Молдавский Государственный Университет
Факультет Математики и Информатики
Департамент Информатики

Лабораторная работа №5
по курсу “Securitatea Sistemelor Informatice”

Тема: “Инструменты сканирования уязвимостей OWASP-ZAP”

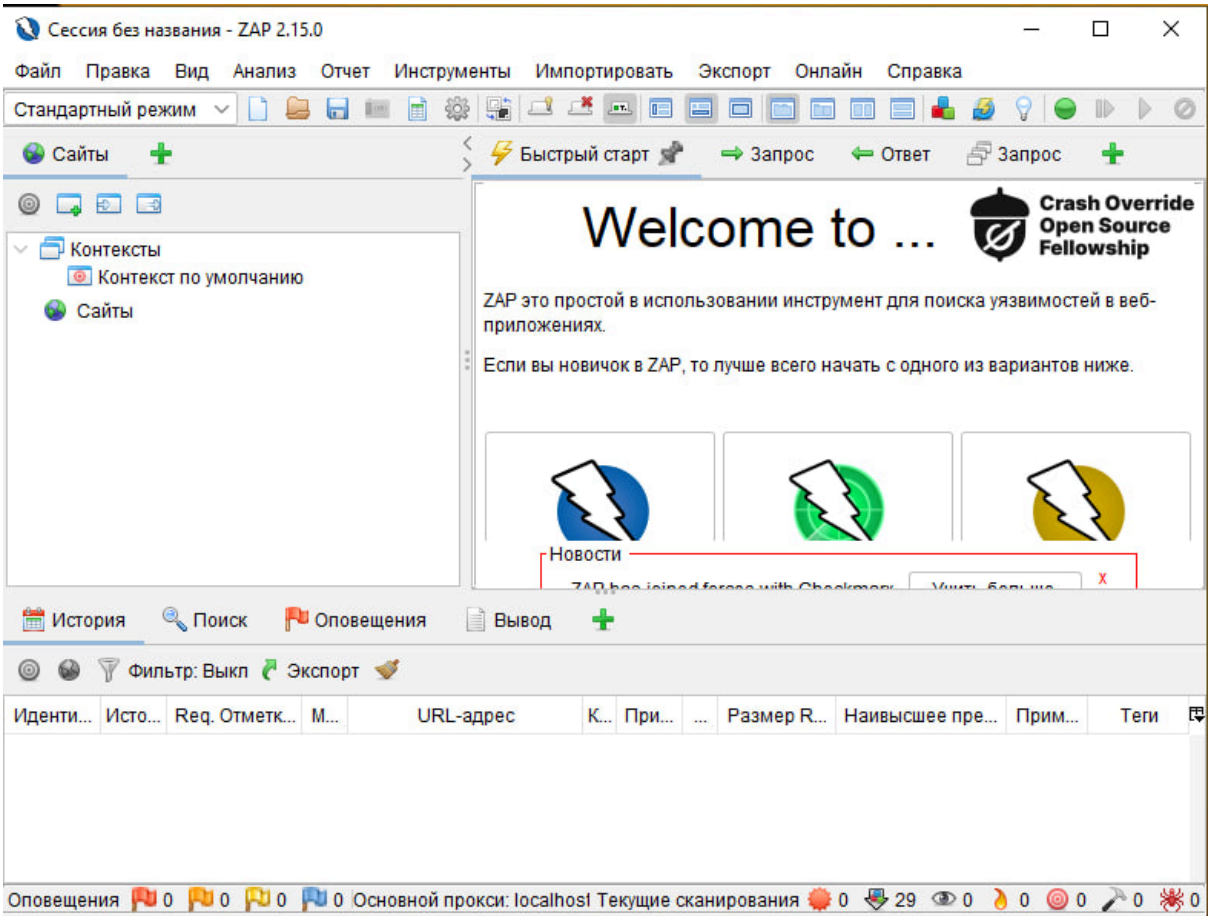
Выполнил: Slavov Constantin,
студент группы I2302
Проверила: L. Novac,
doctor conferențiar universitar

Кишинев, 2024

- **Введение:** Целью данной лабораторной работы является изучение функционала OWASP-ZAP и применения этих знаний на практике. Будет произведен ряд действий по обнаружению уязвимостей некоторых сайтов, а также их описание. Также будут описаны некоторые типы уязвимостей, какими они бывают и как их решить. И напоследок будет приведен список дополнительных приложений для сканирования уязвимостей.

Ход работы: Для начала пару слов про OWASP ZAP (Zed Attack Proxy) — это бесплатный и открытый инструмент для тестирования безопасности веб-приложений. Разработанный проектом OWASP, ZAP помогает обнаруживать уязвимости, такие как SQL Injection, XSS, CSRF и другие. Инструмент поддерживает автоматическое сканирование и предоставляет удобные средства для ручного тестирования, включая перехват и модификацию запросов, анализ ответа, а также отчеты о найденных уязвимостях. ZAP подходит как для начинающих так и опытных специалистов.

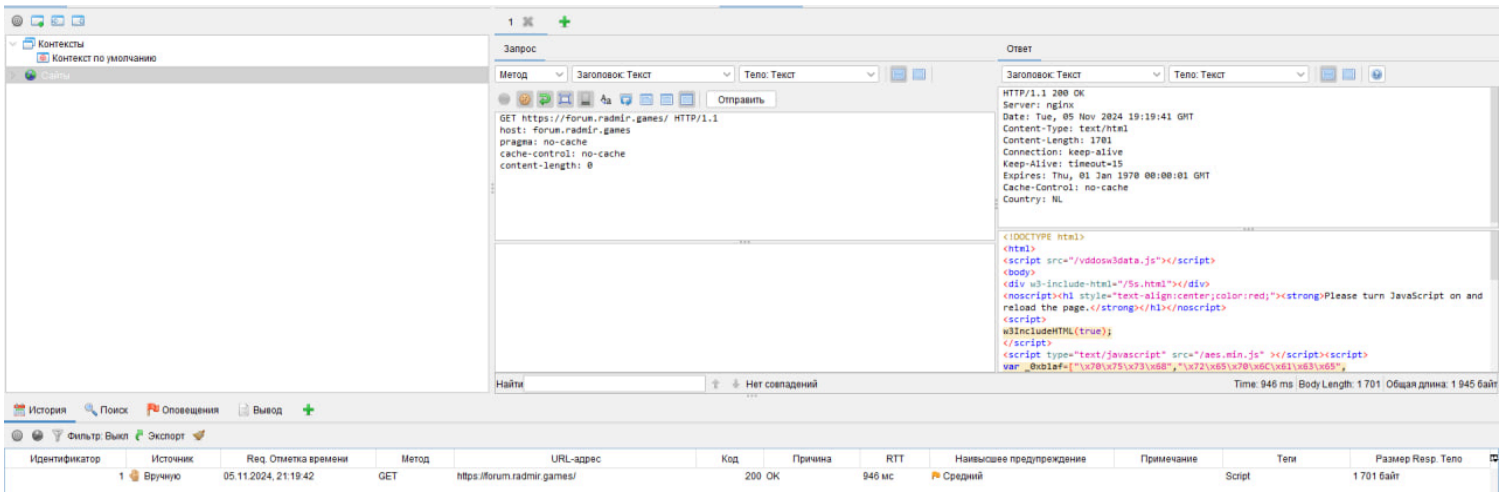
Перед выполнением лабораторной работы я установил OWASP ZAP на свой компьютер с официального сайта. Затем, зайдя в программу можно заметить огромнейший список функции и разнообразное меню.



В программе можно выбрать метод ручного сканирования, а также метод активного сканирования. Для ручного сканирования необходимо зайти во вкладку “Запрос”. Здесь можно выбрать один из предложенных методов GET (является выбранным методом по умолчанию), POST, PUT, DELETE, HEAD, OPTIONS и т.д. После составления HTTP-запроса, необходимо нажать на кнопку «Отправить», и приложение само начнет искать уязвимости в веб-приложении. Но это – вариант для ручного поиска.

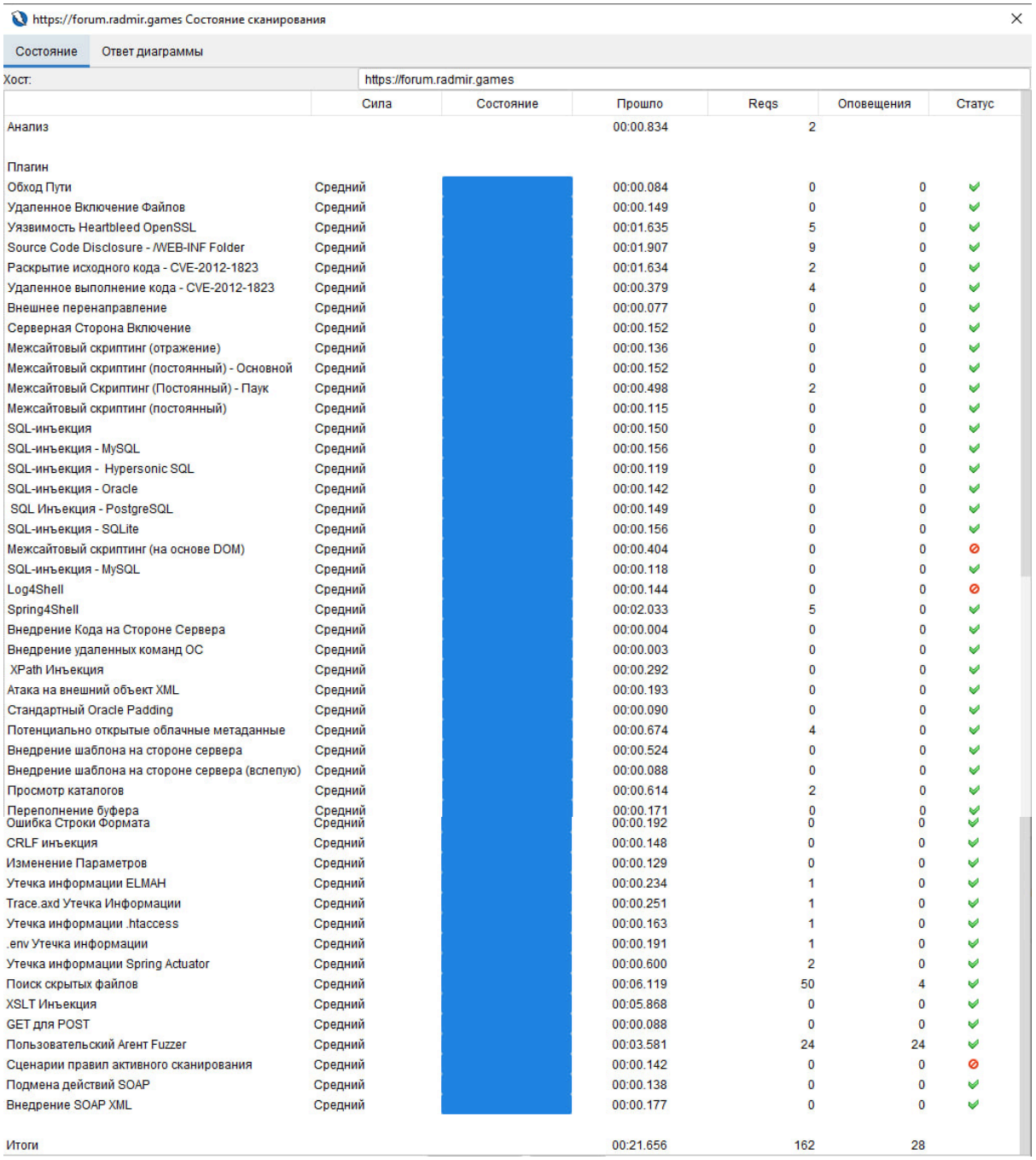
Для активного сканирования необходимо указать лишь URL-адрес, после чего начнется полная проверка на уязвимость для проверки всех необходимых параметров. После окончания проверки, ее результат можно будет посмотреть на экране.
В свою очередь я провел оба типа проверок и сейчас поделюсь результатами.

Результат ручного сканирования сайта forum.radmir.games:



Результат сканирования практически сразу показался на экране. Можно заметить, что уровень уязвимости средний, что не так уж и плохо, но стоит исправить некоторые моменты для еще лучшей защиты веб-страницы.

Результаты активного сканирования сайта forum.radmir.games:



https://forum.radmir.games Состояние сканирования

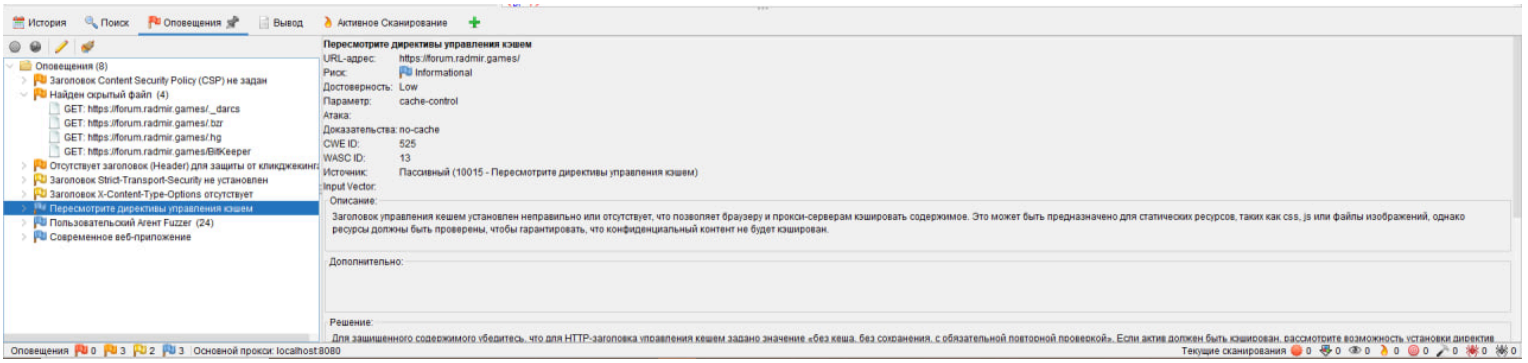
Состояние | Ответ диаграммы

Хост: https://forum.radmir.games

	Сила	Состояние	Прошло	Reqs	Оповещения	Статус
Анализ			00:00.834	2		
Плагин						
Обход Пути	Средний		00:00.084	0	0	✓
Удаленное Включение Файлов	Средний		00:00.149	0	0	✓
Уязвимость Heartbleed OpenSSL	Средний		00:01.635	5	0	✓
Source Code Disclosure - /WEB-INF Folder	Средний		00:01.907	9	0	✓
Раскрытие исходного кода - CVE-2012-1823	Средний		00:01.634	2	0	✓
Удаленное выполнение кода - CVE-2012-1823	Средний		00:00.379	4	0	✓
Внешнее перенаправление	Средний		00:00.077	0	0	✓
Серверная Сторона Включение	Средний		00:00.152	0	0	✓
Межсайтовый скриптинг (отражение)	Средний		00:00.136	0	0	✓
Межсайтовый скриптинг (постоянный) - Основной	Средний		00:00.152	0	0	✓
Межсайтовый Скриптинг (Постоянный) - Паук	Средний		00:00.498	2	0	✓
Межсайтовый скриптинг (постоянный)	Средний		00:00.115	0	0	✓
SQL-инъекция	Средний		00:00.150	0	0	✓
SQL-инъекция - MySQL	Средний		00:00.156	0	0	✓
SQL-инъекция - Hypersonic SQL	Средний		00:00.119	0	0	✓
SQL-инъекция - Oracle	Средний		00:00.142	0	0	✓
SQL Инъекция - PostgreSQL	Средний		00:00.149	0	0	✓
SQL-инъекция - SQLite	Средний		00:00.156	0	0	✓
Межсайтовый скриптинг (на основе DOM)	Средний		00:00.404	0	0	⊘
SQL-инъекция - MySQL	Средний		00:00.118	0	0	✓
Log4Shell	Средний		00:00.144	0	0	⊘
Spring4Shell	Средний		00:02.033	5	0	✓
Внедрение Кода на Стороне Сервера	Средний		00:00.004	0	0	✓
Внедрение удаленных команд ОС	Средний		00:00.003	0	0	✓
XPath Инъекция	Средний		00:00.292	0	0	✓
Атака на внешний объект XML	Средний		00:00.193	0	0	✓
Стандартный Oracle Padding	Средний		00:00.090	0	0	✓
Потенциально открытые облачные метаданные	Средний		00:00.674	4	0	✓
Внедрение шаблона на стороне сервера	Средний		00:00.524	0	0	✓
Внедрение шаблона на стороне сервера (вслепую)	Средний		00:00.088	0	0	✓
Просмотр каталогов	Средний		00:00.614	2	0	✓
Переполнение буфера	Средний		00:00.171	0	0	✓
Ошибка Строки Формата	Средний		00:00.192	0	0	✓
CRLF инъекция	Средний		00:00.148	0	0	✓
Изменение Параметров	Средний		00:00.129	0	0	✓
Утечка информации ELMAH	Средний		00:00.234	1	0	✓
Trace.axd Утечка Информации	Средний		00:00.251	1	0	✓
Утечка информации .htaccess	Средний		00:00.163	1	0	✓
.env Утечка информации	Средний		00:00.191	1	0	✓
Утечка информации Spring Actuator	Средний		00:00.600	2	0	✓
Поиск скрытых файлов	Средний		00:06.119	50	4	✓
XSLT Инъекция	Средний		00:05.868	0	0	✓
GET для POST	Средний		00:00.088	0	0	✓
Пользовательский Агент Fuzzer	Средний		00:03.581	24	24	✓
Сценарии правил активного сканирования	Средний		00:00.142	0	0	⊘
Подмена действий SOAP	Средний		00:00.138	0	0	✓
Внедрение SOAP XML	Средний		00:00.177	0	0	✓
Итоги			00:21.656	162	28	

Активное сканирование показывает полный список проведенных проверок и возможный риск уязвимости. Можно заметить, что почти по всем параметрам данная веб-страница проходит проверку, лишь по некоторым имеются проблемы, что является хорошим показателем защищенности веб-страницы.

Зайдя во вкладку “Оповещения” можно найти более подробную информацию по возможным уязвимостям. Вот некоторые из них для сайта forum.radmir.games:

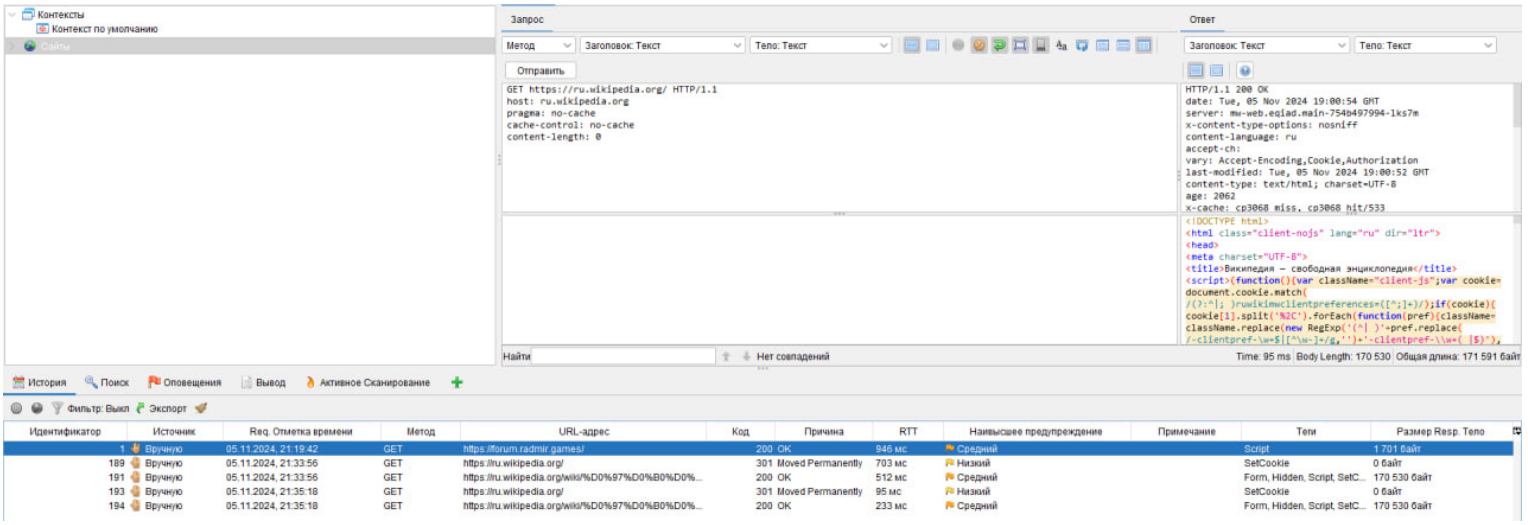


Можно заметить, что по некоторым параметрам сайт имеет достоверность “Low”. Уровень достоверности “Low” (низкий) в контексте безопасности обычно означает, что система или инструмент оценивает вероятность наличия уязвимости как низкую. Это не очень хорошо, так как низкая достоверность указывает на то, что обнаруженная уязвимость может быть ложной или неточной.

В то же время, уровень достоверности “Low” не всегда говорит о реальной опасности — это лишь оценка, которая указывает, что нужно проводить дополнительную проверку, чтобы подтвердить или опровергнуть наличие уязвимости.

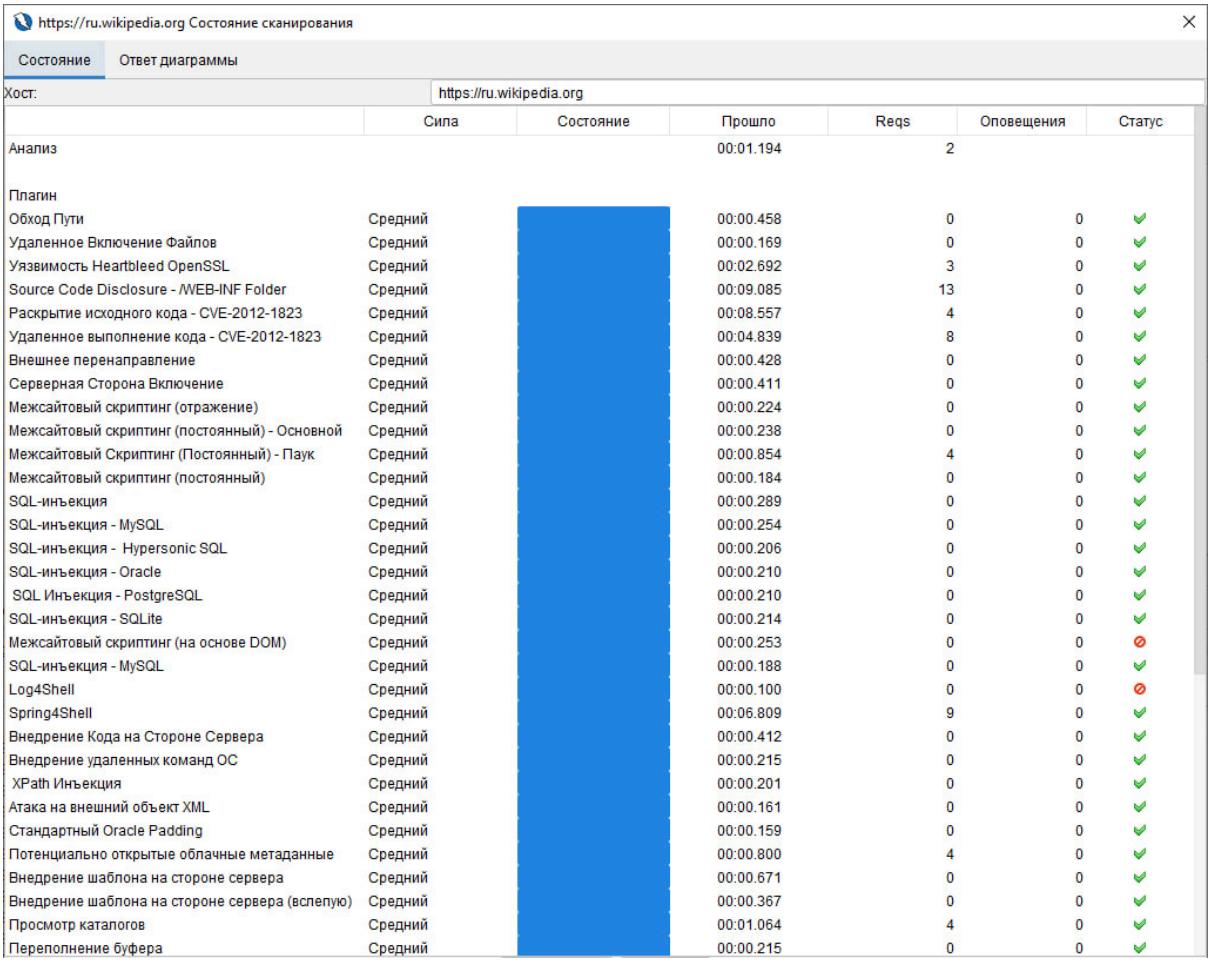
Таким образом можно понять как правильно проверить веб-страницу на уязвимости и выявить проблемы. Проделаю те же самые действия с веб-страницей Википедии.

Для начала проведу ручную проверку сайта Wikipedia:



Сканирование проходит в том же порядке, как и с первым сайтом. Результаты появились на экране практически сразу после проведения процедуры.

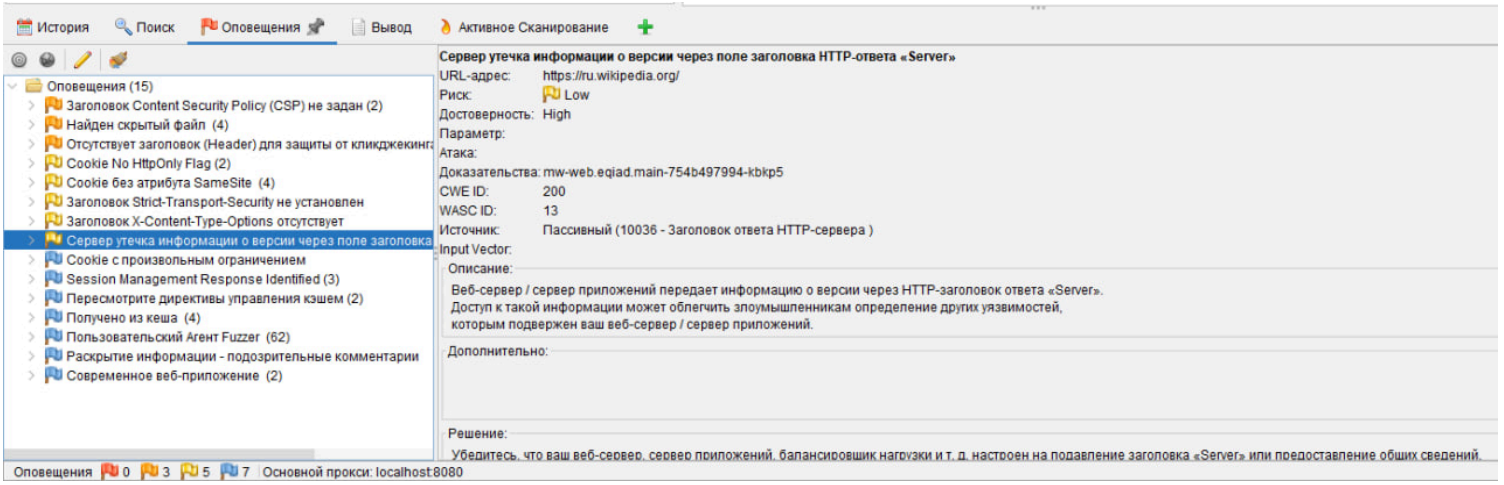
Результаты активного сканирования сайта Wikipedia:

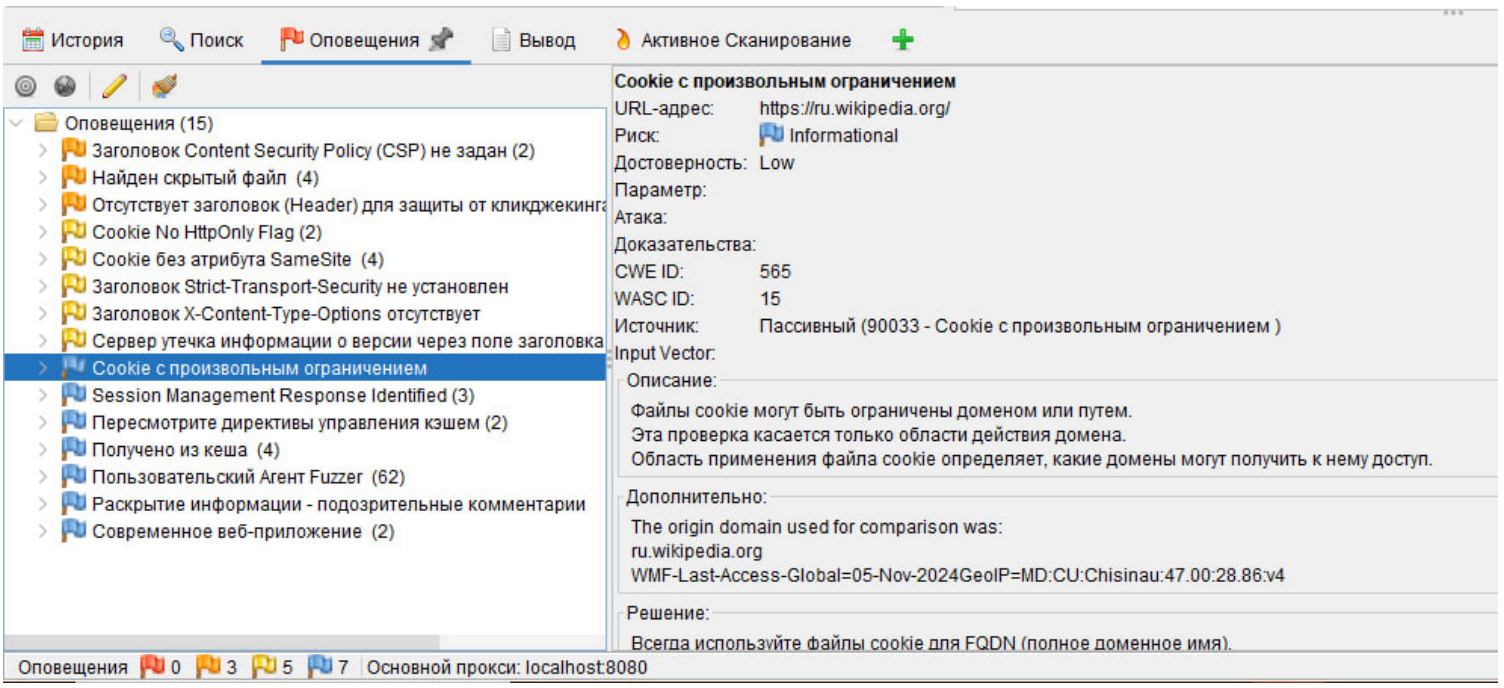


	Сила	Состояние	Прошло	Reqs	Оповещения	Статус
Анализ			00:01.194	2		
Плагин						
Обход Пути	Средний		00:00.458	0	0	✓
Удаленное Включение Файлов	Средний		00:00.169	0	0	✓
Уязвимость Heartbleed OpenSSL	Средний		00:02.692	3	0	✓
Source Code Disclosure - /WEB-INF Folder	Средний		00:09.085	13	0	✓
Раскрытие исходного кода - CVE-2012-1823	Средний		00:08.557	4	0	✓
Удаленное выполнение кода - CVE-2012-1823	Средний		00:04.839	8	0	✓
Внешнее перенаправление	Средний		00:00.428	0	0	✓
Серверная Сторона Включение	Средний		00:00.411	0	0	✓
Межсайтовый скриптинг (отражение)	Средний		00:00.224	0	0	✓
Межсайтовый скриптинг (постоянный) - Основной	Средний		00:00.238	0	0	✓
Межсайтовый Скриптинг (Постоянный) - Паук	Средний		00:00.854	4	0	✓
Межсайтовый скриптинг (постоянный)	Средний		00:00.184	0	0	✓
SQL-инъекция	Средний		00:00.289	0	0	✓
SQL-инъекция - MySQL	Средний		00:00.254	0	0	✓
SQL-инъекция - Hypersonic SQL	Средний		00:00.206	0	0	✓
SQL-инъекция - Oracle	Средний		00:00.210	0	0	✓
SQL Инъекция - PostgreSQL	Средний		00:00.210	0	0	✓
SQL-инъекция - SQLite	Средний		00:00.214	0	0	✓
Межсайтовый скриптинг (на основе DOM)	Средний		00:00.253	0	0	⊘
SQL-инъекция - MySQL	Средний		00:00.188	0	0	✓
Log4Shell	Средний		00:00.100	0	0	⊘
Spring4Shell	Средний		00:06.809	9	0	✓
Внедрение Кода на Стороне Сервера	Средний		00:00.412	0	0	✓
Внедрение удаленных команд ОС	Средний		00:00.215	0	0	✓
XPath Инъекция	Средний		00:00.201	0	0	✓
Атака на внешний объект XML	Средний		00:00.161	0	0	✓
Стандартный Oracle Padding	Средний		00:00.159	0	0	✓
Потенциально открытые облачные метаданные	Средний		00:00.800	4	0	✓
Внедрение шаблона на стороне сервера	Средний		00:00.671	0	0	✓
Внедрение шаблона на стороне сервера (вслепую)	Средний		00:00.367	0	0	✓
Просмотр каталогов	Средний		00:01.064	4	0	✓
Переполнение буфера	Средний		00:00.215	0	0	✓

Ошибка Строки Формата	Средний		00:00.200	0	0
CRLF инъекция	Средний		00:00.251	0	0
Изменение Параметров	Средний		00:00.249	0	0
Утечка информации ELMAH	Средний		00:00.417	1	0
Trace.axd Утечка Информации	Средний		00:00.460	2	0
Утечка информации .htaccess	Средний		00:00.534	2	0
.env Утечка информации	Средний		00:00.901	2	0
Утечка информации Spring Actuator	Средний		00:01.027	2	0
Поиск скрытых файлов	Средний		00:21.150	50	0
XSLT Инъекция	Средний		00:20.740	0	0
GET для POST	Средний		00:00.370	0	0
Пользовательский Агент Fuzzer	Средний		00:16.786	48	38
Сценарии правил активного сканирования	Средний		00:00.146	0	0
Подмена действий SOAP	Средний		00:00.204	0	0
Внедрение SOAP XML	Средний		00:00.195	0	0
Итоги			01:11.977	244	38

Активное сканирование не выявило большого количества проблем с уязвимостью, но опять без них не обошлось. Давайте посмотрим, что есть в оповещениях.





В оповещениях в некоторых случаях можно заметить уровень достоверности “Low” и “High”. Уровень достоверности High (высокий) — это хорошо, так как он указывает на высокую вероятность того, что обнаруженная уязвимость действительно существует и требует внимания. Это означает, что инструмент или система уверены в точности обнаружения уязвимости, и её нужно срочно проверять и устранять.
Высокая достоверность помогает сократить количество ложных срабатываний и позволяет сосредоточиться на реальных проблемах, которые могут представлять угрозу для безопасности системы.

Таким образом, я проверил эти две веб-страницы на уязвимости и смог понять как все это работает.

- Определите, какие уязвимости встречаются, и опишите их.

Вот несколько распространённых уязвимостей, которые часто встречаются в веб-приложениях:

1. SQL Injection

SQL Injection возникает, когда пользовательский ввод напрямую попадает в SQL-запрос без должной обработки. Злоумышленник может внедрить вредоносный код, чтобы получить доступ к данным, изменить их или даже удалить. Например, при вводе специальных символов и команд в поле поиска он может получить конфиденциальные данные, к которым у него нет доступа.

2. Cross-Site Scripting (XSS)

XSS позволяет злоумышленнику внедрить вредоносный скрипт (обычно JavaScript) на страницу, отображаемую другим пользователям. Эта уязвимость возникает, когда приложение не фильтрует или не экранирует пользовательский ввод перед отображением его на странице. Злоумышленник может использовать XSS для кражи данных пользователей, например, сессий или учетных данных.

3. Cross-Site Request Forgery (CSRF)

CSRF-атака заставляет авторизованного пользователя выполнить нежелательное действие на сайте, например, изменить настройки учетной записи или отправить данные. Она использует авторизацию пользователя без его ведома и обычно происходит через скрытые формы или ссылки на сторонних ресурсах.

4. Недостатки управления сессиями

Эти уязвимости связаны с неправильным управлением сессиями и куками. Если сессии не защищены должным образом (например, если идентификаторы сессий передаются по HTTP или не обновляются после входа), злоумышленник может получить доступ к данным пользователя, перехватив его сессию.

Эти уязвимости могут привести к серьезным последствиям, таким как утечка данных, кража учетных записей и нарушение конфиденциальности.

- Каковы методы решения тех проблем, которые вызваны определенными уязвимостями?

Для решения распространённых уязвимостей в веб-приложениях применяют следующие методы:

1. SQL Injection

- Использование подготовленных запросов (prepared statements): они исключают возможность внедрения кода, так как данные пользователя передаются отдельно от SQL-команд.
- Применение ORM (Object-Relational Mapping): ORM-библиотеки автоматически обрабатывают запросы и защищают от SQL-инъекций.
- Экранизированный ввод пользователя: проверка и фильтр всех данных, особенно в формах ввода.

2. Cross-Site Scripting (XSS)

- Экранизированный вывод на страницу: замена специальных символов на HTML-эквиваленты (например, `<` на `<`), чтобы предотвратить выполнение кода.

- Использование Content Security Policy (CSP): эта политика ограничивает выполнение непроверенных скриптов на страницах.
- Проверка и очистка данных: фильтр и проверка данных, особенно при выводе динамического контента.

3. Cross-Site Request Forgery (CSRF)
- Применение CSRF-токенов: они генерируются для каждого запроса и проверяют, что действие выполняет авторизованный пользователь.
- Проверка заголовков реферера: убедитесь, что запросы отправляются с доверенных доменов.
- Ограничение жизненного цикла сессий: это уменьшит вероятность использования украденной сессии в CSRF-атаке.

4. Недостатки управления сессиями
- Использование безопасных куки (HTTPOnly и Secure): это защитит куки от кражи через XSS и их передачу по небезопасным каналам.
- Обновление идентификаторов сессий: меняйте идентификатор после авторизации пользователя для предотвращения кражи сессий.
- Настройка истечение срока сессий: завершение сессий через определенное время поможет снизить риск их компрометации.

Эти методы помогают защитить веб-приложения и значительно снижают риск эксплуатации уязвимостей.

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

Помимо OWASP ZAP, для сканирования уязвимостей в веб-приложениях используются и другие инструменты:

1. Burp Suite

Это один из самых мощных инструментов для тестирования безопасности. Burp Suite предлагает функции для автоматического и ручного сканирования уязвимостей, включая перехват трафика, тестирование на SQL Injection, XSS и другие уязвимости. Инструмент широко используется профессиональными пентестерами, так как имеет мощные плагины и гибкую настройку.

2. Acunetix

Acunetix — это автоматизированный сканер уязвимостей, который выявляет более 6500 известных уязвимостей, таких как SQL Injection, XSS, CSRF и другие. Он поддерживает сканирование как веб-приложений, так и API, предоставляет подробные отчеты и интегрируется с системами разработки (CI/CD) для автоматического анализа кода.

3. Netsparker

Netsparker — еще один автоматизированный сканер, известный высокой точностью обнаружения уязвимостей. Он проверяет уязвимости, такие как SQL Injection и XSS, и может подтвердить их, избегая ложных срабатываний. Netsparker подходит как для небольших, так и для крупных проектов и поддерживает интеграцию с различными DevOps-инструментами.

Эти инструменты предлагают разнообразные функции для анализа и автоматизации, что упрощает обнаружение и устранение уязвимостей в веб-приложениях.

- Вывод: В результате проделанной работы я использовал инструменты OWASP для сканирования уязвимостей в нескольких веб-приложениях и узнал много нового о возможных угрозах и способах их устранения. Проведя анализ, я выявил распространённые уязвимости, такие как SQL Injection, XSS и CSRF, которые часто встречаются в веб-приложениях, и разобрался в том, как каждая из них может повлиять на безопасность системы.

Изучив методы решения этих проблем, я понял, как важно применять такие подходы, как экранирование данных, использование CSRF-токенов и настройка строгих политик безопасности, чтобы минимизировать риски. Также я ознакомился с другими инструментами для сканирования уязвимостей, такими как Burp Suite, Acunetix и Netsparker, которые могут быть полезны для комплексной проверки безопасности веб-приложений.

Эта работа помогла мне лучше понять основы безопасности веб-приложений, а также важность регулярного сканирования и устранения уязвимостей, чтобы обеспечить защиту данных и пользователей.

- Библиография:

1. OWASP Foundation. (2023). OWASP Zed Attack Proxy Project. Retrieved from https://owasp.org/www-project-zap/

Источник, описывающий основные функции и возможности OWASP ZAP — одного из самых популярных инструментов для тестирования безопасности веб-приложений.

2. Acunetix. (2023). Web Vulnerability Scanner. Retrieved from https://www.acunetix.com/

Руководство по использованию Acunetix, автоматизированного инструмента для обнаружения уязвимостей, поддерживающего интеграцию с DevOps и CI/CD.

3. Netsparker. (2023). Netsparker Web Application Security Scanner. Retrieved from https://www.netsparker.com/

Описание возможностей Netsparker, инструмента для автоматического обнаружения уязвимостей с подтверждением обнаруженных проблем.

4. PortSwigger Ltd. (2023). Burp Suite Documentation. Retrieved from https://portswigger.net/burp/documentation

Официальная документация Burp Suite, содержащая информацию о функциях для тестирования безопасности, таких как перехват трафика и анализ на уязвимости.

5. OWASP Foundation. (2023). OWASP Top Ten 2021: The Ten Most Critical Web Application Security Risks. Retrieved from https://owasp.org/www-project-top-ten/

Документ OWASP Top Ten, описывающий десять самых критических уязвимостей веб-приложений и подходы к их предотвращению.