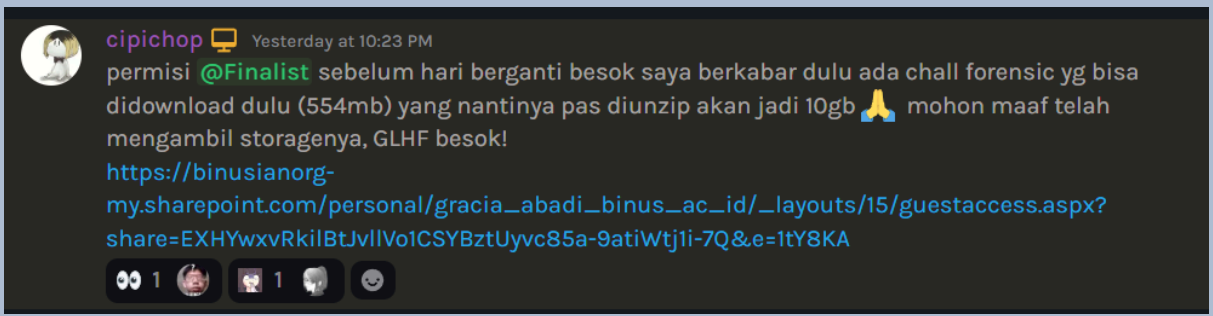
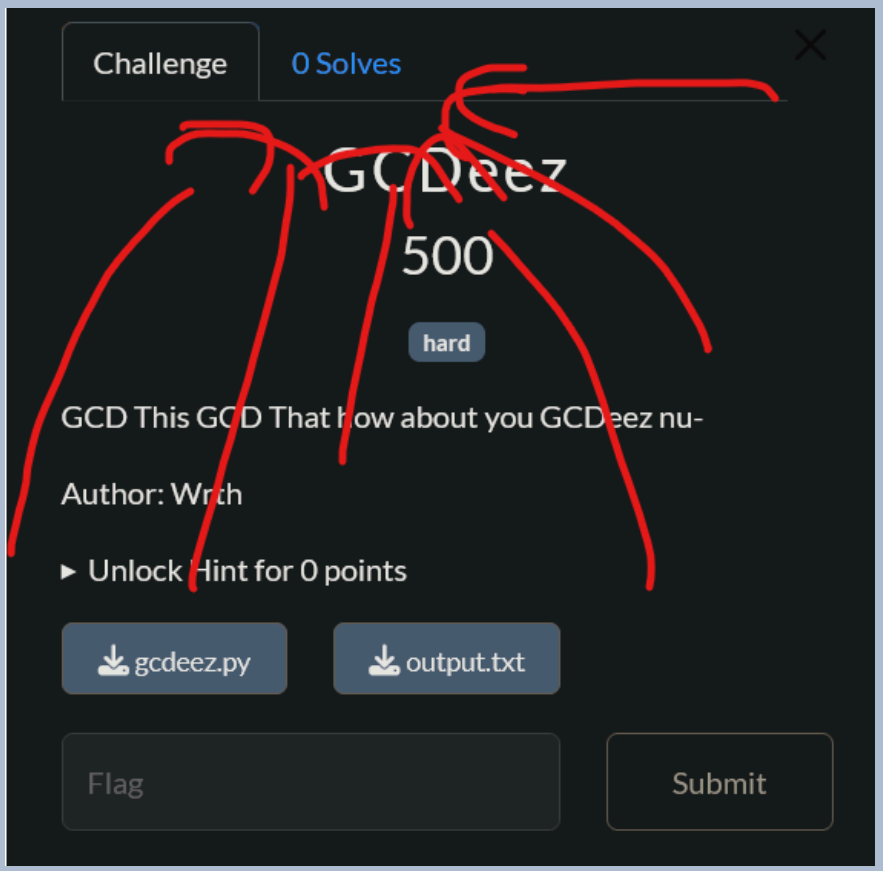
Write-Up Final National Cyber Week (NCW) 2024

HCS - NyanSetsunaNyan





DJumanto

(> _ <)

Etern1ty

Daftar Isi

CRYPTOGRAPHY

A4s lagi 4x

Flag: NCW{awoadkoawdkoawdadko_gw_hampir_lupa_bikin_soal_NCW_lol}

Deskripsi

flags: medium

kemarin R5A lagi, skarang A4S lagi
p.s: Masih sama kek qual, semoga gaada salah setting lagi kek IFEST kemarin 😭

Author: Lawson Schwantz

Informasi Terkait Soal

Diberikan 2 file: **A4s4x.py** dan **output.txt**.

A4s4x.py

```
from Crypto.Cipher import AES
from Crypto.Util.Padding import pad
from Crypto.Util.number import *
from random import getrandbits

FLAG = b'NCW{REDACTED}'

def findrand(intval, stateboolstatus):
      intval = ((intval << 46) | (intval >> 18)) & 18446744073709551615
      if not stateboolstatus:
      return intval & 4294967295, intval
      else:
      return (intval >> 32) & 4294967295, intval

intval = getrandbits(64)

keys, ivs = [], []

for i in range(4):
      if i % 2 == 0:
      temp,intval = findrand(intval,stateboolstatus=False)
      elif i % 2 == 1:
      temp,intval = findrand(intval,stateboolstatus=True)
      keys.append(long_to_bytes(temp))

for i in range(4):
```

```
      if i % 2 == 0:
      temp,intval = findrand(intval,stateboolstatus=False)
      elif i % 2 == 1:
      temp,intval = findrand(intval,stateboolstatus=True)
      ivs.append(long_to_bytes(temp))

key = b''.join(keys)
iv = b''.join(ivs)

cipher = AES.new(key, AES.MODE_CBC, iv)
enc = cipher.encrypt(pad(FLAG,16))

print(f'enc = {enc.hex()}')
print(f'iv = {iv.hex()}')
```

output.txt

```
enc =
c2206a116aa34a455d0c97f7b37d8d34250a71286bcf728b0a85980790ed07f9eb7d3867
e81450d453c312dbc8fdcac8c18851044e7907a0ff6e1b9d397caa95
iv = c929d6a775a9d9b0766c1c4a07129f24
```

Pendekatan

Kita mempunyai 2 parameter dari output, **enc** dan **iv**, yang dimana IV sendiri berasal sistem yang sama untuk melakukan enkripsi ke keys. IV yang kita dapat berasal dari 4 bagian, jadi kita bisa pisah terlebih dahulu(big-endian). Kemudian, kita dapat mencoba untuk mencari **intval** yang ke-4 karena kita bisa mencari intval 0 sampai 3 yang dipakai untuk pembuatan key, yang dimulai dari pencarian **intval** yang ke-7, yang dilanjut dengan **intval_6** dan **intval_5**.

Karena tentunya misal kita mencoba mencari value-value **intval** dengan pencarian biasa, akan sangat memakan waktu. Disini saya mencoba untuk memanfaatkan **multiprocessing** python (tiap CPU bekerja sesuai **step_size** nantinya), sehingga pencarian menjadi lebih cepat. Setelah itu kita bisa melakukan terus melakukan reverse terhadap operasi-operasi yang dilakukan dan dicocokan ke nilai **IV** yang kita punya (iv[1], iv[2], iv[3]).

Misal kita sudah mendapatkan **intval_4**, maka kita bisa mendapatkan **intval_0** sampai **intval3**, yang kemudian bisa dilakukan rekontruksi key yang dipakai untuk enkripsi AES awal. Kemudian karena kita tau format flag **NCW{** maka tinggal exit semua subprocess pas udah ketemu :)

Solusi

solver.py

```
from Crypto.Cipher import AES
from Crypto.Util.Padding import unpad
import multiprocessing, sys

def reverse_rotate(intval):
      return ((intval >> 46) | (intval << 18)) & 18446744073709551615

def worker(args):
      temp_iv, stateboolstatus_list, enc, iv, temp, stateboolstatus,
lower_32_bits_start, lower_32_bits_end = args
      for lower_32_bits in range(lower_32_bits_start,
lower_32_bits_end):

      intval_7 = (temp << 32) | lower_32_bits

      intval_6 = reverse_rotate(intval_7)
      stateboolstatus = stateboolstatus_list[6]
      if stateboolstatus:
            temp_check = (intval_6 >> 32) & 4294967295
      else:
            temp_check = intval_6 & 4294967295

      if temp_check != temp_iv[1]:
            continue

      intval_5 = reverse_rotate(intval_6)
      stateboolstatus = stateboolstatus_list[5]
      if stateboolstatus:
            temp_check = (intval_5 >> 32) & 4294967295
      else:
            temp_check = intval_5 & 4294967295

      if temp_check != temp_iv[2]:
            continue

      intval_4 = reverse_rotate(intval_5)
      stateboolstatus = stateboolstatus_list[4]
      if stateboolstatus:
            temp_check = (intval_4 >> 32) & 4294967295
      else:
            temp_check = intval_4 & 4294967295
```

```
        if temp_check != temp_iv[3]:
                continue

        # find key by intval_4
        intvals = [0]*8
        intvals[4] = intval_4
        intval = intval_4
        for i in reversed(range(4)):
                intval = reverse_rotate(intval)
                intvals[i] = intval

        key_parts = []
        for i in range(4):
                stateboolstatus = stateboolstatus_list[i]
                if stateboolstatus:
                        temp_key = (intvals[i] >> 32) & 4294967295
                else:
                        temp_key = intvals[i] & 4294967295
                key_parts.append(int.to_bytes(temp_key, 4, byteorder='big'))

        key = b''.join(key_parts)

        cipher = AES.new(key, AES.MODE_CBC, iv)
        try:
                decrypted = unpad(cipher.decrypt(enc), 16)
                if b'NCW{' in decrypted:
                        print(decrypted.decode())
                        sys.exit(0) # exit all processes
        except:
                continue
        return

def solve():
        enc =
bytes.fromhex('c2206a116aa34a455d0c97f7b37d8d34250a71286bcf728b0a8598079
0ed07f9eb7d3867e81450d453c312dbc8fdcac8c18851044e7907a0ff6e1b9d397caa95'
)
        iv = bytes.fromhex('c929d6a775a9d9b0766c1c4a07129f24')

        iv_parts = [int.from_bytes(iv[i:i+4], byteorder='big') for i in
range(0, 16, 4)] # bigendian
        stateboolstatus_list = [False, True, False, True, False, True,
False, True]
        temp_iv = iv_parts[::-1] # reverse

        temp = temp_iv[0]
```

```
      stateboolstatus = stateboolstatus_list[7]

      print('init')
      num_processes = multiprocessing.cpu_count()
      pool = multiprocessing.Pool(processes=num_processes)
      args_list = []

      step_size = (1 << 32) // (num_processes * 16)
      if stateboolstatus:
      for i in range(0, 1 << 32, step_size):
            lower_32_bits_start = i
            lower_32_bits_end = min(i + step_size, 1 << 32)
            args = (temp_iv, stateboolstatus_list, enc, iv, temp,
stateboolstatus, lower_32_bits_start, lower_32_bits_end)
            args_list.append(args)
      else:
            pass

      pool.map(worker, args_list)
      pool.close()
      pool.join()

      print('done')

solve()
```

Hasil

```
> python3 solver.py
init
NCW{awoadkoawdkoawdadko_gw_hampir_lupa_bikin_soal_NCW_lol}
done

> eter ~/../cry/a4s_lagi_4x
```

REVERSE ENGINEERING

19s

Flag: NCW{sch3m3_seems_legit_inspiredfromsekaictf:)!!!!}

Deskripsi

Just a warmup final RE, anyway 2024 is ending so I'm bringing you back the 1970s because I'm already rusty to create a challenge and hands out hard ones to the new generation. Don't you agree?

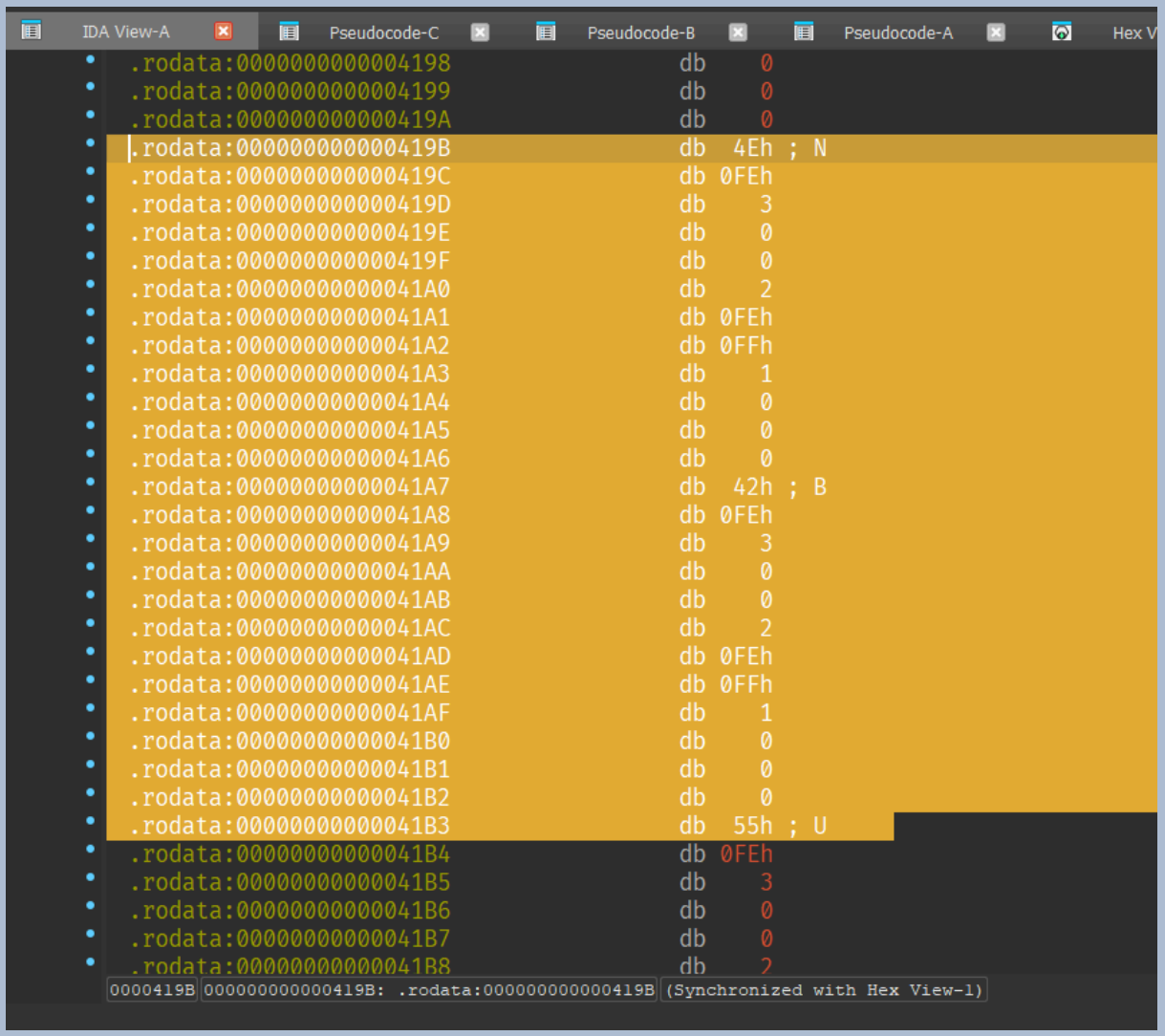
Author: aseng

Penjelasan

Diberikan sebuah binary yang dynamic linking sama libchicken.so.11. Lalu fungsi comparison terdapat pada f_269.

```
1  void __fastcall f_269(__int64 c, __int64 *av)
2  {
3    _QWORD *v2; // r14
4    unsigned __int64 v3; // r13
5    __int64 v4; // rax
6    __int64 v5; // [rsp-Eh] [rbp-60h] BYREF
7    __int64 *a[6]; // [rsp+22h] [rbp-30h] BYREF
8
9    v2 = (_QWORD *)*av;
10   v3 = av[1];
11   a[1] = (__int64 *)__readfsqword(0x28u);
12   if ( --C_timer_interrupt_counter <= 0 )
13     goto LABEL_6;
14   while ( 1 )
15   {
16     if ( ((__int64)a - C_stack_limit) >> 3 <= (c < 2) + C_scratch_usage + 5 )
17     {
18       C_save_and_reclaim(f_269, (unsigned int)c, av, a);
19       JUMPOUT(0x2D88LL);
20     }
21     a[0] = &v5;
22     v4 = C_s_a_i_bitwise_xor(a, 2LL, (v3 >> 7) & 0x3FFFFE | 1, v2[2]);
23     if ( C_i_nequalp(v2[3], v4) != 6 )
24       f_238(v2[4], 30LL);
25     *(_QWORD *)(lf[6] + 8) = 1LL;
26     f_238(v2[4], 1LL);
27 LABEL_6:
28     C_raise_interrupt(255LL);
29   }
30 }
```

Kita bisa sederhanakan (t2 >> 7) & 0x3FFFFE | 1 menjadi 1. Tapi kalau di solver pakek xor entah kenapa :)
Lalu flag yang di enkripsi terdapat pada &unk_4190.



Nah awalnya saya kira ini tiap karakter di + sama current counter, jadi misal karakter pertama itu N + 0 jadi N, terus B + 1 jadi C, U + 2 jadi W. Tapi kalau begitu, flag nya broken

Terus saya coba buat xor sama 1, juga gak bisa. Lalu saya teringat jika suatu karakter di xor dengan 0 akan menghasilkan karakter yang sama, disini saya deduksi (baca: dukun) kalau tiap karakter itu di xor dengan suatu counter, dan benar saja kami mendapatkan flag nya.

Solusi

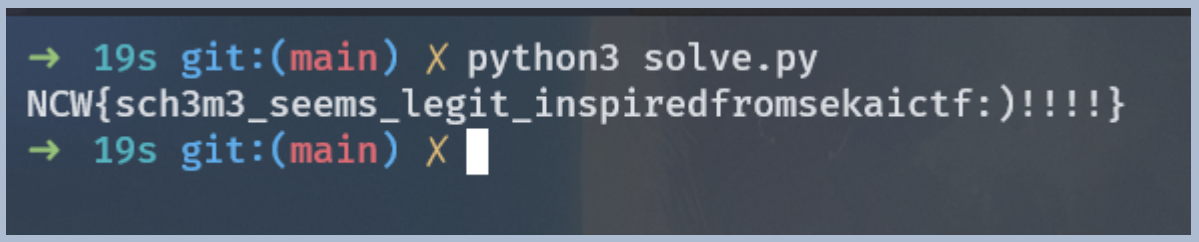
solve.py

```
chars =
b'NBUxwfn4e:Uxihc|O}wt}aI~vjjrnxzyRNOPANGNK]L\x11\x05\x0c\x0f\x0e\x11\x4c'

counter = 0
flag = ''
for i in range(len(chars)):
    flag += chr(chars[i] ^ counter)
    counter += 1

print(flag)
```

Hasil



FORENSIC

意外运行恶意应用程序

Flag: NCW{so_sad_how_this_issue_is_staled_for_months_but_once_you_know_what_to_grep_you_will_get_everything_yay}

Deskripsi

flags: medium

Check the attached file for more information. Password: ncwgogo

Author: cipichop

```
nc 103.145.226.92 18172
```

Informasi Terkait Soal

Diberikan dua file, **description.txt** dan suatu zip (**2836557D-8BC7-4-20241101-141327.zip**) yang berisikan memory dump 10GB 💀

description.txt
This is my story: I have never had any malware installed on my host. I am curious and want to execute some malware I found in Windows Sandbox. It was actually fun, but I am more intrigued by Windows Sandbox itself. I captured its memory to analyze the structure, and it seems very different from an ordinary memory capture. 1. Tell me, where is the source of the malware I used to download? (just the main link, without the subdirectories) 2. How many malwares have I downloaded? 3. When did I run the last malware? 4. I like the Windows Vista prank one. What is its PID? 5. Out of all malware I downloaded, how many did I successfully execute? 6. Mention all malware I executed, separated by space (order does not matter) Challenge file: https://binusianorg-my.sharepoint.com/personal/gracia_abadi_binus_ac_id/_layouts/15/guestaccess.aspx?share=EfKB8LK7jwZLmEcxlmrnp70B8TVP-6JT99_g7aysE9dhyA&e=nw6ge3 Warning: The uncompressed size of this challenge file is 10GB

Pendekatan

Setelah kita unzip, kita mendapatkan file .raw, yang indikatif ke file memory dump. Sebenarnya proses buat mencoba “membuka” file .raw ini yang sangat memakan waktu

lama bagi saya, karena file ini tidak bisa dibuka oleh tool Volatility, yang biasa digunakan untuk analisis file memory dump (3 jam mikirin volatility 😭) sampai akhirnya saya mencoba untuk mencari pendekatan lain. Soal pertama: source malware, berarti harusnya berasal dari URL yang valid, jadi saya coba saja command berikut:

```
strings 2836557D-8BC7-4-20241101-141327.raw | grep 'https://' >
output.txt
```

Terlihat banyak hasil yang menunjukkan URL-URL valid, dan setelah scroll dikit terlihat URL yang lucu: https://github.com/Da2dalus/The-MALWARE-Repo, yang merupakan jawaban dari pertanyaan no 1! Jadi disini saya terus memakai strategi strings > grep > save ke file untuk soal-soal selanjutnya.

Pertanyaan / Jawaban

1. Tell me, where is the source of the malware I used to download? (just the main link, without the subdirectories)

```
strings 2836557D-8BC7-4-20241101-141327.raw | grep 'https://' >
output.txt
```

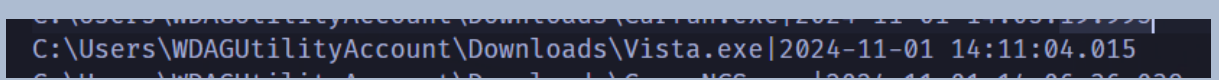
2. How many malwares have I downloaded?

```
strings 2836557D-8BC7-4-20241101-141327.raw | grep '.*Downloads.*.exe' >
output2.txt
```

Curfun, DumpIt, CrazyNCS, ColorBug, DesktopBoom, OperaSetup.exe, Vista.exe, Melting.exe, Mantas.exe

Total ada **9** malware yang didownload.

3. When did I run the last malware?



4. I like the Windows Vista prank one. What is its PID?

```
strings 2836557D-8BC7-4-20241101-141327.raw | grep 'Vista.exe' >
output3.txt
```

```
TRACE,0008,6440,LogEvents,ChainStart,C:\Users\WDAGUtilityAccount\Downloads\Vista.exe,Genome,WinBlueRTM,Value,(
TRACE,0000,0000,Service,6440 | C:\Users\WDAGUtilityAccount\Downloads\Vista.exe | "C:\Users\WDAGUtilityAccount\
TRACE,0000,0000,Service,RAiMonitorProcess, ClientProcessId: 4080 | C:\Users\WDAGUtilityAccount\Downloads\Vista
TRACE,0008,6440,LogEvents,ProcessStart,C:\Users\WDAGUtilityAccount\Downloads\Vista.exe
```

6440

5. Out of all malware I downloaded, how many did I successfully execute?

```
strings 2836557D-8BC7-4-20241101-141327.raw | grep '.*Downloads.*.exe' >
output2.txt
```

Cari yang ada TRACE seperti Vista.exe, dan total ada **4**.

6. Mention all malware I executed, separated by space (order does not matter)

Melting.exe CrazyNCS.exe Vista.exe Curfun.exe

Solusi

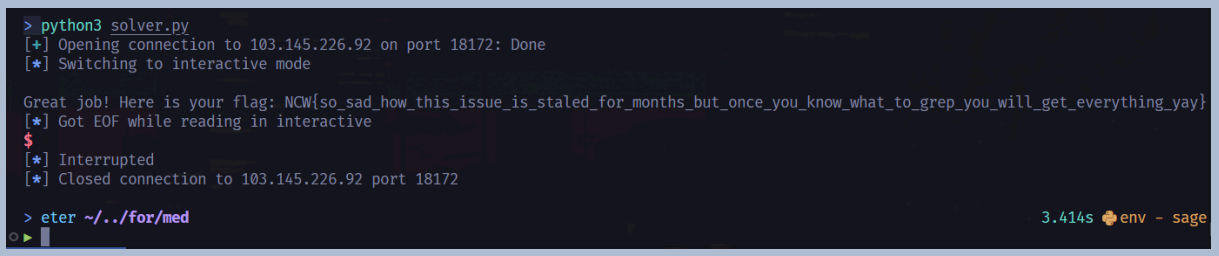
solver.py

```
from pwn import *

# nc 103.145.226.92 18172
def solve():
      r = remote('103.145.226.92', 18172)
      r.sendlineafter(b'>> ',
b'https://github.com/Da2dalus/The-MALWARE-Repo')
      r.sendlineafter(b'>> ', b'9')
      r.sendlineafter(b'>> ', b'2024-11-01 14:11:04')
      r.sendlineafter(b'>> ', b'6440')
      r.sendlineafter(b'>> ', b'4')
      r.sendlineafter(b'>> ', b'Melting.exe CrazyNCS.exe Vista.exe
Curfun.exe')
      r.interactive()

solve()
```

Hasil


> python3 solver.py
[+] Opening connection to 103.145.226.92 on port 18172: Done
[*] Switching to interactive mode
Great job! Here is your flag: NCW{so_sad_how_this_issue_is_staled_for_months_but_once_you_know_what_to_grep_you_will_get_everything_yay}
[*] Got EOF while reading in interactive
$
[*] Interrupted
[*] Closed connection to 103.145.226.92 port 18172
> eter ~/../for/med
3.414s env - sage

ある日私が雲の中にいて攻撃されたとき

Flag: NCW{ingin_menjadi_cloud_engineer_handal}

Deskripsi

flags: hard

Check the attached file for more information.

Author: kangwijen

```
nc 103.145.226.92 23456
```

Informasi Terkait Soal

Diberikan 1 file, yaitu **description.txt**.

description.txt

WijenBank recently suffered a cyberattack targeting its Tokyo cloud infrastructure. Attackers exploited leaked AWS credentials where they then infiltrated our development environment, accessed restricted resources, and compromised a critical S3 bucket, and ultimately breached a production server on DigitalOcean. In response, WijenBank has shut down its services and is actively working to restore the affected infrastructure. As part of the incident response team, your task is to retrace the attacker’s steps and evaluate the full extent of the compromise.

You are tasked with providing answers to the following questions:
1. What is the Access Key ID of the leaked AWS credentials?
2. What is the invoke URL of the other API Gateway that exists?
3. How many Lambda functions were deployed in the AWS account?
4. What is the full URL of the compromised S3 bucket?
5. What is the valid AWS Access Key ID that was exposed in the S3 bucket?
6. What is the statement ID that allowed public access to the directory with the valid AWS Access Key ID?
7. What is the IP address of the DigitalOcean production server?
8. What is the password of the account "deploy"?
9. What is the MD5 hash of the file left by the attacker inside the app deployment directory on the DigitalOcean production server?
10. How many non-default environment keys were set on the DigitalOcean production server?

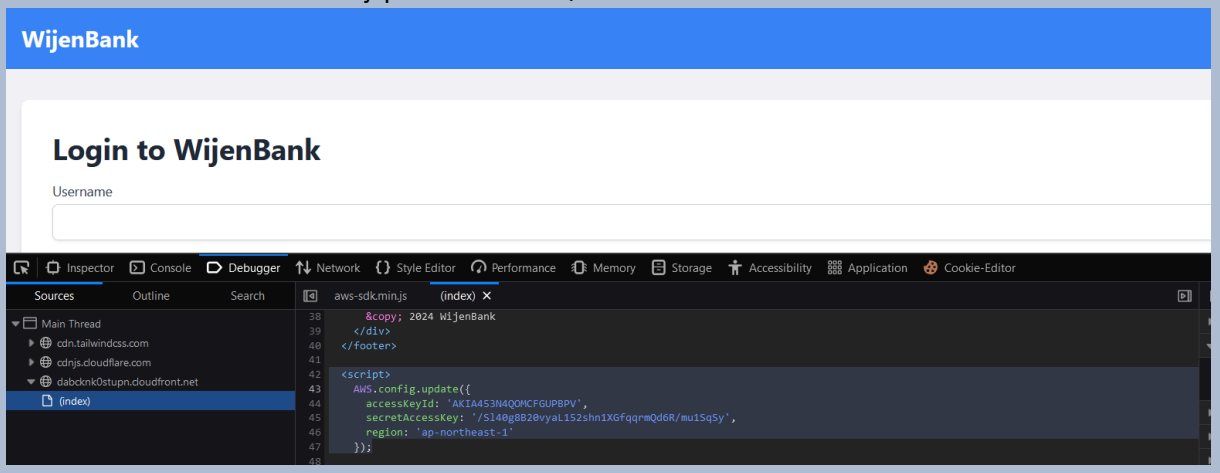
You're given this following information to start your investigation:
https://dabcknk0stupn.cloudfront.net/

Note:
1. You are allowed to read any resources in the AWS and DigitalOcean services.

2. You are NOT allowed to add, delete, or modify any resources in the AWS or DigitalOcean services.
3. Do NOT perform automated testing or run automated scripts on the provided resources, your IP address will be banned.

Pertanyaan / Jawaban

1. What is the Access Key ID of the leaked AWS credentials?
   Kita bisa melihat Access Key pada frontend, karena di hardcode.



AKIA453N4QOMCFGUPBPV

2. What is the invoke URL of the other API Gateway that exists?

   https://bp61amux7k.execute-api.ap-northeast-1.amazonaws.com/list

   Untuk mendapatkan endpoint lain, kita bisa setup aws cli kita menggunakan creds yang didapatkan terlebih dahulu lalu dump seperti di screenshot.

   ➜ ある日私が雲の中にいて攻撃されたとき git:(main) ✗ aws apigateway get-rest-apis

3. How many Lambda functions were deployed in the AWS account?

   4

   Untuk mendapatkan banyak lambda function, kita bisa hit endpoint https://bp61amux7k.execute-api.ap-northeast-1.amazonaws.com/list menggunakan script berikut.

sv.py

```
import boto3
import requests
from requests.auth import AuthBase
from botocore.auth import SigV4Auth
```

```
from botocore.awsrequest import AWSRequest
from botocore.credentials import Credentials

def aws_sigv4_auth(req, service='execute-api', region='ap-northeast-1'):
    credentials = Credentials('AKIA453N4QOMCFGUPBPV',
'/Sl40g8B20vyaL152shn1XGfqqrmQd6R/mu1SqSy')
    aws_request = AWSRequest(method=req.method, url=req.url,
headers=req.headers)
    SigV4Auth(credentials, service, region).add_auth(aws_request)
    return aws_request.headers

url = "https://bp61amux7k.execute-api.ap-northeast-1.amazonaws.com/list"
headers = {}
signed_headers = aws_sigv4_auth(requests.Request('GET', url,
headers=headers))

response = requests.get(url, headers=signed_headers)
print(response.text)
```

```
[*] closed connection to 103.145.226.92 port 25456
➜ ~ python3 -u "/mnt/d/ctf-writeups/NCW CTF 2024/finals/ある日私が雲の中にいて攻撃されたとき/sv.py"
{"statusCode": 200, "body": "[\"log-transactions\", \"manage-user\", \"list-functions\", \"create-backup\"]"}
➜ ~
```

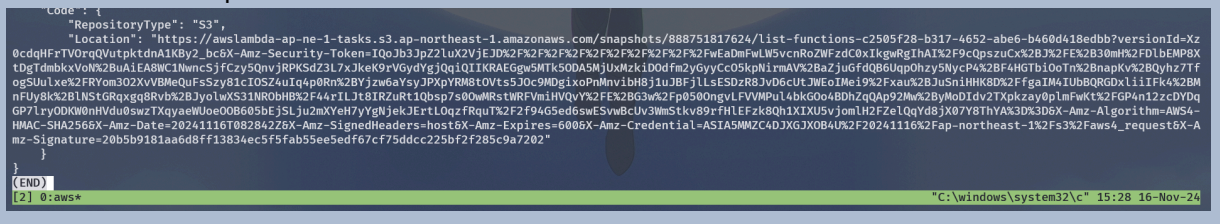
4. What is the full URL of the compromised S3 bucket?

   https://wijenbank-tokyo.s3.ap-northeast-1.amazonaws.com/

   Buat cari tau nama S3 bucket nya, saya coba dump list-functions itu ngapain.

   aws lambda get-function --function-name create_backup.

   Nah nanti kita dapet suatu link ke bucket S3 lain.



   Pas di download dan di extract kita dapet nama bucket 'wijenbank-tokyo'. Kita tinggal formatin sesuai url S3

5. What is the valid AWS Access Key ID that was exposed in the S3 bucket?

   AKIA453N4QOMEIL32YW4

Kita bisa download isi dari S3 menggunakan command.

aws s3 sync s3://wijenbank-tokyo new --profile s3-ctf

Nah disini saya cobain semua creds yang ada :u

```
→ 2024-Q4 git:(main) X cd config-backup
→ config-backup git:(main) X ls
aws-cli.json  aws-s3-policy.json
→ config-backup git:(main) X cat aws-cli.json
{
  "username": "dev",
  "aws_access_key": "AKIA453N4QOMEIL32YW4",
  "aws_secret_key": "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",
  "account_type": "developer",
  "created_at": "2023-12-01 00:00:00"
}%
→ config-backup git:(main) X pwd
```

Dan benernya yang ini.

6. What is the statement ID that allowed public access to the directory with the valid AWS Access Key ID?

PublicReadGetObject

Nah kita bisa liat di file aws-s3-policy.json, lalu cek, ada nama PublicReadGetObject

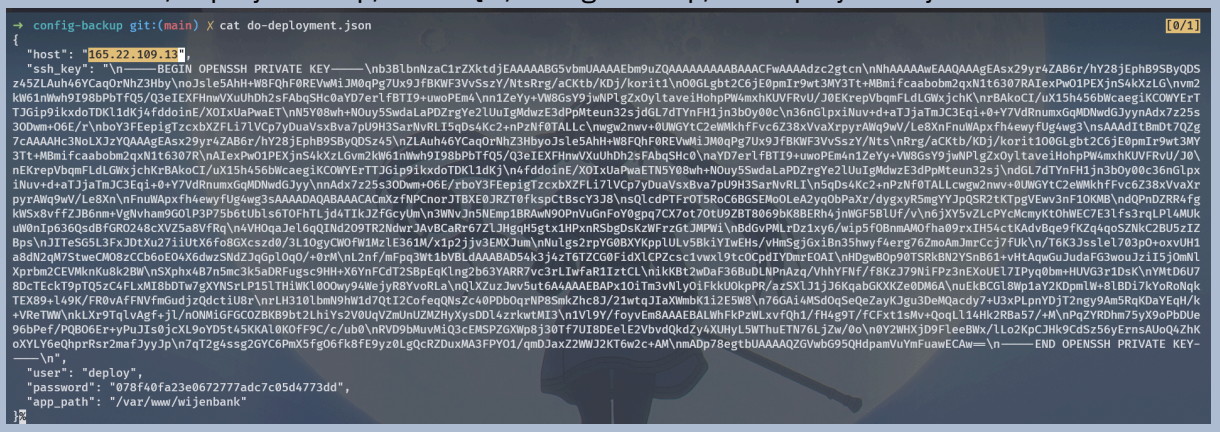
```
→ config-backup git:(main) X cat aws-s3-policy.json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "PublicReadGetObject",
      "Effect": "Allow",
      "Principal": "*",
      "Action": "s3:GetObject",
      "Resource": [
        "arn:aws:s3:::wijenbank-tokyo",
        "arn:aws:s3:::wijenbank-tokyo/*"
      ],
      "Condition": {
        "StringNotLike": {
          "aws:UserAgent": [
            "*Bot*",
            "*bot*",
            "*Crawler*",
            "*crawler*"
          ]
        }
      }
    },
    {
```

7. What is the IP address of the DigitalOcean production server?

165.22.109.13

Nah buat dapetin IP nya, kita bisa ganti creds pakek creds yang developer, yang nomor 5, trus redump ulang, karena ada beberapa file yang kena permission denied.

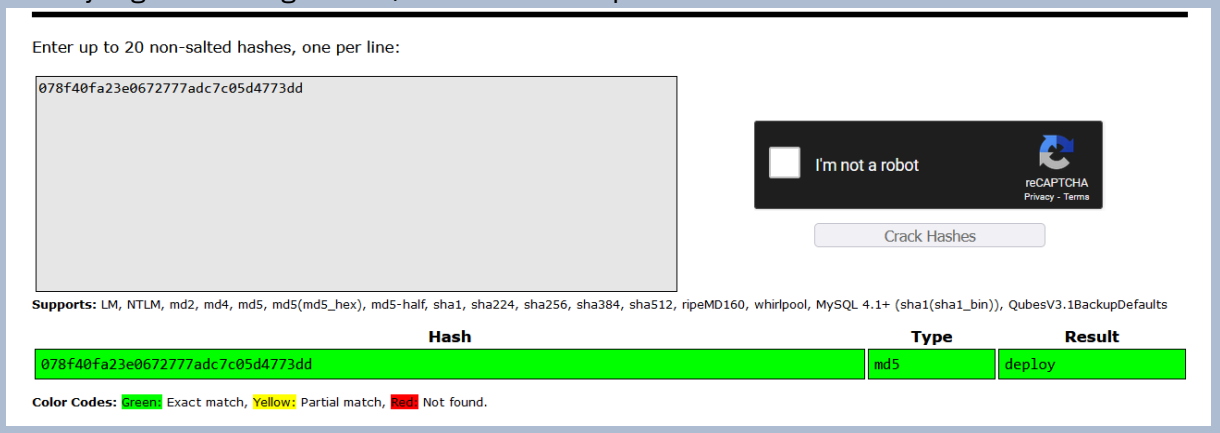
Nanti ada di /deploy-backup/2024-Q4/config-backup/do-deployment.json.



8. What is the password of the account "deploy"?

deploy

Di file yang sama dengan no 7, kita coba crack password di crackstation.



9. What is the MD5 hash of the file left by the attacker inside the app deployment directory on the DigitalOcean production server?

```
deploy@ubuntu-s-1vcpu-512mb-10gb-sgp1-01:~$ cd /var/www/wijenbank/
deploy@ubuntu-s-1vcpu-512mb-10gb-sgp1-01:/var/www/wijenbank$ ls
hello.txt
deploy@ubuntu-s-1vcpu-512mb-10gb-sgp1-01:/var/www/wijenbank$ md5sum
^C
deploy@ubuntu-s-1vcpu-512mb-10gb-sgp1-01:/var/www/wijenbank$ md5sum hello.txt
c8c7209c7a87f99a1d11a53854808f1f  hello.txt
```

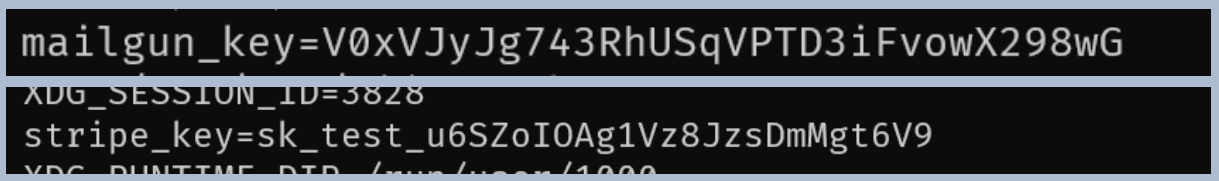
c8c7209c7a87f99a1d11a53854808f1f - hello.txt

Nah di file yang no 7 kan kita dapet private key. Kita bisa pakek itu buat login ke

DigitalOcean nya. File nya ada di /var/www/wijenbank.

10. How many non-default environment keys were set on the DigitalOcean production server?

Dengan menggunakan command env, akan didapatkan 2 buah foreign env yang terdapat dalam server.



mailgun_key=V0xVJyJg743RhUSqVPTD3iFvowX298wG
stripe_key=sk_test_u6SZoIOAg1Vz8JzsDmMgt6V9

2

Solusi

solver.py

```
from pwn import *

io = remote("103.145.226.92", 23456)

answers = ["AKIA453N4QOMCFGUPBPV",
"https://bp61amux7k.execute-api.ap-northeast-1.amazonaws.com/list", "4",
"https://wijenbank-tokyo.s3.ap-northeast-1.amazonaws.com/",
"AKIA453N4QOMEIL32YW4", "PublicReadGetObject", "165.22.109.13", "deploy",
"c8c7209c7a87f99a1d11a53854808f1f"]

for answer in answers:
    io.sendlineafter(b'Your answer: ', answer)

io.interactive()
```

Hasil

```
WSL at  kali 2024-11-16 15:13:23  ~
→ python3 exploit.py
[+] Opening connection to 103.145.226.92 on port 23456: Done
/home/kali/exploit.py:8: BytesWarning: Text is not bytes; assuming ASCII, no guarantees. See https://docs.pwntools.com/#byte
s
  io.sendlineafter(b'Your answer: ', answer)
[*] Switching to interactive mode
Correct

Congratulations! You have successfully completed the challenge.
NCW{ingin_menjadi_cloud_engineer_handal}
[*] Got EOF while reading in interactive
$
```