УДК 004.491:004.423.24

ТЕСТИРОВАНИЕ ВЕБ ПРИЛОЖЕНИЙ НА НАЛИЧИЕ УЯЗВИМОСТЕЙ, СВЯЗАННЫХ С ВНЕДРЕНИЕМ ВРЕДОНОСНОГО КОДА

Мирошниченко А.В., студент гр. 961402

Белорусский государственный университет информатики и радиоэлектроники[1]
г. Минск, Республика Беларусь

Белоусова Е.С. – канд. техн. наук

Аннотация. В последнее время актуализировались вопросы, связанные с тестированием веб приложений на наличие уязвимостей. Целью данной работы является подробное теоритическое и практическое изучение наиболее популярных уязвимостей, а именно SQL-инъекция и межсайтовый скриптинг. Таким образом в работе представлены понятия и виды данных уязвимостей, принцип их реализации и возможные последствия внедрения вредоносного кода посредством эксплуатации SQL-инъекций и межсайтового скриптинга. Практическое изучение уязвимостей, связанных с внедрением вредоносного кода, осуществлялась в веб-приложение PentesterLab. Также в работе предложены ряд мер по защите веб-приложений от уязвимостей, связанных с внедрением вредоносного кода. Предложенный материал будет интересен специалистам, которые занимаются тестированием веб-приложений, а также студентам, обучающимся по специальностям, связанными с информационными системами, программным обеспечением информационных технологий, программируемыми мобильными системами и др.

Ключевые слова. Уязвимость, тестирование, веб-приложение, внедрение вредоносного кода, OWASP, SQL-инъекция, межсайтовый скриптинг (XSS), PDO (PHP Data Object).

Веб-приложение – клиент-серверное приложение, в котором клиент взаимодействует с веб-сервером при помощи браузера. Логика веб-приложения распределена между сервером и клиентом, хранение данных осуществляется, преимущественно, на сервере, обмен информацией происходит по сети. Тестирование веб-приложений – это метод выявления, анализа и сообщения об уязвимостях, существующих в веб-приложении. Топ-10 уязвимостей по статистике OWASP приведен на рисунке 1.

Топ-10 OWASP 2013		Топ-10 OWASP 2017
A1 - Внедрение	→	A1:2017-Внедрение
A2 - Недостатки аутентификации и управления сессиями	→	A2:2017-Недостатки аутентификации
A3 - Межсайтовое выполнение сценариев (XSS)	↘	A3:2017-Разглашение конфиденциальных данных
A4 - Небезопасные прямые ссылки на объекты [Объединено с A7]	U	A4:2017-Внешние сущности XML (XXE) [Новое]
A5 - Некорректная настройка параметров безопасности	↘	A5:2017-Недостатки контроля доступа [Объединено]
A6 - Разглашение конфиденциальных данных	↗	A6:2017-Некорректная настройка параметров безопасности
A7 - Отсутствие контроля доступа на функциональном уровне [Объединено с A4]	U	A7:2017-Межсайтовое выполнение сценариев (XSS)
A8 - Межсайтовая подмена запросов (CSRF)	☒	A8:2017-Небезопасная десериализация [Новое, Сообщество]
A9 - Использование компонентов с известными уязвимостями	→	A9:2017-Использование компонентов с известными уязвимостями
A10 - Непроверенные перенаправления и переадресации	☒	A10:2017-Недостатки журналирования и мониторинга [Новое, Сообщество]

Рисунок 1 – ТОП-10 уязвимостей по статистике OWASP

SQL-инъекция – один из распространенных способов взлома сайтов и программ, работающих с базами данных, основанный на внедрении в запрос произвольного SQL-кода. Внедрение SQL, в зависимости от типа используемой СУБД и условий внедрения

Схема основного принципа действия SQL инъекций приведена на рисунке 2.

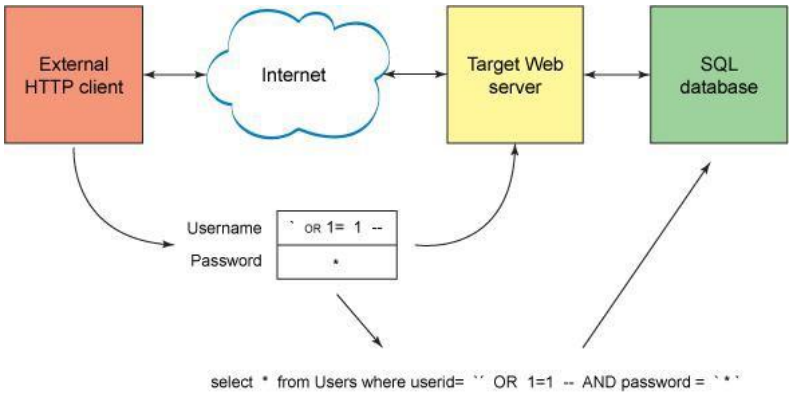

Рисунок 2 – Схема принципа действия SQL инъекций

Виды SQL-инъекций:

– инъекция в строковом параметре;

– инъекция в цифровом параметре.

Изучение данных видов SQL-инъекций было практически реализовано в веб-приложение PentesterLab. Так, например, было проверена работа приложения при реализации следующего запроса: *http://IP-address/sqli/example1.php?name=root' OR 1=1 --+*.

Реализация запроса *http://IP-address/sqli/example1.php?name=root' ORDER BY X* позволила установить количество столбцов в таблице пользователей в базе данных. В представленном запросе X соответствует номеру столбца. Изменяя значение *X* от 1, можно заметить, что приложение будет работать корректно, но при вводе значения *X*=6, приложение выдало ошибку, что позволяет сделать вывод, что общее количество столбцов в таблице пользователей равно 5.

Далее был использован SQL-оператор UNION, который позволяет объединять в одном запросе обращение к разным таблицам базы данных, что позволяет получить данные из таблиц, к которым доступ не предусмотрен. С помощью запроса *http://IP-address/sqli/example1.php?name=root' UNION+SELECT+1,2,3,4,5+FROM+users+WHERE+id=1* были получены данные из таблицы users, в том числе их имена и пароли, хотя приложением не предусмотрен доступ к данной таблице базы данных.

Можно сделать вывод, что вариаций использования SQL-инъекций очень много и большинство из них приводит к явному получению неправомерного доступа к информации.

Предложены следующие способы защиты от SQL-инъекций:

1) Использовать белые списки.
2) Не использовать метод GET в формах.
3) Проверять и обрабатывайте переменные.
4) Проверять источник запросов и сами запросы на наличие SQL-инъекций.
5) Использовать PDO (PHP Data Object).

Межсайтовый скриптинг (XSS) – это одна из разновидностей атак на веб-системы, которая подразумевает внедрение вредоносного кода на определенную страницу сайта и взаимодействие этого кода с удаленным сервером злоумышленников при открытии страницы пользователем. Схема принципа действия XSS приведена на рисунке 3.

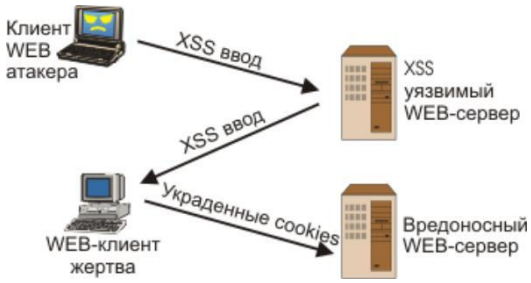

Рисунок 3 – Схема принципа действия межсайтового скриптинга

Выделяют следующие виды XSS атак: пассивные и активные.

В веб-приложение PentesterLab были практически реализованы основные виды XSS атак и установлены следующие особенности реализации данной атаки:

1) Для реализации простейшей XSS атаки достаточно использовать функцию alert.
2) PHP убирает тег из запроса, поэтому необходимо использовать заглавные символы.

3) PHP преобразует все символы в нижний регистр и убирает теги, поэтому необходимо добавлять один тег в другой.

4) Для обхода проверки запросов необходимо воспользоваться стандартным параметром onerror, который возникает при отсутствии изображения.

5) В случае запрета на уровне приложения использования функции alert, нужно проверять возможность внедрения любой другой JS функцию.

Таким образом, уязвимость межсайтового скриптинга является довольно опасной и распространенной. Основная опасность заключается в том, что при обработке HTML реализуются все функции, которые находятся между тегами. Поэтому злоумышленник может без проблем воспользоваться уязвимостью и получить пользовательские данные.

На основе практического изучения данной атаки предложены следующие способы защиты от XXS:

1) Защита функцией htmlspecialchars.
2) Защита функцией strip_tags.
3) BB-коды.
4) Регулярные выражения.
5) Самописные функции.

Список использованных источников:

1. OWASP. Топ-10 OWASP – 2017. Десять самых критических угроз безопасности веб-приложений. [Электронный ресурс]. – Режим доступа: https://owasp.org/www-pdf-archive/OWASP_Top_10-2017-ru.pdf/ – Дата доступа: 25.02.2021.

2. Уязвимости и угрозы веб-приложений в 2019 году / Positive Technologies Электронный ресурс]. – Режим доступа: https://www.ptsecurity.com/ru-ru/research/analytics/web-vulnerabilities-2020/. – Дата доступа: 25.02.2021.

3. Статья SQL injection для начинающих. Часть 1. / Habr [Электронный ресурс]. – Режим доступа: https://habr.com/ru/post/148151/. – Дата доступа: 25.02.2021.

UDC 004.491:004.423.24

TESTING WEB APPLICATIONS FOR VULNERABILITIES RELATED TO THE IMPLEMENTATION OF MALICIOUS CODE

Mirashnichenka A.V., Student of the group 961402

Belarusian State University of Informatics and Radioelectronics
Minsk, Republic of Belarus

Belousova E.S. – PhD

Annotation. Recently, issues related to testing web applications for vulnerabilities have been actualized. The aim of this work is a detailed theoretical and practical study of the most popular vulnerabilities, namely SQL injection and cross-site scripting. Thus, the paper presents the concepts and types of these vulnerabilities, the principle of implementation and the possible consequences of a malicious code injection through the exploitation of SQL injection and cross-site scripting. A practical study of vulnerabilities related to the injection of malicious code was carried out in the PentesterLab web application. The paper also proposes a number of measures for protection web applications from vulnerabilities associated with the malicious code injection. The proposed material will be of interest to specialists who are engaged in testing web applications, as well as to students studying in specialties related to information systems, information technology software, programmable mobile systems, etc.

Keywords. Vulnerability, testing, web application, injection of malicious code, OWASP, SQL-injection, cross-site scripting (XSS), PDO (PHP Data Object).