УДК 004

УЯЗВИМОСТЬ NOSQL НА ПРИМЕРЕ MONGODB

Абрамова Ирина, Афанасьева Ирина

Харьковский национальный университет Радиоэлектроники

Аннотация

В данной публикации рассматриваются вопросы незащищенности нереляционных баз данных на примере MongoDB. Многие полагают, что нереляционные базы данных имеют значительное преимущество, в сравнении с РСУБД, будь-то они являются прочнее из-за отсутствия SQL и, соответственно, связанных с ним инъекций. Тем не менее нереляционные базы данных имеют целый ряд специфических незащищенностей, как в них самих, так и в системе управления, что делает нереляционные базы данных не такими безопасными, как кажется на первый взгляд.

Abstract

This publication examines the issues of insecurity of non-relational databases using MongoDB as an example. Many believe that non-relational databases have a significant advantage, in comparison with RDBMS, whether they are stronger because of the lack of SQL and, consequently, related injections. Nevertheless, non-relational databases have a number of specific insecurities, both in themselves and in the management system, which makes non-relational databases not as secure as they seem at first glance.

Введение

Понятие NoSQL (Not Only SQL или No SQL), то есть уход от использования привычных нам уже РСУБД , получил популярность с 2009 года. На тот момент развитие web-технологий и социальных сервисов требовало нахождения новых подходов к хранению и обработке данных. Началось активное развитие и продвижение всевозможных программных решений с использованием NoSQL. Близкие по значению слова для NoSQL стали огромные объемы данных, линейная масштабируемость, кластеры, отказоустойчивость, нереляционность. Разработчики столкнулись с задачами, для которых традиционные реляционные СУБД оказались либо слишком дороги, либо недостаточно производительны. Кроме того, основателями отказа от универсальных «комбайнов» (реляционные СУБД РСУБД) в пользу специализированных решений стали молодые проекты и те, кому приходится работать в сценариях так называемых Big Data [1].

Можно допустить, что NoSQL СУБД безопасней, благодаря тому что в них отсутствуют SQL-запросы и нет возможности провести SQL-инъёкцию. Частично это так, но если в запросе отсутствует SQL-код, это еще не говорит о том, что система полностью безопасна. NoSQL прикрывает лишь некоторые уязвимости, такие как SQL-инъекции, но оставляет множество других.

Привычно она состоит из трех уровней [3]:

1. Приложение;
2. API базы данных NoSQL;
3. NoSQL-СУБД.

Все вышеперечисленные уровни потенциально уязвимы. Рассмотрим с самого нижнего уровня, иначе с самой СУБД. Как и любое приложение, СУБД может быть подвластна атакам переполнения буфера или иметь незащищенную схему аутентификации. Атаковать этот уровень сложно, потому что за подобными слабыми местами следит сообщество и разработчики.

Второй уровень — API. Многие нереляционные СУБД имеют большое количество библиотек для организации доступа к данным. Большинство таких проектов имеют открытый код, но часть из них уже не поддерживаются, поэтому шанс найти уязвимость на этом уровне гораздо выше.

Высший уровень — атакуемый. Здесь нужно попытаться найти места, в которых разработчик забыл проверить входные данные. При таком обстоятельстве используется тот же подход, что и при поиске SQL-инъекций, однако в данной ситуации придется разбираться с JSON, JavaScript или чем-то подобным.

Рассмотрим слабые места нижнего уровня на основе анализа нескольких версий MongoDB [3]:

1. Незащищенное место в системе аутентификации. Стандартно БД устанавливается без пароля . Учредители MongoDB думают, что БД запускается в полностью доверенном окружении.

2. Незащищенное место в системе авторизации. Всякий созданный пользователь как принято имеет выход на чтение всей базы данных, то есть он имеет доступ ко всему, что у нас имеется.

3. Незащищенное место в системе авторизации администратора . Юзер с доступом к базе данных Admin имеет права на чтение/запись откуда/куда угодно. А т.к. стандартно нет пароля, то мы имеем доступ везде.

4. Нешифрованный текст. Все данные передаются в открытом виде, то есть они могут быть перехвачены любой MITM—атакой.

Большинство современных реляционных СУБД позволяют создавать хранимые процедуры на сервере. У MongoDB имеются аналогичные возможности, среди которых присутствует серверный JavaScript. Это позволяет исполнять почти любой код на сервере БД. С одной стороны, это даёт возможность писать алгоритмы обработки данных, но с другой — делает приложение более уязвимым.

Области применения JavaScript в MongoDB:

1. Запросы с оператором $where.

2. Команда db.eval.

3. Функции для сохранения в базе данных. Для этого используется специальная системная коллекция system.js.

4. Map/Reduce. Это – программный фреймворк, разработанный компанией Google для параллельных вычислений над большими объемами данных.

В данной работе мы рассмотрели уязвимость РСУБД и NOSQL, можем сделать выводы, что на уровне самой БД при стандартных настройках реализации NoSQL (а именно — MongoDB) серьезно проигрывают классическим РСУБД. Однако, нужно заметить, что стандартные настройки не используются на боевых серверах, поэтому, в большинстве случаев, если уязвимость есть только при стандартных настройках, то в серьёзных проектах её не будет. А вот на уровне приложения ситуация обстоит гораздо лучше. Многие незащищенности основаны на очевидных ошибках в коде и существуют только в примерах, а на практике встретить их практически не представляется возможным.

Список использованных источников:

1. СУБД NoSQL – сильные и слабые стороны. [Электронный ресурс]. Режим доступа: http://www.jetinfo.ru/stati/silnye-i-slabye-storony-nosql/

2. Азбука NoSQL-инъекций. [Электронный ресурс]. Режим доступа: https://habrahabr.ru/company/xakep/blog/143909/

3. Mongodb - Security Weaknesses in a typical NoSQL database. [Electronic resource]. URL:https://www.trustwave.com/Resources/SpiderLabs-Blog/Mongodb---Security-Weaknesses-in-a-typical-NoSQLdatabase/