

前言

最近在学习中学到了不少审计时的小技巧，都比较简单，但有一些在代码中比较容易出现的错误，这里做了一下总结分享，有错误希望师傅们斧正，一起交流学习。

for 循环中的 count

这个小技巧是从这里看到的：[关于CMSMS中SQL注入漏洞的复现与分析利用](#)

当 count 出现在 for 循环中，count 每次都会计算，如果这时候数组发生了变化，就会有一些差异。

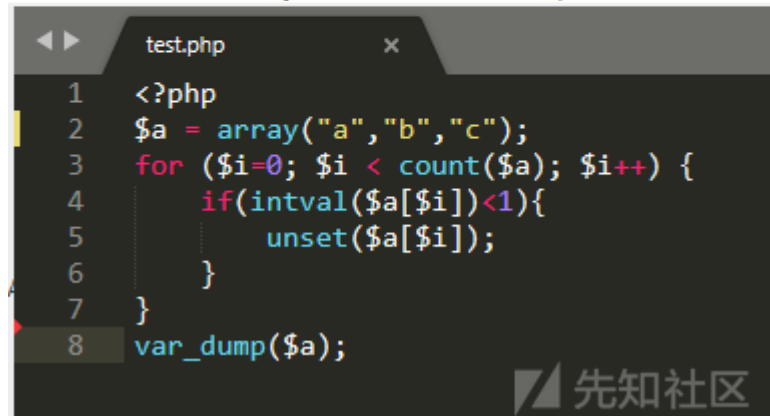
例子：

```
<?php
$a = array("a","b","c");
for ($i=0; $i < count($a); $i++) {
    if(intval($a[$i])<1){
        unset($a[$i]);
    }
}
var_dump($a);
```

```
C:\wamp64\www\learning\test.php:8:
array (size=1)
  2 => string 'c' (length=1)
```

test.php (learning) - Sublime Text (UNREGISTERED)

ew Goto Tools Project Preferences Help



分析

这里原本的目的是将数组中不是数字的值都删掉，我们来分析一下流程：

1. 首先取值 count(\$a)，此时为 3
2. 循环第一次，\$i=0，判断 \$a[0] 是否小于 1，返回 True，unset 掉，此时 \$a 为 ["b","c"]
3. 循环第二次，\$i=1，执行 count(\$a)，此时为 2，因为 1<2，进入循环，判断 \$a[1]，也就是 b 是否小于 1，返回 True，unset 掉，此时数组剩一个 \$a[2]，也就是 c
4. 循环第三次，\$i=2，执行 count(\$a)，此时为 1，因为 2>1，进不了循环了，自然就不能到 if 和 unset 这块。

(可能讲起来还是有点绕，可以自己试试就明白了)

思路扩展

比如下次在审计中，可以留意 for 循环中是否用了 count 这种重新计算的函数，并且在循环内操作了这个变量。

全局变量的覆盖

这是在翻 dedecms 往日漏洞时发现的，如果一个 CMS 注册全局变量时，用了这样的操作：

```
foreach($_REQUEST as $_k=>$_v)
{
    if( strlen($_k)>0 && preg_match('/^(cfg_|GLOBALS)/',$_k) && !isset($_COOKIE[$_k]) )
    {
        exit('Request var not allow!');
    }
}
foreach(Array('_GET','_POST','_COOKIE') as $_request)
{
    foreach($_$_request as $_k => $_v) $_$_k = $_v;
}
```

看起来这样好像控制了不能改 GLOBALS，但是这里却忽略了一个很重要的问题，就是没过滤 _POST（最新版本的 dedecms 是过滤了 _POST 和 _COOKIE 的）

这里怎么构造呢？我们可以控制 get 成这样的值：_POST[GLOBALS][xixi] = 1;

当执行检测时，由于 \$key 是 _POST，所以不会被检测，此时 post 参数是空，然后执行到下面。

执行 foreach 时，第一次循环，此时 \$_request 是 _GET，然后 \$_k 是 _POST，所以是 \$_POST['GLOBALS']['xixi'] = 1;

然后执行到第二次 foreach，此时 \$_request 为 _POST，\$_k 就是数组的键，自然就是 GLOBALS，所以执行的就是 \$_GLOBALS['xixi']=1

这样就覆盖了 \$GLOBALS 了。。

富文本编辑中的 XSS

bbcode，就是一种用短标签代替 html 标签的方法，一般是通过正则匹配替换的，通常出现在富文本编辑器中。

这个小技巧是从 [Mybb 18.20 From Stored XSS to RCE 分析](#) 这个漏洞中学到的。

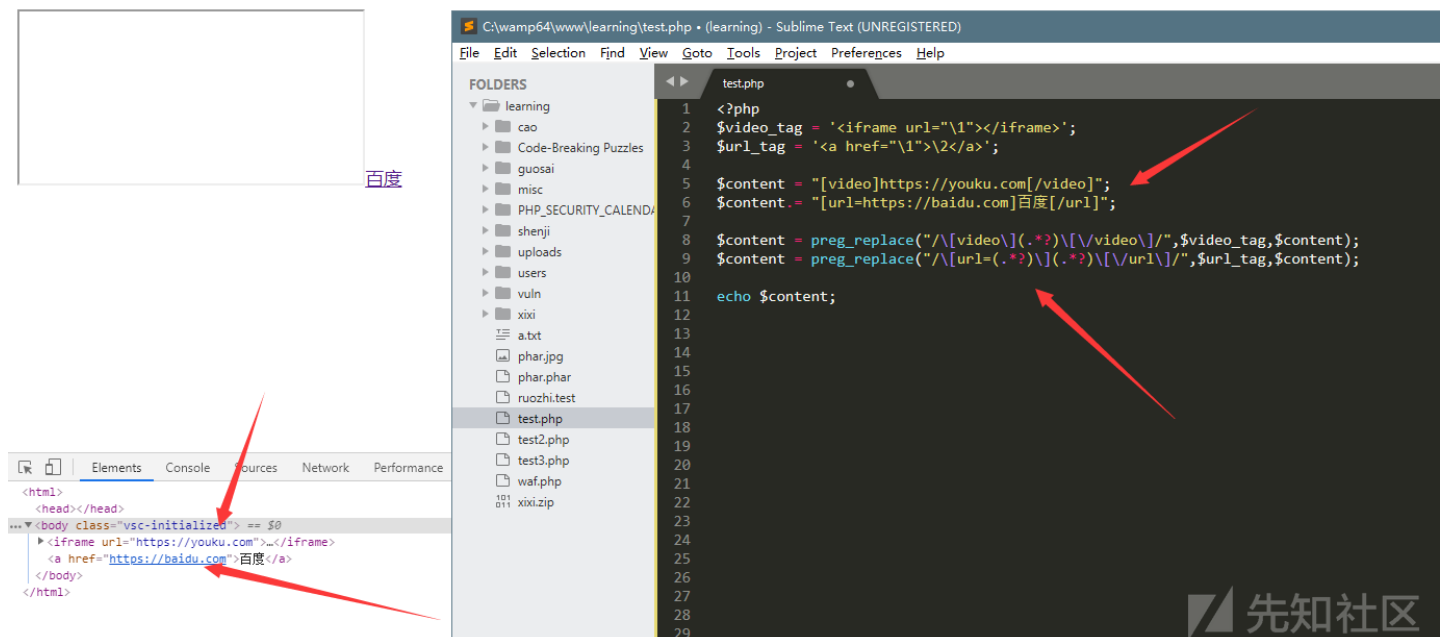
直接说技巧吧，当富文本编辑器中的双引号被转义了，可以考虑在标签中再嵌套标签。说起来比较难理解，直接上代码吧：

```
<?php
$video_tag = '<iframe url="\1"></iframe>';
$url_tag = '<a href="\1">\2</a>';

$content = "[video]https://youku.com[/video]";
$content.= "[url=https://baidu.com]百度[/url]";

$content = preg_replace("/\[video\](.*?)\[\/video\]/",$video_tag,$content);
$content = preg_replace("/\[url=(.*?)\](.*?)\[\/url\]/",$url_tag,$content);

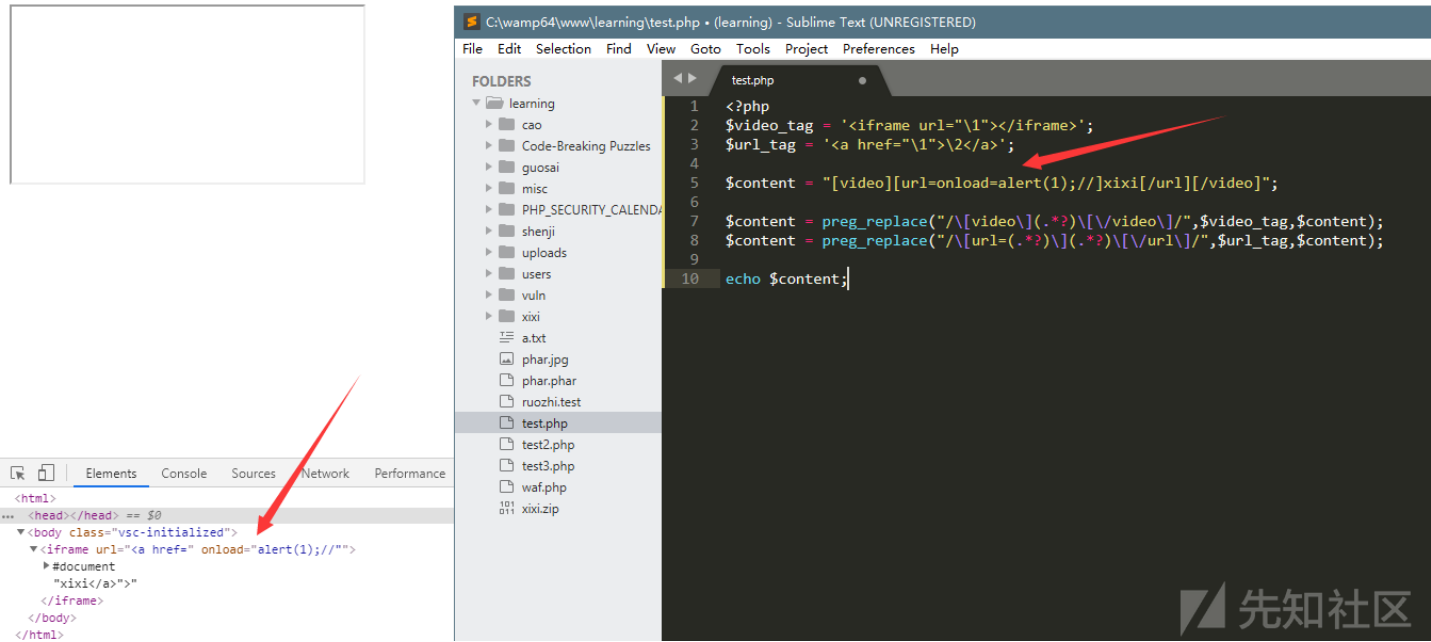
echo $content;
```



初衷当然就是利用一些正则替换代替标签，但是如果没做好过滤时，就可能发生这样的事情，如果我们的 \$content 变形一下：

```
[video][url=onload=alert(1);//]xixi[/url][/video]
```

看看这时候：



a 标签中的双引号闭合了 iframe 标签的，最后的双引号也可以用 // 注释。。

文件上传黑名单绕过

文件上传中的一些小技巧。

NTFS ADS

黑名单的话，什么 php3 4 5 6 7 这些就不提了。。

首先是前几天 lzl1y 表哥的文：[代码审计 xxxdisk前台Getshell](#) 也有提到的文件名为 a.php:\$data 时，可以绕过黑名单检测。

我记得之前有技巧是文件名为 a.php/. 也可以绕过的，但是测试了一下发现又不太行。。

也可以上传一个文件名是 a.php:b.jpg ，这时候文件内容会变成空，如果有文件操作的函数就可以用上这个技巧。

这里补一张图：

上传的文件名	服务器表面现象	生成的文件内容
Test.php:a.jpg	生成Test.php	空
Test.php::\$DATA	生成test.php	<?php phpinfo();?>
Test.php::\$INDEX_ALLOCATION	生成test.php文件夹	
Test.php::\$DATA.jpg	生成0.jpg	<?php phpinfo();?>
Test.php::\$DATA\aaa.jpg	生成aaa.jpg	<?php phpinfo();?>

图片出处：[我的WafBypass之道 \(Upload篇 \)](#)

.htaccess 与 getimagesize

当然如果实在想上传一个 .htaccess 也可以，如果他有 getimagesize 也不怕。

前几天 p牛就发过一篇博客：[imagemagick邂逅getimagesize的那点事儿](#)

因为 XBM 格式只要有

```
#define test_width 16
#define test_height 7
```

这两行就可以，又因为 # 在 .htaccess 里是注释，所以直接加在前两行都没问题。

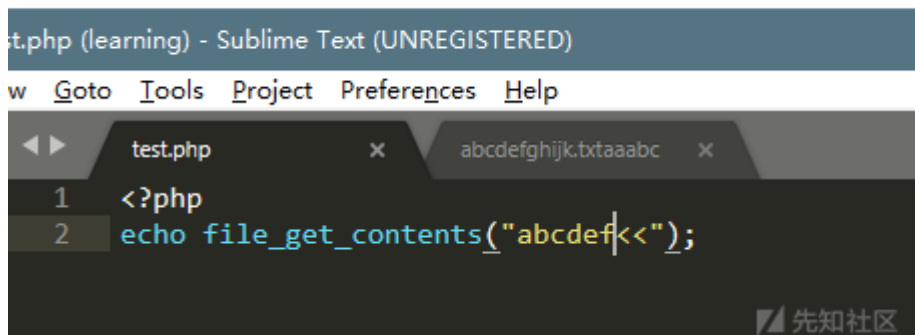
这里就再推一篇文：[Bypass file upload filter with .htaccess](#)

Win下php文件操作的特性

通配符

在 [PHPCMSv9逻辑漏洞导致备份文件名可猜测](#) 这篇文章中提到了大致，在 windows 下，php 操作文件时可以使用 << 通配符。

```
this a.txt
```



其实一个 < 也可以，但是测试时很奇怪，这里就不演示了。

短文件名

在 Windows 中还有一种表示短文件名的方法，虽然不够上面的方法灵活，但不算没用。。就是当文件名超过 6 位时，可以用 ~1 表示后面的字符串，比如我们的文件名叫 abcdefghijk.txt，短文件名表示成:abcdef~1.txt，如果这时候还有个叫 abcdefhhhh.txt 的，就可以把 1 改成 2：abcdef~2.txt

parse_str的一些特性

最近出了一篇文章，[Abusing PHP query string parser to bypass IDS, IPS, and WAF.](#)

里面提到了 parse_str 函数绕过 WAF，里面提到的很有趣，总结一下。

举个例子，当使用 parse_str 时，如果我们的字符串是 abc.123=1，或者 abc 123=1再或者 abc[123=1，最终都会被解析成 abc_123。

所以如果是先检测 \$_GET 然后再调用 parse_str，就会可能带来一些差异，导致绕过。

稍微修改了一下文中的 Fuzz 脚本：

```
<?php
foreach(
    [
        "{chr}foo_bar",
        "foo{chr}bar",
        "foo_bar{chr}"
    ] as $k => $arg) {

    for($i=0;$i<=255;$i++) {
        $o= Array();
        parse_str(str_replace("{chr}",chr($i),$arg). "=bla",$o);
        if(isset($o["foo_bar"])) {
            echo $arg." -> ".bin2hex(chr($i))." (".chr($i).")\n";
            echo "<br/>";
        }
    }
}
```

```
}  
}
```

```
}
```

点击收藏 | 3 关注 | 2

[上一篇：恶意样本家族分类实践](#) [下一篇：CVE-2019-0888：Win...](#)

1. 0 条回复
- 动动手指，沙发就是你的了！

[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](#)