

WolvCTF 2024

[Eternally Pwned: Infiltration \[forensics\]](#)

[Eternally Pwned: Persistence \[forensics\]](#)

[WOLPHV I: Reconnaissance \[osint\]](#)

[WOLPHV II: Infiltrate \[osint\]](#)

[WOLPHV III: p1nesh4dow48 \[osint\]](#)

[Limited 1 \[crypto\]](#)

[WOLPHV IV: d4wgbyte262 \[osint\]](#)

[Log Analysis \[forensics\]](#)

Eternally Pwned: Infiltration [forensics]

I recently had my passwords and other sensitive data leaked, but I have no idea how. Can you figure out how the attacker got in to my PC?

Flag: wctf{I3tS_3teRn4lLy_g0_bLU3_7n9wm4iWnL}

The challenge provides a pcap file that contains the network connection for the victim. From the file, we can easily identify that the victim's IP is 192.168.3.140. While scrolling down, one section of the connection interests me.

Time	Source IP	Destination IP	Protocol	Sequence Number	Acknowledgment Number	Window Size	Length
1157 187.999862	192.168.3.140	192.168.3.133	TCP	54	[TCP Window Update] 49171 → 4444 [ACK] Seq=9598 Ack=510822 Len=0		
1169 187.999862	192.168.3.140	192.168.3.133	TCP	54	49171 → 4444 [ACK] Seq=9598 Ack=526882 Win=8613888 Len=0		
1199 187.999862	192.168.3.140	192.168.3.133	TCP	54	49171 → 4444 [ACK] Seq=9598 Ack=569222 Win=8571648 Len=0		
1222 187.999862	192.168.3.140	192.168.3.133	TCP	54	49171 → 4444 [ACK] Seq=9598 Ack=601342 Win=8539392 Len=0		
1287 187.999864	192.168.3.140	192.168.3.133	TCP	54	49171 → 4444 [ACK] Seq=9598 Ack=648062 Win=8492800 Len=0		
1288 187.999864	192.168.3.140	192.168.3.133	TCP	54	49171 → 4444 [ACK] Seq=9598 Ack=649782 Win=8445952 Len=0		
1327 187.999864	192.168.3.140	192.168.3.133	TCP	54	49171 → 4444 [ACK] Seq=9598 Ack=741502 Win=8399360 Len=0		
1328 187.999864	192.168.3.140	192.168.3.133	TCP	54	49171 → 4444 [ACK] Seq=9598 Ack=750262 Win=8390400 Len=0		
1362 187.999864	192.168.3.140	192.168.3.133	TCP	54	49171 → 4444 [ACK] Seq=9598 Ack=796982 Win=8343808 Len=0		
1427 187.999865	192.168.3.140	192.168.3.133	TCP	54	49171 → 4444 [ACK] Seq=9598 Ack=845162 Win=8295680 Len=0		
1428 187.999865	192.168.3.140	192.168.3.133	TCP	54	49171 → 4444 [ACK] Seq=9598 Ack=891882 Win=8248832 Len=0		
1487 187.999865	192.168.3.140	192.168.3.133	TCP	54	49171 → 4444 [ACK] Seq=9598 Ack=938602 Win=8202240 Len=0		
1488 187.999865	192.168.3.140	192.168.3.133	TCP	66	49171 → 4444 [ACK] Seq=9598 Ack=975102 Win=8165632 Len=0 SLI		
1495 187.999865	192.168.3.140	192.168.3.133	TCP	66	49171 → 4444 [ACK] Seq=9598 Ack=976562 Win=8164096 Len=0 SLI		
1496 187.999865	192.168.3.140	192.168.3.133	TCP	66	49171 → 4444 [ACK] Seq=9598 Ack=978022 Win=8162816 Len=0 SLI		

By looking at the uncommon port (or commonly used as reverse shell port), this tells us that the attacker has IP of 192.168.3.133. This challenge is about infiltration, how the sensitive data is leaked to the attacker. So, I applied the filter `ip.src == 192.168.3.140 && ip.dst == 192.168.3.133` to look for any infiltration tracks. While following one of the huge chunks of TCP, I found a smb echo request with a base64 data `d2N0ZntsM3RTXw==` which then decoded as the first part of flag:

wctf{13tS_ .

Majority of the protocols are either TCP or SMB related, so I decided to dive back to SMB stuff first by applying filter `smb`. After the echo request, I saw another huge chunk of Trans2 Secondary Request Data that has two different base64 encoded data hidden within it.

Encoded data: M3RlUm40bEx5X2cwXw== YkxVM183bj13bTRpV25MfQ==

Decoded data: 3teRn4lLy_g0_ bLU3_7n9wm4iWnL}

Eternally Pwned: Persistence [forensics]

I get that the attackers were in my PC, but how did they achieve persistence?

Flag: wctf{v0lAt1l3_m3m0ry_4qu1r3D_a3fe9fn3al}

Before we start doing memory analysis, we need to look for the profile used. Using vol3 (rather than waiting for vol2 to finish imageinfo), we can get the following info which deduced that the profile is `Win7SP1x64_23418`.

```
Is64Bit True
IsPAE False
layer_name      0 WindowsIntel32e
memory_layer    1 WindowsCrashDump64Layer
base_layer      2 FileLayer
KdDebuggerDataBlock 0xf80001a430a0
NTBuildLab     7601.17514.amd64fre.win7sp1_rtm.
CSDVersion     1
KdVersionBlock 0xf80001a43068
Major/Minor     15.7601
MachineType    34404
KeNumberProcessors 1
SystemTime      2024-03-09 12:05:40
NtSystemRoot    C:\Windows
NtProductType   NtProductServer
NtMajorVersion  6
NtMinorVersion  1
PE MajorOperatingSystemVersion 6
PE MinorOperatingSystemVersion 1
PE Machine      34404
PE TimeDateStamp Sat Nov 20 09:30:02 2010
```

Talking about persistence, we need to check the processes running to identify if there's any suspicious ones.

```
0x111111a8018e3e620 msatcc.exe
0xfffffa801a496450 cmd.exe
0xfffffa801a4a8630 conhost.exe
0xfffffa801a4ba060 notepad.exe
0xfffffa801a8de800 cGFzdGViaW4uY2
0xfffffa801a8601d0 multireader.ex
0xfffffa801a983b30 iexplore.exe
0xfffffa801a983b30 iexplore.exe
```

From here, we can see this random name process weird. Running `cmdline`, we can look at the full filename.

```
*****
notepad.exe pid: 1644
Command line : "C:\Windows\system32\NOTEPAD.EXE" C:\Users\joe\Desktop\schedule
*****
cGFzdGViaW4uY2 pid: 1804
Command line : "C:\temp\cGFzdGViaW4uY29tL3lBYTFhs2l1.exe"
*****
multireader.ex pid: 896
Command line : "C:\temp\multireader.exe"
```

Noticing the filename is actually encoded in base64 [BIG THANKS to warlocksmurf], we can decode it and eventually it will give us a pastebin link. The flag lies in the pastebin.

`cGFzdGViaW4uY29tL3lBYTFhs2l1` → pastebin.com/yAa1aKiu

WOLPHV I: Reconnaissance [osint]

A new ransomware group you may have heard about has emerged: **WOLPHV**

There's already been reports of their presence in articles and posts.

NOTE: Wolphv's twitter/X account and <https://wolphv.chal.wolvsec.org/> are **out of scope** for all these challenges. Any flags found from these are not a part of these challenges

This is a start to a 5 part series of challenges. Solving this challenge will unlock **WOLPHV II: Infiltrate**

Flag: wctf{0k_1_d0nT_th1Nk_A1_w1ll_r3Pl4c3_Us_f0R_4_l0ng_t1me}

Talking about a new ransomware group called **WOLPHV**. With some googling, there are only a few relevant results:

- <https://thecyberexpress.com/new-wolphv-ransomware-group-on-the-dark-web/>
- https://www.linkedin.com/posts/coy-peterman-35474_new-wolphv-ransomware-group-on-the-dark-web-activity-7113152189342081025-GtTT
- <https://twitter.com/FalconFeedsio/status/1706989111414849989>

When looking at the tweet, there was one comment that was different. After base64 decode, we can retrieve the flag.



The screenshot shows a web-based tool for decoding Base64 strings. On the left, under 'Recipe', it says 'From Base64' with an 'Alphabet' dropdown set to 'A-Za-z0-9+/=' and a checked checkbox for 'Remove non-alphabet chars'. There's also an unchecked checkbox for 'Strict mode'. On the right, under 'Input', is a long string of characters: d2N0Znswa18xX2Qwb1RfdGgxTmtfQTFFdzFsbF9yM1BsNGMzX1VzX2YwU180X2wwbmdfdDFtZX0=. Under 'Output', the result is displayed as: wctf{0k_1_d0nT_th1nk_A1_w1ll_r3pl4c3_us_f0r_4_l0ng_t1me}

WOLPHV II: Infiltrate [osint]

Since the WOLPHV twitter/x is out of comission now, I wonder where else the official WOLPHV group posts on social media. Maybe we can also infiltrate what they use to message each other

NOTE: Wolphv's twitter/X account and <https://wolphv.chal.wolvsec.org/> are **out of scope** for all these challenges. Any flags found from these are not a part of these challenges

Solving this challenge will unlock **WOLPHV III**, **WOLPHV IV**, and **WOLPHV V**

Flag: wctf{r0t_52_w0uID_b3_cr4zy_fRfr}

From the scenario, we knew that the wolphv twitter/x was out of commission and they were using other social media. From the previous post by cyberexpress, they mentioned something about WOLPHV using a dark web portal. So, I used duckduckgo as my search engine and looked for any potential portal used.

wolphv portal

All Images Videos News Maps Settings

Volusia. Employees. Intranet Login.

X https://twitter.com › hashtag › Wolphv
#Wolphv | Twitter

https://www.volusiaonlinelearning.com › atlantic-portals
Portals - Volusia Online Learning
Call our ITS Help Desk. 386-734-7190 Ext 25000. Open 7:30am - 5:00pm, Monday thru Friday. Summer Hours: Open 7:00am - 6:00pm. Monday thru Thursday.

f https://www.facebook.com › groups › 921721029413388
WOLPHV OFFICIAL - Facebook
John Smith created the group WOLPHV OFFICIAL. 1d -. Recent posts directory.

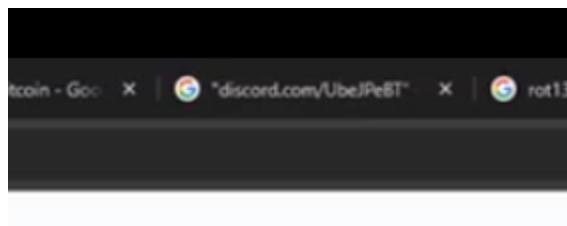
When searching, I found this facebook public group which could be a potential place they used to message each other -

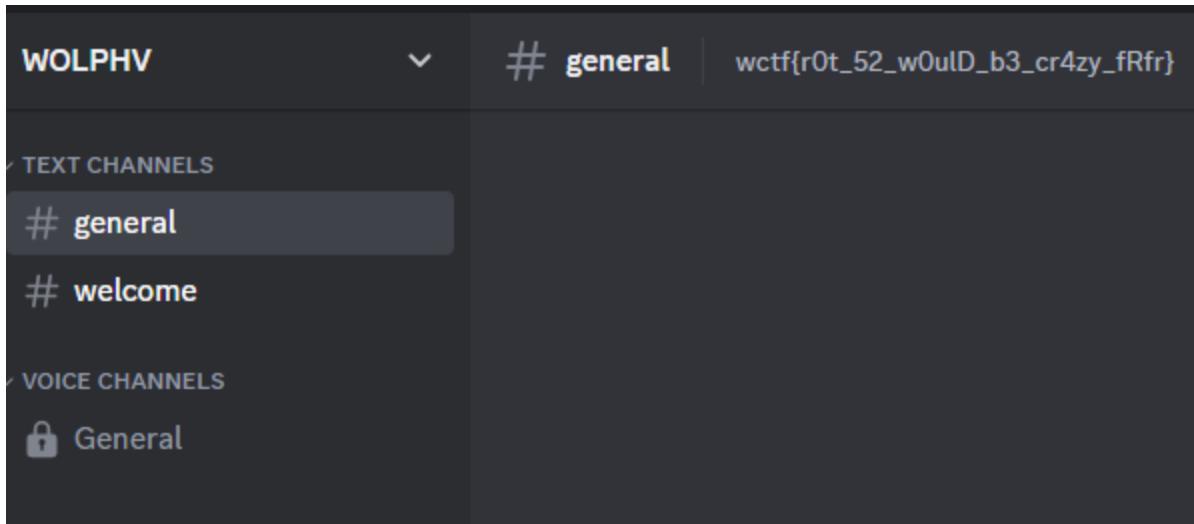
<https://www.facebook.com/groups/921721029413388/>

Looking at the group, there was nothing interesting except for one video.



From the first sight, I saw nothing. However, after taking a close look, I realised that one of the tabs was a discord invite link: discord.com/invite/UbeJPeBT.





From the discord channel, we can see the flag lying around.

WOLPHV III: p1nesh4dow48 [osint]

Locate p1nesh4dow48's home coordinates and submit them as
`wctf{x.xxx,x.xxx}` rouded

For example, if the location was the BBB at the Unviersity of Michigan the flag would be `wctf{42.293, -83.717}`

You have 10 tries to get the location. If you are *sure* you have location but the flag does not work, make a ticket

NOTE: Wolphv's twitter/X account and <https://wolphv.chal.wolvsec.org/> are out of scope for all these challenges. Any flags found from these are not a part of these challenges

Flag: `wctf{46.545,-87.388}`

At first, I tried to use some of the information i knew. From the chat, p1nesh4dow48 was the admin of WOLPHV, meaning he is a uofm (University of Michigan) student. Considering this, I looked for places near the university by checking places nearby pizza subway and mcdonalds which I then realized I'm on the wrong path (cause that's was someone's place, not what we're looking for).

==== [RESET MY BRAIN] ====



p1nesh4dow48 Yesterday at 3:13 PM
guys its such a nice day outside

d4wgbyte262 Yesterday at 3:14 PM
is that your apartment?

p1nesh4dow48 Yesterday at 3:14 PM
no...
why does it matter
smh

luvh4ck573 Yesterday at 3:14 PM
it for sure is

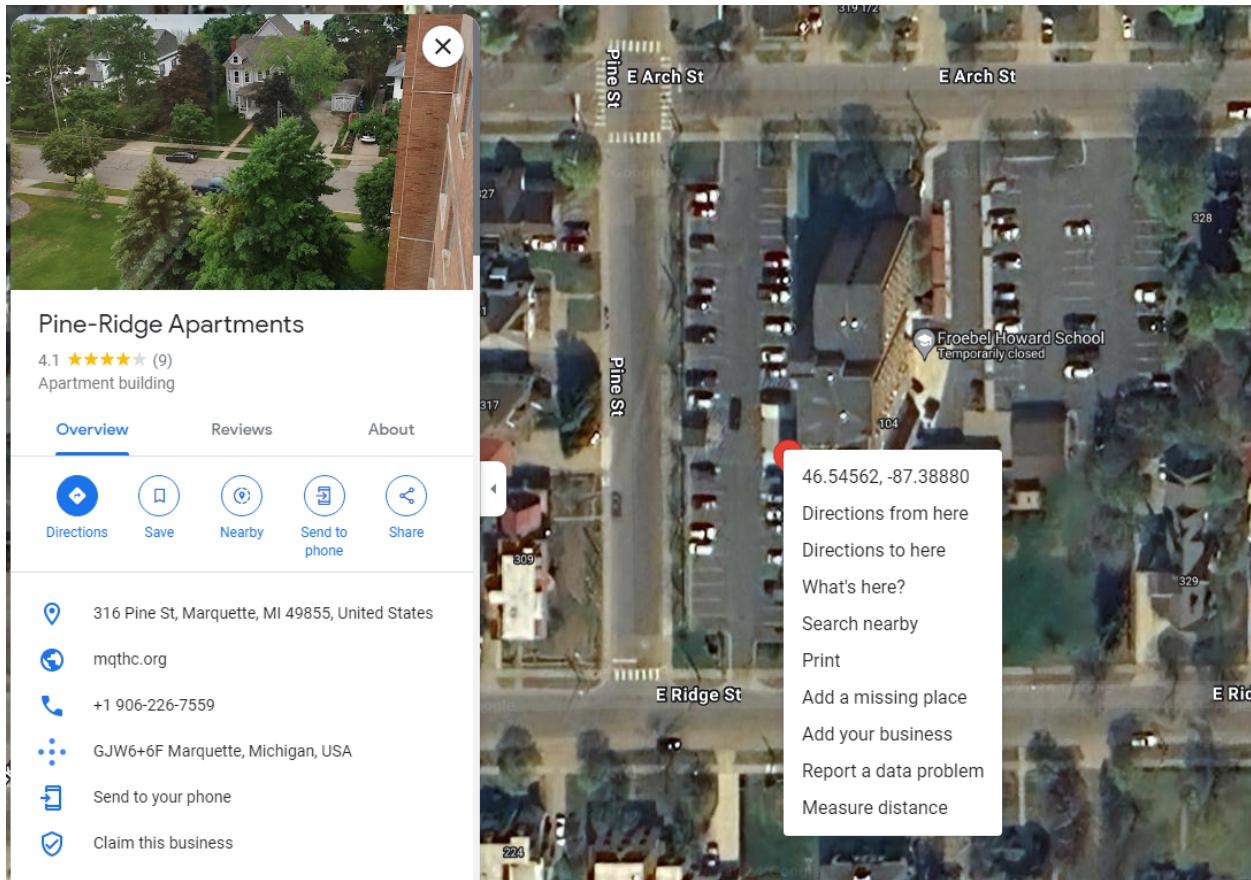
p1nesh4dow48 Yesterday at 3:14 PM
shush



The picture above was sent by p1nesh4dow48. From the bottom left corner, we can see its related to 'Pine Ridge'. Also, someone in the chat mentioned that it could be an apartment. With the information we have, we can look for 'Pine Ridge Apartment' on Google Map.

After a LONGGG search, I finally found the exact place (which I then realized I could have put 'Michigan' when I was searching to speed up the process 😊). We

can easily identify from the carparks on its both sides which matched the picture.



Limited 1 [crypto]

It's pretty easy to find random integers if you know the seed, but what if every second has a different seed?

Flag: wctf{f34R_0f_m1ss1ng_0ut}

The challenge provided the source code.

```

import time
import random
import sys

if __name__ == '__main__':
    flag = input("Flag? > ").encode('utf-8')
    correct = [189, 24, 103, 164, 36, 233, 227, 172, 244, 21
3, 61, 62, 84, 124, 242, 100, 22, 94, 108, 230, 24, 190, 23,
228, 24]
    time_cycle = int(time.time()) % 256
    if len(flag) != len(correct):
        print('Nope :(')
        sys.exit(1)
    for i in range(len(flag)):
        random.seed(i+time_cycle)
        if correct[i] != flag[i] ^ random.getrandbits(8):
            print('Nope :(')
            sys.exit(1)
    print(flag)

```

From the code, we know that for every time updated, the value of random.getrandbits will be changed and thus affects the xor key for the subsequent characters. Knowing that the flag format is `wctf{...}`, we can try to find the starting time cycle and the ending time cycle to know the time used and frequent of seed change when running the program.

Using the code below, we found the potential starting time_cycle value.

```

for i in range(256):
    random.seed(i)
    if 189 ^ random.getrandbits(8) == ord('w'):
        print(i)
# 86
# 188

```

```
>>> random.seed(1+86)
>>> chr(24 ^ random.getrandbits(8))
'='

>>> random.seed(1+188)
>>> chr(24 ^ random.getrandbits(8))
'c'
```

From the code above, we proofed that the time_cycle starts from 188 because the second character of the flag matched the output, which is `c`.

Now, we need to get the time_cycle value when XORing the last character of the flag with the following code.

```
for i in range(256):
    random.seed(i)
    if 24 ^ random.getrandbits(8) == ord('}'):
        print(i)

# 88
# 204
# 212
```

When counting the length of flag, surprisingly its $212 - 188 + 1 = 25$. Assuming that each character uses `time_cycle + 1` in each loop, I wrote the following code and retrieved the flag:

```
flag = ''
start = 188
for i in range(len(correct)):
    random.seed(start + i)
    flag += chr(correct[i] ^ random.getrandbits(8))

print(flag)
```

WOLPHV IV: d4wgbYTE262 [osint]

Locate d4wgbYTE262's home coordinates and submit them as

wctf{x.xxxx, x.xxxx} rouded

For example, if the location was the table next to front entrance to the BBB at the Unviersity of Michigan the flag would be

wctf{42.2926, -83.7162}

You will be able to get this specific

You have 10 tries to get the location. If you are *sure* you have location but the flag does not work, make a ticket

wctf{51.0912,-113.9562}

To look for d4wgbYTE262's home coordinates, there were some hints from the discord chat.

d4wgbYTE262 03/15/2024 3:16 PM
sry need to let my dog out brb

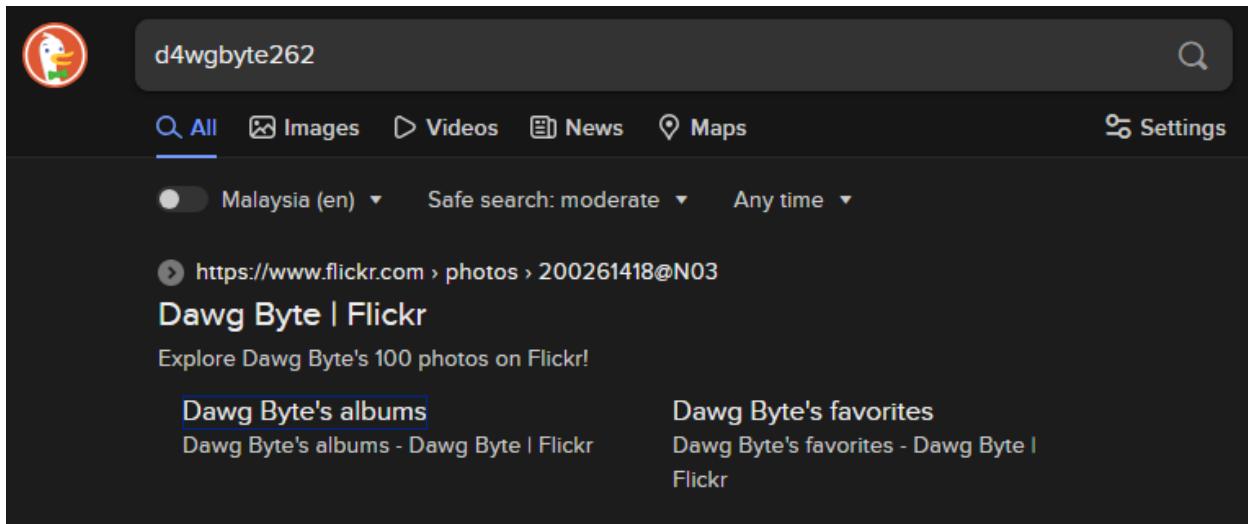
p1nesh4dow48 03/15/2024 3:17 PM
@d4wgbYTE262 how is ur dog
or should i say dawgg

luvh4ck573 03/15/2024 3:19 PM
hehe

@p1nesh4dow48 @d4wgbYTE262 how is ur dog
d4wgbYTE262 03/15/2024 3:21 PM
she's good! been taking a lot of flicks of her when i visit my neighbors and uploading them online

d4wgbYTE262 03/15/2024 4:52 PM
ugh why are firetrucks so loud
i live the closest to a fire station out of all my friends who are neighbors they dont understand my pain

Knowing that d4wgbyte262 might have an account posting his dog pictures and his house is the nearest to the fire station compared to his neighbours. Using duckduckgo search engine (because google was not showing anything), I found his account on Flickr. (actually taking flicks is a hint)



d4wgbyte262

All Images Videos News Maps Settings

Malaysia (en) Safe search: moderate Any time

<https://www.flickr.com/photos/200261418@N03>

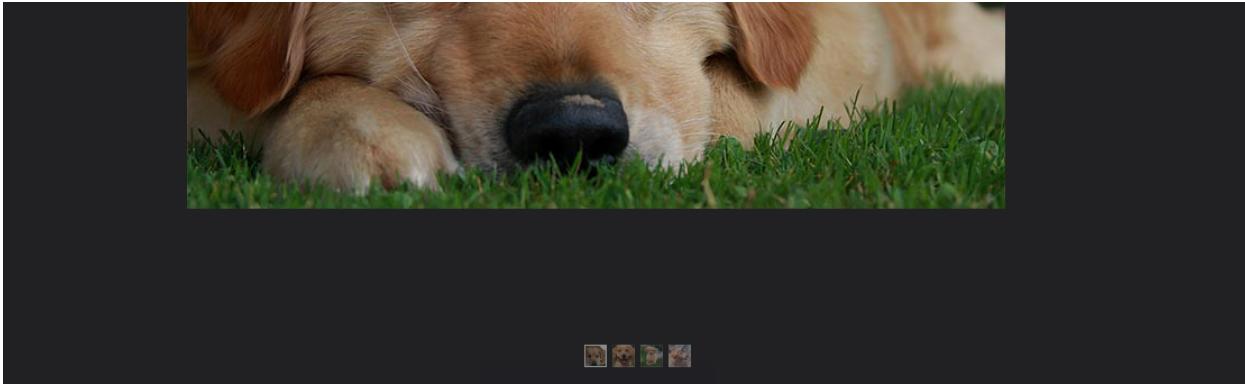
Dawg Byte | Flickr

Explore Dawg Byte's 100 photos on Flickr!

[Dawg Byte's albums](#) [Dawg Byte's favorites](#)

Dawg Byte's albums - Dawg Byte | Flickr Dawg Byte's favorites - Dawg Byte | Flickr

In his album, he mentioned "my dog at my neighbors' houses, well one of the pics was at my house but you get it" which means one of the coordinates of these photos is his house. Flickr contains exif data - location for all pictures - from what I see. By pressing the map on the EXIF data section, it brought us to a bird view of all locations that all those photos in his album was taken.



dog19



Dawg Byte

+ Follow

dog19

109

views

0

faves

0

comments

Uploaded on March 15, 2024

CC All rights reserved



Show your appreciation with the gift of Flickr Pro



Login to comment



Add comment

i Hide EXIF

X-Resolution - 72 dpi

Y-Resolution - 72 dpi

YCbCr Positioning - Centered

GPS Version ID - 2.3.0.0

GPS Latitude Ref - North

GPS Latitude - 51 deg 5' 28.22"

GPS Longitude Ref - West

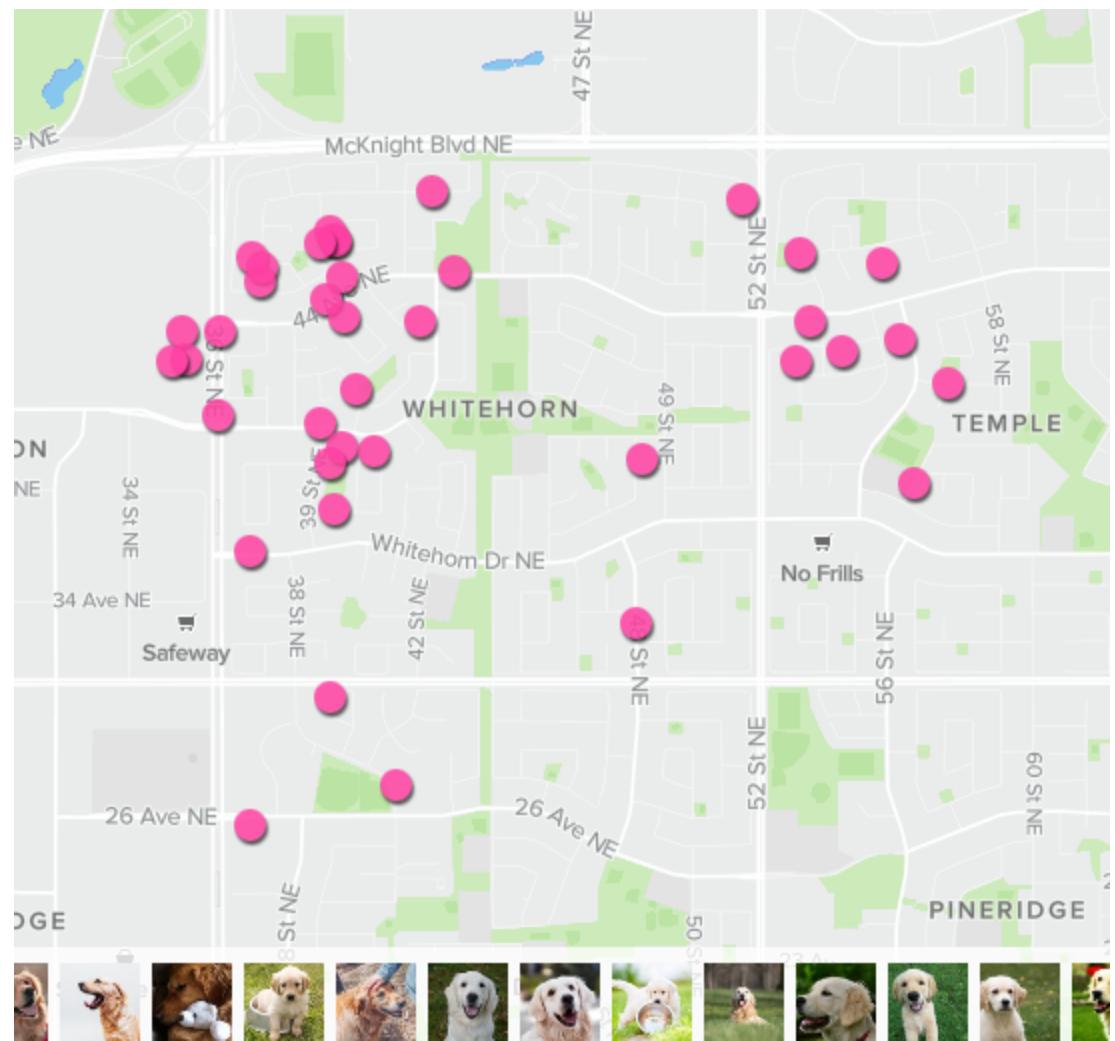
GPS Longitude - 113 deg 58' 22.60"

GPS Altitude Ref - Above Sea Level

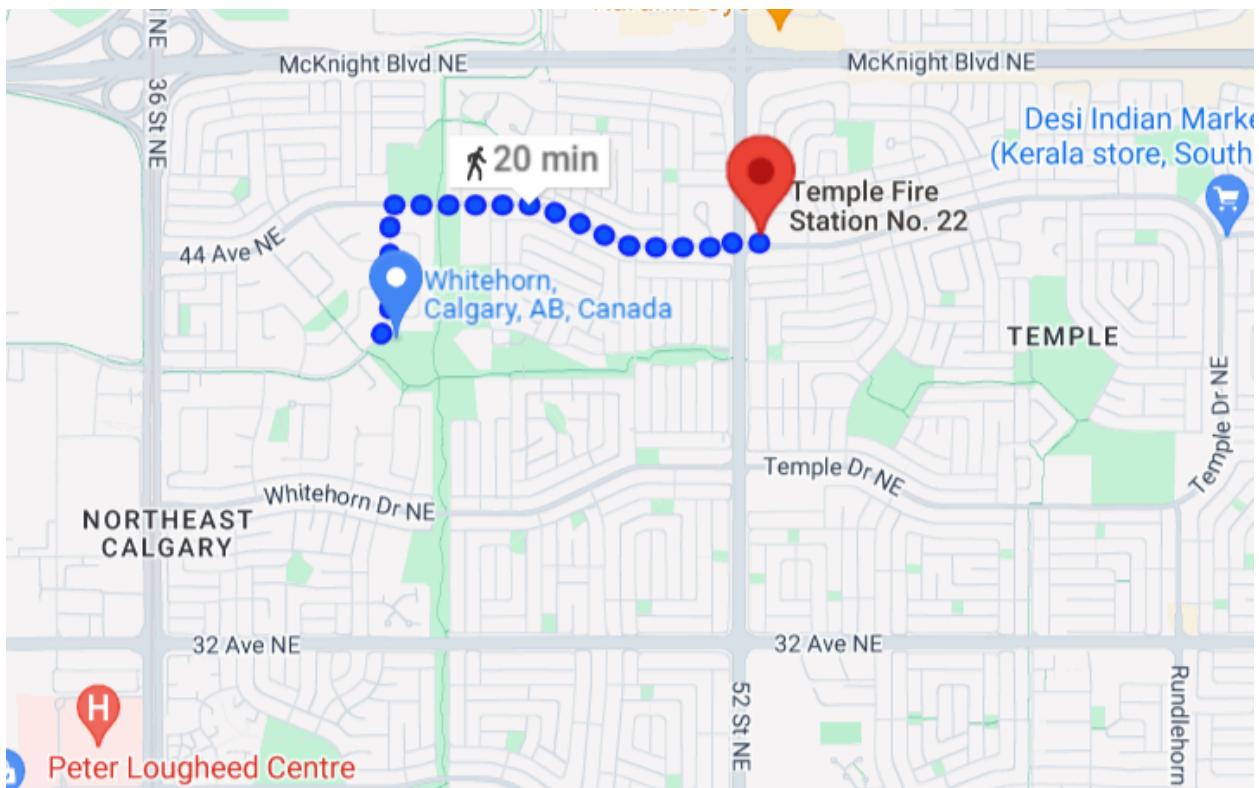
GPS Altitude - 1044.854 m



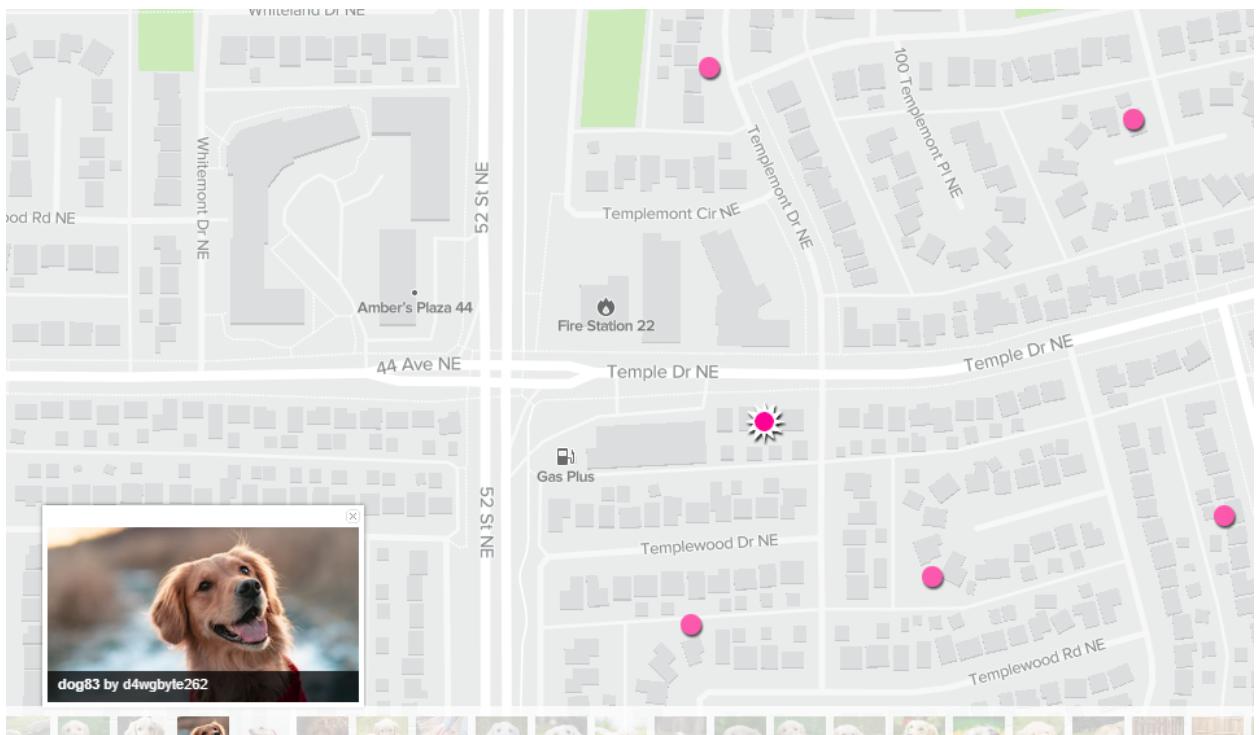
Whitehorn, Alberta, Canada

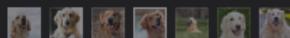
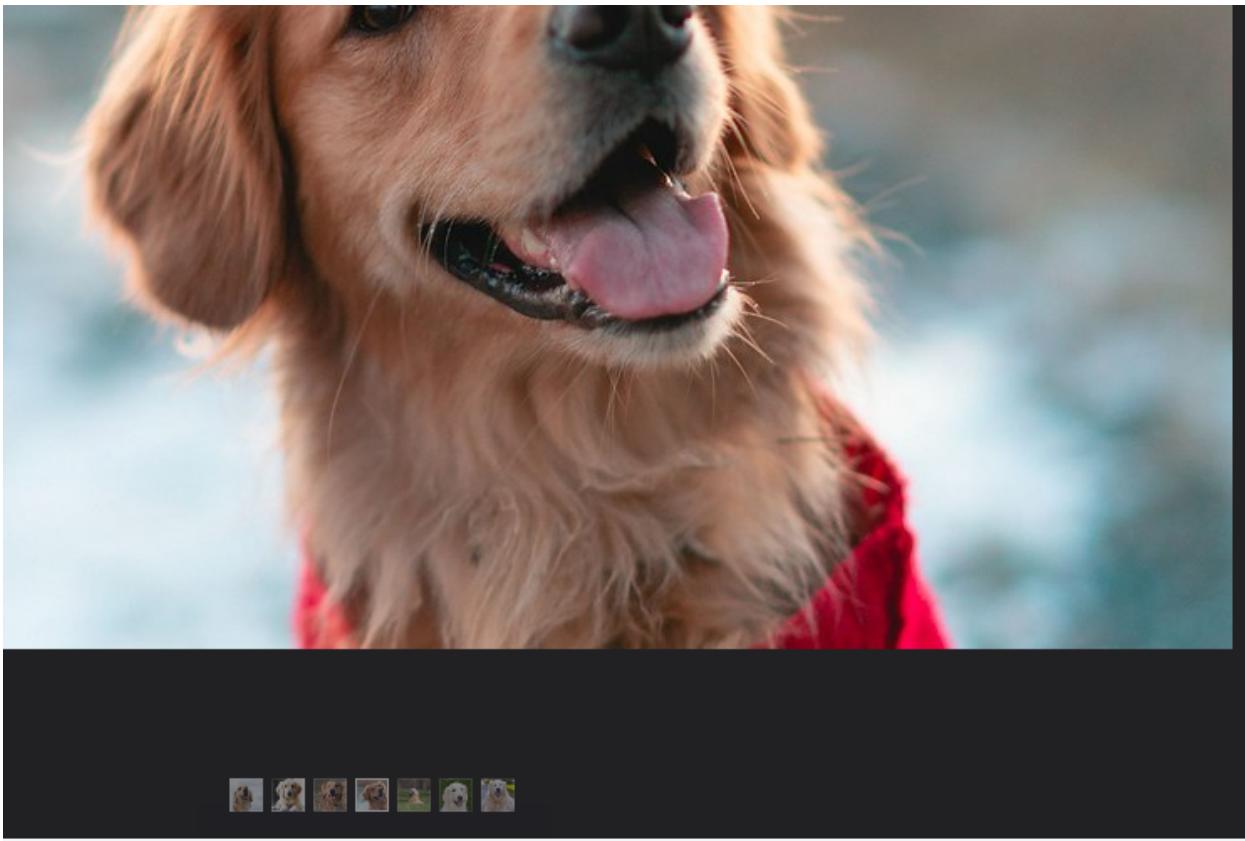


This whole map gave us a brief area of where he and his neighbors live. With this location information, we can narrow down to search for the nearest fire station to Whitehorn.



Comparing with other locations, we found the nearest location to the fire station, which is his house and retrieve the GPS coordinates from the EXIF data.





0
views

0
faves

0
comments

Uploaded on March 14, 2024

All rights reserved

of Flickr Pro

Hide EXIF

X-Resolution - 72 dpi

Y-Resolution - 72 dpi

YCbCr Positioning - Centered

GPS Version ID - 2.3.0.0

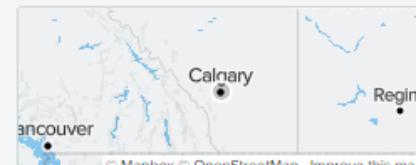
GPS Latitude Ref - North

GPS Latitude - 51 deg 5' 28.20"

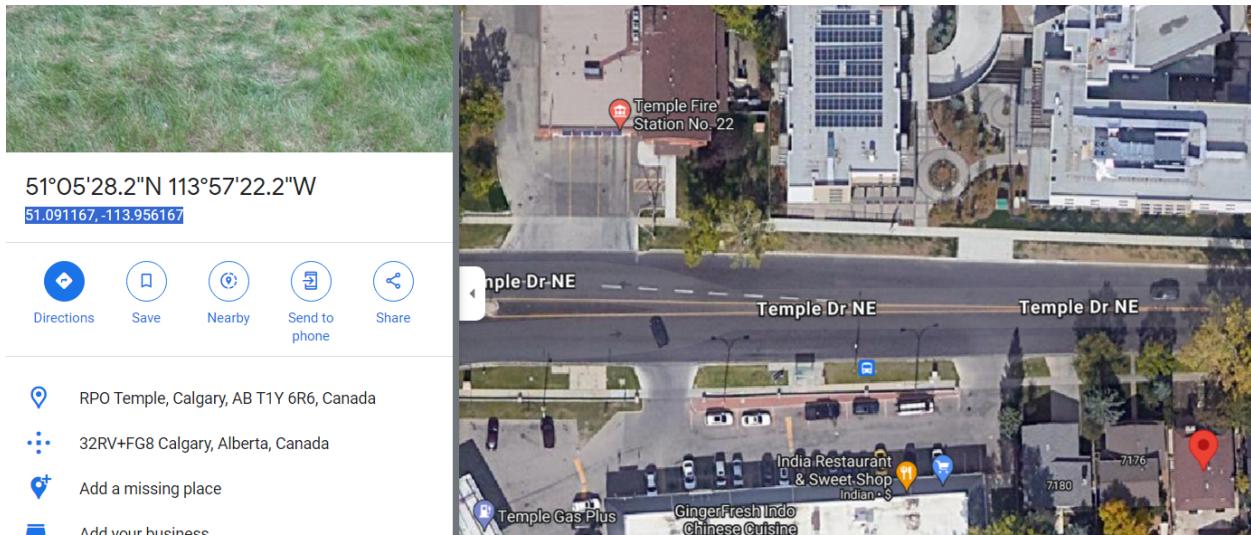
GPS Longitude Ref - West

GPS Longitude - 113 deg 57' 22.20"

GPS Altitude Ref - Above Sea Level



Add comment



Log Analysis [forensics]

Hi there incident responder. So we have this company that was breached sometime last week, but their SOC team only keeps HTTP request logs :(We took down all of our `wolvsecsolutions` websites as a precaution.

Maybe there's still a way to figure out what happened? Why did they click on a suspicious link? Somebody told me there's a flag on the link now?

Flag: wctf{ph1sh3r5_l0v3_c0py1ng_d0m41n_n4m35}

At first, while looking at the logs, I noticed there's a lot of random value in subdomain, which made me thinking, is it extracting values from the url?

```

api-ipperceptions01.cloudapp.net
api-ipperceptions02.cloudapp.net
api-ipperceptions04.cloudapp.net
apimgmths3ad6x7csidzffl92qtkn5chjzvdsmihcqem7p8.cloudapp.net
apimgmths3yw0rizei8npe17wtsn2x2fnjm5t9skxhi8pi.cloudapp.net
apimgmths789th6sr7bksbpiorwzwsge0nfz3gnxp01tzwhu.cloudapp.net
apimgmthsse02dkjat8bpmcsqswnqfssr1nlmbuyutchnhtv9j.cloudapp.net
apimgmthshy7tjbeerwopa36g1877ju3uk9jxeoeua0200.cloudapp.net
apimgmthsj1btj2xja67vohxuw0hiz5vluz2t8jrnasq019z.cloudapp.net
apimgmthsnene2efnh92jyitjm31olzsnuwdcum1dmjd806jwi.cloudapp.net
apimgmthswi1lyteufhkbkzmqblrudosxrnorlueinkaosoge.cloudapp.net
apimgmthswiux11wmq1mtwfwoed7f4ppgvphbdelydfrzysvg.cloudapp.net
apimgmthsxdwkwahgvpl6ziquorapxbtafrzx1a17h19ktf1hh6.cloudapp.net
apimgmthsydngwf6mmi452zzjyyuu05fhbkzbwu04f4bspsvd2.cloudapp.net
apimgmtn0mpedjk1kfavayucodjo8mfyqkompy/twci9wmrow.trafficmanager.net
apimgmtn1urd2rubxuaizmwcvkzw5kglex2gxczgcjskh7ey.trafficmanager.net
apimgmtn9hjscodlhd06farzkduttaem2mjeut1wqxhyvpcg.trafficmanager.net
apimgmtnmmiitrn0kxd13flydubz79igf6ervxsgwfjrahmqj.trafficmanager.net
apimgmtnmyc05ndhj8wrebu05qddzpath1ukgezkmcjg2v28.trafficmanager.net
api-scheduler-teams.trafficmanager.net
api-tvclient-scl.azurewebsites.net
api-userstore-skype.trafficmanager.net
app-admin-contentapi-prod.azurewebsites.net
appconfig-be7f37.firebaseioapp.com
app-digitalgate-prd-ol.azurewebsites.net
appexbingfinance.trafficmanager.net
appexbingweather.trafficmanager.net

content-prod3.trafficmanager.net
content.prod.powerquery.trafficmanager.net
contentsync.onenote.trafficmanager.net
copilot-proxy.trafficmanager.net
copilot-telemetry.githubusercontent.com
cosmic-eastus-ns-ab8c451a9521.trafficmanager.net
cosmicimg-prod.trafficmanager.net
cosmic-westeurope-ns-09ed083b10c0.trafficmanager.net
cosmic-westeurope-ns-13639f4dae2c.trafficmanager.net
cosmic-westeurope-ns-13f4d73d0237.trafficmanager.net
cosmic-westeurope-ns-266f7d7834c.trafficmanager.net
cosmic-westeurope-ns-4fcba45a4249.trafficmanager.net
cosmic-westeurope-ns-5a494359800f.trafficmanager.net
cosmic-westeurope-ns-5cb5ce1fc435.trafficmanager.net
cosmic-westeurope-ns-6772647de65d.trafficmanager.net
cosmic-westeurope-ns-6796f1a7bafb.trafficmanager.net
cosmic-westeurope-ns-70e0cb54fb5.trafficmanager.net
cosmic-westeurope-ns-758b3149c382.trafficmanager.net
cosmic-westeurope-ns-766607e8da1.trafficmanager.net
cosmic-westeurope-ns-7bc2119e5b60.trafficmanager.net
cosmic-westeurope-ns-97f334763d0.trafficmanager.net
cosmic-westeurope-ns-ade81d4a2f21.trafficmanager.net
cosmic-westeurope-ns-b97445317501.trafficmanager.net
cosmic-westeurope-ns-ec39ff6354c3.trafficmanager.net
cosmic-westus-ns-f2dbfd44519.trafficmanager.net
cosmic-westus-ns-5df172da98e8.trafficmanager.net
cosmic-westus-ns-f683781ec2c3.trafficmanager.net
cred-prod-01.trafficmanager.net
csas.symantec.com.trafficmanager.net

```

After some trial and error from the above urls with hex and base64, I realized its not THAT hard. I'll show you some of the dumb stuff I did.

```

# Retrieve urls only
grep -i "Host:" logs.txt | cut -d " " -f 2- | sort | uniq |
less

# Grep for base64 encoded formats in subdomain
grep -i "Host:" logs.txt | cut -d " " -f 2- | sort | uniq |
cut -d '.' -f 1 | grep -P '[A-Za-z0-9+/]{4,}' | less

# Grep for the domain for each urls without tld
grep -i "Host:" logs.txt | cut -d " " -f 2- | rev | cut -d " "
-f 2 | rev | sort | uniq | less

# Suspicious links may have longer length
grep -i "Host:" logs.txt | cut -d " " -f 2- | sort | uniq |
awk 'length($0) > 30' | less

```

Then, I realized the scenario mentioned something about `wolvsecsolutions` taken down. Let's take a look at the results.

```
└─(kali㉿kali)-[~/Downloads]
$ grep -i "Host:" logs.txt | cut -d " " -f 2- | sort | uniq | grep 'wolvsec'
data.wolvsecsolutions.wolvctf.io
dev2.wolvsecsolutions.wolvctf.io
dev3.wolvsecsolutions.wolvctf.io
dev.wolvsecsolutions.wolvctf.io
files.wolvsecsolutions.wolvctf.io
help.wolvsecsolutions.wolvctf.io
solutions.wolvsecsolutions.wolvctf.io
support.wolvsecsolutions.wolvctf.io
wolvsecsolutions-okntin33tq-ul.a.run.app
wolvsecsolutions.wolvctf.io
```

Seems like we found a phishing link that tried to imitate the original website. AND from the link we got the flag.

