SECURITY+ V4 LAB SERIES

Lab 14: Cryptography Concepts

Document Version: 2024-07-29

Material in this Lab Aligns to the Following	
CompTIA Security+ (SY0-601) Exam Objectives	2.8: Summarize the basics of cryptographic concepts
All-In-One CompTIA Security+ Sixth Edition ISBN-13: 978-1260464009 Chapters	16: Cryptographic Concepts

Copyright © 2022 Network Development Group, Inc.
www.netdevgroup.com

NETLAB+ is a registered trademark of Network Development Group, Inc.
KALI LINUX ™ is a trademark of Offensive Security.
Microsoft®, Windows®, and Windows Server® are trademarks of the Microsoft group of companies.
VMware is a registered trademark of VMware, Inc.
SECURITY ONION is a trademark of Security Onion Solutions LLC.
Android is a trademark of Google LLC.
pfSense® is a registered mark owned by Electric Sheep Fencing LLC (“ESF”).
All trademarks are property of their respective owners.

Contents

Copyright © 2024 Network Development Group, Inc. www.netdevgroup.com

Introduction

In this lab, you will be conducting steganography techniques using various tools.

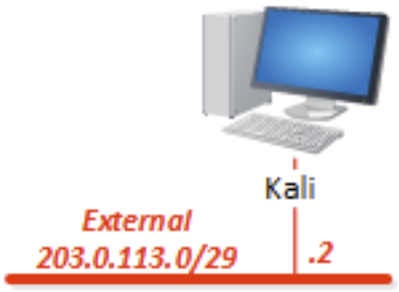
Objective

In this lab, you will perform the following tasks:

- Hiding a Hidden Message Within a Picture
- Hiding Multiple Files Within an Image File
- Observe the avalanche effect on hash functions

Copyright © 2024 Network Development Group, Inc. www.netdevgroup.com

Lab Topology



Copyright © 2024 Network Development Group, Inc. www.netdevgroup.com

Lab Settings

The information in the table below will be needed in order to complete the lab. The task sections below provide details on the use of this information.

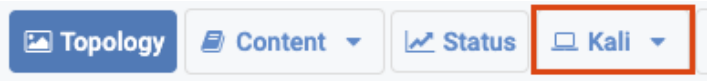
Virtual Machine	**IP Address**	**Account** (if needed)	**Password** (if needed)
Kali	`203.0.113.2`	`kali`	`kali`

Copyright © 2024 Network Development Group, Inc. www.netdevgroup.com

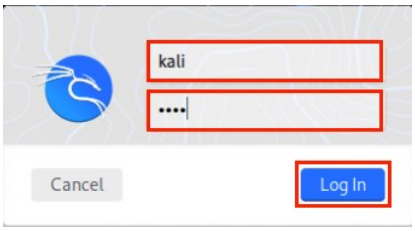
1 Hiding a Hidden Message Within a Picture

1.1 Using Steghide to Hide Hidden Messages

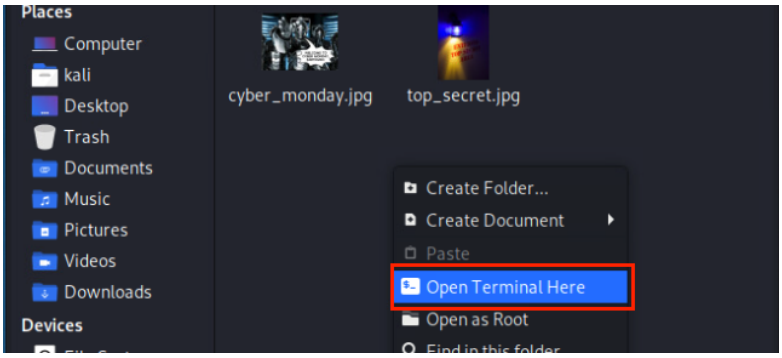
1. Launch the **Kali** virtual machine to access the graphical login screen.



2. Log in as `kali` with `kali` as the password. Open the **Kali** *PC Viewer*.

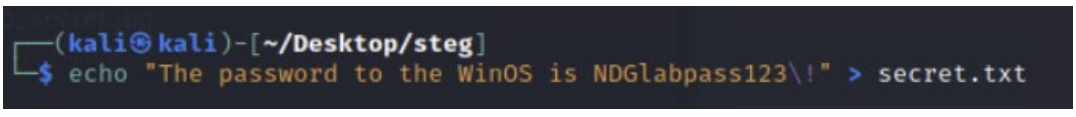


3. Double-click the **steg** folder on the desktop. After the window opens, right-click on an empty space, and choose the option, **Open Terminal Here.**



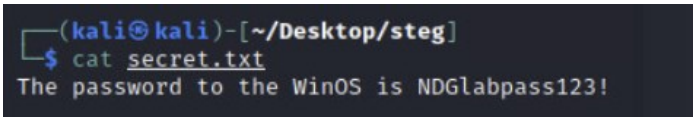
4. In the *Terminal* window, create a new text document with a secret text string. Type the message below:

```
kali@kali$ echo “The password to the WinOS is NDGlabpass123\!” > secret.txt
```



5. Double-check the content of the secret.txt file using the command:

```
kali@kali$ cat secret.txt
```

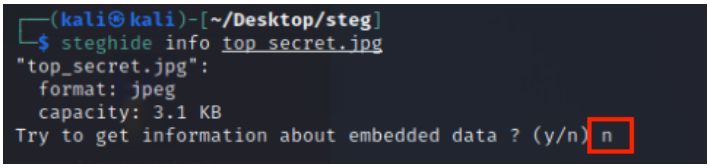


Copyright © 2024 Network Development Group, Inc. www.netdevgroup.com

6. Verify the capacity of the *top_secret.jpg* image to see what the capacity is for being able to hide a message within the image itself.

```
kali@kali$ steghide info top_secret.jpg
```

7. Notice the capacity amount. When asked to get information about embedded data, type **N**.



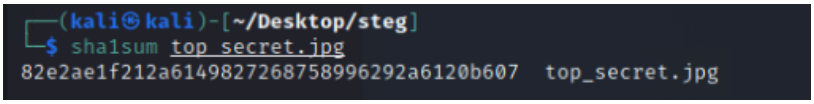
8. See how large the *secret.txt* file is to confirm whether we can hide it within the *top_secret.jpg* image. Use the **du** with **–b** option to display the file size in terms of bytes.

```
kali@kali$ du -b secret.txt
```

Notice that we should be able to fit the *44 byte* sized *secret.txt* file in the *3.1 KB top_secret.jpg* image file.

9. Before we embed the *secret* message, confirm the *sha1 hash* value for the *top_secret.jpg* image file.

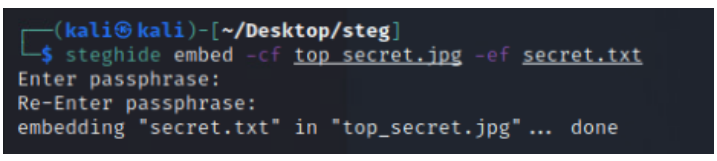
```
kali@kali$ sha1sum top_secret.jpg
```



Take note of this hash value for later comparison. Your value could be different from what is shown here.

10. Type the command below to initialize the process of hiding the secret message. When prompted for a passphrase, type **secret** followed by pressing **Enter**. Type **secret** once more. Press **Enter**.

```
kali@kali$ steghide embed -cf top_secret.jpg -ef secret.txt
```



Copyright © 2024 Network Development Group, Inc. www.netdevgroup.com

11. Verify the *hash value* again with the same *top_secret.jpg* image file.

```
┌──(kali㉿kali)-[~/Desktop/steg]
└─$ sha1sum top_secret.jpg
7a32476d6be9a1c901ed4e44afb357fef18f3d07  top_secret.jpg
```

Notice the integrity has been lost in the steganography process due to a different hash value.

12. Type the command below to gather info on the embedded data within the **top_secret.jpg** file. When asked to get information about embedded data, type Y. Type `secret` as the passphrase. Press **Enter**.

```
kali@kali$ steghide info top_secret.jpg
```

```
┌──(kali㉿kali)-[~/Desktop/steg]
└─$ steghide info top_secret.jpg
"top_secret.jpg":
  format: jpeg
  capacity: 3.1 KB
Try to get information about embedded data ? (y/n) y
Enter passphrase:
  embedded file "secret.txt":
    size: 44.0 Byte
    encrypted: rijndael-128, cbc
    compressed: yes
```

Notice the output, highlighting that there is a *secret.txt* file present within the *top_secret.jpg* file.

13. Now, we are going to delete the secret.txt file we just created. Type `rm secret.txt` and press **Enter**. Then run `ls` to see that the *secret.txt* file was gone.

```
┌──(kali㉿kali)-[~/Desktop/steg]
└─$ rm secret.txt

┌──(kali㉿kali)-[~/Desktop/steg]
└─$ ls
cyber_monday.jpg  top_secret.jpg
```

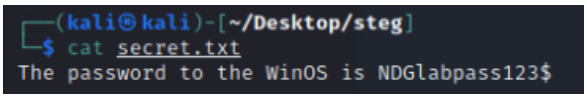
14. Attempt to extract the secret data within the *top_secret.jpg* image file. When prompted for the passphrase, type `secret` followed by pressing **Enter**. Now the secret.txt file is back.

```
kali@kali$ steghide extract -sf top_secret.jpg
```

```
┌──(kali㉿kali)-[~/Desktop/steg]
└─$ steghide extract -sf top_secret.jpg
Enter passphrase:
wrote extracted data to "secret.txt".
```


Copyright © 2024 Network Development Group, Inc. www.netdevgroup.com

15. Let's examine the content of the extracted file; type **`cat secret.txt`** and press **Enter**.



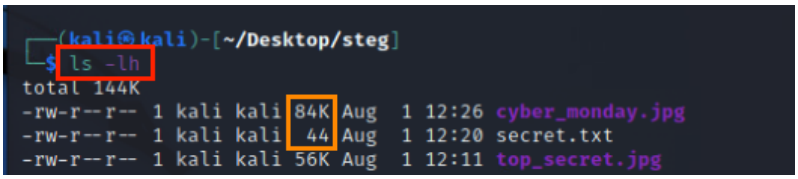
16. Leave the *Terminal* window open to continue with the next task.

Copyright © 2024 Network Development Group, Inc. www.netdevgroup.com

2 Hiding Multiple Files Within an Image File

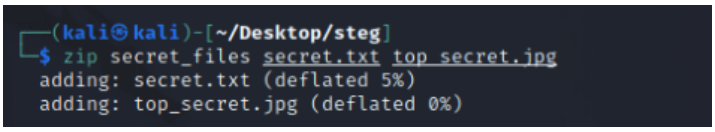
2.1 Using Basic Linux Commands to Hide Zipped Archives

1. While still on the *Kali* system, in the *Terminal* window, verify that you are still in the **~/Desktop/steg** directory. List the files in the current directory using the `ls -lh` command. Take note of the file sizes for both *cyber_monday.jpg* and *secret.txt*.

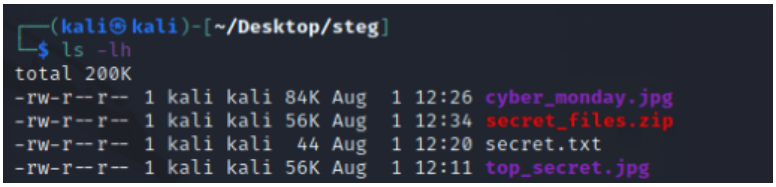


2. Create a *zipped* archive named **secret_files** to include the files: **secret.txt** and **dc.jpg**

```
kali@kali$ zip secret_files secret.txt top_secret.jpg
```

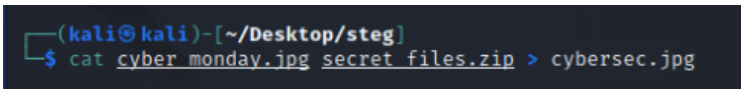


3. Enter `ls -lh` to list the current files in the directory and verify that *secret_files.zip* is present.

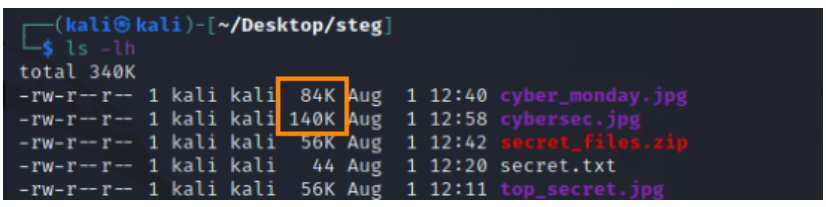


4. Using the **cat** command, enter the command below to hide the zipped archive within the image file called **cyber_monday.jpg**. This will enable the *cat* command to concatenate the image and zip file together in a new file named **cybersec.jpg.**

```
kali@kali$ cat cyber_monday.jpg secret_files.zip > cybersec.jpg
```



5. Enter the command `ls -lh` to list the files in the current directory and verify that the **cybersec.jpg** image file has been successfully created. Also, take note of the file size and compare it to the original **cyber_monday.jpg.**



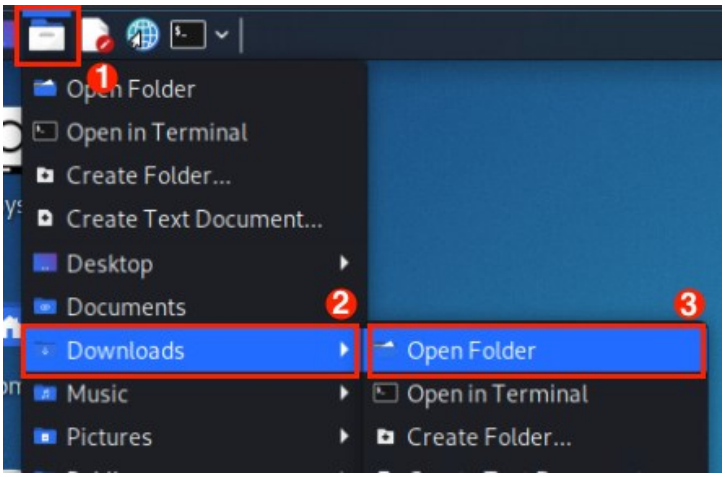
Copyright © 2024 Network Development Group, Inc. www.netdevgroup.com

3 Observe the Avalanche Effect in Hashing Operation

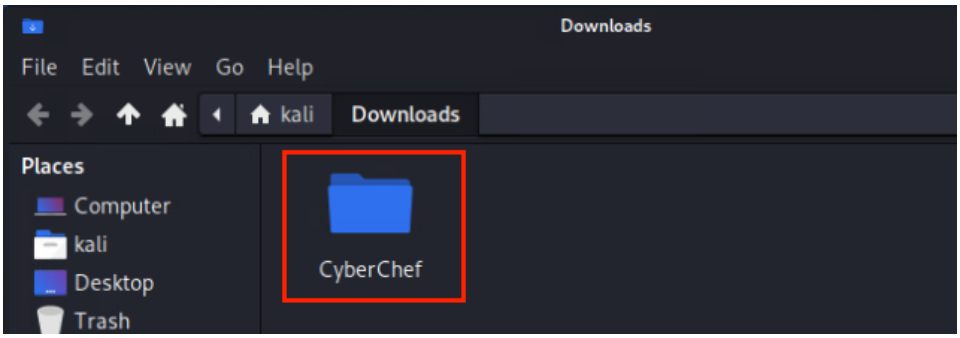
3.1 Observe the Avalanche Effect

The avalanche effect is observed in cryptography when one bit in the plain text flips; the encrypted text will have at least half of the bits flipped. In contrast, if the avalanche effect does not exist or is not so significant, it means the encryption algorithm has poor randomization. Or, in other words, it is less secure. In this section, we will observe the bit flips in the MD5 hash function in a popular tool called CyberChef.

1. First, let's go to the *Downloads* folder by clicking the **Folder** icon, then **Downloads**, then **Open Folder**.

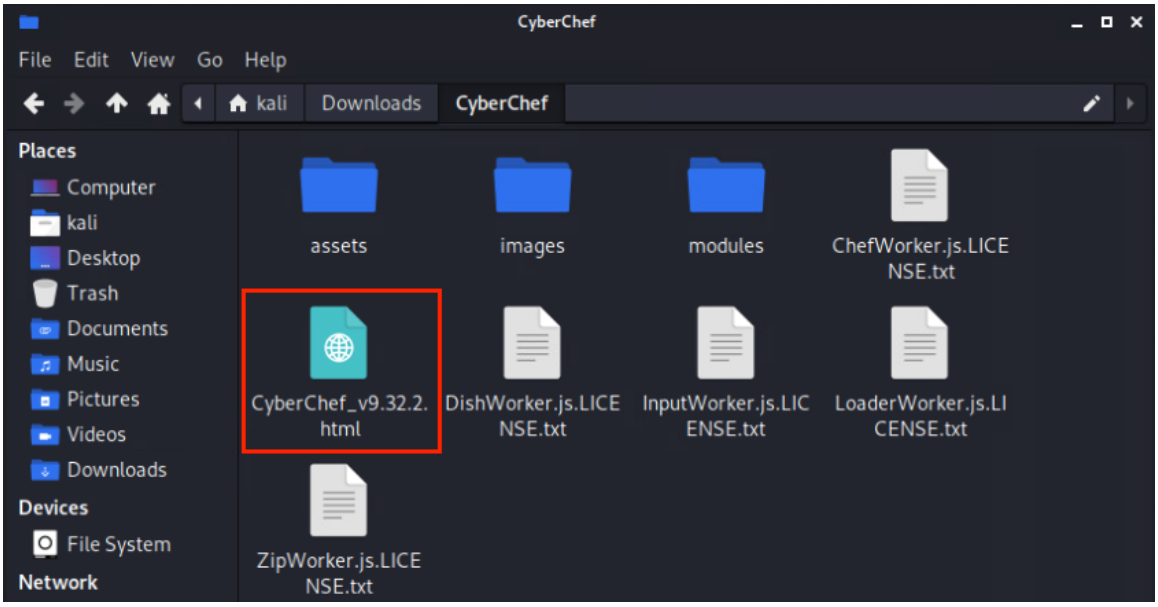


2. A window will open, showing a folder named *CyberChef*. Double-click to open the folder.



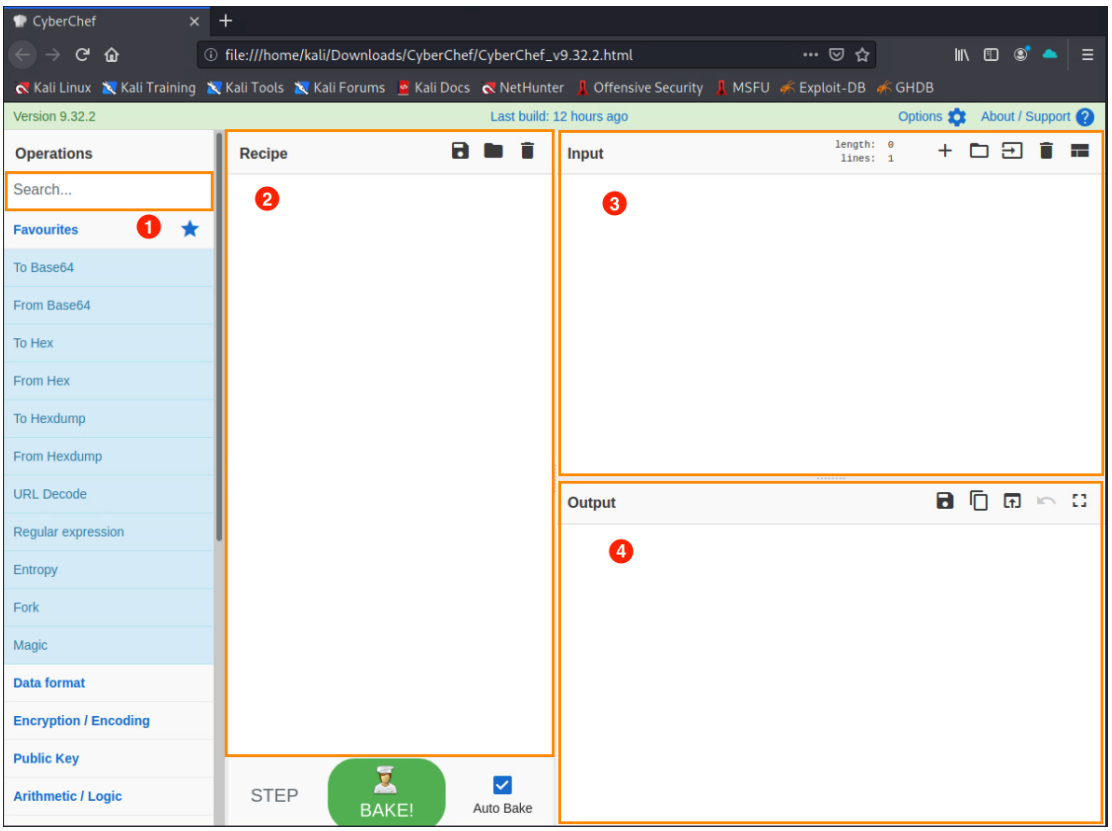
3. Inside the *CyberChef* folder, there is a *CyberChef_v9.32.2.html* file. Double-click to open it in the browser.

Copyright © 2024 Network Development Group, Inc. www.netdevgroup.com



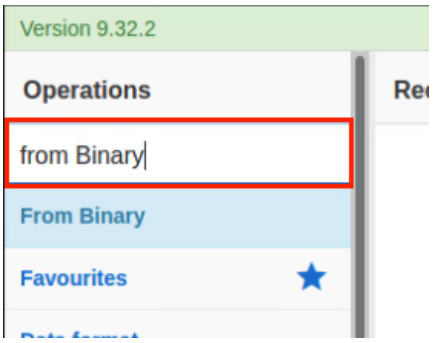
4. A browser window will open. You will now see the *CyberChef* main page.

On the *CyberChef* main page, number 1 indicates the recipe(function) Search field. Number 2 is the main operation field, where the recipe(function) can be added to apply the algorithms. Number 3 is the place that accepts user inputs. Number 4 will output the result to the user.

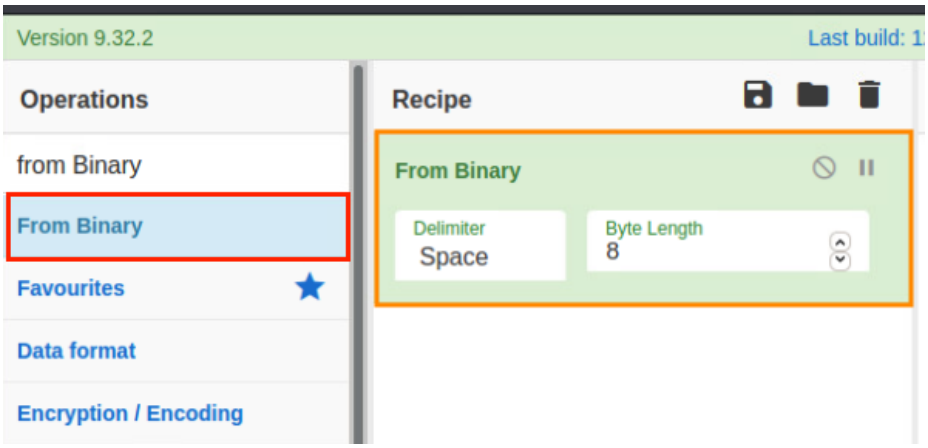


 Copyright © 2024 Network Development Group, Inc. www.netdevgroup.com

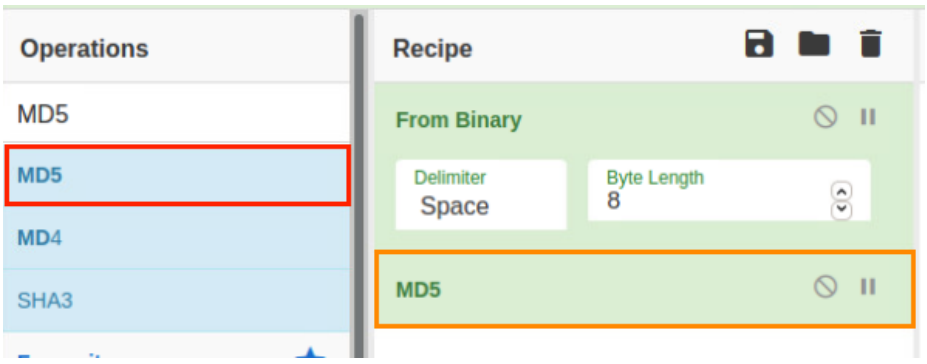
5. Now let's first observe the MD5 hash function. First, type `from Binary` in the search field.



6. In the search result list, double-click on **From Binary** to add it to the *Recipe*. We will leave the default setting.

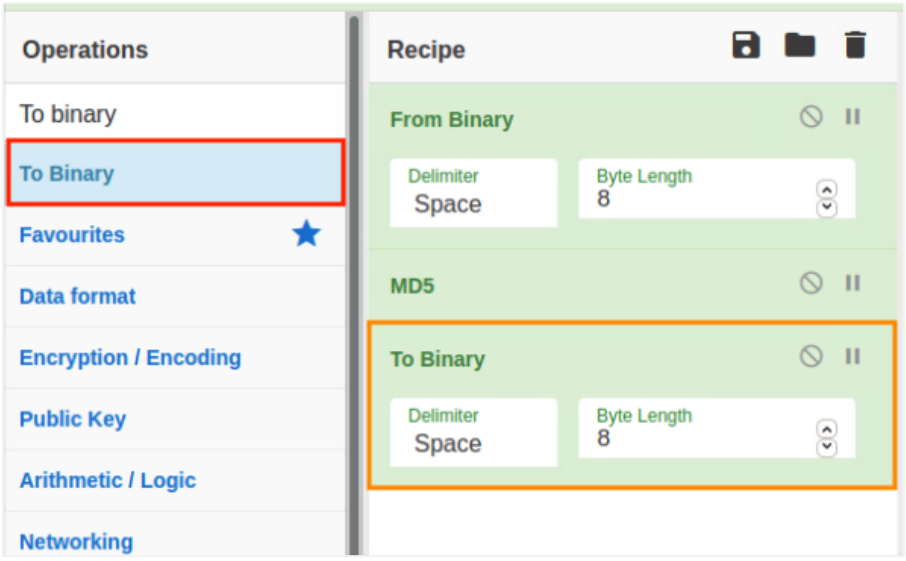


7. After adding the **From Binary** recipe(function), let's delete everything in the search field and type `MD5`. Same as before, double-click to add the **MD5** recipe(function).



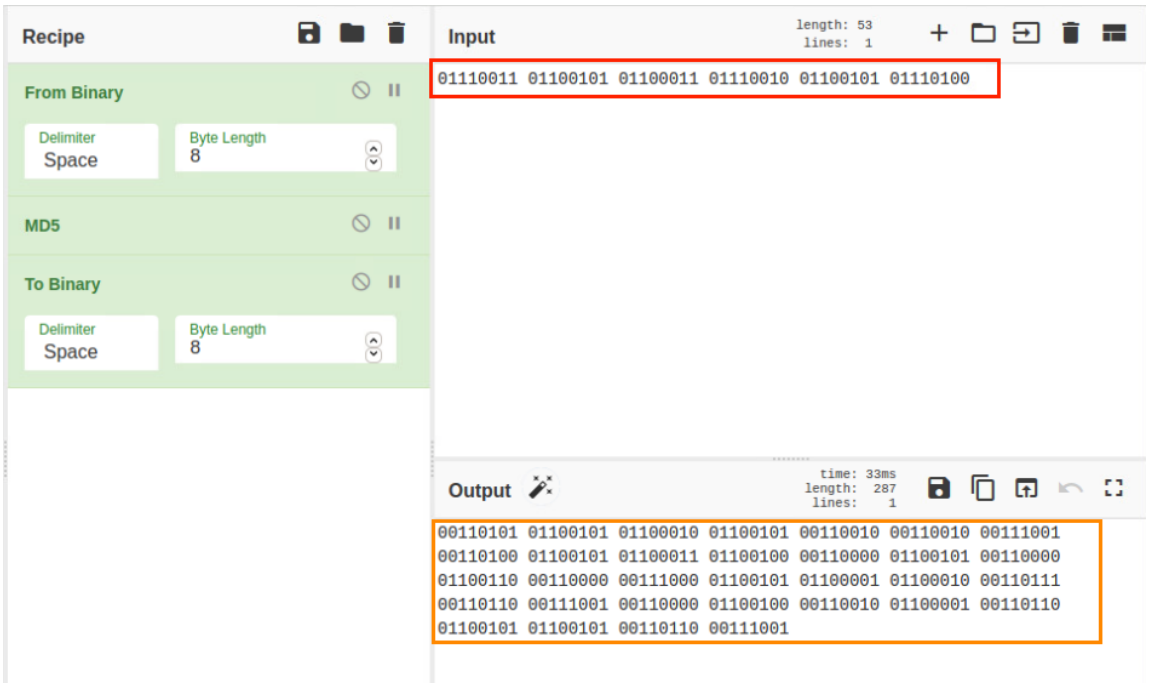
Copyright © 2024 Network Development Group, Inc. www.netdevgroup.com

8. Once again, go to the search field, type **`To Binary`**, double-click to add to the *Recipe*. Leave the default setting.



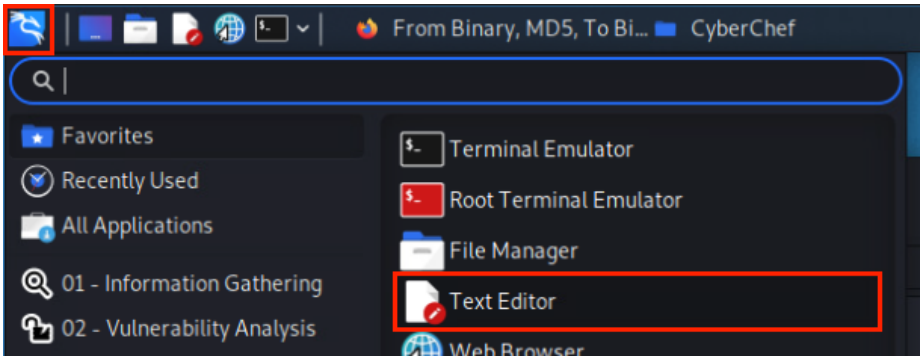
9. To the right side, we will now enter the binary form of the word *secret*; make sure you include the spaces:

```
01110011 01100101 01100011 01110010 01100101 01110100
```

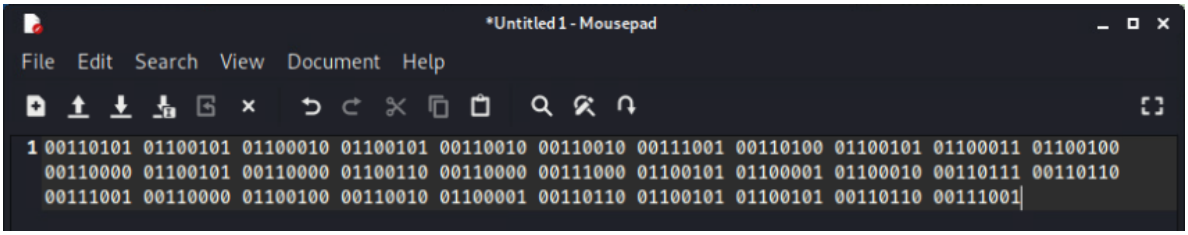


Copyright © 2024 Network Development Group, Inc. www.netdevgroup.com

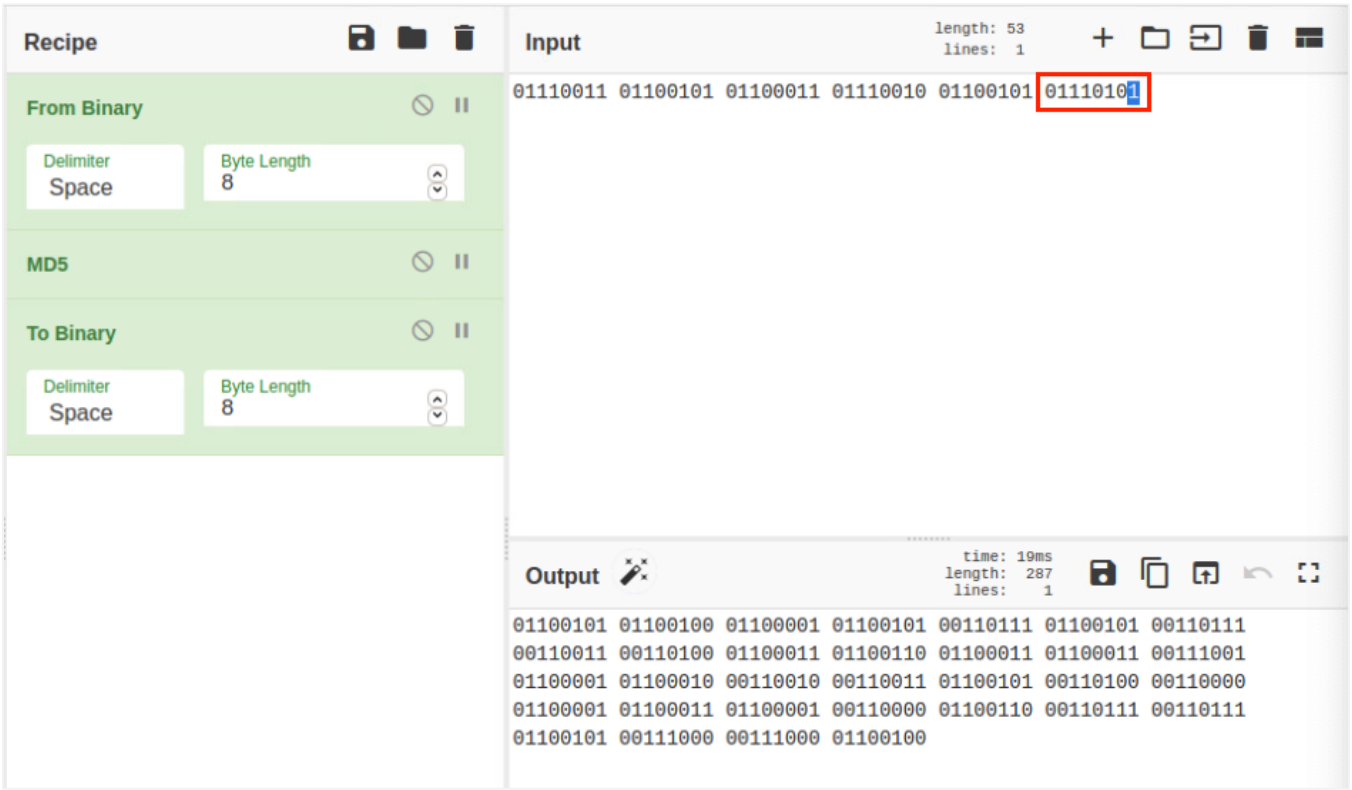
10. As you type the binary in, the Output hash changes. Let's start a *Text Editor*. Click the **Applications** icon, and select to open the **Text Editor.**



11. Go back to the browser window. Copy the output and paste it to the *Text Editor*.

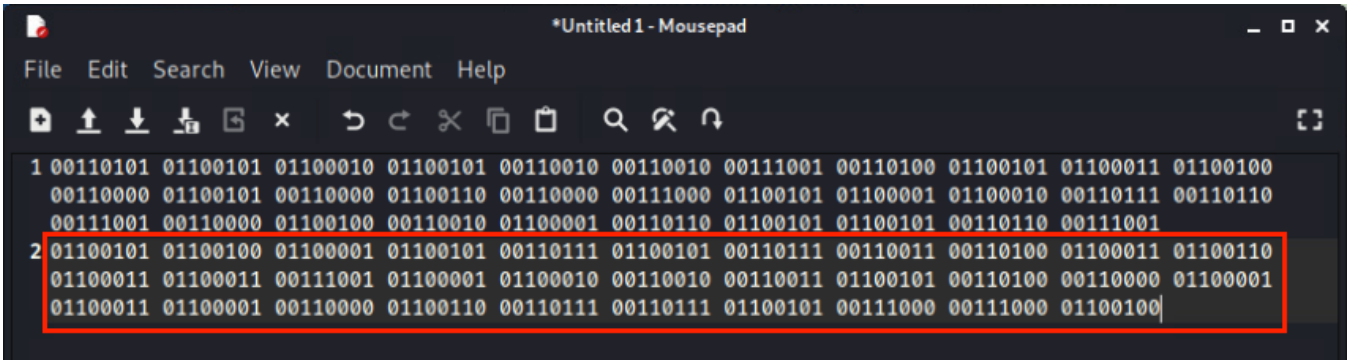


12. With the output stored, let's go back to the browser window. We are going to flip one bit in the input and observe the bit changes in the output.
13. In the input field, change the rightmost bit from 0 to 1. Leave everything else untouched.

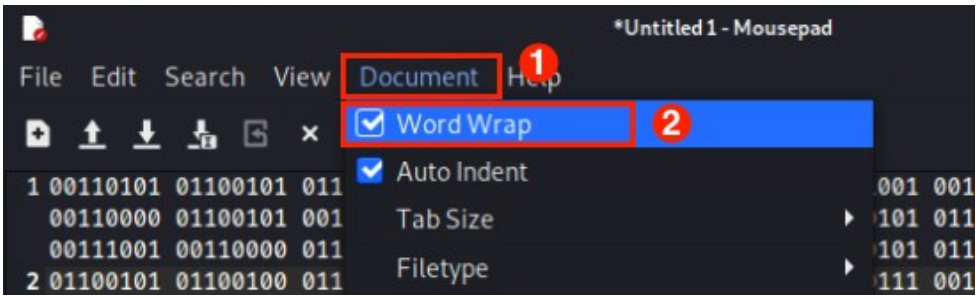


 Copyright © 2024 Network Development Group, Inc. www.netdevgroup.com

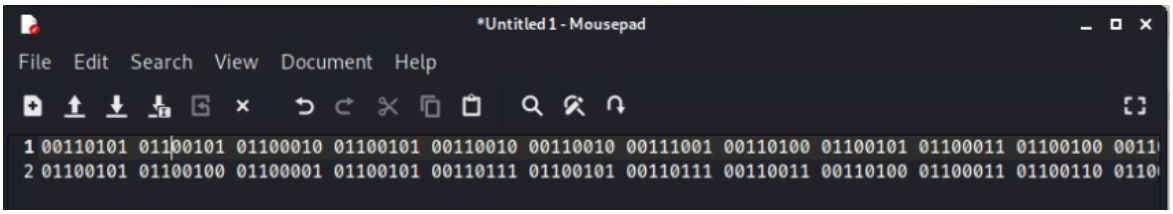
14. To compare the bit flips in the output, let's first copy the output. Then switch to the *Text Editor* window. Press **Enter** to start a new line. Then paste the copied output.



15. To make the comparison easier, in the menu, click **Document**, then uncheck **Word Wrap**.



16. The two lines of binary numbers are now aligned. The top is the original output; the bottom is the one-bit flipped output. Drag the horizontal scroll bar to observe the result and count the bit flips.



17. The same process can also be used to test the SHA1 hash function. Feel free to do so to observe the difference.
18. The lab is now complete; you may end the reservation.

Copyright © 2024 Network Development Group, Inc. www.netdevgroup.com