PHP源码调试之Windows文件通配符分析

安全小飞侠 / 2018-01-30 20:30:47 / 浏览数 3727 安全技术 漏洞分析 顶(0) 踩(0)

0x00 前言

很早之前就有国外的安全研究人员发现PHP语言在Windows上的一些奇妙特性：

- 大于号(>)相等于通配符问号(?)
- 小于号(<)相当于通配符星号(*)
- 双引号(")相当于点字符(.)

具体文章参见：https://soroush.secproject.com/blog/2014/07/file-upload-and-php-on-iis-wildcards/

那么问题来了，根本原因是什么呢？是PHP语言本身的问题还是Windows的锅呢？

0x01 分析

处于对这个问题的好奇，笔者进行了一下的挖掘和分析。
在分析之前为了避免自己做了无用功，先Google了一番，找到了以下这些文章且都是通过Fuzz的方法发现的，并没有太多资料解释更深层次的原因。

- https://github.com/ironbee/ironbee-rules/blob/master/support/php/test_fs_evasion.php
- http://www.ush.it/2009/07/26/php-filesystem-attack-vectors-take-two/

静态分析

既然没有现成的解释，那么我们便可以自己动手分析以下。
首先，下载PHP的源代码进行尝试进行静态分析。

```
git clone https://github.com/php/php-src
git checkout PHP-7.2.1
```

随便选取一个可以操作文件的PHP方法, 如`getimagesize`.
尝试进行全局搜索，我们在`\php-src\ext\standard\image.c`的第1501-1506行发现了该方法的具体定义：

```
/* }}} */

/* {{{ proto array getimagesize(string imagefile [, array info])
  Get the size of an image as 4-element array */
PHP_FUNCTION(getimagesize)
{
   php_getimagesize_from_any(INTERNAL_FUNCTION_PARAM_PASSTHRU, FROM_PATH);
}
```

可见，`getimagesize`方法调用了`php_getimagesize_from_any`方法，那么接下来又是如何调用的呢？当然，你可以继续逐层追踪下去，但是这将会比较费时费力同时也

动态调试

在动态调试之前，我们需要做一些提前的准备如下：

- PHP的源码(本文调试的版本是7.2.1)
- Visual Studio 2017

参考PHP官方文档在Windows上编译PHP-7.2.1：
https://wiki.php.net/internals/windows/stepbystepbuild_sdk_2
具体步骤这里不在赘述，唯一需要注意的点在于在编译之前用以下的命令来建立自己的configure文件：

```
configure --enable-debug --enable-phpdbg
```

编译完成之后，你会看到类似于下图的编译之后的PHP可执行文件：

接下来，我们需要准备一个测试目录，具体结构如下：

```
C:\
- Research\
 -- admin\
     --- test.png
```

-- poc.php

准备一个poc.php文件，具体内容如下：

```
<?php
$a = "c:\\research\\phptest\\ad<\\test.png";
exec('pause');
if(@getimagesize($a)){
   echo "Success";
}else{
   echo "Failed";
}
?>
```

使用我们编译后的PHP来执行的话，正常情况下应该会返回Success：

一切准备就绪，便可以进行动态调试了。
先启动Visual Studio打开我们在静态分析时找到的\php-src\ext\standard\image.c文件在第1505行下一个断点。

进入C:\Research\目录，使用编译后的PHP(例如：C:\Research\php-sdk\phpdev\vc15\x64\php-7.2.1-src\x64\Debug_TS\php.exe)来执行我们的poc.ph
Studio里打开"调试-附加进程"来附加此处的PHP进程。

返回到执行poc文件的命令行下，敲击回车，我们发现前面设置的断点被成功hit了。

接下来的操作就需要特别的仔细和耐心了，使用VS提供的调试命令：

- F10: 单步调试
- F11: 逐语句调试
- Shift + F11： 跳出

经过以上一系列的调试，我们最终发现了PHP的getimagesize方法最终调用了Windows API里的FindFirstFileExW()，调用顺序如下：

- PHP_FUNCTION(getimagesize)
- php_getimagesize_from_any
- _php_stream_open_wrapper_ex
- php_stream_locate_url_wrapper
- wrapper->wops->stream_opener
- php_plain_files_stream_opener
- php_stream_fopen_rel
- _php_stream_fopen
- expand_filepath
- expand_filepath_ex
- expand_filepath_with_mode
- virtual_file_ex
- tsrm_realpath_r
- FindFirstFileExW

而根据StackOverflow上面的一个相关问题和MSDN的解释，这是NtQueryDirectoryFile / ZwQueryDirectoryFile通过FsRtlIsNameInExpression的一个功能特性，对于FsRtlIsNameInExpression有如下描述：

```
The following wildcard characters can be used in the pattern string.

Wildcard character  Meaning

* (asterisk)        Matches zero or more characters.

? (question mark)   Matches a single character.

DOS_DOT             Matches either a period or zero characters beyond the name
                    string.

DOS_QM              Matches any single character or, upon encountering a period
                    or end of name string, advances the expression to the end of
                    the set of contiguous DOS_QMs.

DOS_STAR            Matches zero or more characters until encountering and
                    matching the final . in the name.
```

另外，MSDN的解释并没有提到DOC-*具体指哪些字符，但根据ntfs.h，我们发现了如下的定义：

```
//  The following constants provide addition meta characters to fully
//  support the more obscure aspects of DOS wild card processing.

#define DOS_STAR        (L'<')
#define DOS_QM          (L'>')
#define DOS_DOT         (L'"')
```

因此，我们终于搞明白了为什么前言中说的这三个字符在Windows上被赋予了不同的含义了。

0x02 总结

通过以上分析，我们可以做以下的简短总结：

- 问题的产生的根本原因PHP调用了Windows API里的FindFirstFileExW()/FindFirstFile()方法
- 该Windows API方法对于这个三个字符做了特别的对待和处理
- 任何调用该Windows API方法的语言都有可能存在以上这个问题，比如：Python

0x03 参考

- https://stackoverflow.com/questions/24190389/findfirstfile-undocumented-wildcard-or-bug
- http://www.osronline.com/showThread.cfm?link=36720

上一篇：【企业安全实战】开源HIDS OS... 下一篇：移动安全入门教程（一）

1. 1 条追加内容

追加 于 2018年1月30日 20:46

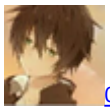
0x04 追加

最终我们通过动态调试在\php-src\Zend\zend_virtual_cwd.c的第841行找到了Window API里的FindFirstFileExW()方法:

```
#ifdef ZEND_WIN32
        if (save) {
            pathw = php_win32_ioutil_any_to_w(path);
            if (!pathw) {
                return -1;
            }
            hFind = FindFirstFileExW(pathw, FindExInfoBasic, &dataw, FindExSearchNameMatch, NULL, 0);
            if (INVALID_HANDLE_VALUE == hFind) {
                if (use_realpath == CWD_REALPATH) {
                    /* file not found */
                    FREE_PATHW()
                    return -1;
                }
                /* continue resolution anyway but don't save result in the cache */
                save = 0;
            } else {
                FindClose(hFind);
            }
        }
```

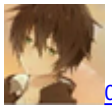
---

1. 5 条回复


0r3ak 2018-01-31 16:02:20

我在php5.X版本的代码包里面没找到
zend_virtual_cwd.c，也没有调用的FindFirstFileExW，只看到调用了tsrm_virtual_cwd.c里面的tsrm_realpath_r，这里的函数调用的是FindFirstFile类似的winapi，不知

0 回复Ta


0r3ak 2018-01-31 16:46:27

@0r3ak 问题的产生的根本原因PHP调用了Windows API里的FindFirstFileExW()/FindFirstFile()方法，这里应该是我看掉了的。

0 回复Ta


小鲜肉 2018-03-06 13:50:04

nice

0 回复Ta


kldbs 2018-03-31 21:49:54

楼主，今天我也刚刚研究了关于php 关于<<通配符的事
事情起因事前一段时间的dedecms 通过 通配符 爆后台
当时一直很困惑 为啥 < 可以当通配符用
后来一篇
PHPCMSv9逻辑漏洞导致备份文件名可猜测
文章里简单说了下是
'''
我们知道windows的FindFirstFile（API)有个特性就是可以把<<当成通配符来用而PHP的opendir(win32readdir.c)就使用了该API。PHP的文件操作函数均调用了opend
'''
后来自己也下了php源码分析来着。
然后晚上就看到楼主的文章了，点个赞

0 回复Ta

kldbs 2018-03-31 22:16:07

楼主，不知道能不能反过来
看到php哪些函数调用了FindFirstFile
具有类似的特性

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