CPRE 431

Module 2 Lab

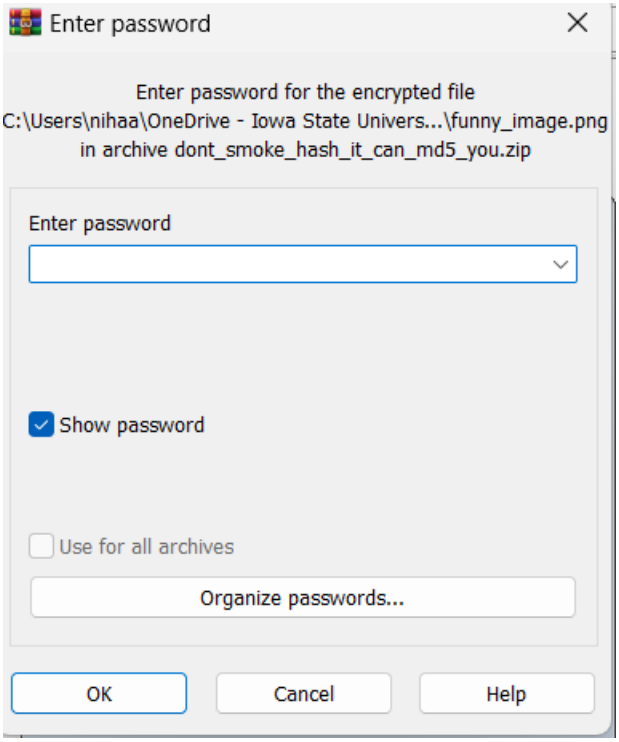
Assignments will be submitted in PDF format via Canvas.

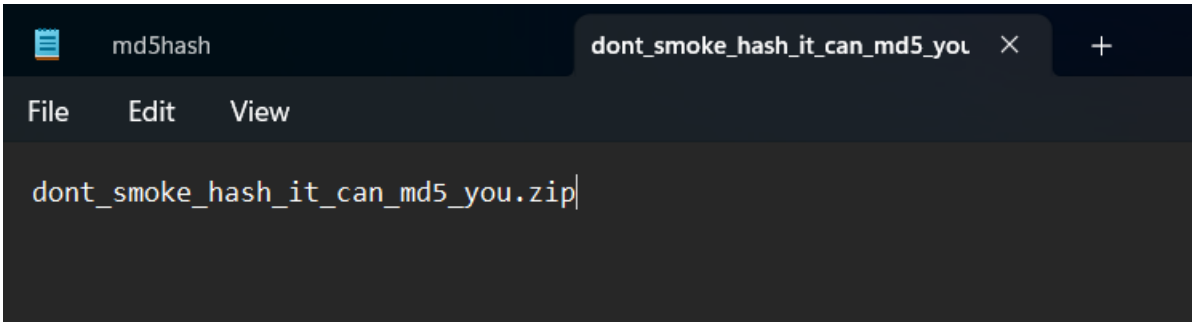
Please submit your homework online through Canvas. Late homework will not be accepted.
Important: Your submission must be in .pdf format ONLY!

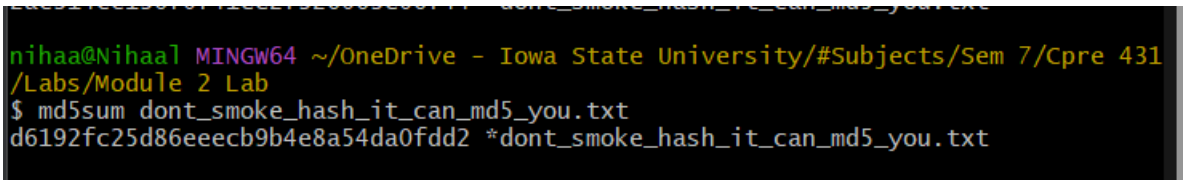
1. You are the security administrator in the XYZ company, and you noticed some strange traffic on the network of one of the employees sending an encrypted file to someone outside of the company. You want to check if the content doesn't have any confidential data. The encrypted file is attached with the homework (dont_smoke_hash_it_can_md5_you.zip). From the file name, you guessed that the password could be the md5 hash of the file name with the extension.

    a. Explain with screenshots how to use OpenSSL tool to determine the md5 hash, then use any file decompression tool (WinRAR, Winzip, etc.) to decrypt contents of the attached file. Hint: write the file name in a text file then get the hash for this text file. Note that hash is space sensitive.

We write the name of the file 'don't_smoke_hash_it_can_md5_you.zip' to notepad. Then hash this text file using 'md5sum' to hash. The hash is the password







b. After decryption, you noticed that the content of the attached file was an excel sheet "my_funny_expenses.csv", an image "funny_image.png" and a text file "funny_jokes.txt". This could look normal, however there is a lot of secrets and hidden messages in the content. Depending on your cryptanalysis skills, you started with the text file and you noticed that it only contained random characters but it started with the word "Salted__". From your experience with OpenSSL tool, this means that this text file is encrypted using Symmetric Cipher (des, des3, aes128, aes192, aes256, etc.). Also, you noticed that all the three files have the word "funny" in their name, maybe this could be the encryption key. Explain with screenshots how to use OpenSSL tool to decrypt the original message in the text file.

Using openSSL, we can decrypt a text file. Taking our example, we use;

openssl enc -d -aes-256-cbc -in funny_jokes.txt -out decrypted_jokes.txt -k funny

openssl enc: encryption/decryption

-d: decryption

-aes-256-cbc: encryption algorithm used

-in funny_jokes.txt: input encrypted file

-out decrypted_jokes.txt: decrypted output file

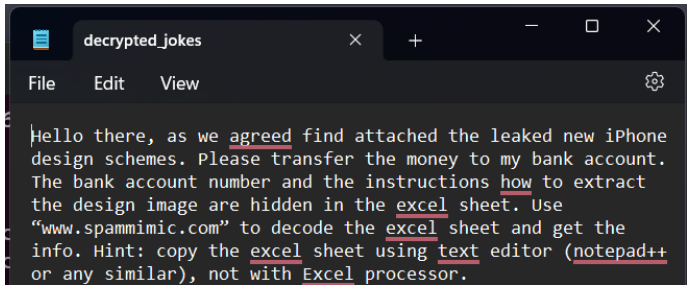
-k funny: encryption key

c. The decrypted message of part (a) will contain the instructions for this question. Explain with screenshots how did you solve this secret and get the new key.
On analysis, we can guess that the password must be 'funny', using OpenSSL we can try to decrypt the funny_jokes.txt file. Now, we must guess what symmetric algorithm it uses and try each one. Finally, we find that 'aes-256-cbc' gives us sensible data.
The decrypted text file gives instructions of decrypting a CSV file called my_funny_expenses.csv.

```
root@Nihaal:~# openssl help
help:

Cipher commands (see the `enc' command for more details)
aes-128-cbc       aes-128-ecb       aes-192-cbc       aes-192-ecb
aes-256-cbc       aes-256-ecb       aria-128-cbc      aria-128-cfb
aria-128-cfb1     aria-128-cfb8     aria-128-ctr      aria-128-ecb
aria-128-ofb      aria-192-cbc      aria-192-cfb      aria-192-cfb1
aria-192-cfb8     aria-192-ctr      aria-192-ecb      aria-192-ofb
aria-256-cbc      aria-256-cfb      aria-256-cfb1     aria-256-cfb8
aria-256-ctr      aria-256-ecb      aria-256-ofb      base64
bf                bf-cbc            bf-cfb            bf-ecb
bf-ofb            camellia-128-cbc  camellia-128-ecb  camellia-192-cbc
camellia-192-ecb  camellia-256-cbc  camellia-256-ecb  cast
cast-cbc          cast5-cbc         cast5-cfb         cast5-ecb
cast5-ofb         des               des-cbc           des-cfb
des-ecb           des-ede           des-ede-cbc       des-ede-cfb
des-ede-ofb       des-ede3          des-ede3-cbc      des-ede3-cfb
des-ede3-ofb      des-ofb           des3              desx
rc2               rc2-40-cbc        rc2-64-cbc        rc2-cbc
rc2-cfb           rc2-ecb           rc2-ofb           rc4
rc4-40            seed              seed-cbc          seed-cfb
seed-ecb          seed-ofb          sm4-cbc           sm4-cfb
sm4-ctr           sm4-ecb           sm4-ofb
```

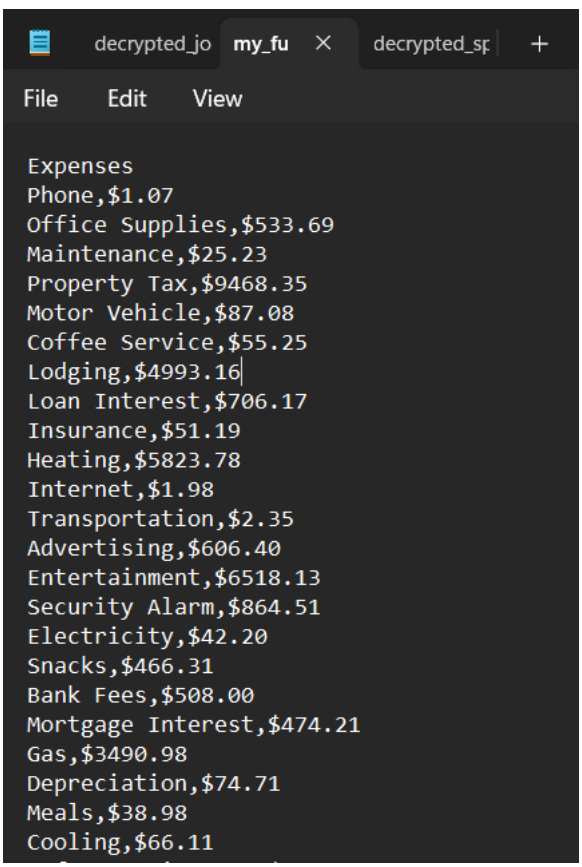
```
nihaa@Nihaal MINGW64 ~/OneDrive - Iowa State University/#Subjects/Sem 7/Cpre 431
/Labs/Module 2 Lab/For Windows Users_ dont_smoke_hash_it_can_md5_you.zip
$ openssl enc -d -aes-192 -in funny_jokes.txt -out decrypted1_jokes.txt -k funny
enc: Unknown cipher: aes-192
enc: Use -help for summary.
7C3E0000:error:0308010C:digital envelope routines:inner_evp_generic_fetch:unsupp
orted:../openssl-3.1.2/crypto/evp/evp_fetch.c:341:Global default library context
, Algorithm (aes-192 : 0), Properties (<null>)

nihaa@Nihaal MINGW64 ~/OneDrive - Iowa State University/#Subjects/Sem 7/Cpre 431
/Labs/Module 2 Lab/For Windows Users_ dont_smoke_hash_it_can_md5_you.zip
$ openssl enc -d -des-ecb -in funny_jokes.txt -out decrypted1_jokes.txt -k funny
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
Error setting cipher DES-ECB
```

```
nihaa@Nihaal MINGW64 ~/OneDrive - Iowa State University/#Subjects/Sem 7/Cpre 431
/Labs/Module 2 Lab/For Windows Users_ dont_smoke_hash_it_can_md5_you.zip
$ openssl enc -d -aes-256-cbc -in funny_jokes.txt -out decrypted_jokes.txt -k fu
nny
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
```

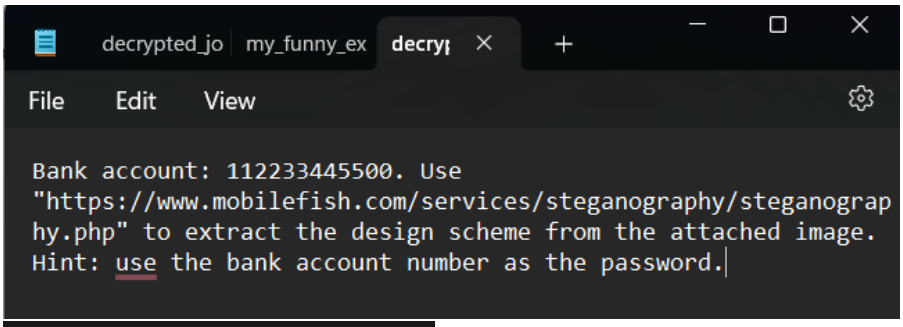


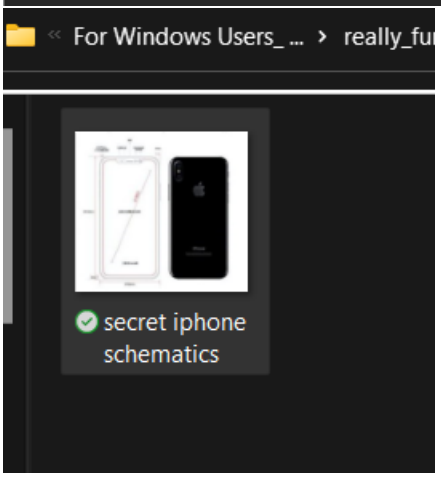
d. The decrypted message of part (b) will contain the instructions for this question. Explain with screenshots how did you solve this secret and get the final leaked confidential information. Use the file on Canvas instead of the website given from the previous step to complete the final deliverable of the lab.

Using the decrypted instructions on decrypted_jokes.txt output file, we open the csv file in notepad and copy its contents and paste it the spammic.com website. Once that is decrypted, we get another message. For readability, we copy this to another txt file. Using the bank account number as password, we use a stenography tool, or in this case the zip file given on Canvas, extract it (password is the account number). Finally we get an image of a supposedly leaked iPhone dimensions.




mimic
Decoded Spreadsheet
Your spreadsheet Expenses Phone,$1.07 Office Supplies,$... decodes to:
Bank account: 112233445500. Use "https://ww
Encode
Copyright © 2000-2023 spammimic.com, All rights reserved
back


decrypted_jo
my_funny_ex
decryp
File
Edit
View
Bank account: 112233445500. Use "https://www.mobilefish.com/services/steganography/steganography.php" to extract the design scheme from the attached image. Hint: use the bank account number as the password.


For Windows Users_ ...
really_fu
secret iphone schematics


2.577mm
4mm
5.76吋
137.54mm
4mm
67.54mm
iPhone