УДК 004.056.53

Р.Х. Хамдамов, К.Ф. Керимов

МЕТОД ЗАЩИТЫ БАЗЫ ДАННЫХ НА ОСНОВЕ БРАНДМАУЭРА ВЕБ-ПРИЛОЖЕНИЙ

Ключевые слова: информационная безопасность, веб-приложение, база данных, брандмауэр, аномальные запросы, SQL-выражения.

Введение

Веб-приложения все больше используются при чтении новостей, оплате счетов и онлайн-покупках. По мере роста этих услуг можно видеть увеличение количества и степени атак на них, таких как: кража персональной информации, банковских данных и другие случаи киберпреступлений. Все вышеперечисленное является следствием открытости информации в базе данных (БД). Безопасность веб-приложения сильно зависит от безопасности БД [1–6].

Данные запросов клиентов обычно извлекаются набором запросов, которые запрашивают пользователя приложения. Если вводимые пользователем данные проверяются не очень тщательно, можно собрать целое множество видов атак, использующих веб-приложения для создания угроз с безопасностью для БД. К сожалению, из-за временных ограничений программисты веб-приложений обычно фокусируются на функциональности веб-приложений, но лишь немногие беспокояться о безопасности.

Например, электронный ресурс может пренебрегать строгой валидацией и введенными ограничениями вводимых пользователем данных, что дает взломщикам возможность использовать этот изъян: менять структуру запроса и текущее значение запроса с помощью ошибок в вводимых данных. Если это случится, взломщики могут без авторизации манипулировать БД, красть секретную информацию о пользователях. Следствием всего этого может быть разрушение БД, утечка информации и т.д., что влечет серьезные последствия.

Разработка метода защиты БД на основе брандмауэра веб-приложений

Разработанный брандмауэр баз данных служит для защиты от атак на БД веб-приложений. Он работает как прокси, это означает, что принимаемые от клиента запросы на SQL-выражения сначала отправляются на разработанный брандмауэр (БМ), а не на сервер БД. Брандмауэр анализирует запрос: запросы, считающиеся странными, блокируются БМ и клиенту возвращается пустой результат. В противном случае он вызовет сервер БД для выполнения запроса.

На рис.1 изображена архитектура БМБД. По существу, это виртуальный драйвер соединения с БД. По умолчанию, он прослушивает локальный порт 127.0.0.1:3305,

перенаправляющий SQL-запросы на 127.0.0.1:3306 (MySQL-настройка по умолчанию). Он может работать разными способами, настраивая работу в разных режимах.

Разработано три режима защиты БД от угроз информационных технологий.

1. Режим защиты от проникновений в БД.

2. Режим обучения.

3. Режим обнаружения аномалий.

Рассмотрим подробно каждый режим.

1. Режим защиты от проникновений в БД**.** В этом режиме система будет блокировать SQL-запрос и возвращать пустой запрос клиенту, если при использовании эвристического метода обнаружения аномалий были объявлены неправильные действия; иначе система вызовет сервер БД для выполнения запроса.

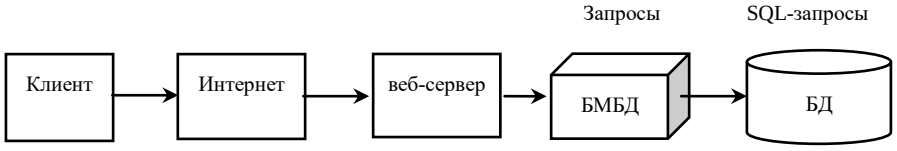

Рис. 1

Используя эвристический метод обнаружения аномалий, определяем функции SQL-выражений, которые вызывают сомнения, далее определяем значения для каждой функции. Рассмотрим аномальные функции SQL-запросов: административные запросы, включая запросы клиентов, использующиеся для создания или хранения таблиц и БД, смену структуру таблиц данных; запросы, которые ведут к утечке важной информации во время выполнения; другими подобными являются один комментарий в SQL-выражении, пустая строка пароля, знак «OR» внутри запроса, SQL-выражение, которое всегда «возвращает правду».

Примеры аномальных функций и их значения указаны в табл. 1

Таблица 1

Функции	Аномальные функции SQL1	Значение
*K*1 (SQL1)	Существующее выражение всегда возвращает TRUE	30
*K*2 (SQL1)	Существует пустой пароль: PASS = «	30
*K*3 (SQL1)	Есть элемент «OR»	5
Аномальное значение для SQL1	—	65

В одном SQL-выражении может быть много аномальных функций (*K*1, *K*2, *K*3, *K*4, ..., *Kn*), соответствующие им метки *S*(*K*1), *S*(*K*2), *S*(*K*3), *S*(*K*4), ..., *S*(*Kn*), а сумма значений аномальных функций одного SQL-выражения: SUM(запрос) = = *S*(*K*1) + *S*(*K*2) + *S*(*K*) + *S*(*K*4) + ... + *S*(*Kn*). Также определяется порог значения *K* (по умолчанию равно 30, но это можно перенастроить). Для того чтобы найти аномалии, брандмауэр использует свой собственный обработчик SQL-языка для поиска операторов SQL. Он считает значение аномалий для каждого прослушивания запроса; любой SQL-запрос будет блокирован, если значение аномалий превышает заданное в настройках значение, например:

SQL-выражение SQL1:
Select * from admin where email = «or «1» = «1» and pass =«.

Вышеприведенное выражение имеет три аномальные функции.

T1: существующее выражение всегда будет возвращать TRUE: 1 = 1;
T2: существует пустой пароль: PASS = «;

Т3: есть элемент «OR».
SUM (запрос) = $S(K1) + S(K2) + S(K3) + 30+30+5 = 65 > 30$.

Запрос будет рассматриваться как аномальный и будет заблокирован БМБД.

2. Режим обучения. В режиме обучения БМ изучает нормальные функции доступа легитимных пользователей к БД: структуру SQL-запроса и вводимые пользователем постоянные символы.

Нормализация. SQL-запросы сначала воспринимаются как последовательность символов. Каждый символ имеет свой признак, который сообщает, является ли символ постоянным. Константы — единственные элементы SQL-запроса, которые должны содержать вводимые пользователем данные. Символы, отображающие введенные пользователем данные, увеличиваются свойством типа данные, включающим строку и данные. Переменные типа строки должны заключаться в одиночные кавычки, а номера не заключаются в кавычки. Номера, заключенные в кавычки, рассматриваем как строки. Затем функция-вектор инициализируется посредством извлечения всех символов, помеченных как константы, и ввода их с их атрибутами, который внедрится в запрос. Затем структура запроса генерируется заменой всех найденных констант в SQL-выражении пустым заполнителем. Структура запроса характеризует структуру SQL-выражения. Запишем структуру запроса в другой список, который называем белым листом, например, для запроса:

SELECT * FROM table WHERE a = «a1» AND b=b2 OR c=«c3»;

Извлекаем все символы, помеченные как константы и вводим их с их атрибутами в список, который внедрится в запрос, результатом которого будет следующая вектор-функция:

Вектор: a1 b2 c3. Структура запроса:
SELECT * FROM *table* WHERE *a* = *«заполнитель 1»* AND *b* = *«заполнитель 2»* OR *c* = *«заполнитель 3».*

Каждая вектор-функция имеет соответствующую структуру запроса, и наоборот, структура каждого запроса имеет соответствующую вектор-функцию.

Режим обучения имеет две фазы: подготовка и определение порога значения аномалии.

В фазе подготовки БМ слушает SQL-запросы от клиентов и записывает кодовое имя SQL-запроса. Затем система нормализует SQL-выражение, добавляя структуру запроса в белый список структур запросов. Множество профилей запросов называется набором профилей. Система слушает SQL-запросы от клиента продолжительно в режиме подготовки и нормализует их. Если профиль запроса найден для комбинации имени запроса/структуры, то каждый элемент вектор-функции передается соответствующей модели для обновления нормальности модели. Если профиль не найден, создается новый профиль и добавляется в набор профилей БД. Профиль создается как экземпляр набора профилей для каждого элемента вектор-функции. Тип экземпляра моделей зависит от типа данных элемента. Например, элемент символьного типа ассоциируется с моделями, подходящими для моделирования строк, в то время как элемент числового типа соединится с моделями, моделирующими числовые элементы. Здесь определим модели только для двух типов элементов: строки и числа. Элементы, заключенные в кавычки, рассматриваются как строковый тип.

Перейдем к фазе определения порога значения аномалии. Система все еще нуждается в нормализации принимаемых SQL-запросов и обращается к профилю тем же способом, что и в предыдущей фазе. Но вектор-функция в этой фазе не используется для обновления модели. Наоборот, модели используются для генерации значения, что является мерой того, насколько хорошо вектор-функция подходит модели. Усредненное значение называется значением аномалии запроса и вычисляется как сумма отри-

цательных логарифмов каждого значения отдельной модели. Для каждого профиля наибольшее значение аномалии во время выполнения всей фазы называется порогом значения аномалии запроса. Если ни один профиль не найден, брандмауэр автоматически создаст новый профиль для запроса.

3. Режим обнаружения аномалий. Система работает в этом режиме, пока SQL-запрос все еще нормализуется, а затем проверяет, находится ли структура запроса в белом списке структур запросов. Если нет, БМ блокирует запрос; иначе система вычислит значение аномалии запроса способом, идентичным предыдущей фазе, но если значение аномалии превышает порог значения аномалии, записанный на предыдущей фазе, на определенный процент, запрос будет блокирован. Процесс работы системы в этом режиме показан на рис. 2.

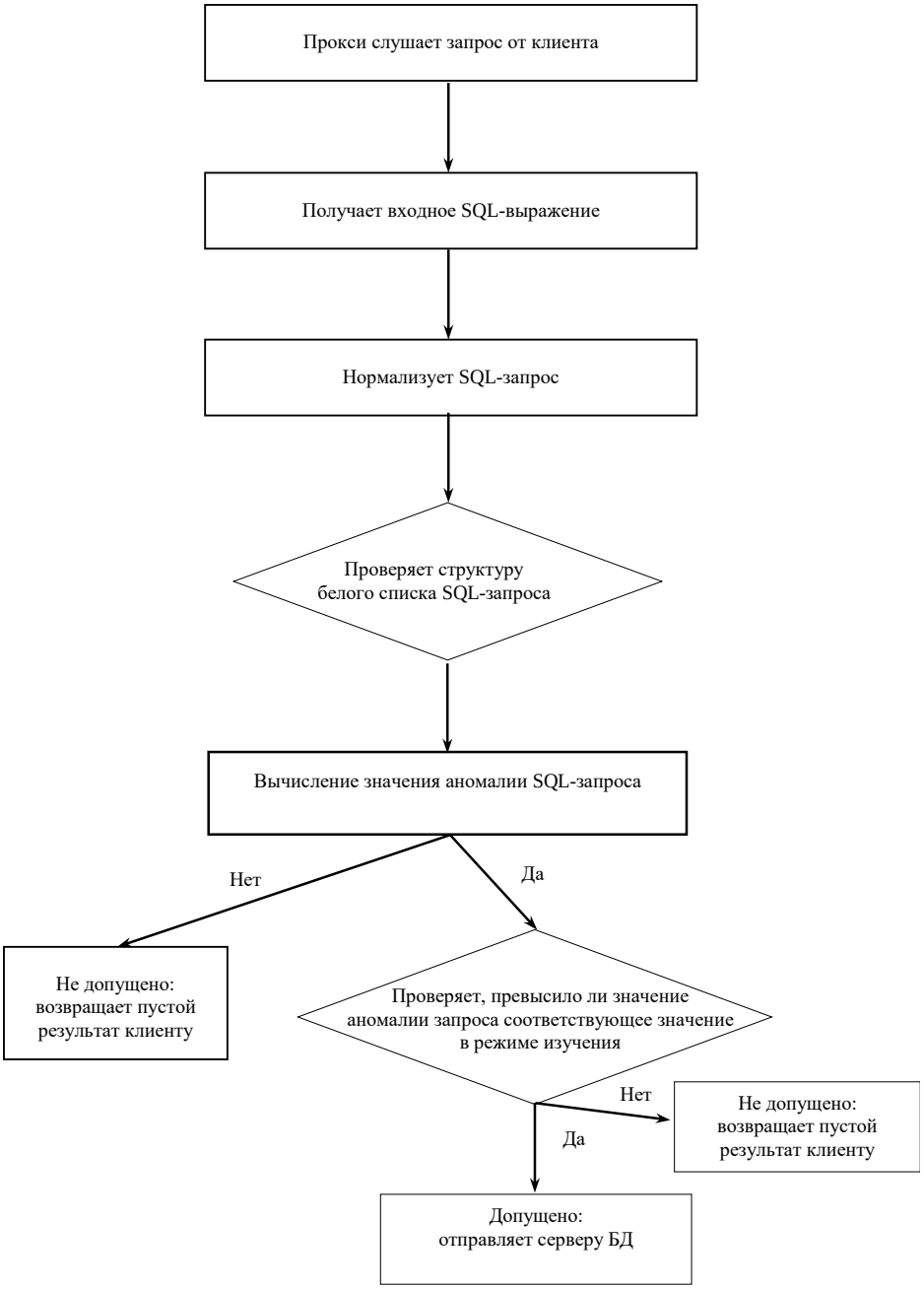

Рис. 2

Оценка эффективности метода защиты БД на основе БМ веб-приложений

Для оценки эффективности настроим БМ двумя способами.

1. С помощью эвристического метода обнаружения аномалий обнаруживать аномальные запросы и блокировать их.

2. Нормализация запросов к БД, а также проведение этапа обучения.

Также вручную создано несколько необычных атак для проверки, может ли БМ блокировать необычные атаки двумя способами.

1. При использовании этого способа необходимо только настроить его работу в режиме защиты от проникновений в БД, а также изменить настройки веб-приложения. Проводим проверки необычных атак посредством ручного оперирования веб-сайтами, может ли наш БМ предотвратить новые типы атак во время работы этим способом (табл. 2).

Все перечисленные веб-приложения коммерческие, и все они разного масштаба. Приложения не являются большими, поэтому можно понять поведение веб-приложений и облегчить работу тестирования данных. (Правильный + вредоносный) означает, что можно определить количество правильных и вредоносных запросов; формируется различное количество правильных и вредоносных запросов для каждого веб-приложения из-за уязвимости в веб-приложениях и для вычисления различных видов атак (известный и неизвестные атаки) на БД каждого веб-приложения.

Кроме того, вручную формируем другой набор вредоносных запросов для каждого веб-приложения. Предотвращение означает количество запросов, которые блокируются БМ; ложные срабатывания означают количество запросов, которые не были блокированы. По результатам можем видеть, что при этом способе количество ложных срабатываний относительно выше.

2. При использовании второго способа сначала необходимо сконфигурировать БМ, пока он работает в режиме обучения; после режима обучения наш БМ автоматически начинает работать в режиме обнаружения аномалий. Здесь вручную приводим другое определенное количество защищенных от атак запросов для каждого веб-приложения, кроме правильных наборов данных. Эти наборы данных используются в двух фазах режима обучения. Четыре веб-приложения невелики по объему, поэтому легко изучить профиль доступа к нормальной базе данных. В режиме обнаружения использовали те же наборы данных, что и в первом способе.

Количество использованных в двух фазах режима обучения запросов и результаты эксперимента приведены в табл. 2.

Таблица 2

Тематика	Правильный + вредоносный		Количество предотвращений		Ложные срабатывания		Режим обучения
	1-й способ	2-й способ	1-й способ	2-й способ	1-й способ	2-й способ	2-й способ
Интернет-магазин	576 + 1100	576 + 1100	950	1050	55	15	2678
Портал	324 + 1200	324 + 1200	1100	1150	72	9	4689
Система обучения	623 + 1600	623 + 1600	1433	1550	120	7	5686

ISSN 0572-2691

По результатам видно, что, используя ту же самую тестовую систему, количество ложных срабатываний БМ, работающего этим способом, меньше, нежели количество ложных срабатываний при первом способе. Данный метод может не только блокировать известные атаки на БД, но и неизвестные.

Заключение

Представлен механизм обнаружения проникновений с помощью БМ веб-приложения. БМ веб-приложения уменьшает уровень активности или блокирует почти все обычные атаки, используя фильтры с правильным набором правил.

Разработан метод защиты баз данных веб-приложений от угроз ИБ, осуществлена реализация и оценка работы БМ для баз данных.

Данный метод позволяет построить защиту баз данных на основе БМ, это обеспечивает защиту данных от угроз ИБ вида SQL-инъекция. Увеличивается защищенность электронных ресурсов от 30 до 100 %.

Р.Х. Хамдамов, К.Ф. Керимов

МЕТОД ЗАХИСТУ БАЗИ ДАНИХ НА ОСНОВІ БРАНДМАУЕР ВЕБ-ДОДАТКІВ

Все ширше використовуються веб-додатки при читанні новин, оплат рахунків і онлайн-покупці. У міру зростання цих послуг збільшується кількість і ступінь атак на них, таких як: крадіжка персональної інформації, банківських даних і інші випадки кіберзлочинів. Все це є наслідком відкритості інформації в БД. Безпека веб-додатків значно залежить від безпеки БД. Дані запитів клієнтів зазвичай беруться набором запитів у користувача програми. Якщо введені користувачем дані перевіряються не дуже ретельно, можна зібрати безліч видів атак, що використовують веб-додатки для створення загроз з безпекою для БД. Із-за тимчасових обмежень програмісти веб-додатків зазвичай фокусуються на функціональності веб-додатків, але лише деякі турбуються про безпеку. Наводяться методи виявлення аномалій, використовуючи БМ для БД. Досліджуються методи проникнень і види зломів. Запропонований БМБД здатний блокувати відомі і невідомі атаки на веб-додатки. Це програмне забезпечення може працювати залежно від конфігурації. Помилкових спрацьовувань майже не буває, а накладних витрат на продуктивність відносно мало. Розроблений БМБД служить для захисту від атак на БД веб-додатків. Він працює як проксі, а це означає, що прийняті від клієнта запити на SQL-вирази спочатку відправлені на розроблений БМ сервера БД. БМ аналізує запит: запити, які вважаються дивними, блокуються БМ і клієнту повертається порожній результат.

Ключові слова: інформаційна безпека, веб-додаток, база даних, БМ, аномальні запити, SQL-вирази.

R.Kh. Khamdamov, K.F. Kerimov

DATABASE PROTECTION BASED ON WEB APPLICATION FIREWALL

Web applications are increasingly being used in activities such as reading news, paying bills, and shopping online. As these services grow, you can see an increase in the number and extent of attacks on them, such as: theft of personal

information, bank data and other cases of cybercrime. All of the above is a consequence of the openness of information in the database. Web application security is highly dependent on database security. Client request data is usually retrieved by a set of requests that request the application user. If the data entered by the user is not scanned very carefully, you can collect a whole host of types of attacks that use web applications to create security threats to the database. Unfortunately, due to time constraints, web application programmers usually focus on the functionality of web applications, but only few worry about security. This article provides methods for detecting anomalies using a database firewall. The methods of penetration and types of hacks are investigated. A database firewall is proposed that can block known and unknown attacks on Web applications. This software can work in various ways depending on the configuration. There are almost no false positives, and the overhead of performance is relatively small. The developed database firewall is designed to protect against attacks on web application databases. It works as a proxy, which means that requests for SQL expressions received from the client will first be sent to the developed firewall, rather than to the database server itself. The firewall analyzes the request: requests that are considered strange are blocked by the firewall and an empty result is returned to the client.

Keywords: information security, web application, database, firewall, abnormal queries, SQL expressions

1. Rustam Kh. Khamdamov, Komil F. Kerimov, Jalol Oybek ugli Ibrahimov. Method of developing a web-application firewall. *Journal of Automation and Information Sciences*. DOI: 10.1615/ JAutomatInfScien.v51.i6.60 New York, USA 6, 2019.
2. Рябко Д.М. Подход к тестированию уязвимости web-приложений от атак типа SQL-инъекций. УкрПРОГ. Киев, 2006.
3. Громов Ю.Ю., Драчев В.О., Иванова О.Г. Информационная безопасность и защита информации: Ст. Оскол: ТНТ, 2017. 384 с.
4. Низамутдинов М.К. Тактика защиты и нападения на ИТ-приложения. Санкт-Петербург: БХВ-Петербург, 2005. С. 30–60.
5. Запечников С.В. Информационная безопасность открытых систем: В 2-х т. Т.1 Угрозы, уязвимости, атаки и подходы к защите. М. : ГЛТ, 2017. 536 с.
6. Opanasenko V.N., Kryvyi S.L. Synthesis of adaptive logical networks on the basis of Zhegalkin polynomials. *Cybernetics and Systems Analysis*. New York : Springer, 2015. **51**, N 6. P. 969–977. DOI: 10.1007/s10559-015-9790-1.

Получено 11.06.2020

После доработки 27.08.2020