УДК 004

Построение защиты сайта от XSS, SQL и CSRF-уязвимостей

Круглик Роман Игоревич
Приамурский государственный университет им. Шолом-Алейхема
Студент

Брыкова Анастасия Леонидовна
Приамурский государственный университет им. Шолом-Алейхема
Студент

Аннотация
В статье разобраны примеры использования XSS, SQL и CSRF-уязвимостей. Рассмотрены причины возникновения таких атак и выстраивание защиты.
Ключевые слова: XSS, SQL, CSRF, защита информации.

Building site protection against XSS, SQL and CSRF vulnerabilities

Kruglik Roman Igorevich
Sholom-Aleichem Priamursky State University
Student

Brykova Anastasia Leonidovna
Sholom-Aleichem Priamursky State University
Student

Abstract
In article explores examples of using XSS, SQL, and CSRF vulnerabilities. The reasons for the occurrence of such attacks and the formation of protection are considered.
Keywords: XSS, SQL, CSRF, information protection.

Для того чтобы понять, как организовать правильную защиту сайтов, нужно понимать где находятся уязвимые места и как не допустить их возникновения.

XSS – тип атаки на веб-системы, заключающийся во внедрении в выдаваемую веб-системой страницу вредоносного кода, (который будет выполнен на компьютере пользователя при открытии им этой страницы) и взаимодействии этого кода с веб-сервером злоумышленника.

SQL инъекция - это один из самых доступных способов взлома сайта. Суть таких инъекций - внедрение в данные (передаваемые через GET, POST запросы или значения Cookie) произвольного SQL кода.

CSRF – вид атак на посетителей веб-сайтов, использующий недостатки протокола HTTP. Если жертва заходит на сайт, созданный

злоумышленником, от её лица тайно отправляется запрос на другой сервер (например, на сервер платёжной системы), осуществляющий некую вредоносную операцию (например, перевод денег на счёт злоумышленника).

На сегодняшний день исследования в данной области являются актуальными. В статье Сторчак С.А. [1] рассматривает способность встроенной функции «htmlspecialchars» языка PHP противостоять межсайтовому выполнению сценариев. Полухин П.В. [2] рассматривает вопросы интеграции фаззинга в процесс обнаружения XSS уязвимостей современных веб-приложений, уделено особое внимание генерации тестовых данных для обхода механизмов фильтрации используемых в настоящее время веб-приложений и их компонентов. В статье Алексеев Д.М., Иваненко К.Н., Убирайло В.Н. [3] рассматривают XSS - атаки и способы защиты от них. Работа Садоян Р.А. [4] посвящена защите и обнаружению XSS-уязвимостей в web-приложениях. В статье Шелягов С.А., Кудрявченко И.В. [5] рассматривают вариант атаки на основе уязвимости СУБД, связанной с усечением данных в SQL. Коровин К.С. [6] рассматривает онлайн-сервис, выполняющий комплексное сканирование WEB-сайтов на сетевые уязвимости.

XSS уязвимость

Далее можно проверить как работает XSS. Ниже приведена структура формы авторизации и обработчик (рис.1).

```php
<?php

if ($_GET['n'] == 1) {
    echo "Ваше имя-";
    echo $_POST['name'];
    echo "<br>Ваш пароль-";
    echo $_POST['password'];
}
 ?>
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>XSS</title>
</head>
<body>
    <form action="http://localhost/SVG/XSS.php?n=1" method="POST">
    <input type="text" name="name" placeholder="Логин"><br><br>
    <input type="text" name="password" placeholder="Пароль"><br><br>
    <input type="submit" value="Отправить">
    </form>
</body>
</html>
```

Рисунок 1. Структура

Данный код создаёт 2 поля для ввода и при нажатии на кнопку, выводит данные (рис.2).

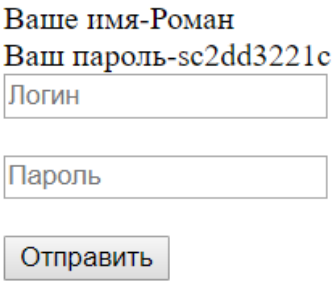

Рисунок 2. Результат работы формы

Можно попробовать внедрить скрипт для проверки на уязвимость:

<script>alert('XSS уязвимость найдена!') </script>

После чего была выведена соответствующая ошибка (рис.3).

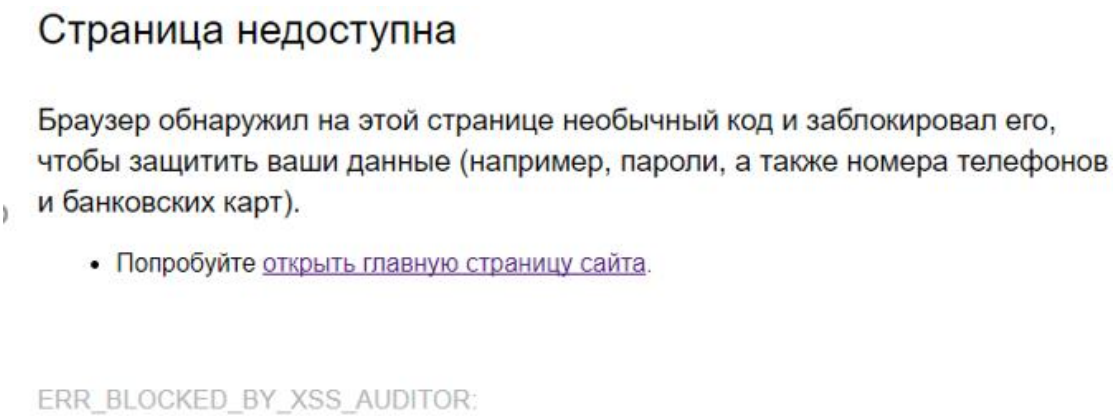

Рисунок 3. Срабатывание XSS Auditor

В браузере существует система защиты в виде XSS Auditor, который как раз и занимается данной атакой. Проблема решается одной строкой PHP кода:

header("X-XSS-Protection: 0");

После чего скрипт выполняется успешно (рис.4).

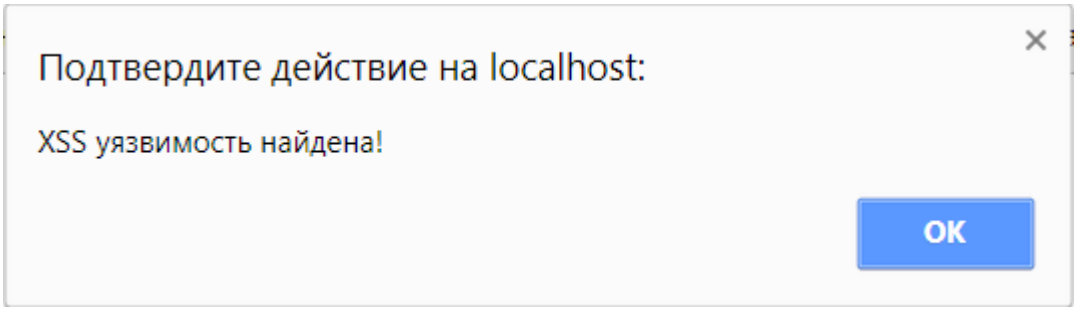

Рисунок 4. Успешное выполнение скрипта

При нахождение такой уязвимости её можно использовать в различных целях, для вытаскивания Cookie зарегистрированных пользователей:

<script>alert(document.cookie)</script>

Теперь понятно, как организуется атака и можно сделать вывод, что все данные которые вводятся на стороне клиента должны фильтроваться, можно использовать очень полезную функцию как htmlspecialchars, которая почти полностью исправляет проблему. Но честно говоря, хакеры в большинстве случаев не практикуют XSS уязвимости в силу своей нерентабельности, но от этого всё равно нужно быть защищенным.

SQL уязвимость

SQL инъекция появляется из входящих данных, которые не фильтруются. Самая распространенная ошибка - это не фильтрация передаваемого ID:

http://localhost/test.ru/post.php?id=3

Идёт обращение к файлу постов и передаётся id, который участвует в запросе к БД (рис.5).

```php
<?php
$id=$_GET['id'];
$zapros="SELECT * FROM `$post` WHERE `id`=$id";
 ?>
```

Рисунок 5. SQL запрос в БД

Принимается переданная переменная и записывается в запрос, соответственно, если изменить значение передаваемого значения, можно изменить запрос:

http://localhost/test.ru/post.php?id=3 UNION SELECT * FROM `posts` WHERE `id`=1;

UNION является функцией объединения запросов.

Далее идёт второй запрос, который достаёт все строки из id=1.

SQL уязвимость опасней, чем XSS.

CSRF уязвимость

Такую атаку можно легче всего спрятать на форумах и чатах, где люди обмениваются мнениями или картинками. Как говорилось ранее главное, это то что жертва атаки должна перейти по ссылки, которую ей подкидывает злоумышленник. Представим, что жертву зовут X, а злоумышленника Y, тогда получаем сообщение:

Y: Привет, дорогой X! Вы выйграли Айфон 10, забрать ваш выигрыш вы можете тут: <img src= "http://sberbank.test.ru/ withdraw?account=X&amount =1000000&for=Y">.

Ссылка переведёт человека X на сайт и выполнит определённые действия если система сохраняет её информацию об аутентификации в cookie и соответственно если они на момент перехода ещё действуют, то при попытке загрузить картинку, браузер отправит запрос на перевод денег на счет Y и подтвердит аутентификацию при помощи Cookie. После чего, транзакция будет успешно обработана и деньги дойдут до злоумышленников.

Можно сделать вывод о том, что нужно быть осторожным пользователем, не переходить на незнакомые и не очень хорошо проработанные сервисы. Разработчики должны проверять свои форумы и фильтровать вводимые пользователем данные.

Библиографический список

1. Сторчак С.А. Эффективность использования встроенной функции php для защиты от XSS//Перспективы развития информационных технологий. 2014. № 17. С. 161-166.
2. Полухин П.В. Интеграция инновационного подхода фаззинга для анализа XSS уязвимостей//В сборнике: Инновационное развитие России: проблемы и перспективы сборник статей III Международной научно-практической конференции. 2014. С. 46-49.
3. Алексеев Д.М., Иваненко К.Н., Убирайло В.Н. XSS - атаки и способы защиты от них//Новая наука: Теоретический и практический взгляд. 2016. № 9 (99). С. 98-99.
4. Садоян Р.А. Рейтинг уязвимостей в WEB-приложениях XSS-уязвимости защита и обнаружение//сборник трудов V Всероссийского конгресса молодых ученых материалы конгресса. 2016. С. 128-131.
5. Шелягов С.А., Кудрявченко И.В. Анализ применения механизма SQL-инъекции для получения несанкционированного доступа к информации на основе усечение данных в SQL//Символ науки. 2015. № 9-1. С. 123-126.
6. Коровин К.С. Разработка онлайн-анализатора веб-сайтов на наличие сетевых уязвимостей//Научные исследования и разработки студентов Сборник материалов IV Международной студенческой научно-практической конференции. 2017. С. 170-173.