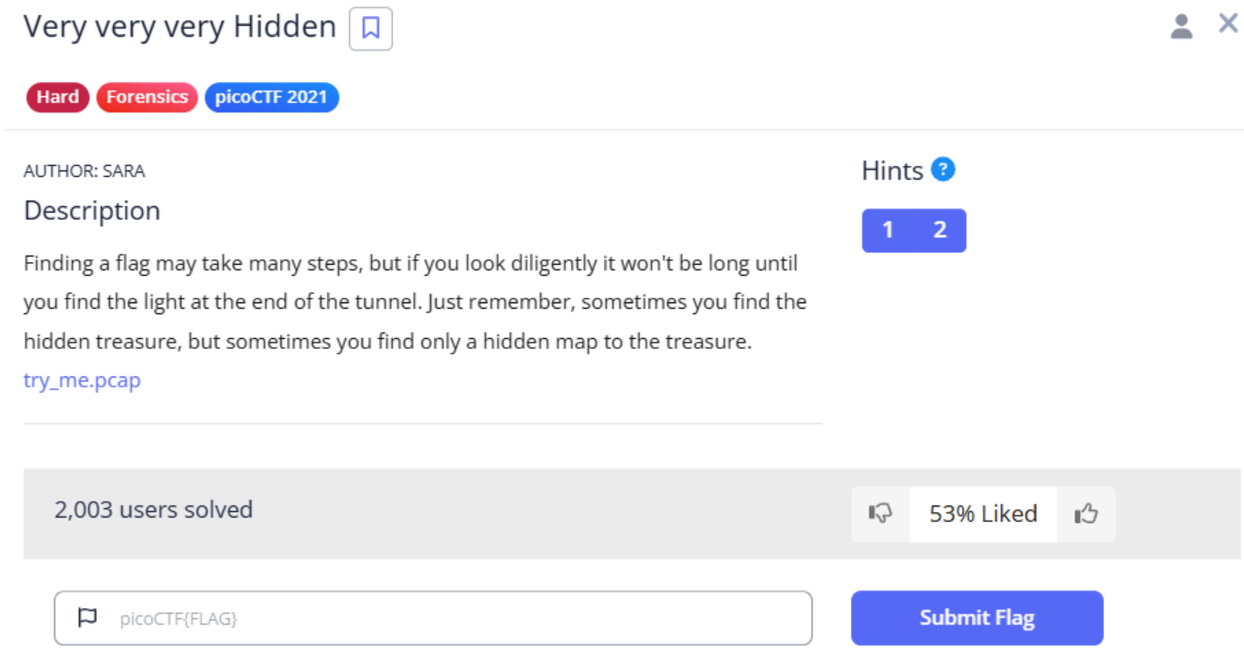
# Very very very Hidden



## 1. Open the capture and focus on clear-text traffic

99 % of frames are TLS noise; the interesting bits are the handful of vanilla-HTTP requests. Load `try_me.pcap` in wireshark and apply

```
(http.request or ssl.handshake.type == 1) and !(udp.port == 1900)
```

Five HTTP requests remain; two of them fetch images from an AWS host:

```
GET /NothingSus/duck.png HTTP/1.1
GET /NothingSus/evil_duck.png HTTP/1.1
```

## 2. Export the objects

`File ▸ Export Objects ▸ HTTP ▸ Save All`
Now we have `duck.png` (≈ 45 kB) and `evil_duck.png` **(≈ 2.6 MB)** on disk.

## 3. Identify the hiding method

A sweep with `binwalk`, `zsteg`, `steghide`, etc. comes up blank.
So next, I look at the browsing history we captured. Among the HTTPS SNI fields there's a `powershell.org`. That, plus a gigantic PNG, screams **PowerShell/Invoke-PSImage** steganography.

## 4. Extract the hidden PowerShell payload

Any Invoke-PSImage decoder works; the Windows-friendly binary in **PCsXcetra/Decode_PS_Stego** is the easiest drop-in tool, you can get it here here, after which you can run the following:

```
.\PowershellStegoDecode.exe
```

The decoder spits out a script

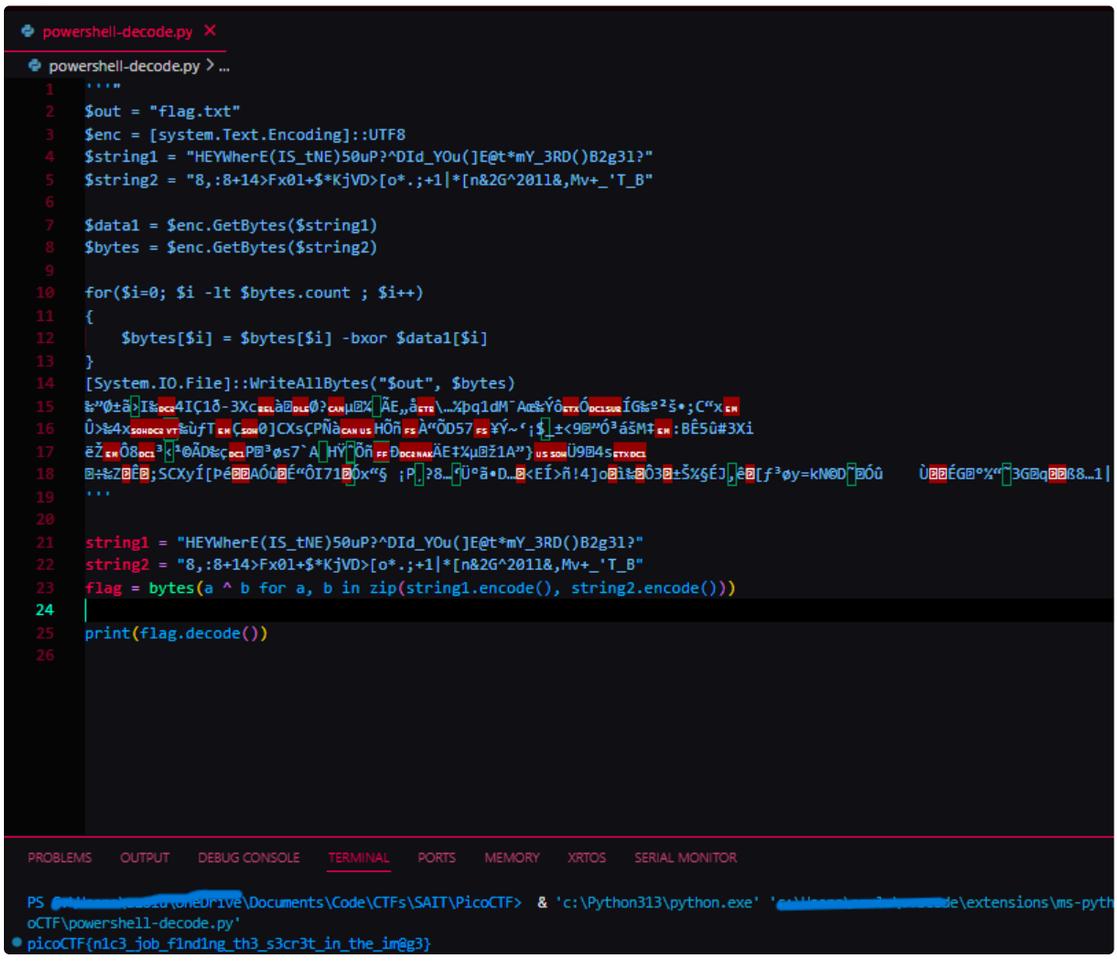
```
$out = "flag.txt"
$enc = [system.Text.Encoding]::UTF8

$string1 = "HEYWherE(IS_tNE)50uP?^DId_YOu(]E@t*mY_3RD()B2g3l?"
$string2 = "8,:8+14>Fx0l+$*KjVD>[o*.;+1|*[n&2G^201l&,Mv+_'T_B"

$data1 = $enc.GetBytes($string1)
$bytes = $enc.GetBytes($string2)
for($i=0; $i -lt $bytes.count ; $i++)
{
    $bytes[$i] = $bytes[$i] -bxor $data1[$i]
}
[System.IO.File]::WriteAllBytes("$out", $bytes)
%”Ø±ã›I%□4IÇ1ð−3Xc□à□□Ø?□µ□¾ ÃE„å□\…¾þq1dM¯Aœ%Ýô□Ó□□ÍG%°²š•;C“x□
Û>%4x□□□%ùƒT□Ç□0]CXsÇPÑà□□HÕñ□À“ÕD57□¥Ý~‘¡$_±<9□”Ó³ášM‡□:BÊ5û#3Xi
ëŽ□Ô8□³‹¹©ÃD%ç□P□³øs7`A HŸˆÕñÐ□□ÄE‡¾µ□ž1A”}□□Ü9□4s□□
□÷%Z□Ê□;SCXyÍ[Þé□□AÓû□É“ÔI71□Óx“§ ¡P¸?8…‘Ü°ã•D…□<EÍ>ñ!4]o□ì%□Ô3□±Š½§ÉJ¸ê□[ƒ³øy=kN©D˜□Óû    Ù□□ÉG□º½“˜3G□q□□ß8…1|
```

## 5. Recover the flag

I run this python script to get the flag

```
s1 = b"HEYWherE(IS_tNE)50uP?^DId_YOu(]E@t*mY_3RD()B2g3l?"
s2 = b"8,:8+14>Fx0l+$*KjVD>[o*.;+1|*[n&2G^201l&,Mv+_'T_B"
flag = bytes(a ^ b for a, b in zip(s1, s2))
print(flag.decode())
```

Output:




```
picoCTF{n1c3_job_f1nd1ng_th3_s3cr3t_in_the_im@g3}
```