CITS 3004
Cybersecurity

THE UNIVERSITY OF WESTERN AUSTRALIA

Lab 8: Steganography

Task 1: White Space Steganography

Written by Jin Hong

This task will explore hiding secret messages using the white space. The basic idea is to embed a bit information into every line of the original file. For example, if I want to embed a message "1101001", the original text can be converted as shown in Figure 1. You can use the provided original.txt file, which is sufficiently large for a moderate length of the secret message. You can also find larger text file to be used instead. You are given `WSencoder.py` and `WSdecoder.py`, which are used to encode and decode secret messages hidden using the white space. The `WSencoder.py` code is incomplete, which you have to do. Complete this first. The instruction is provided in the comments section. Once you completed, make sure it works by trying to encode and decode secret messages.

```
But tehre was was no great luxury about the furnishng of these rooms eithar.
the beatifuly decorated table.
both the latter made in the monastery, and famous in the neigborhood.
There was no vodka.
then salmon cutlets, ice pudding and compote, and finally, blanc-mange.
where he already had a footing.
He had a footting everywhere, and got informaiton about everything.
```

(a) without white space steganography

```
But tehre was was no great luxury about the furnishng of these rooms eithar.
the beatifuly decorated table.
both the latter made in the monastery, and famous in the neigborhood.
There was no vodka.
then salmon cutlets, ice pudding and compote, and finally, blanc-mange.
where he already had a footing.
He had a footting everywhere, and got informaiton about everything.
```

(b) with white space steganography, white spaces are added at the end of some sentences.

Figure 1. The use of white space steganograhy

Note: some people find the encoded message has smaller size, and still provide the valid secret message when decoded. In this case, create a new original file by copying the content from the old original file (you should notice the file size has reduced with the same content). Then use the new original file to carry out this lab. The problem happens due to different file payload handing by different operating systems.

(Optional to do): Currently, there is no confidentiality for the hidden message (i.e., anyone who intercepts the original text can view the hidden message). We want to make the hidden message encrypted, but not the original text. You can incorporate an encryption with a set password to hide the hidden message even if the eavesdropper finds it. You can try implementing this on your own time.

Answer Q1 ~ Q3 on LMS

CITS 3004
Cybersecurity

THE UNIVERSITY OF WESTERN AUSTRALIA

Task 2: Least Significant Bit Steganography

Written by Jin Hong

Instead of text files and using white spaces, using the least significant bits (LSB) in the image to include a hidden message is more efficient. Since the image file is already large, and each pixel contains three RGB components, that's 3 bits that can be included for each pixel.

For this task, you will first need to install the `Python Pillow` package. We will use Python3.6 that we installed already from Lab 4. If you are using a fresh copy of the VM, go back to lab 4 task 2A to setup python3.6. Then, you can install the pillow package by using the following command:

```
# sudo -H pip3.6 install pillow
```

Ensure that you are using the right pip version for python 3.6 (could be `pip`, `pip3` or `pip3.6`). If it is already installed, but you get an import error, then you need to reinstall `Pillow`. You can do it by:

```
# sudo pip uninstall pillow
# sudo apt-get purge python3-pil
# sudo -H pip install pillow
```

You are provided with three Python files: `LSBencoder.py`, `LSBdecoder.py` and `LsbSteg.py`. The `LsbSteg.py` code provides all the functions to encode and decode secret messages into the image file. You should read the functions to understand what they do. The encoder code is completed for you, but the decoder is not. Your task is to complete the decoder code. The instructions are provided to you in the code. You can run the encoder as the following command:

```
# python3.6 LSBencoder.py
```

Inspect the original image and the stegano image and see if you can find any difference. Try using an image which is all black or white pixels. Can you spot the difference now?

(Optional to do): Currently, you can only generate a .png stegano images as outputs. You can try to extend the current code to expand the compatible image formats. However, it would only work with images that can support lossless data compression. What would happen if you try with images that does not support this? In addition, you can also add encryption features to protect the hidden message being read by eavesdroppers.

Answer Q4 ~ Q6 on LMS

CITS 3004
Cybersecurity

THE UNIVERSITY OF WESTERN AUSTRALIA

Task 3: Steganography Tool: Steghide

Written by Jin Hong

For this task, we will be using a tool named Steghide. You can use this tool on any Debian distributions. First, install Steghide using the following command.

```
# sudo apt-get install steghide
```

3A) HIDING TEXT FILE IN AN IMAGE

To hide a hidden message, we first need (1) an image such as jpg, png etc, and (2) a secret text file, such as .txt etc. You can directly embed the secret message into the original image using the following command, given the image is named `linux.jpg`, and the secret message is saved in `secret.txt` file.

```
# steghide embed -cf linux.jpg -ef secret.txt
```

You will be prompted to enter the password. Set your own password here. You should be able to see that it seems like nothing happened. To verify, we will delete the secret message file, and recover it in the next question.

3B) EXTRACTING HIDDEN TEXT FILE FROM AN IMAGE

The image file that has the secret message embedded can now be sent to the receiver. For our purpose, we will just delete the secret text file. The receiver must also have the steghide tool available, and knows the password that you used to embed the secret message. Enter the following command to extract the secret message from the original image.

```
# steghide extract -sf linux.jpg
```

Again, you will be prompted to enter the password. If the correct password is provided, then you should see the secret message file appearing in the same folder. Ensure the secret message is correct.

CITS 3004
Cybersecurity

THE UNIVERSITY OF WESTERN AUSTRALIA

Task 4: Steganography Tool: Stegcracker

Written by Alex Brown

4A) VIEWING THE EMBEDDED DATA INFORMATION

If you would like to view what has been embedded in the image file before extracting it, you can use the info command to preview the embedded file. The command is:

```
# steghide info linux.jpg
```

You will have to provide a correct password to view the information detail.

Answer Q7 ~ Q10 on LMS

4B) CRACKING STEGHIDE MESSAGES WITH STEGCRACKER

In this task we will be cracking the password that was used to encrypt a message using Steghide using a utility called Stegcracker. To use Stegcracker, you need to have Steghide and Python 3.6 installed on your VM. You should have installed Python 3.6 for a previous lab, but if you missed it you can run the following commands to install Python 3.6.

```
# sudo apt-get install software-properties-common
# sudo add-apt-repository ppa:deadsnakes/ppa
# sudo apt-get update
# sudo apt-get install python3.6
```

We also need to have pip installed for Python 3.6 to install Stegcracker which can be installed using the following command.

```
# curl https://bootstrap.pypa.io/get-pip.py | sudo -H python3.6
```

Finally, we can now install Stegcracker with the command.

```
# pip3.6 install stegcracker
```

How Stegcracker works is by testing each password in a wordlist by running the `steghide extract` command and checking if it executed successfully. You can have a look at the source code to have a better understanding of how it works (https://github.com/Paradoxis/StegCracker).

CITS 3004
Cybersecurity

The wordlist that we will you use for this lab task is called `rockyou.txt`. It is actually a password dump of over 14 million real user passwords from the data breach of a social application site called RockYou back in 2009. The wordlist is commonly used by penetration testers and hackers to crack weak passwords that people use. You can download the wordlist by using the following commands.

```
# curl \
https://raw.githubusercontent.com/danielmiessler/SecLists/master/Passw
ords/Leaked-Databases/rockyou.txt.tar.gz \
-o rockyou.txt.tar.gz
# tar zxvf rockyou.txt.tar.gz
```

Now you can run the following command to try and crack the password used by Steghide by checking every password in the `rockyou.txt` wordlist.

```
# stegcracker <file> rockyou.txt
```

Now let us try to crack the password for the `hackerman.jpg` image and get the secret flag hidden inside!
(The file is on LMS)

Answer Q11 and Q12 on LMS

CITS 3004
Cybersecurity

Task 5: Complex Steganography

Written by Alex Brown

The methods of Steganography that we investigated in the earlier tasks of this laboratory are just the basics and can be made more difficult to decode by stacking multiple methods on top of each other or by using a more obscure methodology.

A recent example of a highly sophisticated use of Steganography was discovered in the hack of Belgacom that was allegedly perpetrated by GCHQ in an eavesdropping mission called “Operation Socialist”. The malware that was used to infect the Belgacom servers was called Regin and is widely believed that it was created by either the NSA, GCHQ or both of them in collaboration. One of the methods that Regin uses to hide data being sent to and from infected servers is by encrypting it first and then hiding the bits in custom TCP and UDP protocol headers. Spreading the data across multiple packets makes the malicious traffic appear to be legitimate web traffic from the perspective of a security analyst.

If you want to learn more about Operation Socialist, listen to the episode by Darknet Diaries which goes through the story (https://darknetdiaries.com/episode/48/).

For this task though, we will try to decode the secret flag that is hidden inside the image `iloveshrek.jpg`. The flag is hidden by stacking the methods of Steganography that we investigated in this lab.

Good luck!

Answer Q13 and Q14 on LMS