ETHICAL HACKING V2
LAB SERIES

Lab 23: Covering Your Tracks

Document Version: 2021-05-18

Material in this Lab Aligns to the Following	
Books/Certifications	**Chapters/Modules/Objectives**
All-In-One CEH Chapters ISBN-13: 978-1260454550	4: Sniffing and Evasion 11: Cryptography 101
EC-Council CEH v10 Domain Modules	12: Evading IDS, Firewalls, and Honeypot 20: Cryptography

Contents

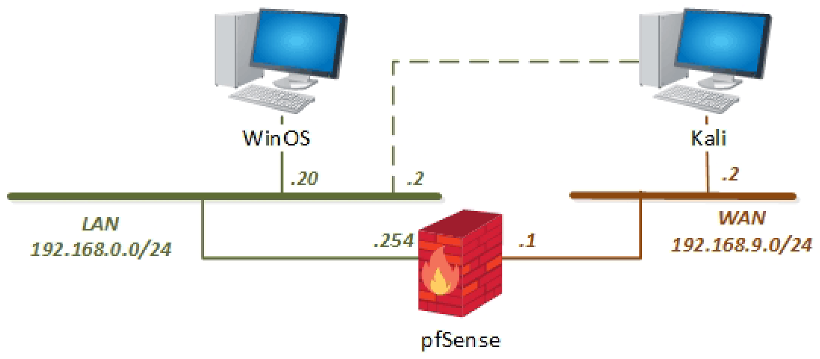
Introduction

A stream consists of data associated with a main file or directory. Each file and directory in NTFS can have multiple streams that are generally hidden from a user. Once the hacker has compromised the system, installed their trojan, port redirection, backdoors, and obtained all information from the targeted system, they will move to other targets on the network. Expert ethical hackers and penetration testers must understand how data hiding is done using NTFS streams.

Objectives

- Using NTFS Streams
- Finding hidden files ADS Spy
- Extracting hidden files using OpenStego
- Extending Linux File Attributes to hide data
- Covering tracks by deleting Linux logs

Lab Topology



Lab Settings

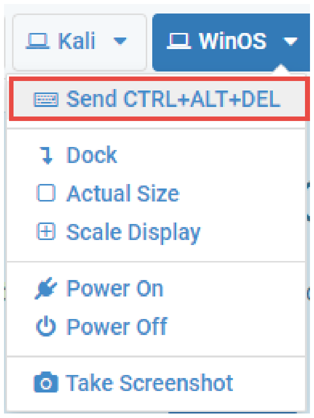
The information in the table below will be needed to complete the lab. The task sections below provide details on the use of this information.

Virtual Machine	**IP Address / Subnet Mask**	**Account** (if needed)	**Password** (if needed)
WinOS	192.168.0.20	Administrator	Train1ng$
Kali Linux	192.168.9.2 192.168.0.2	root	toor

1 Covering Your Tracks - NTFS Streams

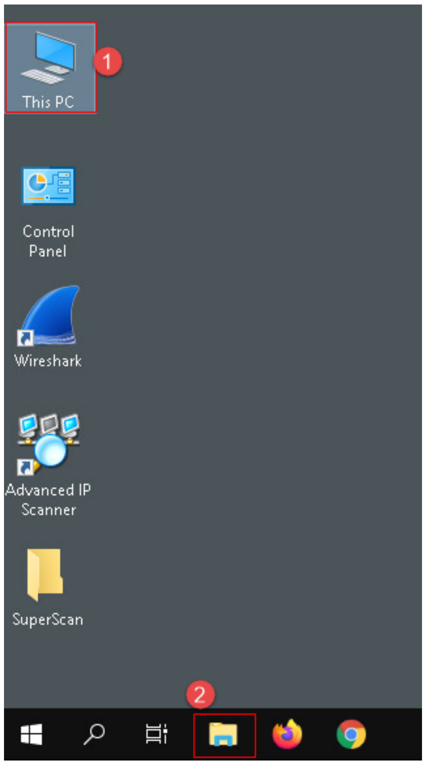
A New Technology File System (NTFS) can have multiple data streams that are generally hidden from the user. It is very effective in hiding data in plain sight. A common use is to transmit illicit files and communicate secretly. In this exercise, we will teach you how NTFS stream files are created and embedded within another file.

1. Launch the **WinOS** virtual machine to access the graphical login screen.
   1.1. Select **Send CTRL+ALT+DEL** from the dropdown menu to be prompted with the login screen.

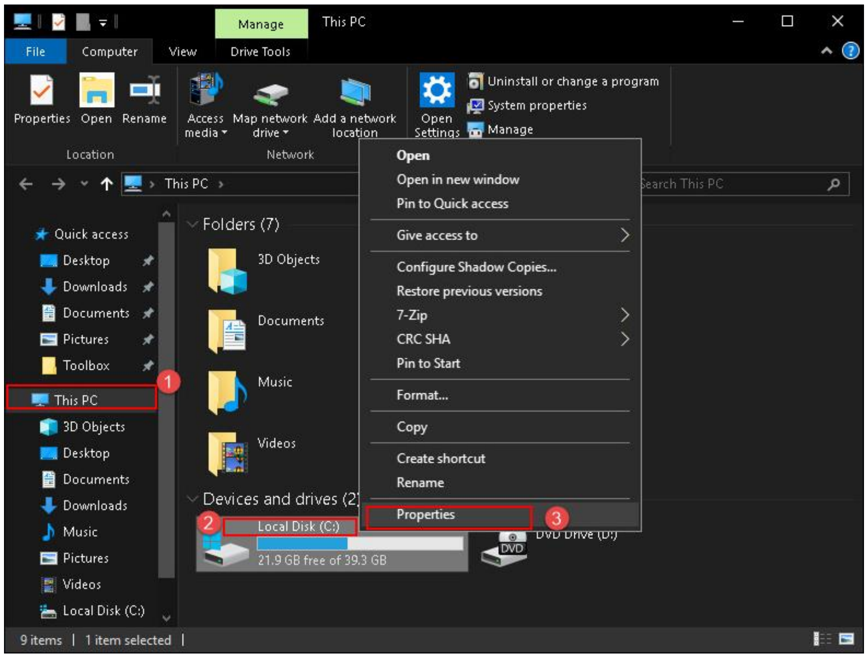


   1.2. Log in as `Administrator` using the password: `Train1ng$`

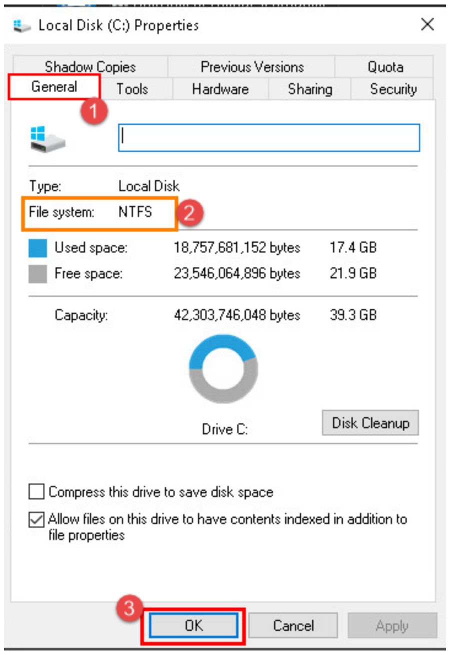
2. Let us begin by looking at the properties of the hard drive. Most Windows Operating System's configured hard drives will always be formatted to NTFS; however, for the purposes of the exercise, we will confirm the file system by double-clicking on **This PC** or opening *Windows File Explorer,* as seen in *items* **1** and **2** below.



3. In *File Explorer*, select **This PC** as highlighted in *item* ***1***. There you will see **Local Disk (C:)** under *Devices and Drives*. Right-click the drive to open the context menu, then navigate to **Properties** as seen in *items* **2** and **3** below.



4. The **Local Disk (C:) Properties** window should appear. Select the **General tab** to locate the file system, then click **OK** to close the properties as seen highlighted at *items* **1, 2,** and **3** below.

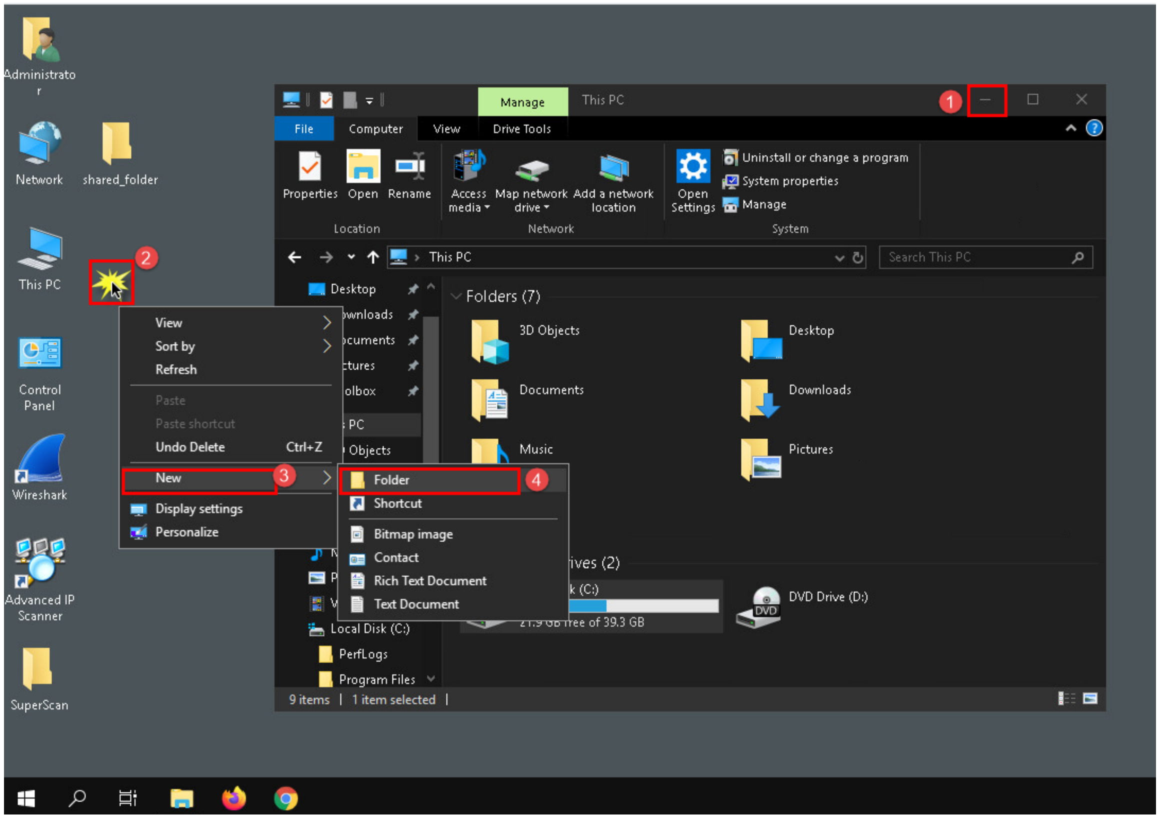


Type: Local Disk
File system: NTFS

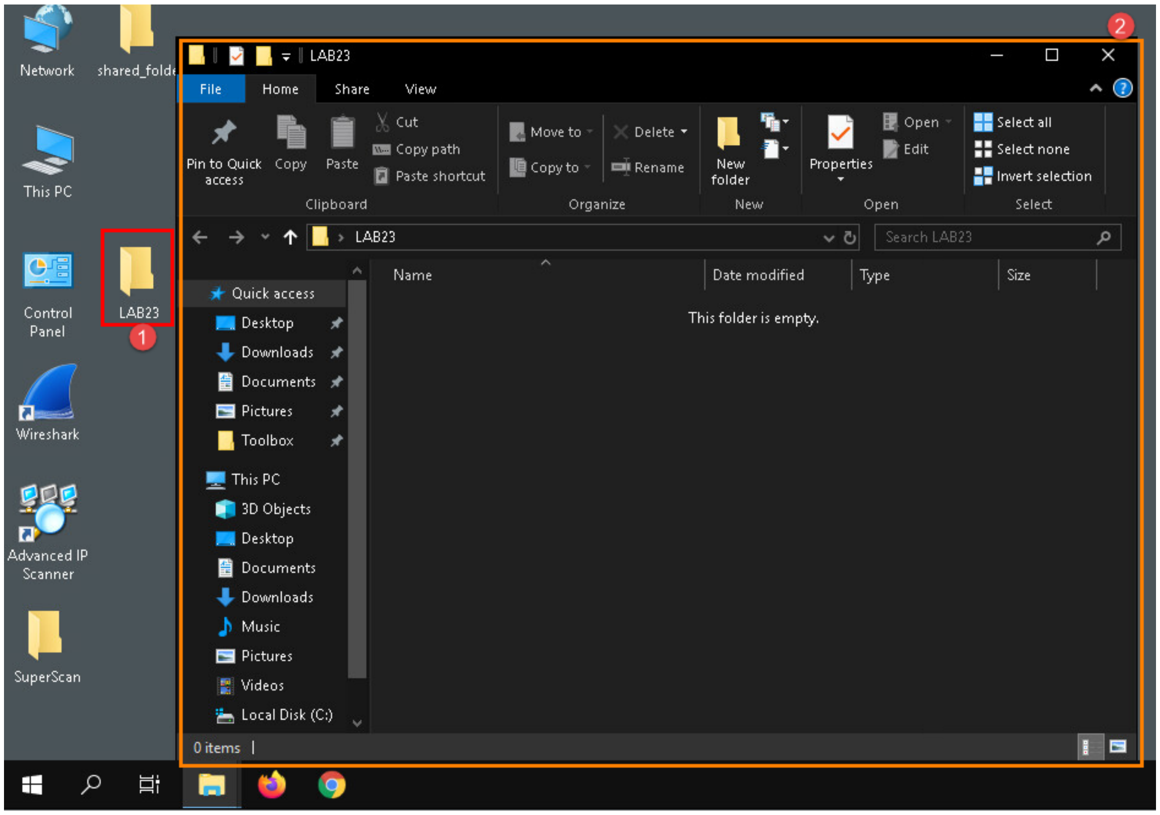
🟦	Used space:	18,757,681,152 bytes	17.4 GB
⬜	Free space:	23,546,064,896 bytes	21.9 GB
	Capacity:	42,303,746,048 bytes	39.3 GB

☐ Compress this drive to save disk space
☑ Allow files on this drive to have contents indexed in addition to file properties

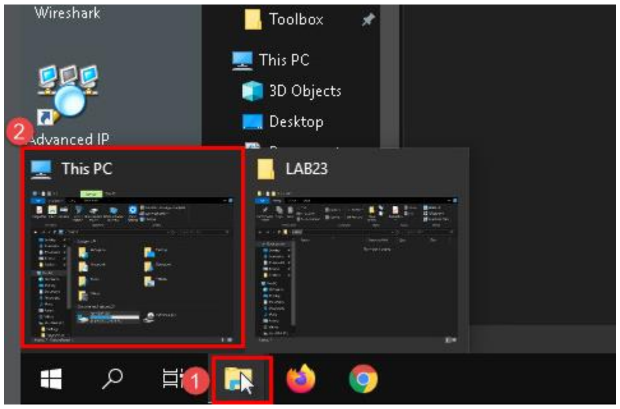
5. Minimize **Windows File Explorer** to view the **Desktop,** then right-click any empty spot to open a context menu, navigate to **New,** and select **Folder** to create a new folder on the Desktop, as highlighted in *items* **1**, **2**, **3,** and **4**.



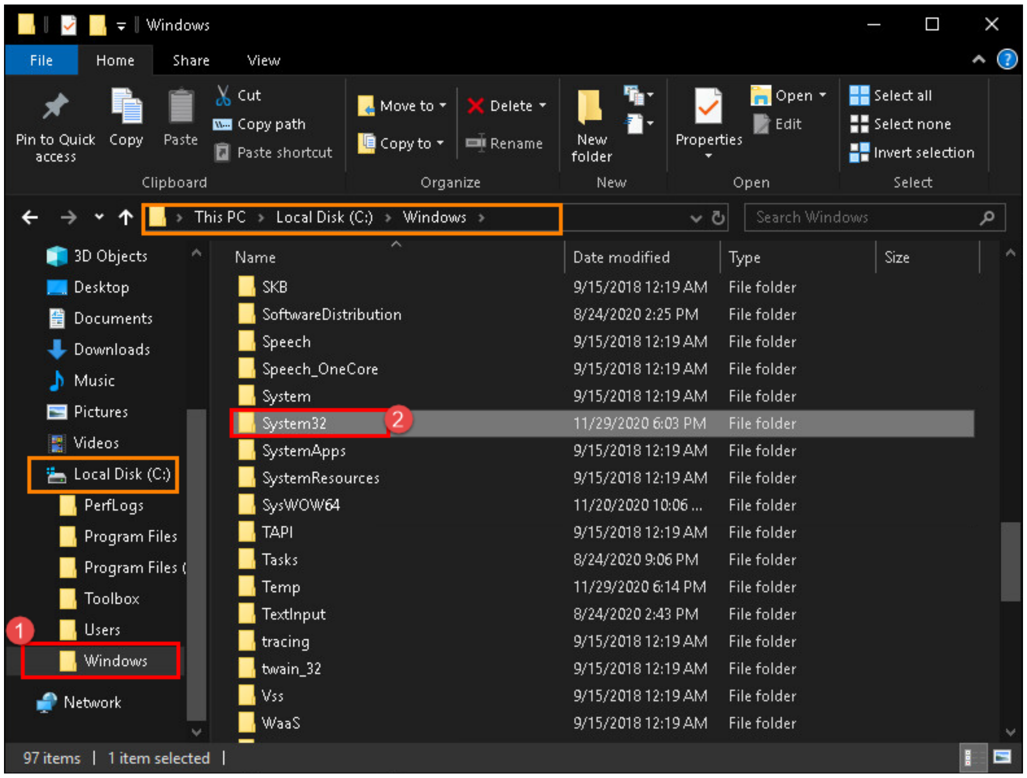
6. Name the newly created folder **LAB23** and double-click the folder to open it, as seen in *items* **1** and **2** below.



7. *Windows File Explorer* should still be open. Use the cursor and hover over **Windows File Explorer**; this will display both currently active windows. Select the window titled **This PC,** as seen in *items* **1** and **2** below.

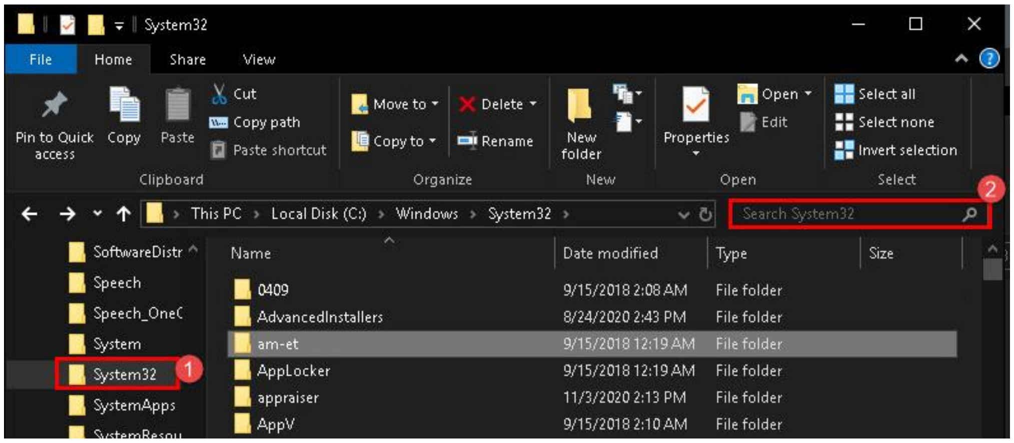


8. Now double-click **Local Disk (C:)** then navigate to the path **C:\Windows\System32** as seen in *items* **1** and **2**.



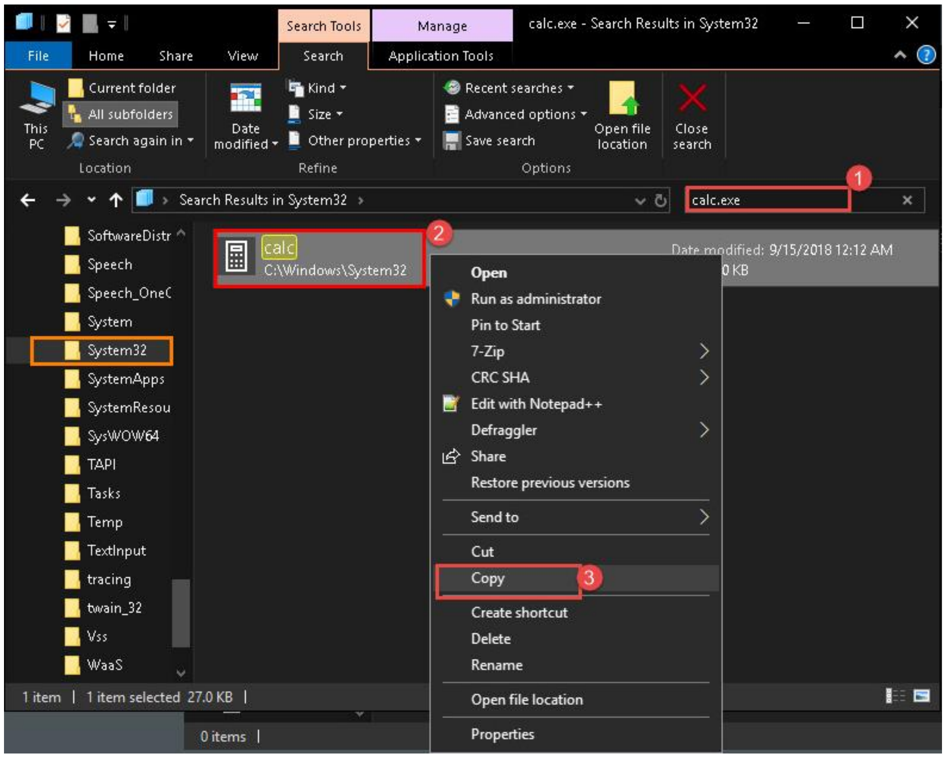
Name	Date modified	Type	Size
SKB	9/15/2018 12:19 AM	File folder	
SoftwareDistribution	8/24/2020 2:25 PM	File folder	
Speech	9/15/2018 12:19 AM	File folder	
Speech_OneCore	9/15/2018 12:19 AM	File folder	
System	9/15/2018 12:19 AM	File folder	
System32	11/29/2020 6:03 PM	File folder	
SystemApps	9/15/2018 12:19 AM	File folder	
SystemResources	9/15/2018 12:19 AM	File folder	
SysWOW64	11/20/2020 10:06 ...	File folder	
TAPI	9/15/2018 12:19 AM	File folder	
Tasks	8/24/2020 9:06 PM	File folder	
Temp	11/29/2020 6:14 PM	File folder	
TextInput	8/24/2020 2:43 PM	File folder	
tracing	9/15/2018 12:19 AM	File folder	
twain_32	9/15/2018 12:19 AM	File folder	
Vss	9/15/2018 12:19 AM	File folder	
WaaS	9/15/2018 12:19 AM	File folder	

9. Double-click the folder called ***System32,*** then select the search bar called **Search System32** located at the top-right side of the window, as seen in *items* **1** and **2** below.

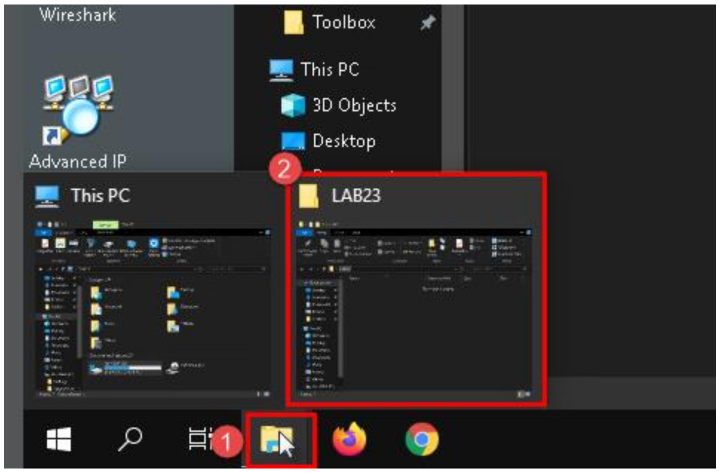


Name	Date modified	Type	Size
0409	9/15/2018 2:08 AM	File folder	
AdvancedInstallers	8/24/2020 2:43 PM	File folder	
am-et	9/15/2018 12:19 AM	File folder	
AppLocker	9/15/2018 12:19 AM	File folder	
appraiser	11/3/2020 2:13 PM	File folder	
AppV	9/15/2018 2:10 AM	File folder	

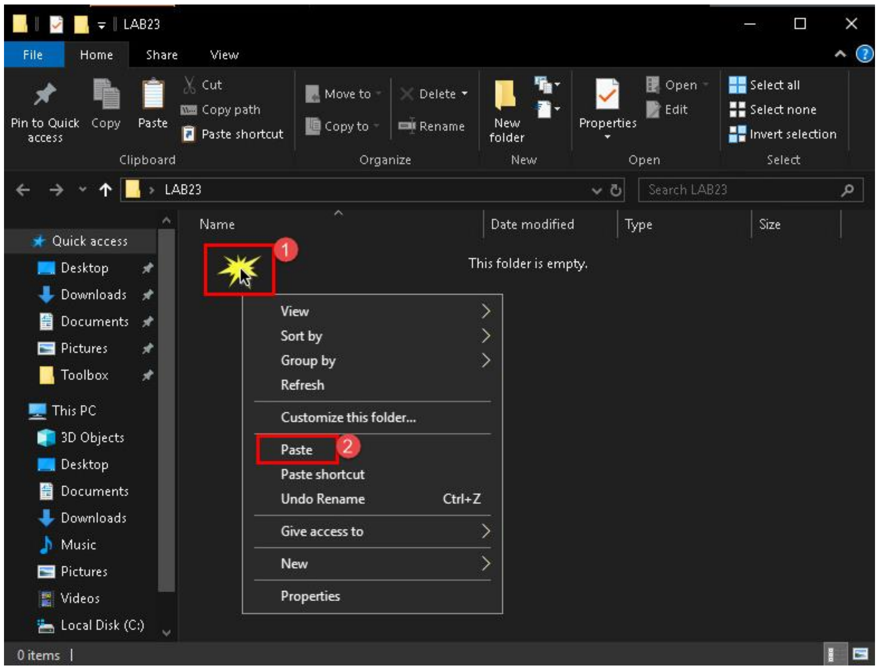
10. Type `calc.exe` in the search bar to filter the results and locate the *Calculator* executable. Click the file called **calc** to select it, then right-click on it to open a context menu and then select **Copy** as highlighted in *items* **1, 2,** and **3** below.



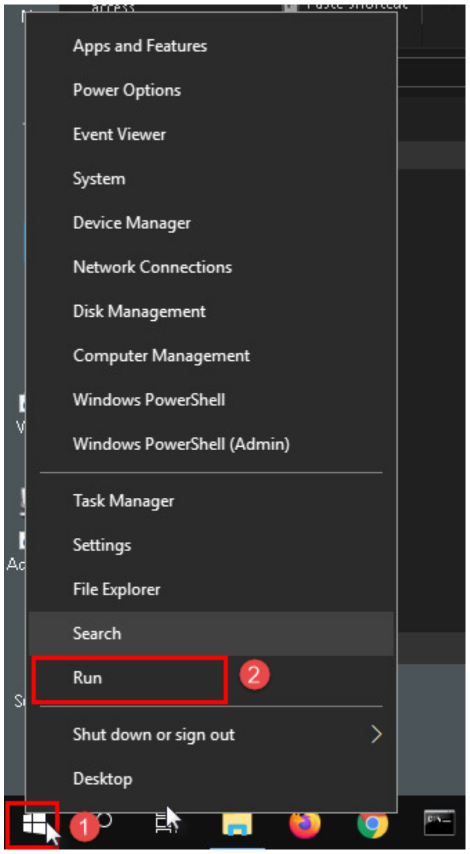
11. Use the cursor and hover over **Windows File Explorer** again. Select the window titled **LAB23** as seen in *items* **1** and **2** below.



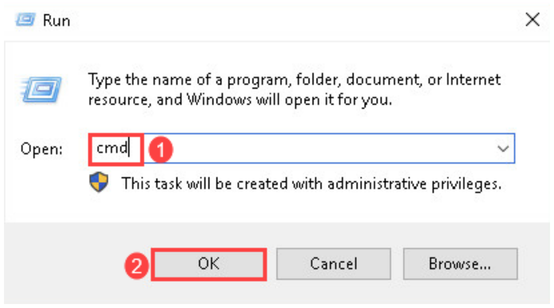
12. Now right-click inside the empty folder to open the context menu, navigate to and select **Paste** to copy the file called *calc.exe* to the ***LAB23*** folder, as seen in *items* **1** and **2**.



13. Next, right-click on the **Start** button and click the **Run** option from the context menu that appears, as seen in *items* **1** and **2** below.

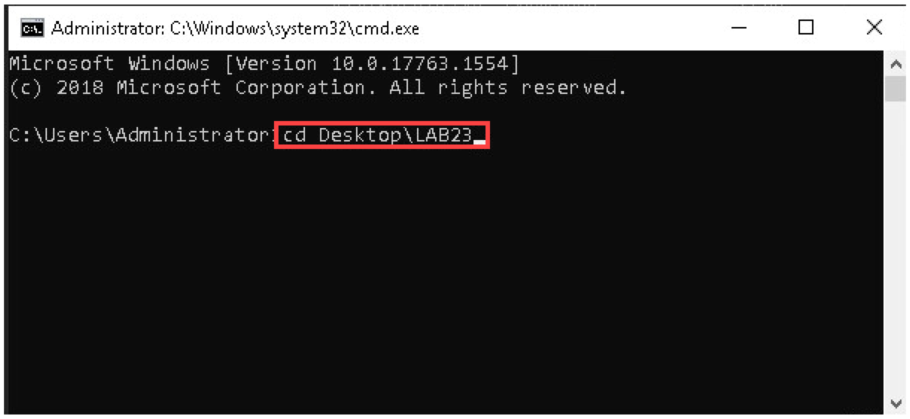


14. When the *Run* window appears, type `cmd` and click **OK,** as seen in *items* **1** and **2** below. This will launch the *Windows command prompt* with Administrative privileges.


Run
Type the name of a program, folder, document, or Internet resource, and Windows will open it for you.
Open:
cmd
1
This task will be created with administrative privileges.
2
OK
Cancel
Browse...

15. Once the command line window appears, we will change the directory to the path where the files we want to manipulate are located. First, let us change to the **Desktop** and then to the folder **LAB23** by typing the following command:

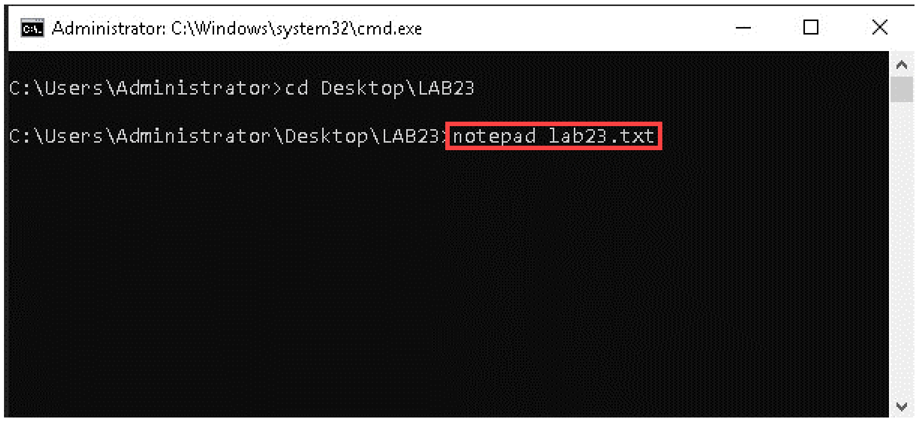
```
C:\Users\Administrator>cd Desktop\LAB23
```



```
Microsoft Windows [Version 10.0.17763.1554]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\Administrator>cd Desktop\LAB23_
```

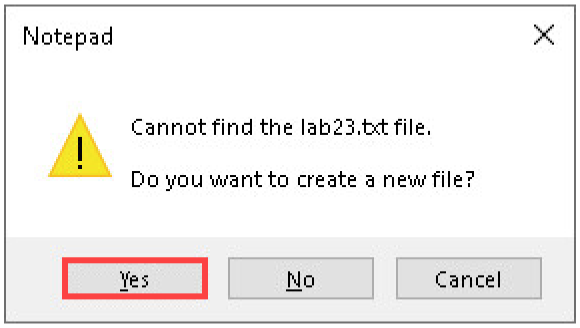
16. Now let us create a new text file by typing `notepad lab23.txt` and press **Enter**.

```
C:\Users\Administrator\Desktop\LAB23>notepad lab23.txt
```



```
C:\Users\Administrator>cd Desktop\LAB23

C:\Users\Administrator\Desktop\LAB23>notepad lab23.txt
```

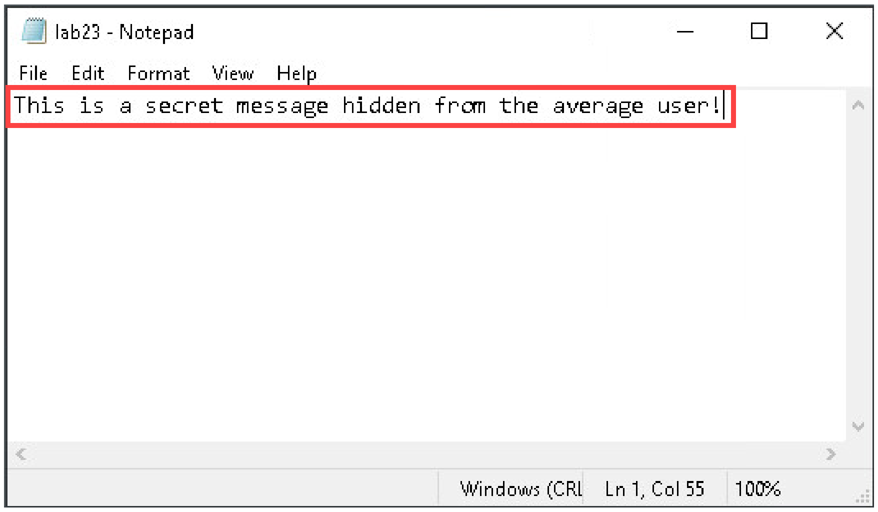
17. If you typed the command correctly, *Notepad* should launch; click the **Yes** button when prompted to create the new *lab23.txt* file, as highlighted below.


Notepad

Cannot find the lab23.txt file.

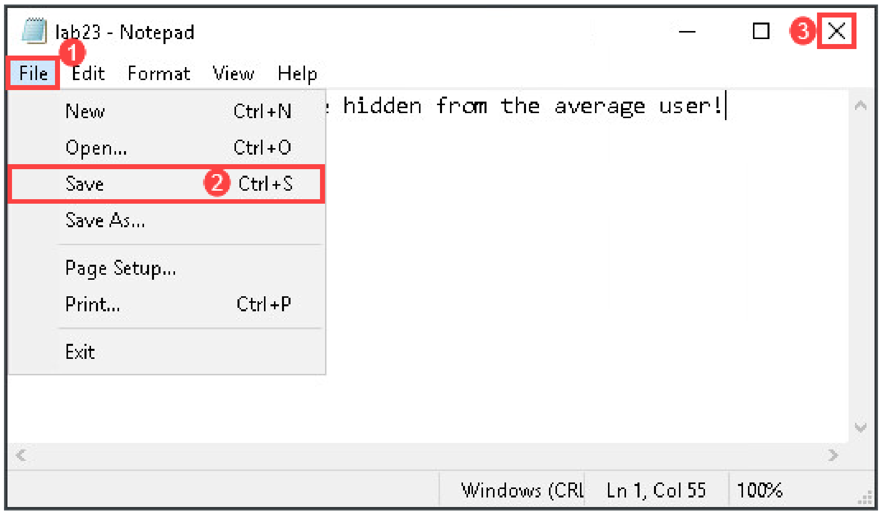
Do you want to create a new file?

18. In the newly created text file, type `This is a secret message hidden from the average user!` as seen below.



This is a secret message hidden from the average user!

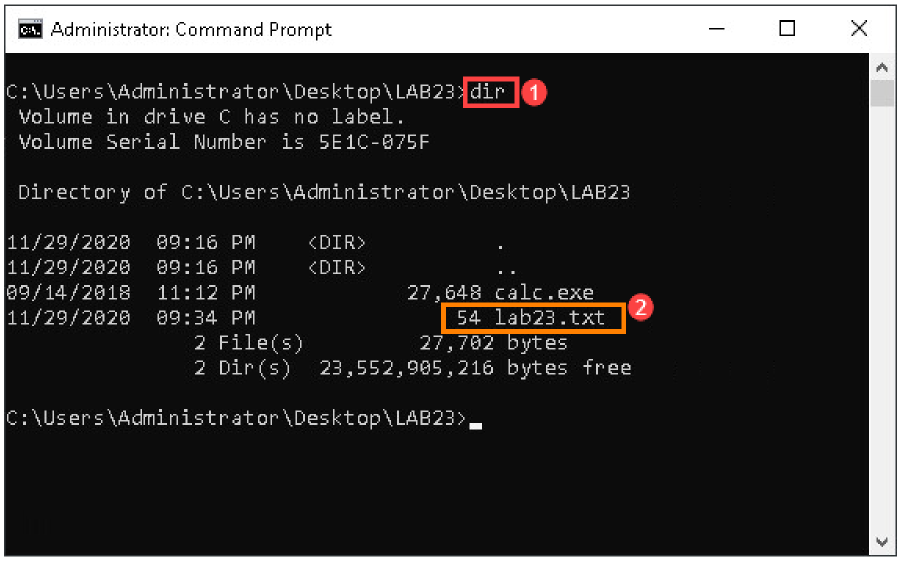
19. Now, navigate to the main menu and select **File** > **Save** to save the **lab23.txt** notepad file, then close *Notepad* by clicking the **X** in the top-right corner as seen in *items* **1, 2,** and **3** below.



: hidden from the average user!

20. Let us go back to the *Command Prompt* and type `dir` and press **Enter** to list all present files in the *LAB23* folder. See the command in *item* **1** in the screenshot below. This should reveal the executable file called *calc.exe* and the text file called *lab23.txt,* as seen in *item* **2**. We will use the NTFS Stream, Alternative Data Stream (ADS), to hide the application called ***calc.exe*** into the text file called ***lab23.txt***.

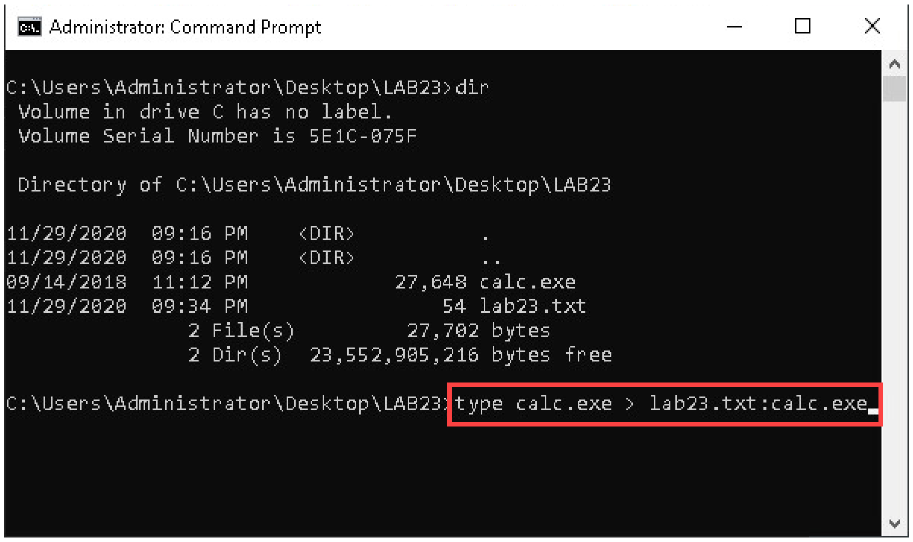
```
C:\Users\Administrator\Desktop\LAB23>dir
```



```
C:\Users\Administrator\Desktop\LAB23>dir
 Volume in drive C has no label.
 Volume Serial Number is 5E1C-075F

 Directory of C:\Users\Administrator\Desktop\LAB23

11/29/2020  09:16 PM    <DIR>          .
11/29/2020  09:16 PM    <DIR>          ..
09/14/2018  11:12 PM            27,648 calc.exe
11/29/2020  09:34 PM                54 lab23.txt
               2 File(s)         27,702 bytes
               2 Dir(s)  23,552,905,216 bytes free

C:\Users\Administrator\Desktop\LAB23>
```

21. Now, hide **calc.exe** inside the **lab23.txt** file by typing the following at the command prompt: `type calc.exe > lab23.txt:calc.exe` then press **Enter**.

```
C:\Users\Administrator\Desktop\LAB23>type calc.exe > lab23.txt:calc.exe
```

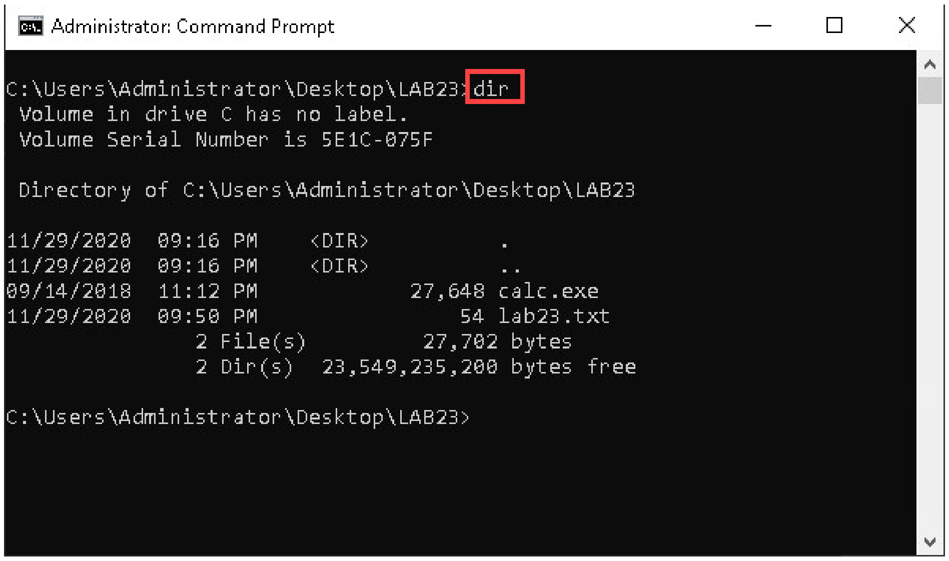


```
C:\Users\Administrator\Desktop\LAB23>dir
 Volume in drive C has no label.
 Volume Serial Number is 5E1C-075F

 Directory of C:\Users\Administrator\Desktop\LAB23

11/29/2020  09:16 PM    <DIR>          .
11/29/2020  09:16 PM    <DIR>          ..
09/14/2018  11:12 PM            27,648 calc.exe
11/29/2020  09:34 PM                54 lab23.txt
               2 File(s)         27,702 bytes
               2 Dir(s)  23,552,905,216 bytes free

C:\Users\Administrator\Desktop\LAB23>type calc.exe > lab23.txt:calc.exe
```

It is important to point out that you can damage files by doing this. The exercise should not be performed on live host systems without written permission, it can severely compromise the authenticity of a file, hence the reason a copy of the file was made.

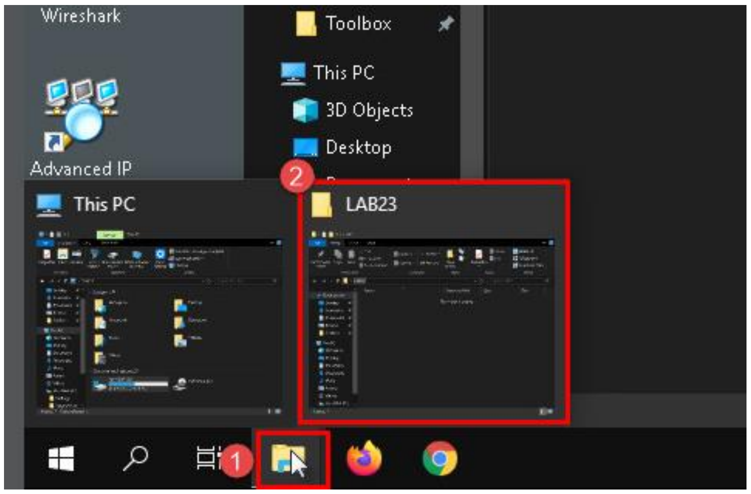
22. Type `dir` and press **Enter.** Note the file size of *lab23.txt*, which should *have no visible change.*

```
C:\Users\Administrator\Desktop\LAB23>dir
```

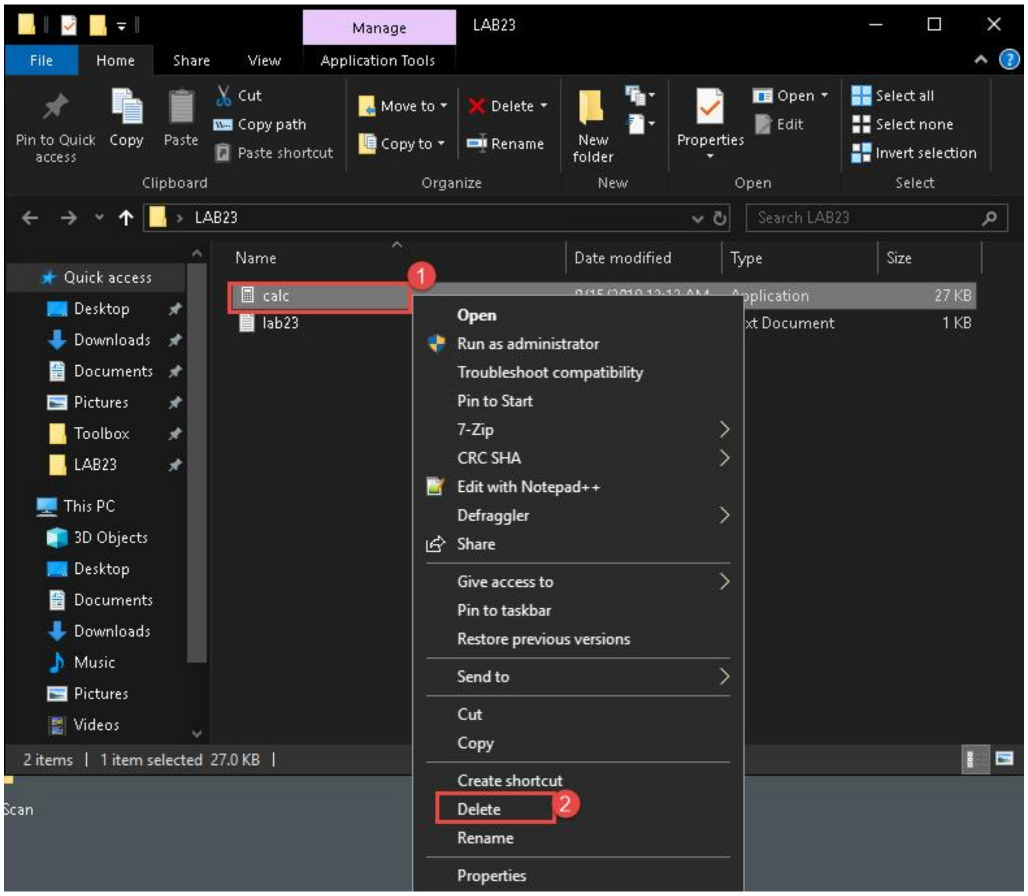


```
C:\Users\Administrator\Desktop\LAB23>dir
 Volume in drive C has no label.
 Volume Serial Number is 5E1C-075F

 Directory of C:\Users\Administrator\Desktop\LAB23

11/29/2020  09:16 PM    <DIR>          .
11/29/2020  09:16 PM    <DIR>          ..
09/14/2018  11:12 PM            27,648 calc.exe
11/29/2020  09:50 PM                54 lab23.txt
               2 File(s)         27,702 bytes
               2 Dir(s)  23,549,235,200 bytes free

C:\Users\Administrator\Desktop\LAB23>
```

23. Let us switch back. We will use the cursor and hover over **Windows File Explorer** again and select the window titled *LAB23,* as seen in *items* **1** and **2** below.

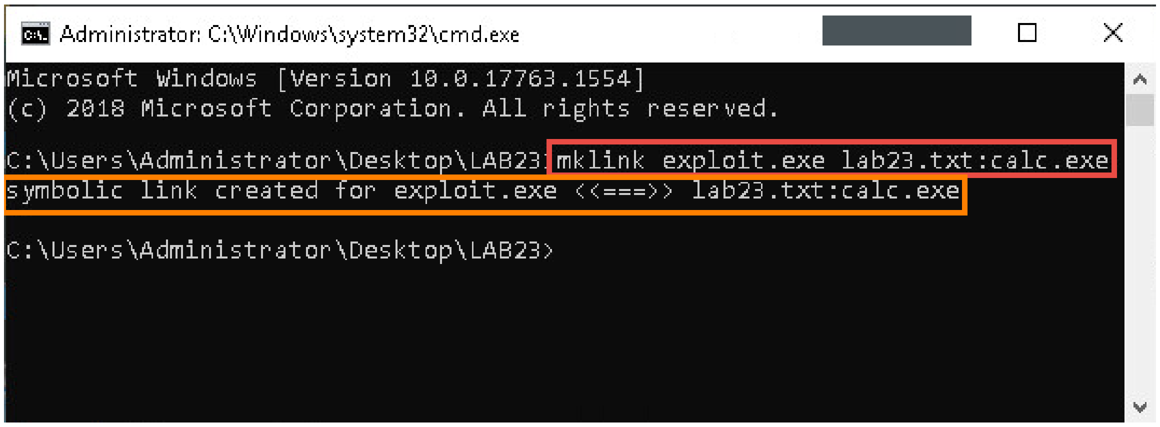


24. Navigate to the file **calc.exe**, right-click and select **Delete** as seen in *items* **1** and **2** below.



25. Go back to the *Command Prompt* and type `mklink exploit.exe lab23.txt:calc.exe` and press **Enter.**

```
C:\Users\Administrator\Desktop\LAB23>mklink exploit.exe lab23.txt:calc.exe
```

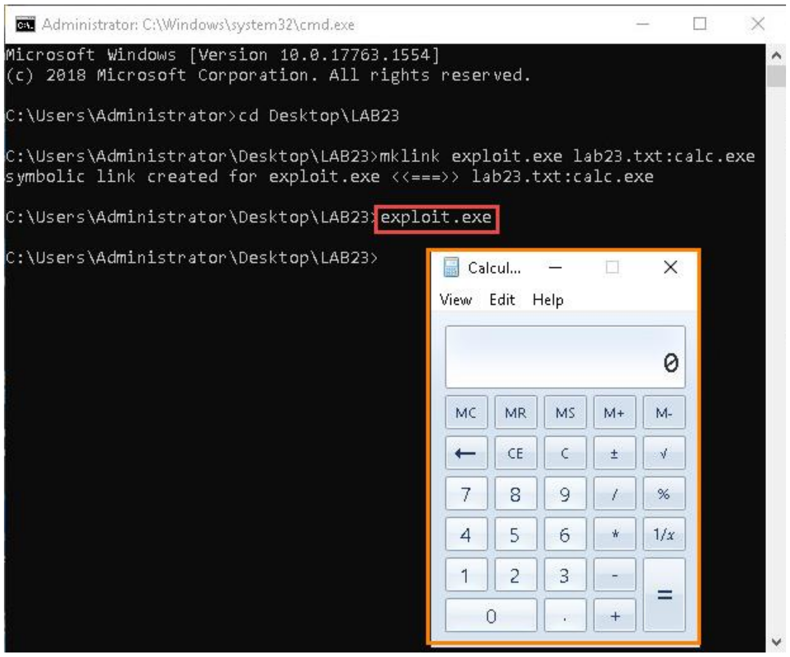


```
Microsoft Windows [Version 10.0.17763.1554]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\Administrator\Desktop\LAB23>mklink exploit.exe lab23.txt:calc.exe
symbolic link created for exploit.exe <<===>> lab23.txt:calc.exe

C:\Users\Administrator\Desktop\LAB23>
```

26. Now, let us say the system was compromised by a hacker. The hacker hid a malicious exploit within the *lab23.txt* text file in order to execute it on the target system later. By using the NTFS streams, the file was invisible to the legitimate user(s) of the computer. The hacker can now execute the malicious file as required.

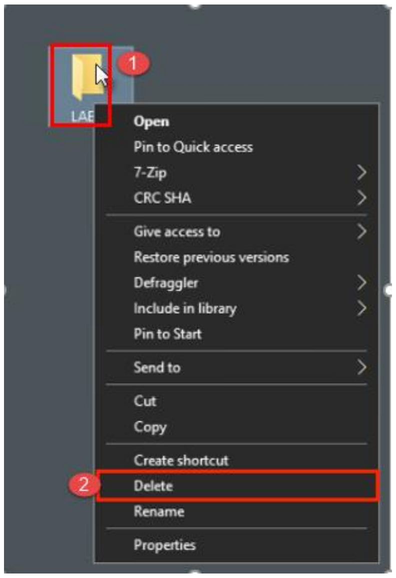
27. Type `exploit.exe` and press **Enter**. The *Calculator* application will be executed, and the attacker would have successfully launched the malicious executable.

```
C:\Users\Administrator\Desktop\LAB23>exploit.exe
```

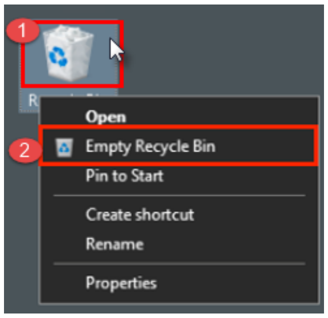


```
Microsoft Windows [Version 10.0.17763.1554]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\Administrator>cd Desktop\LAB23

C:\Users\Administrator\Desktop\LAB23>mklink exploit.exe lab23.txt:calc.exe
symbolic link created for exploit.exe <<===>> lab23.txt:calc.exe

C:\Users\Administrator\Desktop\LAB23>exploit.exe

C:\Users\Administrator\Desktop\LAB23>
```

28. You may close all opened windows now. Before you move to the next part, please delete the *LAB23* folder and its contents by right-clicking the folder and selecting **Delete** as seen in *items* **1** and **2** below.



29. Then, navigate to the Recycle Bin and right-click on it, and select **Empty Recycle Bin** as seen in *items* **1** and **2**.



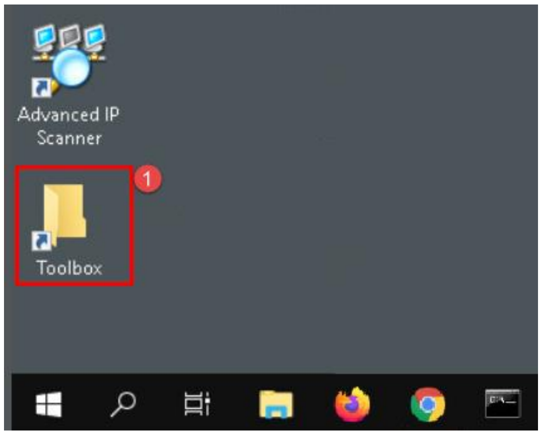
Click **Yes** if prompted.

30. This exercise is now complete.

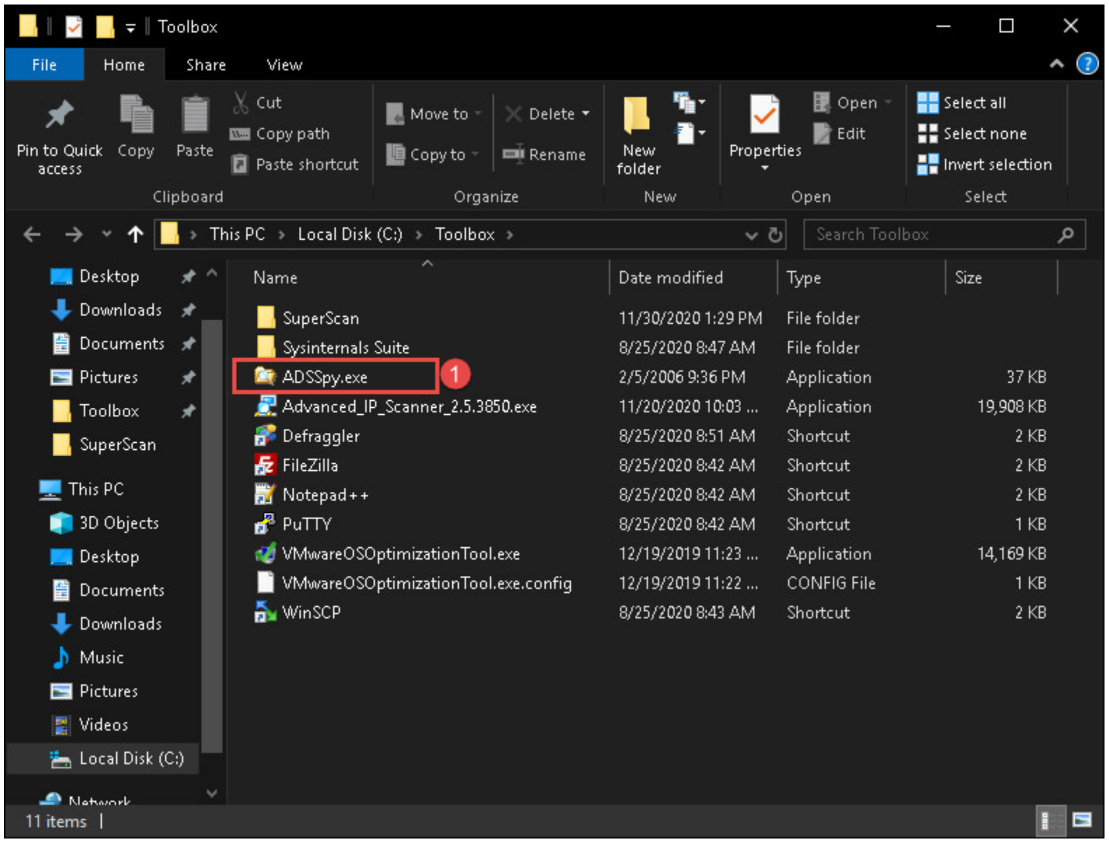
2 Locating Hidden Files Using ADS Spy

In this exercise, we will be using *ADS Spy* to list, view, or delete Alternate Data Stream (ADS) within the NTFS file system. As you know, attackers use NTFS streams to hide sensitive information and even store trojan executable files in ADS streams of random files on the system.

1. Navigate to and double-click the **Toolbox** directory located on the Desktop, as seen in *item* **1** below.



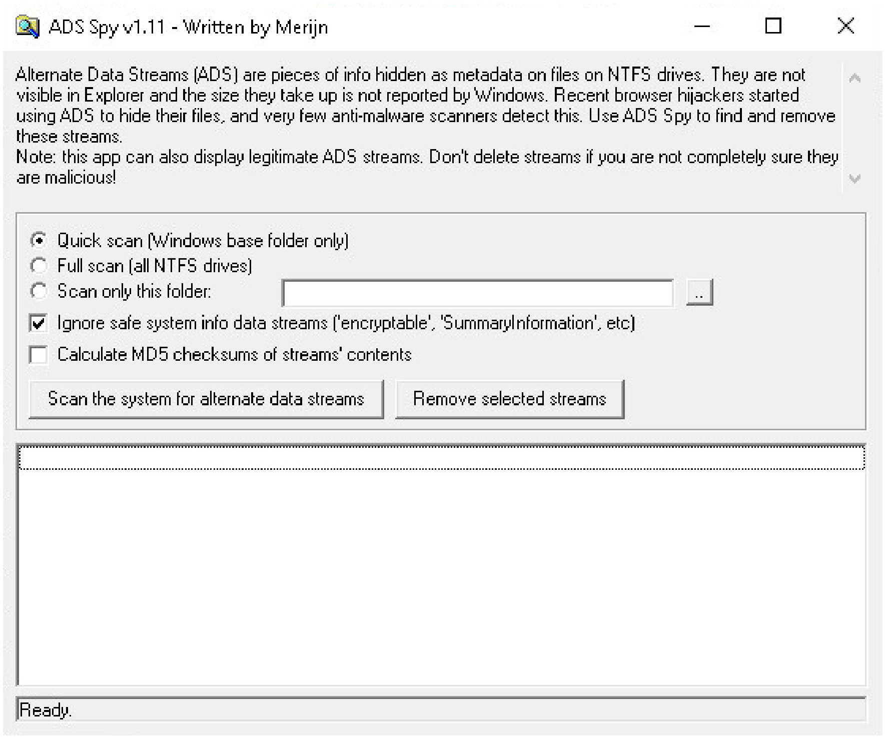
2. In *Windows File Explorer*, double-click **ADSSpy.exe** to launch the application, as highlighted in *item 1.*



Name	Date modified	Type	Size
SuperScan	11/30/2020 1:29 PM	File folder	
Sysinternals Suite	8/25/2020 8:47 AM	File folder	
ADSSpy.exe	2/5/2006 9:36 PM	Application	37 KB
Advanced_IP_Scanner_2.5.3850.exe	11/20/2020 10:03 ...	Application	19,908 KB
Defraggler	8/25/2020 8:51 AM	Shortcut	2 KB
FileZilla	8/25/2020 8:42 AM	Shortcut	2 KB
Notepad++	8/25/2020 8:42 AM	Shortcut	2 KB
PuTTY	8/25/2020 8:42 AM	Shortcut	1 KB
VMwareOSOptimizationTool.exe	12/19/2019 11:23 ...	Application	14,169 KB
VMwareOSOptimizationTool.exe.config	12/19/2019 11:22 ...	CONFIG File	1 KB
WinSCP	8/25/2020 8:43 AM	Shortcut	2 KB

3. The **ADS Spy** main window should appear.

If prompted by an Open File Security warning, click **RUN.**



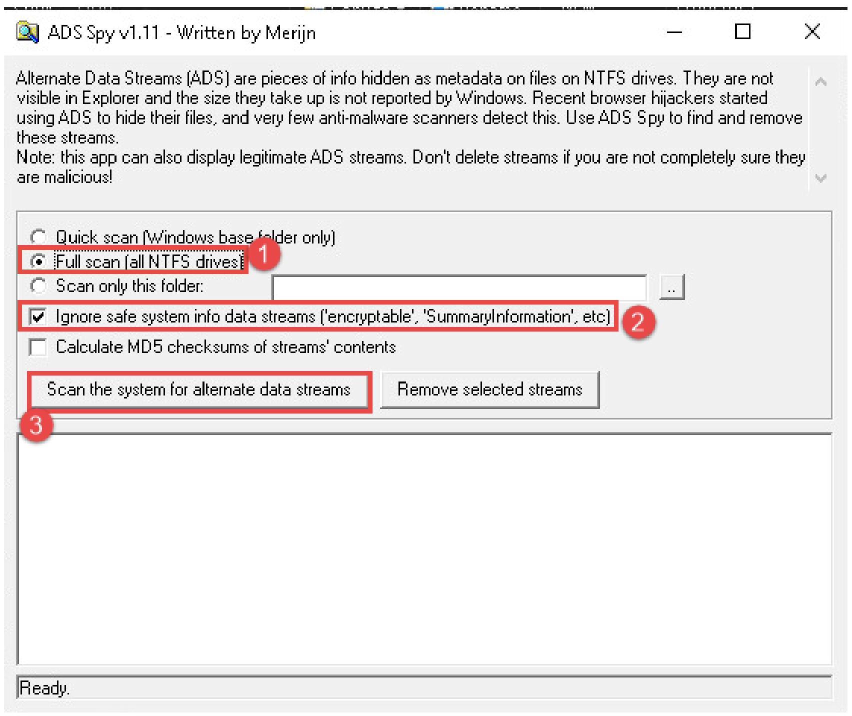
ADS Spy v1.11 - Written by Merijn

Alternate Data Streams (ADS) are pieces of info hidden as metadata on files on NTFS drives. They are not visible in Explorer and the size they take up is not reported by Windows. Recent browser hijackers started using ADS to hide their files, and very few anti-malware scanners detect this. Use ADS Spy to find and remove these streams.
Note: this app can also display legitimate ADS streams. Don't delete streams if you are not completely sure they are malicious!

- [✓ Quick scan (Windows base folder only)]
- Full scan (all NTFS drives)
- Scan only this folder:
- [✓ Ignore safe system info data streams ('encryptable', 'SummaryInformation', etc)]
- Calculate MD5 checksums of streams' contents

Ready.

4. Let us select **Full scan (All NTFS drives)**, check the option to **Ignore safe system info data streams ('encryptable', 'SummaryInformation', etc**), and then click **Scan the system for alternate data streams** as seen in *items* **1, 2,** and **3** below.


ADS Spy v1.11 - Written by Merijn

Alternate Data Streams (ADS) are pieces of info hidden as metadata on files on NTFS drives. They are not visible in Explorer and the size they take up is not reported by Windows. Recent browser hijackers started using ADS to hide their files, and very few anti-malware scanners detect this. Use ADS Spy to find and remove these streams.
Note: this app can also display legitimate ADS streams. Don't delete streams if you are not completely sure they are malicious!

Quick scan (Windows base folder only) [✓ Full scan (all NTFS drives)] Scan only this folder:

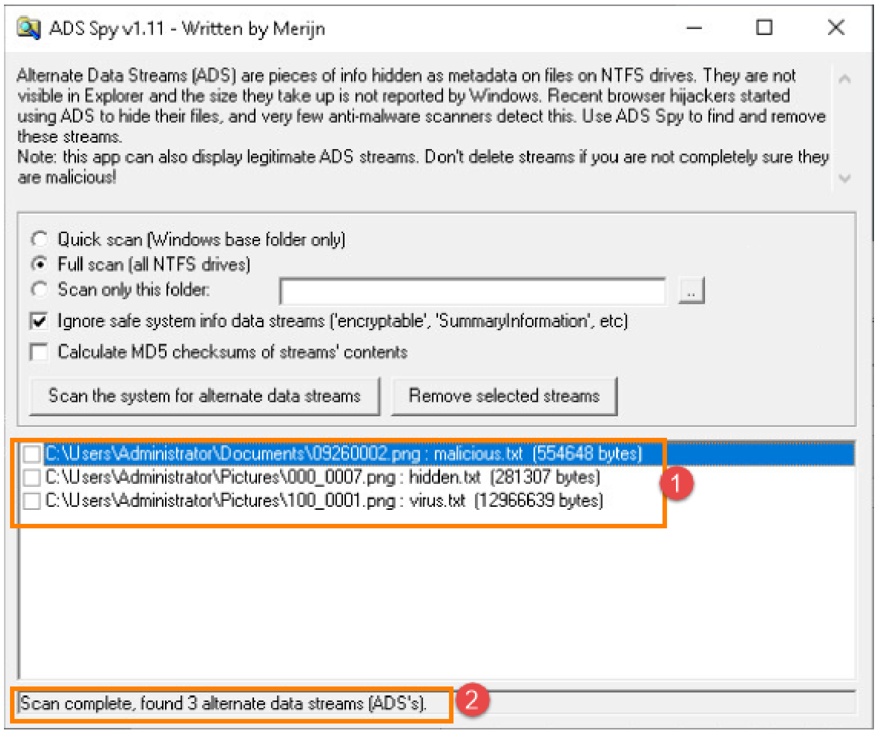
[✓] Ignore safe system info data streams ('encryptable', 'SummaryInformation', etc)

[ ] Calculate MD5 checksums of streams' contents

Ready.

Selecting **Calculate MD5 checksums of the streams' contents** would be an important step if you were review a live host. The MD5 checksum generated could then be used to search sandboxes and flag dangerous files.

5. *ADS Spy* will search the entire NTFS and list all hidden streams. As you can see, *ADS Spy* was able to identify three (3) hidden streams; note the paths as highlighted in *items* **1** and **2** below.


ADS Spy v1.11 - Written by Merijn
Scan the system for alternate data streams
Remove selected streams
1
2

Alternate Data Streams (ADS) are pieces of info hidden as metadata on files on NTFS drives. They are not visible in Explorer and the size they take up is not reported by Windows. Recent browser hijackers started using ADS to hide their files, and very few anti-malware scanners detect this. Use ADS Spy to find and remove these streams.
Note: this app can also display legitimate ADS streams. Don't delete streams if you are not completely sure they are malicious!

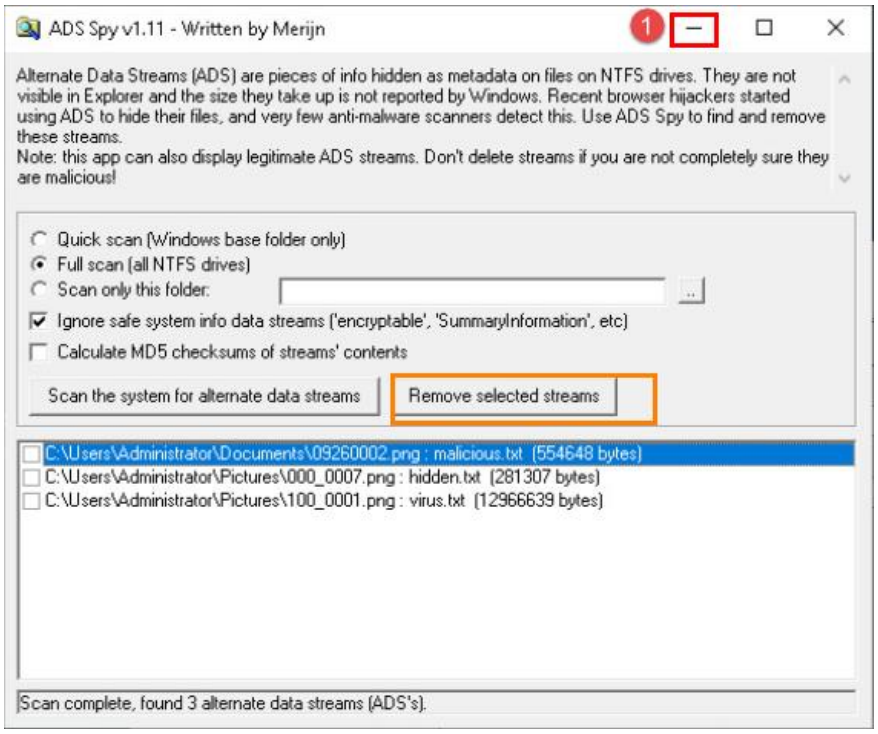
- Quick scan (Windows base folder only)
- [✓ Full scan (all NTFS drives)]
- Scan only this folder:
- [✓] Ignore safe system info data streams ('encryptable', 'SummaryInformation', etc)
- [ ] Calculate MD5 checksums of streams' contents

```
C:\Users\Administrator\Documents\09260002.png : malicious.txt  (554648 bytes)
C:\Users\Administrator\Pictures\000_0007.png : hidden.txt  (281307 bytes)
C:\Users\Administrator\Pictures\100_0001.png : virus.txt  (12966639 bytes)
```

Scan complete, found 3 alternate data streams (ADS's).

At times, the scanner may identify miscellaneous or false positive files, however, we are only interested in the files identified above.

6. Here you would select and delete unwanted/malicious streams from the file system by selecting each checkbox and click **Remove select streams.** However, let us leave them as is until the end of the next task. Minimize the window by clicking the **–** as seen in *item* **1** below.



Alternate Data Streams (ADS) are pieces of info hidden as metadata on files on NTFS drives. They are not visible in Explorer and the size they take up is not reported by Windows. Recent browser hijackers started using ADS to hide their files, and very few anti-malware scanners detect this. Use ADS Spy to find and remove these streams.
Note: this app can also display legitimate ADS streams. Don't delete streams if you are not completely sure they are malicious!

- ( ) Quick scan (Windows base folder only)
- (•) Full scan (all NTFS drives)
- ( ) Scan only this folder:
- [✓] Ignore safe system info data streams ('encryptable', 'SummaryInformation', etc)
- [ ] Calculate MD5 checksums of streams' contents

```
C:\Users\Administrator\Documents\09260002.png : malicious.txt  (554648 bytes)
C:\Users\Administrator\Pictures\000_0007.png : hidden.txt  (281307 bytes)
C:\Users\Administrator\Pictures\100_0001.png : virus.txt  (12966639 bytes)
```

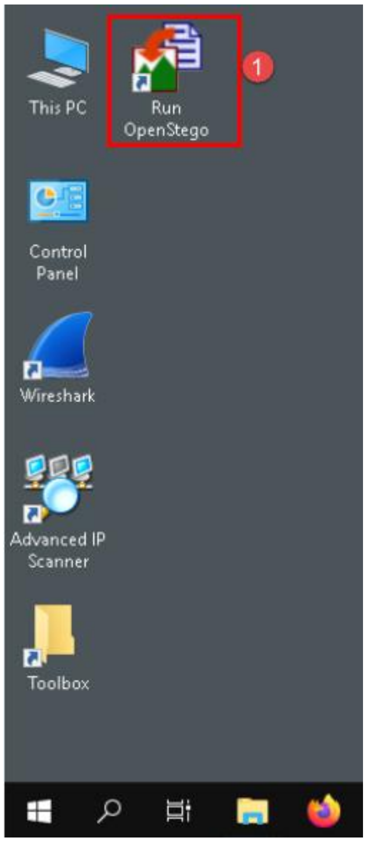
Scan complete, found 3 alternate data streams (ADS's).

Note the path to each ADS stream file.

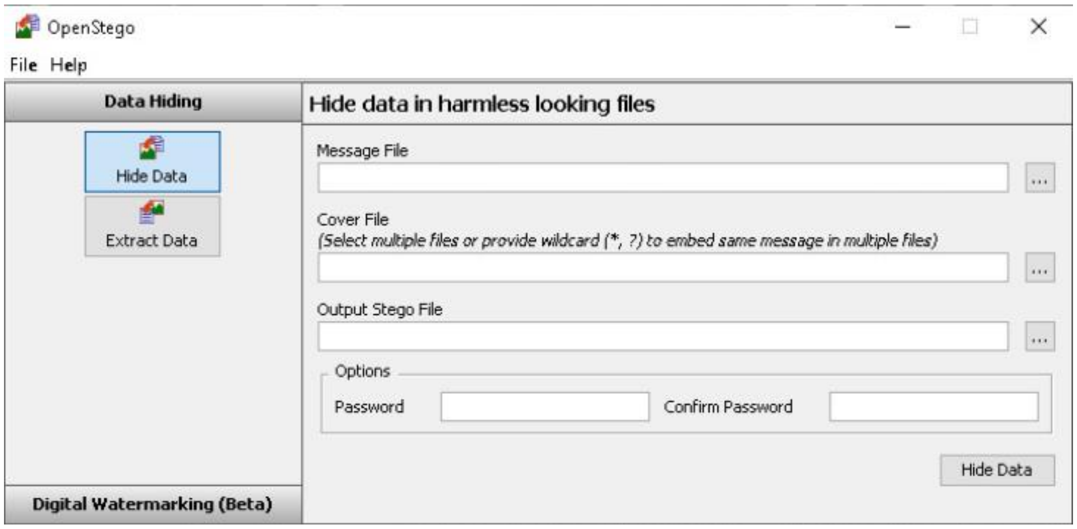
3 Extract ADS File Using OpenStego

Knowing that several different types of files can hold all sorts of hidden/malicious information, tracking or finding these files can be an almost impossible task. ADS can fall under stenographic/data hiding techniques. This allows the attacker to retrieve secret messages, hide a malicious file or send updates within the users knowing.

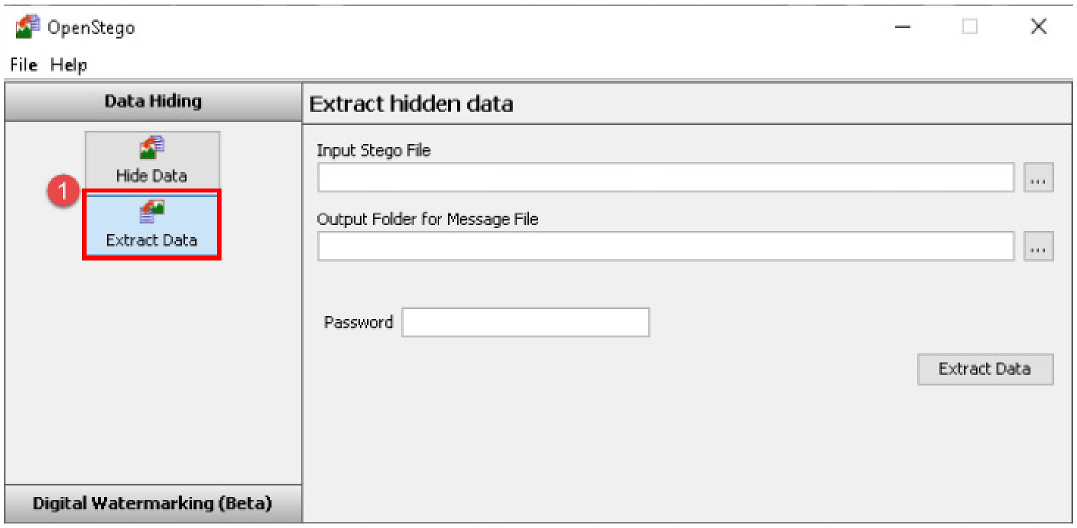
1. Navigate to and double-click **Run OpenStego** located on the Desktop to launch the application, as seen in *item* **1** below.



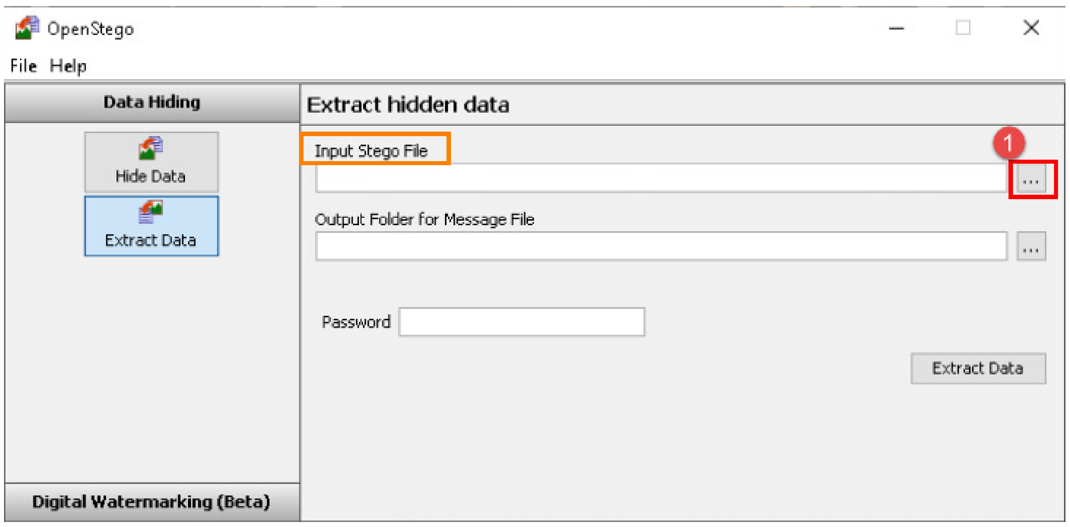
2. The *OpenStego* main window should appear.


OpenStego
File Help
Data Hiding
Hide Data
Extract Data
Digital Watermarking (Beta)
Hide data in harmless looking files
Message File
Cover File
(Select multiple files or provide wildcard (*, ?) to embed same message in multiple files)
Output Stego File
Options
Password
Confirm Password
Hide Data

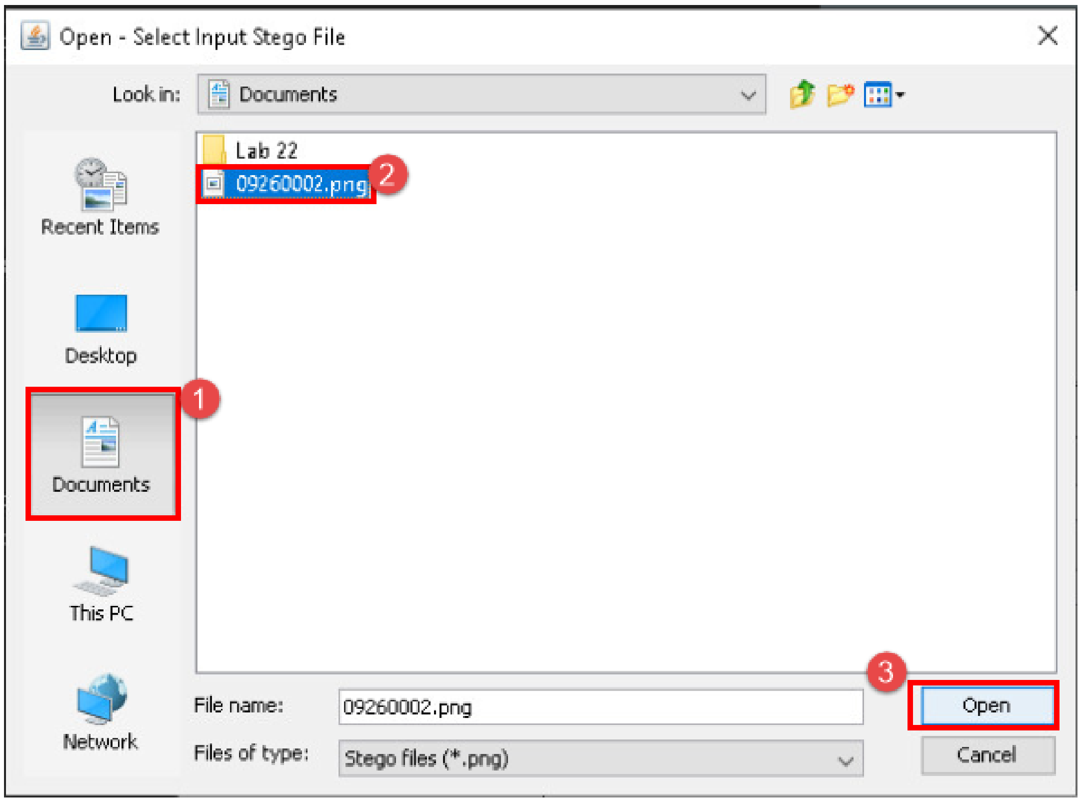
3. Since we located three (3) ADS files from the previous exercise, we will now be extracting the data and examining the content. As we outlined, ADS files can contain malicious files, and we should take due care when attempting to examine the stream files. Click **Extract Data** under the Data Hiding pane, as seen in *item* **1** below.



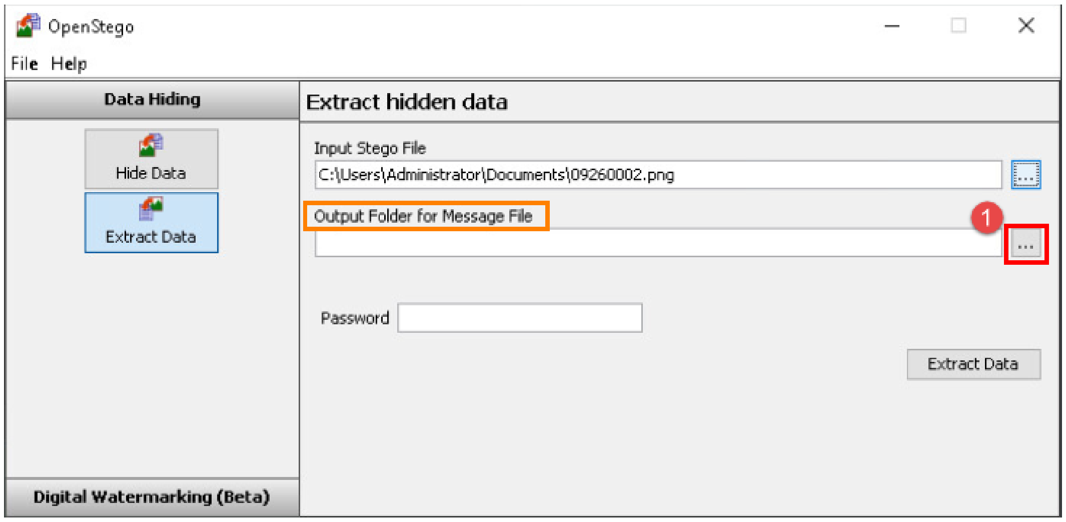
4. Next, click the **ellipsis** button to the right of the **Input Stego File** box, as seen in *item* **1** below.



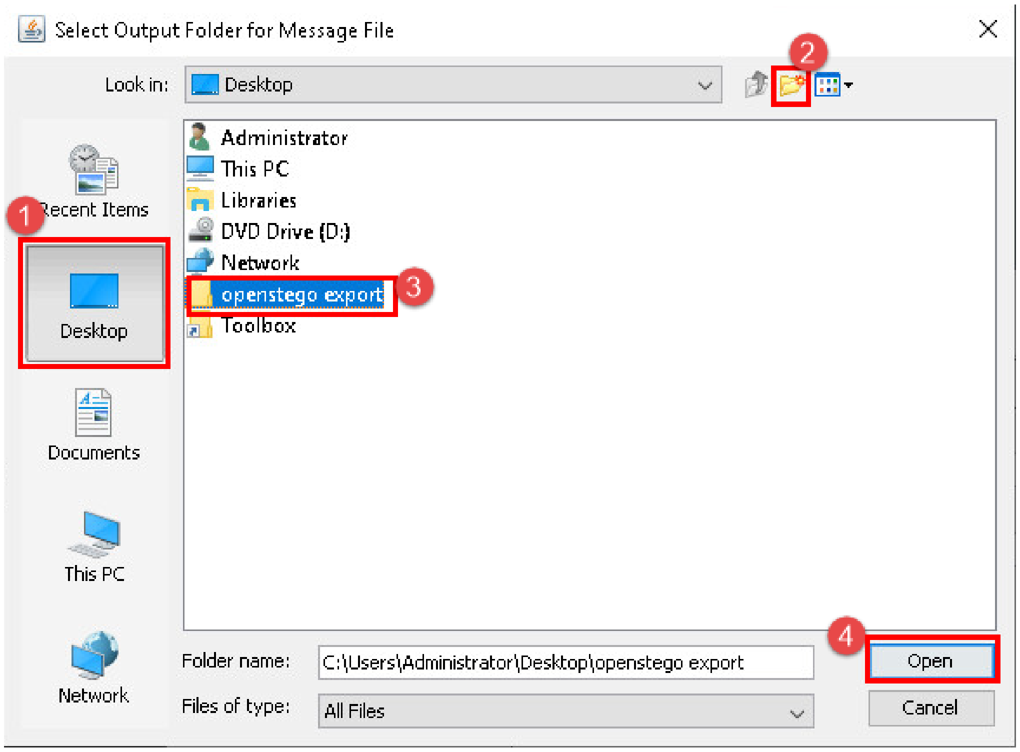
5. The **Open – Select Input Stego File** window will open. Navigate to the **Documents** directory, select **09260002.png** and click **Open** as seen in *items* **1, 2,** and **3** below.



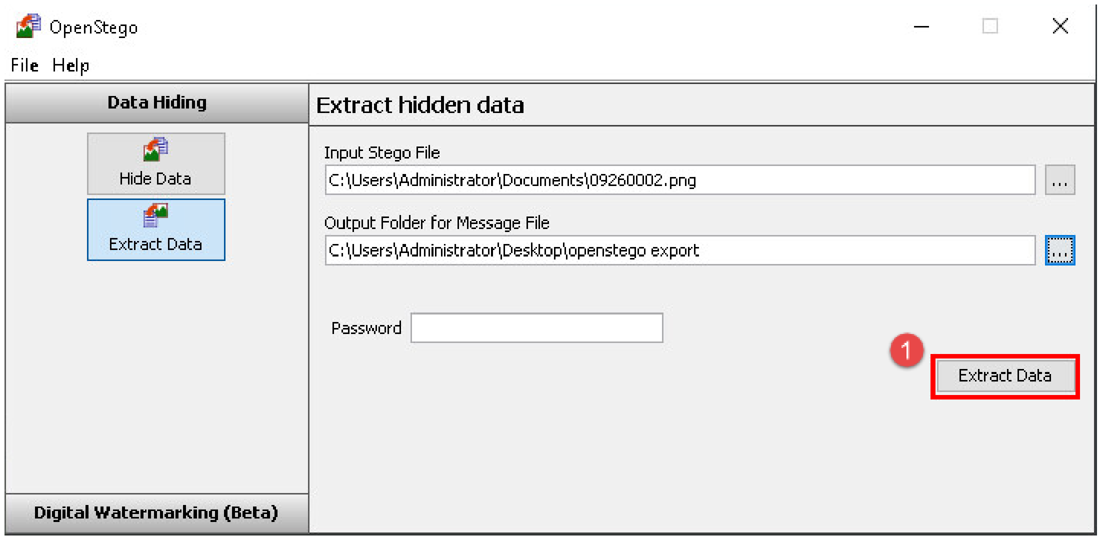
6. Click the **ellipsis** button to the right of the **Output Folder for Message File** box, as seen in *item* **1** below.



7. The **Select Output Folder for Message File** window will appear. Navigate to the **Desktop**, select **Create New Folder** and name it `openstego export` and click **Open** as seen in *items* **1, 2, 3,** and **4** below.



8. Click **Extract Data** as seen in *item* **1**. This will extract the file from within the image and saves it to the destination folder.

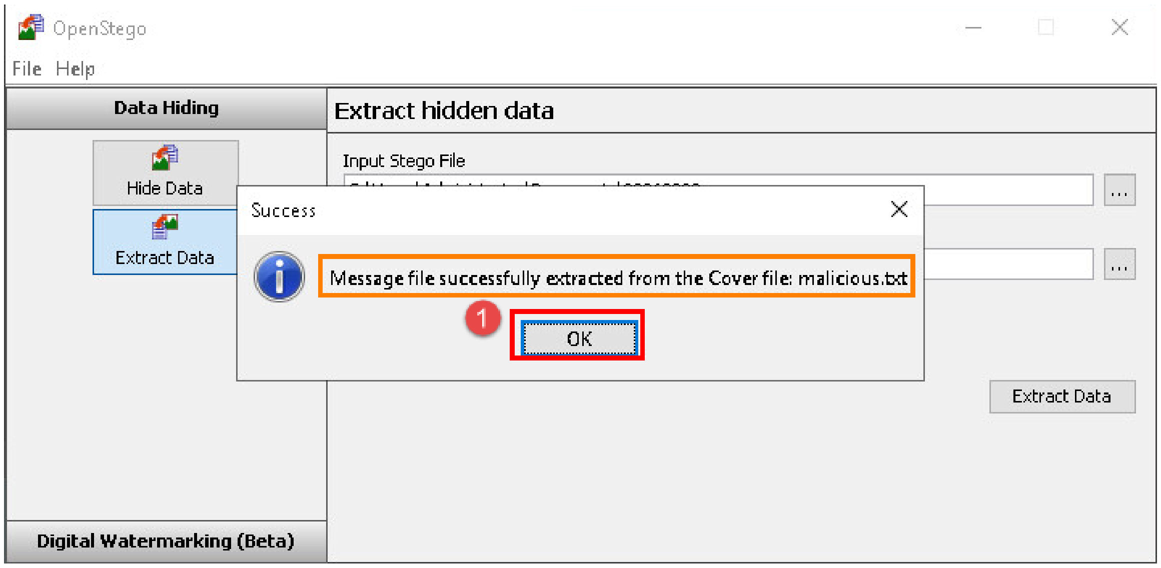


Input Stego File: C:\Users\Administrator\Documents\09260002.png

Output Folder for Message File: C:\Users\Administrator\Desktop\openstego export

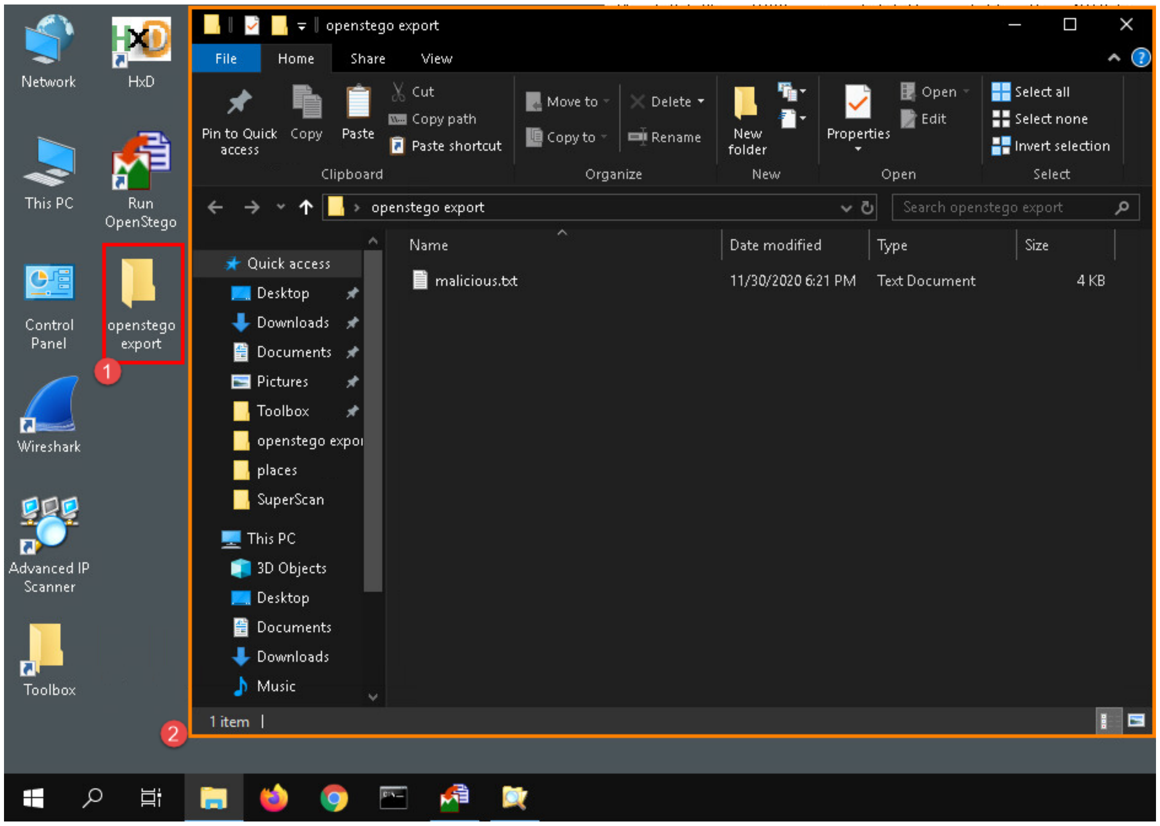
Password

9. A success dialog box prompt should appear, stating that the message file has been successfully extracted from the cover file Click **OK** as seen in *item* **1** below. The message file will be displayed in the destination folder specified in step 7.



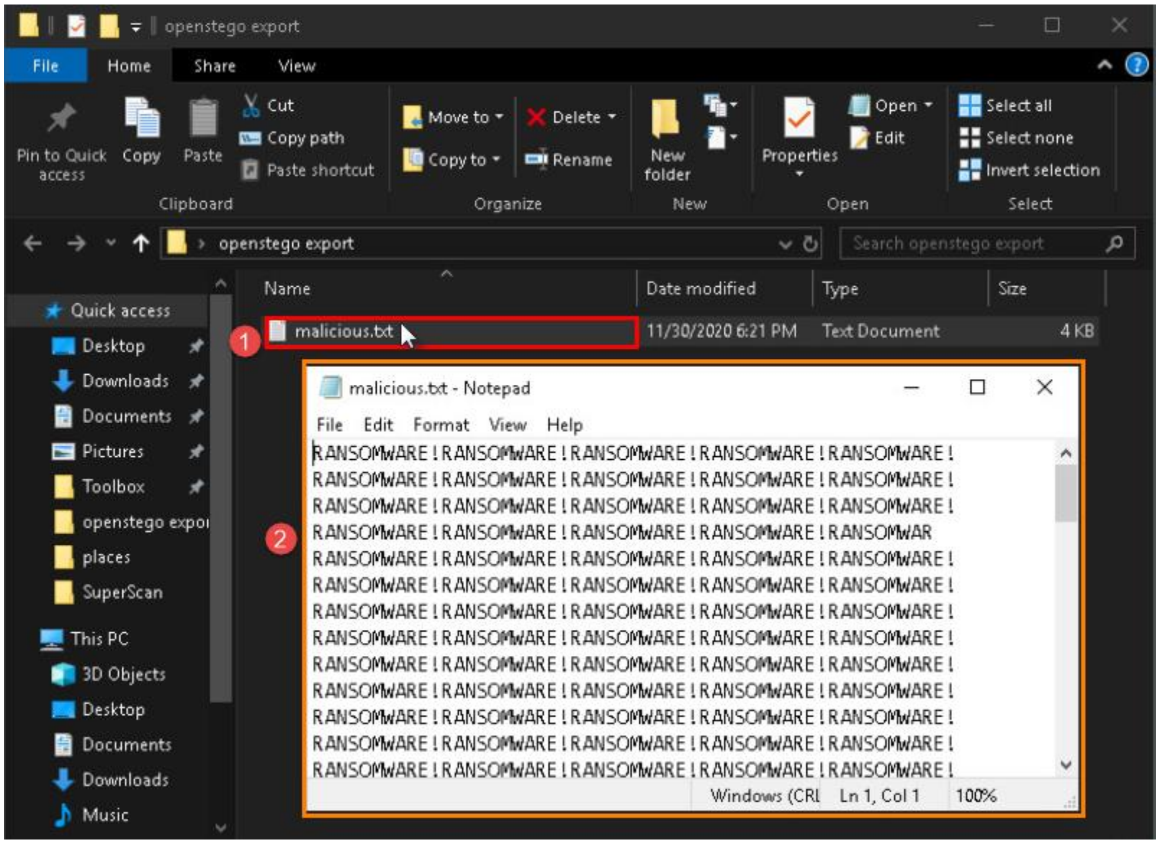
Message file successfully extracted from the Cover file: malicious.txt

10. If you did everything correctly, the file should be saved. Now, navigate to the **Desktop** and double-click the folder **openstego export** to open it, as seen at *items* **1** and **2** below.



Name	Date modified	Type	Size
malicious.txt	11/30/2020 6:21 PM	Text Document	4 KB

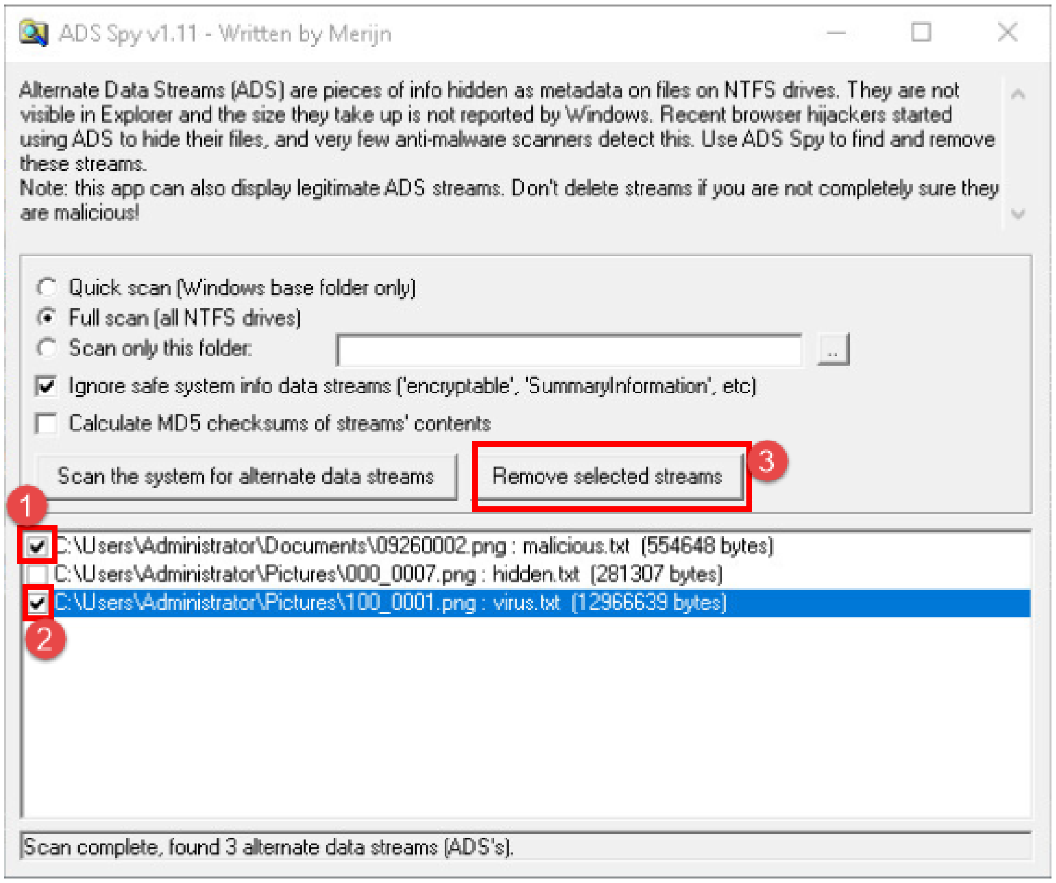
11. Since the extracted ADS file is a text file, there is no risk of harming the computer. However, if the files were malicious, you would first generate the MD5 hashes before attempting to move the files to a sandbox. So, let's **double-click** the file to view the content, as seen in *items* **1** and **2** below.



Name	Date modified	Type	Size
malicious.txt	11/30/2020 6:21 PM	Text Document	4 KB

```
RANSOMWARE!RANSOMWARE!RANSOMWARE!RANSOMWARE!RANSOMWARE!
RANSOMWARE!RANSOMWARE!RANSOMWARE!RANSOMWARE!RANSOMWARE!
RANSOMWARE!RANSOMWARE!RANSOMWARE!RANSOMWARE!RANSOMWARE!
RANSOMWARE!RANSOMWARE!RANSOMWARE!RANSOMWARE!RANSOMWAR
RANSOMWARE!RANSOMWARE!RANSOMWARE!RANSOMWARE!RANSOMWARE!
RANSOMWARE!RANSOMWARE!RANSOMWARE!RANSOMWARE!RANSOMWARE!
RANSOMWARE!RANSOMWARE!RANSOMWARE!RANSOMWARE!RANSOMWARE!
RANSOMWARE!RANSOMWARE!RANSOMWARE!RANSOMWARE!RANSOMWARE!
RANSOMWARE!RANSOMWARE!RANSOMWARE!RANSOMWARE!RANSOMWARE!
RANSOMWARE!RANSOMWARE!RANSOMWARE!RANSOMWARE!RANSOMWARE!
RANSOMWARE!RANSOMWARE!RANSOMWARE!RANSOMWARE!RANSOMWARE!
RANSOMWARE!RANSOMWARE!RANSOMWARE!RANSOMWARE!RANSOMWARE!
RANSOMWARE!RANSOMWARE!RANSOMWARE!RANSOMWARE!RANSOMWARE!
```

12. **Maximize** *ADS Spy* and select the malicious streams' checkboxes **malicious.txt** and **virus.txt** and click **Remove selected streams.**



ADS Spy v1.11 - Written by Merijn

Alternate Data Streams (ADS) are pieces of info hidden as metadata on files on NTFS drives. They are not visible in Explorer and the size they take up is not reported by Windows. Recent browser hijackers started using ADS to hide their files, and very few anti-malware scanners detect this. Use ADS Spy to find and remove these streams.
Note: this app can also display legitimate ADS streams. Don't delete streams if you are not completely sure they are malicious!

- [ ] Quick scan (Windows base folder only)
- [✓] Full scan (all NTFS drives)
- [ ] Scan only this folder:
- [✓] Ignore safe system info data streams ('encryptable', 'SummaryInformation', etc)
- [ ] Calculate MD5 checksums of streams' contents

- [✓] C:\Users\Administrator\Documents\09260002.png : malicious.txt (554648 bytes)
- [ ] C:\Users\Administrator\Pictures\000_0007.png : hidden.txt (281307 bytes)
- [✓] C:\Users\Administrator\Pictures\100_0001.png : virus.txt (12966639 bytes)

Scan complete, found 3 alternate data streams (ADS's).

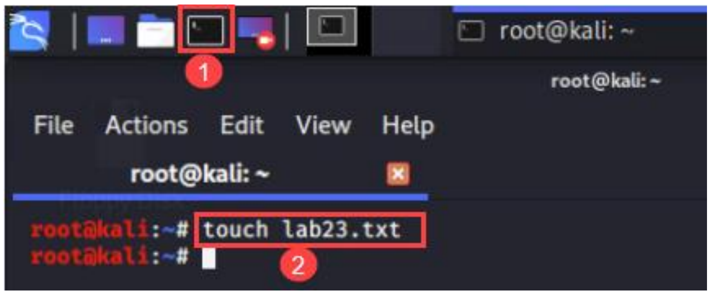
13. This is the end of the exercise. Close all windows to complete the lab.

4 Using Extended File Attributes on Linux to Hide Data

While EXT4 file systems do not have alternate data streams, Linux operating systems can still hide data in plain sight. This data hiding technique is done by hiding data in the extended metadata of a file. In this lab, we will create a file and add the extended attributes and then view them.

1. To begin, switch to the Kali Linux VM and log in as `root` using the password `toor`. Then, click the terminal icon from the navigator panel, as seen in *item* 1. Once the *Terminal* window opens, type `touch lab23.txt` and press **Enter** as seen in *item* **2.**

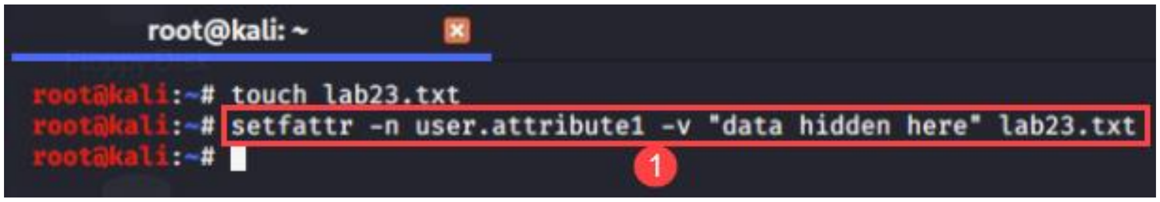
```
root@kali:~# touch lab23.txt
```



```
root@kali:~# touch lab23.txt
root@kali:~#
```

2. Now let us add the text we intend to hide in the extended attributes of this file. To do this, type `setfattr -n user.attribute1 -v “data hidden here” lab23.txt` and press **Enter** as seen in *item* **1**.

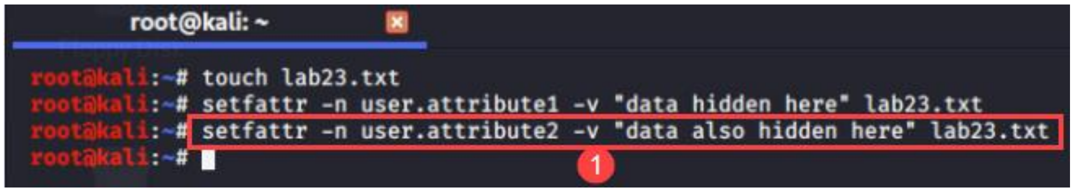
```
root@kali:~# setfattr -n user.attribute1 -v “data hidden here” lab23.txt
```



```
root@kali:~# touch lab23.txt
root@kali:~# setfattr -n user.attribute1 -v "data hidden here" lab23.txt
root@kali:~#
```

3. Next, we will add another attribute by typing `setfattr -n user.attribute2 -v “data also hidden here” lab23.txt` and press **Enter** as seen in *item* **1.**

```
root@kali:~# setfattr -n user.attribute2 -v “data also hidden here” lab23.txt
```

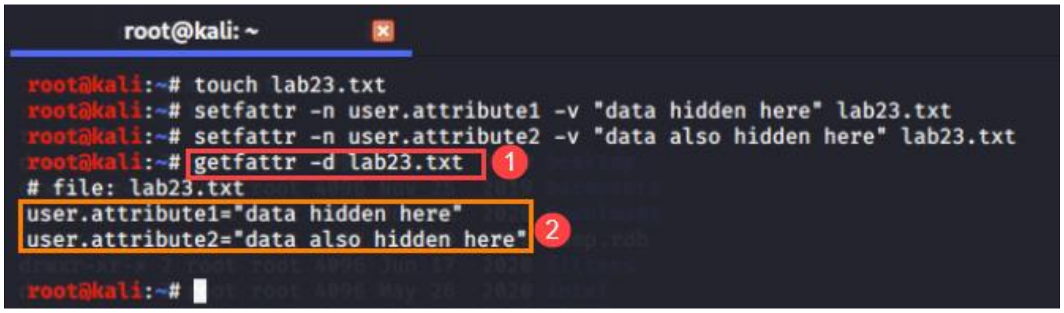


```
root@kali: ~
root@kali:~# touch lab23.txt
root@kali:~# setfattr -n user.attribute1 -v "data hidden here" lab23.txt
root@kali:~# setfattr -n user.attribute2 -v "data also hidden here" lab23.txt
root@kali:~#
```

The **-n** option refers to the name of the attribute we will be creating. The **-v** option is the value of the attribute.

4. The attributes are now set for the file. These attributes will not be seen by the user in the File Manager or in a file list. To view the attributes, type `getfattr -d lab23.txt` and press **Enter** as seen in *item* **1.** As you can see in *item* **2,** the attribute names and their related data are printed in the *Terminal*.

```
root@kali:~# getfattr -d lab23.txt
```



```
root@kali: ~
root@kali:~# touch lab23.txt
root@kali:~# setfattr -n user.attribute1 -v "data hidden here" lab23.txt
root@kali:~# setfattr -n user.attribute2 -v "data also hidden here" lab23.txt
root@kali:~# getfattr -d lab23.txt
# file: lab23.txt
user.attribute1="data hidden here"
user.attribute2="data also hidden here"
root@kali:~#
```

The **-n** option refers to the name of the attribute we will be creating. The **-v** option is the value of the attribute. The **-d** option lists all extended attributes.

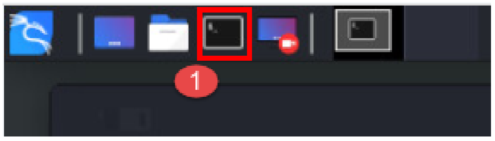
5. This is a quick and simple way of hiding data on Linux systems and can be used to store a small amount of information for various purposes.

attr is not installed on all Linux distributions by default. Use **sudo apt install attr** on Ubuntu systems. (Research the installation for attr on other systems).

5 Covering Tracks by Deleting Linux Logs

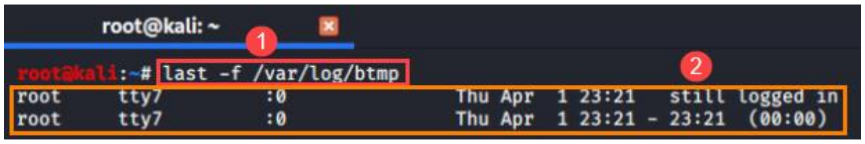
Threat Actors often try to cover their tracks, and one very common tactic used is the removal of logs that can indicate their activity. In this exercise, we will review a few of the logs that will leave traces.

1. The *Terminal* should still be open; if not, reopen it by clicking the **Terminal** icon on the navigation panel as seen in *item* **1.**



2. The first log we will review is from the **btmp** file, which provides details about the last login sessions on the system. The *btmp* file is not a raw text file, so let us type the command `last -f /var/log/btmp` and press **Enter** as seen in *item* **1.** The results will populate the terminal window (your output may differ from the screenshot). The entry seen in *item* **2** indicates the username of the user, the time of login, and the duration of the session. It indicates that the user that logged in on *Thursday, April 1 at 23:21* is still logged in.

```
root@kali:~# last -f /var/log/btmp
```

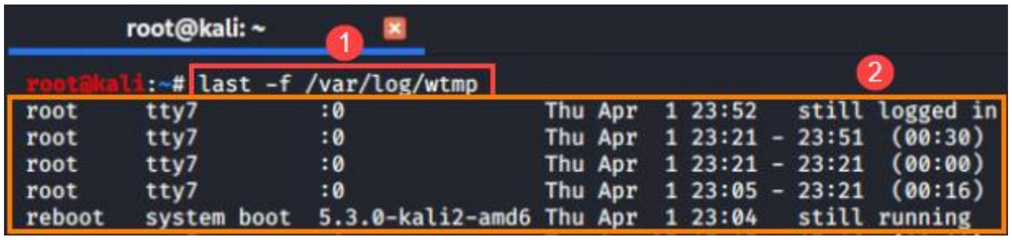


```
root@kali:~# last -f /var/log/btmp
root     tty7          :0                Thu Apr  1 23:21   still logged in
root     tty7          :0                Thu Apr  1 23:21 - 23:21  (00:00)
```

tty7 is a service used by *Gnome* desktop manager and other desktop environments. It is an indicator that the log in was through graphical user interface and not terminal etc.

3. Next, we will look at the *wtmp* file, which is similar to the *btmp* but contains additional system data such as reboots. To view the contents, `type last -f /var/log/wtmp` and press **Enter** as seen in *item* **1.** The results will appear in the *Terminal* window and will provide additional login dates and times as well as system boots and reboots, as seen in *item* **2.**

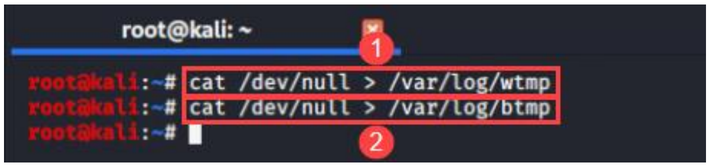
```
root@kali:~# last -f /var/log/wtmp
```



```
root@kali:~# last -f /var/log/wtmp
root     tty7          :0                Thu Apr  1 23:52   still logged in
root     tty7          :0                Thu Apr  1 23:21 - 23:51  (00:30)
root     tty7          :0                Thu Apr  1 23:21 - 23:21  (00:00)
root     tty7          :0                Thu Apr  1 23:05 - 23:21  (00:16)
reboot   system boot  5.3.0-kali2-amd6 Thu Apr  1 23:04   still running
```

4. To remove this data and further cover our tracks, type `cat /dev/null > /var/log/wtmp` and press **Enter** as seen in *item* **1** to replace *wtmp* with a blank file. Then type `cat /dev/null > /var/log/btmp` and press **Enter** as seen in *item* **2** to replace *btmp* with a blank file.

```
root@kali:~# cat /dev/null > /var/log/wtmp
root@kali:~# cat /dev/null > /var/log/btmp
```

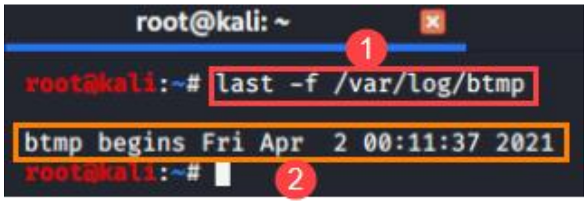


```
root@kali:~
root@kali:~# cat /dev/null > /var/log/wtmp
root@kali:~# cat /dev/null > /var/log/btmp
root@kali:~#
```

The command *cat /dev/null* command is used for discarding unwanted data from files and is a safe way to ensure the data is deleted while keeping the system functioning.

5. Now that the commands appeared to run successfully, let us verify that the contents are cleared. To do this, type `last -f /var/log/btmp` and press **Enter** as seen in *item* **1.** As you can see in *item* **2,** the log indicates that the *btmp* begins at the current date and time.

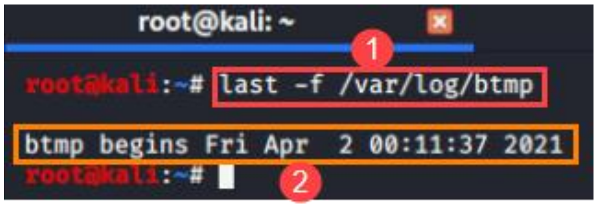
```
root@kali:~# last -f /var/log/btmp
```



```
root@kali:~
root@kali:~# last -f /var/log/btmp

btmp begins Fri Apr  2 00:11:37 2021
root@kali:~#
```

6. Next, type `last -f /var/log/wtmp` and press **Enter** as seen in *item* **1.** As you can see in *item* **2,** this log is also cleared.

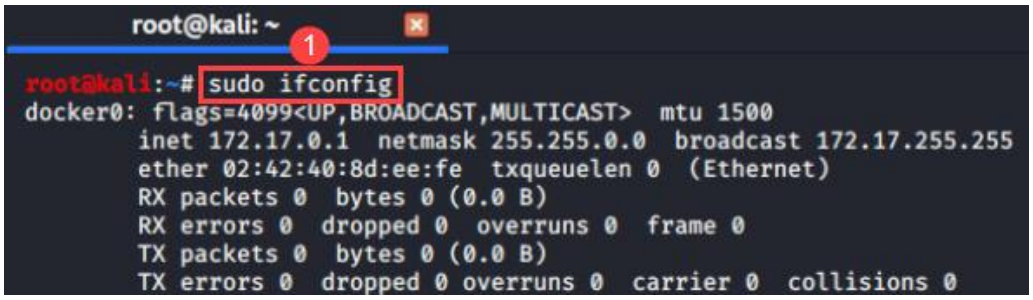
```
root@kali:~# last -f /var/log/wtmp”
```



```
root@kali:~
root@kali:~# last -f /var/log/btmp

btmp begins Fri Apr  2 00:11:37 2021
root@kali:~#
```

7. The next log we will look at is the *sudo* log. This can be found in *auth.log* file and shows the specific commands typed as a privileged user. The *auth.log* file can contain lots of data that makes it hard to read. We will use the *grep* command to focus on only the instances that the *sudo* command was used. First, let us ensure we will get some results by running a command using *sudo*. Do this by typing **`sudo ifconfig`** and pressing **Enter** as seen in *item* **1.**

```
root@kali:~# sudo ifconfig
```

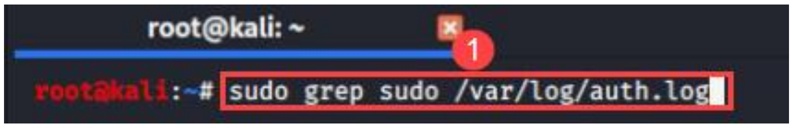


```
root@kali:~# sudo ifconfig
docker0: flags=4099<UP,BROADCAST,MULTICAST>  mtu 1500
        inet 172.17.0.1  netmask 255.255.0.0  broadcast 172.17.255.255
        ether 02:42:40:8d:ee:fe  txqueuelen 0  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

The **sudo** command allows the user to request root privileges to execute the command that follows. However, we are already logged in as root so the reason for **sudo** now is to allow the command to be placed inside the **auth.log** file.

8. Now that we are sure there is a *sudo* command for this session, type **`sudo grep sudo /var/log/auth.log`** and press **Enter** as seen in *item* **1.**

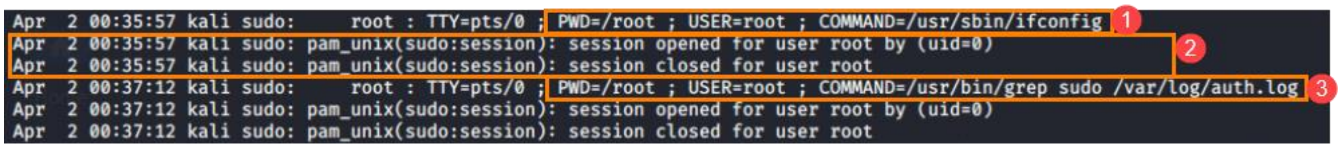
```
root@kali:~# sudo grep sudo /var/log/auth.log
```



```
root@kali:~# sudo grep sudo /var/log/auth.log
```

Alternatively, you can type the command **`sudo cat /var/log/auth.log | grep sudo`** both commands will produce the similar results.

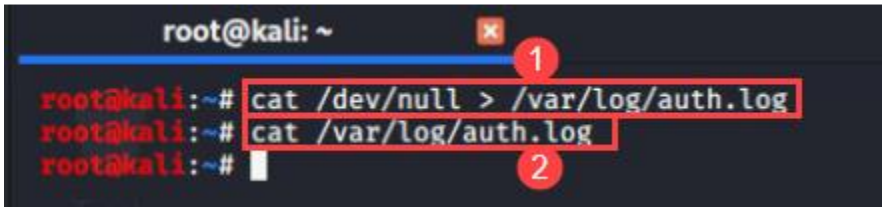
9. Now, look at the results to see if you can identify the *ifconfig* command you typed earlier. As you can see in *item* **1,** the command shows up in the log. The reason why it looks different is that the *auth.log* shows the path the command was run from. In this case, *ifconfig* was run from *sbin,* which is a folder that stores binaries for privileged users. The entry in *item* **1** also shows the directory the command was run from, which is */root,* and the user that ran the command, which is *root. Item* **2** shows the date and time the session was opened and closed, which gives an idea of how long the session was active. Finally, in *item* **3,** is the command we ran to show these results. Notice the working directory and user are the same, but the path of the binary is *bin* instead of *sbin.* This is because the *grep* command is not stored in the privileged users' binaries.



```
Apr  2 00:35:57 kali sudo:     root : TTY=pts/0 ; PWD=/root ; USER=root ; COMMAND=/usr/sbin/ifconfig
Apr  2 00:35:57 kali sudo: pam_unix(sudo:session): session opened for user root by (uid=0)
Apr  2 00:35:57 kali sudo: pam_unix(sudo:session): session closed for user root
Apr  2 00:37:12 kali sudo:     root : TTY=pts/0 ; PWD=/root ; USER=root ; COMMAND=/usr/bin/grep sudo /var/log/auth.log
Apr  2 00:37:12 kali sudo: pam_unix(sudo:session): session opened for user root by (uid=0)
Apr  2 00:37:12 kali sudo: pam_unix(sudo:session): session closed for user root
```

10. As you saw in the previous step, this data is very detailed and should be removed to cover tracks as well. To remove the contents of this file, type `cat /dev/null > /var/log/auth.log` and press **Enter** as seen in *item* **1.** To verify that it was emptied, type `cat /var/log/auth.log` and press **Enter** as seen in *item* **2.** If nothing appears, then you have successfully cleared the *sudo* logs.

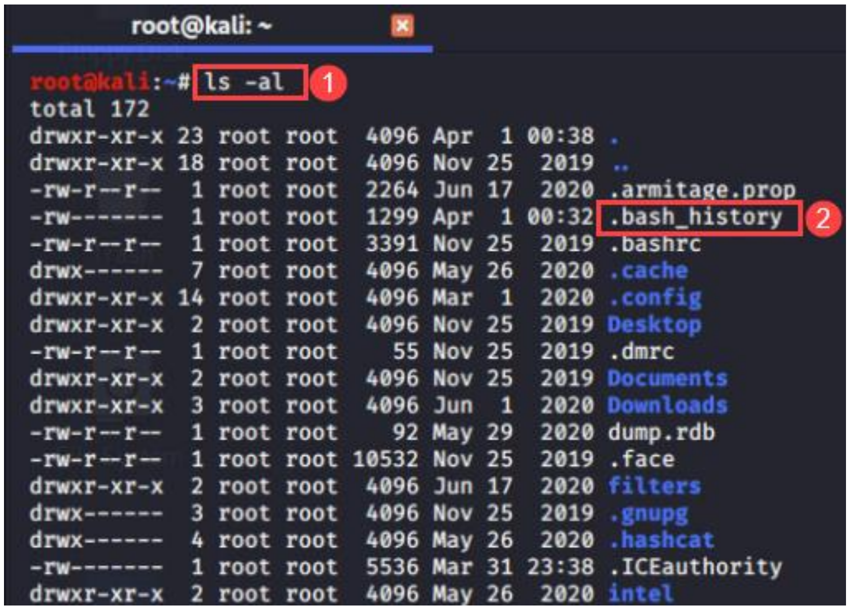
```
root@kali:~# cat /dev/null > /var/log/auth.log
root@kali:~# cat /var/log/auth.log
```



```
root@kali:~# cat /dev/null > /var/log/auth.log
root@kali:~# cat /var/log/auth.log
root@kali:~#
```

11. The last, but one of the most important logs we will look at, is the *Bash history* which is found in each user's root directory. Since we are already in the root directory, let us locate it by typing **ls -al** and pressing **Enter** as seen in *item* **2.** Notice the file called *.bash_history* seen in *item* **3**. This file stores all the commands typed in the *Terminal* window and can give away the attacker's activity on the system. This is also the reason why this log should be cleared when you are about to exit the system.

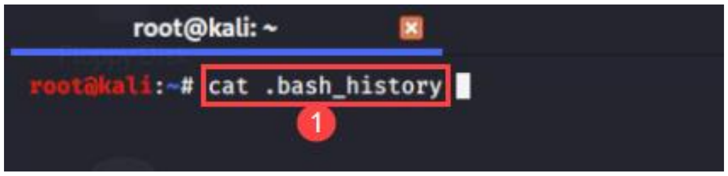
```
root@kali:~# ls -al
```



```
root@kali:~# ls -al
total 172
drwxr-xr-x 23 root root  4096 Apr  1 00:38 .
drwxr-xr-x 18 root root  4096 Nov 25  2019 ..
-rw-r--r--  1 root root  2264 Jun 17  2020 .armitage.prop
-rw-------  1 root root  1299 Apr  1 00:32 .bash_history
-rw-r--r--  1 root root  3391 Nov 25  2019 .bashrc
drwx------  7 root root  4096 May 26  2020 .cache
drwxr-xr-x 14 root root  4096 Mar  1  2020 .config
drwxr-xr-x  2 root root  4096 Nov 25  2019 Desktop
-rw-r--r--  1 root root    55 Nov 25  2019 .dmrc
drwxr-xr-x  2 root root  4096 Nov 25  2019 Documents
drwxr-xr-x  3 root root  4096 Jun  1  2020 Downloads
-rw-r--r--  1 root root    92 May 29  2020 dump.rdb
-rw-r--r--  1 root root 10532 Nov 25  2019 .face
drwxr-xr-x  2 root root  4096 Jun 17  2020 filters
drwx------  3 root root  4096 Nov 25  2019 .gnupg
drwx------  4 root root  4096 May 26  2020 .hashcat
-rw-------  1 root root  5536 Mar 31 23:38 .ICEauthority
drwxr-xr-x  2 root root  4096 May 26  2020 intel
```

12. Let us look at the contents of this file. Type `cat .bash_history` and press **Enter** as seen in *item* **1.** The results will appear, with a command listed on each line. Look at the list to see if you are familiar with the commands.

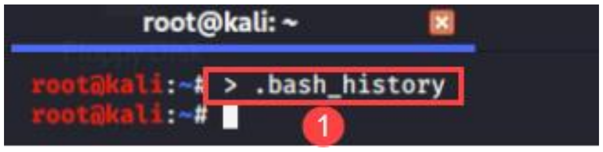
```
root@kali:~# cat .bash_history
```



```
root@kali:~# cat .bash_history
```

13. Now that you are aware of the content, you can see why attackers would not want forensic examiners to find this information. To remove the data without deleting the file, simply type `> .bash_history` and press **Enter** as seen in *item* **1.**

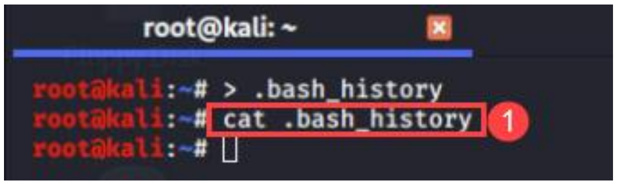
```
root@kali:~# > .bash_history
```



```
root@kali:~# > .bash_history
root@kali:~#
```

14. Now type `cat .bash_history` and press **Enter** to verify whether the contents were overwritten, as seen in *item* **1.** As you can see, there is no data in this file.

```
root@kali:~# cat .bash_history
```



```
root@kali:~# > .bash_history
root@kali:~# cat .bash_history
root@kali:~#
```

There is a *bash_history* file for each user account including the root user. Check all files to ensure that you are not leaving any data in the files.

15. There are many other log files that can be used to prove attackers accessed the system, but the ones we covered focused on removing evidence of the attacker's login and *Terminal* activity. To learn more, you can research other types of logs that can help prove the attacker's access.
16. This is the end of the lab; please close all windows and terminals to complete the lab.