WARNING

Вся информация предоставлена исключительно в ознакомительных целях. Ни редакция, ни автор не несут ответственности за любой возможный вред, причиненный материалами данной статьи.


Алексей Осипов
@GiftsUngiven


Михаил Фирстов
@cyberpunkych


Денис Баранов
@dsbaranov


Юрий Гольцев
@ygoltsev

БОЕВОЙ ХОНИПОТ ИЗ БАЗЫ ДАННЫХ

Сногсшибательный вектор атак на клиенты MySQL

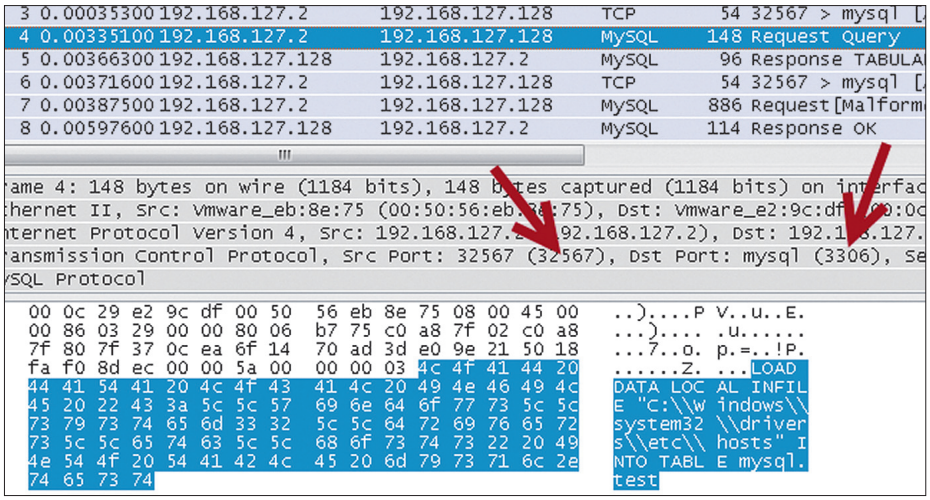
База данных — лакомый кусочек для хакера. Но атакующий в погоне за информацией в СУБД сам может стать жертвой. Из-за лазейки, оставленной разработчиком, клиент вместо того, чтобы прочитать данные из базы, сам того не подозревая, может передать на сервер произвольный файл со своей системы! Как такое может быть?

КРАТКОЕ СОДЕРЖАНИЕ ДЛЯ САМЫХ НЕТЕРПЕЛИВЫХ

Теплый июньский день перед выходным. Ничто не предвещало беды, и тут некто Денис Баранов совместно с Михаилом Фирстовым и Алексеем Осиповым находят 0-day-фичу в реализации SQL-оператора LOAD DATA LOCAL INFILE и пишут под нее сплоит.

ТЕПЕРЬ ПО ПОРЯДКУ

Все началось с того, что в Сети обнаружился забавный хак-квест. В одном из уровней необходимо было влиять на строку соединения к базе данных. Идея понятная: если подменить адрес СУБД, то MySQL-клиент подключится не к серверу разработчика, а к серверу, контролируемому хакером. А значит, для обхода авторизации достаточно поднять свой MySQL-сервер и подменить таблицу с пользователями на свою. Тема не новая: подробнее об атаках с изменением строки соединения можно прочитать, к примеру, на rdot.



Клиент отправляет запрос на сервер #1

org (bit.ly/14rYQA9). Но появилась идея: если человек сам соединяется с твоим сервером, почему бы не найти вариант получения доступа к хосту, с которого он осуществляет соединение?

Есть идея — нужно пробовать! На ум сразу пришел замечательный оператор LOAD DATA LOCAL, который позволяет пользователю базы данных считать файлы со своей локальной системы и отправить их на сервер базы данных. Удобная функция, чтобы не приходилось писать скрипты или загружать файлы на сервер. И более того, она не требует повышенных привилегий, так как считается, что клиент читает собственные файлы. Найти бы вариант выполнить команду LOAD DATA LOCAL INFILE, причем без желания клиента.

В первую очередь изучаем материалы по теме, например на rdot.org (bit.ly/yqUzYw). Помимо описанных трюков, также можно попробовать поиграться с триггерами, но в случае select-запросов они бесполезны.

В итоге для решения задачи нужно представить себе логику программиста, разрабатывающего СУБД. Для того чтобы выполнить запрос, его необходимо распарсить. После этого можно выполнить нужные команды и отдать данные клиенту. Соответственно, запрос с оператором LOAD DATA LOCAL тоже должен где-нибудь разобраться. Варианта всего два: на сервере или на клиенте, но второй случай мы сразу отбрасываем, так как это было бы абсолютно неправильно с точки зрения архитектуры. Значит, оператор все-таки обрабатывается на сервере, после чего клиенту, видимо, отправляет служебное сообщение «Отдай мне файл с таким-то именем». Клиент получает сообщение и отдает запрошенные данные серверу.

А теперь самое главное! А что, если на любой SQL-запрос от клиента отвечать служебным сообщением «Отдай мне файл»? :)

ПРАКТИКА

В обычном случае взаимодействие с БД выглядит так: мы авторизуемся → если авторизация успешна, отправляем запрос → получаем результат. Последние два пункта можно повторять сколько угодно раз. Однако для LOAD DATA LOCAL INFILE все немного иначе. Обрати внимание на IP-адреса и порты, с которых приходят запросы:

1. Предположим, мы уже авторизованы и посылаем запрос LOAD DATA LOCAL INFILE "C:\\Windows\\system32\\drivers\\etc\\hosts" INTO TABLE mysql.test.
2. Сервер отвечает нам неким пакетом, содержащим имя файла, которое было передано клиентом.
3. Клиент отправляет содержимое файла на сервер.

Странное поведение, не находишь? Сразу бросается в глаза, что всю эту последовательность можно сократить до одного шага — шага номер 3. Почуяв неладное, мы решили разобраться. И немного заморочившись с настройками проксирования, получили неожиданный результат.

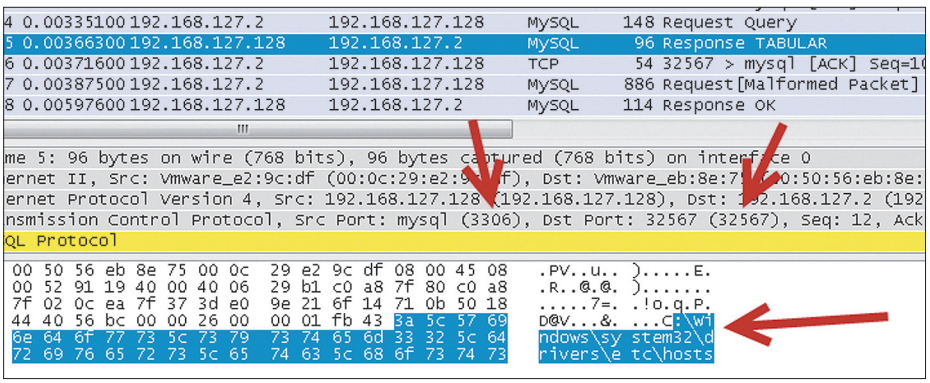
Если на произвольный запрос MySQL-клиента мы отправим пакет #2, клиент сразу отправит нам содержимое файла независимо от того, какой изначально был запрос!

Сервер просит файл? Дадим ему файл!

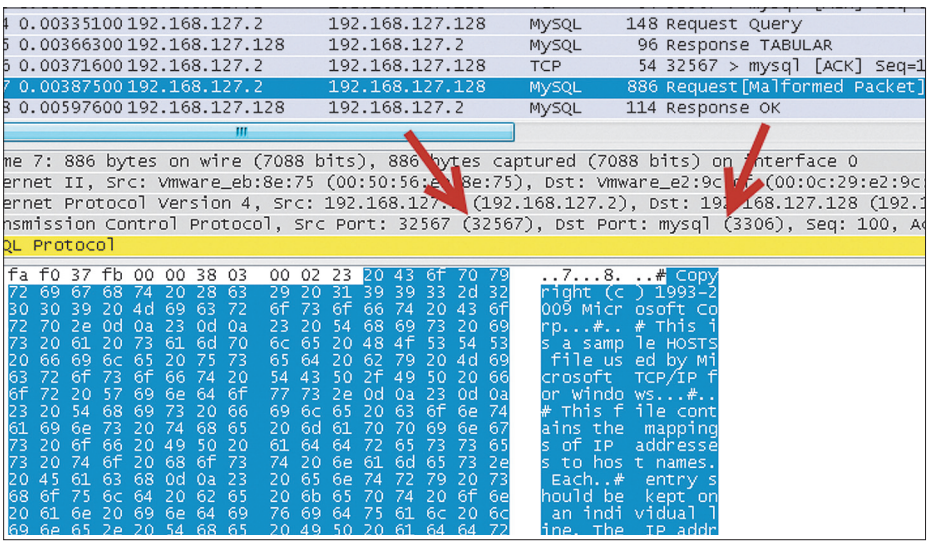
ПРОЧТИ ФАЙЛ! ПОЖАЛУЙСТА

В срочном порядке был набросан небольшой сплоит на питоне с гордым названием rogue_

> ***В результате простейший скрипт на Python'е, выдающий себя за MySQL-сервер, может прочитать любой файл с подключающего клиента. Разве это не прекрасно?***



Ответ сервера на запрос LOAD DATA LOCAL #2



Контент файла летит на сервер #3

```
MySQL Protocol
    Packet Length: 61
    Packet Number: 0
  Server Greeting
      Protocol: 10
      Version: 5.1.66-0+squeeze1
      Thread ID: 54
      Salt: evilsalt
    Server Capabilities: 0xF7DF
        .... .... .... ...1 = Long Password: Set
        .... .... .... ..1. = Found Rows: Set
        .... .... .... .1.. = Long Column Flags: Set
        .... .... .... 1... = Connect With Database: Set
        .... .... ...1 .... = Don't Allow database.table.column: Set
        .... .... ..0. .... = Can use compression protocol: Not set
        .... .... .1.. .... = ODBC Client: Set
        .... .... 1... .... = Can Use LOAD DATA LOCAL: Set
        .... ...1 .... .... = Ignore Spaces before '(': Set
        .... ..1. .... .... = Speaks 4.1 protocol (new flag): Set
        .... .1.. .... .... = Interactive Client: Set
        .... 0... .... .... = Switch to SSL after handshake: Not set
        ...1 .... .... .... = Ignore sigpipes: Set
        ..1. .... .... .... = Knows about transactions: Set
        .1.. .... .... .... = Speaks 4.1 protocol (old flag): Set
        1... .... .... .... = Can do 4.1 authentication: Set
      Charset: latin1 COLLATE latin1_swedish_ci (8)
    Server Status: 0x0002
      Unused:
      Salt: otheaarsal
```

Бит, говорящий о возможности использования LOAD DATA LOCAL

mysql. Весь его функционал заключался в том, что он принимает подключения от MySQL-клиента (причем неважно, какой логин и пароль используется для подключения) и на любой запрос клиента отправляет запрос на чтение определенного файла. Общение выглядит следующим образом:

1. Клиент выполняет запрос SELECT * FROM mysql.user (1).
2. Сервер отвечает: «А прочитай-ка лучше свой файл c:/boot.ini» (2).
3. Прочитать файл? Ну лааадно, держи (3).

В результате простейший скрипт на Python'е, выдающий себя за MySQL-сервер, может прочитать любой файл с подключающего клиента (как в данном случае boot.ini). Разве это не прекрасно?

IN THE WILD

Способ запуска: python rogue_mysql.py. По дефолту читается случайный файл из списка в самом начале скрипта.

Где может пригодиться? К примеру, теперь могут пригодиться установочные скрипты вордпресса или phpMyAdmin с нерабочими базами MySQL. Обращаемся на наш сервер и получаем полноценную читалку файлов в контексте уязвимого клиента. Ниже представлен небольшой пример для наглядности:

```
<?php
  $conn = mysql_connect↵
  ($_GET['mysql_host_port'],↵
  'root', '12345');
  mysql_query('SELECT * ↵
  FROM mysql.user');
?>
```

Или другой вариант: можно реализовать небольшой honeypot, в котором файлы будут читаться с сервера брутера/атакующего. Это вообще может получиться забавный эксперимент с интересными результатами. Так, оставив работающим скрипт rogue_mysql.py, мы насобирали порядочное количество файлов hosts, принадлежащих нашим серверам, с которых мы тестировали брутфорс :).

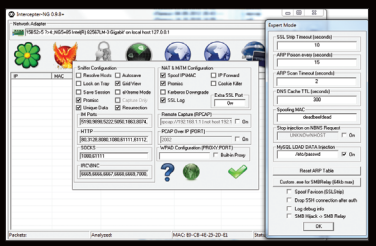
НЕМНОГО ДЕГТЯ

Уязвимость интересная, но, к сожалению (или к счастью?), не все MySQL-клиенты по умолчанию собраны с флагом, отвечающим за возможность исполнения функционала LOAD DATA LOCAL. Это серьезное ограничение (парни сильно расстроились, когда так и не смогли пробить ни мой GUI-клиент на Mac'е, ни стандартный MySQL-клиент, запущенный на Ubuntu; видимо, в гости в офис PT больше не пустят. — Прим. главреда).

Выяснить, поддерживает ли клиент этот функционал или нет, можно по данным первого пакета, пришедшего серверу от клиента, в секции Server Greeting. Пример данных из такого пакета представлен на рисунке.

НОВАЯ ФИШКА INTERCEPTER-NG

Естественно, эту особенность клиентов MySQL вполне можно использовать при MITM-атаках в процессе проведения тестирования на проникновение. Не надо ждать, пока клиент сам выполнит оператор LOAD DATA LOCAL, — достаточно просто оказаться между ним и сервером :). Ares, автор тулкита Intercepter (intercepter.nerf.ru), заинтересовался нашей находкой и с радостью добавил функционал для реализации подобной атаки. Официально эта функция станет доступной после релиза нового билда. Но если не терпится, ты всегда можешь попросить у автора сборку, включающую в себя этот новый функционал.



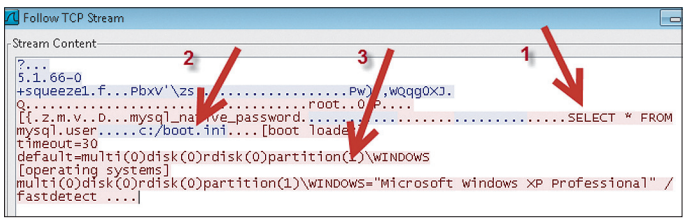
Новый функционал в Intercepter-NG

ВМЕСТО ЗАКЛЮЧЕНИЯ

По правде говоря, оператор LOAD DATA LOCAL — это уже сама по себе интересная штука, которая не раз помогала нам при проведении тестов на проникновения, еще до момента, когда мы нашли возможность для выполнения обратной атаки.

Немного покопавшись в документации, мы обнаружили, что разработчики частично в курсе этой особенности. Так, на официальном сайте сказано, что теоретически между клиентом и сервером MySQL может вмешаться третье лицо и модифицировать запрос LOAD DATA LOCAL, в результате в базу будет записан совершенно другой файл. Но такая ситуация маловероятна. А вот что действительно реально, так это описанная сегодня концепция хонипота.

Как ты видишь, тема атаки на клиенты со стороны сервера на данный момент раскрыта не полностью. И вполне реально обнаружить подобные забавные особенности и для других сервисов, доступных по другим портам. Помни, хакер: подключился к чужому серверу — будь осторожен! Атаковать тебя может и легальный владелец ресурса!



Сливаем boot.ini

```
127.0.0.1       localhost
127.0.1.1       bt.foo.org      bt

# The following lines are desirable for IPv6 capable hosts
::1     localhost ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
ff02::3 ip6-allhosts
```

Файл атакующего