УДК 004.655.3 **Технические науки**

Бондаренко Е.С., *студентка 5 курса электротехнического факультета,*
Пермский национальный исследовательский политехнический университет

ИССЛЕДОВАНИЕ УЯЗВИМОСТИ БАЗ ДАННЫХ ОТ SQL-ИНЪЕКЦИЙ С ИСПОЛЬЗОВАНИЕМ ВЕБ-ПРИЛОЖЕНИЯ WEBGOAT

Аннотация. В статье рассматриваются основные угрозы баз данных от SQL-инъекций. На основании угроз проводится исследование методов реализации наиболее распространенных угроз информационной безопасности баз данных.

Ключевые слова: sql-инъекция, система управления базой данных (СУБД), информационная безопасность, злоумышленник, база данных (БД), угроза, веб-приложение, WebGoat.

Abstract. In this article the basic threats of database from SQL-injection is considered. Based on threats the research of methods of realization the most spread threats of database's information security is conducted.

Keywords: SQL-injection, database management system (DMS), information security, malefactor, database, threat, web-application, WebGoat.

В настоящее время Веб-технологии используются повсеместно в интернет-магазинах, банках, веб-страницах предприятий. При этом базы данных часто используются для написания WEB-приложений. В наше время работа с базами данных поддерживается практически всеми языками программирования: BASIC, C++, Java, PERL, PHP, Assembler и даже JavaScript. Эти специфические программы - системы управления базами данных (СУБД). Их использование наиболее уместно для хранения пользовательских регистрационных данных, идентификаторов сессий, организации поиска, а также других задач требующих обработки большого количества данных.

Для СУБД характерны угрозы информационной безопасности. В процессе аутентификации для получения доступа к данным злоумышленник может использовать информационную атаку типа SQL Injection. Суть данной атаки

заключается в использовании ошибки на стыке WEB технологий и SQL. Это обусловлено тем, что многие web-страницы для обработки пользовательских данных, формируют специальный SQL запрос к БД, что может привести к внедрению вредоносного кода.

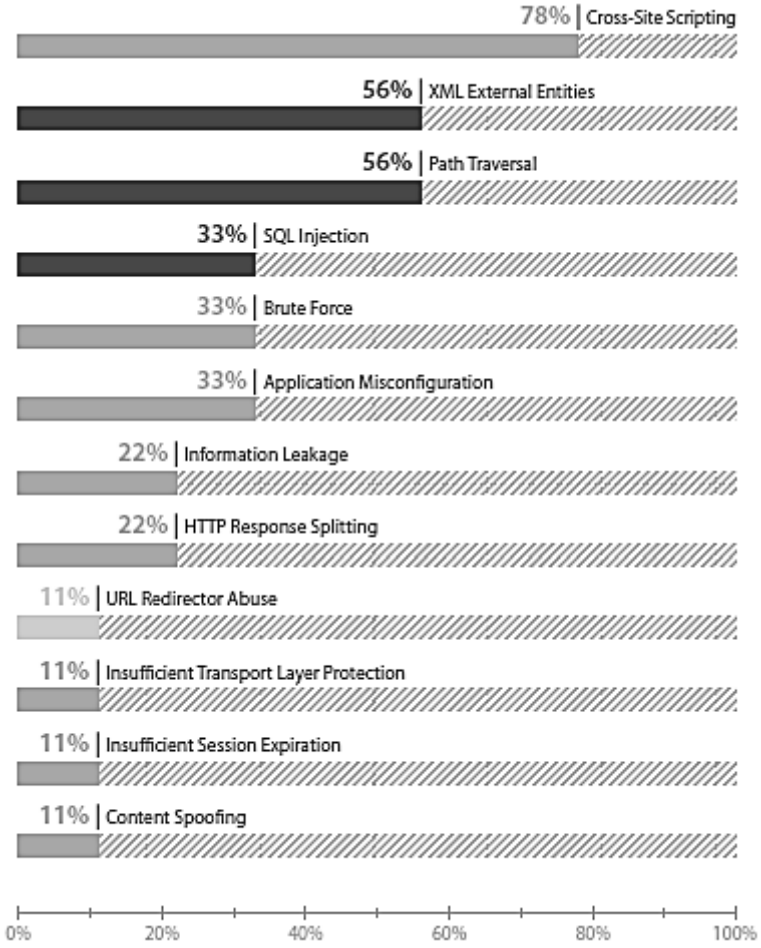

Рисунок 1 – Перечень наиболее опасных уязвимостей веб-приложений

Positive Research – один из крупнейших в Европе исследовательских центров в области информационной безопасности. Он является инновационным подразделением компании Positive Technologies, ключевого эксперта в сегменте практических аспектов защиты.

По данным Positive Research за 2016 год внедрение SQL-запросов заняло 4 место в десятке самых опасных уязвимостей веб-приложений.[1]

Большинство недостатков (54%) связаны с ошибками, позволяющими проводить атаки на клиентов веб-приложений. 89% выявленных уязвимостей вызваны ошибками в программном коде, допущенными разработчиками. Оставшиеся 11% связаны с некорректной конфигурацией веб-приложений.[2]

К основным угрозам баз данных от SQL-инъекций относятся:

1. Угроза обнаружения базы данных на сервере. Первое, что необходимо сделать злоумышленнику – выяснить, на каком сервере расположена база данных.

2. Угроза получения доступа к базе данных. После обнаружения базы данных, для манипуляции с данными злоумышленнику необходимо получить несанкционированный доступ к базе данных, т.е. либо создать пользователя с привилегиями администратора, либо получить данные для авторизации для уже существующей учетной записи администратора.

3. Угроза получения доступа к консоли (доступа к выполнению различных команд). Получения доступа к консоли позволит злоумышленнику выполнять команды непосредственно в операционной системе базы данных. Могут выполняться такие действия, как: создание или удаление файлов, включение или выключение устройств ввода, копирование информации с экрана, или системных файлов.

4. Угроза превышения привилегий пользователя. Эта угроза может возникнуть, если злоумышленнику не удалось получить доступ к учетной записи с привилегиями администратора. Но в его распоряжении может быть учетная запись с некоторыми функциями администрирования, но без права выполнять консольные команды.

5. Угроза уничтожения всей базы данных.

К основным принципам действия и цели злоумышленника, относятся:

- изменение запроса, нарушение логики его выполнения;
- возов ошибки синтаксиса SQL-запроса;
- внедрение своего запроса, эксплуатируя SQL-инъекцию;
- получение учетных данных доступа к сайту из базы данных.

В таблице 1 представлены некоторые примеры SQL-инъекций [3], на основании которых были проведены исследования реализации наиболее распространенных угроз информационной безопасности. Цель исследования заключалась в выявлении характерных SQL-инъекций, позволяющих реализовать информационную атаку на СУБД.

Таблица 1 – Сопоставление угрозе БД методу SQL-инъекции

№	Угроза	Применяемый метод SQL-инъекции
1.	Угроза обнаружения базы данных на сервере	Первое действие злоумышленника - сбор информации о сервисах, расположенных на сервере. Самое главное, что нужно знать для поиска Microsoft SQL Server, — номера портов, которые он «слушает». Например, порты 1433 (TCP) и 1434 (UDP). Чтобы проверить, имеется ли MS SQL на сервере жертвы, необходимо его сканирование. Для этого можно использовать Nmap со скриптом ms-sql-info. Помимо Nmap, используется специальный сканирующий модуль для Metasploit mssql_ping, позволяющий также определять наличие MS SQL на атакуемом сервере
2.	Угроза получения доступа к базе данных	Осуществляется с помощью модуля Metasploit mssql_login. С помощью чего происходит подбор пароля. Возможен вариант, что необходимо так же будет подобрать и сам логин
3.	Угроза получения доступа к консоли (доступа к выполнению различных команд)	После получения доступа осуществляется авторизация. Далее включается хранимая процедура, позволяющая выполнять команды на уровне операционной системы, устанавливаем на сервер Meterpreter shell. Для этого используется метод mssql_payload, автоматизирующий данный процесс. После создания сессии Meterpreter’a, может осуществляться полный доступ к выполнению различных команд
4.	Угроза повышения привилегий пользователя	Используются хранимые процедуры и активированное свойство TRUSTWORTHY. Для просмотра активности свойства TRUSTWORTHY используется модуль для Metasploit mssql_escalate_dbowner_sqli
5.	Угроза уничтожения всей базы данных	Выполнение команды от имени администратора DROP DATABASE

Существуют различные рекомендации для защиты веб-приложения от SQL-инъекций, к основным из которых относятся [4]:

1. Не размещать в базах данных данные без предварительной обработки. Это осуществляется либо с помощью подготовленных выражений, либо путем обработки параметров вручную. Если запрос оставляется вручную, то:

– все числовые параметры должны быть приведены к нужному типу;

– все остальные параметры должны быть обработаны функцией mysql_real_escape_string() и заключены в кавычки.

2. Не помещать в запрос управляющие структуры и идентификаторы, введенные пользователем. Необходимо заранее прописывать в скрипте список возможных вариантов, и выбирать только из них.

3. Соблюдать специальные правила составления SQL-запросов. Например, все вставляемые строковые данные должны быть заключены в одинарные или

двойные кавычки (рекомендуется использовать одинарные), чтобы данные были экранированы специальными символами.

4. Использовать динамическое составление запросов.
5. Вести правильную работу со спецсимволами при составлении запросов.
6. Использовать подготовленные выражения. Суть его заключается в том, что подготавливается шаблон запроса, со специальными маркерами, на место которых будут подставлены динамические компоненты.

Специалисты по безопасности и разработчики должны контролировать сеть с точки зрения потенциального взломщика, понимая суть атак и выявляя уязвимые места системы. Разнообразные руководства по обеспечению информационной безопасности, которые легко найти в общедоступных источниках могут дать лишь теоретические знания.

Без их практического закрепления они не очевидны и будут не достаточно эффективными. В данном случае разработчиками **OWASP (Open Web Application Security Project)** создана специальная обучающая система **WebGoat**, позволяющая в наглядном виде изучать приемы взлома веб-приложений. [5] В данной системе реализована база для проведения около 30 различных видов атак. Основной акцент в обучении сделан именно на образовательную сторону вопроса, а не создание уязвимой платформы для опытов.

В WebGoat реализованы все сопутствующие элементы – лекции, проверки знаний, лабораторная работа и результирующий экзамен. По ходу обучения ведется статистика, показывающая результат на каждом этапе. Список курсов обширен и затрагивает базовые знания по HTML, контроль доступа, и различные виды атак – XSS, различные виды Injections, Buffer Overflow, работа со CSS и скрытыми полями в формах и так далее.

В процессе обучения объясняется суть проблемы, даются все необходимые подсказки и код, этап завершается практической демонстрацией взлома с использованием уязвимости. Пройденные лекционные материалы выделяются специальным образом. В WebGoat уже есть все необходимое, то есть не нужно самостоятельно собирать тестовую среду, чтобы проверить все на практике.

Таким образом, необходимость защиты СУБД требует лучшего понимания принципов внедрения SQL-инъекций и, как следствие, умения защищать веб-приложения от них более эффективно, на основе получения практического опыта. Исследование уязвимости баз данных от SQL-инъекций с использованием Веб-приложения WebGoat позволяет выполнить данные задачи, повысить качество защиты информации.

Библиографический список

1. Аналитика [Электронный ресурс]. – *URL:* http://www.ptsecurity.ru/upload/ptru/analytics/Web-Vulnerability-2016-rus.pdf - (Дата обращения 18.08.2016).

2. Статистика уязвимостей веб-приложений (2014г.) [Электронный ресурс]. – *URL:* http://www.ptsecurity.ru//download/WEB_APP_VULNERABILITY_2014.A4.RUS.242465.14.OCT.2015.pdf - (Дата обращения 22.04.2016).

3. Sqlmap: SQL-инъекции — это просто [Электронный ресурс]. – *URL:* https://xakep.ru/2011/12/06/57950/ - (Дата обращения 22.04.2016).

4. Защита от SQL-инъекций [Электронный ресурс]. – *URL:* http://phpfaq.ru/mysql/slashes - (Дата обращения 22.04.2016).

5. Обучение защите веб-приложений с WebGoat [Электронный ресурс]. – *URL:* http://www.tux.in.ua/articles/2637 - (Дата обращения 22.04.2016).