Молдавский Государственный Университет

Факультет Математики и Информатики

Департамент Информатики

# Лабораторная работа №5

# По курсу: Безопасность информационных систем

# На тему: Инструменты сканирования уязвимостей

Проверила: Novac L., doctor, conf. universitar

Выполнила: I-2302, Ciobanu Stanislav.

Кишинев, 2024

**Цель работы:**

1. Используйте инструменты OWASP для сканирования уязвимостей в веб-приложениях (2-3 веб-приложения).

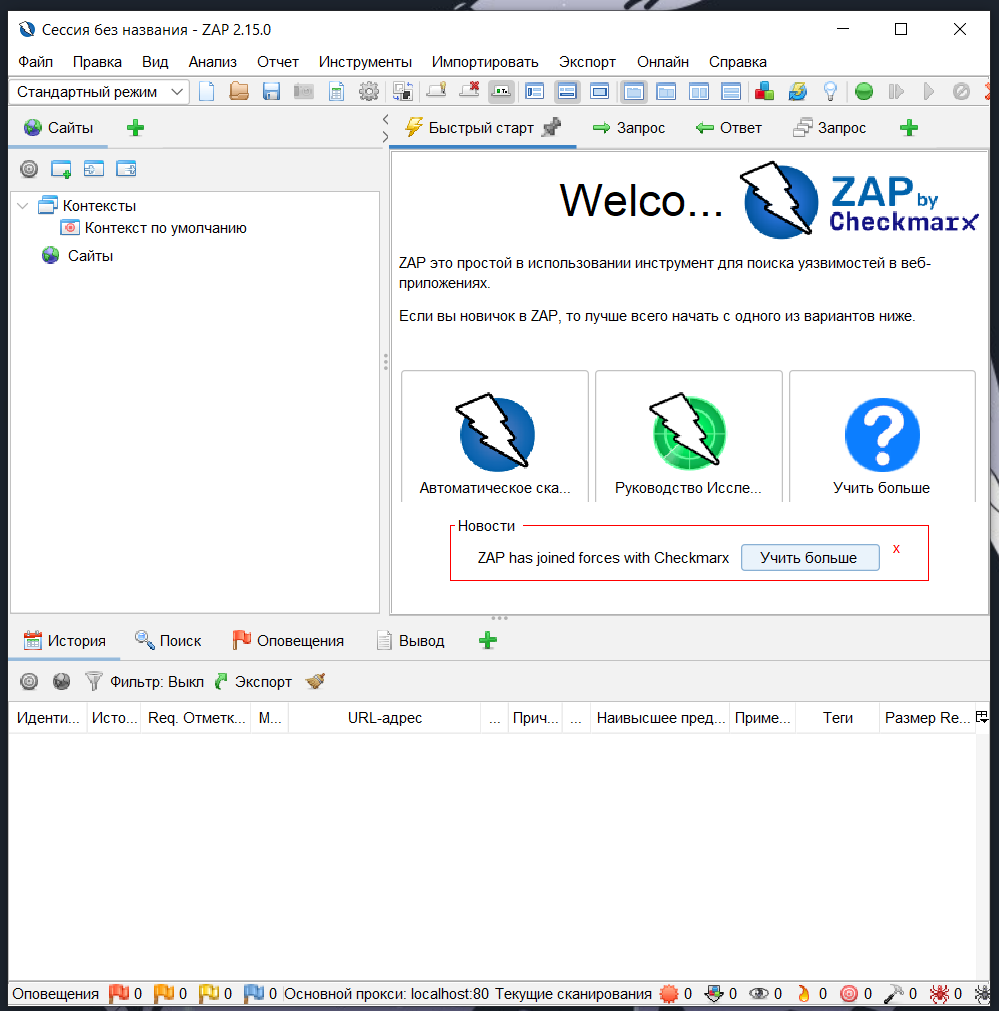
2. Определите, какие уязвимости встречаются, и опишите их.

3. Каковы методы решения тех проблем, которые вызваны определенными уязвимостями?

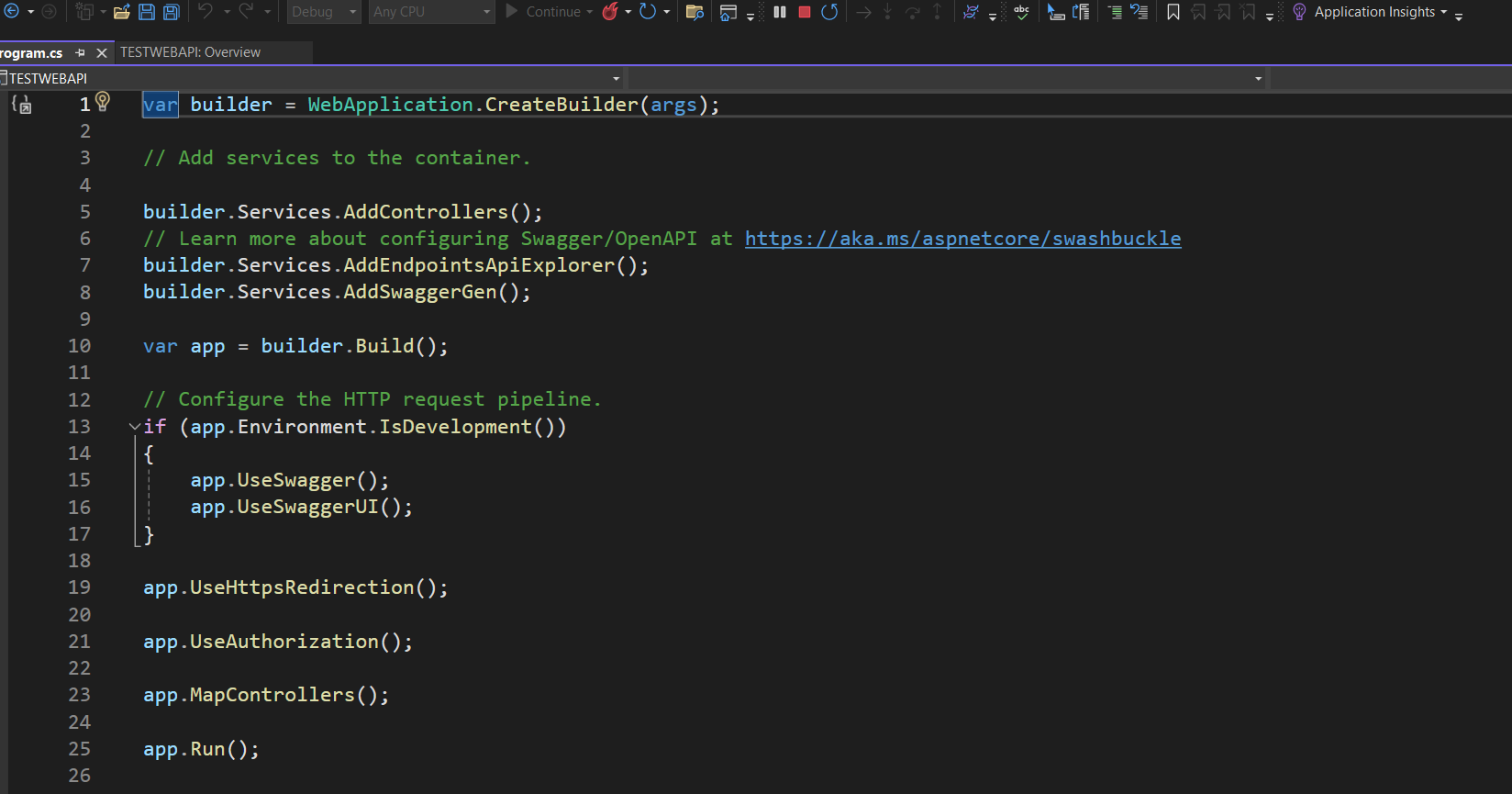
4. Определите другие приложения для сканирования уязвимостей для веб-приложений.

1. **Сканирование уязвимостей**

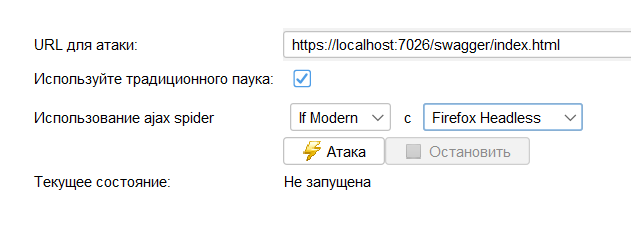
Установим OWASP-ZAP

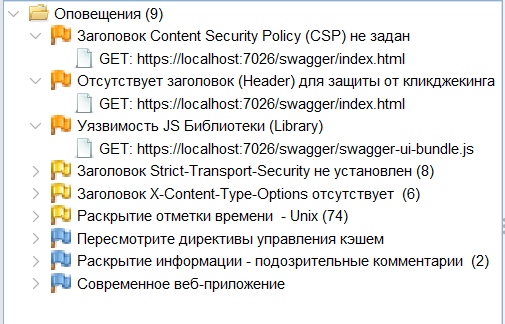


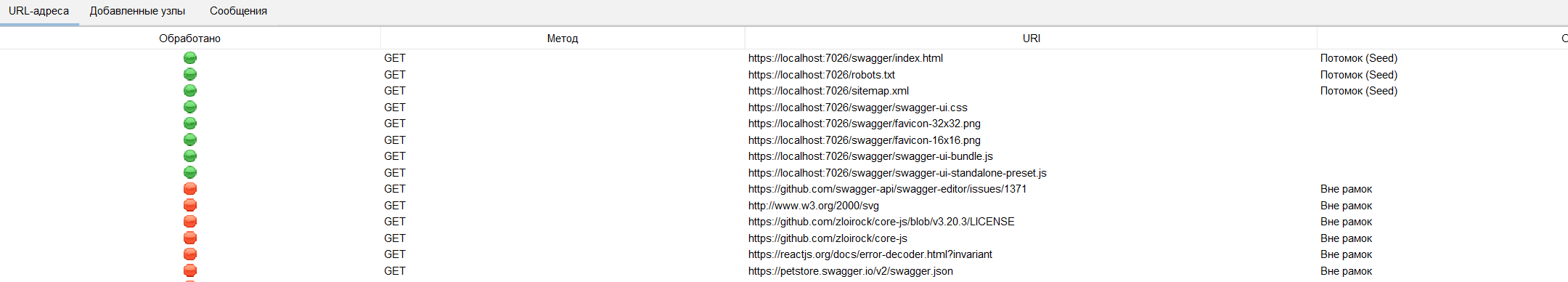
Создадим шаблон Web приложения на C#



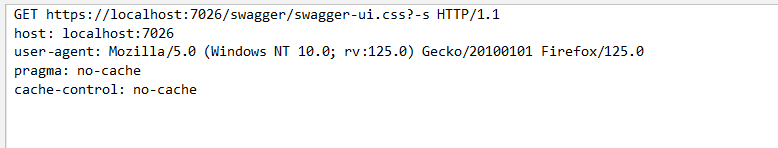
Просканируем его

****

****

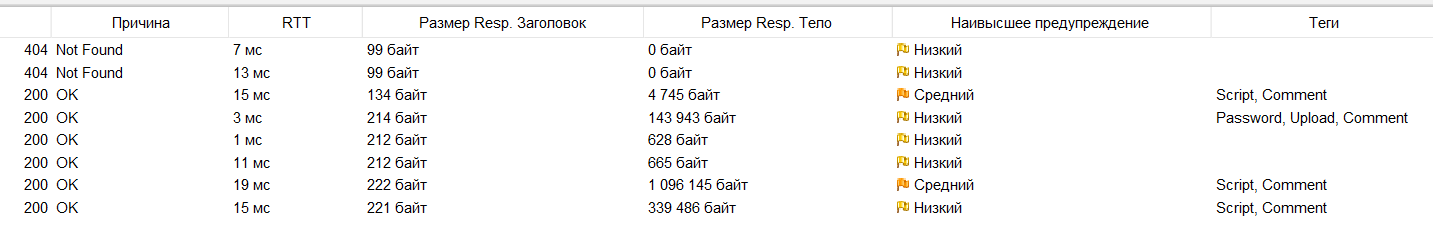
****

Как видно, ZAP произвёл множество запросов и определил уязвимости. Произведённые запросы можно просмотреть, а также увидеть ответы на них.

****

****

ZAP даёт подробную информацию о запросах и предупреждает об уязвимостях.

****

**2. Определение уязвимостей и их описание**

Основные уязвимости, которые могут быть обнаружены при сканировании веб-приложений с использованием OWASP ZAP:

* SQL Injection (SQLi): Уязвимость, при которой вредоносный SQL-код может быть внедрен в запрос к базе данных, позволяя атакующему выполнить произвольные команды.

Пример: Запрос в URL или форму, которая не фильтрует ввод пользователя.

* Cross-Site Scripting (XSS): Уязвимость, при которой злоумышленник может внедрить вредоносный скрипт в веб-страницу, чтобы выполнить его на стороне пользователя.

Пример: Ввод в поле формы или в URL, который отображается на странице без фильтрации.

* Cross-Site Request Forgery (CSRF): Уязвимость, при которой злоумышленник может заставить пользователя выполнить нежелательные действия на веб-приложении, используя авторизацию жертвы.

Пример: Отправка фальшивого запроса через злоумышленный сайт.

* Insecure Deserialization: Уязвимость, связанная с неправильной обработкой сериализованных объектов, что позволяет атакующему выполнить произвольный код.

Пример: Веб-приложение принимает и десериализует данные от пользователя без проверки их безопасности.

* Broken Authentication and Session Management: Проблемы с управлением сессиями, такие как слабые пароли, уязвимости в механизмах сессий, которые могут быть использованы для захвата учетных данных.

**3. Методы решения выявленных проблем**

Для каждой из уязвимостей, обнаруженных в процессе сканирования, можно применить следующие методы решения:

* SQL Injection: Использование подготовленных запросов (prepared statements), параметрических запросов и фильтрации ввода для предотвращения внедрения SQL-кода.
* Cross-Site Scripting (XSS): Внедрение механизмов экранирования вывода (например, HTML-экранирование) и использование Content Security Policy (CSP), чтобы ограничить выполнение небезопасных скриптов.
* Cross-Site Request Forgery (CSRF): Внедрение токенов CSRF, которые проверяют подлинность запросов от пользователя, чтобы предотвратить выполнение непреднамеренных действий.
* Insecure Deserialization: Использование безопасных методов сериализации и десериализации, проверка целостности и аутентичности данных, а также отказ от использования небезопасных форматов данных.
* Broken Authentication: Использование сильных методов аутентификации, таких как многофакторная аутентификация (MFA), а также правильное управление сессиями, например, с использованием безопасных cookies.

**4. Другие приложения для сканирования уязвимостей в веб-приложениях**

Кроме инструментов от OWASP, существует ряд других приложений, которые могут быть использованы для сканирования уязвимостей веб-приложений:

* Burp Suite: Мощный инструмент для тестирования безопасности веб-приложений. Включает в себя прокси-сервер для перехвата трафика, сканер уязвимостей, а также возможности для ручного тестирования.
* Acunetix: Коммерческий инструмент для автоматического сканирования уязвимостей веб-приложений. Он обнаруживает более 7000 различных уязвимостей.
* Nessus: Один из самых популярных инструментов для сканирования уязвимостей, который может анализировать как веб-приложения, так и инфраструктуру.
* Nikto: Простой инструмент с открытым исходным кодом, который сканирует веб-серверы на наличие уязвимостей.

**5. Выводы**

1 **Сканирование уязвимостей в веб-приложениях**:

* Использование инструмента **OWASP ZAP** позволило провести автоматическое сканирование веб-приложения на наличие популярных уязвимостей, таких как SQL-инъекции, XSS, CSRF и других.
* Результаты сканирования показали наличие уязвимостей, которые могут угрожать безопасности приложений, таких как инъекции SQL, внедрение вредоносных скриптов и уязвимости, связанные с неправильной обработкой пользовательского ввода.

2 **Определение и описание уязвимостей**:

* В ходе работы были выявлены основные уязвимости, такие как **SQL Injection**, **Cross-Site Scripting (XSS)**, **Cross-Site Request Forgery (CSRF)** и другие. Каждая из этих уязвимостей представляет собой потенциальную угрозу для безопасности веб-приложений, и для каждой из них были предложены решения для минимизации рисков.

3 **Методы решения уязвимостей**:

* Для каждой уязвимости были предложены методы защиты, такие как использование **подготовленных запросов** для предотвращения SQL-инъекций, **экранирование вывода** для защиты от XSS, использование **токенов CSRF** для защиты от подмены запросов и другие меры.
* Также были рекомендованы практики, связанные с улучшением **управления сессиями** и **аутентификацией** пользователей, такие как внедрение **многофакторной аутентификации** и безопасных cookie.

4 **Другие инструменты для сканирования уязвимостей**:

* Помимо **OWASP ZAP**, были рассмотрены и другие инструменты для тестирования безопасности веб-приложений, такие как **Burp Suite**, **Acunetix**, **Nessus** и **Nikto**. Каждый из этих инструментов предлагает различные возможности для обнаружения уязвимостей и может быть использован в зависимости от конкретных потребностей и задач.

**6. Библиография**

https://www.zaproxy.org/

<https://en.wikipedia.org/wiki/SQL_injection>

https://ru.wikipedia.org/wiki/%D0%9C%D0%B5%D0%B6%D1%81%D0%B0%D0%B9%D1%82%D0%BE%D0%B2%D0%B0%D1%8F\_%D0%BF%D0%BE%D0%B4%D0%B4%D0%B5%D0%BB%D0%BA%D0%B0\_%D0%B7%D0%B0%D0%BF%D1%80%D0%BE%D1%81%D0%B0