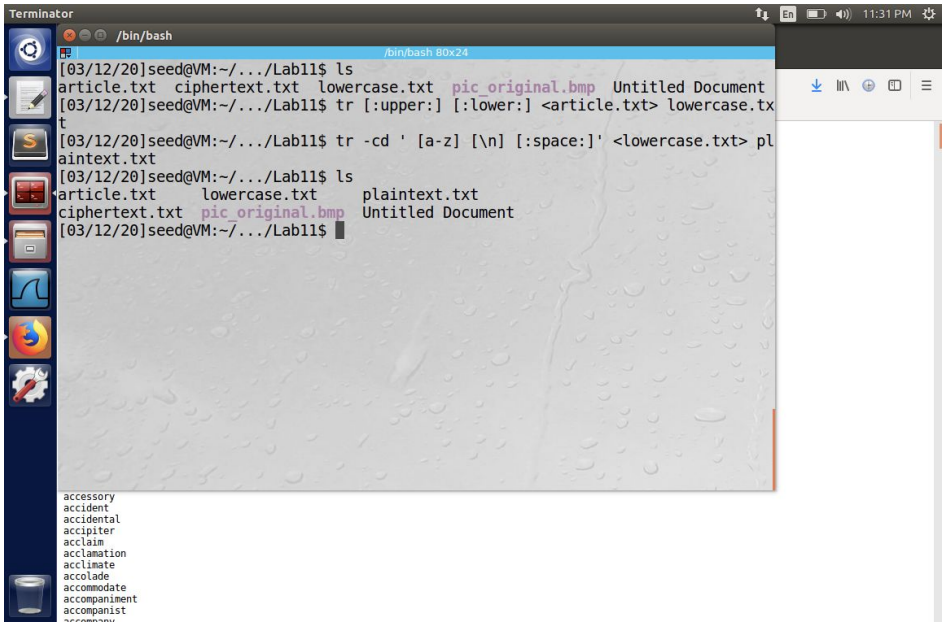
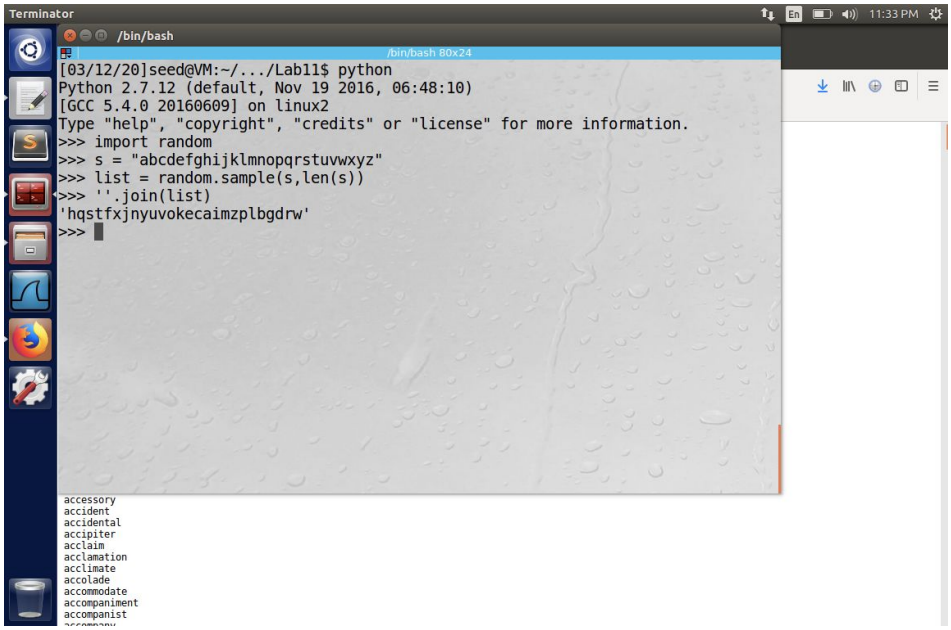
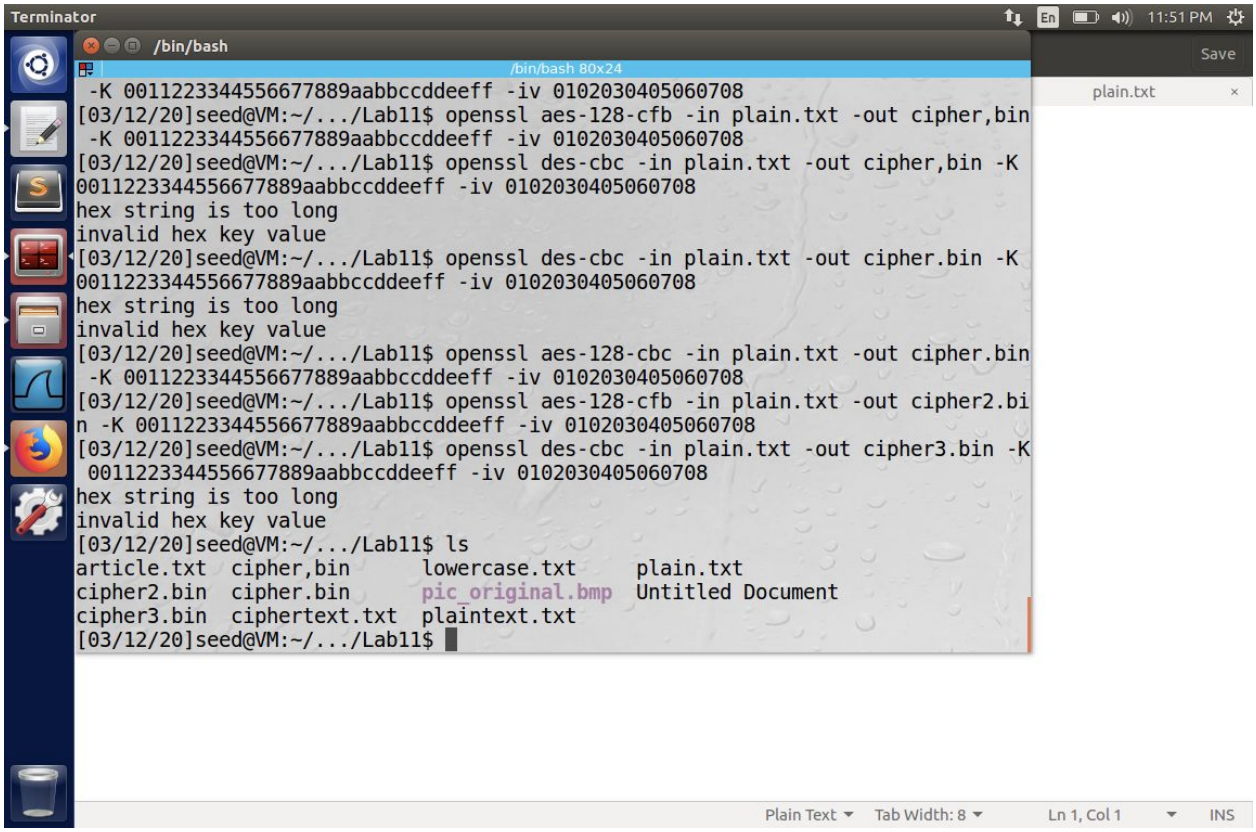
Lab 11
CSP 544
Karsen Diepholz

Task 1





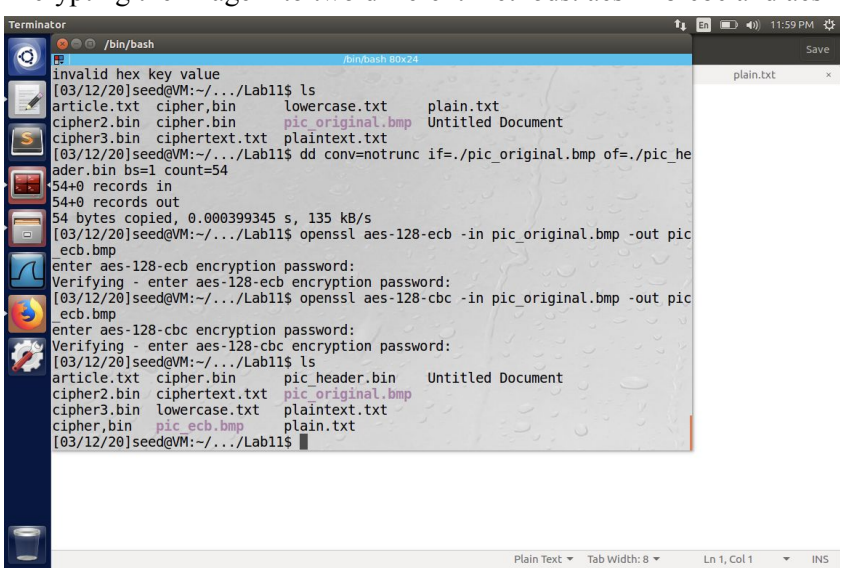
Task 2



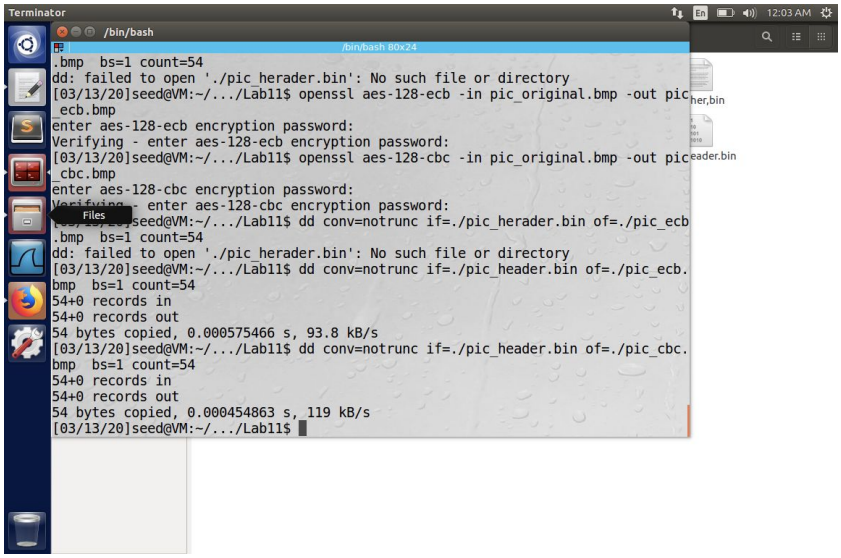
We encrypt the file using 3 different methods: aes-128-cbc, aes-128-cfb, des-cbc.

Task 3:

Encrypting the image into two different methods: aes-128-cbc and aes-128-cfb



Showing the two different images:


Terminator
/bin/bash
/bin/bash 80x24
.bmp bs=1 count=54
dd: failed to open './pic_herader.bin': No such file or directory
[03/13/20]seed@VM:~/.../Lab11$ openssl aes-128-ecb -in pic_original.bmp -out pic
_ecb.bmp
enter aes-128-ecb encryption password:
Verifying - enter aes-128-ecb encryption password:
[03/13/20]seed@VM:~/.../Lab11$ openssl aes-128-cbc -in pic_original.bmp -out pic
_cbc.bmp
enter aes-128-cbc encryption password:
Verifying - enter aes-128-cbc encryption password:
[03/13/20]seed@VM:~/.../Lab11$ dd conv=notrunc if=./pic_herader.bin of=./pic_ecb
.bmp bs=1 count=54
dd: failed to open './pic_herader.bin': No such file or directory
[03/13/20]seed@VM:~/.../Lab11$ dd conv=notrunc if=./pic_header.bin of=./pic_ecb.
bmp bs=1 count=54
54+0 records in
54+0 records out
54 bytes copied, 0.000575466 s, 93.8 kB/s
[03/13/20]seed@VM:~/.../Lab11$ dd conv=notrunc if=./pic_header.bin of=./pic_cbc.
bmp bs=1 count=54
54+0 records in
54+0 records out
54 bytes copied, 0.000454863 s, 119 kB/s
[03/13/20]seed@VM:~/.../Lab11$
Files

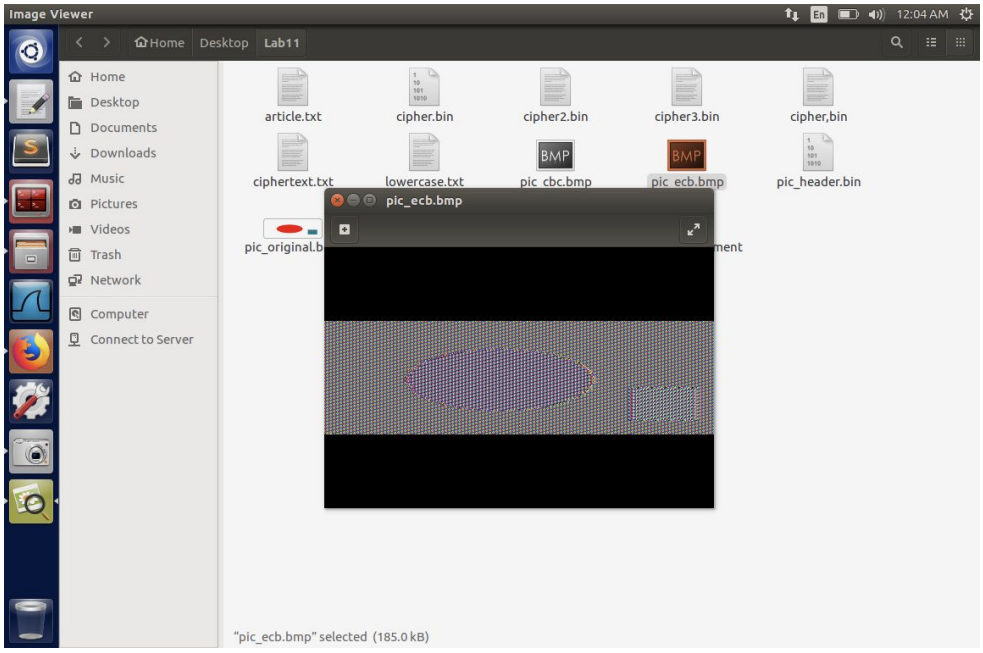
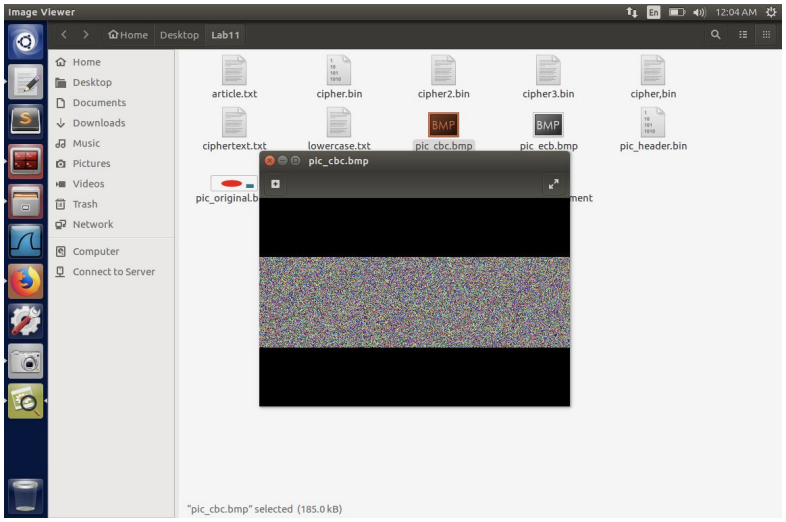
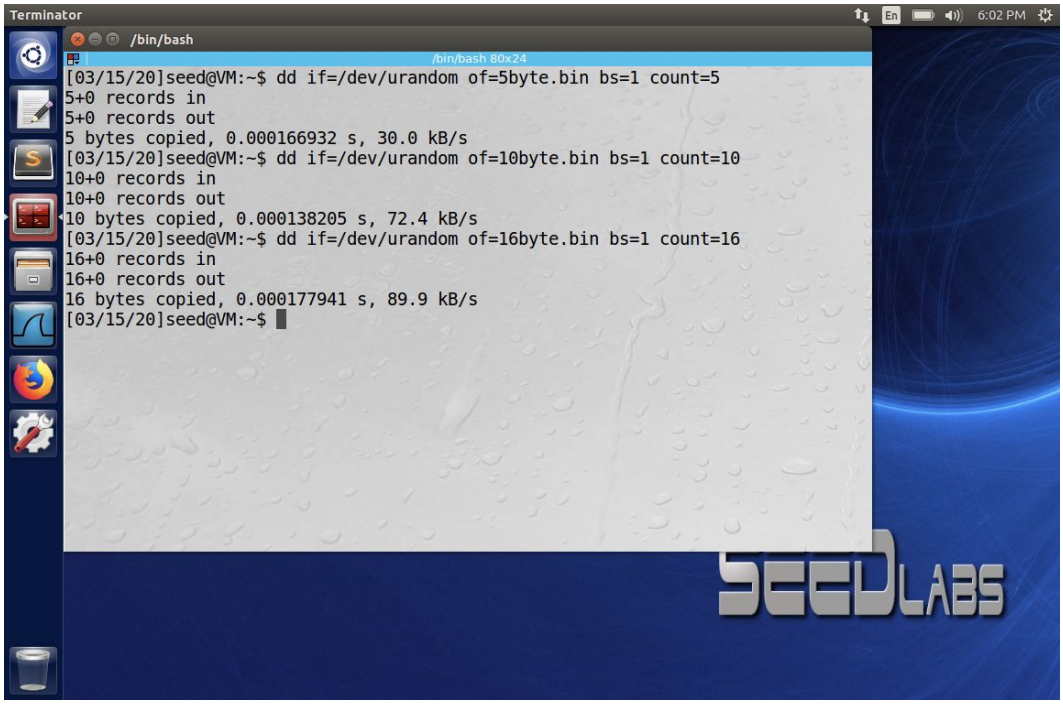


Image Viewer
Home
Desktop
Lab11
Home
Desktop
Documents
Downloads
Music
Pictures
Videos
Trash
Network
Computer
Connect to Server
article.txt
cipher.bin
cipher2.bin
cipher3.bin
cipher,bin
ciphertext.txt
lowercase.txt
pic_cbc.bmp
pic_ecb.bmp
pic_header.bin
pic_ecb.bmp
"pic_ecb.bmp" selected (185.0 kB)

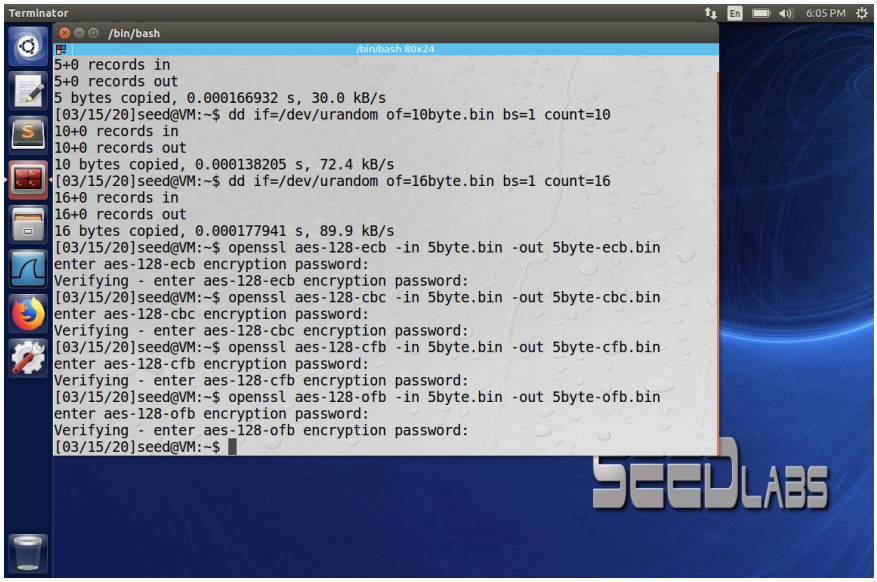


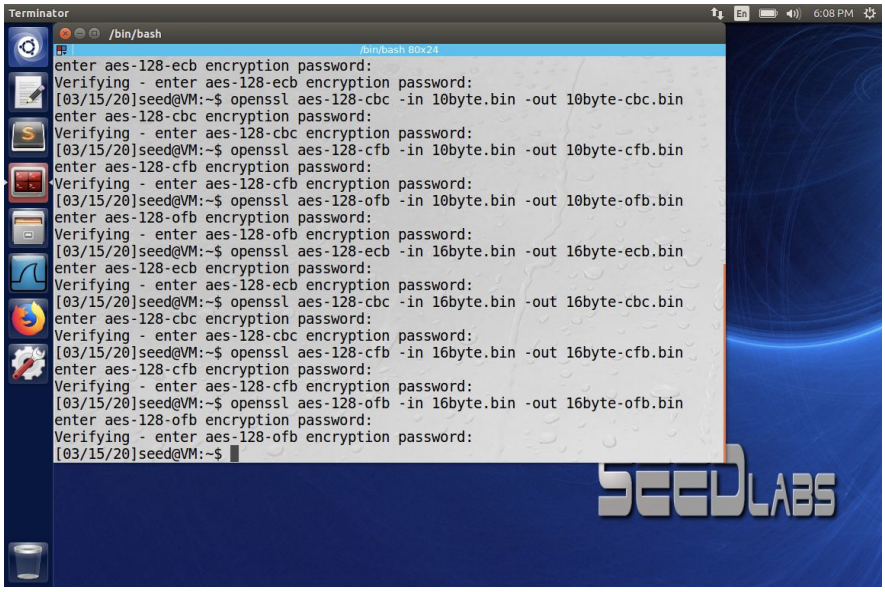
Task 4

Making 3 files…

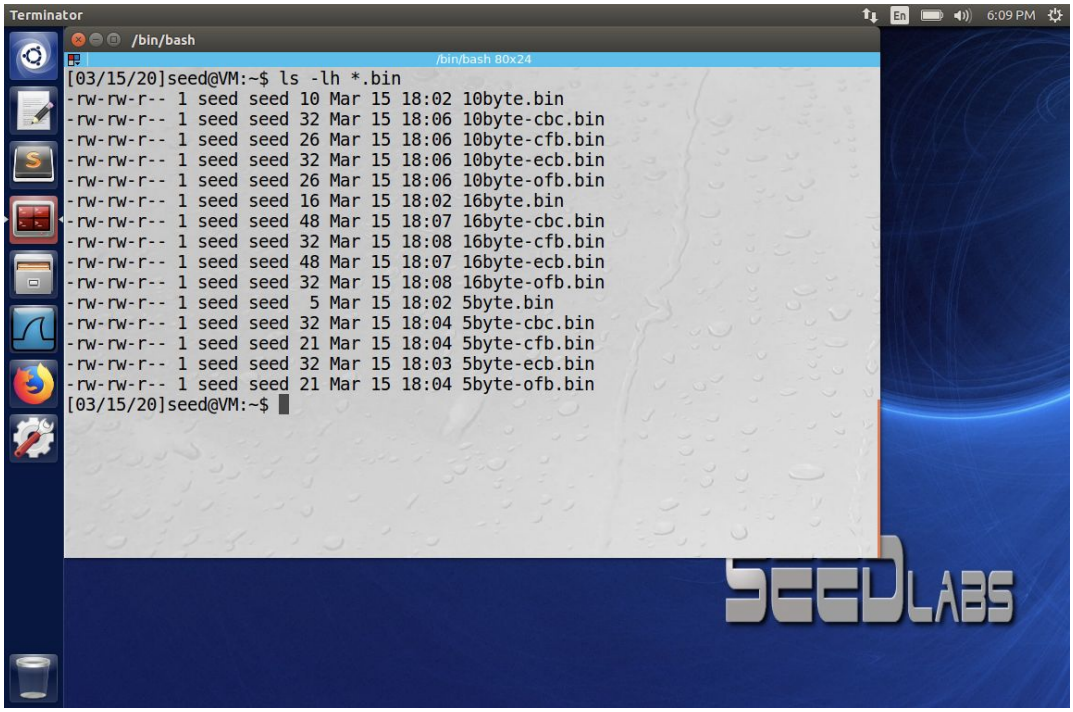


Encrypting…


Terminator
6:05 PM
/bin/bash
/bin/bash 80x24
5+0 records in
5+0 records out
5 bytes copied, 0.000166932 s, 30.0 kB/s
[03/15/20]seed@VM:~$ dd if=/dev/urandom of=10byte.bin bs=1 count=10
10+0 records in
10+0 records out
10 bytes copied, 0.000138205 s, 72.4 kB/s
[03/15/20]seed@VM:~$ dd if=/dev/urandom of=16byte.bin bs=1 count=16
16+0 records in
16+0 records out
16 bytes copied, 0.000177941 s, 89.9 kB/s
[03/15/20]seed@VM:~$ openssl aes-128-ecb -in 5byte.bin -out 5byte-ecb.bin
enter aes-128-ecb encryption password:
Verifying - enter aes-128-ecb encryption password:
[03/15/20]seed@VM:~$ openssl aes-128-cbc -in 5byte.bin -out 5byte-cbc.bin
enter aes-128-cbc encryption password:
Verifying - enter aes-128-cbc encryption password:
[03/15/20]seed@VM:~$ openssl aes-128-cfb -in 5byte.bin -out 5byte-cfb.bin
enter aes-128-cfb encryption password:
Verifying - enter aes-128-cfb encryption password:
[03/15/20]seed@VM:~$ openssl aes-128-ofb -in 5byte.bin -out 5byte-ofb.bin
enter aes-128-ofb encryption password:
Verifying - enter aes-128-ofb encryption password:
[03/15/20]seed@VM:~$
SEEDLABS


Terminator
6:08 PM
/bin/bash
/bin/bash 80x24
enter aes-128-ecb encryption password:
Verifying - enter aes-128-ecb encryption password:
[03/15/20]seed@VM:~$ openssl aes-128-cbc -in 10byte.bin -out 10byte-cbc.bin
enter aes-128-cbc encryption password:
Verifying - enter aes-128-cbc encryption password:
[03/15/20]seed@VM:~$ openssl aes-128-cfb -in 10byte.bin -out 10byte-cfb.bin
enter aes-128-cfb encryption password:
Verifying - enter aes-128-cfb encryption password:
[03/15/20]seed@VM:~$ openssl aes-128-ofb -in 10byte.bin -out 10byte-ofb.bin
enter aes-128-ofb encryption password:
Verifying - enter aes-128-ofb encryption password:
[03/15/20]seed@VM:~$ openssl aes-128-ecb -in 16byte.bin -out 16byte-ecb.bin
enter aes-128-ecb encryption password:
Verifying - enter aes-128-ecb encryption password:
[03/15/20]seed@VM:~$ openssl aes-128-cbc -in 16byte.bin -out 16byte-cbc.bin
enter aes-128-cbc encryption password:
Verifying - enter aes-128-cbc encryption password:
[03/15/20]seed@VM:~$ openssl aes-128-cfb -in 16byte.bin -out 16byte-cfb.bin
enter aes-128-cfb encryption password:
Verifying - enter aes-128-cfb encryption password:
[03/15/20]seed@VM:~$ openssl aes-128-ofb -in 16byte.bin -out 16byte-ofb.bin
enter aes-128-ofb encryption password:
Verifying - enter aes-128-ofb encryption password:
[03/15/20]seed@VM:~$
SEEDLABS

Results…



CFB and OFB pad 16 bytes to the end, CBC and ECB pad until a multiple of 8 is reached.

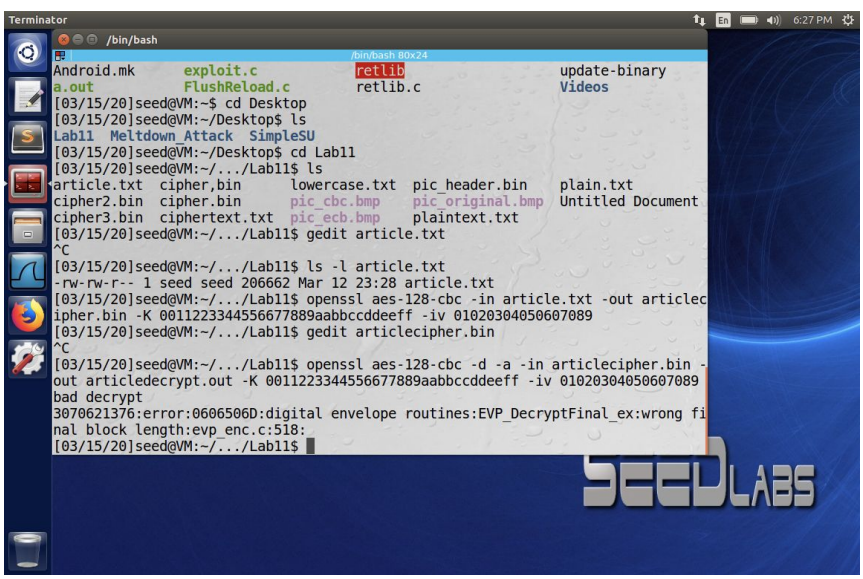
Task 5:

ECB: All but 1 corrupted block

CBC: All but 2 corrupted block

CFB: All but 2 blocks

OFB: All but 1 block



Cannot be decrypted because we were the ones who corrupted it!

Task 6

.1

Each IV needs to be different because if they are the same, the attacker can use any message to figure out what the plaintext is for every message that is received, because every word is the same (eg “word” is 6b32 for every instance of it)

.2

Task 7

*fileKey.c (~/Desktop/Lab11) - gedit

```c
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <openssl/env.h>

void pad(char *s, int length);
int print_result(unsigned char *buf, char *s, int len, FILE *outFile, char *match);
int strcicmp(char const *a, char const *b);

int main(){
        unsigned char match[]= "MATCH";
        unsigned char noMatch[] = "NO MATCH";
        int i;
        char words[16], t;
        FILE *key, *outfile;
        unsigned char outbuf[1024 + EVP_MAX_BLOCK_LENGTH];
        unsigned char iv[16] = [0];
        int outlen, tmplen;
        int num;

        EVP_CIPHER_CTX ctx;
        EVP_CIPHER_ETX_init(&ctx);
        char inText[] = "This is a top secret.";
        char cipherTextInit[] = "764aa26b55a4da654df6b19e4bce00f4ed05e09346fb0e762583cb7da2ac93a2";
        key = fopen("words.txt", "r") == -1){
                perror("Error deleting");
        }
        outfile = fopen("matching.txt", "a+");
        if (key <0 || outfile <0){
                perror ("Error opening");
                exit(1);
        }
        char pbuffer[1024];

        while (fgets(words, 16, key)
        {
                i = strlen(words);
                words[i-1] = '\0';
                i = strlen(words);
```

*fileKey.c (~/Desktop/Lab11) - gedit

```c
                perror ("Error opening");
                exit(1);
        }
        char pbuffer[1024];

        while (fgets(words, 16, key)
        {
                i = strlen(words);
                words[i-1] = '\0';
                i = strlen(words);
        if(i<16){
                pad(words, (16));
        }
        EVP_EncryptInit_ex(&ctx, EVP_aes_128_cbc(), NULL, words, iv);
        if(!EVP_EncryptUpdate(&ctx, outbuf, &outlen, inText, strlen(inText)))
        {
                EVP_CIPHER_CTX_cleanup(&ctx);
                return 0;
        }
        if(!EVP_EncryptFinal_ex(&ctx, outbuf + outlen, &tmplen))
        {
        EVP_CIPHER_CTX_cleanup(&ctx);
        return 0;
        }
        outlen += tmplen;

        int i;
        char* buf_str = (*char) malloc (2*outlen +1);
        char* buf_ptr = buf_str;
        for (i=0; i<outlen,i++)
        {
                buf_ptr += sprinf(buf_ptr, "%02X", outbuf[i]);
        }
        *(buf_ptr +1) = '\0';
        if(strcicmp(cicpherTextInit, buf_str) ==0)
                print_result(outbuf, words, outlen, outFile, match);
        else
        print_result(outbuf, words, outlen, outFile, noMatch);
        }
```

*fileKey.c (~/Desktop/Lab11) - gedit

```c
}

int print_result(unsigned char *buf, char *s, int len, FILE *outFile, char *match)
{
        int i, n, j, k;
        char x = '\n';
        char space = ' ';
        for(j=0; j <strlen(s); j++){
        fprintf(outFile, "%c", s[j]);
        }
        fprintf(outFile, "%c", space);
        for(i=0;i<len;i++){
        fprintf(outFile, "%02x", buf[i);
        }
        fprintf(outFile, "%c", space);
        for(k=0; k<strlen(match);k++){
        fprintf(outFile, "%c",match[k]);
        }
        fprintf(outFile, "%c",x);
        return(0);
}

void pad(char *s, int length){
        int l;
        l = strlen(s);
        while(l<length){
        s[l] = ' ';
        l++;
        }
        s[l] = '\0';
}

int strcicmp(char const *a, char const *b){
        for (;; a++, b++){
        int d = tolower(*a) - tolower(*b);
        if(d !=0 || !*a)
        return d;
}
}
```

C Tab Width: 8 Ln 110, Col 2 INS