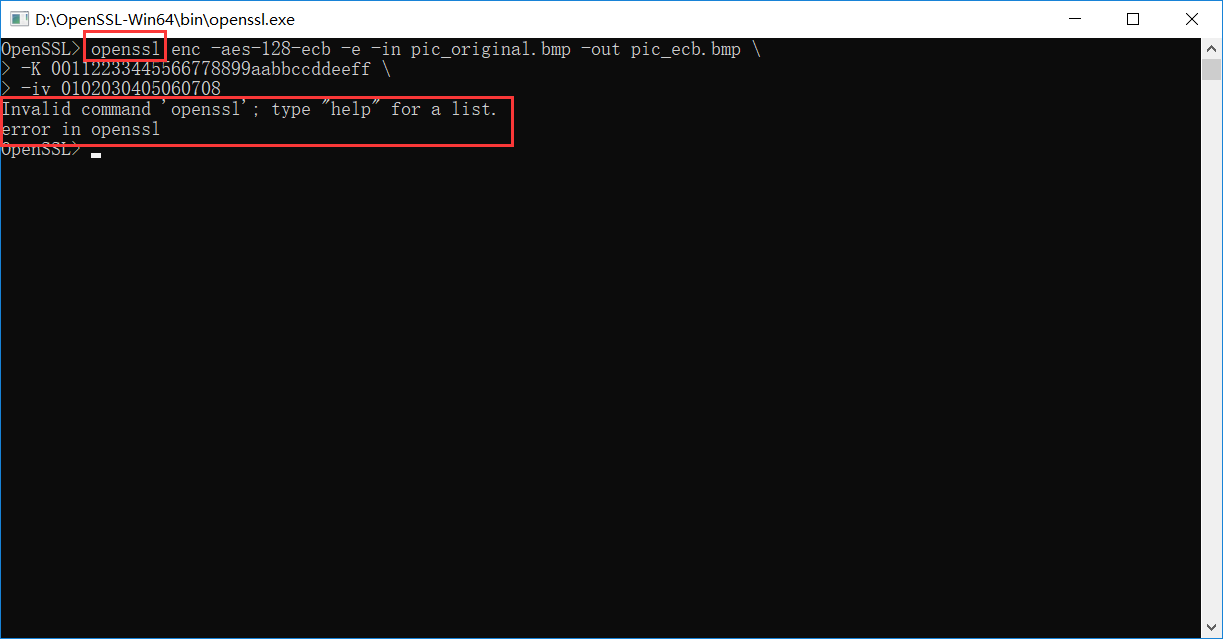
}

D. 程序的流程图：

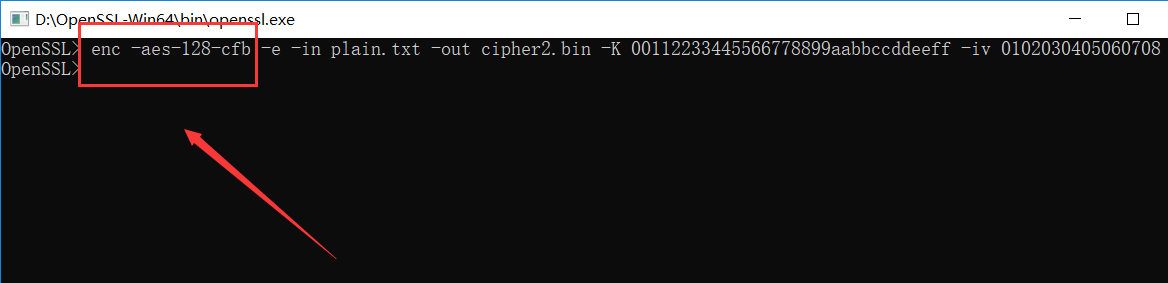


**四、实验总结（此部分必须有）**

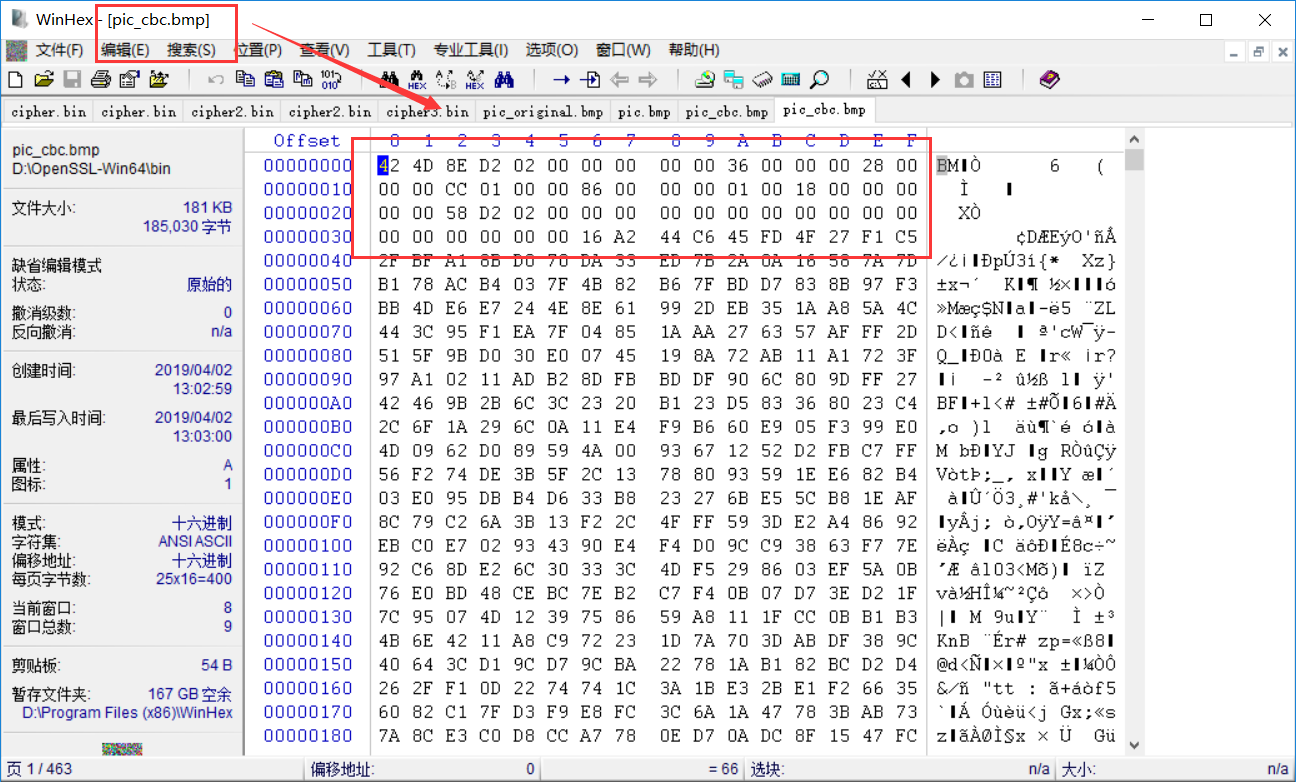
1. 遇到的问题
2. 在Windows环境下不知哪个是运行openssl的命令行工具；
3. 在openssl.exe执行openssl文件时，一直显示“Invalid command 'openssl'”，如图所示：



1. 在加密bmp格式图片文件时，加密后的文件无法打开。
2. 相应的解决方法：
   1. **针对不知用哪个命令行工具运行openssl命令**：曾经以为是用D:\0university\MinGW\msys\1.0路径下的msys.bat，但是执行的效果不尽人意，在请教老师与“百度”之后，发现是用openssl.exe文件。
   2. **针对command一直显示报错“Invalid command 'openssl'”**：发现去掉前面的“openssl”，直接进入enc命令就能解决。



* 1. **针对加密bmp格式图片文件时，加密后的文件无法打开：**用WinHex打开加密后的bmp文件，将其前缀改为标准的图像文件前缀（bmp 头长度为 54 字节）。



1. 收获：看到实验任务里关于环境的要求时，第一反应是想挑战一下自己，不用实验楼的环境，而是自己搭环境做实验。此次实验用的是Win10的系统环境，实验过程中第一次遇到的问题在于使用Bb平台上的MinGW文件夹中的openssl.exe时，显示“WARNING: can't open config file: /usr/local/ssl/openssl.cnf”，当时一直认为是与本机环境不兼容的问题，所以直接武断地去重新下载安装了新的Openssl库去跑命令。事后请教老师，才得知虽然命令行报了warning，但是命令还是可以正常跑的。这也提醒我自己，遇到报错/警告时，应该首先检查错误信息，会不会影响到我的任务执行效果，才决定是否需要重新搭建环境。此外，本次实验也极大地提高了我各方面的综合能力，使我更加了解加密/解密的原理。对于使用WinHex查看文件的十六进制格式，以及亲自动手修改bmp图片文件的前缀，也使我了解了更多关于此方面的知识，让我收获颇丰！