CSE 5382: Secure Programming – Spring 2019 Student Name: **Rishabh Agrawal**
Capture the Flag Bonus Exercise Student ID: **1001623226**

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)	flag{backoff_malware}	passDs5Bu9Te7	M0d1c0nF14G	lm	flag{covert_channel}
Level 2 (3 Points)	Wanna Cry ransomware	NA	admin, admin	ClippyHasReturned	Cookies_n_milk
Level 3 (4 Points)	flagslashdotdash	NA	HaveX RAT	Flag{**infected**}	6th file, R0075RU5
Level 4 (5 Points)	did cicero say anything	NA	Port 1255	c97ff22e75b6cf465b70ae0dddc8e773	Angel Cries Quickly
Level 5 (6 Points)		NA	p.sswordBasisk	flag{.}	Comment attribute: **"nope"**

Explanation of Approaches

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

Cryptography

Level 1 – Orange Julius: Text: "**flag{backoff_malware}**" with key shift of 13. Used website: https://www.xarg.org/tools/caesar-cipher/ to decipher the text. The character shift was: 13.

Level 2 – Block Chain

The decoded string from the given cipher is:
INTHISCOLUMNARCIPHERTHEFLAGISWANNACRYWHICHISNOWAWELLKNOWNMALWARE

Wanna Cry Ransomware

Level 3 – DASH DOT COM

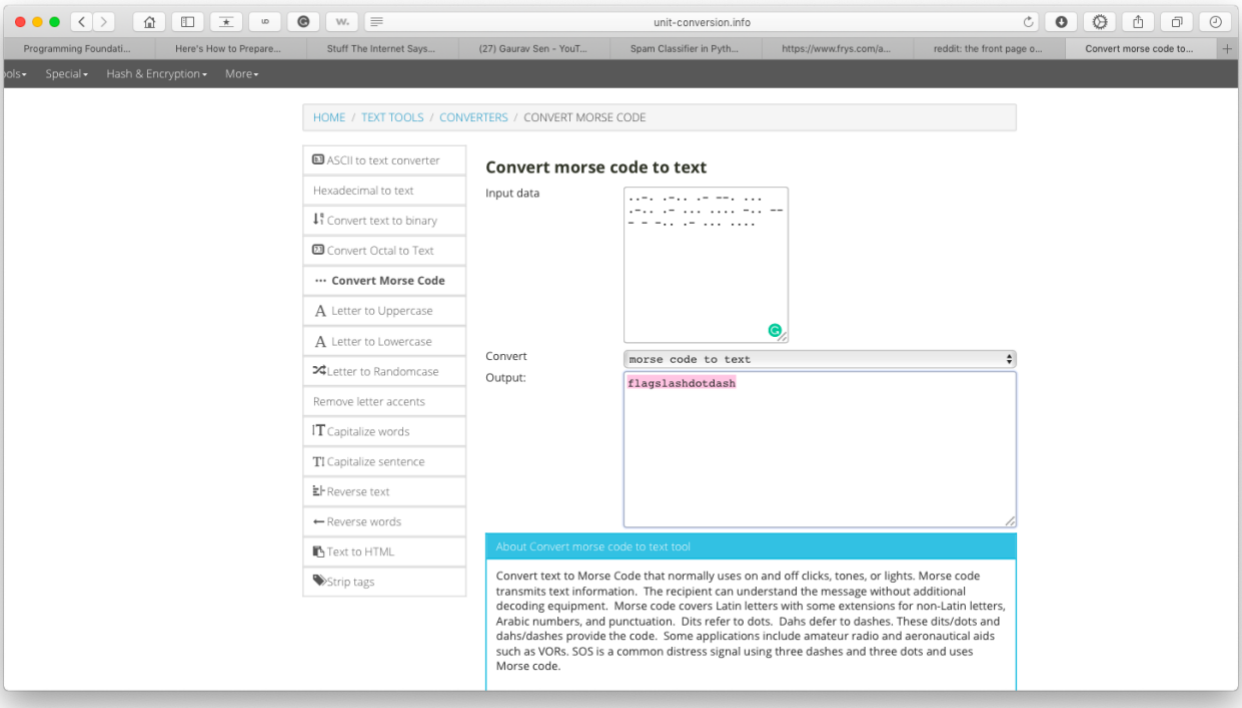
The flag is: **flagslashdotdash**
The morse code is: **..-. .-.. .- --. ... .-.. .- ... . -.. --- - -.. .- ... .**

The only variation found in the logs was that of the request type and the resource requested. Sometimes, the request specifies the image format other times it does not.

The apache log file was a morse code for the flag. The logs were converted into the morse code which was then a simple conversion process from morse code to human readable string text format.

Screenshots are attached below:


3-DASH DOT COM — -bash — 80×24
[Rishabhs-MacBook-Air:3-DASH DOT COM rishabh$ cat access.log | awk '{print $6 $10]
;}' | tr -d '\n' | sed -e 's/"POST207/\n/g' -e 's/"GET//g' -e 's/211/./g' -e 's/
212/-/g' -e 's/207/ /g'
..-. .-.. .- --.n... .-.. .- ... .n-.. --- -n-.. .- ... .n
Rishabhs-MacBook-Air:3-DASH DOT COM rishabh$


unit-conversion.info
Programming Foundati...
Here's How to Prepare...
Stuff The Internet Says...
(27) Gaurav Sen - YouT...
Spam Classifier in Pyth...
https://www.frys.com/a...
reddit: the front page o...
Convert morse code to...
Special
Hash & Encryption
More
HOME / TEXT TOOLS / CONVERTERS / CONVERT MORSE CODE
ASCII to text converter
Hexadecimal to text
Convert text to binary
Convert Octal to Text
Convert Morse Code
Letter to Uppercase
Letter to Lowercase
Letter to Randomcase
Remove letter accents
Capitalize words
Capitalize sentence
Reverse text
Reverse words
Text to HTML
Strip tags
Convert morse code to text
Input data
..-. .-.. .- --. ...
.-.. .- ... . -.. --
- - -.. .- ... .
Convert
morse code to text
Output:
flagslashdotdash
About Convert morse code to text tool
Convert text to Morse Code that normally uses on and off clicks, tones, or lights. Morse code transmits text information. The recipient can understand the message without additional decoding equipment. Morse code covers Latin letters with some extensions for non-Latin letters, Arabic numbers, and punctuation. Dits refer to dots. Dahs defer to dashes. These dits/dots and dahs/dashes provide the code. Some applications include amateur radio and aeronautical aids such as VORs. SOS is a common distress signal using three dashes and three dots and uses Morse code.

Level 4 – VIII A Small Example

“did cicero say anything”
Using RSA decyprion process, values: d=157, n=2773
Message, M= (4 bits)^157 mod 2773

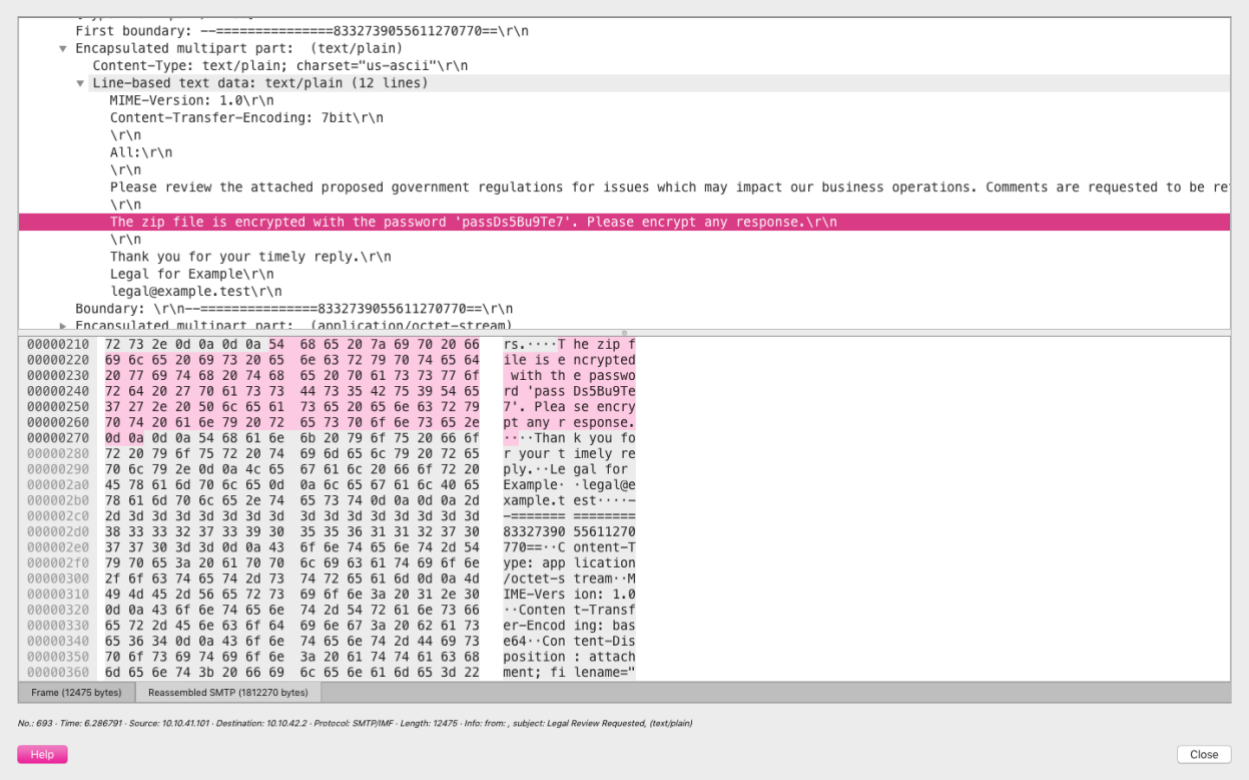
Level 5 – Big Blue Randu

Malware

Level 1: MLWR-1-PCAP

Using Wireshark packet forensics techniques

Extract file password: passDs5Bu9Te7



Level 2:

Not provided

Level 3:

Not provided

Level 4:

Not provided

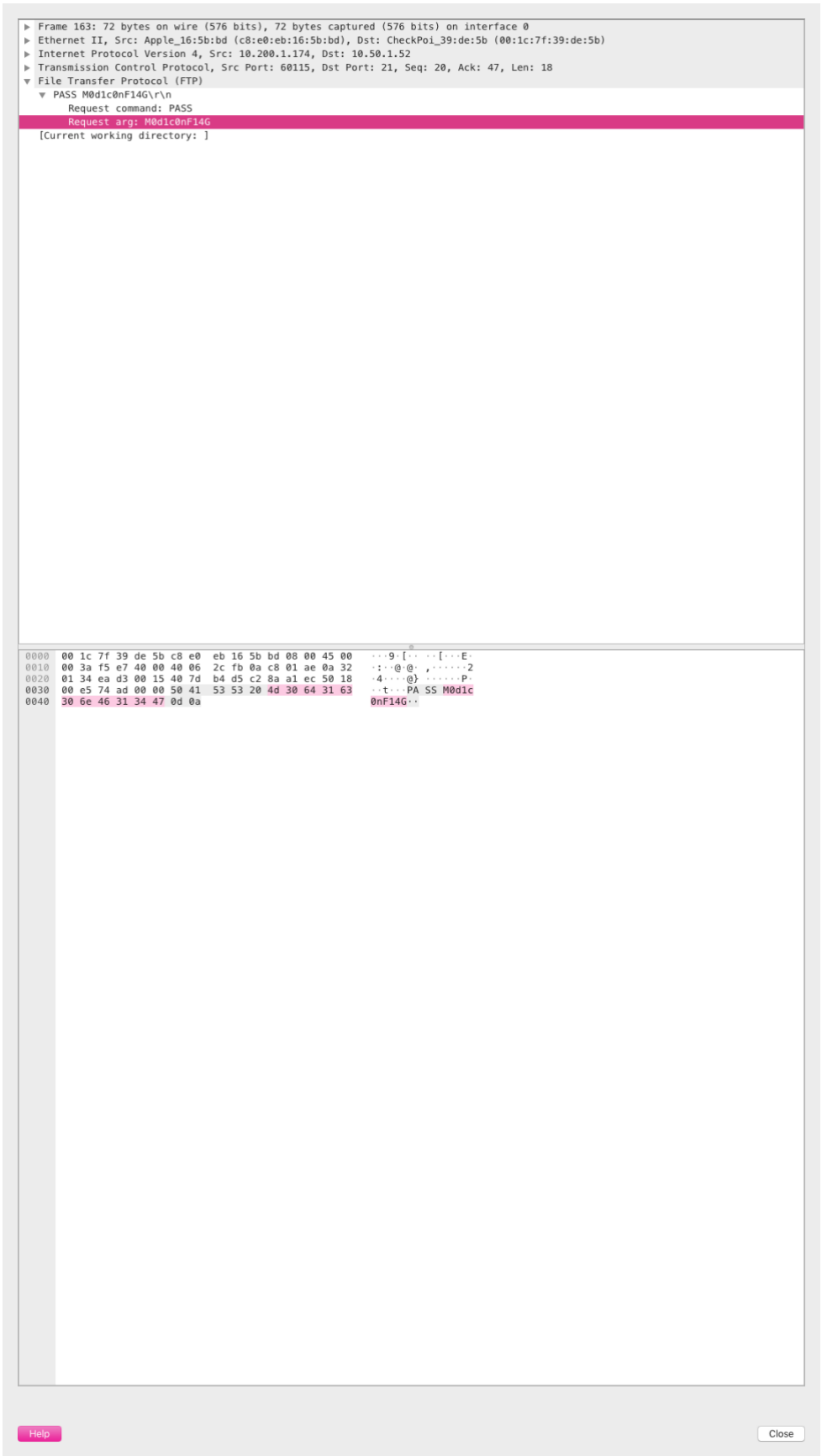
Level 5:

Not provided

Network Capture

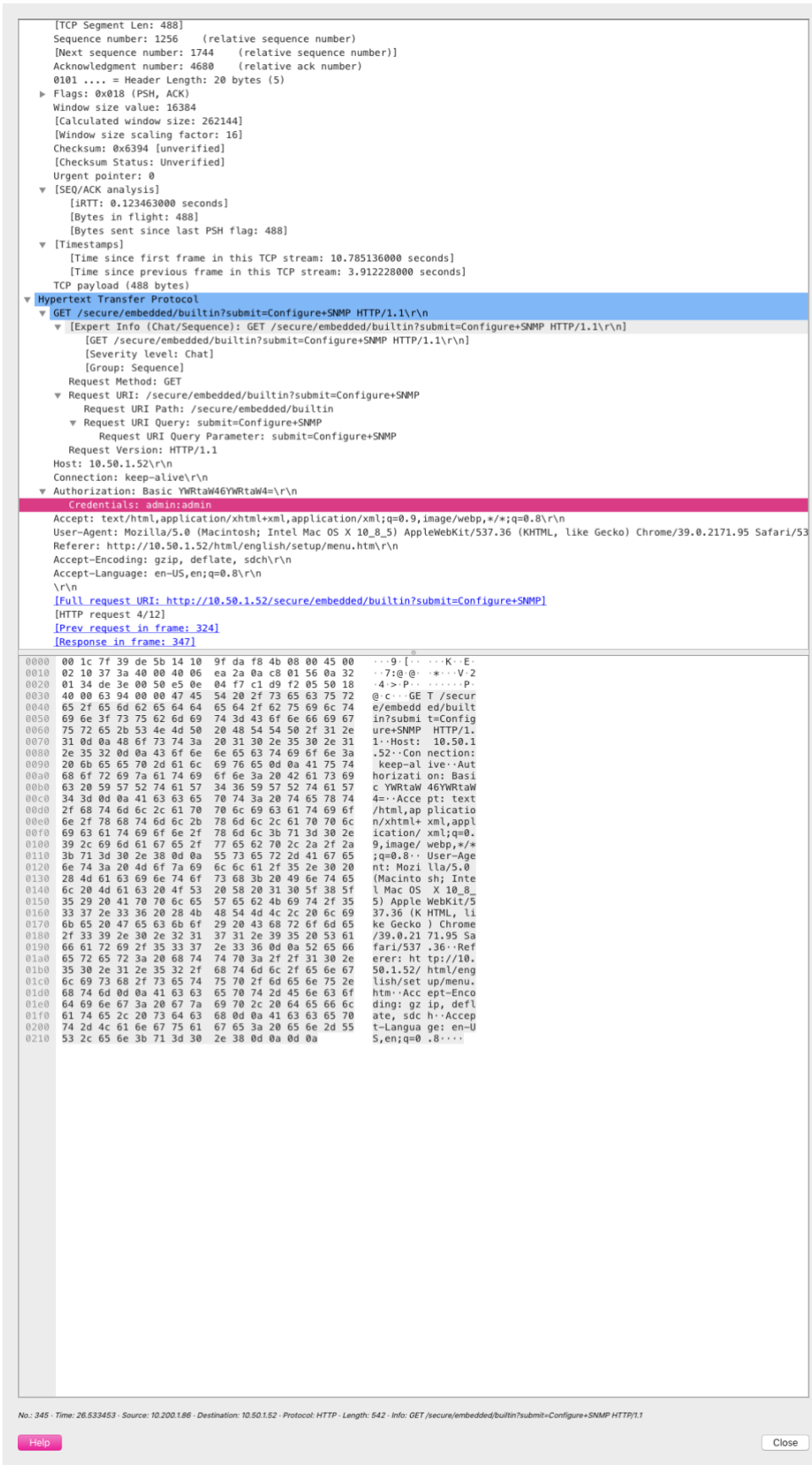
Level 1 – Modicon-FTP

Password is: M0d1c0nF14G


▶ Frame 163: 72 bytes on wire (576 bits), 72 bytes captured (576 bits) on interface 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)
▼ PASS M0d1c0nF14G\r\n
Request command: PASS
Request arg: M0d1c0nF14G
[Current working directory:]
0000 00 1c 7f 39 de 5b c8 e0 eb 16 5b bd 08 00 45 00
0010 00 3a f5 e7 40 00 40 06 2c fb 0a c8 01 ae 0a 32
0020 01 34 ea d3 00 15 40 7d b4 d5 c2 8a a1 ec 50 18
0030 00 e5 74 ad 00 00 50 41 53 53 20 4d 30 64 31 63
0040 30 6e 46 31 34 47 0d 0a
Help
Close

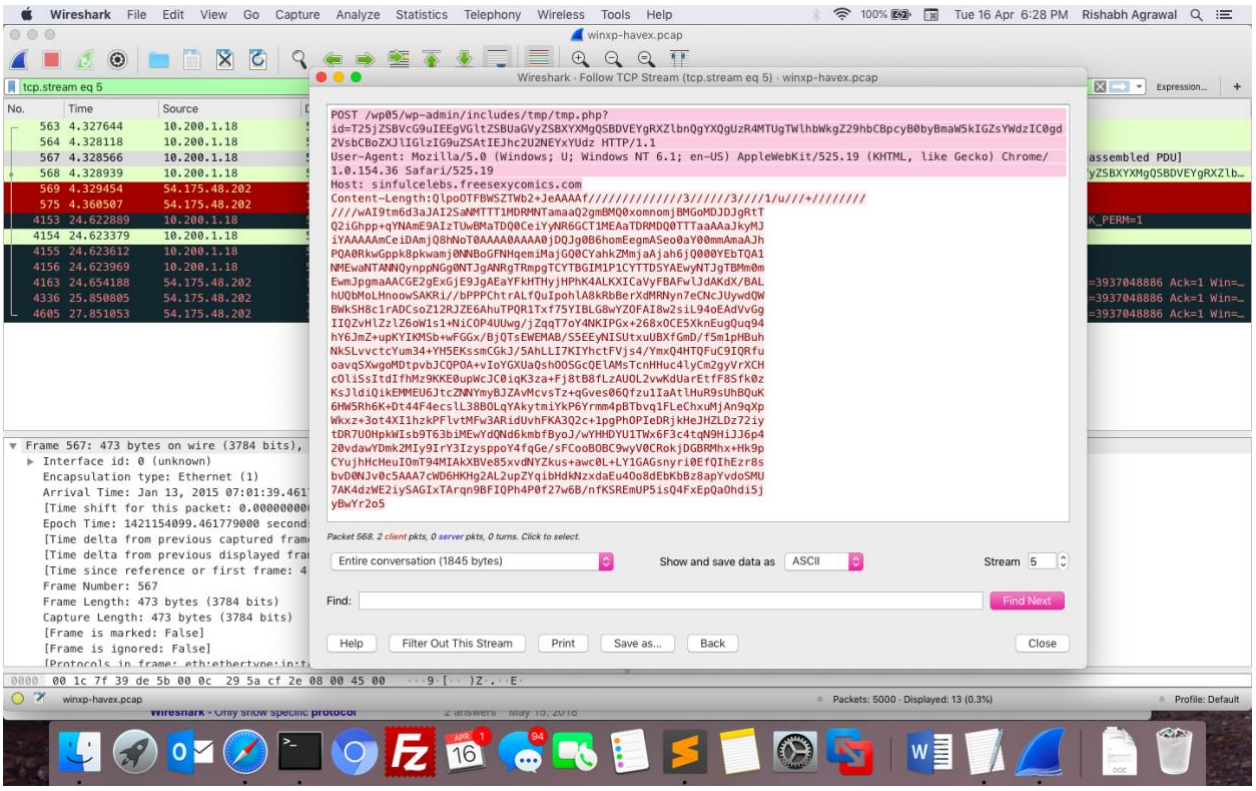
Level 2 – Modicon-HTTP

Password is: admin, username:admin


[TCP Segment Len: 488]
Sequence number: 1256 (relative sequence number)
[Next sequence number: 1744 (relative sequence number)]
Acknowledgment number: 4680 (relative ack number)
0101 .... = Header Length: 20 bytes (5)
Flags: 0x018 (PSH, ACK)
Window size value: 16384
[Calculated window size: 262144]
[Window size scaling factor: 16]
Checksum: 0x6394 [unverified]
[Checksum Status: Unverified]
Urgent pointer: 0
[SEQ/ACK analysis]
[iRTT: 0.123463000 seconds]
[Bytes in flight: 488]
[Bytes sent since last PSH flag: 488]
[Timestamps]
[Time since first frame in this TCP stream: 10.785136000 seconds]
[Time since previous frame in this TCP stream: 3.912228000 seconds]
TCP payload (488 bytes)
Hypertext Transfer Protocol
GET /secure/embedded/builtin?submit=Configure+SNMP HTTP/1.1\r\n
[Expert Info (Chat/Sequence): GET /secure/embedded/builtin?submit=Configure+SNMP HTTP/1.1\r\n]
[GET /secure/embedded/builtin?submit=Configure+SNMP HTTP/1.1\r\n]
[Severity level: Chat]
[Group: Sequence]
Request Method: GET
Request URI: /secure/embedded/builtin?submit=Configure+SNMP
Request URI Path: /secure/embedded/builtin
Request URI Query: submit=Configure+SNMP
Request URI Query Parameter: submit=Configure+SNMP
Request Version: HTTP/1.1
Host: 10.50.1.52\r\n
Connection: keep-alive\r\n
Authorization: Basic YWRtaW46YWRtaW4=\r\n
Credentials: admin:admin
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;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/53
Referer: http://10.50.1.52/html/english/setup/menu.htm\r\n
Accept-Encoding: gzip, deflate, sdch\r\n
Accept-Language: en-US,en;q=0.8\r\n
\r\n
[Full request URI: http://10.50.1.52/secure/embedded/builtin?submit=Configure+SNMP]
[HTTP request 4/12]
[Prev request in frame: 324]
[Response in frame: 347]
0000 00 1c 7f 39 de 5b 14 10 9f da f8 4b 08 00 45 00 ···9·[·· ···K··E·
0010 02 10 37 3a 40 00 40 06 ea 2a 0a c8 01 56 0a 32 ··7:@·@· ·*···V·2
0020 01 34 de 3e 00 50 e5 0e 04 f7 c1 d9 f2 05 50 18 ·4·>·P·· ·····P·
0030 40 00 63 94 00 00 47 45 54 20 2f 73 65 63 75 72 @·c···GE T /secur
0040 65 2f 65 6d 62 65 64 64 65 64 2f 62 75 69 6c 74 e/embedd ed/built
0050 69 6e 3f 73 75 62 6d 69 74 3d 43 6f 6e 66 69 67 in?submi t=Config
0060 75 72 65 2b 53 4e 4d 50 20 48 54 54 50 2f 31 2e ure+SNMP HTTP/1.
0070 31 0d 0a 48 6f 73 74 3a 20 31 30 2e 35 30 2e 31 1··Host: 10.50.1
0080 2e 35 32 0d 0a 43 6f 6e 6e 65 63 74 69 6f 6e 3a .52··Con nection:
0090 20 6b 65 65 70 2d 61 6c 69 76 65 0d 0a 41 75 74 keep-al ive··Aut
00a0 68 6f 72 69 7a 61 74 69 6f 6e 3a 20 42 61 73 69 horizati on: Basi
00b0 63 20 59 57 52 74 61 57 34 36 59 57 52 74 61 57 c YWRtaW 46YWRtaW
00c0 34 3d 0d 0a 41 63 63 65 70 74 3a 20 74 65 78 74 4=··Acce pt: text
00d0 2f 68 74 6d 6c 2c 61 70 70 6c 69 63 61 74 69 6f /html,ap plicatio
00e0 6e 2f 78 68 74 6d 6c 2b 78 6d 6c 2c 61 70 70 6c n/xhtml+ xml,appl
00f0 69 63 61 74 69 6f 6e 2f 78 6d 6c 3b 71 3d 30 2e ication/ xml;q=0.
0100 39 2c 69 6d 61 67 65 2f 77 65 62 70 2c 2a 2f 2a 9,image/ webp,*/*
0110 3b 71 3d 30 2e 38 0d 0a 55 73 65 72 2d 41 67 65 ;q=0.8·· User-Age
0120 6e 74 3a 20 4d 6f 7a 69 6c 6c 61 2f 35 2e 30 20 nt: Mozi lla/5.0
0130 28 4d 61 63 69 6e 74 6f 73 68 3b 20 49 6e 74 65 (Macinto sh; Inte
0140 6c 20 4d 61 63 20 4f 53 20 58 20 31 30 5f 38 5f l Mac OS X 10_8_
0150 35 29 20 41 70 70 6c 65 57 65 62 4b 69 74 2f 35 5) Apple WebKit/5
0160 33 37 2e 33 36 20 28 4b 48 54 4d 4c 2c 20 6c 69 37.36 (K HTML, li
0170 6b 65 20 47 65 63 6b 6f 29 20 43 68 72 6f 6d 65 ke Gecko) Chrome
0180 2f 33 39 2e 30 2e 32 31 37 31 2e 39 35 20 53 61 /39.0.21 71.95 Sa
0190 66 61 72 69 2f 35 33 37 2e 33 36 0d 0a 52 65 66 fari/537 .36··Ref
01a0 65 72 65 72 3a 20 68 74 74 70 3a 2f 2f 31 30 2e erer: ht tp://10.
01b0 35 30 2e 31 2e 35 32 2f 68 74 6d 6c 2f 65 6e 67 50.1.52/ html/eng
01c0 6c 69 73 68 2f 73 65 74 75 70 2f 6d 65 6e 75 2e lish/set up/menu.
01d0 68 74 6d 0d 0a 41 63 63 65 70 74 2d 45 6e 63 6f htm··Acc ept-Enco
01e0 64 69 6e 67 3a 20 67 7a 69 70 2c 20 64 65 66 6c ding: gz ip, defl
01f0 61 74 65 2c 20 73 64 63 68 0d 0a 41 63 63 65 70 ate, sdc h··Accep
0200 74 2d 4c 61 6e 67 75 61 67 65 3a 20 65 6e 2d 55 t-Langua ge: en-U
0210 53 2c 65 6e 3b 71 3d 30 2e 38 0d 0a 0d 0a S,en;q=0 .8····
No.: 345 · Time: 26.533453 · Source: 10.200.1.86 · Destination: 10.50.1.52 · Protocol: HTTP · Length: 542 · Info: GET /secure/embedded/builtin?submit=Configure+SNMP HTTP/1.1
Help
Close

Level 3 – WinXP-HAVEX

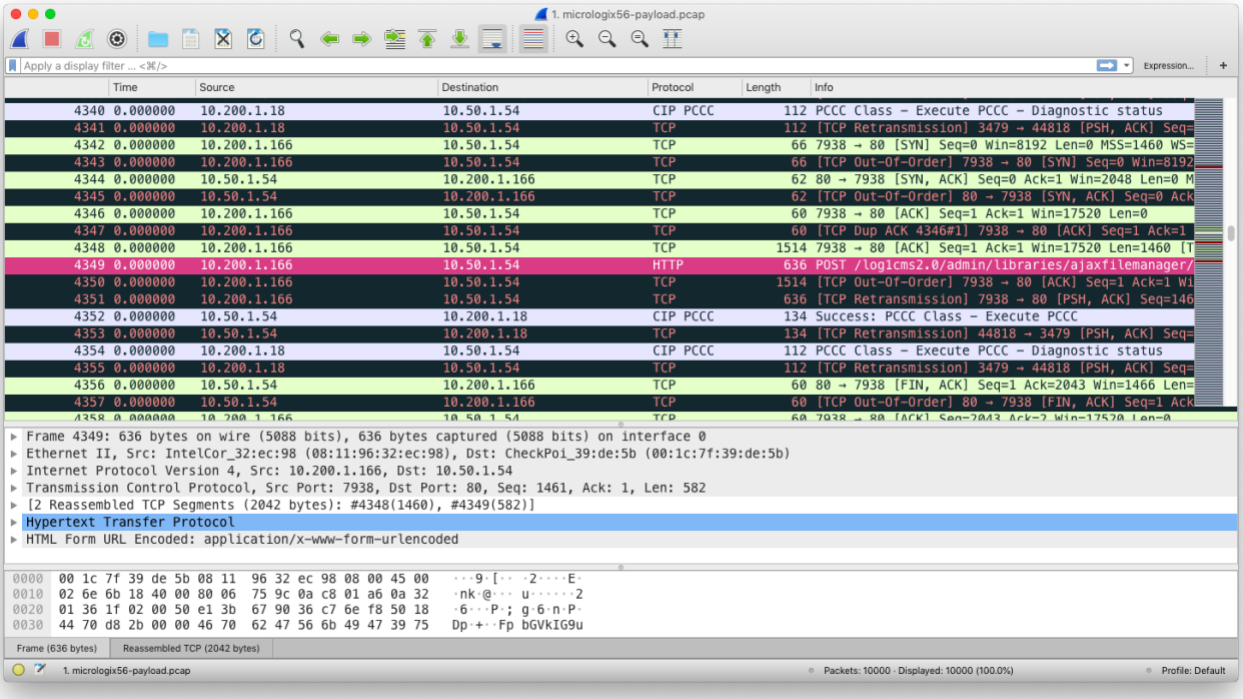
Havex is a Remote Access Trojan (RAT) that in this case is attempting to install a backdoor access by putting the php file in the server.



Level 4 – MICROLOGIX56

Port: 1255

Packet number 4349 is trying to post a payload at the destination. The backdoor is most likely the port 1255. Here, **10.50.1.54** is the target machine. Screenshot below:



Level 5 – BACNET-COVERT

I started out with just looking at the pcap with tcpdump:

tcpdump -nn -l -r 2.\ bacnet-covert.pcap | less

I then looked at the ascii content of the pcap with the command:

tcpdump -nn -l -A -r 2.\ bacnet-covert.pcap | less

Some ascii printable in the traffic to port 47808 and the word SpotUdp in the traffic to port 5762. But port number 47808 is the default port for BACnet.

On Wireshark, we can see all traffic to 47808 decoded as “Unconfirmed-REQ who-Is xx xx” request.

Next, we move to separating the last number from the traffic and convert it to both binary and hex.

00000000 86 d6 fb 87 b6 b7 91 70 d5 c7 db e3 95 fc ce 00 |.......p........|
00000010 b0 cb e4 82 e1 92 e5 73 f6 d5 bd 84 d3 9b bb 73 |.......s.......s|
00000020 90 df c0 d9 f4 ee a1 77 ab d5 85 e9 82 d8 b9 6f |.......w.......o|
00000030 e4 ff 87 ba 8a a4 f2 72 e6 aa b7 e1 a7 c7 c9 64 |.......r.......d|
00000040 a3 82 e2 ff f4 d5 a8 42 a5 b0 e7 92 d1 df 85 61 |.......B.......a|
00000050 95 fc 96 9b 9b ca d5 73 84 c4 a6 9d 9a b8 c2 69 |.......s.......i|
00000060 e0 a2 fe e3 94 96 88 73 99 e6 c6 8e d1 ff e3 6b |.......s.......k|

Message therefore is: “p.sswordBasisk”

Reverse Engineering

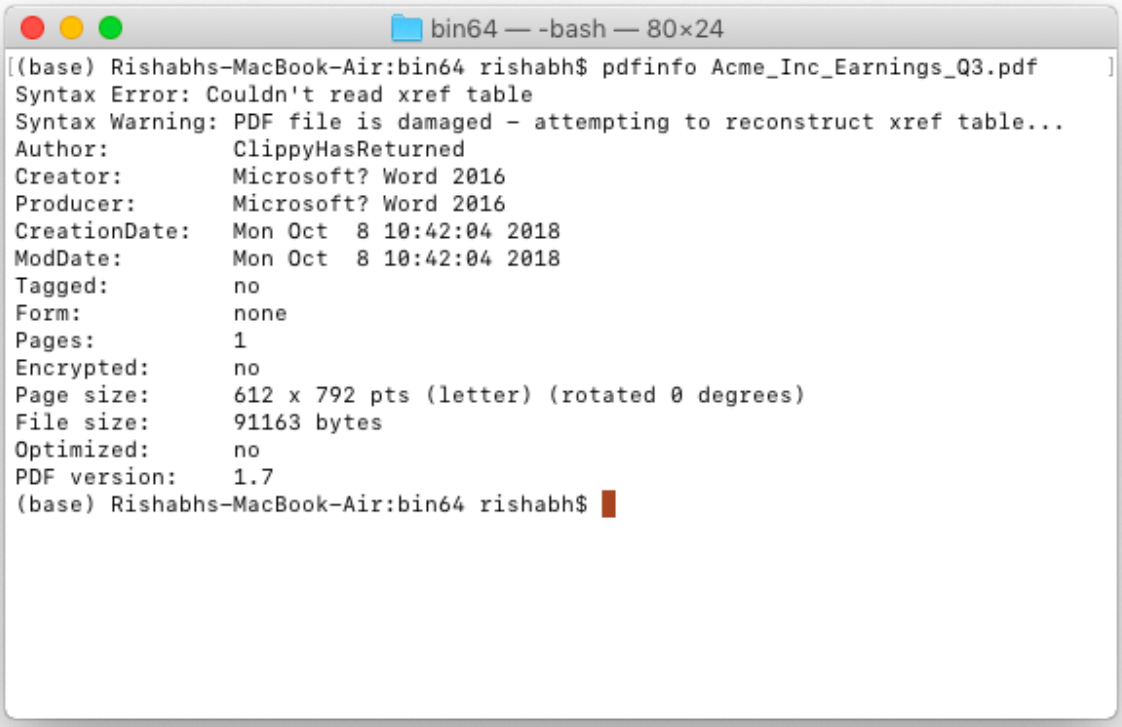
Level 1 – Script Kiddie:

Password is: “lm”, the password is a substring of characters from the password variable.

Level 2 – The PDF Your Parents Warned You About I

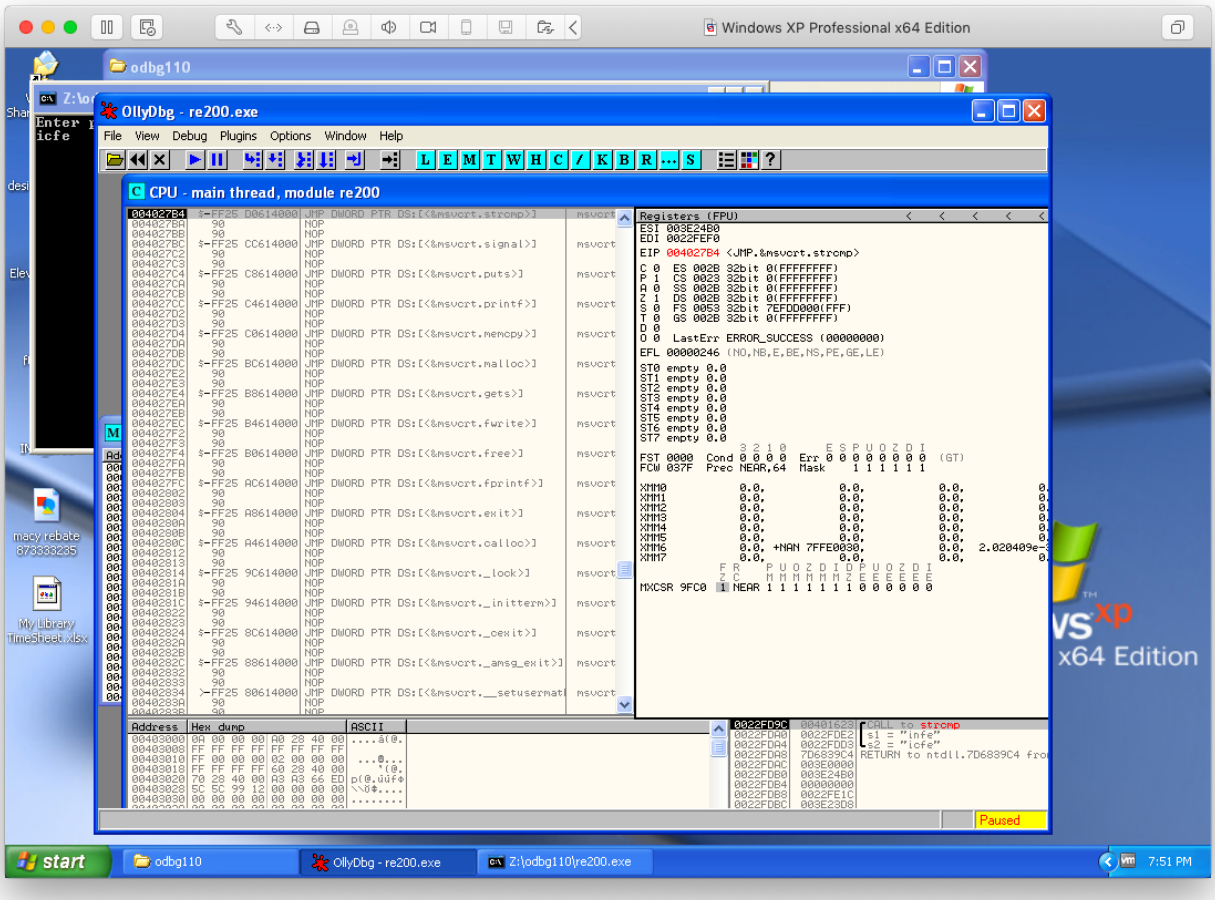
Author: ClippyHasReturned

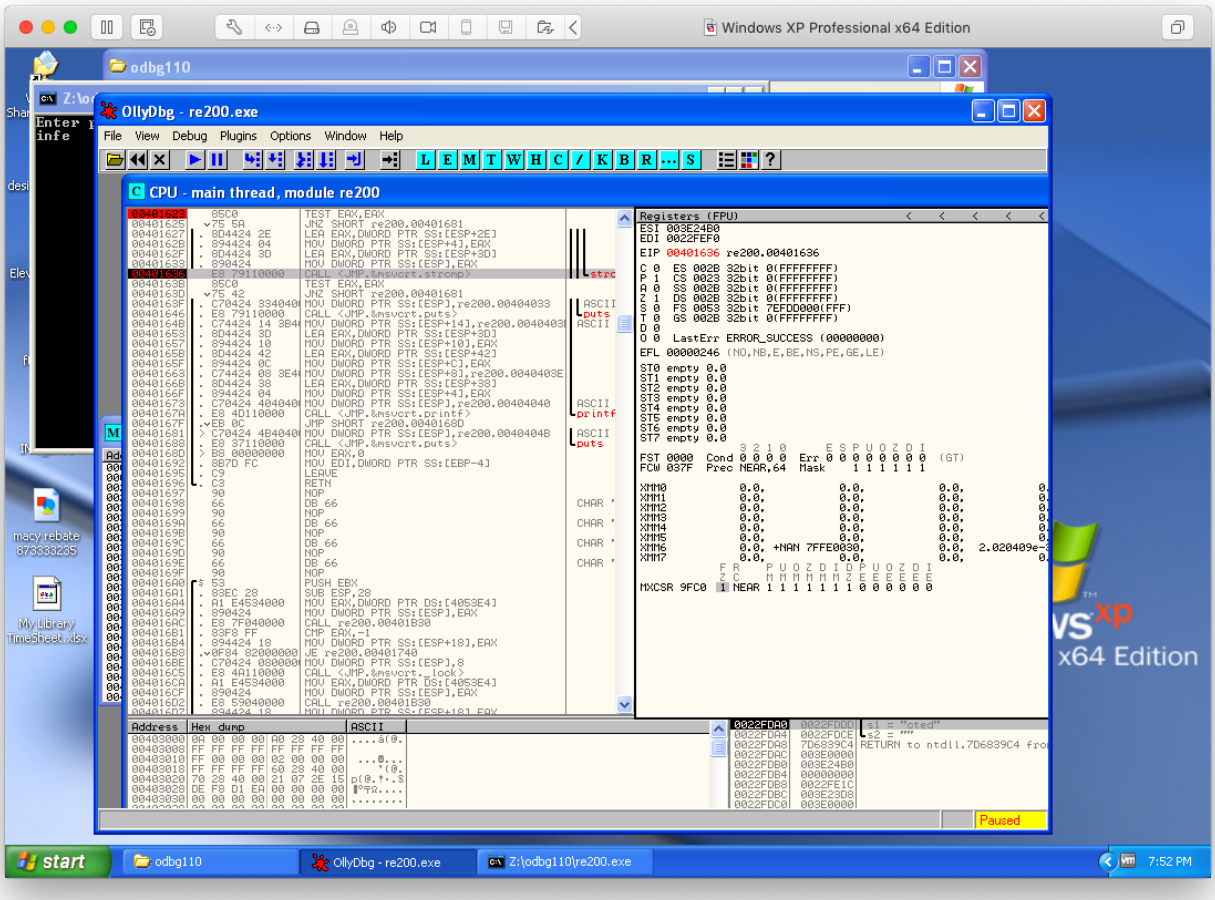
Using XpdfReader, it was an easy task to find the Author of the file. Screenshot attached.

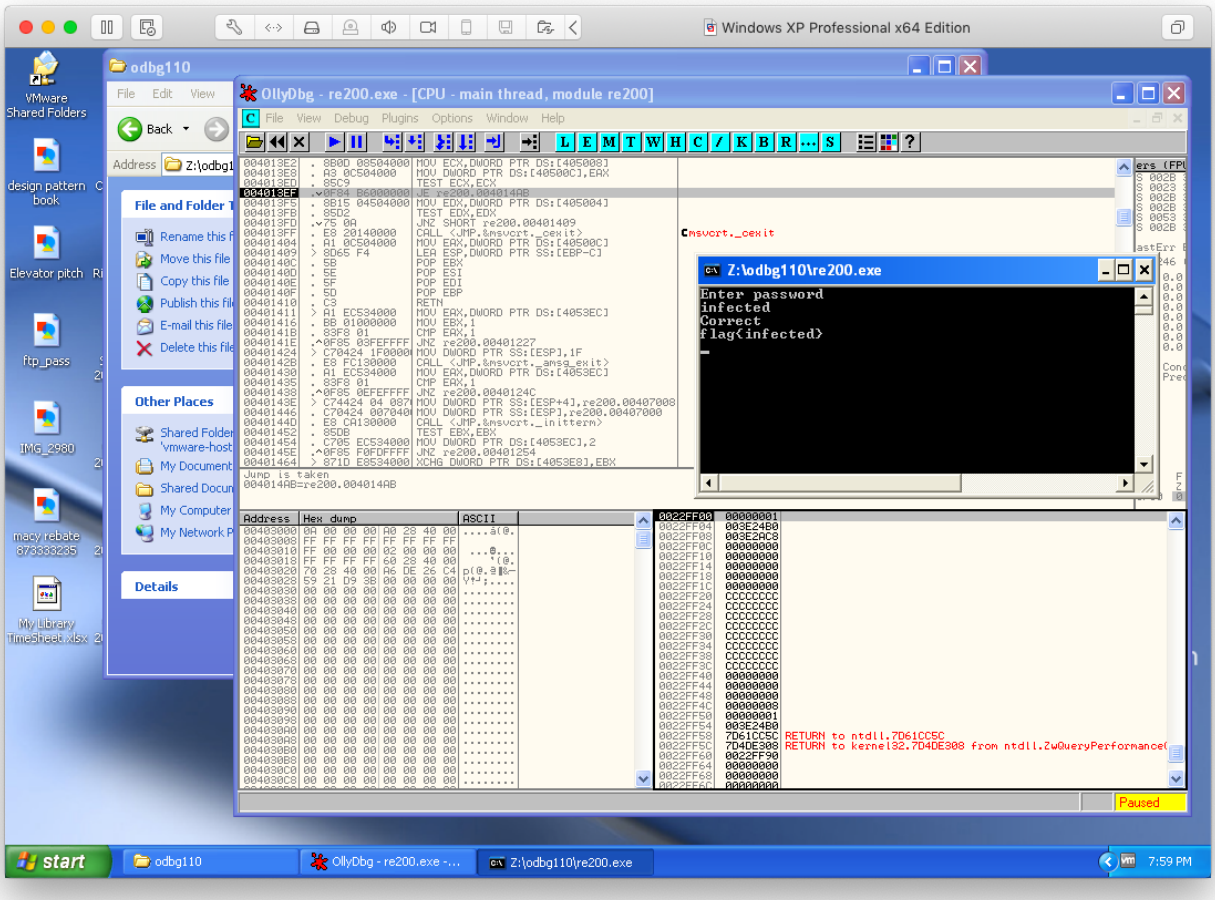


Level 3 – Split Ends

The task involved reverse engineering the executable build file. Tools used for this operation: Ollydbg, run on Windows XP as guest OS on top of MAC host OS. Breakpoints were created at the strcmp() and the Z-flag was turned to 1 which then caused the register to reveal the value of the source it is comparing to. Password was revealed in a span of 2 iterations where each iteration revealed 4 characters of the password. After a breakpoint was reached, each step was manually iterated to see the output in real-time on the DOS window. Screenshots are attached below:


Windows XP Professional x64 Edition
odbg110
OllyDbg - re200.exe
File View Debug Plugins Options Window Help
CPU - main thread, module re200
Registers (FPU)
Paused
start
odbg110
OllyDbg - re200.exe
Z:\odbg110\re200.exe
7:51 PM

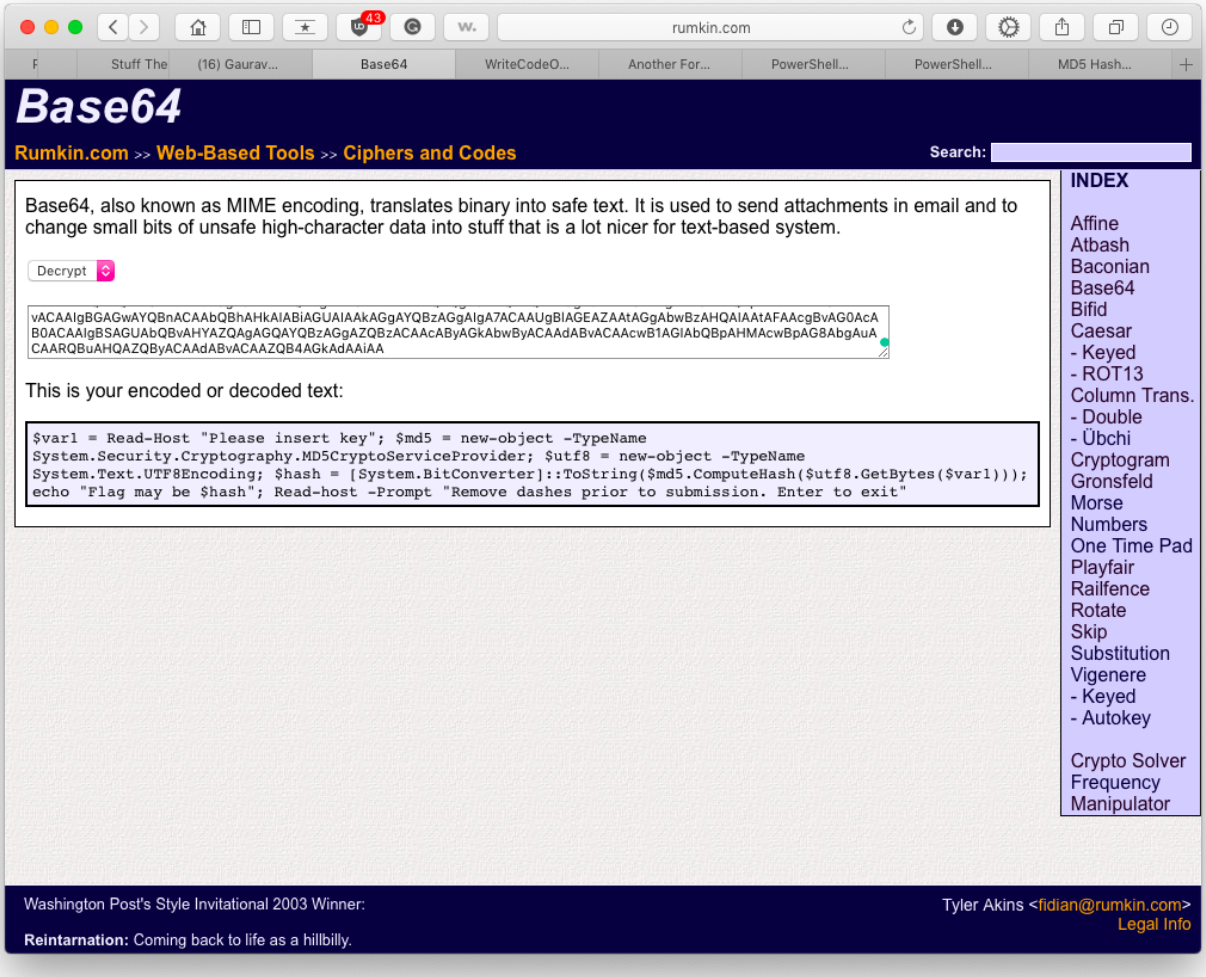

Windows XP Professional x64 Edition
odbg110
OllyDbg - re200.exe
File View Debug Plugins Options Window Help
CPU - main thread, module re200
Registers (FPU)
Paused
start
odbg110
OllyDbg - re200.exe
Z:\odbg110\re200.exe
7:52 PM

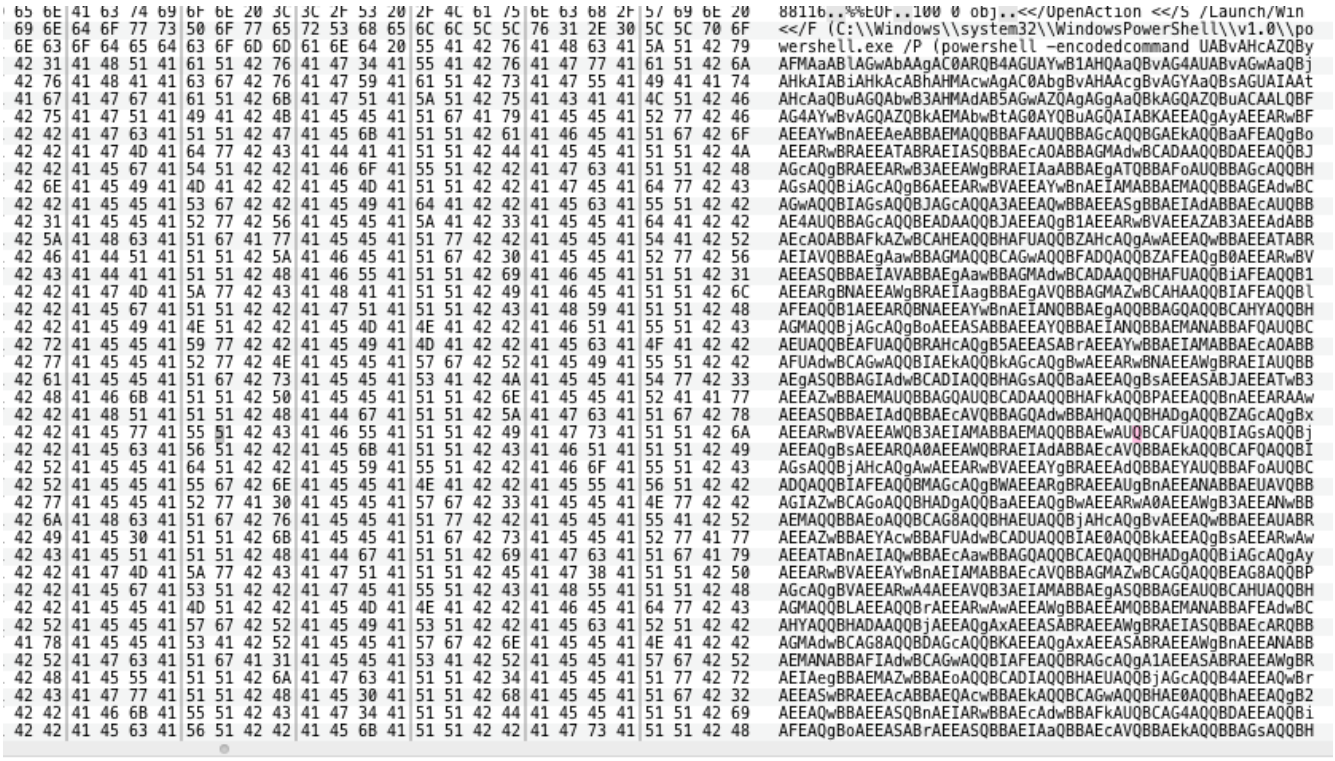


Level 4 – The PDF Your Parents Warned You About II

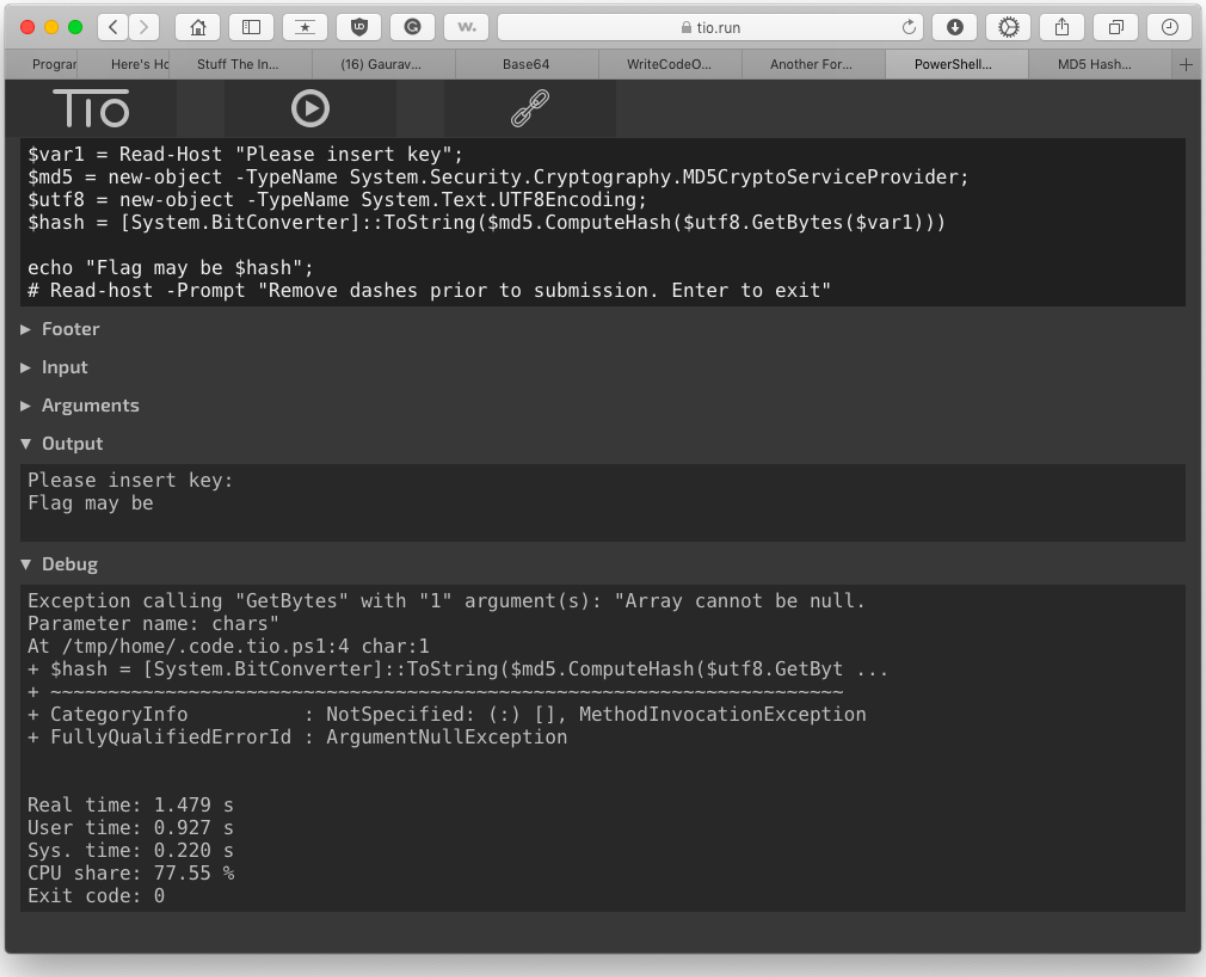
The pdf file had an base64 encoded powershell script. The script basically computes the MD5 hash for the string: "*Please insert key*"
Hash: **c97ff22e75b6cf465b70ae0dddc8e773**

Script and the screenshot are attached below:


rumkin.com
Stuff The
(16) Gaurav...
Base64
WriteCodeO...
Another For...
PowerShell...
PowerShell...
MD5 Hash...
Base64
Rumkin.com >> Web-Based Tools >> Ciphers and Codes
Search:
Base64, also known as MIME encoding, translates binary into safe text. It is used to send attachments in email and to change small bits of unsafe high-character data into stuff that is a lot nicer for text-based system.
Decrypt
vACAAIgBGAGwAYQBnACAAbQBhAHkAIABiAGUAIAAkAGgAYQBzAGgAIgA7ACAAUgBlAGEAZAAtAGgAbwBzAHQAIAAtAFAAcgBvAG0AcA
B0ACAAIgBSAGUAbQBvAHYAZQAgAGQAYQBzAGgAZQBzACAAcAByAGkAbwByACAAdABvACAAcwB1AGIAbQBpAHMAcwBpAG8AbgAuA
CAARQBuAHQAZQByACAAdABvACAAZQB4AGkAdAAiAA
This is your encoded or decoded text:
$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"
INDEX
Affine
Atbash
Baconian
Base64
Bifid
Caesar
- Keyed
- ROT13
Column Trans.
- Double
- Übchi
Cryptogram
Gronsfeld
Morse
Numbers
One Time Pad
Playfair
Railfence
Rotate
Skip
Substitution
Vigenere
- Keyed
- Autokey
Crypto Solver
Frequency
Manipulator
Washington Post's Style Invitational 2003 Winner:
Reintarnation: Coming back to life as a hillbilly.
Tyler Akins <fidian@rumkin.com>
Legal Info

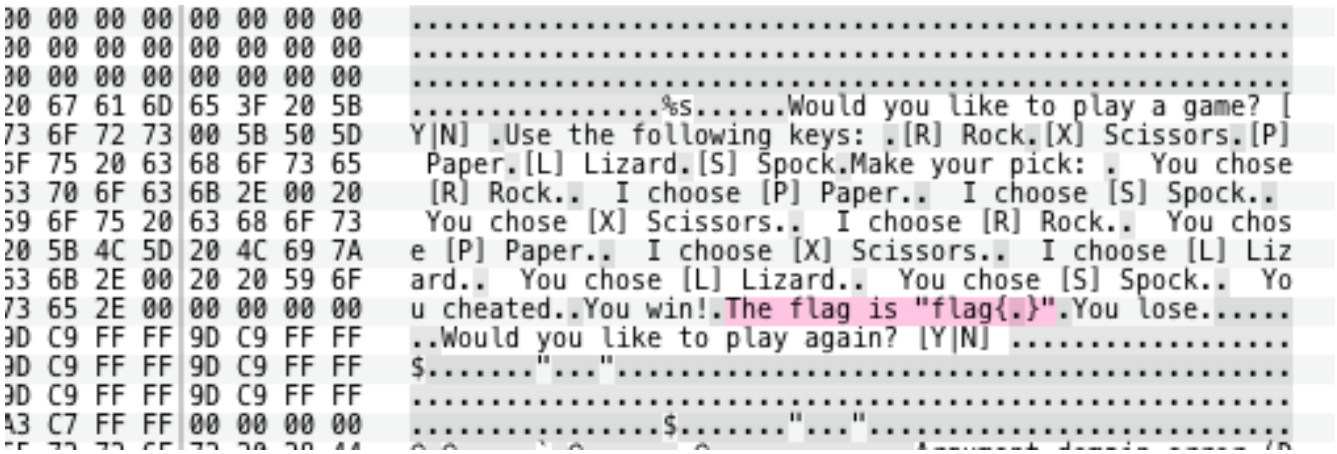


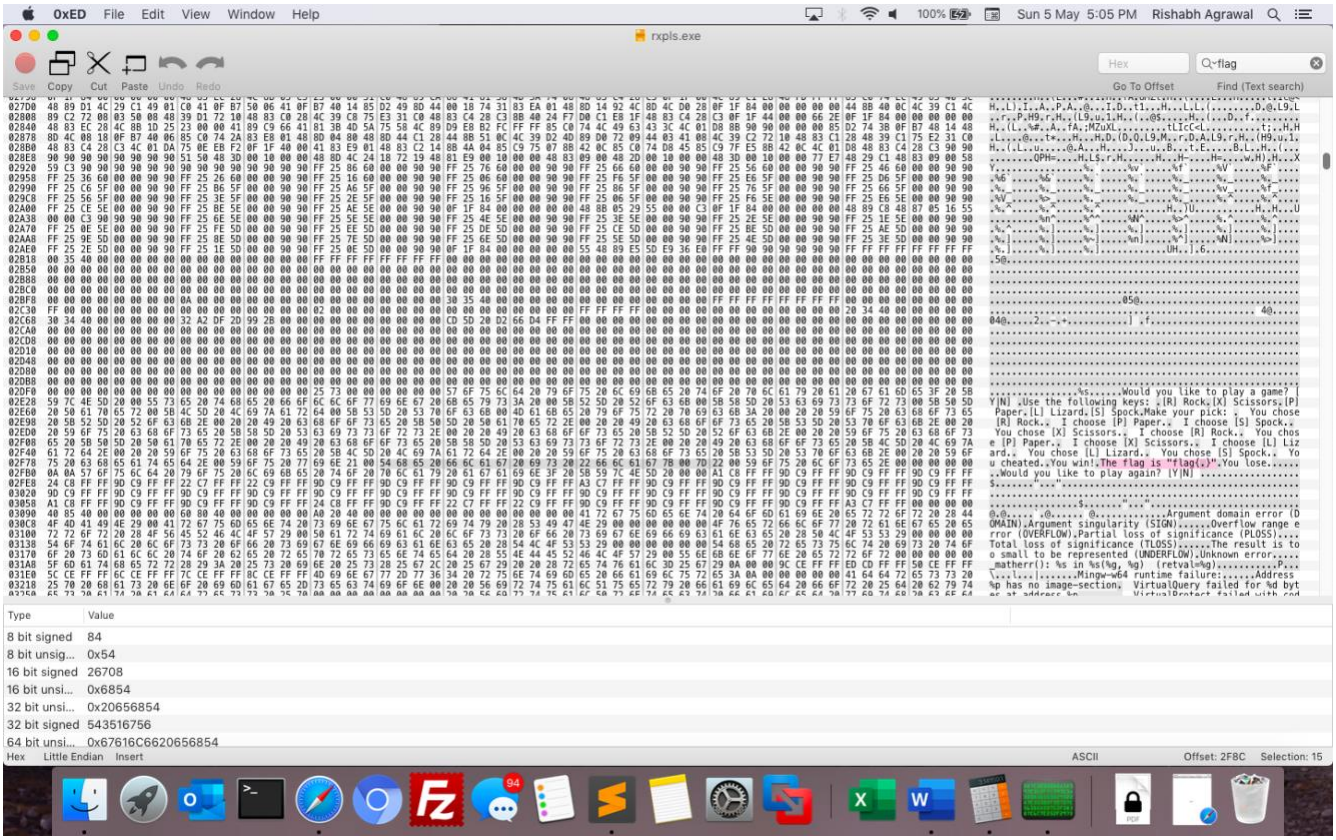
Manually computed the MD5 hash for the string. The console for the script gave me some error but it was easy to understand what the program is doing. Screenshot for that is also attached below:



Level 5 – Rock Scissors Paper Lizard Spock

Using OxED for MacOS, the .exe file was opened and analyzed. The flag found is: “flag{.}”. Screenshot below:

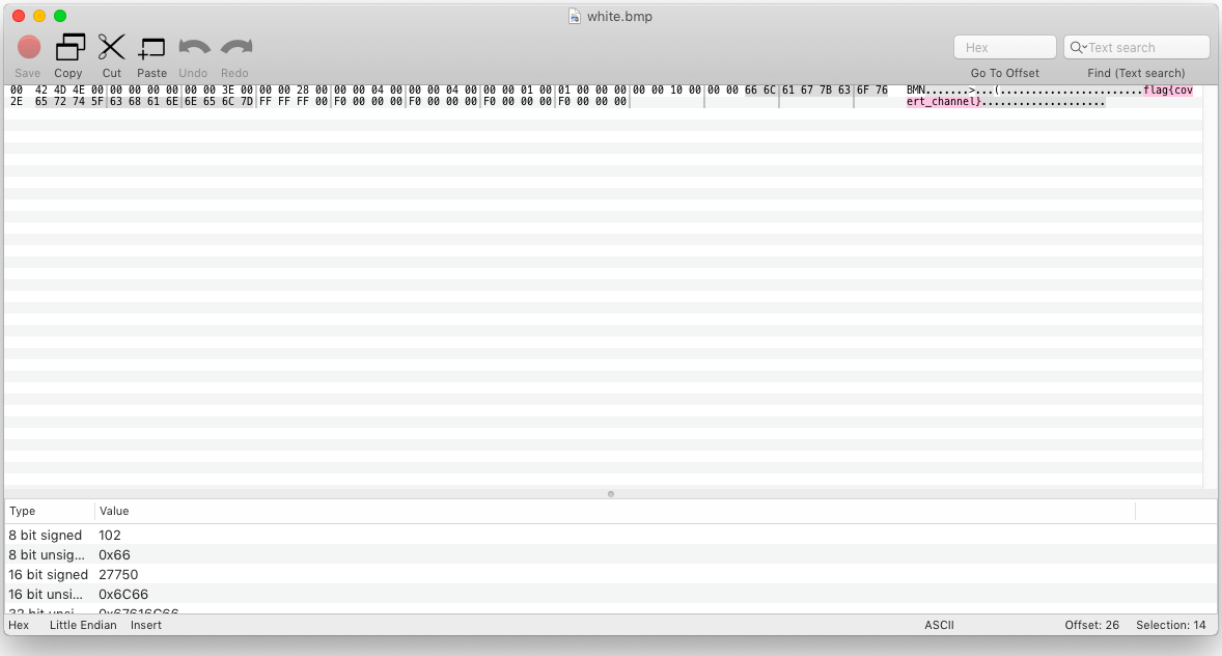




Steganography

Level 1 – Moo Cow

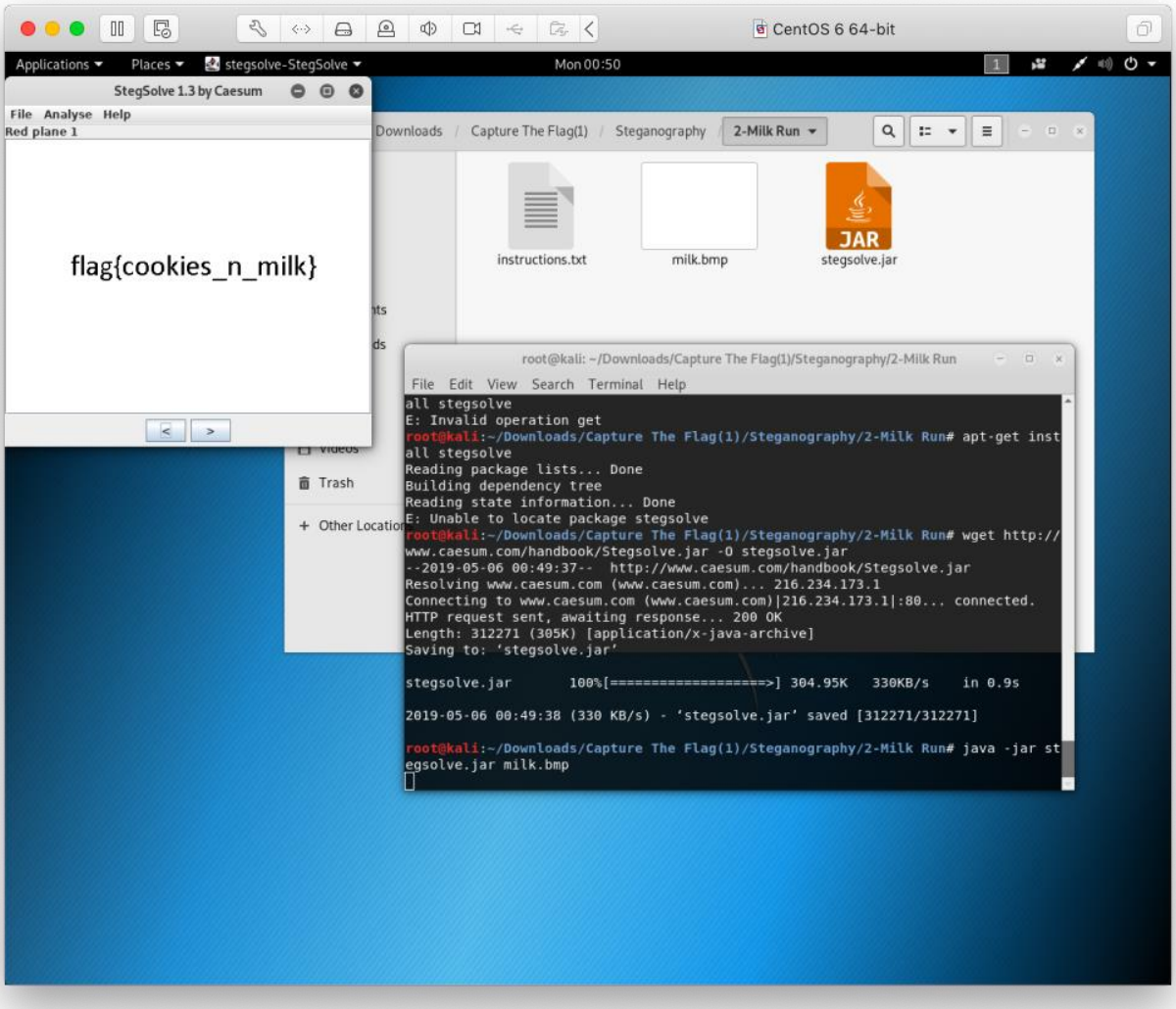
flag{covert_channel}
Screenshot attached below:



Level 2 – Milk Run

Flag{cookies_n_milk}

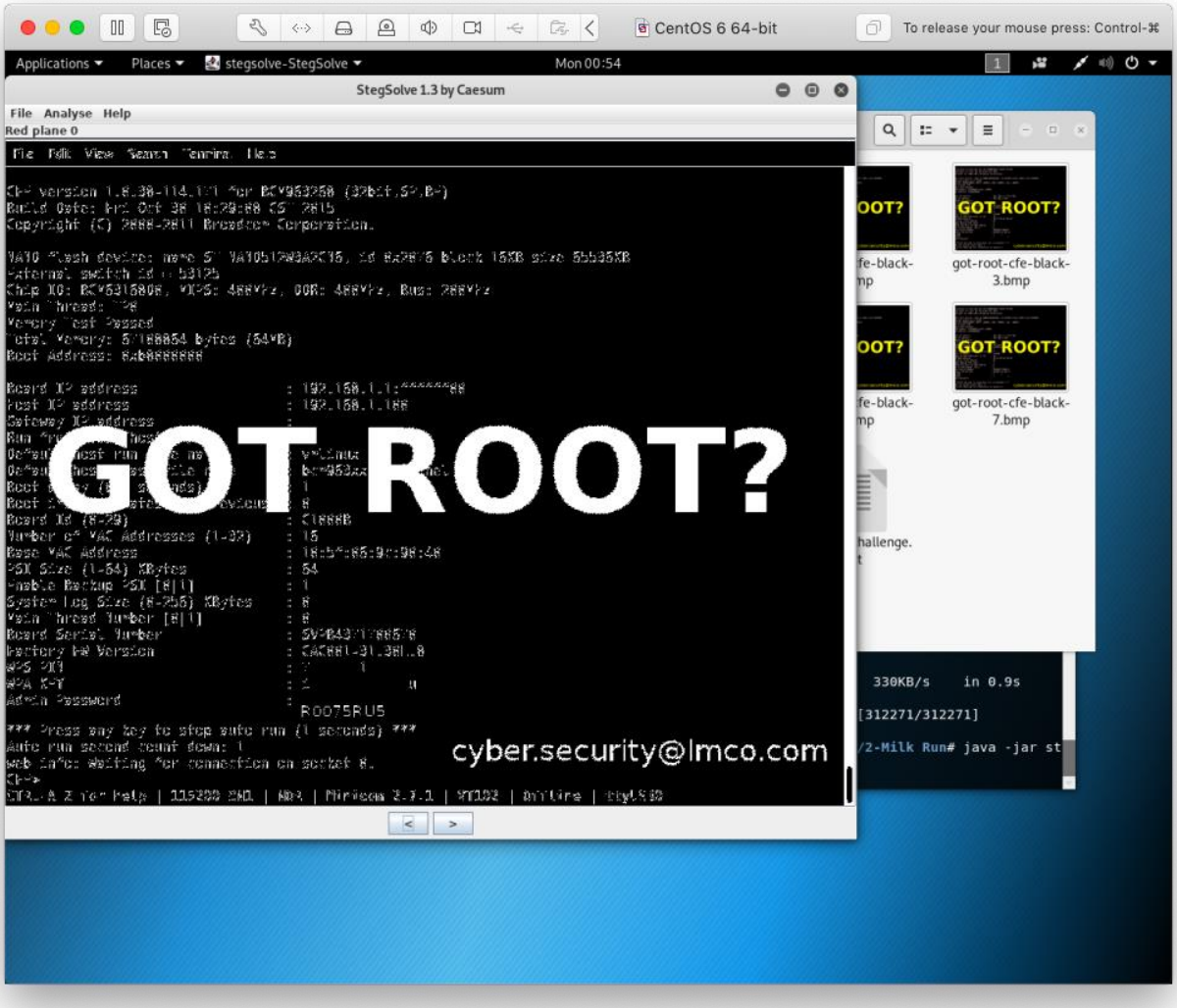
Using StegSolve in Kali Linux, we were able to change color channel and expose the hidden flag for the particular image.

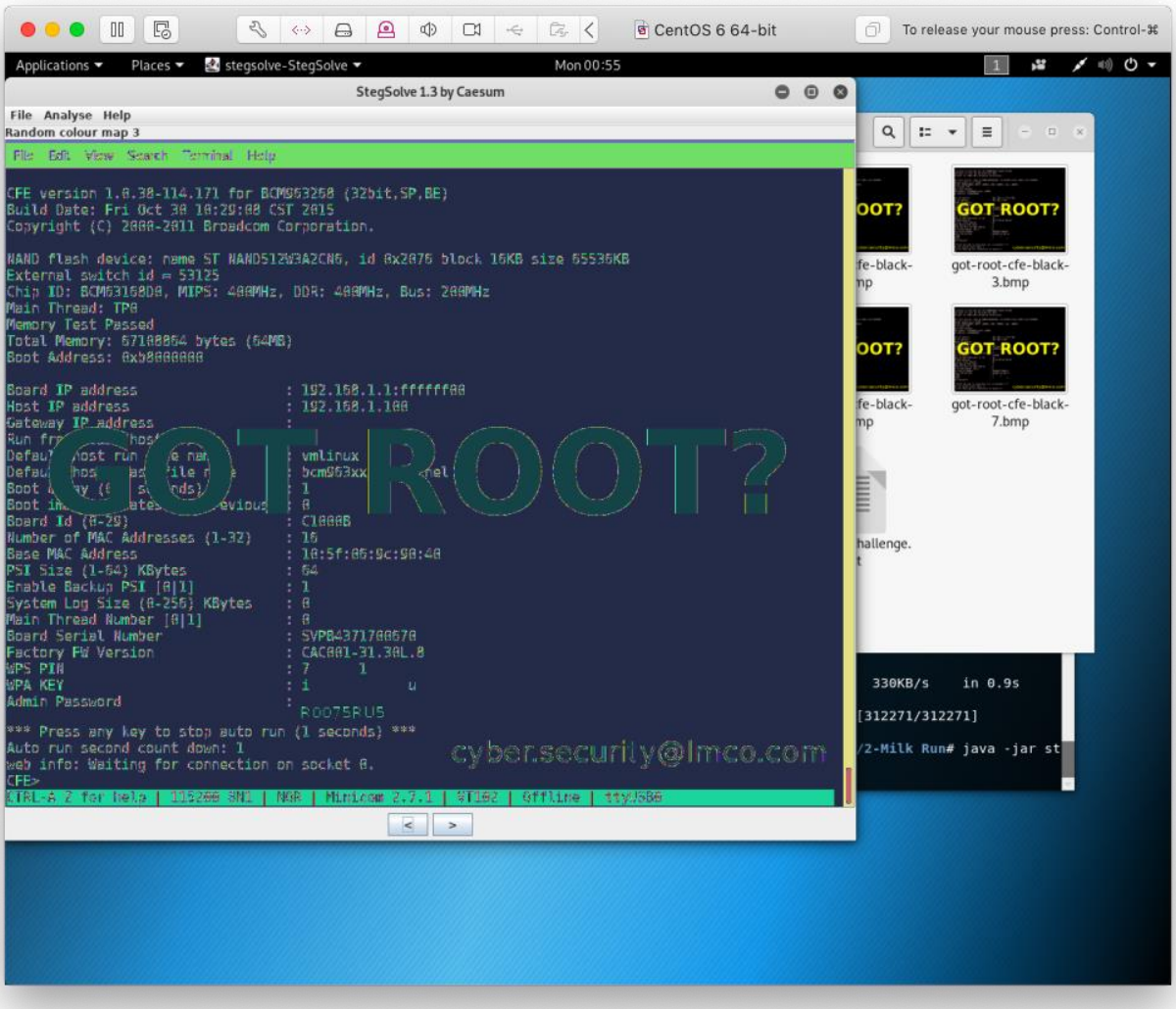


Level 3 – Got Root

Using StegSolve in Kali Linux, we were able to change color channel and expose the hidden flag for the particular image. The flag was hidden in the 6th file, and the password is: R0075RU5

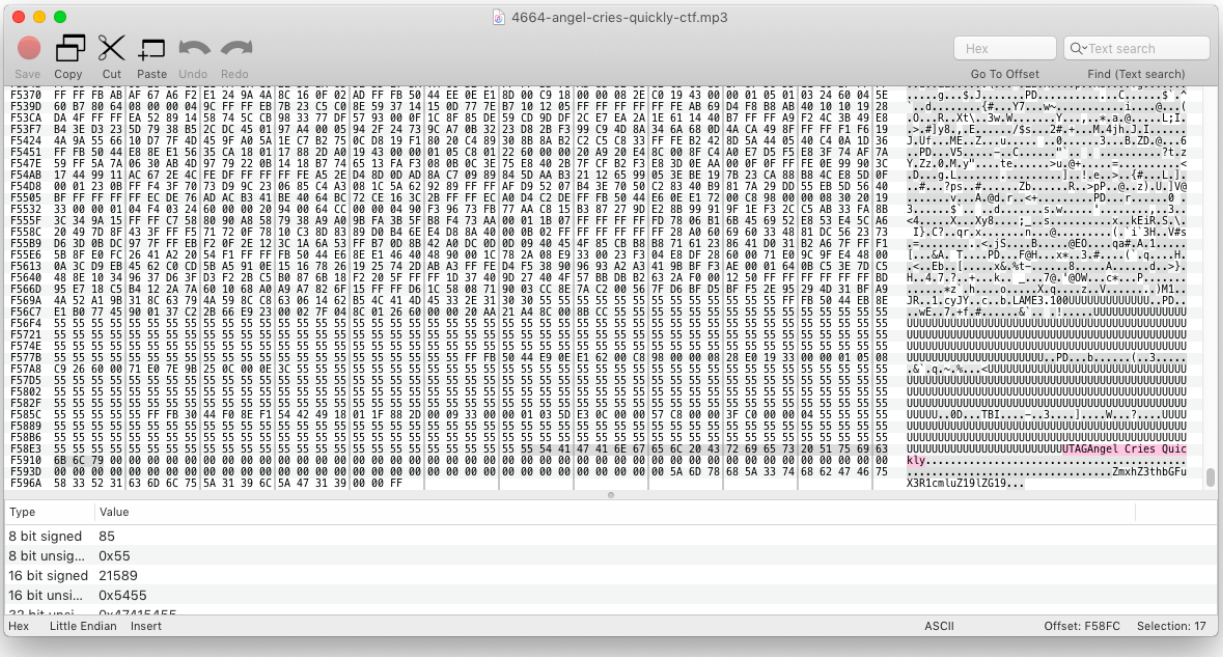
See the screenshot below:


CentOS 6 64-bit
To release your mouse press: Control-⌘
Applications
Places
stegsolve-StegSolve
Mon 00:54
StegSolve 1.3 by Caesum
File Analyse Help
Red plane 0
GOT ROOT?
cyber.security@lmco.com
got-root-cfe-black-3.bmp
got-root-cfe-black-7.bmp
330KB/s in 0.9s
[312271/312271]



Level 4 – AI EDM

TAG Angel Cries Quickly
Opening with an hex editor, it uncovered the hidden flag at the end of the file
Program: OxED for Mac OS
Screenshot:



Level 5 – Final Frontier

Comment: nope

I uploaded the file on virustotal.com and then on the description tab on the page I scanned through the Exif data of the video file.

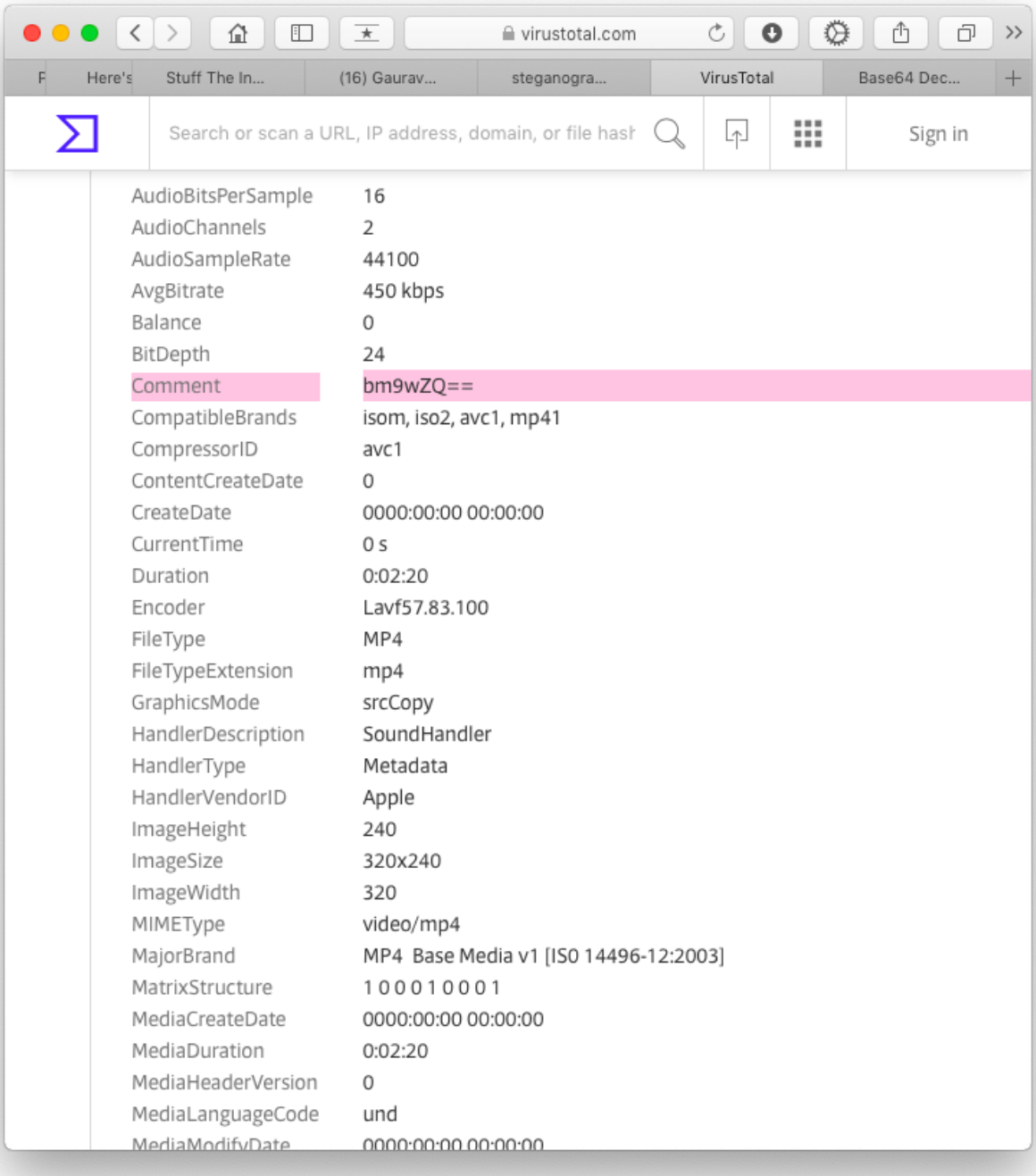
The Exif data of the file was extracted and we found base64 encoded string in one of the file attributes.

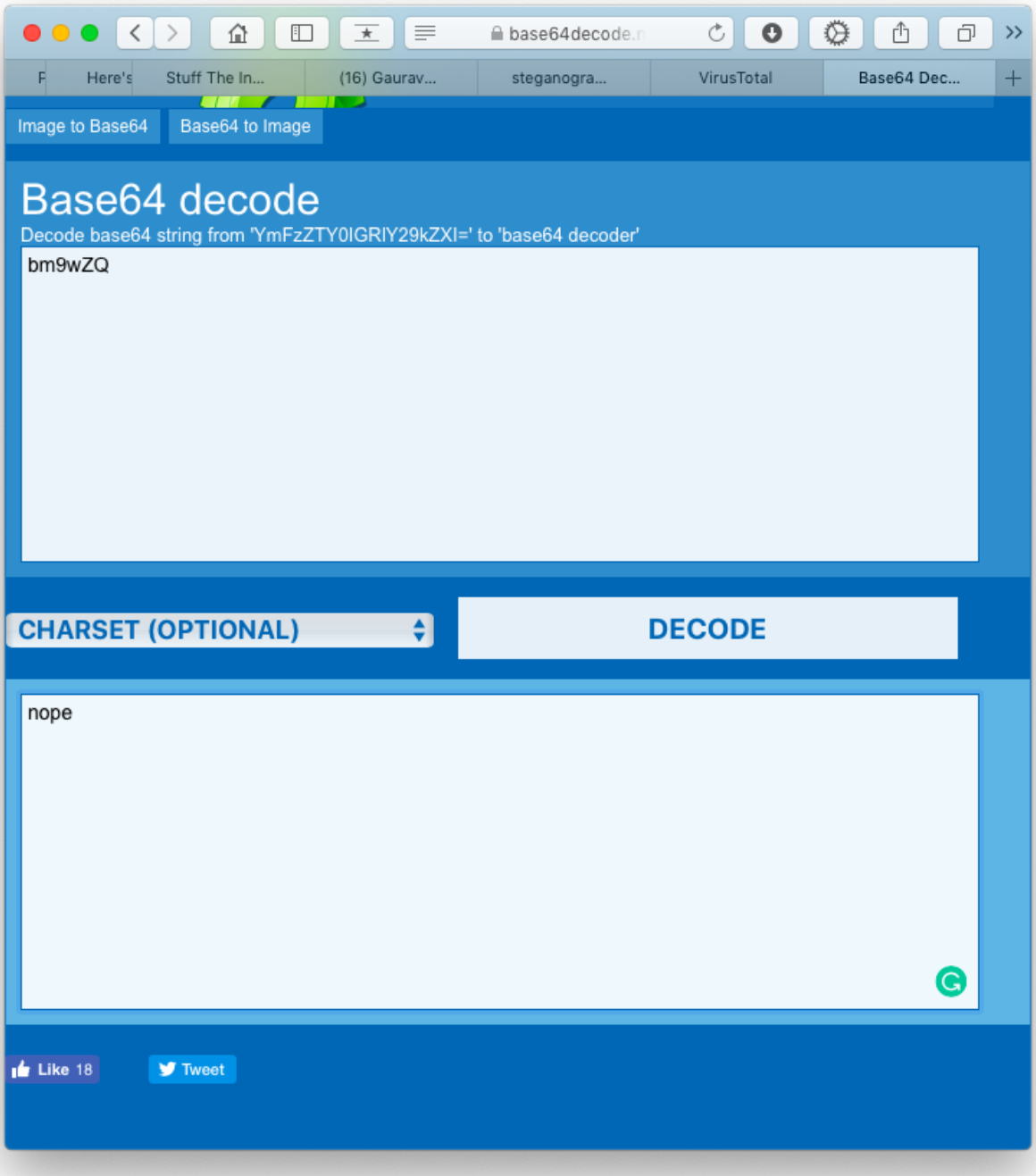
The video file has the comment parameter: "bm9wZQ==" which is a base64 encoded string text. The video file was scanned for any strings present in it but didn't yield much result except for some sting tokens. Kali Linux has forensic tools for digital forensics but not the best solution when it's a video file.

Any random characters that ends with == means it's a base64 encoded string and '=' characters are padded to the string to match its length.

Thus on decoding the base64 string, it is: "nope"

Screenshot attached.


virustotal.com
Here's
Stuff The In...
(16) Gaurav...
steganogra...
VirusTotal
Base64 Dec...
Search or scan a URL, IP address, domain, or file hash
Sign in
AudioBitsPerSample 16
AudioChannels 2
AudioSampleRate 44100
AvgBitrate 450 kbps
Balance 0
BitDepth 24
Comment bm9wZQ==
CompatibleBrands isom, iso2, avc1, mp41
CompressorID avc1
ContentCreateDate 0
CreateDate 0000:00:00 00:00:00
CurrentTime 0 s
Duration 0:02:20
Encoder Lavf57.83.100
FileType MP4
FileTypeExtension mp4
GraphicsMode srcCopy
HandlerDescription SoundHandler
HandlerType Metadata
HandlerVendorID Apple
ImageHeight 240
ImageSize 320x240
ImageWidth 320
MIMEType video/mp4
MajorBrand MP4 Base Media v1 [IS0 14496-12:2003]
MatrixStructure 1 0 0 0 1 0 0 0 1
MediaCreateDate 0000:00:00 00:00:00
MediaDuration 0:02:20
MediaHeaderVersion 0
MediaLanguageCode und
MediaModifyDate 0000:00:00 00:00:00



Well, then it looks like Lockheed Martin isn't sending anyone back to the moon.