Proof Of Concept
0Byte CTF 2023

NAMA Peserta : Fikri Muhammad Abdillah (FlaB)

Minggu, 19 Agustus 2023

Cryptography

Token

Executive Summary (Penjelasan singkat soal)

Kita diberikan TCP connection, attachment file (chall.py) yang merupakan source code dari TCP tersebut. Kita disuruh untuk connect ke TCP lalu memberikan token yang telah di encrypt menggunakan RSA dengan tepat sebanyak 100 kali, lalu kita akan di berikan flag nya.

Technical Report (Penjelasan detail beserta screenshot step-by-step)

```
def get_prime(n):
    r = getRandomInteger(n)
    p = nextprime(r)
    q = nextprime(r + getRandomInteger(32))
    return p, q
```

Dapat dilihat bahwa n (modulus) adalah hasil dari p dan q yang merupakan prime number, dapat dilihat bahwa p dan q adalah angka yang dekat, jika seperti itu maka kita bisa factor dengan menggunakan fermat.

```python
from Crypto.Util.number import *
from math import isqrt
from gmpy2 import iroot
from pwn import *

# nc 0x7e7ctf.zerobyte.me 10021
HOST = "0x7e7ctf.zerobyte.me"
PORT = 10021

def is_square(x):
    """
    Returns the square root of x if x is a perfect square, or None otherwise.
    :param x: x
    :return: the square root of x or None
    """
    y = isqrt(x)
    return y if y ** 2 == x else None
```

```python
def factorize(N):
    """
    Recovers the prime factors from a modulus using Fermat's factorization method.
    :param N: the modulus
    :return: a tuple containing the prime factors, or None if the factors were not found
    """
    a = isqrt(N)
    b = a * a - N
    while b < 0 or not is_square(b):
        a += 1
        b = a * a - N

    p = a - isqrt(b)
    q = N // p
    if p * q == N:
        return p, q

io = remote(HOST, PORT)

for _ in range(100):
    n = int(io.recvline().decode().strip().split(" = ")[1])
    e = int(io.recvline().decode().strip().split(" = ")[1])
    c = int(io.recvline().decode().strip().split(" = ")[1])

    p, q = factorize(n)
    phi = (p - 1) * (q - 1)
    d = pow(e, -1, phi)
    m = pow(c, d, n)

    token = long_to_bytes(m)

    io.sendlineafter(b"[TOKEN]>", token)
    io.recvline()
io.interactive()
```

(src: https://github.com/jvdsn/crypto-attacks/blob/master/attacks/factorization/fermat.py)

dengan code python di atas, saya dapat memberikan token dengan tepat setelah men-decrypt nya dengan private key (d) yang saya dapatkan setelah memfactor kan n. lalu saya mendapat flag nya
FLAG: 0byteCTF{emang_boleh_sedekat_ini_dek?}

Conclusion (Kesimpulan dari soal)

ini adalah soal tentang factorisasi menggunakan fermat yang memanfaatkan kelemahan pengambilan prime number yang terlalu dekat.

Enkripsi Jadoel

Executive Summary (Penjelasan singkat soal)

Kita diberikan TCP connection, attachment file (app.py) yang merupakan source code dari TCP tersebut. Didalam TCP tersebut, kita dapat encrypt message (hex format) dengan encryption AES (ECB) dan akan keluar hasil nya, lalu kita juga dapat decrypt message yang kita masukkan tadi, lalu akan mendapatkan plaintext tadi. Tugas utamanya adalah untuk mengambil flag yang sebelumnya telah di sisipkan pada plaintext yang kita berikan, dan mendapatkan hasil encrypt yang berisi “plaintext + salt + flag”, masalahnya adalah function decrypt akan menghapuskan flag sebelum memunculkan hasil decrypt kepada user.

Technical Report (Penjelasan detail beserta screenshot step-by-step)

Sekilas dapat kita lihat bahwa encryption menggunakan AES (ECB) yang memang tidak terlalu aman, karena ECB sendiri hanya encrypt setiap block sendiri2, karena itu setiap block tidak ada kesinambungan satu sama lain, yang membuat nya dapat leak plaintext. awalnya saya mengira menggunakan plaintext_recovery_attack

(src: https://github.com/jvdsn/crypto-attacks/blob/master/attacks/ecb/plaintext_recovery.py)

ternyata saya salah, saya lupa kalau ada function untuk decrypt, karena itu saya menggunakan metode yang akan swap block cipher yang tepat, agar dapat memunculkan flag.

pertama tama saya akan mencari panjang flag

```python
from pwn import *

# nc 0x7e7ctf.zerobyte.me 10027
HOST = "0x7e7ctf.zerobyte.me"
PORT = 10027

SALT_SIZE = 8

def enc(io, plaintext: bytes):
    io.sendlineafter(b"Masukkan pilihan: ", b"1")
    io.sendlineafter(b"Masukkan pesan: ", plaintext.hex().encode("latin-1"))
    return bytes.fromhex(io.recvline().decode().strip().split(": ")[-1])

def dec(io, ciphertext: bytes):
    io.sendlineafter(b"Masukkan pilihan: ", b"2")
    io.sendlineafter(b"Masukkan pesan terenkripsi: ", ciphertext.hex().encode("latin-1"))
    return bytes.fromhex(io.recvline().decode().strip().split(": ")[-1])

def get_flag_size(io, oracle):
    last_len = len(oracle(io, b""))
    flag_size = last_len // 2
    padding = b"A"
    while True:
        flag_size -= 1
        t_length = len(oracle(io, padding))
        if t_length != last_len:
            print(padding)
            break
        last_len = t_length
        padding += b"A"
    return flag_size

io = remote(HOST, PORT)

# print(attack(io, enc, dec, 0))
print(get_flag_size(io, enc))
```

setelah saya mendapatkan panjang flag, saya akan generate padding agar posisi flag sesuai, lalu encrypt padding dan mengambil hasil encrpyt tersebut, lalu swap setiap block cipher satu per satu

```python
c = "be8519cf449c7437db24bd7e0b4a03359a21ca657ed910a545eb1272da3596f09792ba2087c21e17dd2de1aa22acb99a67de643ba5bd7d06d88fe435143a4055a047

length_flag = 26
length_salt = 8

number_block = 5
total_padding = (16 * 4) - (length_flag + length_salt) + 1

print(total_padding)
print((b"A" * total_padding).hex())
print(len(c) // 32)
offset_block = 5

p = c[len(c) - (32 * offset_block):len(c) - (32 * (offset_block - 1))] + c[:len(c) - (32 * offset_block)] + c[len(c) - (32 * (offset_block
q = c[len(c) - 64:len(c) - 32] + c[:len(c) - 64] + c[len(c) - 32:]
print(p)
```

```
Meskipun sistem enkripsi jadoel, tapi harusnya masih aman!
[1] Enkrip pesan
[2] Dekrip Pesan
[3] Keluar
Masukkan pilihan: 2
Masukkan pesan terenkripsi: 2248c016d3afbeef1197b233118cdcd8be8519cf449c7437db24bd7e0b4a03359a21ca657ed910a545eb1272da3596f09792ba2087c21e17dd2de1aa22acb99a67de643ba5b
d7d06d88fe435143a4055a04769f0b422291f4046ed4ef97888f69830c8cb6508bda26afb79f693e7cf43
Hasil dekripsi: 4d03l_347fd48d64AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
----- Layanan enkripsi pesan -----
Meskipun sistem enkripsi jadoel, tapi harusnya masih aman!
[1] Enkrip pesan
[2] Dekrip Pesan
[3] Keluar
Masukkan pilihan: 2
Masukkan pesan terenkripsi: a04769f0b422291f4046ed4ef97888f6be8519cf449c7437db24bd7e0b4a03359a21ca657ed910a545eb1272da3596f09792ba2087c21e17dd2de1aa22acb99a67de643ba5b
d7d06d88fe435143a40552248c016d3afbeef1197b233118cdcd89830c8cb6508bda26afb79f693e7cf43
Hasil dekripsi: _l4g1_3nkr1ps1_jAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
----- Layanan enkripsi pesan -----
Meskipun sistem enkripsi jadoel, tapi harusnya masih aman!
[1] Enkrip pesan
[2] Dekrip Pesan
[3] Keluar
Masukkan pilihan: 2
Masukkan pesan terenkripsi: 67de643ba5bd7d06d88fe435143a4055be8519cf449c7437db24bd7e0b4a03359a21ca657ed910a545eb1272da3596f09792ba2087c21e17dd2de1aa22acb99aa04769f0b42
2291f4046ed4ef97888f62248c016d3afbeef1197b233118cdcd89830c8cb6508bda26afb79f693e7cf43
Hasil dekripsi: y4_m4u_b4g4im4n4AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
----- Layanan enkripsi pesan -----
Meskipun sistem enkripsi jadoel, tapi harusnya masih aman!
[1] Enkrip pesan
[2] Dekrip Pesan
[3] Keluar
Masukkan pilihan: 2
```

```
----- Layanan enkripsi pesan -----
Meskipun sistem enkripsi jadoel, tapi harusnya masih aman!
[1] Enkrip pesan
[2] Dekrip Pesan
[3] Keluar
Masukkan pilihan: 1
Masukkan pesan: 41414141414141414141414141414141414141414141414141414141414141
Hasil enkripsi: be8519cf449c7437db24bd7e0b4a03359a21ca657ed910a545eb1272da3596f09792ba2087c21e17dd2de1aa22acb99a67de643ba5bd7d06d88fe435143a4055a04769f0b422291f4046ed4
ef97888f62248c016d3afbeef1197b233118cdcd89830c8cb6508bda26afb79f693e7cf43
```

```
Masukkan pilihan: 2
Masukkan pesan terenkripsi: 2b0c116fe391550f9ce2b649022ab5f8710f71ecc0487b359396f938b36f92cf710f71ecc0487b359396f938b36f92cf64c7f72ee87166d977819bf094a7a4e43ea388f4a2e
976a40738ae53d0928ae00923c720de5186a9fd56a48367a6f27050502fb7e6b3b37c5c08ba855a4790fa
Hasil dekripsi: byteCTF{y4_m4u_bAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
----- Layanan enkripsi pesan -----
Meskipun sistem enkripsi jadoel, tapi harusnya masih aman!
[1] Enkrip pesan
[2] Dekrip Pesan
[3] Keluar
Masukkan pilihan: ^C
```

FLAG: 0ByteCTF{y4_m4u_b4g4im4n4_l4g1_3nkr1ps1_j4d03l_347fd48d64}

Conclusion (Kesimpulan dari soal)

soal ini adalah tentang kerentanan metode block cipher yang lawas (jadul), karena metode block cipher tersebut sudah di temukan kerentanannya.a

Reverse Engineering

0Pyte

Executive Summary (Penjelasan singkat soal)

kita diberikan attachment yang berisi source code (python) dan encrypted file (flag.png.enc). dalam source code python tersebut, susunan code nya sangat kacau, code yang harusnya bisa

beberapa baris, jadi hanya satu baris. kita di suruh reverse function encrypt nya agar dapat decrypt encrypted file nya.

Technical Report (Penjelasan detail beserta screenshot step-by-step)

karena ini hanya code python yang menjadi 1 baris saja bukan seperti obfuscated javascript, saya hanya melakukannya satu persatu sampai code python dapat di pahami dengan lebih mudah.

looping yang ada dalam list (cth: `[i for i in range(100)]`) saya pisah menjadi beberapa baris agar lebih mudah dipahami.

ini hasil nya:

```python
def encrypt(this_encoder_file_name: str, file_name: str) -> bytes:
    seed('0Byte')
    def xor_0x69(inp: list) -> bytes:
        res = bytearray()
        for i in inp:
            for j in bytes.fromhex(i):
                res.append(j ^ 0x69)
        return bytes(res)

    iv = bytes.fromhex(md5(file_name.encode()).hexdigest())
    key = bytes.fromhex(md5(this_encoder_file_name.encode()).hexdigest())
    aes = AES.new(key, AES.MODE_CBC, iv)

    with open(file_name, 'rb') as FILE:
        padded_file = pad(FILE.read(), 16)
        enc = aes.encrypt(padded_file).hex().encode()

    x = []
    for i, e in enumerate(enc):
        if i % 2 == 0:
            x.append(e ^ randint(1, 255))
        else:
            x.append(e ^ ord(choice(ascii_letters)))

    y = []
    for i, j in enumerate(x):
        if i % 2 == 1:
            y.append(md5(chr(j).encode()).hexdigest())
        else:
            y.append('{:02X}'.format(j).lower())


    enc2 = xor_0x69(y)
    enc2 = b64encode(str(bytes_to_long(enc2)).encode())

    res = ""
    for i, z in enumerate(enc2):
        if i % 2 == 0:
            res += chr(z ^ 0x01)
        else:
            res += chr(z ^ 0x02)

    return res.encode()
```

saat membalikkannya (reverse) agar menjadi function decrypt, saya tidak ada masalah sampai di function `xor_0x69`, karena sebelum melewati function itu, input merupaka list yang isinya berbeda

panjang nya, menjadi 1 baris byte string yang membuat saya sedikit kebingungan, tapi untungnya list tersebut hanya memiliki 2 panjang yang berbeda, karena isi list yang lebih panjang adalah hasi dari md5, kerena itu saya dapat melanjutkannya

```python
def dexor_0x69(inp: bytes) -> list:
    res = []
    for i, j in enumerate(inp):
        temp = '{:02x}'.format(j ^ 0x69)
        if i % 17 in [0, 1]:
            res.append(temp)
        else:
            res[(i // 17) + ((i // 17) + 1)] += temp
    return res
```

setelah itu, setiap byte yang melewati md5 saya bruteforce

```python
def brute_md5(hashes):
    for i in range(256):
        if md5(chr(i).encode()).hexdigest() == hashes:
            return i
```

setelah itu cipher akan melewati decrypt dengan AES, tapi tidak perlu khawatir, karena iv didapat dari md5 hash nama file yang akan di encrypt, dan key adalah nama file encryptor python ini (0pyteware.py).

Final:

```python
def decrypt(this_encoder_file_name: str, file_name: str) -> str:
    seed('0Byte')
    def dexor_0x69(inp: bytes) -> list:
        res = []
        for i, j in enumerate(inp):
            temp = '{:02x}'.format(j ^ 0x69)
            if i % 17 in [0, 1]:
                res.append(temp)
            else:
                res[(i // 17) + ((i // 17) + 1)] += temp
        return res

    with open(f"{file_name}.enc", 'rb') as FILE:
        cipher = FILE.read()

    enc2 = bytearray()
    for i, z in enumerate(cipher):
        if i % 2 == 0:
            enc2.append(z ^ 0x01)
        else:
            enc2.append(z ^ 0x02)

    enc2 = bytes(enc2)
    enc2 = long_to_bytes(int(b64decode(enc2).decode()))

    y = dexor_0x69(enc2)
    x = []
    for i, j in enumerate(y):
        if i % 2 == 1:
            x.append(brute_md5(j))
        else:
            x.append(int(j, 16))

    enc = ""
    for i, j in enumerate(x):
        if i % 2 == 0:
            enc += chr(j ^ randint(1, 255))
        else:
            enc += chr(j ^ ord(choice(ascii_letters)))
```

```python
    iv = bytes.fromhex(md5(file_name.encode()).hexdigest())
    key = bytes.fromhex(md5(this_encoder_file_name.encode()).hexdigest())
    aes = AES.new(key, AES.MODE_CBC, iv)

    res = aes.decrypt(bytes.fromhex(enc))
    res = unpad(res, 16)

    return res

# argv_1 = "flag.png"
argv_1 = "flag.png"
# write_to_file(argv_1, encrypt(argv[0], argv_1))
# encrypt("test", argv_1)
# print(argv[0])
argv_0 = "0pyteware.py"
res = decrypt(argv_0, argv_1)
print(res)

with open(argv_1, 'wb') as FILE:
    FILE.write(res)
```

saya dapat sedikit masalh setelah decrypt file, karena file yang terdecrypt tadi corrupt, tapi saya menggunakan tool online (url: https://compress-or-die.com/repair)

0ByteCTF{satu_baris_mah_ez_dekkkk}

FLAG: 0ByteCTF{satu_baris_mah_ez_dekkkk}

Conclusion (Kesimpulan dari soal)

itu adalah tentang deobfuscating python code sederhana.

Web Exploitation

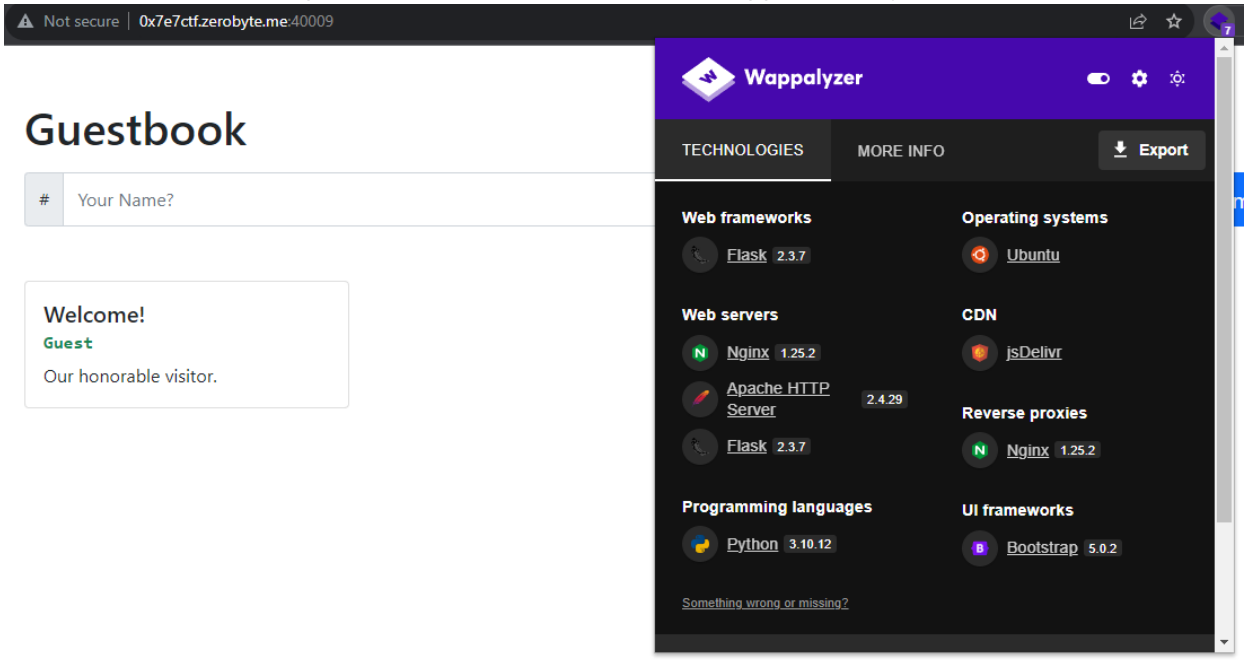
Guestbook (Beta)

Executive Summary (Penjelasan singkat soal)

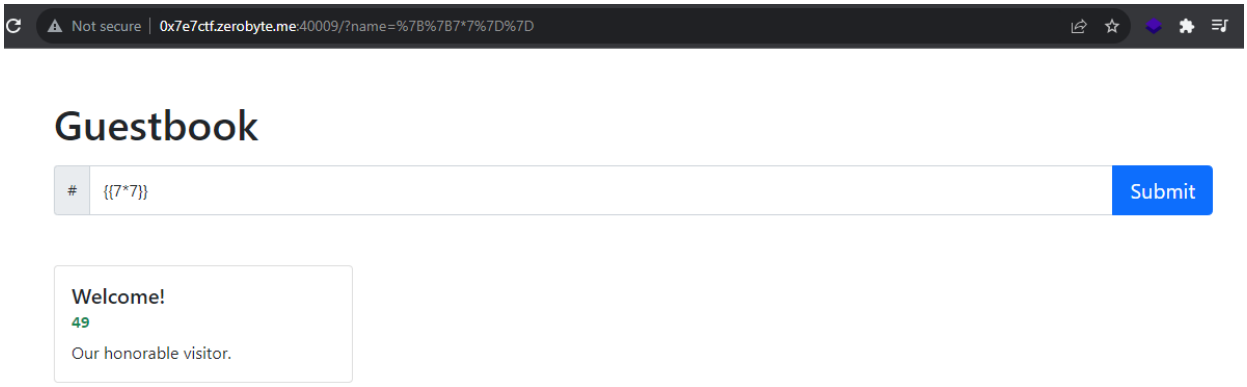
Kita diberikan link website, dan disuruh untuk mencari flag di sana.

Technical Report (Penjelasan detail beserta screenshot step-by-step)

dapat dilihat dengan wappalizer, bahwa web ini menggunakan python

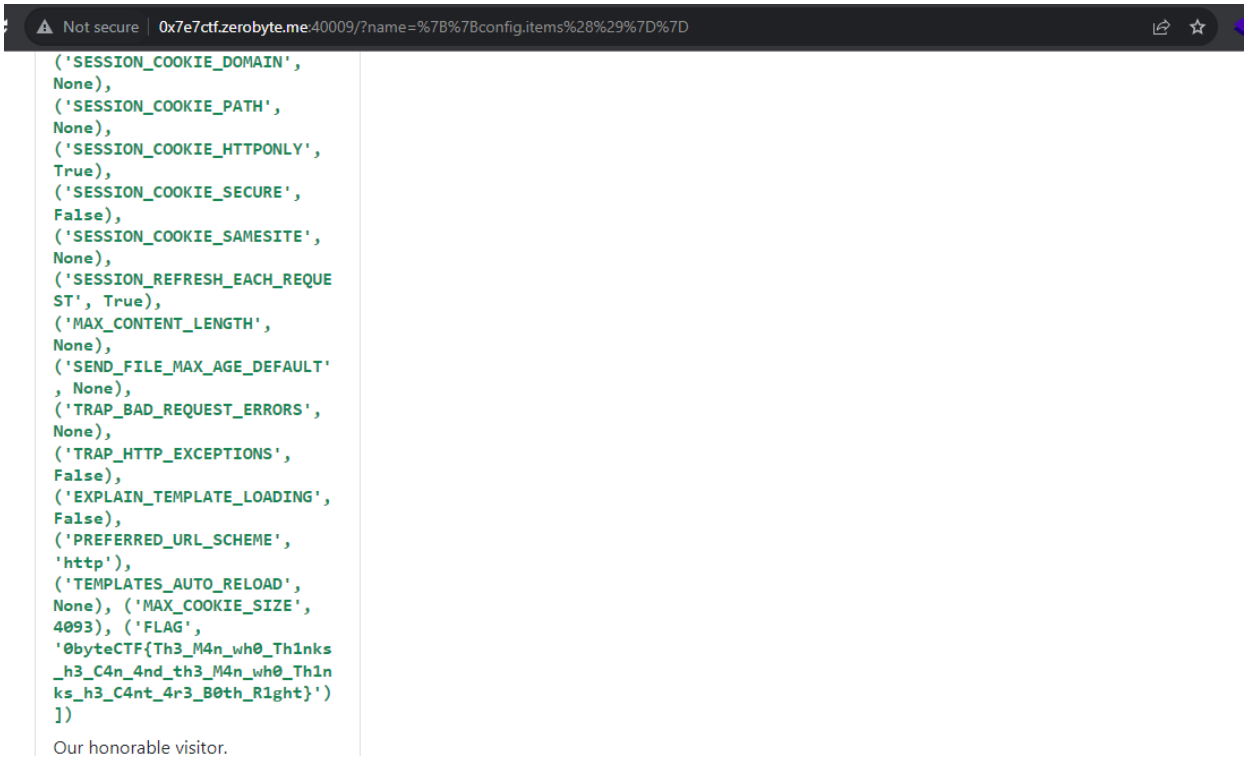


karena web ini menggunakan python, saya coba akan menggunakan SSTI exploitation



dan ternyata bisa, selanjutnya tinggal cari function untuk rce atau revshell. saat saya melihat2 config, ternyata flag sudah tertera di sana.

Payload: {{config.item()}}



FLAG: 0byteCTF{Th3_M4n_wh0_Th1nks_h3_C4n_4nd_th3_M4n_wh0_Th1nks_h3_C4nt_4r3_B0th_R1ght}

Conclusion (Kesimpulan dari soal)

Soal ini tentang exploitasi python base web dengan jinja sebagai renderer_template yang dapat di exploitasi dengan STTI method untuk mendapatkan flag nya.

Digital Forensic

Who The Hack

Executive Summary (Penjelasan singkat soal)

Challange ini menceritakan tentang perusahaan yang terkena hack, dan kita sebagai Tim It security di suruh untuk mencari pelakunya. kita mendapatkan access.log yang merupakan log dari apache2.

Technical Report (Penjelasan detail beserta screenshot step-by-step)

Karena dari ceritanya kita di suruh mencari pelakunya, saya menganalisis access.log dengan berfokus pada ip mana yang paling mencurigakan. lalu saya menggunaka code python ini untuk menganalisisnya

```python
from collections import defaultdict

log_file_path = 'access.log'

suspicious_ip = defaultdict(list)

with open(log_file_path, 'r') as log_file:
    for line in log_file:
        fields = line.split()

        if fields[8] == '403':
            suspicious_ip[fields[0]].append(f"Access to forbidden file: {fields[6]}")
        if fields[8] == '401':
            suspicious_ip[fields[0]].append(f"Failed login attempt: {fields[6]}")

print()
for ip, activities in suspicious_ip.items():
    print(f"Suspicious IP: {ip}")
    for activity in activities:
        print(f"  - {activity}")
```

dan mendapatkan satu hasil ip yang mencurigakan

```
[Running] python -u "d:\Programming\Cyber Security\CTF\2023\0byte-ctf\Digital Forensic\Who The Hack\solve.py"

Suspicious IP: 178.19.45.123
  - Access to forbidden file: /WebReport/ReportServer
  - Access to forbidden file: /.env
  - Access to forbidden file: /../../../etc/passwd
  - Access to forbidden file: /Vagrantfile
  - Access to forbidden file: /.git/config
  - Access to forbidden file: /../WEB-INF/web.xml
  - Access to forbidden file: /_phpmyadmin/index.php
  - Access to forbidden file: /.htaccess
  - Access to forbidden file: /.DS_Store
  - Access to forbidden file: /.github/workflows/build.yaml
  - Access to forbidden file: /.aws/config.yaml
  - Access to forbidden file: /info.php
  - Access to forbidden file: /phpinfo.php
  - Access to forbidden file: /.htaccess
  - Access to forbidden file: /download.php?file=../../../../../etc/passwd
  - Access to forbidden file: /.ssh/id_rsa
  - Access to forbidden file: /.travis.yml
  - Access to forbidden file: /.travis.yml.swp
  - Access to forbidden file: /.travis.yml~
  - Access to forbidden file: /remote/fgt_lang?lang=/../../../../../dev/cmdb/sslvpn_websession
  - Access to forbidden file: /.config.php.swp
  - Access to forbidden file: /../etc/passwd
  - Access to forbidden file: /../../etc/passwd
  - Access to forbidden file: /../../../etc/passwd
  - Access to forbidden file: /../../../../etc/passwd
  - Access to forbidden file: /../../../../../etc/passwd
  - Access to forbidden file: /posts/1%27+OR+SLEEP%28999%29+--+-+
  - Access to forbidden file: /%60whoami%60
  - Access to forbidden file: /%60id%60
  - Access to forbidden file: /%60pwd%60
  - Access to forbidden file: /%60ls%60
  - Access to forbidden file: /%60uname%60
  - Access to forbidden file: /posts/1%27+OR+1=1+--+-+
  - Access to forbidden file: /posts/1%27+ORDER+BY+1+--+-+
  - Access to forbidden file: /posts/1%27+ORDER+BY+2+--+-+
  - Access to forbidden file: /posts/1%27+ORDER+BY+3+--+-+
  - Access to forbidden file: /posts/1%27+ORDER+BY+4+--+-+
  - Access to forbidden file: /posts/1%27+ORDER+BY+5+--+-+
  - Access to forbidden file: /posts/1%27+ORDER+BY+6+--+-+
```

selanjutnya saya mengambil semua request url dari ip tersebut

```python
# --- get all requests urls with spesific ip ---
ip = "178.19.45.123"
url = []
with open(log_file_path, 'r') as log_file:
    for line in log_file:
        fields = line.split()

        if ip in fields[0]:
            url.append(fields[6])

for i in url:
    print(i)
```

lalu saya mendapatkan request yang aneh, yang kemungkinannya adalah flag

```
/posts/1%27+ORDER+BY+19+--+-+
/posts/1%27+UNION+SELECT+ALL+1,2,3+--+-+
/posts/1%27+UNION+SELECT+ALL+1,2,3,4+--+-+
/posts/1%27+UNION+SELECT+ALL+1,2,3,4,5--+-+
/posts/1%27+UNION+SELECT+ALL+1,2,3,4,5,6+--+-+
/posts/1%27+UNION+SELECT+ALL+1,2,3,4,5,6,7+--+-+
/posts/1%27+UNION+SELECT+ALL+1,2,3,4,5,6,7,8+--+-+
/posts/1%27+UNION+SELECT+ALL+1,2,3,4,5,6,7,8,9+--+-+
/posts/1%27+UNION+SELECT+ALL+1,2,3,4,5,6,7,8,9,10+--+-+
/%30
/%62
/%79
/%74
/%65
/%43
/%54
/%46
/%7B
/%57
/%33
/%5F
/%53
/%75
/%66
/%66
/%33
/%72
/%5F
/%4D
/%30
/%72
/%33
/%5F
/%30
/%66
/%74
/%33
/%6E
/%5F
```

lalu saya convert ke ascii

```python
# --- get flag ---
ip = "178.19.45.123"
url = []
with open(log_file_path, 'r') as log_file:
    for line in log_file:
        fields = line.split()

        if ip in fields[0]:
            url.append(fields[6])

for i in url:
    if "/%" in i and len(i.strip()) <= 4:
        print(bytes.fromhex(i.strip()[2:]).decode(), end="")
```

```
PROBLEMS 1  OUTPUT  DEBUG CONSOLE  TERMINAL
[Running] python -u "d:\Programming\Cyber Security\CTF\2023\0byte-ctf\Digital Forensic\Who The Hack\solve.py"
0byteCTF{W3_Suff3r_M0r3_0ft3n_1n_1m4g1n4t10n_Th4n_1n_R34l1ty}
[Done] exited with code=0 in 0.801 seconds
```

FLAG: 0byteCTF{W3_Suff3r_M0r3_0ft3n_1n_1m4g1n4t10n_Th4n_1n_R34l1ty}

Conclusion (Kesimpulan dari soal)

ini adalah soal yang berhubungan dengan analysis log apache2..

Romeo and Dulliet

Executive Summary (Penjelasan singkat soal)

Challange ini mirip dengan sebelumnya, tapi dengan perbedaan cerita dan kita di berikan 2 file

Technical Report (Penjelasan detail beserta screenshot step-by-step)

saat saya mau menganalisis seperti sebelumnya, saya malah menemukan error yang tidak biasa

```
Traceback (most recent call last):
  File "d:\Programming\Cyber Security\CTF\2023\0byte-ctf\Digital Forensic\Romeo and Dulliet\solve.py", line 13, in <module>
    if fields1[8] == '403':
       ~~~~~~~^^^
IndexError: list index out of range
```

saat saya debug, saya menemukan ini

```
['{']
Traceback (most recent call last):
  File "d:\Programming\Cyber Security\CTF\2023\0byte-ctf\Digital Forensic\Romeo and Dulliet\solve.py", line 14, in <module>
    if fields1[8] == '403':
       ~~~~~~~~^^^
IndexError: list index out of range
```

```
27.28.161.141 - - [01/
{
229.239.15.87 - - [01/
```

saat saya memisahkan itu dan print hal tersebut, membentuk sesuatu yang menarik

```
[Running] python -u "d:\Programming\Cyber Security\CTF\2023\0byte-ctf\Digital Forensic\Romeo and Dulliet\solve.py"
{
3
4
n
A
_
m
A
_
1
4
1
3
}

Suspicious IP: 178.19.45.123
  - Access to forbidden file: /download.php?file=../../../../../etc/passwd"
  - Access to forbidden file: /download.php?file=../../../../../etc/passwd"
  - Failed login attempt: /posts/1%27+UNION+SELECT+ALL+1,2,3+--+-+"
  - Failed login attempt: /posts/1%27+UNION+SELECT+ALL+1,2,3+--+-+"
  - Access to forbidden file: /.dbeaver/data-sources.json"
  - Access to forbidden file: /.dbeaver/data-sources.json"
  - Access to forbidden file: /posts/1%27+OR+1=1+--+-+"
  - Access to forbidden file: /posts/1%27+OR+1=1+--+-+"
```

```
{34nA_mA_1413}
```

lalu saya terpikir untuk menggabungkan kedua file log tersebut yang sebelumnya saya hanya analysis 1 log dan terbentuk lah flag

```python
from collections import defaultdict

log_file_path1 = 'Romeo.txt'
log_file_path2 = 'Dulliet.txt'

suspicious_ip = defaultdict(list)

with open(log_file_path1, 'r') as log_file1, open(log_file_path2, 'r') as log_file2:
    for line1, line2 in zip(log_file1, log_file2):
        fields1 = line1.split()
        fields2 = line2.split()

        if len(fields1) <= 1:
            print(line1.rstrip(), end='')
        else:
            if fields1[8] == '403':
                suspicious_ip[fields1[0]].append(f"Access to forbidden file: {fields1[6]}")
            if fields1[8] == '401':
                suspicious_ip[fields1[0]].append(f"Failed login attempt: {fields1[6]}")

        if len(fields2) <= 1:
            print(line2.rstrip(), end='')
            pass
        else:
            if fields2[8] == '403':
                suspicious_ip[fields2[0]].append(f"Access to forbidden file: {fields2[6]}")
            if fields2[8] == '401':
                suspicious_ip[fields2[0]].append(f"Failed login attempt: {fields2[6]}")

print()
for ip, activities in suspicious_ip.items():
    print(f"Suspicious IP: {ip}")
    for activity in activities:
        print(f"  - {activity}")
```

```
[Running] python -u "d:\Programming\Cyber Security\CTF\2023\0byte-ctf\Digital Forensic\Romeo and Dulliet\solve.py"
{s3M4n9At_3mPAt_l1m4_13377}
Suspicious IP: 178.19.45.123
  - Access to forbidden file: /download.php?file=../../../../../etc/passwd"
  - Access to forbidden file: /download.php?file=../../../../../etc/passwd"
  - Failed login attempt: /posts/1%27+UNION+SELECT+ALL+1,2,3+--+-+"
  - Failed login attempt: /posts/1%27+UNION+SELECT+ALL+1,2,3+--+-+"
  - Access to forbidden file: /.dbeaver/data-sources.json"
  - Access to forbidden file: /.dbeaver/data-sources.json"
  - Access to forbidden file: /posts/1%27+OR+1=1+--+-+"
  - Access to forbidden file: /posts/1%27+OR+1=1+--+-+"
  - Failed login attempt: /../../etc/passwd"
  - Failed login attempt: /../../etc/passwd"
  - Failed login attempt: /../../../../etc/passwd"
  - Failed login attempt: /../../../../etc/passwd"
  - Access to forbidden file: /.travis.yml~"
```

FLAG: 0byteCTF{s3M4n9At_4mPAt_l1m4_13377}

Conclusion (Kesimpulan dari soal)

sama seperti sebelumnya, ini tentang analysis file access.log (apache log)