Worcester Polytechnic Institute
Department of Computer Science

Lab 6: Steghide / Stegseek / ccrypt

Estimated lab time: 15 minutes - 25 minutes

Disclaimer: This lab is for educational purposes only.

Introduction:

Steghide is a steganography program that is able to hide data in various kinds of image- and audio-files. The color- respectivly sample-frequencies are not changed thus making the embedding resistant against first-order statistical tests.

Stegseek is a lightning fast steghide cracker that can be used to extract hidden data from files. It is built as a fork of the original steghide project and, as a result, it is thousands of times faster than other crackers and can run through the entirety of rockyou.txt* in under 2 seconds.

ccrypt is a tool for encrypting and decrypting files and streams. It is based on the Rijndael block cipher, a version of which is also used in the Advanced Encryption Standard (AES, see http://www.nist.gov/aes). This cipher is believed to provide very strong security.

Steghide: http://steghide.sourceforge.net/index.php

Stegseek: https://github.com/RickdeJager/stegseek

Ccrypt: http://ccrypt.sourceforge.net

Objectives:

- Learn how to hide data with Steghide
- Learn how to find the passphrase of a hidden data with Stegseek
- Learn encrypting and decrypting with ccrypt

Tasks

Task 1: Software preparation

1. Install a Kali Linux virtual machine. You can install Oracle VirtualBox or VMware workstation and then add Kali Linux virtual machine.

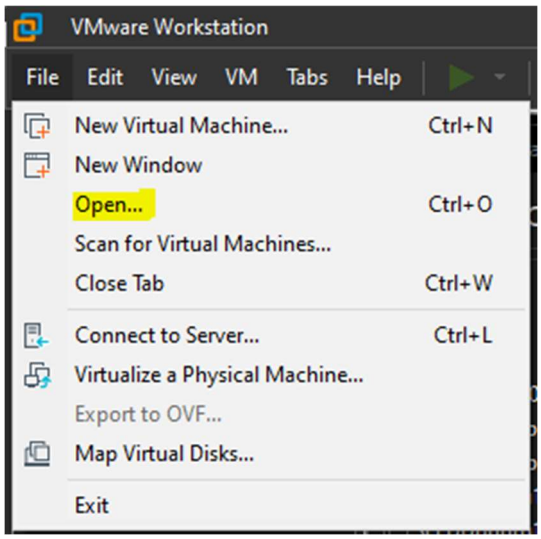
The download page for Oracle VirtualBox is: https://www.virtualbox.org/wiki/Downloads

The download page for Kali Linux is: https://www.kali.org/get-kali/

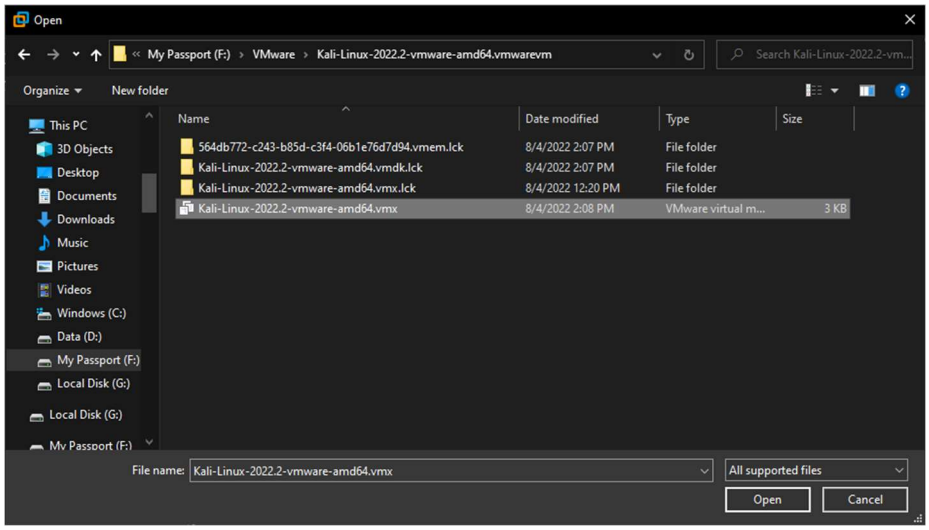
(Install Kali in Vmware workstation)

1.1 Unzip the downloaded kali

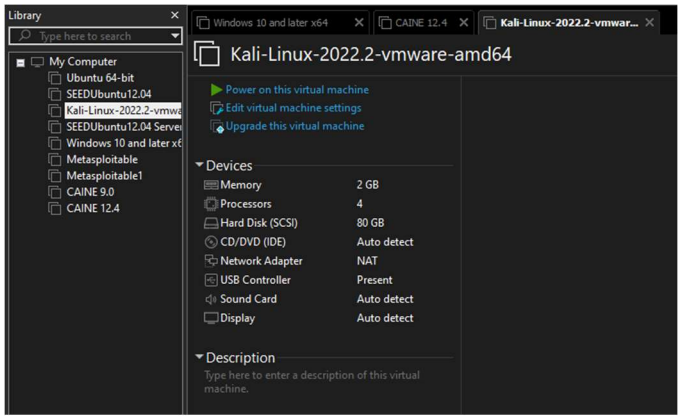
1.2 In VMware Workstation, select File, Open…



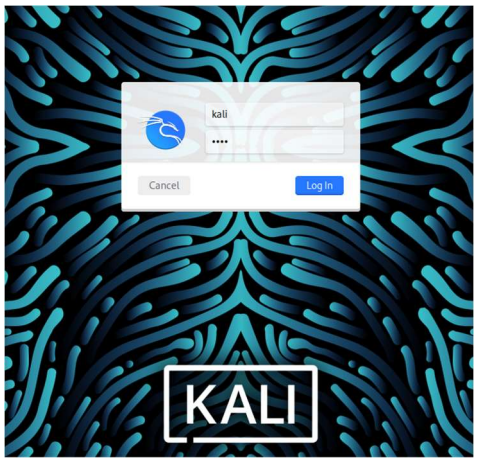
1.3 Select the vmx file.



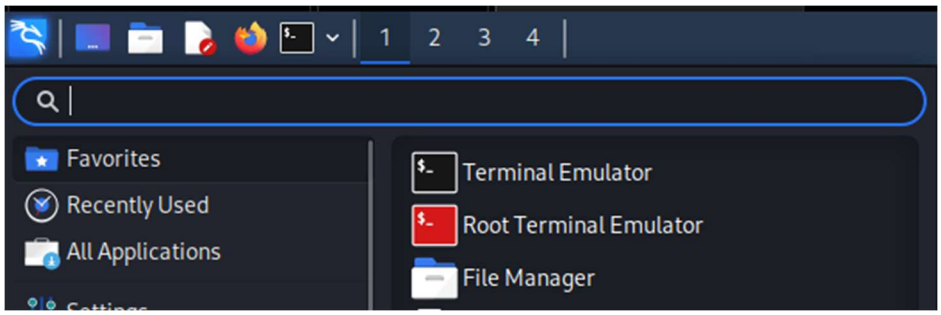
2. Power on Kali Linux virtual machine



3. Login to Kali. The default credentials "kali/kali".



4. Open the terminal



5. Install Steghide

   5.1 Enter **sudo apt install steghide** in the terminal

```
┌──(kali㉿kali)-[~]
└─$ sudo apt install steghide
Reading package lists ... Done
Building dependency tree ... Done
Reading state information ... Done
The following additional packages will be installed:
  libmcrypt4 libmhash2
Suggested packages:
  libmcrypt-dev mcrypt
The following NEW packages will be installed:
  libmcrypt4 libmhash2 steghide
0 upgraded, 3 newly installed, 0 to remove and 7 not upgraded.
Need to get 311 kB of archives.
After this operation, 907 kB of additional disk space will be used.
Do you want to continue? [Y/n] 
```

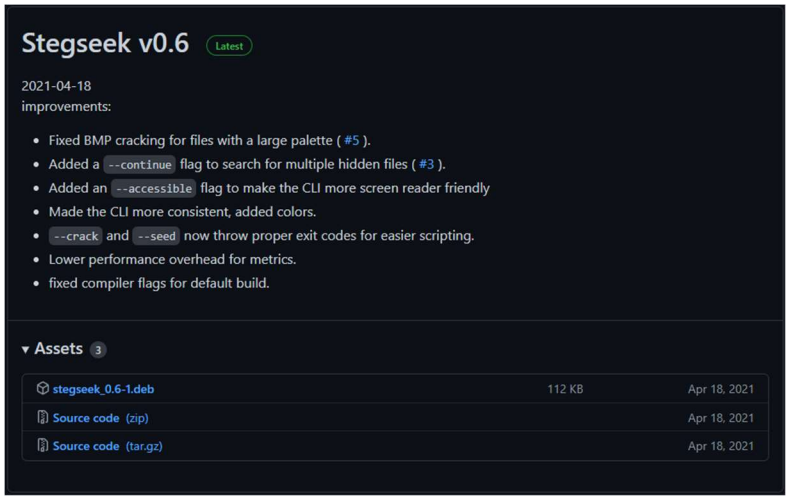
5.2 Enter Y

```
Do you want to continue? [Y/n] y
Get:1 http://mirrors.jevincanders.net/kali kali-rolling/main amd64 libmcrypt4
 amd64 2.5.8-7 [72.6 kB]
Get:2 http://mirrors.jevincanders.net/kali kali-rolling/main amd64 libmhash2
amd64 0.9.9.9-9 [94.2 kB]
Get:3 http://mirrors.jevincanders.net/kali kali-rolling/main amd64 steghide a
md64 0.5.1-15 [144 kB]
Fetched 311 kB in 1s (270 kB/s)
Selecting previously unselected package libmcrypt4.
(Reading database ... 348519 files and directories currently installed.)
Preparing to unpack .../libmcrypt4_2.5.8-7_amd64.deb ...
Unpacking libmcrypt4 (2.5.8-7) ...
Selecting previously unselected package libmhash2:amd64.
Preparing to unpack .../libmhash2_0.9.9.9-9_amd64.deb ...
Unpacking libmhash2:amd64 (0.9.9.9-9) ...
Selecting previously unselected package steghide.
Preparing to unpack .../steghide_0.5.1-15_amd64.deb ...
Unpacking steghide (0.5.1-15) ...
Setting up libmhash2:amd64 (0.9.9.9-9) ...
Setting up libmcrypt4 (2.5.8-7) ...
Setting up steghide (0.5.1-15) ...
Processing triggers for libc-bin (2.33-6) ...
Processing triggers for man-db (2.10.2-1) ...
Processing triggers for kali-menu (2022.2.0) ...
```

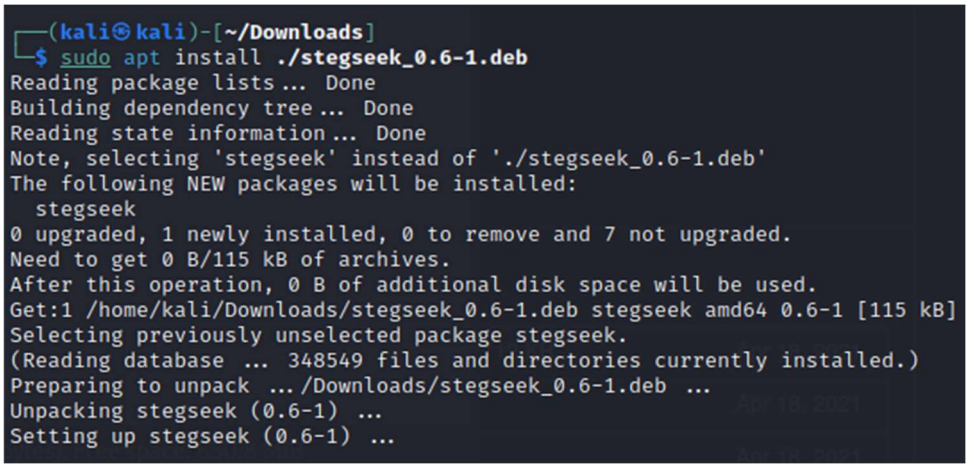
6. Install Stegseek

6.1 Download the latest Stegseek release from this page:

https://github.com/RickdeJager/stegseek/releases



6.2 Install the **.deb** file using **sudo apt install ./stegseek_0.6-1.deb** in the terminal



7. Install ccrypt

7.1 Enter **sudo apt install ccrypt** in the terminal

```
┌──(kali㉿kali)-[~]
└─$ sudo apt install ccrypt
Reading package lists ... Done
Building dependency tree ... Done
Reading state information ... Done
Suggested packages:
  elpa-ps-ccrypt
The following NEW packages will be installed:
  ccrypt
0 upgraded, 1 newly installed, 0 to remove and 7 not upgraded.
Need to get 64.4 kB of archives.
After this operation, 185 kB of additional disk space will be used.
Get:1 http://mirrors.jevincanders.net/kali kali-rolling/main amd64 ccrypt amd
64 1.11-2 [64.4 kB]
Fetched 64.4 kB in 1s (102 kB/s)
Selecting previously unselected package ccrypt.
(Reading database ... 348551 files and directories currently installed.)
Preparing to unpack .../ccrypt_1.11-2_amd64.deb ...
Unpacking ccrypt (1.11-2) ...
Setting up ccrypt (1.11-2) ...
Processing triggers for kali-menu (2022.2.0) ...
Processing triggers for man-db (2.10.2-1) ...
```

8. Unzipping Rockyou.txt.gz in Kali Linux

   8.1 Enter **sudo gzip -d /usr/share/wordlists/rockyou.txt.gz** in the terminal

9. Download fun.bmp in Kali linux

   https://drive.google.com/file/d/1SlGqpmFZ8CKYVD_Z7Ru6or12NqfbcexC/view?usp=sharing

10. Download lake.jpeg.cpt in Kali linux

    https://drive.google.com/file/d/19tRHTsuxis0Gz9eDuh0F2eKgCRakXfY4/view?usp=sharing

Task 2: Hiding and encrypting data using Steghide and ccrypt

1. Create a secret message document

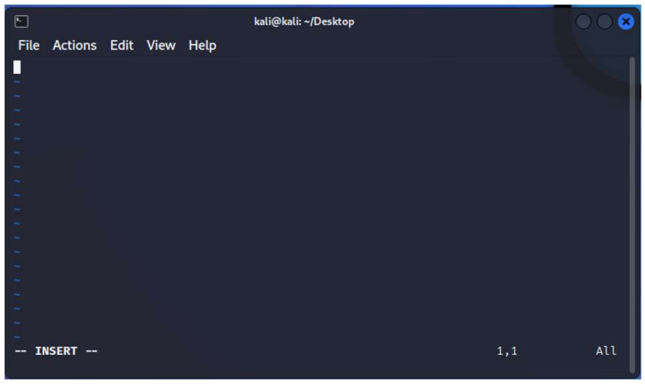
1.1 Enter **cd Desktop/**

```
┌──(kali㉿kali)-[~]
└─$ cd Desktop
```

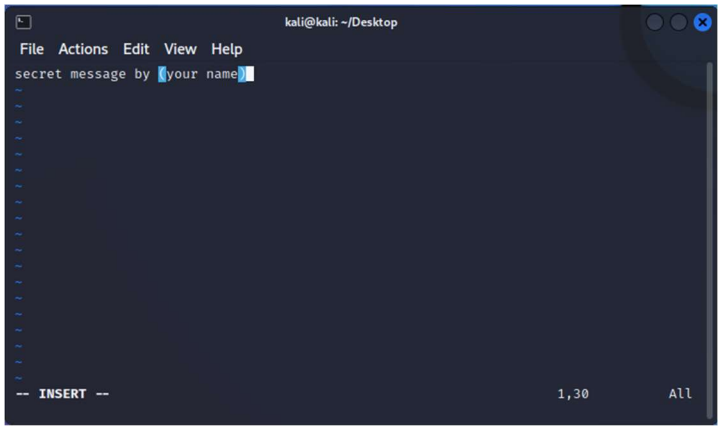
1.2 Enter **vim message**

```
┌──(kali㉿kali)-[~/Desktop]
└─$ vim message
```

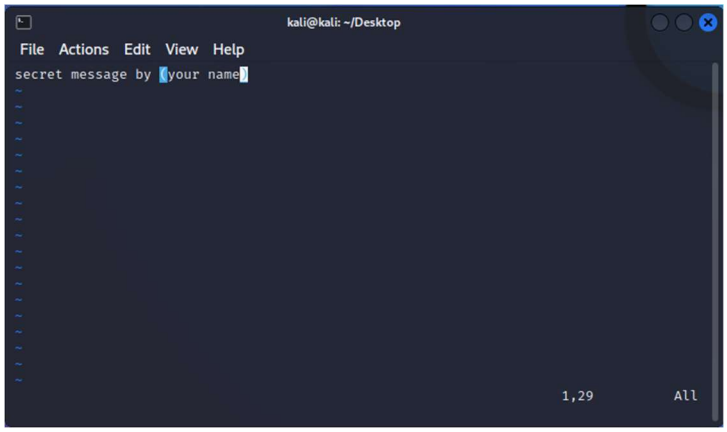
1.3 Type **i** to get into insert mode



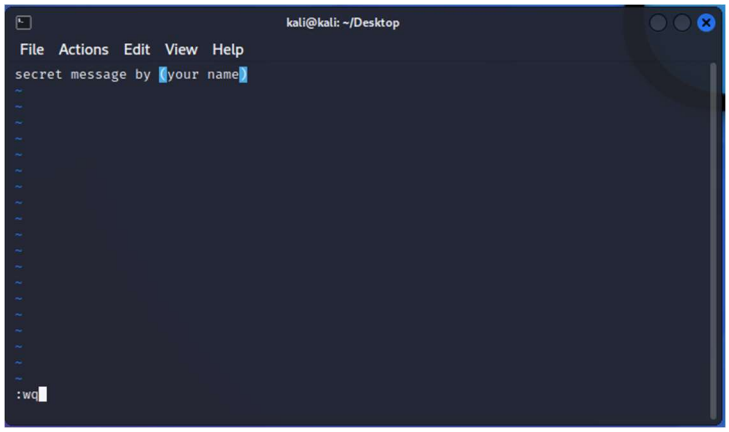
1.4 Enter the message **secret message by (your name)**



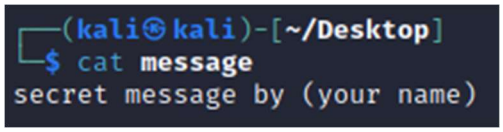
1.5 Press **Esc** to exit insert mode



1.6 Type **:wq** to save the document and quit the vim



1.7 You can use **cat message** to view and check the document



2. Embedding message into image using Steghide

(Before embedding message into image, check and take a screenshot of the size of the original image by using the command **ls -l**)

2.1 Type **steghide –help** to see the help menu

```
└─$ steghide --help
steghide version 0.5.1

the first argument must be one of the following:
 embed, --embed          embed data
 extract, --extract      extract data
 info, --info            display information about a cover- or stego-file
   info <filename>       display information about <filename>
 encinfo, --encinfo      display a list of supported encryption algorithms
 version, --version      display version information
 license, --license      display steghide's license
 help, --help            display this usage information

embedding options:
 -ef, --embedfile        select file to be embedded
   -ef <filename>        embed the file <filename>
 -cf, --coverfile        select cover-file
   -cf <filename>        embed into the file <filename>
 -p, --passphrase        specify passphrase
   -p <passphrase>       use <passphrase> to embed data
 -sf, --stegofile        select stego file
   -sf <filename>        write result to <filename> instead of cover-file
 -e, --encryption        select encryption parameters
   -e <a>[<m>]|<m>[<a>]  specify an encryption algorithm and/or mode
   -e none               do not encrypt data before embedding
 -z, --compress          compress data before embedding (default)
   -z <l>                 using level <l> (1 best speed ... 9 best compression)
 -Z, --dontcompress      do not compress data before embedding
 -K, --nochecksum        do not embed crc32 checksum of embedded data
 -N, --dontembedname     do not embed the name of the original file
 -f, --force             overwrite existing files
 -q, --quiet             suppress information messages
 -v, --verbose           display detailed information

extracting options:
 -sf, --stegofile        select stego file
   -sf <filename>        extract data from <filename>
 -p, --passphrase        specify passphrase
   -p <passphrase>       use <passphrase> to extract data
 -xf, --extractfile      select file name for extracted data
   -xf <filename>        write the extracted data to <filename>
 -f, --force             overwrite existing files
 -q, --quiet             suppress information messages
 -v, --verbose           display detailed information

options for the info command:
 -p, --passphrase        specify passphrase
   -p <passphrase>       use <passphrase> to get info about embedded data

To embed emb.txt in cvr.jpg: steghide embed -cf cvr.jpg -ef emb.txt
To extract embedded data from stg.jpg: steghide extract -sf stg.jpg
```

2.2 Type **steghide embed -ef message -cf fun.bmp**

```
┌──(kali㉿kali)-[~/Desktop]
└─$ steghide embed -ef message -cf fun.bmp
Enter passphrase:
```

2.3 Enter the passphrase **computer** and re-enter the passphrase again.

```
┌──(kali㉿kali)-[~/Desktop]
└─$ steghide embed -ef message -cf fun.bmp
Enter passphrase:
Re-Enter passphrase:
embedding "message" in "fun.bmp"... done
```

(Now the secret message is hidden in the fun.bmp)

You can check the info of the image by using **steghide info fun.bmp**

You need to type y and enter the passphrase

```
┌──(kali㉿kali)-[~/Desktop]
└─$ steghide info fun.bmp
"fun.bmp":
  format: Windows 3.x bitmap
  capacity: 975.4 KB
Try to get information about embedded data ? (y/n) y
Enter passphrase:
  embedded file "message":
    size: 30.0 Byte
    encrypted: rijndael-128, cbc
    compressed: yes
```

(Check and take a screenshot of the size of the original image by using the command **ls -l**)

3. Encrypt data with ccrypt

3.1 Type **ccrypt -help** to see the help menu for ccrypt

```
┌──(kali㉿kali)-[~/Desktop]
└─$ ccrypt -h
ccrypt 1.11. Secure encryption and decryption of files and streams.

Usage: ccrypt [mode] [options] [file ... ]
       ccencrypt [options] [file ... ]
       ccdecrypt [options] [file ... ]
       ccat [options] file ...

Modes:
    -e, --encrypt         encrypt
    -d, --decrypt         decrypt
    -c, --cat             cat; decrypt files to stdout
    -x, --keychange       change key
    -u, --unixcrypt       decrypt old unix crypt files

Options:
    -h, --help            print this help message and exit
    -V, --version         print version info and exit
    -L, --license         print license info and exit
    -v, --verbose         print progress information to stderr
    -q, --quiet           run quietly; suppress warnings
    -f, --force           overwrite existing files without asking
    -m, --mismatch        allow decryption with non-matching key
    -E, --envvar var      read keyword from environment variable (unsafe)
    -K, --key key         give keyword on command line (unsafe)
    -k, --keyfile file    read keyword(s) as first line(s) from file
    -P, --prompt prompt   use this prompt instead of default
    -S, --suffix .suf     use suffix .suf instead of default .cpt
    -s, --strictsuffix    refuse to encrypt files which already have suffix
    -F, --envvar2 var     as -E for second keyword (for keychange mode)
    -H, --key2 key        as -K for second keyword (for keychange mode)
    -Q, --prompt2 prompt  as -P for second keyword (for keychange mode)
    -t, --timid           prompt twice for encryption keys (default)
    -b, --brave           prompt only once for encryption keys
    -y, --keyref file     encryption key must match this encrypted file
    -r, --recursive       recurse through directories
    -R, --rec-symlinks    follow symbolic links as subdirectories
    -l, --symlinks        dereference symbolic links
    -T, --tmpfiles        use temporary files instead of overwriting (unsafe)
    --                    end of options, filenames follow
```

3.2 Type **ccrypt -e fun.bmp**

3.2 Enter the encryption key **computer** and re-enter the encryption key again.

```
┌──(kali㉿kali)-[~/Desktop]
└─$ ccrypt -e fun.bmp
Enter encryption key:
Enter encryption key: (repeat)
```

(Now the encrypted data is generated under the same directory, which is desktop)

The encrypted data called **fun.bmp.cpt**

Task 3: Extract and decrypt data using Steghide and ccrypt

1. Decrypt data via ccrypt

1.1 Type **ccrypt -d fun.bmp.cpt**

1.2 Enter the decryption key **computer**

```
┌──(kali㉿kali)-[~/Desktop]
└─$ ccrypt -d fun.bmp.cpt
Enter decryption key:
```

(Now the decrypted data is generated under the same directory, which is desktop)

The data return to **fun.bmp**

2. Extract Data From image via Steghide

(Before extracting data from the image, you can delete the message document we created in Task 2 section 1.)

The command to remove the message document is **rm message**

2.1 Type **steghide extract -sf fun.bmp**

2.2 Enter the passphrase **computer** and re-enter the passphrase again.

```
┌──(kali㉿kali)-[~/Desktop]
└─$ steghide extract -sf fun.bmp
Enter passphrase:
wrote extracted data to "message".
```

2.3 Type **cat message** to reveal the message

```
┌──(kali㉿kali)-[~/Desktop]
└─$ cat message
secret message by (your name)
```

Task 4: Extract and decrypt data via brute force attack

1. Extract .cpt data via ccguess

1.1 Type **ccguess –h** to see the help menu

1.2 Type **ccguess lake.jpeg.cpt**

1.3 Since we don't know the passphrase of the encrypted data, we left it blank, and press enter

```
┌──(kali㉿kali)-[~/Desktop]
└─$ ccguess lake.jpeg.cpt
Enter approximate key:
```

1.4 Take a screenshot when it finds the possible match, the process should not be taken longer than 10 seconds.

```
┌──(kali㉿kali)-[~/Desktop]
└─$ ccguess lake.jpeg.cpt
Enter approximate key:

Generating patterns ... 1 .. 2 .. 3 .. 4 .. 5 .. sorting ... done.
```

1.5 Type **ccrypt –d lake.jpeg.cpt** to decrypt the data. Enter the key that you found from last step.

```
┌──(kali㉿kali)-[~/Desktop]
└─$ ccrypt -d lake.jpeg.cpt
Enter decryption key:
```

(Now you will be able to see the data changed from lake.jpeg.cpt to lake.jpeg and you can view the image)

2. Extract Data From image via Stegseek

2.1 Type **stegseek lake.jpeg /usr/share/wordlists/rockyou.txt**

```
┌──(kali㉿kali)-[~/Desktop]
└─$ stegseek lake.jpeg /usr/share/wordlists/rockyou.txt
```

2.2 Take a screenshot when it finds the passphrase, the process should not be taken longer than 10 seconds.

2.3 Type **cat lake.jpeg.out** the view the document and take a screenshot of the message.

Congratulations. You have completed this lab. Now that you understand we can hide the data by using various steganography tools. Furthermore, you can hide the data by altering the extension name, changing the header file.

Question:

1. In task 2, What is the size of the original image? What is the size of the embedded image? Is there any difference?

2. In task 4, What is the passphrase of lake.jpeg.cpt?

3. In task 4, What is the passphrase of lake.jpeg?

4. In task 4, What is the message hidden inside the image?

(Optional, Extra challenge)

5. What's the hidden message (Hint: Caesar-cipher)