# 奇安信攻防社区 – 微擎最新版前台某处无回显 SSRF 漏洞

奇安信攻防社区 – 微擎最新版前台某处无回显 SSRF 漏洞

## 微擎最新版前台某处无回显 SSRF 漏洞 ## 0x0 前言    [代码审计之某通用商城系统 getshell 过程] (https://mp.weixin.qq.com/s/rSP8LQJplkP-Ahljkof5sA), 续之前这篇文章 v1...

## 微擎最新版前台某处无回显 SSRF 漏洞

## 0x0 前言

代码审计之某通用商城系统 getshell 过程 (https://mp.weixin.qq.com/s/rSP8LQJplkP-Ahljkof5sA) , 续之前这篇文章 v1.8.2 版本，这次分享一个最新版 v2.7.6 相对来说比较鸡肋的无回显 SSRF，漏洞不是最主要的，主要是分享下自己的审计过程。

写文章还有补天的粽子领就很开心。

## 0x1 影响版本

经过测试应该是通杀到最新版的，不过不同版本利用方式有些不同，下面将从 v1.8.2 版本开始分析然后过渡到 v2.7.6 版本，来构造出对应的 POC。

## 0x2 漏洞点

v1.8.2 版本系统安装目录下的根目录文件: `api.php`

662 line: `analyzeImage` 函数, 直接将 `$message['picurl']` 传入 `ihttp_get` 函数, 结合前篇我们文章的分析，这个函数是采用了 `curl` 请求并设置跟随的, 如果我们可控 `$message['picurl']` 那么这里就会是一个支持任意协议，但是没回显的 SSRF。

这个漏洞可玩性与 UEditor SSRF (https://paper.seebug.org/606/) 差不多，不过这个属于 Blind 类型的。

```
private function analyzeImage(&$message) {
    load()->func('communication');
    if (!empty($message['picurl'])) {
        $response = ihttp_get($message['picurl']);
        if (!empty($response)) {
            $md5 = md5($response['content']);
            $event = pdo_get('menu_event', array('picmd5' => $md5), array('keyword', 'type'));
            if (!empty($event['keyword'])) {
                pdo_delete('menu_event', array('picmd5' => $md5));
            } else {
                $event = pdo_get('menu_event', array('openid' => $message['from']), array('keyword', 'type')
                    );
            }
            if (!empty($event)) {
                $message['content'] = $event['keyword'];
                $message['eventkey'] = $event['keyword'];
                $message['type'] = 'text';
                $message['event'] = $event['type'];
                $message['redirection'] = true;
                $message['source'] = $event['type'];
                return $this->analyzeText($message);
            }
        }
        return $this->handler('image');
    }
}
```

我们看一下，`$message` 是否可控

```
public function start() {
    global $_W;
    if(empty($this->account)) {
        exit('Miss Account.');
    }
    //var_dump($this->account);
    if(!$this->account->checkSign()) {
        exit('Check Sign Fail.');
    }
    if(strtolower($_SERVER['REQUEST_METHOD']) == 'get') {
        $row = array();
        $row['isconnect'] = 1;
        pdo_update('account', $row, array('acid' => $_W['acid']));
        cache_delete(cache_system_key('uniaccount', array('uniacid' => $_W['uniacid'])));
        exit(htmlspecialchars($_GET['echostr']));
    }
    if(strtolower($_SERVER['REQUEST_METHOD']) == 'post') {
        $postStr = file_get_contents('php://input');
                    if(!empty($_GET['encrypt_type']) && $_GET['encrypt_type'] == 'aes') {
            $postStr = $this->account->decryptMsg($postStr);
        }
        WeUtility::logging('trace', $postStr);
        $message = $this->account->parse($postStr);
```

可以看到在 `start` 函数里面获取了 POST 的内容然后进入 `$this->account->parse` 函数进行解析

251 line: 位于 `/framework/class/account/account.class.php` 的 `parse` 函数

201 行处，位于 `/framework/class/account/account.class.php` 的 `parse` 函数

```
public function parse($message) {
```

跟进 `x ml2array`，很简单就是解析 x ml 格式的内容，微擎官方文档 消息概述 (https://www.kancloud.cn/donknap/we7/134649) 里面就给出了使用案例。

```
function xml2array($xml) {
    if (empty($xml)) {
        return array();
    }
    $result = array();
    $xmlobj = isimplexml_load_string($xml, class_name: 'SimpleXMLElement', options: LIBXML_NOCDATA);
    if($xmlobj instanceof SimpleXMLElement) {
        $result = json_decode(json_encode($xmlobj), true);
        if (is_array($result)) {
            return $result;
        } else {
            return '';
        }
    } else {
        return $result;
    }
}
```

(https://shs3.b.qianxin.com/attack_forum/2021/08/attach-bb730e8b0b1bd1239431181e5b2fb8ba423a6705.png)

到这里就可以确定 `$message['picurl']` 是直接从 POST 的数据包中提取然后没有任何过滤进入到 `ihttp_get` 函数的，从而造成了 SSRF 漏洞的。

下面就是如何进行漏洞的触发。

## 0x3 触发漏洞

当我们访问 `http://localhost:8887/wq2/wq2/api.php`，要确保能走进漏洞函数，首先就要先进入到 `start()` 函数。

```
}
if(empty($id)) {
    $id = intval($_GPC['id']);
}
if (!empty($id)) {
    $uniacid = pdo_getcolumn( tablename: 'account', array('acid' => $id),  field: 'uniacid');
    $_W['account'] = uni_fetch($uniacid);
}
if(empty($_W['account'])) {
    exit('initial error hash or id');
}
if(empty($_W['account']['token'])) {
    exit('initial missing token');
}
$_W['debug'] = intval($_GPC['debug']);
$_W['acid'] = $_W['account']['acid'];
$_W['uniacid'] = $_W['account']['uniacid'];
$_W['uniaccount'] = uni_fetch($_W['uniacid']);
$_W['account']['groupid'] = $_W['uniaccount']['groupid'];
$_W['account']['qrcode'] = $_W['attachurl'].'qrcode_'.$_W['acid'].'.jpg?time='.$_W['timestamp'];
$_W['account']['avatar'] = $_W['attachurl'].'headimg_'.$_W['acid'].'.jpg?time='.$_W['timestamp'];
$_W['attachurl'] = attachment_set_attach_url();

    visit_update_today( type: 'web',  module_name: 'we7_api');


$engine = new WeEngine();
if (!empty($_W['setting']['copyright']['status'])) {
    $engine->died( content: '抱歉，站点已关闭，关闭原因：' . $_W['setting']['copyright']['reason']);
}
if (!empty($_W['uniaccount']['endtime']) && TIMESTAMP > $_W['uniaccount']['endtime']) {
    $engine->died( content: '抱歉，您的公众号已过期，请及时联系管理员');
}

if($_W['isajax'] && $_W['ispost'] && $_GPC['flag'] == 1) {
    $engine->encrypt();
}
if($_W['isajax'] && $_W['ispost'] && $_GPC['flag'] == 2) {
    $engine->decrypt();
}
load()->func('compat.biz');
$_W['isajax'] = false;
$engine->start();
```

绕过这里

要确保走到start

这里需要绕过前面判断，其实也很简单。

```
global $_W;
```

我们通过传入 `api.php?appid=wx570bc396a51b8ff8` ，便能成功构造出一个 `$_W['account']` 出来，绕过上面所说即如下的两个非空判断。

```
if (!empty($message)){
```

继续向下走，还需要绕过 `$this->account->checkSign()` ，继续跟进：

```
    public function start() {
        global $_W;  $_W: {config => [4], timestamp => 1623748378, charset => "
        if(empty($this->account)) {
            exit('Miss Account.');
        }
        //var_dump($this->account);
        if(!$this->account->checkSign()) {  account: WeixinPlatform
            exit('Check Sign Fail.');
        }
```

可以看到, 这个 Sign 其实是固定的，所需要的 3 个信息分别为 `$token, $_GET['timestamp'], $_GET['nonce']`，这里 `$token` 就是上面程序预留的信息值为： `platformtestaccount`，其他两个不传入留空值即可。

29 line: `f ramework/class/account/weixin.account.class.php` 的 `checkSign` 函数

```
public function checkSign() {
    $token = $this->account['token'];  account: [5]  $token: "platformtestaccount"
    $signkey = array($token, $_GET['timestamp'], $_GET['nonce']);  $token: "platformtestaccount"  $signkey: {null, null, "platformtestaccount"}[3
    sort( &array: $signkey,  sort_flags: SORT_STRING);
    $signString = implode($signkey);  $signkey: {null, null, "platformtestaccount"}[3]  $signString: "976a497ee3f68bc655ddcf4e7e7aab97d117ef0a"
    $signString = sha1($signString);
    return $signString == $_GET['signature'];  $signString: "976a497ee3f68bc655ddcf4e7e7aab97d117ef0a"
}
```

那么我们只要传入 `signature=976a497ee3f68bc655ddcf4e7e7aab97d117ef0a` 即可绕过 `checkSign` 函数。

然后回到 `api.php` 继续向下执行, 182 line, 对 `$message` 进行分析, 跟进该函数。

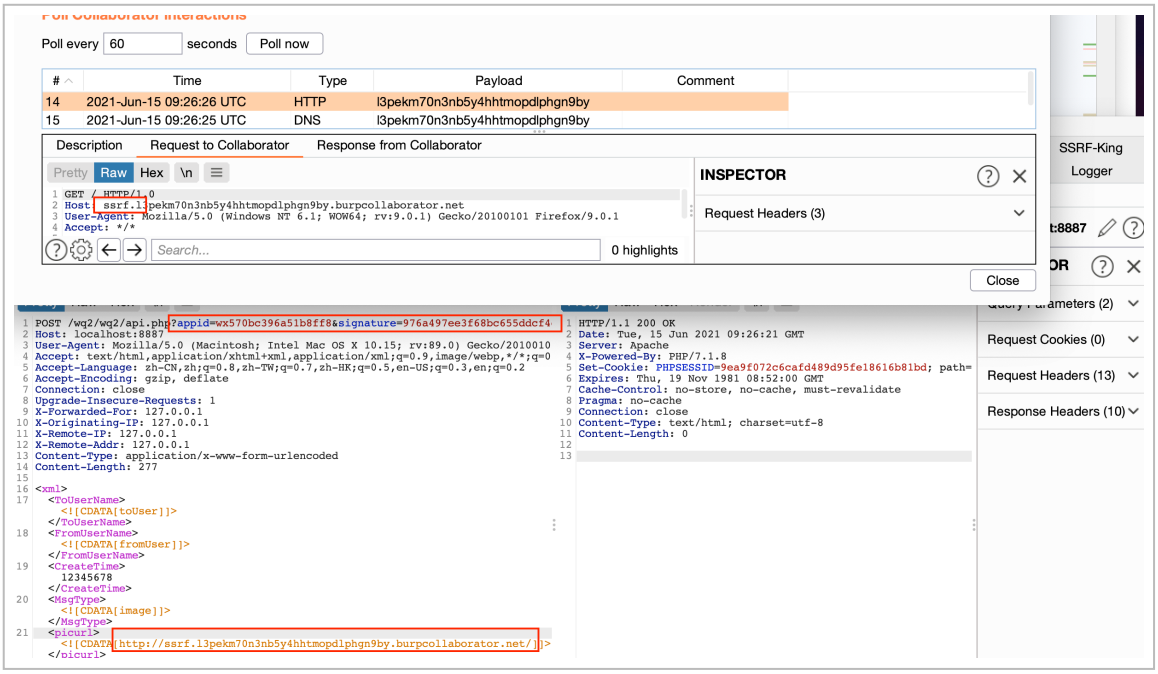
```
//解析内容
```

```
private function analyze(&$message) {  $message: {tousername => "toUser", fromusername => "fromUser", createtime => "12345678", msgtype => "image", picurl => "http://9hr2yalo1r1zjmi5vh0c31zdv41
    global $_W;  $_W: {config => [4], timestamp => 1623748378, charset => "utf-8", clientip => "127.0.0.1", ishttps => false, isajax => false, ispost => true, sitescheme => "http://", script_na
    $params = array();  $params: [0]
    if(in_array($message['type'], array('event', 'qr'))) {
        $params = call_user_func_array(array($this, 'analyze' . $message['type']), array(&$message));
        if(!empty($params)) {
            return (array)$params;
        }
    }
    if(!empty($_SESSION['__contextmodule']) && in_array($_SESSION['__contextmodule'], $this->modules)) {  modules: [16]
        if($_SESSION['__contextexpire'] > TIMESTAMP) {
            $params[] = array(
                'message' => $message,
                'module' => $_SESSION['__contextmodule'],
                'rule' => $_SESSION['__contextrule'],
                'priority' => $_SESSION['__contextpriority'],
                'context' => true
            );
            return $params;  $params: [0]
        } else {
            unset($_SESSION);  $_SESSION: {openid => "fromUser"}[1]
            session_destroy();
        }
    }

    $reply_times_info = (array)$_SESSION['__reply_times'];  $_SESSION: {openid => "fromUser"}[1]  $reply_times_info: [0]
    if (!empty($_W['account']['setting']) && !empty($reply_times_info) && intval($_W['account']['setting']['reply_setting']) > 0 && strtotime($reply_times_info['date']) >= strtotime(date( format
        exit('success');
    }

    if(method_exists($this,  method_name: 'analyze' . $message['type'])) {
        $temp = call_user_func_array(array($this, 'analyze' . $message['type']), array(&$message));  $message: {tousername => "toUser", fromusername => "fromUser", createtime => "12345678", msg
        if(!empty($temp) && is_array($temp)){
            $params += $temp;
        }
    } else {
        $params += $this->handler($message['type']);
```

这里进行了函数拼接，动态调用analyze_image函数

最终就会进入我们上述漏洞点 `analyzeImage` 函数，造成 SSRF。

## 0x4 POC 验证

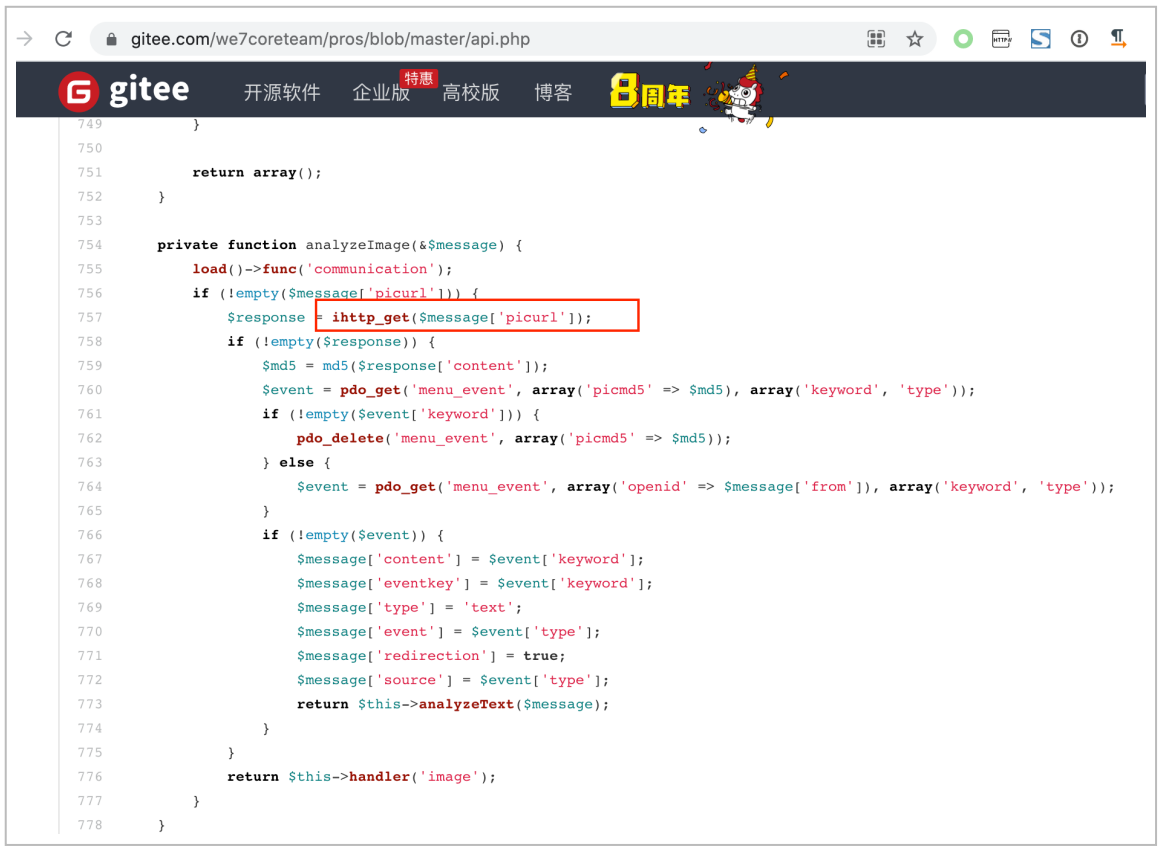



可以看到构造如下格式，便可成功触发。

```
$message = x ml2array($message);
```

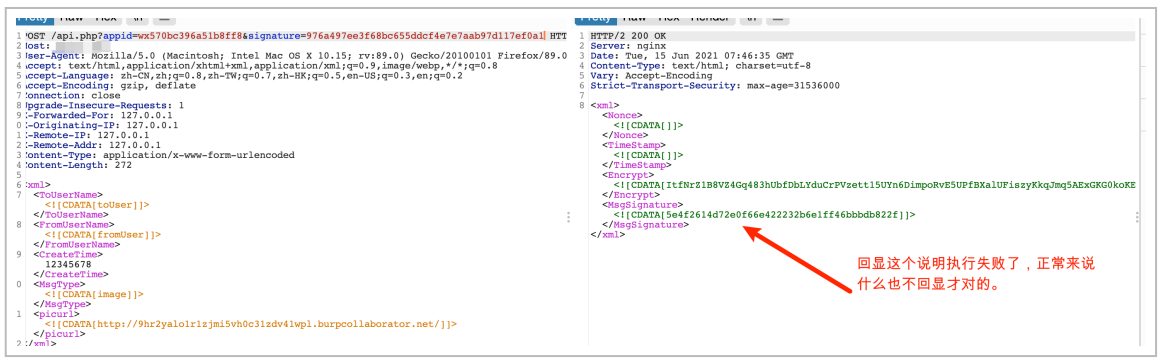
# 0x5 出现问题

我简单看了一下 Gitee 上该系统的最新版 2.7.6 的代码 api.php (https://gitee.com/we7coreteam/pros/blob/master/api.php)，发现漏洞点还是存在的。




```php
        }

        return array();
    }

    private function analyzeImage(&$message) {
        load()->func('communication');
        if (!empty($message['picurl'])) {
            $response = ihttp_get($message['picurl']);
            if (!empty($response)) {
                $md5 = md5($response['content']);
                $event = pdo_get('menu_event', array('picmd5' => $md5), array('keyword', 'type'));
                if (!empty($event['keyword'])) {
                    pdo_delete('menu_event', array('picmd5' => $md5));
                } else {
                    $event = pdo_get('menu_event', array('openid' => $message['from']), array('keyword', 'type'));
                }
                if (!empty($event)) {
                    $message['content'] = $event['keyword'];
                    $message['eventkey'] = $event['keyword'];
                    $message['type'] = 'text';
                    $message['event'] = $event['type'];
                    $message['redirection'] = true;
                    $message['source'] = $event['type'];
                    return $this->analyzeText($message);
                }
            }
            return $this->handler('image');
        }
    }
```

但是我在网上找了几个最新版的站打了下，发现并没有成功。




尝试删减一些参数，可以得到原因是没进入 `start` 函数就结束了，通过 debug 发现问题主要是在

在初始化 `$this->account = WeAccount::create($_W['account']);` 时会调用到这个 `getAccountInfo` 函数，这里对内置的测试用户做了个判断，导致进入了 `$this->openPlatformTestCase();` 而这个函数最终都是走入了 `exit()` ，所以这里我们不能使用这个账户。

```
$packet = iarray_change_key_case($message, CASE_LOWER);
```

## 0x6 解决问题

回到 `api.php`

```
$hash = $_GPC['hash'];
if(!empty($hash)) {
    $id = pdo_fetchcolumn( sql: "SELECT acid FROM " . tablename( table: 'account') . " WHERE hash = :hash", array(':hash' => $hash));
}
if(!empty($_GPC['appid'])) {
    $appid = ltrim($_GPC['appid'], charlist: '/');
    if ($appid == 'wx570bc396a51b8ff8') {
        $_W['account'] = array(
            'type' => '3',
            'key' => 'wx570bc396a51b8ff8',
            'level' => 4,
            'token' => 'platformtestaccount'
        );
    } else {
        $id = pdo_fetchcolumn( sql: "SELECT acid FROM " . tablename( table: 'account_wechats') . " WHERE `key` = :appid", array(':appid' => $appid));
        if (empty($id)) {
            $id = table( name: 'account_wxapp')->where( condition: 'key', $appid)->getcolumn( field: 'acid');
        }
    }
}
if(empty($id)) {
    $id = intval($_GPC['id']);
}
if (!empty($id)) {
    $uniacid = pdo_getcolumn( tablename: 'account', array('acid' => $id),  field: 'uniacid');
    if (visit_app_pass_visit_limit($uniacid)) {
        exit('success');
    }
    $_W['account'] = $_W['uniaccount'] = uni_fetch($uniacid);
}
if(empty($_W['account'])) {
```

可以看到除了测试用户，我们也可以通过传入 `$id` 来获取 account，跟进 `uni_fetch` 函数。

```
 * 获取指定统一公号下默认子号的的信息
 * @param int $uniacid 公众号ID
 * @return array 当前公众号信息
 */
function uni_fetch($uniacid = 0) {  $uniacid: 1
    global $_W;  $_W: {config => [4], timestamp => 1623753791, charset
    $uniacid = empty($uniacid) ? $_W['uniacid'] : intval($uniacid);
    $account_api = WeAccount::createByUniacid($uniacid);  $uniacid: 1
    if (is_error($account_api)) {
        return $account_api;
    }
    $account_api->__toArray();
    $account_api['accessurl'] = $account_api['manageurl'] = wurl( segn
    $account_api['roleurl'] = wurl( segment: 'account/post-user/edit',
    return $account_api;
}
```

查询 account 获取 id=1 的信息

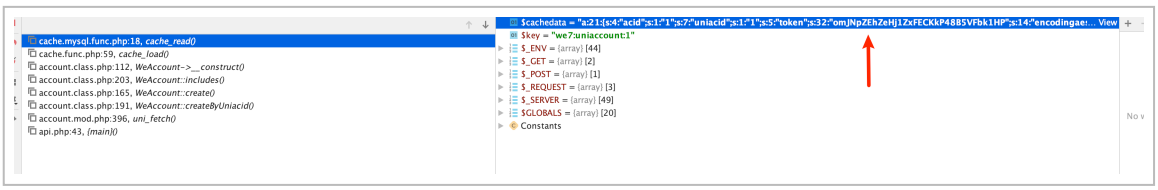
```
public static function createByUniacid($uniacid = 0) {  $uniacid: 1
    global $_W;  $_W: {config => [4], timestamp => 1623753791, charset => "utf-8", clientip => "127.0.0.1", ishttps
```

```
        $uniacid = intval($uniacid) > 0 ? intval($uniacid) : $_W['uniacid'];
        if (!empty(self::$accountObj[$uniacid])) {
            return self::$accountObj[$uniacid];
        }
        $uniaccount = table( name: 'account')->getUniAccountByUniacid($uniacid);  $uniaccount: {acid => "1", uniacid =>
        if (empty($uniaccount)) {
            return error( errno: '-1',  message: '帐号不存在或是已经被删除');
        }
        load()->model( name: 'permission');
        if (!empty($_W['uid']) && !$_W['isadmin'] && !permission_account_user_role($_W['uid'], $uniacid)) {  $_W: {conf
            return error( errno: '-1',  message: '无权限操作该平台账号');
        }
        return self::create($uniaccount);  $uniaccount: {acid => "1", uniacid => "1", hash => "uRr8qvQV", type => "1",
    }
```

继续跟下去，最终你会发现 token 其实存储在了 ims_core_cache 表中，并且只有唯一一个，这个 Token 值是固定的。

> 这个信息是从 `/data/db.php` 获取的，也就是初始化的默认数据，刚好这个值不是随机生成的，所有版本都是一样的。

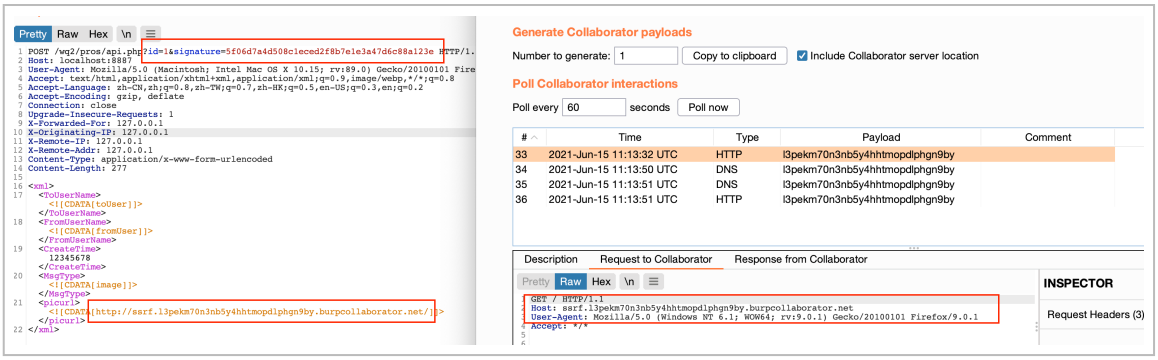
相关调用栈如下:




所以我们只要重新获取一下 signature 就行了，即如下

```
    public function checkSign() {
        $token = $this->account['token'];  account: [37]  $token: "omJNpZEhZeHj1ZxFECKkP48B5VFbk1HP"
        $signkey = array($token, $_GET['timestamp'], $_GET['nonce']);  $token: "omJNpZEhZeHj1ZxFECKkP48B5VFbk1HP"  $signkey: {null, null, "omJNpZEhZeHj1ZxFECKkP48B5VFbk1HP"}[3]
        sort( &array: $signkey,  sort_flags: SORT_STRING);
        $signString = implode($signkey);  $signkey: {null, null, "omJNpZEhZeHj1ZxFECKkP48B5VFbk1HP"}[3]  $signString: "5f06d7a4d508c1eced2f8b7e1e3a47d6c88a123e"
        $signString = sha1($signString);

        return $signString == $_GET['signature'];  $signString: "5f06d7a4d508c1eced2f8b7e1e3a47d6c88a123e"
    }
```

# 0x7 新 POC



## 0x8 总结

本文回顾了以前的文章，在此基础上对新版本进行类似漏洞的挖掘，遇到了版本差异导致的问题，尝试解决的时候，发现了关键的检验参数 Token 存在默认值，导致可以直接构造，完成了利用。最后，关于临时修复方案，账户是可以在后台进行删除的，步骤分别是”所有平台”–> 放入回收站 –> 彻底删除，这样就可以避免猜测到 Token 值。