

# Very very very Hidden

## Description :

Finding a flag may take many steps, but if you look diligently it won't be long until you find the light at the end of the tunnel. Just remember, sometimes you find the hidden treasure, but sometimes you find only a hidden map to the treasure. [try me.pcap](https://try-me.pcap).

1. Open the file with a pcap analyzer like Wireshark.
2. When I checked the HTTP traffic, I found two requests for PNG files: duck.png and evil\_duck.png. I downloaded these files by following these steps: File --> Export Objects --> HTTP --> Save All. I tried using "zsteg" and other steganography tools, but I couldn't reveal anything.
3. I decided to look at the other HTTP requests and the sites that were visited. I searched for this in Wireshark using the filter: (http.request or ssl.handshake.type==1). I found that many sites were visited, including GitHub, Amazon, Microsoft, and finally, PowerShell.org. To be honest, I have doubts about the last one, "PowerShell.org."

Look at this:

444/ 186.372312	192.168.1.189	52.114.133.105	TLSv1.2	571 Client Hello (SNI=powershell.org)
4584 110.441210	192.168.1.189	54.147.39.126	HTTP	546 GET /nothing\$ut/evil_duck.png HTTP/1.1
4605 110.459289	192.168.1.189	65.55.44.109	TLSv1.2	571 Client Hello (SNI=web.vortex.data.microsoft.com)
6481 124.387379	192.168.1.189	40.126.7.100	TLSv1.2	274 Client Hello (SNI=login.microsoftonline.com)
6540 124.899917	192.168.1.189	52.113.194.132	TLSv1.2	630 Client Hello (SNI=teams.microsoft.com)
6572 125.190643	192.168.1.189	142.250.73.206	QUIC	1392 Initial, DCID=ea33d6b91203db53, PKN: 1, CRYPTO, PADDING
6741 127.997120	192.168.1.189	104.21.2.183	HTTP	495 GET / HTTP/1.1
6749 128.043434	192.168.1.189	104.21.2.183	TLSv1.3	571 Client Hello (SNI=powershell.org)

After downloading "evil\_duck.png," there is a handshake with the PowerShell.org site.

Hmm, this made me think about the PowerShell script on GitHub called Invoke-PSImage. This script hides data within an image's pixels (more information on GitHub).

So, I searched for a tool that could extract data hidden by Invoke-PSImage, and I found this: [https://github.com/PCsXcetra/Decode\\_PS\\_Stego](https://github.com/PCsXcetra/Decode_PS_Stego). It's a Windows tool that can help us extract the hidden data.

After running the tool and opening the PNG file with it, we found the following script:

```
PowerShell Stego Decode 1.0.0.2
Select File      Output Length
Selected File    1907
C:\Users\EXTRA\Downloads\evil_duck.png
Get Data
Output
$out = "flag.txt"
$enc = [system.Text.Encoding]::UTF8
$string1 = "HEYWherE(I$INE)50uP?Dld_YOu(E@t'mY_3RD()B2g3l?"
$string2 = "8..8+14>Fx0l-$*KjVD>[o'..:1]"[n&2G^201l&Mv+_'T_B"

$sd1 = $enc.GetBytes($string1)
$bbytes = $enc.GetBytes($string2)

for($i=0; $i -lt $bbytes.count; $i++)
{
    $bbytes[$i] = $bbytes[$i] -bxor $sd1[$i]
}
[System.IO.File]::WriteAllBytes("$out", $bbytes)
%02587%04103Xcl00701u14AE,a0_2pqlm"Aoe%Y000[G%02$*C'x0U>%4x00%0fTl00)CXsCPN00H0nA^0D57#Y~j$-x<9^0^0sM#0BE50#3Xi080P^0AD
%0cIP0es7A HY^0n0ED0AE+7421A7]0U9400+0ZIE0:SCXyl[P00A000E^0710:0x$P,78..U^a^D..0E0h04]000%030zS7$0EJ.00[f0ey=kn0D^00 U00EG^7s^3Gq00B0.1]
```

Just xor string1 and string2 and you get the flag :  
picoCTF{n1c3\_job\_f1nd1ng\_th3\_s3cr3t\_in\_the\_im@g3}.