CSE 5382: Secure Programming - Spring 2021 Student Name: Sudharsan Srinivasan

Capture the Flag Bonus Exercise Student ID: 1001755919

# Flag Table

Place the flags once you find them in the appropriate table entries below. Each category has a maximum of 20 points with each higher level of difficulty granting an increasing number of points. Maximum score for all challenges completed is 100 points.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Cryptography** | **Malware** | **Network Capture** | **Reverse Engineering** | **Steganography** |
| **Level 1 (2 Points)** | backoff\_malware | flag{sandworm\_apt} | M0d1c0nF14G | lm | flag{covert\_channel} |
| **Level 2 (3 Points)** | WANNACRY | 2e1afcef9baa15b8db764274b0e45d3f | YWRtaW46YWRtaW4= | ClippyHasReturned | flag{cookies\_n\_milk} |
| **Level 3 (4 Points)** |  | flag{duqu\_aint\_dooku} | NTLMSSP\_NEGOTIATE | infected flag | R0075RUS |
| **Level 4 (5 Points)** |  | 123123 | 1255 | 73C972DF8DB0E1EDC289F491A09AF330 | flag{alan\_turing\_edm} |
| **Level 5 (6 Points)** |  | flag{kragany\_uses\_up$x} | passwordBasisk |  |  |

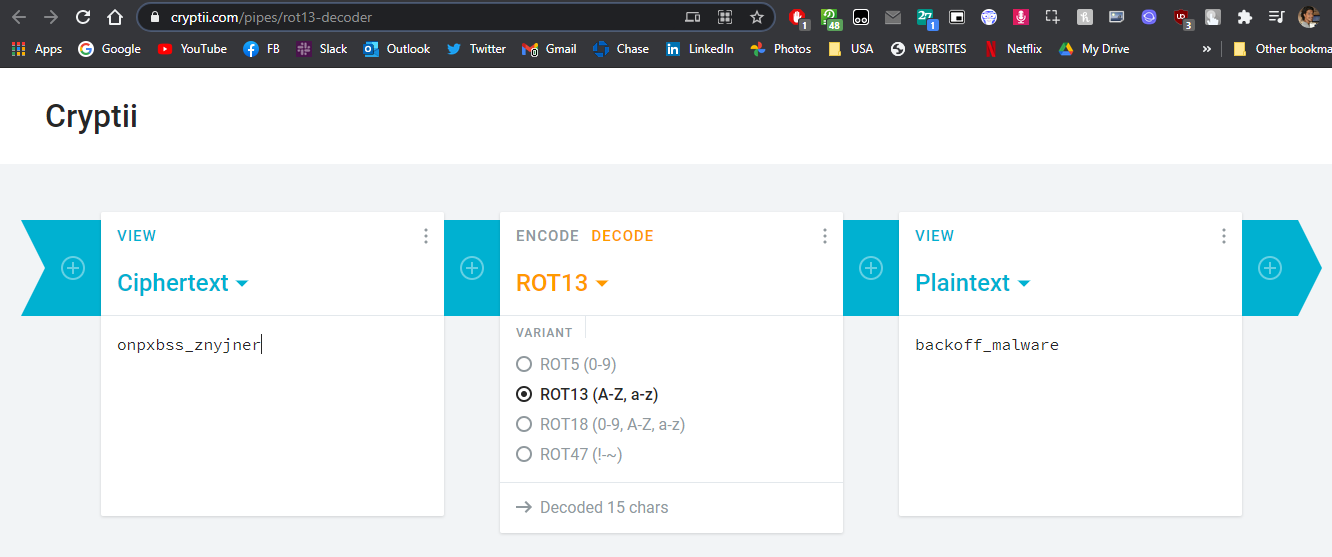
# Explanation of Approaches

Include descriptions along with screen shots of the approaches you took to solve these challenges.

## Cryptography

### Level 1 – Orange Julius

* To decrypt the given text, there exists a website <https://cryptii.com/pipes/rot13-decoder>
* Paste the given code in the ciphertext section to find out that the decrypted text is **backoff\_malware**

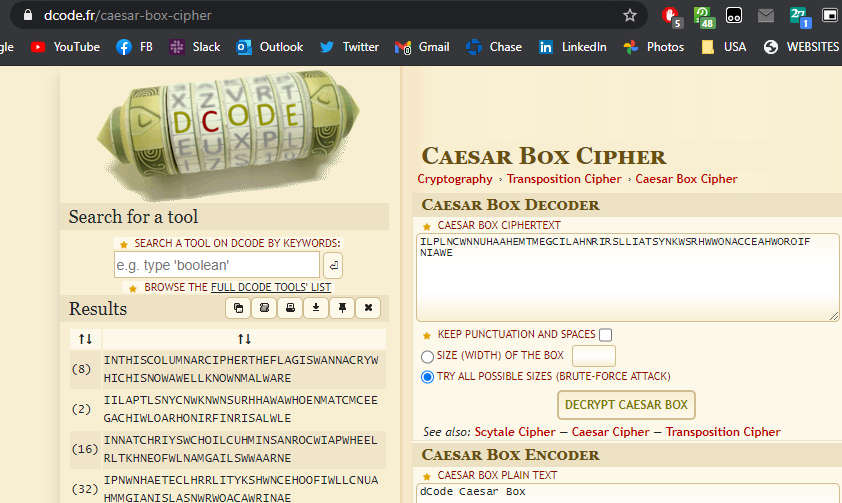


### Level 2 – Block Chain

* The given block is ILPLNCWNNUHAAHEMTMEGCILAHNRIRSLLIATSYNKWSRHWWONACCEAHWOROIFNIAWE
* To find the hidden malware in the above phrase, we arrange the given phrase in a chess board. We use chess board because the instruction has the phrase “the pawn can beat the king” indicating that this might have something to do with chess.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| I | L | P | L | **N** | C | W | N |
| N | U | H | A | **A** | H | E | M |
| T | M | E | G | **C** | I | L | A |
| H | N | R | I | **R** | S | L | L |
| I | A | T | S | **Y** | N | K | W |
| S | R | H | **W** | W | O | N | A |
| C | C | E | **A** | H | W | O | R |
| O | I | F | **N** | I | A | W | E |

* As seen in the above board, we have a hidden name **WANNACRY**. This could be a potential malware name.
* To cross check this, we visit the site <dcode.fr/caesar-box-cipher> and enter the given phrase in the box and click decrypt.



* From the results, we see the first result as follows

INTHISCOLUMNARCIPHERTHEFLAGIS**WANNACRY**WHICHISNOWAWELLKNOWNMALWARE

* Here also, we see that the hidden malware name is **WANNACRY**

### Level 3 – DASH DOT COM

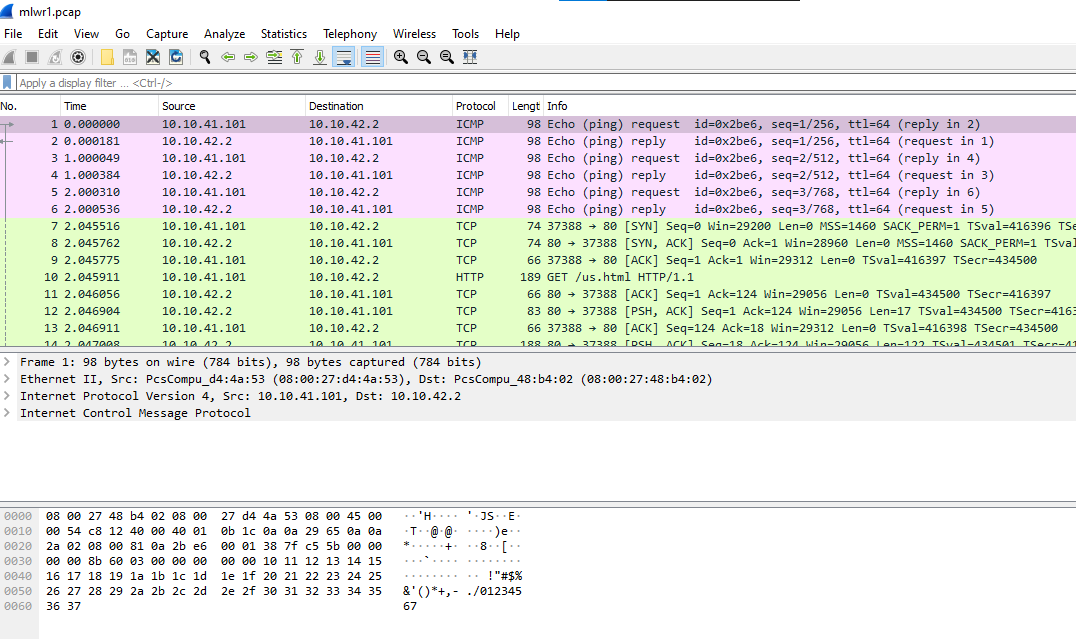
### Level 4 – VIII A Small Example

### Level 5 – Big Blue Randu

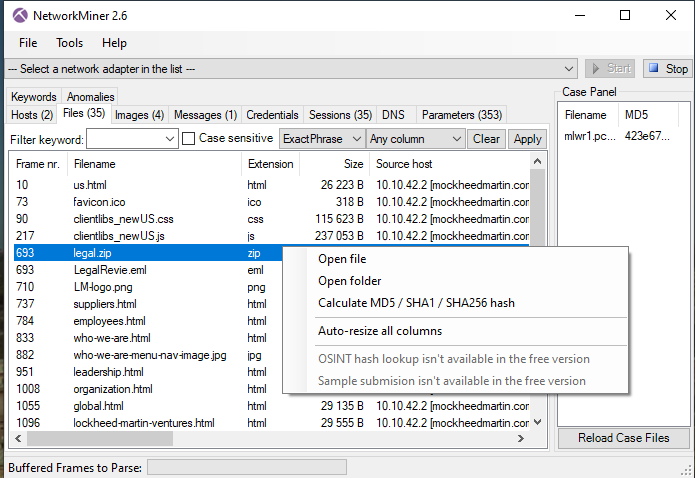
## Malware

### Level 1

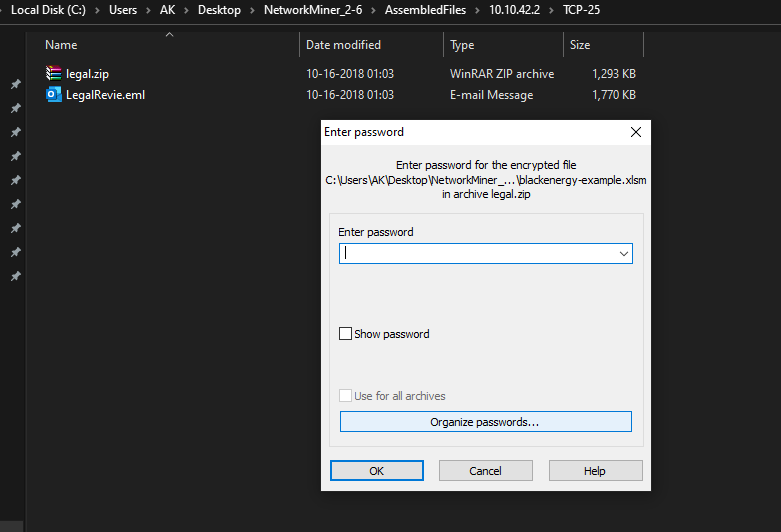
* The given pcap file for the below levels of tasks is shown below. And there is no flag value explicitly found on searching in this pcap file using **Wire Shark**  tool. There we try to use another software **Network Miner** which is also another packet capturing tool and try to analyze using that.



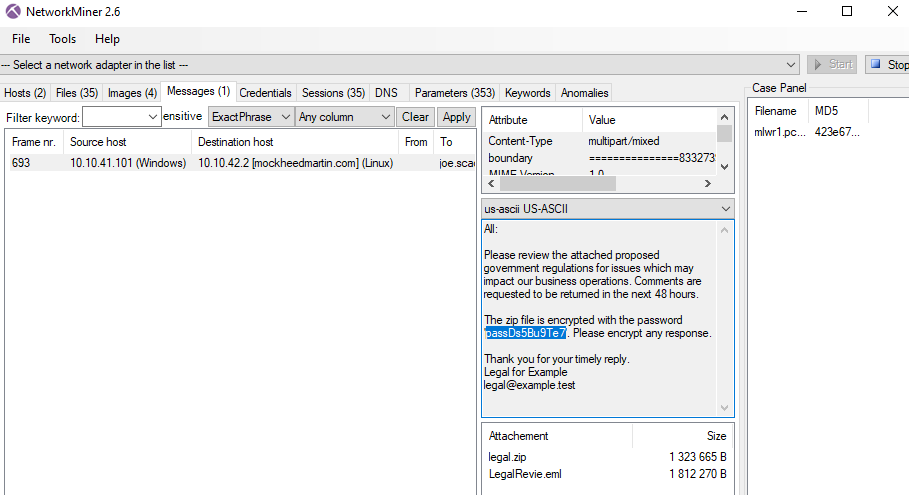
* We use the Network Miner application and open the pcap file. After this, we navigate to Files tab and we find a zip file named **legal.zip**. Right click and select **Open Folder**



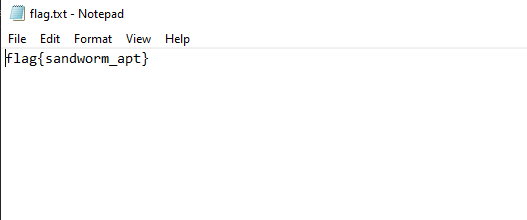
* On trying to access the zip file, it is asking for password, since it is encrypted



* To find the password, we navigate to **Messages** tab of Network Miner application. In there, we have the below information of which files have been encrypted and the password details for the same.

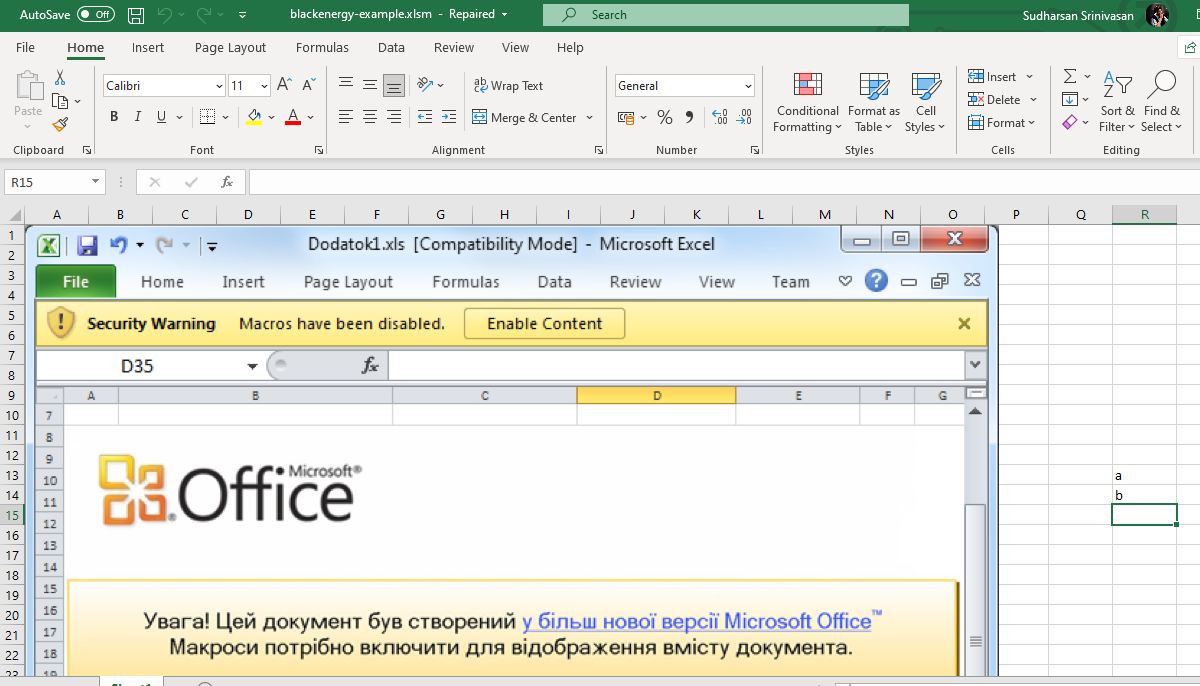


* After using the above password to unzip the folder, we have a file named **flag.txt** inside which has the below contents in it. Therefore, the hidden flag value here is **flag{sandworm\_apt}**



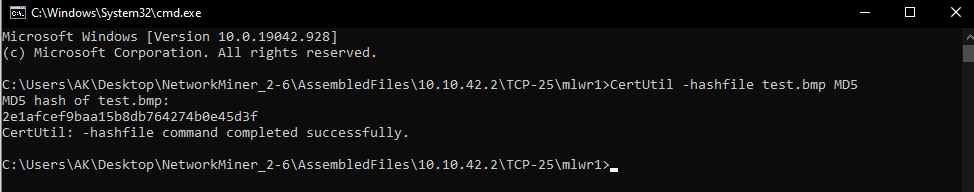
### Level 2

* The given excel file is opened and then when Macros is enabled, another bmp file gets opened.





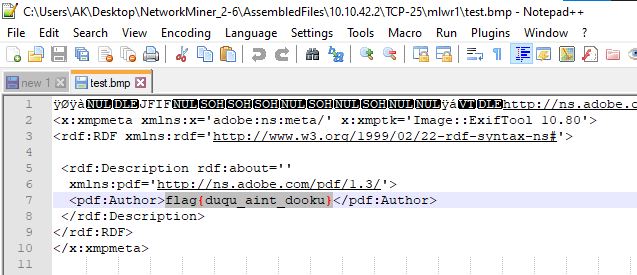
* This above file is saved as **test.bmp**. Now to find out the MD5 checksum value of this file, we open command prompt and navigate to the location where this file is present.
* Then we enter the command **CertUtil -hashfile test.bmp MD5**. This command is used to generate the cryptographic hash values of a files corresponding to different hash algorithms such as MD2, MD4, MD5 etc.,



* On using the command, we see that the MD5 hash value is **2e1afcef9baa15b8db764274b0e45d3f**

### Level 3

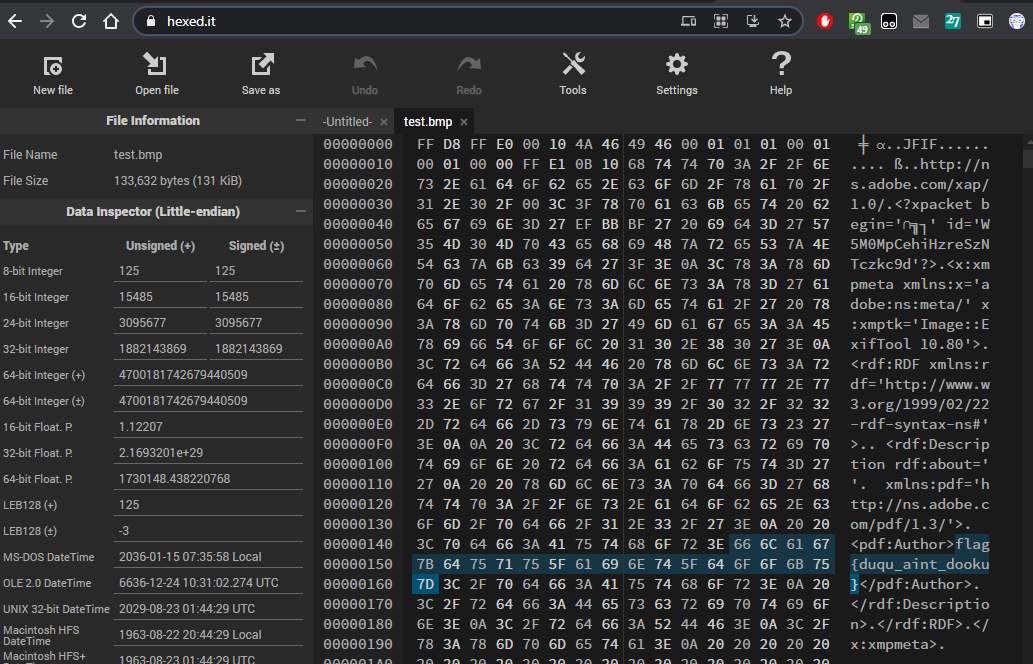
* In this task, we are required to find the flag value of the bmp file. To do so, open the bmp file in any text editor such as Notepad++.



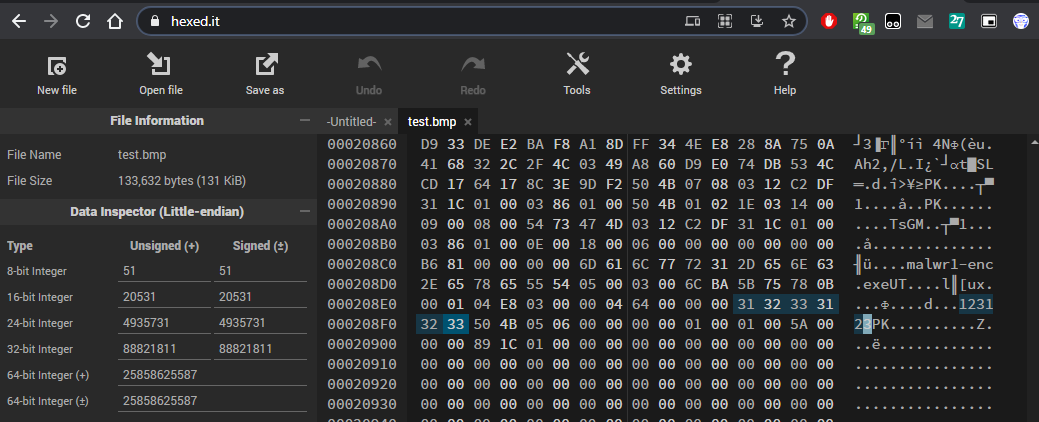
* As seen above, the hidden flag value is **flag{duqu\_aint\_dooku}**

### Level 4

* The aim here is to decrypt the bmp file and find the password. To do so, open the file in an online editor such as <hexed.it>

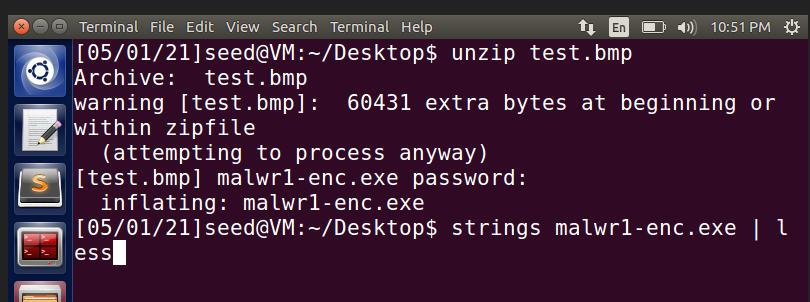


* When opened in an online editor, as we navigate through the file, we see the same flag value present indicating the value we found in the previous level.
* As we scroll down through the file, we see a value **malwr1-enc.exe** followed by the value 123123 which is the password. Therefore, the hidden flag value here is **123123**

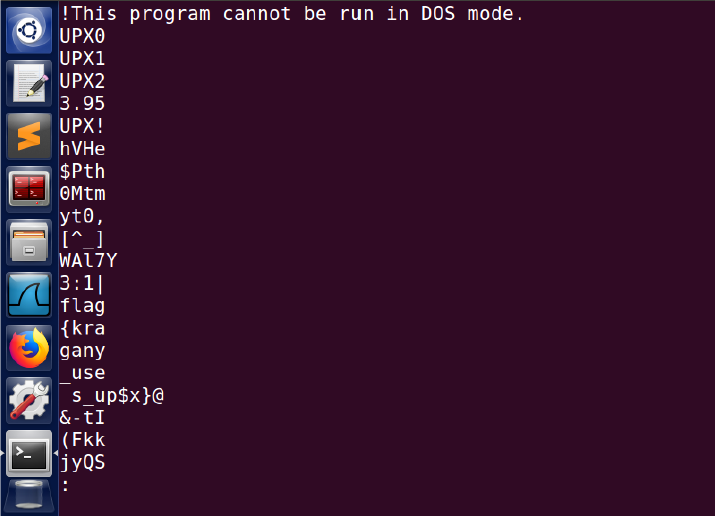


### Level 5

* We now have to find the final flag associated with **malwr1-enc.exe** found in test.bmp. To do so, unzip the test.bmp file. When prompted, give the password **123123**



* After this, we run the command **strings** on the extracted exe file malwr1-enc.exe

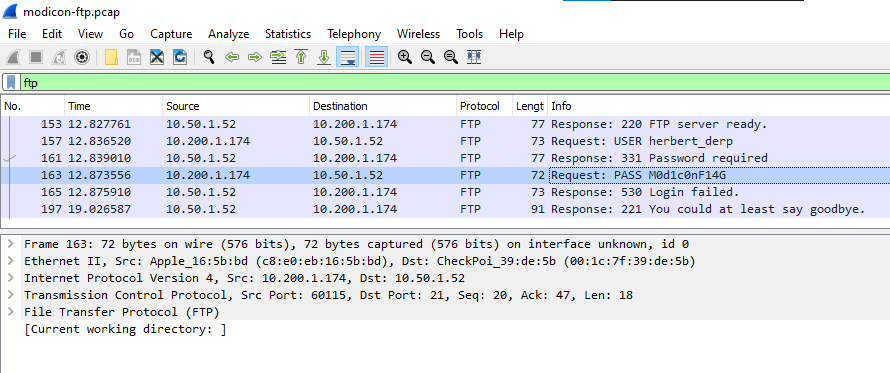


* As seen here, we have the flag value shown here as **flag{kragany\_uses\_up$x}** which is the required flag value.

## Network Capture

### Level 1 – Modicon-FTP

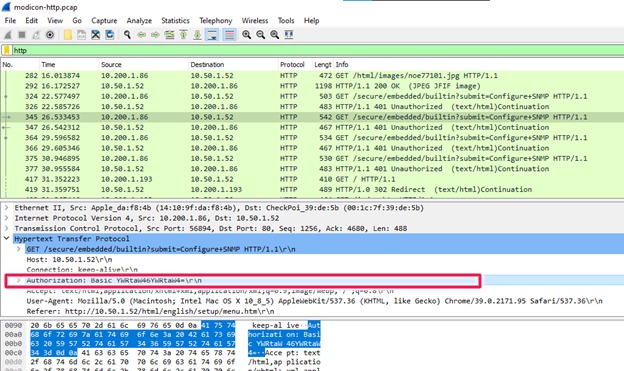
* Here, we need to find the password of FTP used in the failed attempt. To do so, first open the given pcap file and filter records based on FTP protocol.



* As seen here above, we have the password column with the value we are looking for, **M0d1c0nF14G**

### Level 2 – Modicon-HTTP

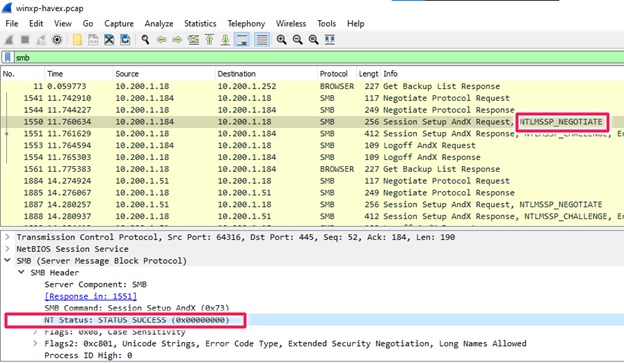
* This is same as the previous where we are required to find the failed login attempt value incase of HTTP. The given pcap file is opened and filtered in terms of HTTP.



* As highlighted above, we have the password that we used to attempt a login. We know that it is a failed attempt because the GET request passed with this password ends up giving an error **HTTP/1.1 401 Unauthorized**, meaning it is not the right password. However, since we want only the password used, it is **YWRtaW46YWRtaW4=**

### Level 3 – WinXP-HAVEX

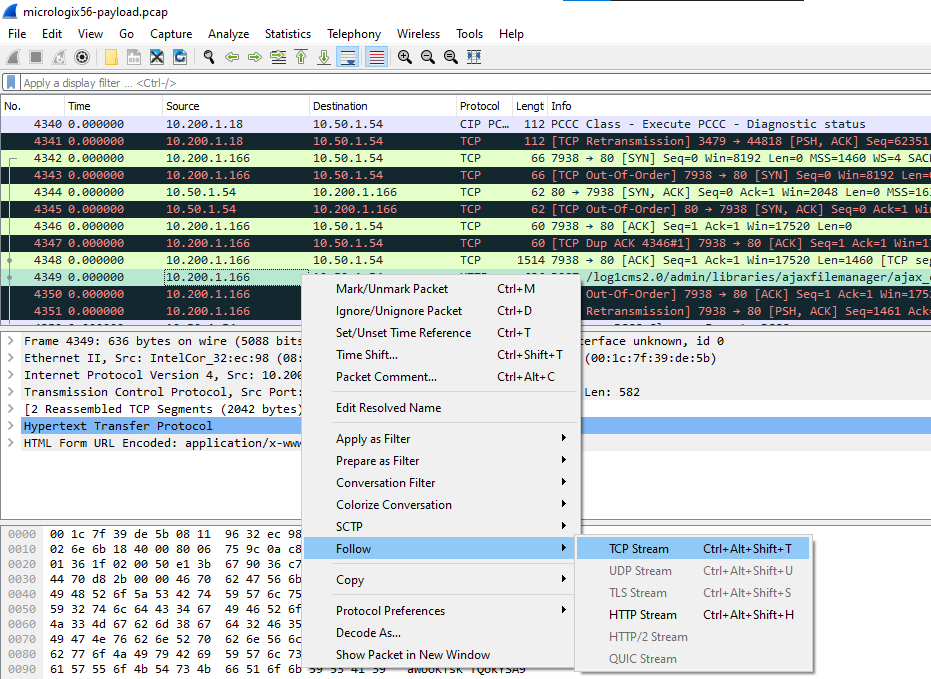
* We are required to find havex malware IOC value here. Open the pcap file, then filter the records in terms of **SMB** protocol, because this protocol provides unrestricted access to a machine, despite the threats involved.



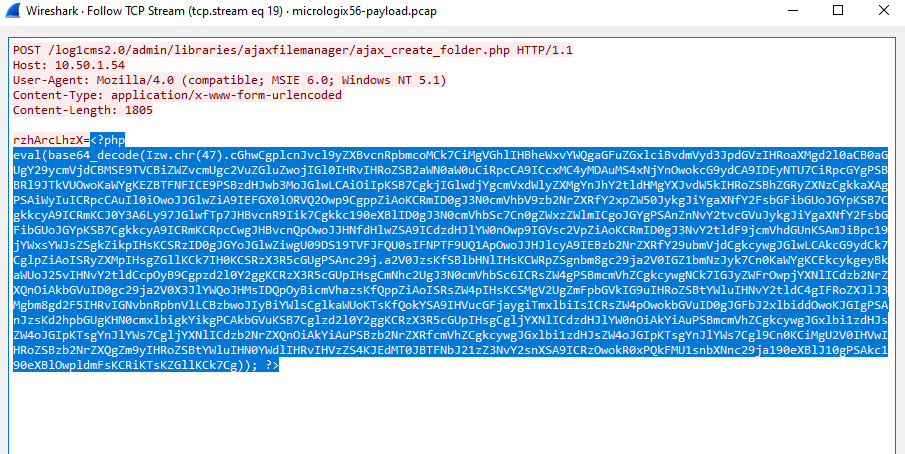
* As indicated in the above image, we have NT Status as **STATUS SUCCESS (0x00000000)** indicating that unauthorized access has been granted. The corresponding IOC value is **NTLMSSP\_NEGOTIATE.**

### Level 4 – MICROLOGIX56

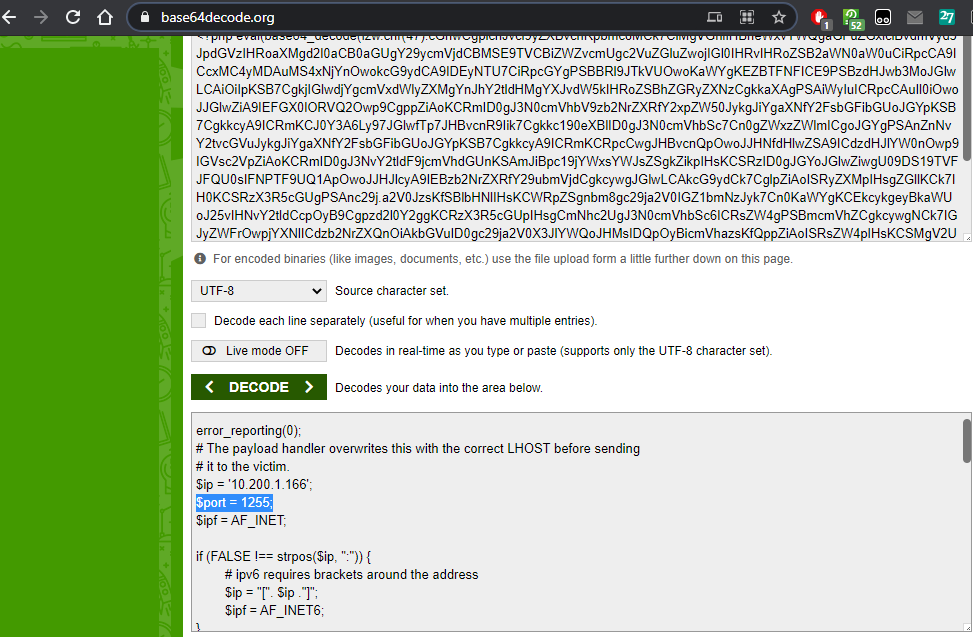
* We need to find the port number used by the attacker. First, open the pcap file, there exists a HTTP protocol in the stream, click on that and Follow that stream as shown below.



* On following the stream, we have a php code there as part of that stream. We analyze this php code now, to find any outcomes.



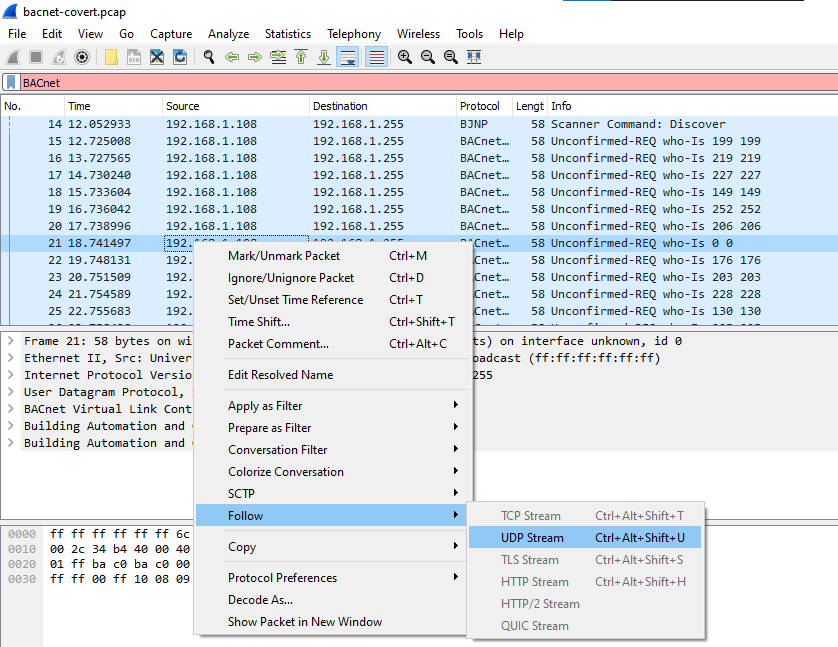
* Navigate to any website to decode the script such as <base64decode.org>, paste the script and observe.



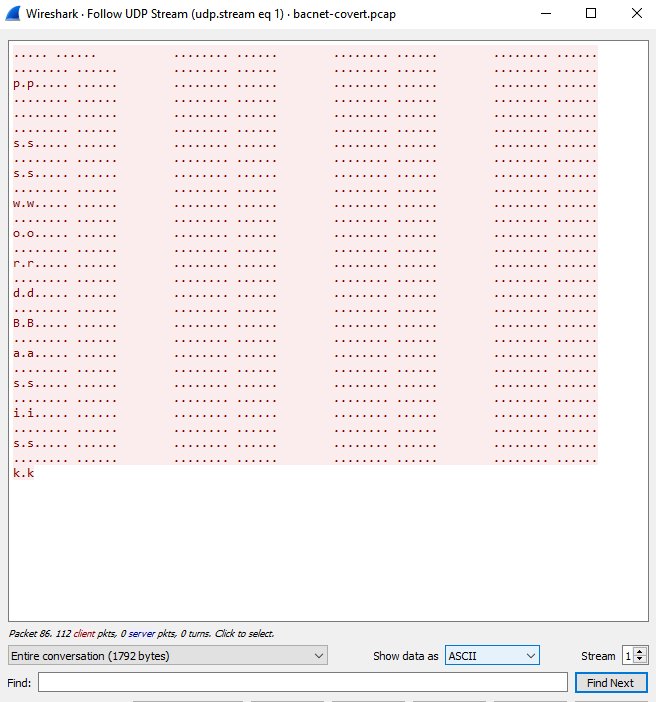
* On decoding the above code, we see that there is a $PORT variable with value **1255.** This is the port number used by the attacker backdoor to attempt an install.

### Level 5 – BACNET-COVERT

* Here, we need to find the covert message passed using BACNET protocol. Open the given cap file, filter records based on BACnet protocol.



* We have a record **Unconfirmed-REQ who-Is 0 0** which could potentially contain the message. So, we follow this UDP stream as shown above.

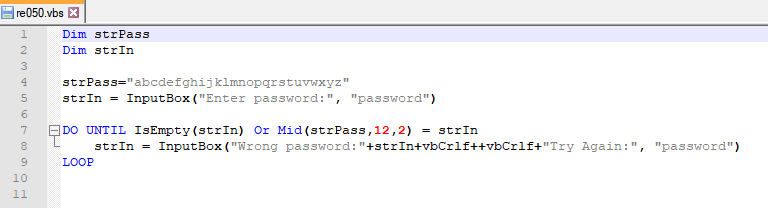


* On following the stream, we have the following value **passwordBasisk** which is the covert message we are looking for.

## Reverse Engineering

### Level 1 – Script Kiddie

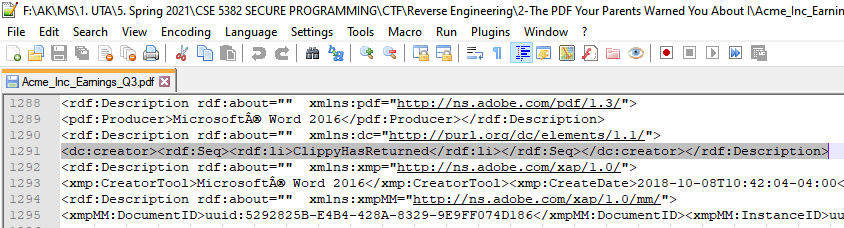
* The aim here is to read the VBScript script file given and locate the password. We open the file using Notepad++ and observe.



* We see above that we have values **12,2** indicating that the password in strPass string starts from 12th position and is of length 2. Therefore, the password flag value is **lm**

### Level 2 – The PDF Your Parents Warned You About I

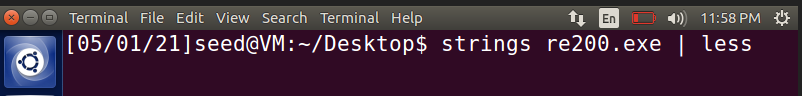
* A pdf file is given, and we are required to find who created it. We open the pdf in Notepad++ and try to observe if we have any values or tags corresponding to the author of the pdf.

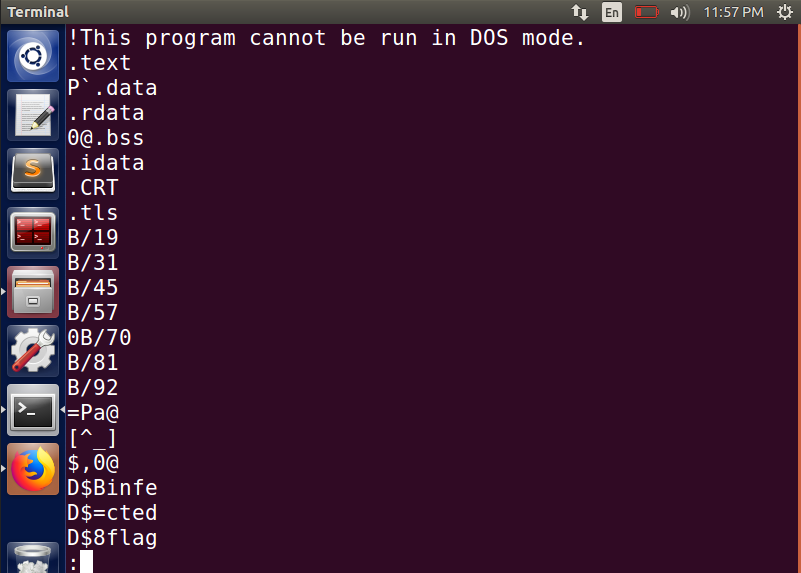


* We see here that in line 1291, we have a **creator** tag with a value for it. Therefore, the creator name **ClippyHasReturned** has created this PDF file.

### Level 3 – Split Ends

* We are given an exe file and we need to find the compiled flag value of it. To do so, we use **strings** command followed by the given file, re200.exe and observe what happens when the program runs.

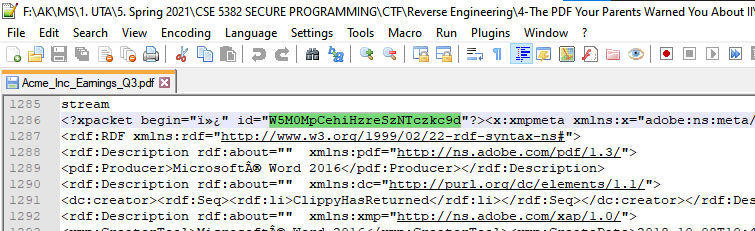




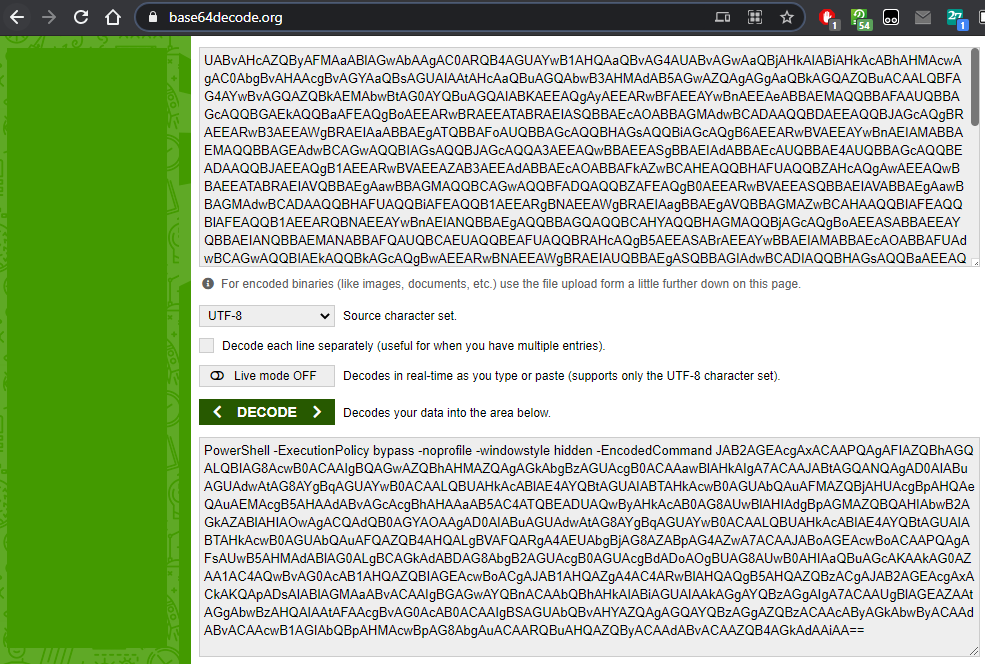
* On running the exe file, we see a message split into parts and then it ends. This could indicate the SPLIT ENDS and therefore the compiled flag value is **infected flag**

### Level 4 – The PDF Your Parents Warned You About II

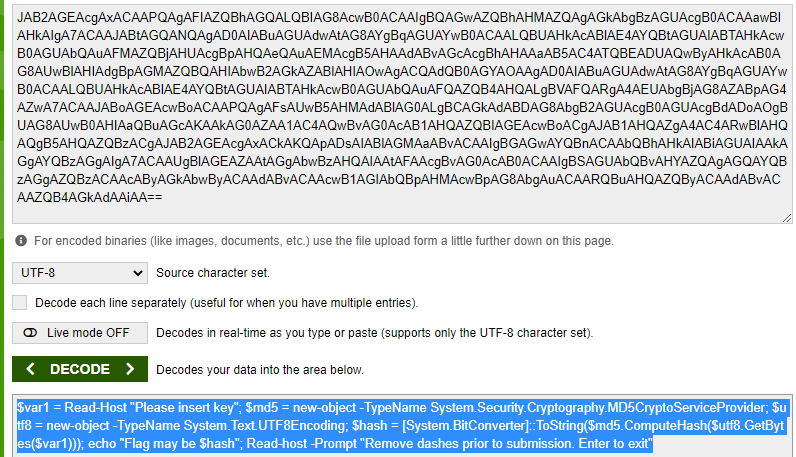
* We need to find the hidden key value from the given pdf. Open the pdf using Notepad++. On doing so, we see that there is an ID field with a value **W5M0MpCehiHzreSzNTczkc9d**



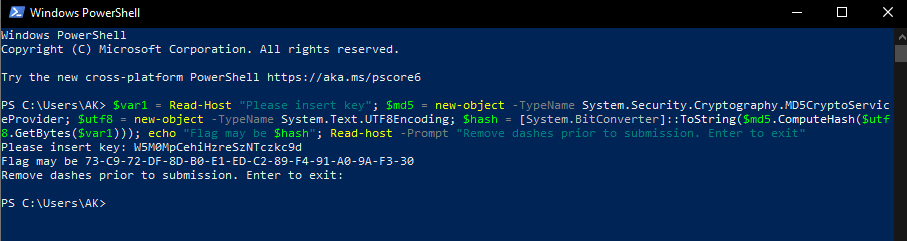
* As we navigate to the end of the file, we have a powershell path specified along with a long-encoded string. We copy the string and decode it using online decoder we used earlier.
* On trying to decode it, we get another lengthy string. We try to decode it too.



* On decoding the second string, we get the below result, which looks like a PHP code.



* Since we already saw at the end of the file that we had a powershell path, we paste this code in powershell and run.



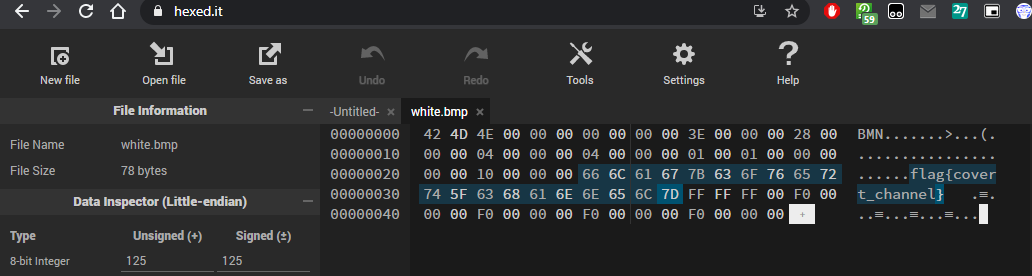
* When prompted with key, we enter the ID value we observed earlier in the PDF. On giving the ID value, we see that we get a flag value as **73-C9-72-DF-8D-B0-E1-ED-C2-89-F4-91-A0-9A-F3-30**. After removing the dashes, the flag value is **73C972DF8DB0E1EDC289F491A09AF330**

### Level 5 – Rock Scissors Paper Lizard Spock

## Steganography

### Level 1 – Moo Cow

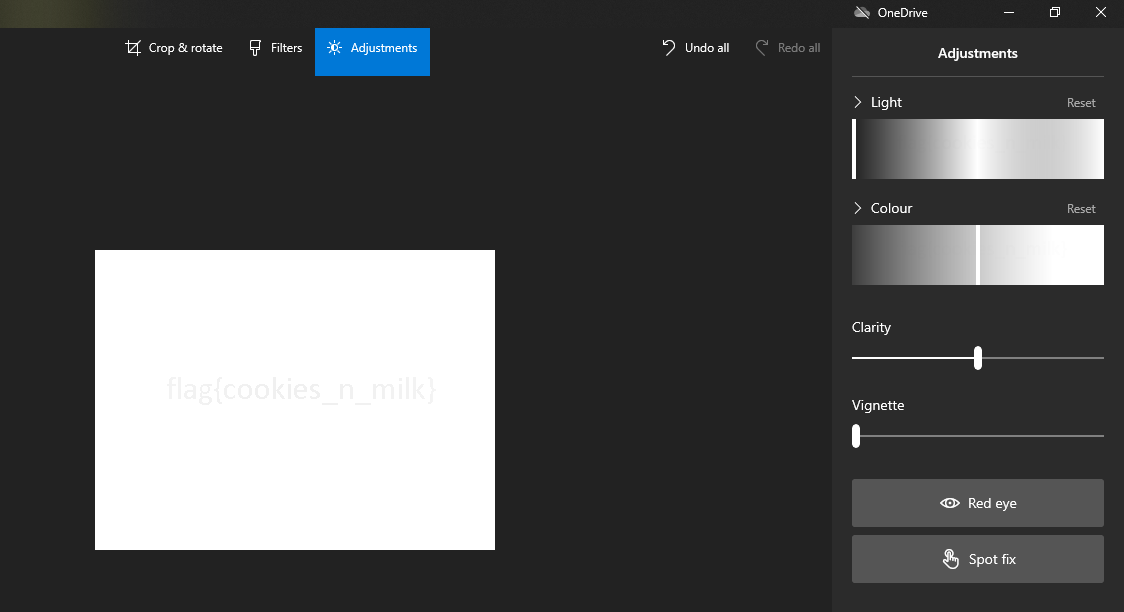
* We are required to find the flag value corresponding to the cow from the given bmp file. On opening the file in an online hex editor, we observe the below.



* As seen in the above result, we get the flag value as **flag{covert\_channel}**

### Level 2 – Milk Run

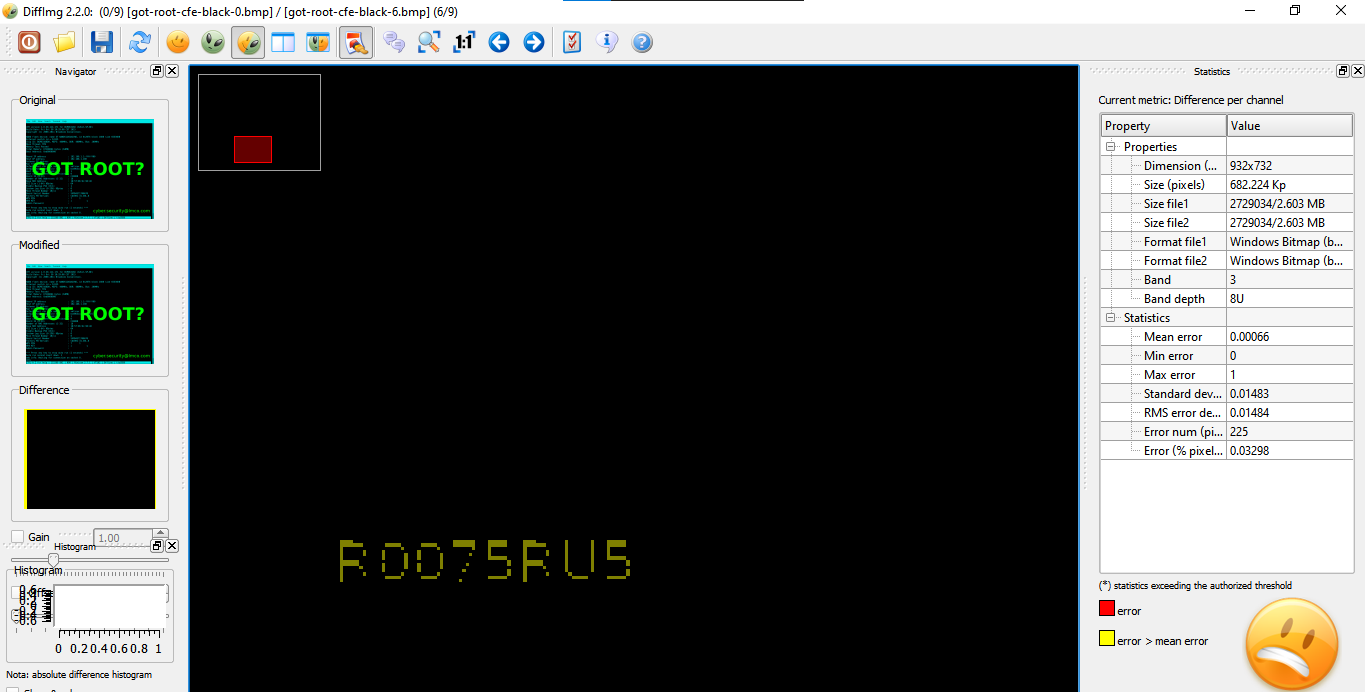
* We are asked to find the flag value present in the bmp file. We open the white bmp file in Photos application and try to make adjustments to the image.



* On reducing the light effect to -100 and reducing the Vignette value, we are able to see the flag value **flag{cookies\_n\_milk}** inside the image file.

### Level 3 – Got Root

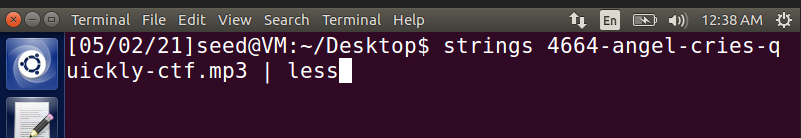
* We have to find if there is any difference in all the images given and if any password is hidden in any of the files. To do so, we first open the images normally and see that all the images are identical.
* Now, we use **DiffImg** software to find difference between images. On constantly comparing one image with another, we find that on comparing **Image 0** and **Image 6**, we see the following.



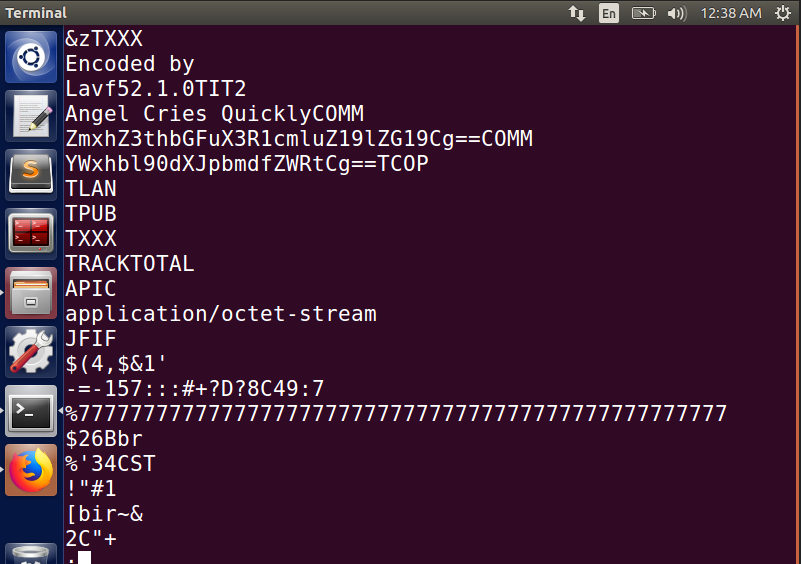
* We see that there is a password flag value shown here **R0075RUS**. This is the value we are looking for.

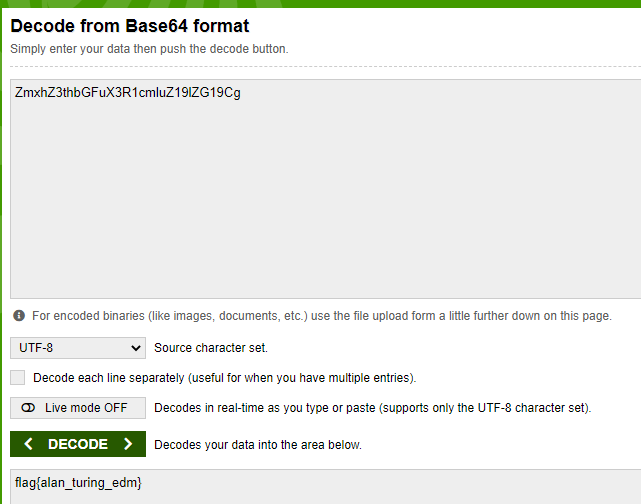
### Level 4 – AI EDM

* A mp3 file is given and we need to find the flag value hidden in it. We first use the command **strings** followed by the mp3 file name and run the file.



* On observing the following program when it is run, we see that there is a string present in it. We decode the string **ZmxhZ3thbGFuX3R1cmluZ19lZG19Cg** to see if it contains any message or flag.





* The hidden flag value present in the mp3 is **flag{alan\_turing\_edm}**

### Level 5 – Final Frontier