МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н. Г. ЧЕРНЫШЕВСКОГО»

АТАКИ НА WEB ПРИЛОЖЕНИЯ. XSS, CSRF, SQL-ИНЪЕКЦИИ И ЗАЩИТА ОТ НИХ.

ОТЧЕТ О ПРАКТИКЕ

студента 4 курса 411 группы
направления 02.03.02 — Фундаментальная информатика и информационные технологии
факультета КНиИТ
Вильцева Данила Денисовича

Проверил
Преподаватель _____ А. А. Лобов

Саратов 2023

СОДЕРЖАНИЕ

1 Атака SQL-Injection (Внедрение SQL кода)

1.1 Условие задачи

Вам нужно войти за пользователя admin, но за него войти нельзя, так как пароль не задан. Используйте атаку SQL-инъекция из класса внедрения кода для того, чтобы это сделать.

1.2 Выполнение задачи

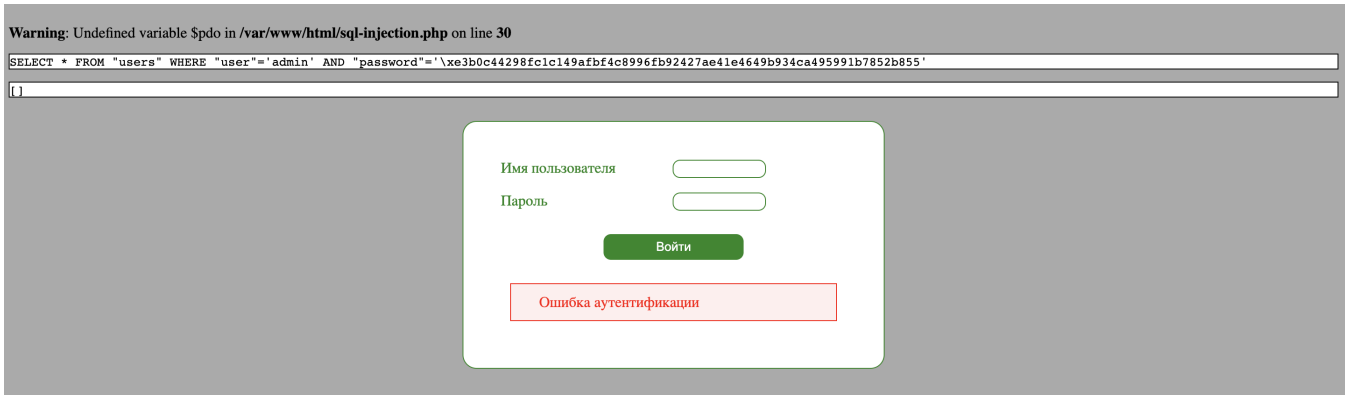

Рисунок 1

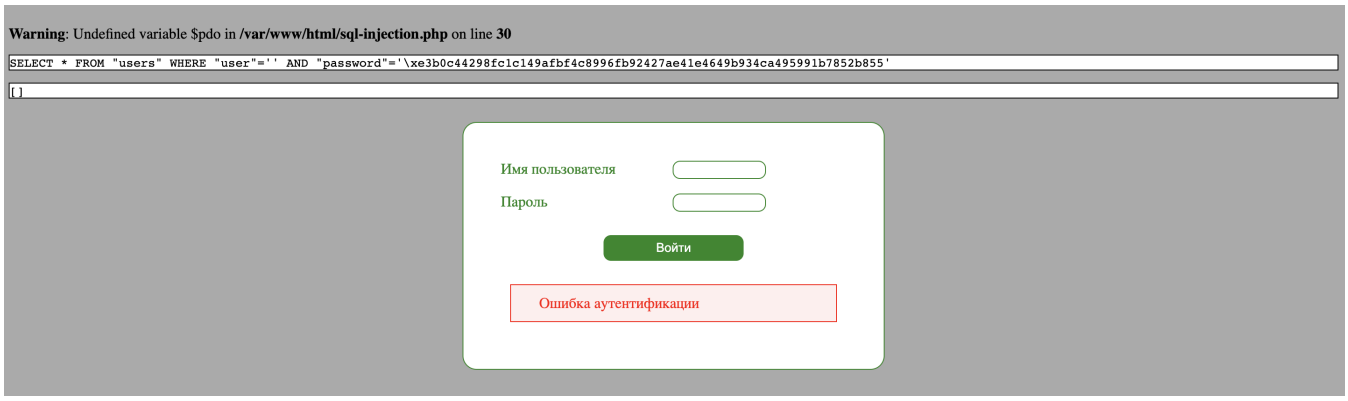

Рисунок 2

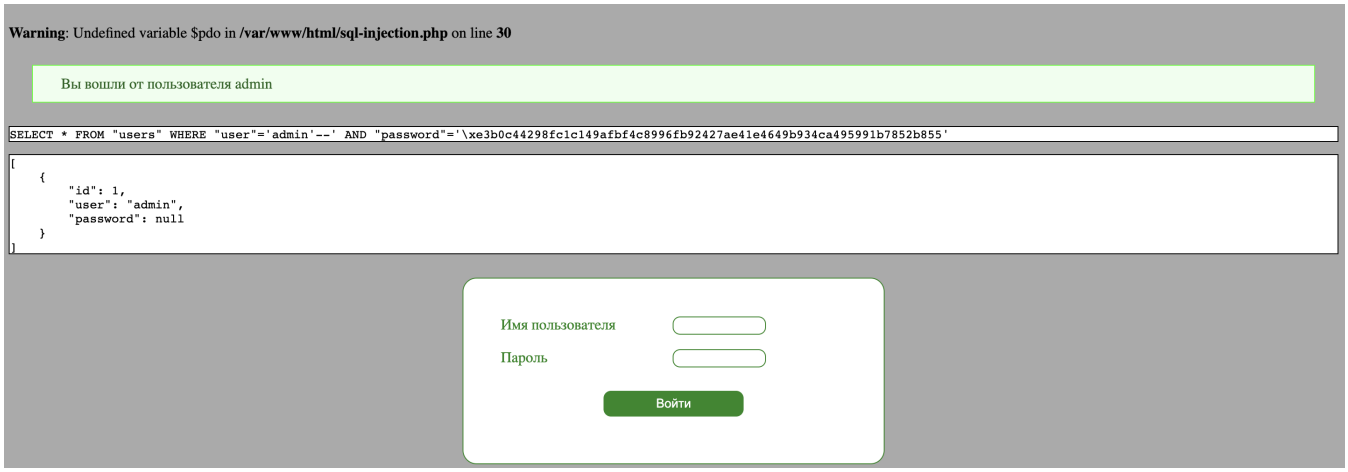

Рисунок 3

1.3 Контрольные вопросы

Подумайте, и напишите в отчёте, как от такой атаки защищаться.

— Поможет ли экранирование пришедших извне параметров, и что это вообще такое экранирование?

Ответ: Экранирование параметров, введенных пользователем, может предотвратить SQL-инъекции. Экранирование — это преобразование или фильтрация пользовательского ввода путем преобразования специальных символов в их соответствующие эквиваленты при передаче данных через различные протоколы и форматы.

— Помогут ли "вставки"параметров? То есть запросы вида: SELECT * FROM table WHERE column1=? AND column2=? с последующей подстановкой значений в заранее заданные места с помощью специальной функции?

Ответ:

Использование параметризованных запросов (подстановка значений в заранее заданные места) может защитить от SQL-инъекции. Это позволяет библиотекам баз данных обрабатывать ввод данных как данные, а не как часть SQL кода, и предотвращает возможность внедрения зловредного SQL-кода.

— Помогут ли для этого специальные библиотеки ORM (Object-Relation Mapping) и прочие, которые сами составляют запросы для получения и сохранения объектов из/в базу данных?

Ответ:

Да, специализированные библиотеки ORM (Hibernate, SQLAlchemy) могут помочь предотвратить SQL-инъекции, так как они генерируют SQL-

запросы автоматически на основе объектов и не требуют явного написания SQL-кода.

2 Атака XSS (Межсайтовый скриптинг)

2.1 Условие задачи

Напишите комментарий такого содержания: <script> alert(’Данный сайт подвержен XSS атаке’); </script>. Опишите, что случилось?

Попробуйте написать написать нужный комментарий, который перенаправит форму входа со страницы /xss.php на страницу /xss-server.php. Для этого нужно в комментарий вставить JavaScript-код в <script> </script>, где найти первую форму (тег form, можете использовать document.forms[0] в коде JavaScript) и заменить аттрибут action на /xss-server.php

2.2 Выполнение задачи

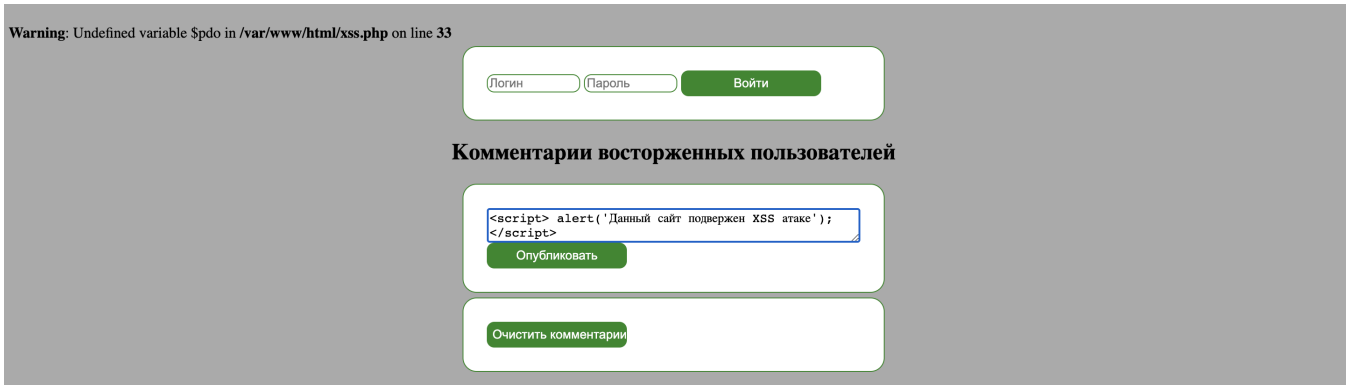

Рисунок 4

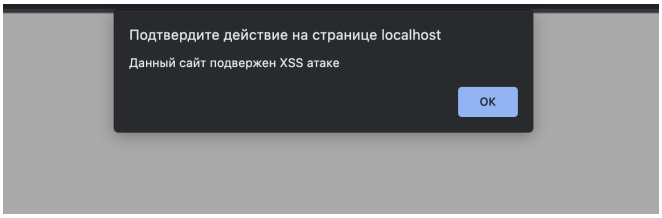

Рисунок 5

В этом случае, в результате атаки появилось всплывающее окно, которое появляется при заходе на страницу, обновление страницы и нажатии на кнопки. Введенный комментарий вставляется напрямую в HTML-код страницы и, как следствие, выполняется.

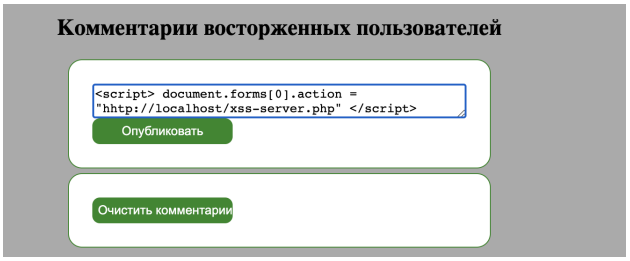

Рисунок 6

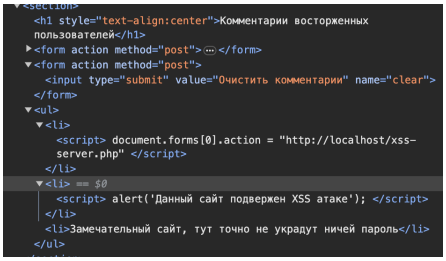

Рисунок 7

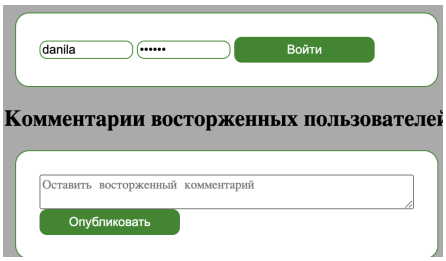

Рисунок 8

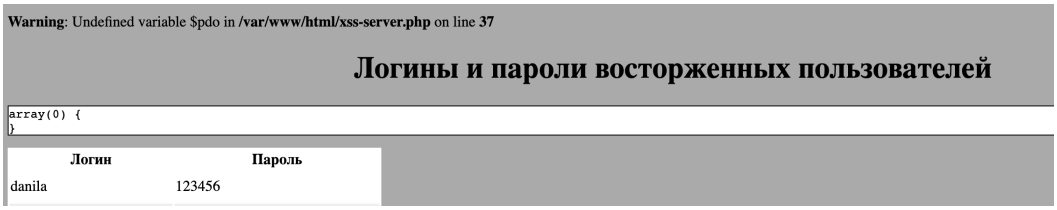

Рисунок 9

2.3 Контрольный вопрос

Как можно бороться с XSS-атаками?

— Валидация. Прием и обработка данных от пользователей должны включать валидацию ввода. Необходимо, отбрасывать данные, не соответствующие ожидаемым форматам.

— Шаблонизация. Позволяет избежать вставки сырого HTML и JavaScript кода в шаблоны. Шаблоны обрабатываются так, чтобы ничего не интерпретировалось как код.

— Экранирование. В некоторых языках программирования экранирование можно реализовать при помощи специальных функций, библиотек, веб-фреймворков и шаблонизаторов.
— Content Security Policy (CSP). Это HTTP-заголовок, который позволяет определить, какие ресурсы могут быть выполнены на странице. Он может помочь предотвратить выполнение внешних скриптов и сторонних ресурсов, которые могли бы быть использованы злоумышленником.

3 Атака CSRF

3.1 Условие задачи

Используя любой инструмент, отличный от браузера, пошлите запрос серверу. Можно использовать программу curl. Запрос POST, укажите отрицательный параметр ammount. Не забудьте про ваши печеньки (cookies), их нужно указать в запросе в заголовке Set-Cookie. Можно посмотреть в браузере. Для входа в инспектор используйте клавишу F12 (или другую с менее стандартными браузерами, уточните сами).

3.2 Выполнение задачи

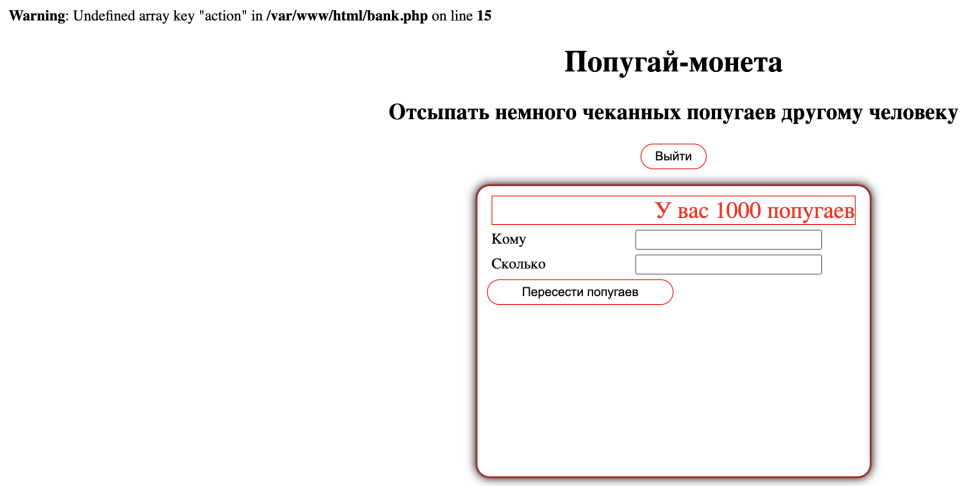

Рисунок 10

Рисунок 11

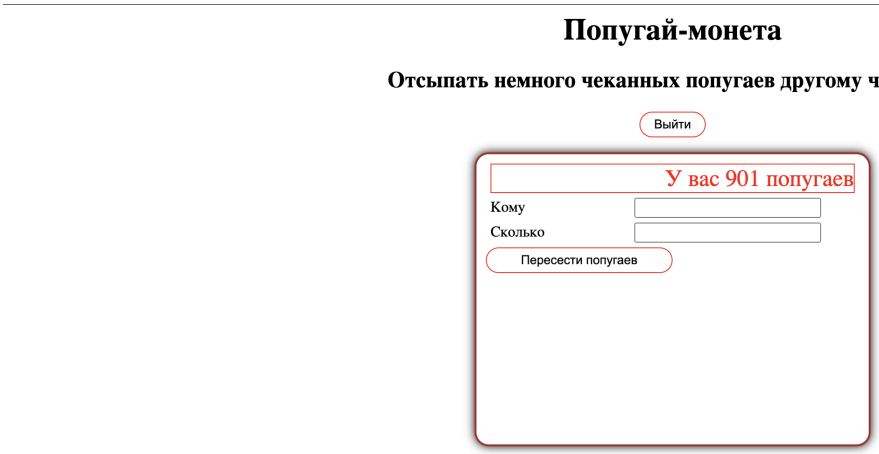

Рисунок 12

```
(base) danila@MacBook-Pro-Danila task3 % curl -d "user=user&ammount=-100&action=send" -X POST http://localhost/bank.php/ --cookie "auth=dXNlcg%3D%3D"

<html>

<head>
        <meta charset="utf-8">
        <title>Попугай-банк</title>
        <style>

.ammount {
        color:#f00;
        font-size:25px;
        border:1px solid #f00;
        width:200px
        padding:5px 10px;
        text-align:right;
}

form.default {
        border:solid 2px #a22;
        border-radius:15px;
        width:400px;
        margin:auto;
        height:300px;
        background:#fff;
        box-shadow:0 0 10px #000;
        padding:5px 10px;
}
```

Рисунок 13

```
        </div>
        <input type="submit" value="Пересести попугаев">
        <input type="hidden" name="action" value="send">
</form>


<div class="correct">Успешно отправлено -100 попугаев пользователю user</div>


</body>

</html>
```

Рисунок 14

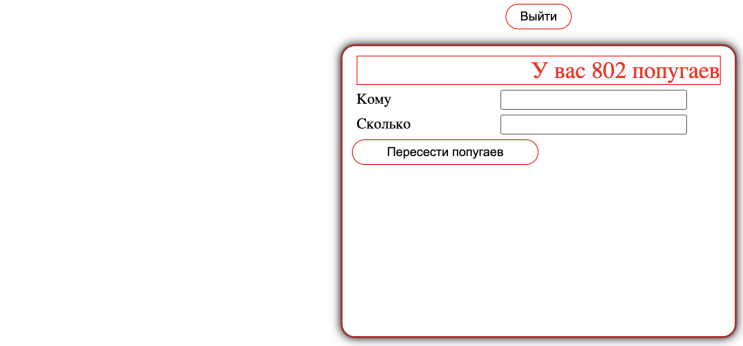

Рисунок 15

```
(base) danila@MacBook-Pro-Danila task3 % curl -d "user=user&ammount=198&action=send" -X POST http://localhost/bank.php/ --cookie "auth=aGFja2Vy"

<html>

<head>
        <meta charset="utf-8">
        <title>Попугай-банк</title>
        <style>

.ammount {
        color:#f00;
        font-size:25px;
        border:1px solid #f00;
        width:200px
        padding:5px 10px;
        text-align:right;
}
```

Рисунок 16

```
                <input name="user" id="user" >
        </div>
        <div>
                <label for="ammount">Сколько</label>
                <input name="ammount" type="number" min="0">
        </div>
        <input type="submit" value="Пересести попугаев">
        <input type="hidden" name="action" value="send">
</form>


<div class="correct">Успешно отправлено 198 попугаев пользователю user</div>


</body>

</html>
```

Рисунок 17

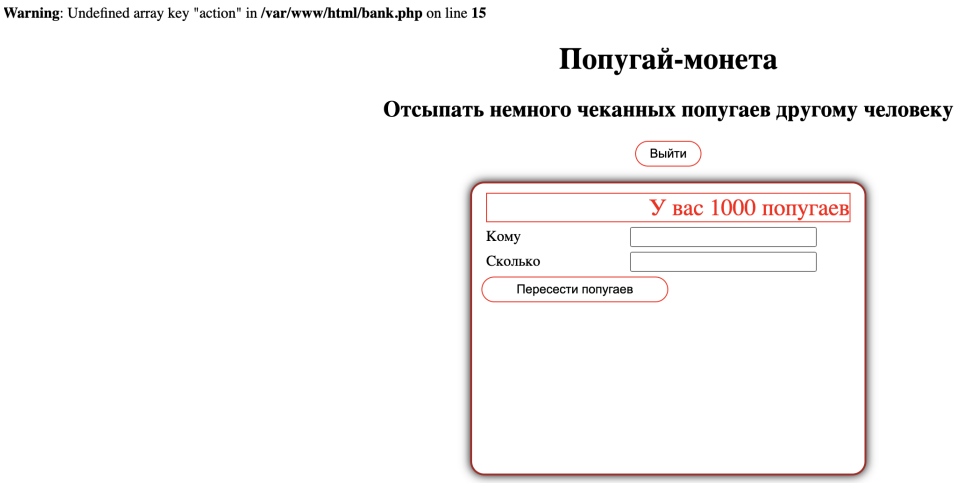

Рисунок 18

3.3 Контрольный вопрос

Как защититься от такой атаки? Подумайте или поищите что такое CSRF-токен.

CSRF токен — это уникальное, секретное и непредсказуемое значение, генерируемое приложением на стороне сервера и передающееся клиенту. При отправке запроса на выполнение конфиденциального действия, такого как отправка формы, клиент должен включить правильный токен CSRF. В противном случае сервер откажется выполнять запрошенное действие.

Использование CSRF-токенов позволяет защищититься от CSRF-атак. Приложение должно включать уникальный CSRF-токен в каждый запрос, который изменяет состояние приложения. Токен обычно хранится в сессии пользователя и включается в формы или заголовки запросов. При обработке запроса сервер проверяет, соответствует ли предоставленный токен ожидаемому значению.

CSRF-токены должны использоваться только один раз. После каждого успешного запроса токен должен быть обновлен.

SameSite — атрибут для куки, который при установке значения «Strict» или «Lax», позволяет ограничить отправку куки в запросах, и это может снизить риск CSRF.

Можно ограничить возможные для использования методы. Например, использовать только методы POST для изменения состояния приложения, так как многие CSRF- атаки основаны на запросах GET, которые могут быть выполнены без согласия владельца сервера.

Защита от сторонних источников: можно ограничить доступ к приложению только с известных и доверенных источников (например, через HTTP заголовок Origin или Referer).