**Write-Up Hack-Today 2023**

**Name : Fikri Muhammad Abdillah  
Username : FlaB  
Team : Banana 🍌🍌🍌**

**# Solved Category :  
 1. Cryptography** - DejaVu (3 solve – 492pts)  
 - AES Enjoyer (6 solve – 465pts)  
 - spam (17 solve – 212pts) **2. Forensic** - doodled (30 solve – 100pts) **3. Misc  
 -** Simulasi UTBK (10 solve – 401pts) **4. Reverse Engineering** - OnlyAdminCanSee (30 solve – 100pts) **5. Web Exploitation** - LoginInspek (44 solve - 100 pts)

1. **Cryptography**

**DejaVu (3 solve – 492pts)**

Disini kita di berikan source (server.py) dan connection tcp. Fokus Soal tersebut adalah tentang LCG cracking dan leaking states dari LCG. Setiap parameter LCG di generate secara random menggunakan function **`get\_all\_key()`** yang is function nya tidak di ketahui.

Untuk process cracking LCG, kita harus mendapatkan beberapa baris state yang di hasilkan, kita dapat tahu dengan cara leaking. Leaking dilakukan dengan cara bruteforce encrypt byte per byte dengan inputan `\x01`, dan jika ada delay yang terjadi. Maka state telah mendapatkan akhir byte nya atau bisa dikatakan 1 state telah di dapatkan.

Setelah crack LCG, kita input start state dengan state pertama yg kita dapatkan, lalu mengambil state sebelumnya dengan algoritma:

Dengan persyaratan (Coprime)

Berikut code saya:

from Crypto.Util.number import \*

import functools

from pwn import \*

import time

from math import floor

import sys

sys.set\_int\_max\_str\_digits(100000)

def bytes2bin(msg: bytes):

    return bin(int.from\_bytes(msg, "big"))[2:]

def bin2bytes(msg: bytes):

    return int(msg, 2).to\_bytes((int(msg, 2).bit\_length() + 7) >> 3, "big") or b"\x00"

class KeyGen:

    def \_\_init\_\_(self, x: int, m: int, c: int, n: int):

        self.m = m

        self.c = c

        self.n = n

        self.state = x % n

        self.bitstate = bin(self.state)[2:]

    def update\_state(self, isflag=0):

        self.state = (self.state \* self.m + self.c) % self.n

        self.bitstate = bin(self.state)[2:]

        if isflag:

            return

        time.sleep(1)

    def downdate\_state(self):

        self.state = (self.state - self.c) \* inverse(self.m, self.n) % self.n

        self.bitstate = bin(self.state)[2:]

        return self.state

    def get\_bit(self, isflag=0):

        b = self.bitstate[-1]

        self.bitstate = self.bitstate[:-1]

        if not self.bitstate.isdigit():

            self.update\_state(isflag)

        return int(b)

def crack\_unknown\_increment(states, modulus, multiplier):

    increment = (states[1] - states[0]\*multiplier) % modulus

    return modulus, multiplier, increment

def crack\_unknown\_multiplier(states, modulus):

    multiplier = (states[2] - states[1]) \* inverse(states[1] - states[0], modulus) % modulus

    return crack\_unknown\_increment(states, modulus, multiplier)

def crack\_unknown\_modulus(states):

    diffs = [s1 - s0 for s0, s1 in zip(states, states[1:])]

    zeroes = [t2\*t0 - t1\*t1 for t0, t1, t2 in zip(diffs, diffs[1:], diffs[2:])]

    modulus = abs(functools.reduce(GCD, zeroes))

    return crack\_unknown\_multiplier(states, modulus)

def enc\_message(io, msg:bytes):

    io.sendlineafter(b'[>]', b'1')

    io.sendlineafter(b':', msg)

    start\_time = time.time()

    res = bytes.fromhex(io.recvline().strip().decode().split(': ')[1])

    long\_time = time.time() - start\_time

    return res, floor(long\_time)

def get\_flag(io):

    io.sendlineafter(b'[>]', b'2')

    return bytes.fromhex(io.recvline().strip().decode().split(': ')[1])

def get\_6\_state(io):

    state = []

    half\_first\_state = None

    for \_ in range(7):

        temp\_state = ""

        while True:

            temp = enc\_message(io, b'\x01')

            temp\_state = str(int(temp[0].hex(), 16) ^ 1) + temp\_state

            if temp[1] >= 1:

                if half\_first\_state == None:

                    half\_first\_state = temp\_state

                else:

                    state.append(int(temp\_state, 2))

                break

    return state, half\_first\_state

def decrypt\_flag(io, ct):

    state, half\_first\_state = get\_6\_state(io)

    modulus, multiplier, increment = crack\_unknown\_modulus(state)

    print(f'[+] modulus = {modulus}')

    print(f'[+] multiplier = {multiplier}')

    print(f'[+] increment = {increment}')

    print(f'[+] half\_first\_state = {half\_first\_state}')

    key = KeyGen(state[0], multiplier, increment, modulus)

    keys = bin(key.downdate\_state())[2:].replace(half\_first\_state, "")

    res = ""

    for i in bytes2bin(ct)[::-1]:

        res = str(int(i) ^ int(keys[0])) + res

        keys = keys[1:]

        if len(keys) == 0:

            keys = bin(key.downdate\_state())[2:]

    return bin2bytes(res)

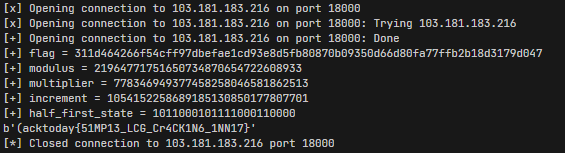
# nc 103.181.183.216 18000

io = remote('103.181.183.216', 18000)

flag = get\_flag(io)

print(f'[+] flag = {flag.hex()}')

print(decrypt\_flag(io, flag))



**Flag: hacktoday{51MP13\_LCG\_Cr4CK1N6\_1NN17}**

**AES Enjoyer (6 solve – 465pts)**

Pertama kita diberikan source (server.py) dan connection tcp, disini berisi soal tentang AES, Disini focus soal adalah tentang vulnerability dari type mode encryption yaitu CFB, CFB (Cipher FeedBack) vulnerable dengan secret recovery dari leak xor-key yang di dapat di awal karena kita bebas untuk memasukkan iv nya. Kita mendapat 2 part file, untuk yang pertama kita di suruh untuk recovery secret (flag) yang telah di tambahkan ke dalam encryption yg di masukkan namun untuk case soal ctf ini, kita hanya di beri kesempatan 4 kali untuk action, untuk part 2 kita hanya perlu decryption secara otomatis, karena kita bebas mengisi key & iv nya.

Berikut code nya:

from Crypto.Util.number import \*

from pwn import \*

from Crypto.Cipher import AES

def dec\_gift(pt : bytes, key):

    iv = b"hektoday"\*2

    assert len(key) == 16 and len(iv) == 16

    aes = AES.new(key.encode(), AES.MODE\_CBC, iv=iv)

    return aes.decrypt(pt)

def encrypt(io, msg: bytes, IV: bytes):

    io.sendlineafter(b"[>] ", b"1")

    io.sendlineafter(b"[>] IV (hex): ", IV.hex().encode())

    io.sendlineafter(b"[>] Plaintext (hex): ", msg.hex().encode())

    return bytes.fromhex(io.recvline().split(b": ")[-1].strip().decode())[16:]

def get\_gift(io):

    io.sendlineafter(b"[>] ", b"2")

    key1 = io.recvline().strip().decode()

    key2 = io.recvline().strip().decode()

    return bytes.fromhex(io.recvline().split(b": ")[-1].strip().decode()), key1, key2

# nc 103.181.183.216 18002

io = remote("103.181.183.216", 18002)

gift, key1, key2 = get\_gift(io)

gift = dec\_gift(gift, key2)

gift = dec\_gift(gift, key1)

key1 = encrypt(io, b"\x00"\*16, b"\x00"\*16)[:16]

enc = encrypt(io, key1, b"\x00"\*16)

key2 = encrypt(io, b"\x00"\*16, enc[16:32])[:16]

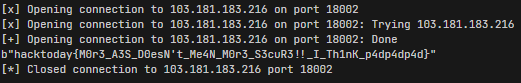
flag = xor(enc[32:48], key2)

flag += gift

print(flag)

saya mendapat beberapa masalah sebelum code saya diatas, karena saat mencoba block cipher yang kedua, saya hanya mendapat padding nya saja, jadi saya leaking xor-key yang ke 3, untuk decrypt block cipher yg ke 3.

Hasil:



**Flag: hacktoday{M0r3\_A3S\_D0esN’t\_Me4N\_M0r3\_S3cur3!!\_I\_Th1nk\_p4dp4dp4d}**

**Spam (17 solve – 212pts)**

Kita diberikan source (server.py) dan connection tcp, saat connect ke dalam tcp kita akan di beri banyak sekali message berisi email spam dan password yang telah di encrypt menggunakan rsa. Setelah itu kita harus memasukkan password yang telah teracak ke dalam message tersebut dan habis itu kita akan di berikan flag.

challenge ini dapat di serang menggunakan broadcast attack (ada email banyak yg terduplikat) dan pollard-rho (karena salah satu prime factor dari N itu kecil, 16-bits). Saya di sini menggunakan pollard-rho untuk factor dan mengambil password nya

code:

from pwn import \*

from Crypto.Util.number import \*

from tqdm import tqdm

from primefac import pollardrho\_brent

def factor(N):

    res = []

    for n in tqdm(N):

        p = pollardrho\_brent(n)

        res.append([p, n // p])

    return res

io = remote('103.181.183.216', 18001)

N = []

C = []

raw\_temp = io.recvuntil(b"Input Full Password").decode().split("\n")

message = {}

for temp in raw\_temp:

    if "Input Full Password" in temp:

        for i, n in enumerate(factor(N)):

            phi = 1

            for p in n:

                phi \*= p - 1

            d = inverse(65537, phi)

            m = long\_to\_bytes(pow(C[i], d, N[i])).decode()

            if message.get(m) is None:

                message[m] = 1

            else:

                message[m] += 1

        message = {k: v for k, v in sorted(message.items(), key=lambda item: item[1])}

        res = []

        for m in message:

            if "happy\_birthday" not in m:

                res.append(m)

    elif "n" in temp:

        N.append(int(temp.split("=")[-1]))

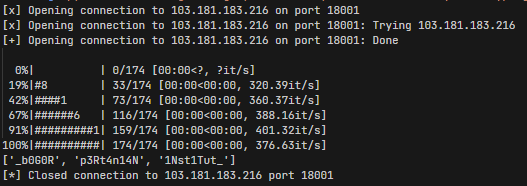
    elif "c" in temp:

        C.append(int(temp.split("=")[-1]))

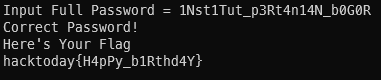
print(res)

io.interactive()

Hasil:



Password: 1Nst1Tut\_p3Rt4n14N\_b0G0R



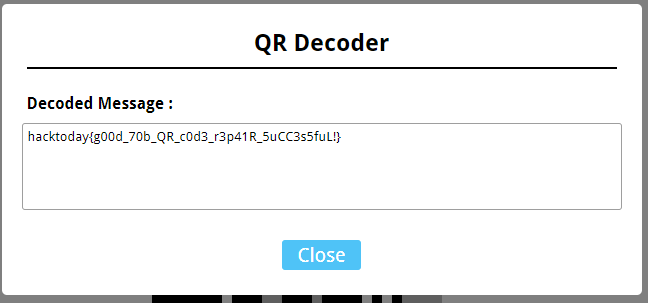
**Flag: hacktoday{H4pPy\_b1Rthd4Y}**

1. **Forensic**

**Doodled (30 solve – 100pts)**

Disini kita diberikan gambar yang berupa QR Code yang telah di coret2, saya di sini menggunakan tool yaitu QRazyBox (src: <https://github.com/Merricx/qrazybox>). Disini saya menggambarnya dengan manual, Hasil:





**Flag: hacktoday{g00d\_70b\_QR\_c0d3\_r3p41R\_5uCC3s5fuL!}**

1. **Misc**

**Simulasi UTBK (10 solve – 401)**

Disini kita diberikan connection TCP dan saat connect ke dalamnya, kita benar2 di beri soal UTBK. Karena tidak mungkin untuk soalnya itu tidak terbatas, saya bruteforce soal tersebut untuk dump semua soal, dan menjawabnya secara otomatis jika soal telah ada dalam local dumped database, code:

from pwn import \*

import json

from time import sleep

context.log\_level = "warning"

try:

    with open("DICT\_QUESTION.json", "r") as f:

        DICT\_QUESTION = json.load(f)

except:

    DICT\_QUESTION = {}

while True:

    try:

        io = remote("103.181.183.216", 19003)

        life = 3

        point = 0

        while True:

            io.recvuntil(b"nyawa kamu")

            io.recvline()

            question = io.recvline().decode().strip()

            if question not in DICT\_QUESTION:

                io.sendlineafter(b":", b"aaaaa")

                io.recvuntil(b"jawaban yang benar adalah ")

                answer = io.recvline().decode().strip()

                DICT\_QUESTION[question] = answer

                point = 0

                life -= 1

            else:

                io.sendlineafter(b":", DICT\_QUESTION[question].encode())

                temp = io.recvline().decode().strip()

                if "jawaban yang benar adalah " in temp:

                    point = 0

                    DICT\_QUESTION[question] = temp.split("jawaban yang benar adalah ")[1]

                    life -= 1

                else:

                    point += 1

            if life == 0:

                io.close()

                break

            if point == 100:

                io.interactive()

                with open("DICT\_QUESTION.json", "w") as f:

                    json.dump(DICT\_QUESTION, f)

                exit()

    except KeyboardInterrupt:

        print("point: ", point)

        print("life: ", life)

        with open("DICT\_QUESTION.json", "w") as f:

            json.dump(DICT\_QUESTION, f)

        break

    except Exception as e:

        print("point: ", point)

        print("life: ", life)

        print("error: ", e)

        with open("DICT\_QUESTION.json", "w") as f:

            json.dump(DICT\_QUESTION, f)

        sleep(0.5)

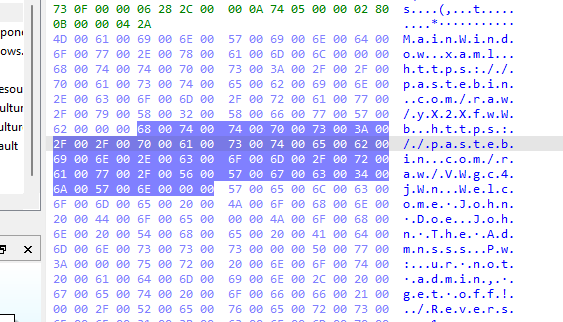
Hasil:



**Flag: hacktoday{just\_make\_your\_own\_bank\_soal\_ab1329fa9b}**

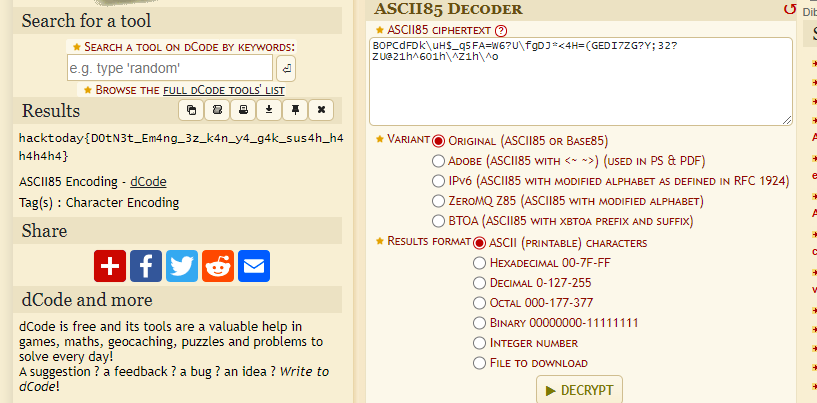
1. **Reverse Engineering**

**OnlyAdminCanSee (30 solve – 100pts)**

Disini kita diberikan attachment berupa file .exe (executable - windows), saat saya jalankan program nya, kita disuruh memasukkan password, jadi saya decompile file .exe tersebut menggunaka IDA. Saat saya decompile saya tidak familiar dengan bahasanya, lalu saya coba lihat di hex, namun saya melihat sesuatu yang menarik  


Terdapat 2 link

* <https://pastebin.com/raw/yX2XfwWb> - Berisi fake flag
* <https://pastebin.com/raw/VWgc4jWn> - berisi encrypted flag dengan ASCII-85 Encoding

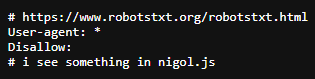


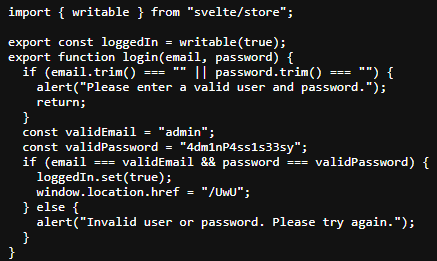
**Flag: hacktoday{DOtN3t\_Em4ng\_3z\_k4n\_y4\_g4k\_sus4h\_h4h4h4h4}**

1. **Web Exploitation**

**LoginInspek (44 solve – 100pts)**

Disini kita diberikan link web, karena di deskripsi di singgung tentang Mr.Robot, jadi saya mencoba untuk akses **`/robots.txt`** dan ternyata berisi hal menarik yaitu **`/nigol.js`** saat saya membuka file js tersebut, saya dapat email dan password yang tertulis secara hardcoded di file js tersebut, lalu saya langsung saja memasukkan email dan password ke dalam halaman login dan mendapatkan flag nya.





\*note: hari ini saat saya membuat write up, saya tidak dapat login dengan cara yang sama, jadi saya tidak dapat mencantumkan flag nya