Windows下的"你画我猜" -- 告别效率低下的目录扫描方法

watcher / 2018-05-01 15:24:59 / 浏览数 6937 技术文章 技术文章 顶(2) 踩(0)

Windows下的"你画我猜" -- 告别效率低下的目录扫描方法

0x01 前言

对Web应用进行安全测试时，我们常常需要猜测网站的结构与目录，若能得到网站详尽的拓扑结构，得到重要的目录名，文件名信息，可以为我们后续的测试创造良好的基

在Linux下，我们并没有特别好的办法，只能尝试去识别应用指纹，或者是暴力枚举，这是一种效率很低下的目录扫描方法，而且效果很差。

但是如果碰到的是Windows环境，我们就拥有了很多能与Web管理员斗智斗勇的方法，和他们玩一场"你画我猜"的游戏，告别效率低下的目录扫描方法。

由于篇幅长度原因，本文暂对如下几种环境进行测试

```
Windows + IIS    + .net
Windows + IIS    + PHP
Windows + Apache + PHP
Windows + Nginx  + PHP
```

先讲两部分的前置知识:

0x02 WinAPI与通配符的那些事

在Windows环境下使用PHP时，PHP中的部分函数会调用2个底层Windows API函数FindFirstFileExW()，FindFirstFile()

这两个函数对 < > " 三个字符做了特别的对待和处理:

大于号 > 等价于 通配符 ？ 0次或1次

小于号 < 等价于 通配符 * 0次或多次

双引号 " 等价于 通配符 . 匹配除换行符(\n, \r)之外的任意单个字符

也就是说，在Windows下的PHP的某些函数中，我们可以使用< > "来匹配一些文件名/目录名

这里的部分函数包括但不限于:

```
include()
include_once()
require()
require_once()
fopen()
copy()
file_get_contents()
readfile()
file_put_contents()
mkdir()
opendir()
readdir()
move_uploaded_file()
getimagesize()
......
```

其实我们可以发现，这主要是WinAPI的问题，任何调用这些WinAPI的语言都有可能存在以上问题，不只是PHP，这里只是以PHP为例。

0x03 IIS短文件名的那些事

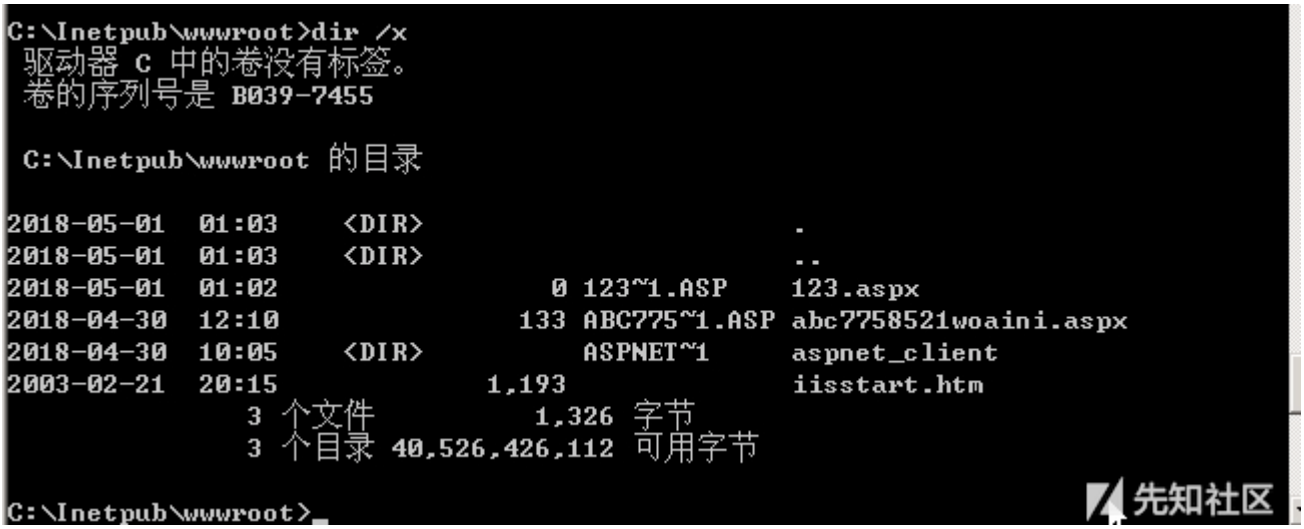
远古时代的DOS下存在这样的文件命名规则:

■■■■■■■■8■■■■■■■■■■■■3■■■

所以我们也把它叫为8.3格式。

Windows为了兼容MS-DOS，为文件名超过8位，扩展名超过3位的文件都会对应地创建一个8.3格式的文件名，也称为短文件名。

在cmd下使用命令 dir /x 可以查看文件对应的短文件名



可以发现短文件名有如下2个特征:

1.文件名只显示前6个字符，后续字符用~1代替。当存在多个文件名类似的文件时(文件名前6位相同，且后缀名前3位相同)，数字1会进行递增。

2.后缀只显示前3个字符。

0x04 Windows + IIS 6 + .net 文件/目录猜测

适用条件与局限:

1.被猜测文件的文件名长度需超过8位，或后缀名超过3位。

2.环境为Windows + IIS + .net

3.只能猜测出文件名的前6位，后缀名的前3位。

说到猜测目录，不得不提IIS短文件名漏洞，该漏洞于2012年由一位安全研究员发现。

当我们访问某个存在的短文件名时，会返回404。而当我们访问某个不存在的短文件名时，会返回400，依据返回结果的不同，就可以进行逐位猜测了。

比如要猜测出如下文件

```
abc7758521woaini.aspx
```

- 猜测文件名/目录名

进行如下猜解，不断向下猜解完所有的6个字符，

```
http://192.168.219.129/a*~1****/xxx.aspx   ■■404  ■■■■

http://192.168.219.129/aa*~1****/xxx.aspx  Bad Request ■■■■

http://192.168.219.129/ab*~1****/xxx.aspx  ■■404

http://192.168.219.129/abc*~1****/xxx.aspx ■■404

......

http://192.168.219.129/abc774*~1****/xxx.aspx  Bad Request

http://192.168.219.129/abc775*~1****/xxx.aspx  ■■404
```

到了这一步，我们要来判断这是一个目录还是一个文件。

若如下请求返回404，则代表它是一个目录，否则就是一个文件

```
http://192.168.219.129/abc775*~1/xxx.aspx
```

- 猜测文件后缀

接下来开始猜解文件后缀，不断向下猜解完所有的3个字符

```
http://192.168.219.129/abc775*~1*a**/xxx.aspx
```

```
http://192.168.219.129/abc775*~1*as*/xxx.aspx

http://192.168.219.129/abc775*~1*asp/xxx.aspx
```

返回如下结果:

至此，我们的猜测也就结束了，剩下的位数只能靠脑洞去猜了。

0x05 Windows + IIS 7.x + .net 文件/目录猜测

之所以把IIS 7.x单独拿出来说，是因为之前踩过一次很深的坑，郁闷了很久。

根据Soroush Dalili的研究报告，不同版本的IIS返回结果如下:

IIS Version	URL	Result/Error Message
IIS 6	/valid*~1*/.aspx	HTTP 404 - File not found
IIS 6	/Invalid*~1*/.aspx	HTTP 400 - Bad Request
IIS 5.x	/valid*~1*	HTTP 404 - File not found
IIS 5.x	/Invalid*~1*	HTTP 400 - Bad Request
IIS 7.x .Net.2 No Error Handling	/valid*~1*/	Page contains: "Error Code 0x00000000"
IIS 7.x .Net.2 No Error Handling	/Invalid*~1*/	Page contains: "Error Code 0x80070002"

作者给出了IIS 7.x开启了详细错误之后，不同页面的变化情况，然而默认情况下都是显示一个自定义的404页面，并不会给出详细错误。

我们的目标是猜测出根目录下的一个目录名:

```
abcdefg1234567aaabbb
```

在默认情况下，无论该文件/目录是否存在，都会返回一个404，返回结果没有差异的话，我们也就无法猜测了。

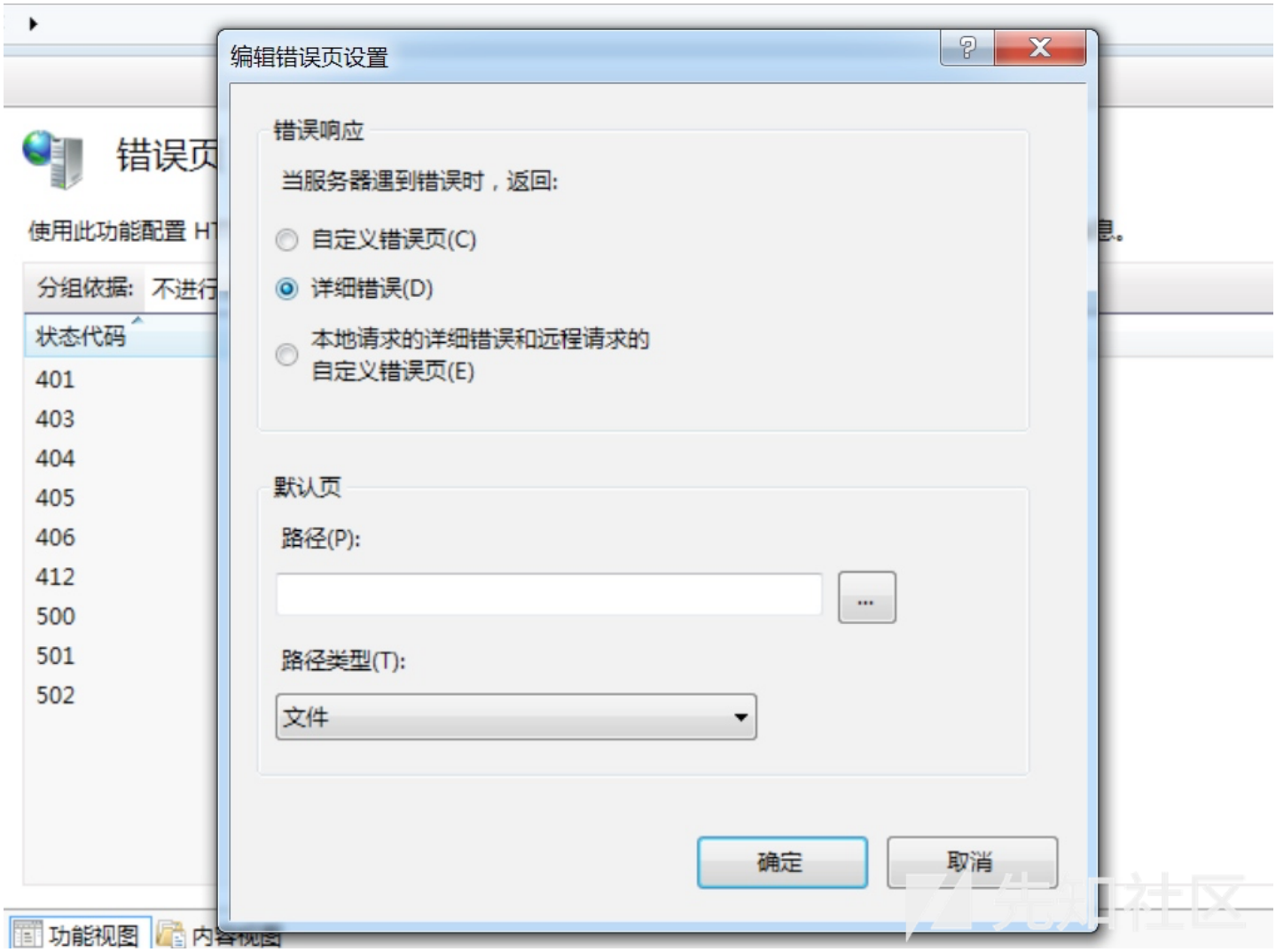
```
http://192.168.219.240/a*~1****/xxx.aspx
```

```
http://192.168.219.240/b*~1****/xxx.aspx
```

这里郁闷了好久，最后发现使用OPTIONS方式请求，可以得到不一样的返回结果:

存在时会返回404，不存在时会返回200，根据返回结果的差异，我们就可以探测出目录名的短文件名:

当我们主动开启了详细错误之后，使用GET请求也可以得到有差异的结果了。



若存在 http://192.168.219.243/a*~1****/xxx.aspx 错误代码为 0x00000000

若不存在 http://192.168.219.243/b*~1****/xxx.aspx 错误代码为 0x80070002

0x06 Windows + Apache + PHP 文件/目录猜测

在该环境下，我们就无法像在IIS下直接用URL访问的方式去逐位猜测短文件名了，但是在该环境下，有两个杀伤力更大的特性:

1.当Web程序中存在某些函数时(前置知识中提到的)，我们借助它们来逐位猜测出完整目录名，文件名，并且没有长度大小的限制。

2.虽然无法直接用URL访问的方式去逐位猜测出 短文件名，但是可以用URL访问的方式直接访问/下载 已知短文件名的目录/文件

首先介绍第1个特性

- 特性1

这里以HITB 2018的一道CTF题目为例，示例代码如下:

```
<?php

$filename = $_GET['filename'];

$file = "./abcdefg1234567aaabb/" . $filename;

var_dump(getimagesize($file));

?>
```

网站存在上传功能，我们将一个php木马上传到了某目录下(abcdefg1234567aaabb)，但是不知道其目录名。

幸运的是，我们可以向1.php的filename参数传入该目录下的一个图片名，程序会返回该图片的尺寸信息。

执行如下访问

```
http://192.168.219.197/1.php?filename=../a</01.png  ■■■■

http://192.168.219.197/1.php?filename=../aa</01.png ■■■■

http://192.168.219.197/1.php?filename=../ab</01.png ■■■■

......

http://192.168.219.197/1.php?filename=../abcdefg</01.png ■■■■

......

http://192.168.219.197/1.php?filename=../abcdefg1234</01.png ■■■■

......

http://192.168.219.197/1.php?filename=../abcdefg1234567aaabb</01.png  ■■■■
```

接下来的26 + 10 次尝试中，均返回错误，证明我们已经将目录名猜解完毕，得到目录名:

```
abcdefg1234567aaabb
```

假设我们连php木马名也不知道呢?猜解文件名也是同一个道理

```
http://192.168.219.197/1.php?filename=../abcdefg1234567aaabb/w<.php

http://192.168.219.197/1.php?filename=../abcdefg1234567aaabb/wo<.php

http://192.168.219.197/1.php?filename=../abcdefg1234567aaabb/woa<.php

http://192.168.219.197/1.php?filename=../abcdefg1234567aaabb/woai<.php

......

http://192.168.219.197/1.php?filename=../abcdefg1234567aaabb/woainihhhh<.php
```

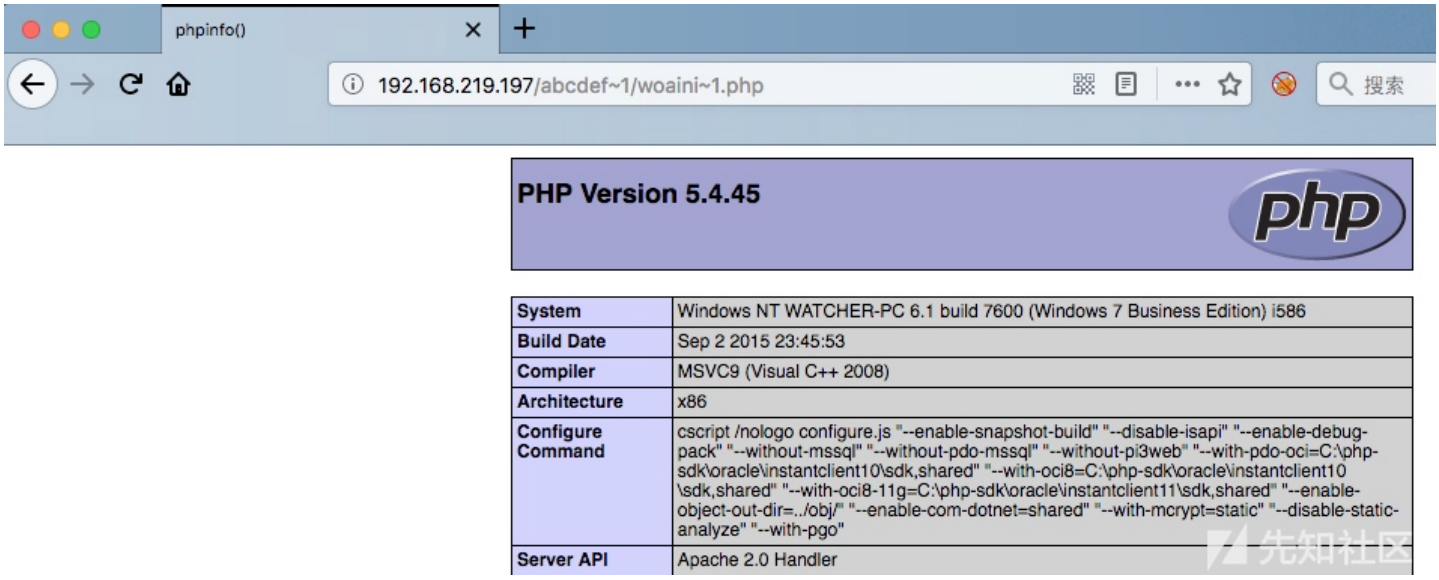
接下来的26 + 10 次尝试中，均返回错误，证明我们已经将文件名猜解完毕，得到文件名。将目录名和文件名拼接:

```
http://192.168.219.197/abcdefg1234567aaabb/woainihhhh.php
```

- 特性2

事实上，在上一步中，我们并不需要猜测出完整文件名，目录名，用短文件名就可以直接访问，下载了。

```
http://192.168.219.197/abcdef~1/woaini~1.php
```

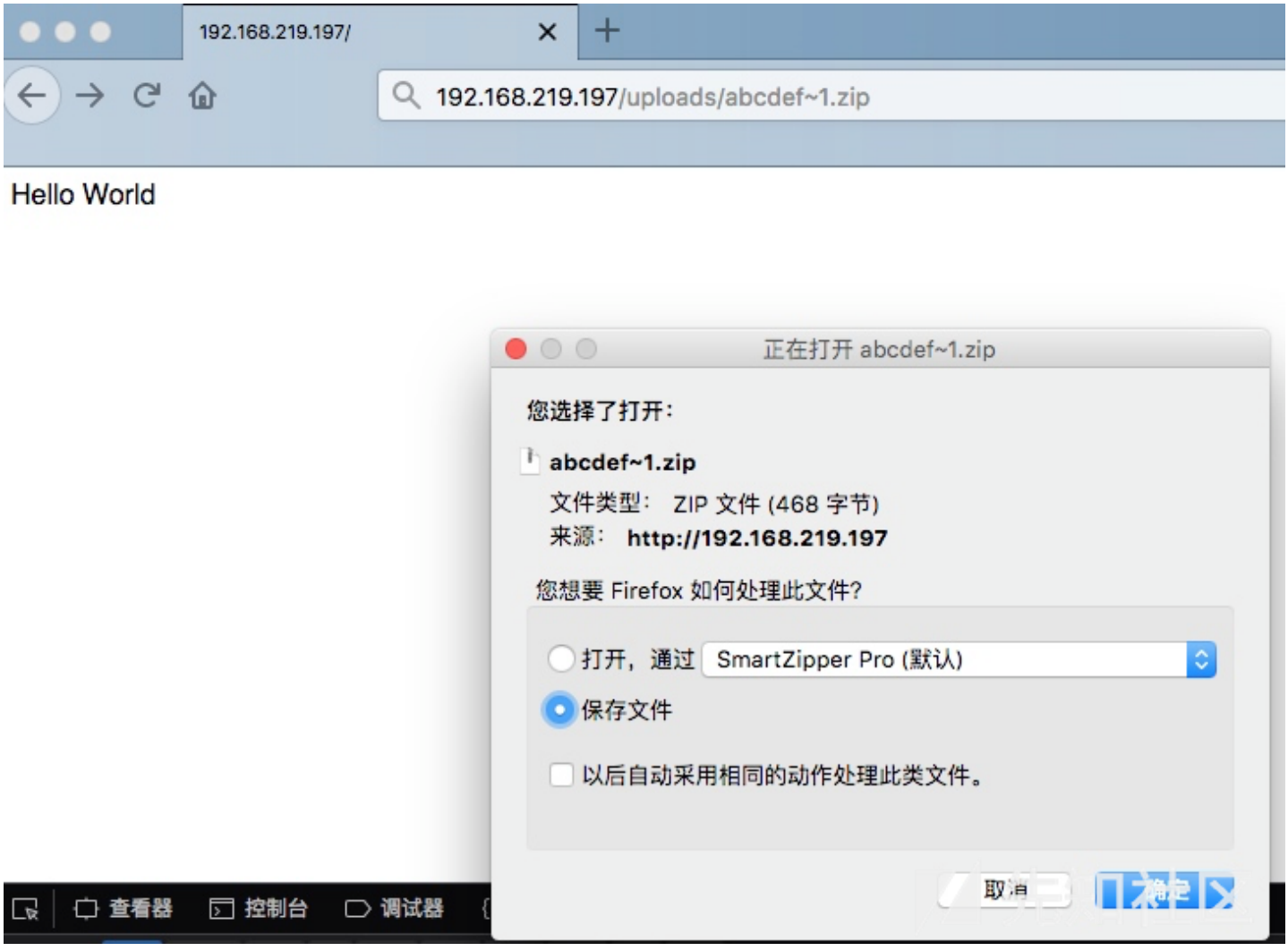


想要下载如下文件

```
http://192.168.219.197/uploads/abcdefgasd1241asd123sgadg123sdgasd123dzg.zip
```

同样可以使用短文件名去下载

```
http://192.168.219.197/uploads/abcdef~1.zip
```



这种方法在IIS下无法使用，IIS不接受直接用短文件名访问的请求方式。

0x07 Windows + IIS + PHP 文件/目录猜测

在该环境下，可以用到很多前面提到的特性，所以遇到Windows + IIS + PHP就偷偷笑吧！

1.可以使用URL访问的方式，来逐位猜测出目录/文件的短文件名

2.当Web程序中存在某些函数时(前置知识中提到的)，我们借助它们来逐位猜测出完整目录名，文件名，并且没有长度大小的限制。

3.可以使用URL访问的方式，用通配符直接访问文件(但不能访问目录)

首先介绍特性1

- 特性1

我们使用前面提到过的OPTIONS请求方式来逐位猜测目录的短文件名

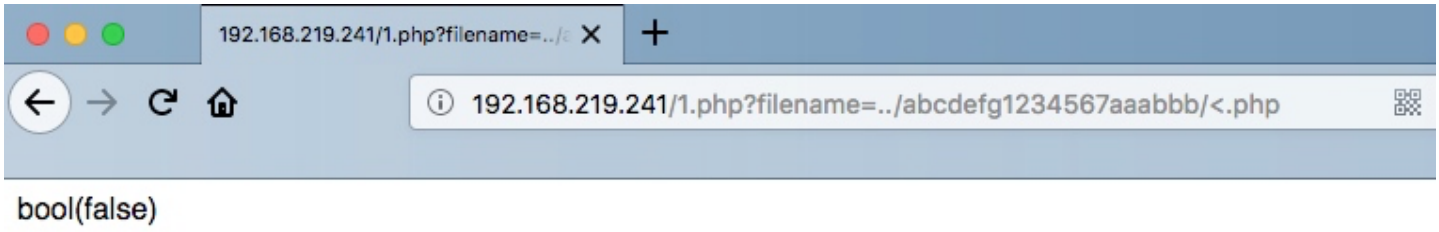
存在时返回404，不存在时返回200

- 特性2

该特性上节介绍过了，是PHP的特性，与Web服务器种类无关。

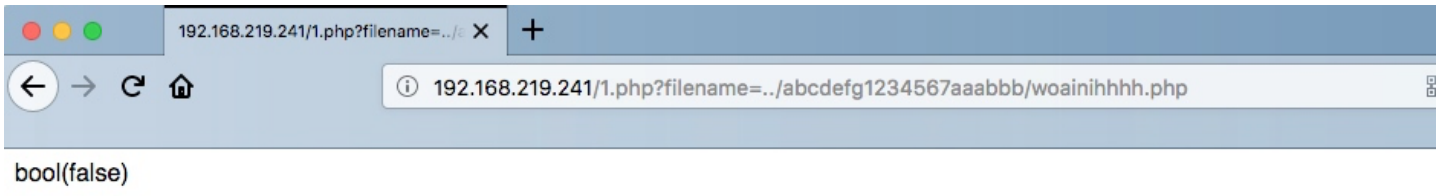
首先逐位猜测出目录名，过程略

```
http://192.168.219.241/1.php?filename=../abcdefg1234567aaabbb/<.php
```



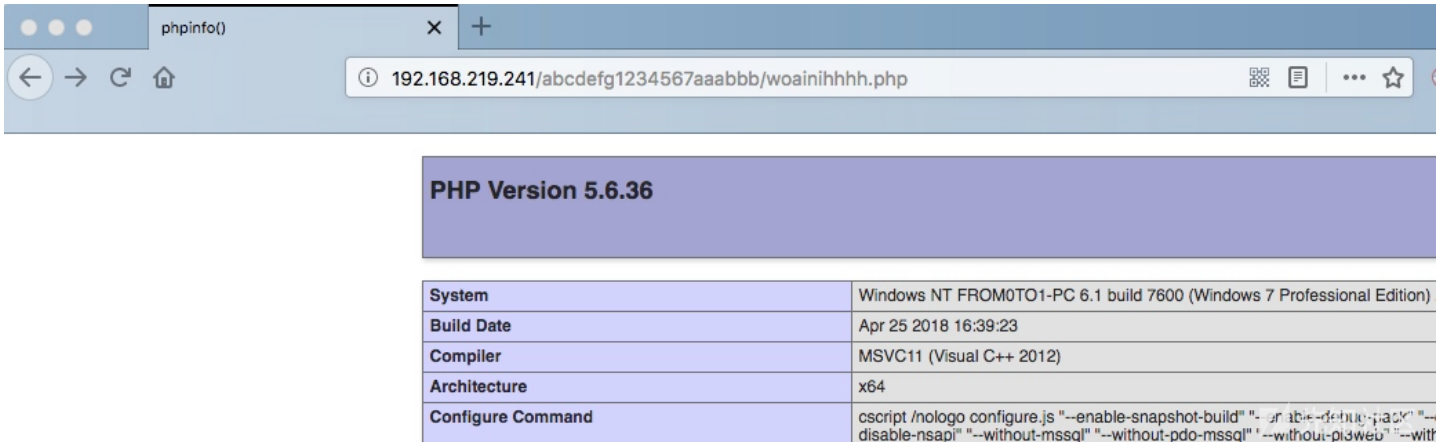
用同样的方法逐位猜测出文件名:

```
http://192.168.219.241/1.php?filename=../abcdefg1234567aaabbb/woainihhhh.php
```



继而得到了完整文件路径:

```
http://192.168.219.241/abcdefg1234567aaabbb/woainihhhh.php
```



- 特性3

事实上我们猜测出目录名就足够了，文件名可以直接使用通配符去访问:

```
http://192.168.219.241/abcdefg1234567aaabbb/w<.php
```

0x08 Windows + Nginx + PHP 文件/目录猜测

在该环境下，依然可用使用提到的多个特性，但是有一点微小的差异:

1.可以直接用短文件名，访问目录和文件

2.可以直接用通配符访问文件，但是不能访问目录

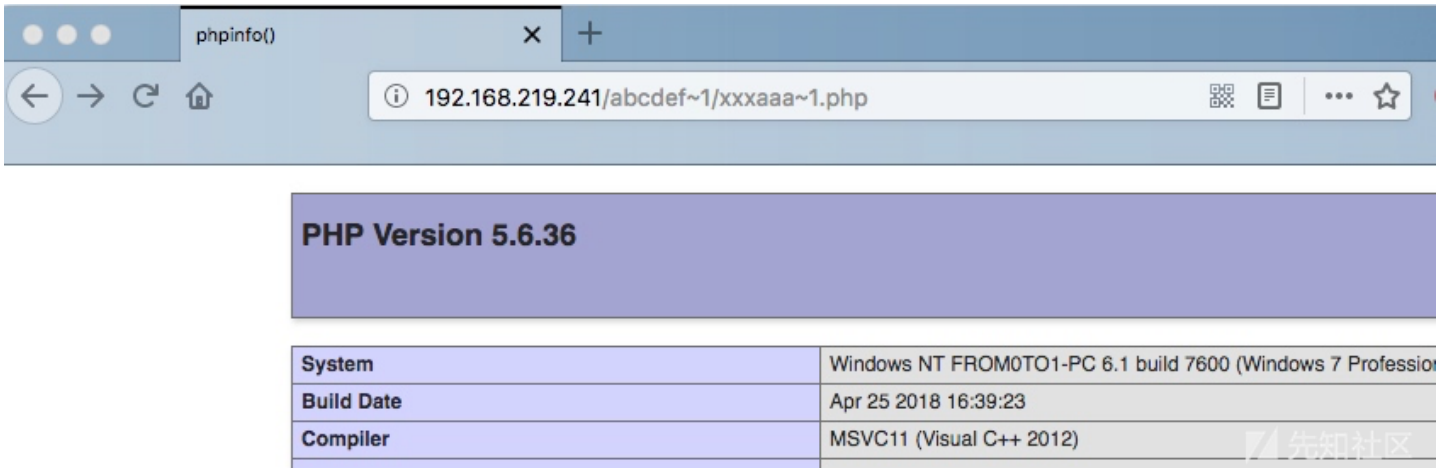
3.PHP某些函数的特性

- 特性1

```
http://192.168.219.241/abcdefgwoaini123/xxxaaabbbccc.php
```

■■■

```
http://192.168.219.241/abcdef~1/xxxaaa~1.php
```

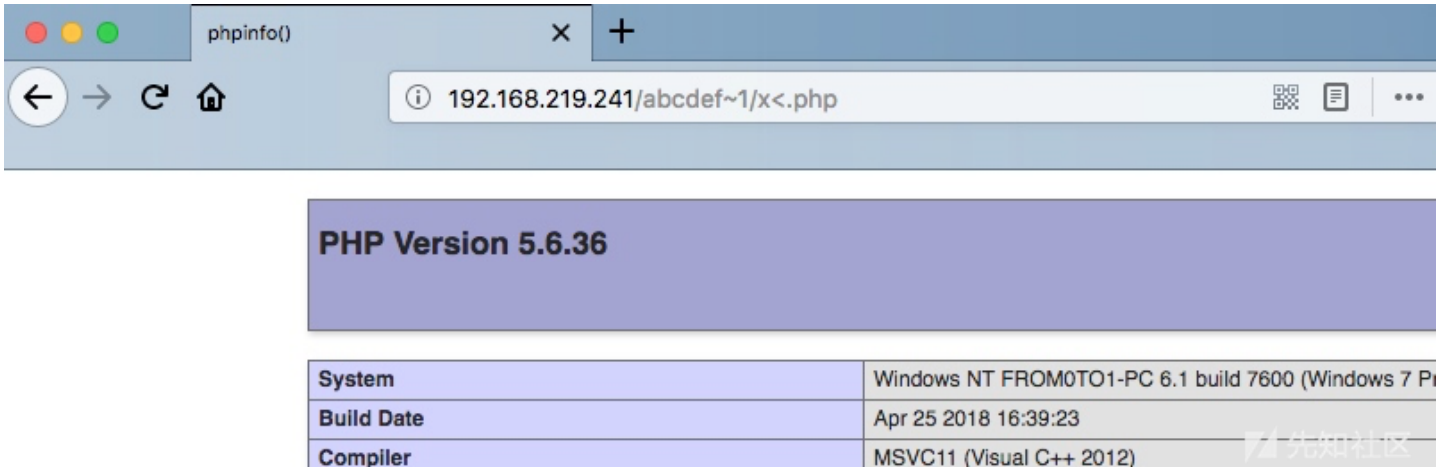


- 特性2

```
http://192.168.219.241/abcdefgwoaini123/xxxaaabbbccc.php
```

■■■

```
http://192.168.219.241/abcdef~1/x<.php
```



- 特性3

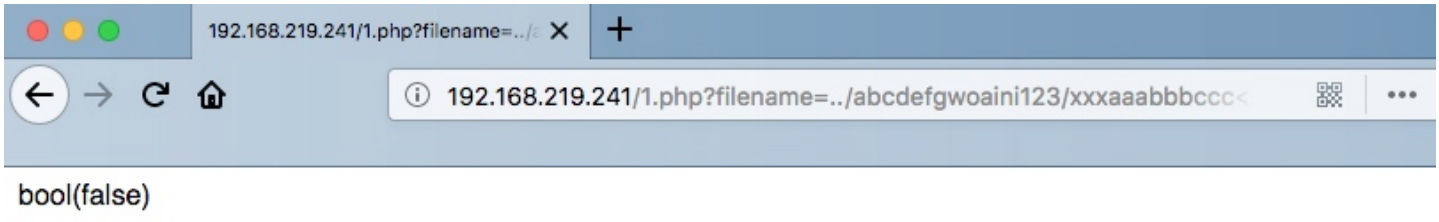
逐位猜测目录名:

```
http://192.168.219.241/1.php?filename=../abcdefgwoaini123</1.php
```

这里稍有不同，Read error代表存在

逐位猜测文件名

```
http://192.168.219.241/1.php?filename=../abcdefgwoaini123/xxxaaabbbccc<.php
```



进而拼接处完整文件路径:

http://192.168.219.241/abcdefgwoaini123/xxxaaabbbccc.php

0x09 结尾

前面提到的各种环境下不同的特性，其实都是由WinAPI，通配符，短文件名这三者，与Web服务器，Web语言本身的一些特性结合之后产生的。

所以说，影响范围不只是上面这5种，与其它语言，其它类型的Web服务器，结合之后，很可能又会产生更多更奇怪的特性来，有兴趣的同学可以去测试。

借助于这些特性，在遇到Windows环境下的目录探测问题时，我们可以更从容，更巧妙地进行高效的文件,目录的探测。

这篇文章也从另一个角度证明，Linux确实更适合作为服务器，安全性在某种程度上也要更好一些，至少不存在这么多奇怪的特性。

0x10 参考

- https://www.cnblogs.com/palidin/p/7831061.html
- https://soroush.secproject.com/downloadable/microsoft_iis_tilde_character_vulnerability_feature.pdf
- https://github.com/lijiejie/IIS_shortname_Scanner
- https://xz.aliyun.com/t/2004
- http://www.moonsec.com/post-304.html
- https://www.xctf.org.cn/library/details/hitb-quals-2018/#upload-web

上一篇：Pwn with File结构体之... 下一篇：内网渗透中用到的计划任务

1. 1 条回复

pepsi 2018-05-17 22:33:40

总结的很好，学习了！

0 回复Ta

登录 后跟帖

先知社区

现在登录

热门节点

技术文章

社区小黑板

目录

RSS 关于社区 友情链接 社区小黑板