УДК 004.056, 004.75

РАЗРАБОТКА ФРЕЙМВОРКА ДЛЯ ОЦЕНКИ ТЕХНИЧЕСКОЙ ЗАЩИЩЁННОСТИ WEB-РЕСУРСОВ ПО РЕКОМЕНДАЦИЯМ OWASP

Соколов Илья Сергеевич[1], Шевченко Алексей Валерьевич[2]

[1]Специалист по защите информации;
АО «ОКБ «Аэрокосмические системы»;
Россия, 141893, Московская область, г. Дубна, ул. Программистов, 4;
e-mail: ilya20834@gmail.com.

[2]Аспирант;
Государственный университет «Дубна»;
Россия, 141980, Московская обл., г. Дубна, ул. Университетская, 19;
e-mail: leviathan0909@gmail.com.

В статье описан процесс разработки модульного фреймворка для оценки технической защищённости веб-приложений на основе рекомендаций OWASP. Проведён анализ актуальных угроз, рассмотрены методики OWASP Top 10, а также существующие инструменты сканирования. Предложенная архитектура включает модули сбора данных, проверки уязвимостей (SQL-инъекции, XSS, API-ошибки) и генерации отчётов с использованием языковой модели. Разработанное решение обеспечивает автоматизацию процессов аудита безопасности и может применяться в учебных, исследовательских и практических целях.

Ключевые слова: OWASP, фреймворк, уязвимости, SQL-инъекция, XSS, безопасность веб-приложений, автоматизация аудита, информационная безопасность.

Для цитирования:

Соколов И. С., Шевченко А. В. Разработка фреймворка для оценки технической защищённости WEB-ресурсов по рекомендациям OWASP // Системный анализ в науке и образовании: сетевое научное издание. 2025. № 3. С. 42-51. EDN: LKSJVJ. URL: https://sanse.ru/index.php/sanse/article/view/680.

DEVELOPMENT OF A FRAMEWORK FOR EVALUATING THE TECHNICAL SECURITY OF WEB RESOURCES BASED ON OWASP RECOMMENDATIONS

Sokolov Ilya S.[1], Shevchenko Alexey V.[2]

[1] Information Security Specialist;
JSC "OKB "Aerospace Systems";
4 Programistov Str., Dubna, Moscow region, 141983, Russia;
e-mail: ilya20834@gmail.com.

[2] PhD student;
Dubna State University;
19 Universitetskaya Str., Dubna, Moscow region, 141980, Russia;
e-mail: leviathan0909@gmail.com

The article describes the process of developing a modular framework for evaluating the technical security of web applications based on OWASP recommendations. The analysis of current threats is carried out, OWASP Top 10 techniques are considered, as well as existing scanning tools. The proposed architecture includes modules for data collection, vulnerability checking (SQL injection, XSS, API errors) and report generation using the language model. The developed solution provides automation of security audit processes and can be used for educational, research and practical purposes.

Keywords: OWASP, framework, vulnerabilities, SQL injection, XSS, web application security, audit automation, information security.

For citation:
Sokolov I. S., Shevchenko A. V. Development of a framework for assessing the technical security of WEB resources based on OWASP recommendations. *System analysis in science and education*, 2025;(3):42–51 (in Russ). EDN: LKSJVJ. Available from: https://sanse.ru/index.php/sanse/article/view/680.

Введение

В условиях информационного общества значительная часть деловой, социальной и государственной деятельности осуществляется через веб-приложения. Рост количества пользователей, объёма данных и функционала сопровождается увеличением числа и сложности кибератак. Поэтому защита веб-приложений от угроз безопасности – приоритетная задача как для коммерческих, так и для государственных структур.

Особое значение приобретает своевременное выявление уязвимостей, ведущих к утечке информации, финансовым потерям и потере доверия пользователей. Для систематизации основных угроз и выработки рекомендаций создан международный проект Open Web Application Security Project (OWASP), регулярно публикующий аналитические отчёты [1]. Наиболее известен из них список OWASP Top 10 – перечень наиболее серьёзных уязвимостей веб-приложений.

По данным OWASP 94% современных веб-приложений подвержены критическим уязвимостям, в первую очередь – внедрению кода [2]. Успешные атаки могут привести к компрометации конфиденциальной информации, нарушению прав субъектов персональных данных, финансовым потерям и репутационным рискам. В соответствии с ФЗ-149 «Об информации, информационных технологиях и о защите информации» и ФЗ-152 «О персональных данных» владельцы веб-ресурсов обязаны обеспечивать защиту обрабатываемых данных. Поэтому разработка эффективных средств обнаружения уязвимостей становится ключевой задачей повышения защищённости веб-приложений.

Актуальность исследования связана с дефицитом комплексных открытых решений, соответствующих рекомендациям OWASP. Существующие сканеры либо коммерческие и дорогие (например, Burp Suite Pro), либо узкоспециализированные (например, SQLmap). Создание собственного фреймворка позволит адаптировать процесс проверки под конкретные задачи, объединить функциональность различных инструментов и реализовать проверки по категориям OWASP Top 10. Такой инструмент будет полезен специалистам ИБ и системным администраторам для раннего выявления уязвимостей уже на этапах проектирования и внедрения веб-приложений.

1. Обзор и роль OWASP в сфере информационной безопасности

Проект OWASP (Open WEB Application Security Project) был создан в 2001 году группой профессионалов в сфере информационной безопасности и веб-разработки с целью борьбы с растущим количеством уязвимостей и угроз безопасности WEB-приложений. Изначально OWASP формировался как открытое сообщество, объединяющее программистов, аналитиков и специалистов по безопасности для совместной работы над созданием и распространением свободных знаний в этой области.

Основной задачей OWASP стало системное повышение уровня безопасности веб-приложений по всему миру. Важно отметить, что проект базируется на принципе полной открытости и доступности информации, поэтому все материалы, стандарты и инструменты OWASP распространяются свободно и без ограничений. Таким образом, сообщество стремится максимально снизить барьер входа и обеспечить доступ к критически важным знаниям в области WEB-безопасности каждому заинтересованному лицу или организации, независимо от их размера или финансовых возможностей. Основные цели OWASP заключаются в следующем:

1. Распространение знаний и повышение осведомлённости. *OWASP* проводит регулярные образовательные мероприятия, включая конференции, вебинары, тренинги и семинары, направленные на информирование профессионалов и организаций об актуальных угрозах и способах противодействия им.
2. Создание и развитие открытых стандартов. Сообщество разрабатывает и публикует открытые руководства и стандарты (например, *OWASP Application Security Verification Standard*, *OWASP*

Testing Guide), которые помогают компаниям и разработчикам внедрять единые и проверенные подходы к обеспечению безопасности [3].

3. Инструментарий для защиты веб-приложений. *OWASP* активно разрабатывает и поддерживает бесплатные программные инструменты, такие как *OWASP ZAP*, *OWASP Dependency Check*, *OWASP Juice Shop* и другие, которые позволяют обнаруживать и исправлять наиболее распространённые уязвимости и ошибки.
4. Формирование культуры безопасной разработки. важным аспектом работы *OWASP* является создание сообщества, в рамках которого безопасность становится не дополнительным этапом, а неотъемлемой частью процесса разработки программного обеспечения с самых ранних стадий проектирования.

Особо значимым элементом деятельности OWASP является тесное взаимодействие с регуляторами и международными организациями, такими как ISO, IEEE и национальными органами стандартизации и сертификации, включая ФСТЭК России. Благодаря этому OWASP оказывает непосредственное влияние на формирование регуляторной среды и международных стандартов, связанных с информационной безопасностью.

Благодаря своей деятельности OWASP внес значительный вклад в развитие мировой практики обеспечения безопасности WEB-приложений, став одним из ведущих источников методологических материалов и практических инструментов в данной области. Комплексное и системное внедрение разработанных сообществом стандартов и рекомендаций позволяет организациям эффективно снижать уровень уязвимостей в информационных системах и повышать их устойчивость к актуальным киберугрозам.

2. Применение рекомендаций OWASP на практике

Рекомендации *OWASP* широко используются на практике для выполнения требований действующего российского законодательства и норм ФСТЭК России, а также соответствия международным стандартам информационной безопасности:

1. Этап проектирования и разработки.

Руководство *OWASP Secure Coding Practices* позволяет внедрить подходы безопасной разработки [4]. Которые в свою очередь удовлетворяют требованиям приказов ФСТЭК России (например, приказ ФСТЭК №17 «О защите информации»). Также рекомендации учитываются при выполнении требований ФЗ №152 «О персональных данных».

2. Этап тестирования приложений.

Методология *OWASP Testing Guide* позволяет проверять *WEB*-приложения на наличие типовых уязвимостей [5]. Данные подходы полностью соответствуют рекомендациям ФСТЭК России по организации контроля защищенности информационных систем.

3. Эксплуатация и мониторинг.

Рекомендации *OWASP* в части журналирования и мониторинга событий безопасности (*Security Logging and Monitoring*) позволяют удовлетворить требования российского законодательства и ФСТЭК по контролю и учету событий информационной безопасности, в том числе требования ФЗ №149 «Об информации, информационных технологиях и о защите информации».

4. Обучение специалистов.

OWASP предоставляет актуальные знания и методики, которые соответствуют требованиям ФСТЭК России в области подготовки и повышения квалификации специалистов по информационной безопасности.

Использование рекомендаций *OWASP* на практике позволяет организациям эффективно выполнять требования российского законодательства, в том числе ФЗ №149, ФЗ №152, нормативных актов ФСТЭК России и международных стандартов (ГОСТ Р ИСО/МЭК 27001, 27002), существенно повышая уровень безопасности *WEB*-приложений и *WEB*-ресурсов.

3. Постановка задачи и выбор архитектурного подхода

Одной из актуальных задач в области обеспечения безопасности веб-приложений является необходимость регулярной, комплексной и воспроизводимой оценки защищённости на наличие уязвимостей, отражённых в классификациях *OWASP*. В практике информационной безопасности для этих целей применяются как коммерческие, так и *Open Source* инструменты. Однако анализ существующих решений показывает, что большинство из них ориентированы либо на решение узких задач (например, *SQL*-инъекции, межсайтовый скриптинг), либо не предоставляют достаточной гибкости при адаптации к требованиям конкретной инфраструктуры.

Формализация требований к создаваемому программному средству позволила определить предпочтительный подход к реализации. В качестве основной формы организации кода был выбран архитектурный фреймворк, представляющий собой программную платформу с заранее определённой логикой взаимодействия компонентов, правилами расширения функциональности и типовыми сценариями использования [6].

Использование фреймворка обеспечивает достижение следующих проектных целей:

- централизация логики анализа уязвимостей при сохранении независимости модулей;
- возможность поэтапного наращивания функциональности без переработки существующего кода;
- повышенная удобочитаемость архитектуры за счёт формализованной структуры компонентов;
- повторное использование и инкапсуляция ключевых функций;
- упрощение тестирования отдельных модулей и всей системы в целом.

В отличие от набора автономных утилит, фреймворк обеспечивает строгую структурную организацию, при которой модули анализа функционируют на основе единых протоколов взаимодействия, используют общие форматы данных и формируют стандартизированные отчёты. Такой подход соответствует ключевым принципам инженерии программного обеспечения, включая модульность, изоляцию зависимостей, расширяемость и повторное использование кода.

Выбор модульной архитектуры в рамках фреймворка обусловлен необходимостью обеспечения масштабируемости и адаптивности. Каждый модуль выполняет строго определённую функцию: сбор информации, анализ определённого класса уязвимостей, взаимодействие с языковой моделью и генерация отчётов. Компоненты реализованы как независимые элементы с возможностью замены, отключения или модификации без влияния на остальную часть системы. Такая структура соответствует паттерну *plugin-based architecture* и отвечает требованиям к отказоустойчивым и поддерживаемым системам.

4. Общая структура и компоненты фреймворка

Разработанный фреймворк представляет собой модульный программный комплекс, предназначенный для автоматизированного анализа *WEB*-приложений на наличие наиболее распространённых и критических уязвимостей в соответствии с рекомендациями *OWASP*. Фреймворк реализован на языке *Python* с использованием современных библиотек и инструментов, обеспечивающих высокую эффективность и гибкость анализа.

Основными компонентами разработанного фреймворка являются:

- Модуль сбора информации (*WEB Crawler*) – отвечает за автоматизированное сканирование *WEB*-ресурса, выявление структуры сайта, сбор метаинформации, форм, параметров запросов, *API*-эндпоинтов, а также данных об *SSL*-сертификатах и *IP*-адресах.
- Модуль анализа *SQL*-инъекций – автоматически проверяет все собранные формы и *URL*-параметры на наличие уязвимостей типа *SQL*-инъекции, используя специализированный набор *payloads* [7].
- Модуль анализа межсайтового скриптинга (*XSS*) – проводит тестирование всех выявленных элементов сайта, таких как формы и поля ввода, на уязвимости типа *XSS* (*reflected* и *stored*), используя соответствующие *payloads* и методы анализа ответов сервера [8].

- Модуль анализа *API*-эндпоинтов – обеспечивает детальный анализ *API*-интерфейсов, проверяя авторизацию, утечки конфиденциальной информации и наличие типовых ошибок, влияющих на безопасность.

- Модуль интеграции с локальной *LLM* (*Ollama*) – предназначен для автоматической интерпретации результатов работы остальных модулей, формирования понятных и детализированных рекомендаций, а также предоставления возможности диалогового взаимодействия с пользователем [9].

- Подсистема генерации отчётов и логирования – обеспечивает структурированное сохранение результатов проверок, автоматическое формирование отчётов в форматах *JSON, DOCX* и ведение логов всех операций фреймворка для дальнейшего анализа.

- Пользовательский интерфейс (*CLI*) – обеспечивает удобное и интуитивно понятное взаимодействие пользователя с системой посредством командной строки, позволяя задавать параметры сканирования, запускать процессы анализа и получать отчёты.

5. Принцип работы и взаимодействие компонентов

Для наглядного представления логики функционирования фреймворка приведена диаграмма бизнес-процесса, отражающая основные этапы взаимодействия между пользователем, компонентами системы и хранилищем данных (см. рис. 1).

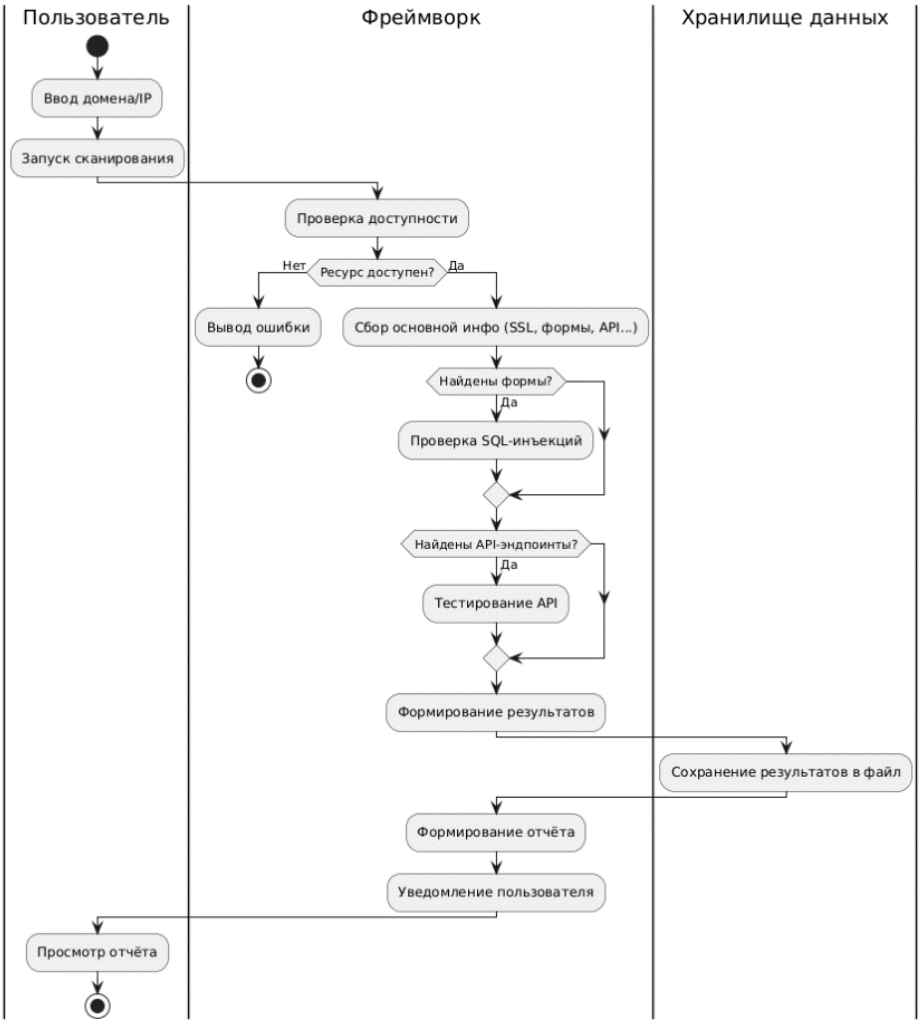


Рис. 1. BPMN-диаграмма

Фреймворк реализован на основе модульной архитектуры, предусматривающей последовательную передачу управления и данных между компонентами. Архитектурная структура

сочетает этапное выполнение с возможностью частично параллельной обработки. Каждый этап представлен в виде специализированного модуля. Взаимодействие между модулями осуществляется через формализованные структуры данных, преимущественно в формате *JSON*. Такая организация обеспечивает чёткое разделение ответственности, повышает масштабируемость системы и упрощает сопровождение.

На этапе конфигурации пользователь задаёт параметры сканирования через интерфейс командной строки (*CLI*). К числу параметров относятся: начальный *URL* веб-ресурса, максимальная глубина обхода, ограничение по количеству страниц, интервал между запросами, а также дополнительные настройки – использование прокси-серверов, настройка заголовков, *cookies*, включение или исключение отдельных модулей анализа. Указанные параметры формируют конфигурацию анализа и позволяют адаптировать поведение фреймворка к конкретным условиям эксплуатации и требованиям информационной безопасности.

После настройки запускается модуль сбора информации, реализующий стратегию обхода веб-приложения. Модуль выполняет анализ *HTML*-структуры страниц, идентификацию форм, ссылок, *API*-эндпоинтов и информации о домене. В рамках работы также осуществляется логирование и сбор метаданных, включая *SSL*-сертификаты, *IP*-адреса, а также фильтрация нежелательных форматов (например, *PDF*-документы, изображения, архивы). Все собранные данные структурируются в формате *JSON* и сохраняются для последующего анализа.

На следующем этапе выполняется запуск модулей анализа уязвимостей. Их выполнение организовано последовательно, при этом каждый модуль использует результаты предыдущего этапа.

Модуль обнаружения *SQL*-инъекций анализирует *URL*-параметры и формы на наличие уязвимостей путём подстановки различных типов *SQL-payloads*, включая классические выражения, *blind, time-based* и *union*-инъекции. Идентификация уязвимостей осуществляется на основе анализа сообщений об ошибках сервера, характерных исключений и временных задержек. Все подозрительные случаи фиксируются в журнале и классифицируются по типу и уровню критичности.

Модуль анализа *XSS*-уязвимостей предназначен для выявления как отражённых, так и сохраняемых *XSS*. Он осуществляет автоматическую подстановку *payloads* в поля форм, включая конструкции с *JavaScript*-событиями, *HTML*-сущностями и элементами, предназначенными для обхода фильтров. Верификация выполняется в несколько этапов, начиная с определения факта отражения *payload* в теле ответа и заканчивая анализом его расположения в различных контекстах, таких как *DOM*-элементы или встроенные скрипты. Обнаружение осуществляется с применением регулярных выражений и методов контекстного анализа *HTML*-кода.

Модуль анализа *API*-эндпоинтов использует как данные, полученные на этапе сканирования, так и собственные эвристики для выявления типовых путей (например, /*API*/, /*admin*/, /*graphql*). Модуль проверяет доступность эндпоинтов, осуществляет тестирование с различными *HTTP*-методами, а также анализирует уязвимости, включая обход аутентификации, утечку данных, *SQL*- и *XSS*-инъекции, ошибки обработки и некорректные *HTTP*-заголовки.

После завершения всех этапов анализа запускается модуль интерпретации результатов, использующий локальную языковую модель, доступную через интерфейс *Ollama*. Данный модуль обрабатывает сведения о каждой выявленной уязвимости, включая её тип, расположение и уровень критичности. На основе анализа формируется текстовое описание проблемы, рекомендации по устранению, а также возможные последствия. Это повышает читаемость и практическую ценность отчёта.

Заключительный этап работы фреймворка заключается в генерации итогового отчёта, содержащего систематизированную информацию по категориям уязвимостей. Отчёт может быть сформирован в виде документа формата *DOCX* с таблицами и заголовками либо в виде текстового файла в случае отсутствия соответствующих библиотек. Дополнительно сохраняются журналы работы всех модулей, что обеспечивает воспроизводимость анализа и возможность последующего аудита.

Разработанный фреймворк представляет собой многофункциональную интегрированную систему, объединяющую в себе широкий набор модулей, каждый из которых выполняет отдельную задачу в рамках процесса оценки защищённости *WEB*-приложений. В его состав входят компоненты для автоматизированного сбора и предварительной обработки данных, анализа различных типов

уязвимостей, интерпретации полученных результатов с учётом контекста исследуемой системы, а также формирования подробных аналитических отчетов и рекомендаций по устранению выявленных проблем. Такой комплексный подход обеспечивает значительное сокращение времени, затрачиваемого на проведение аудита безопасности, одновременно повышая точность, полноту и актуальность обнаружения уязвимостей. Кроме того, использование в рамках фреймворка методических рекомендаций и классификаций, разработанных международным проектом *OWASP*, позволяет привести процесс анализа в соответствие с современными международными стандартами и лучшими практиками в области обеспечения информационной безопасности *WEB*-приложений.

6. Реализация и унификация

Одной из ключевых задач при разработке фреймворка стало не только создание отдельных модулей, но и обеспечение их согласованной и предсказуемой работы. Для этого была внедрена унификация – как в логике работы, так и в форматах представления результатов.

Унифицированный формат логирования стал основой удобства эксплуатации и отладки. Каждый модуль – от сканера до анализатора уязвимостей – выводит события в консоль в одном стиле: с точной временной меткой, уровнем события (*INFO*, *WARNING* и др.), названием модуля и описанием происходящего. Например, в модуле сбора данных это позволяет буквально по секундам отследить, как проходил анализ сайта: какие страницы были обработаны, где возникли ошибки, сколько форм и *API*-эндпоинтов было обнаружено. Такой подход позволяет быстро локализовать сбои, легко ориентироваться в журнале и повышает читаемость логов даже при масштабных сканированиях (см. рис. 2).

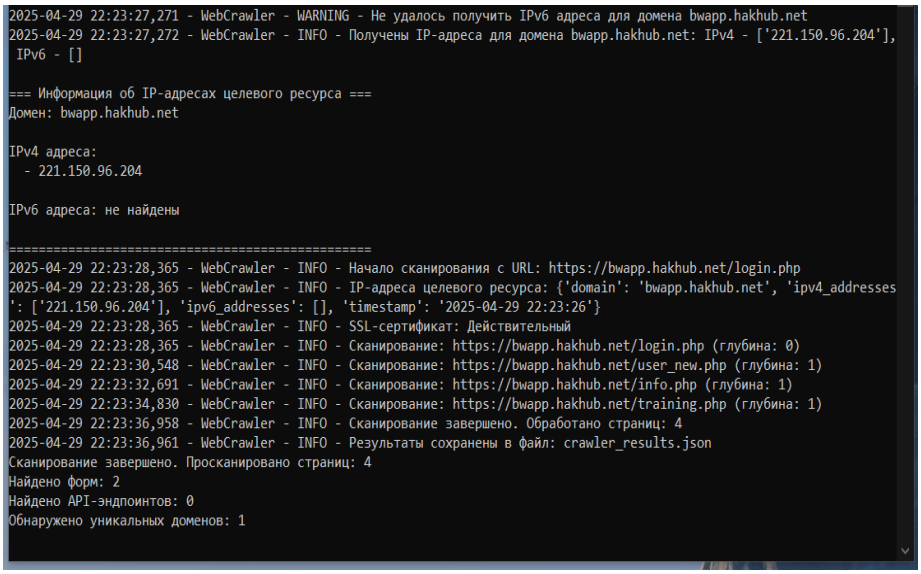


Рис. 2. Пример унифицированного логирования в модуле сбора данных

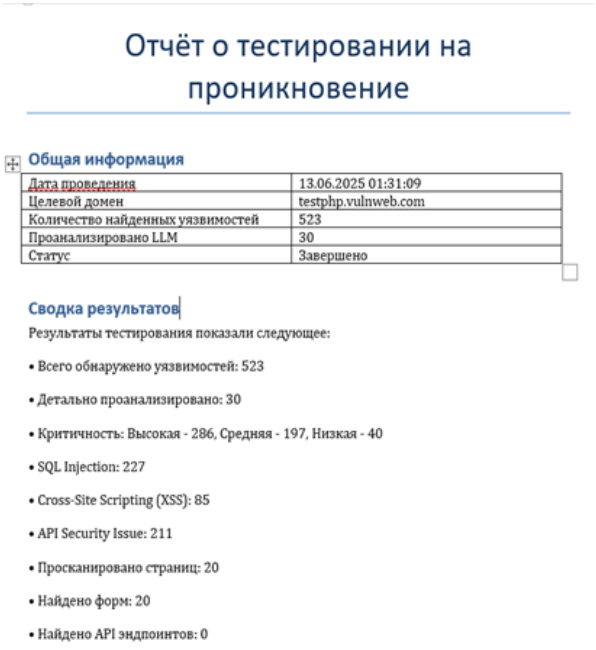
Кроме логов, фреймворк активно использует унифицированный *JSON*-формат для вывода всех результатов. Независимо от типа модуля – будь то сканирование, *XSS*-анализ или проверка *API* – структура выходного файла остаётся стабильной. В ней всегда присутствуют поля с целевым *URL*, статусом выполнения, временной меткой и блоком данных, характерных для конкретного анализа. Так, например, в модуле сбора данных это будет информация об *IP*-адресах, *SSL*-сертификате, страницах сайта и его структуре. Такой подход упрощает интеграцию с внешними сервисами, автоматическую генерацию отчётов и повторное использование данных другими модулями (см. рис. 3).

```
{
    "scan_info": {
        "start_url": "https://bwapp.hakhub.net/login.php",
        "base_domain": "bwapp.hakhub.net",
        "target_ips": {
            "domain": "bwapp.hakhub.net",
            "ipv4_addresses": [
                "221.150.96.204"
            ],
            "ipv6_addresses": [],
            "timestamp": "2025-04-29 22:23:26"
        },
        "is_https": true,
        "ssl_info": {
            "has_ssl": true,
            "cert_details": {
                "subject": {
                    "commonName": "hakhub.net"
                },
                "issuer": {
                    "countryName": "US",
                    "organizationName": "Let's Encrypt",
                    "commonName": "R10"
                },
                "version": 3,
                "serial_number": "055CEA1D8901C5D8C466B5BE229E43CF6911",
                "not_before": "Mar 16 15:53:18 2025 GMT",
                "not_after": "Jun 14 15:53:17 2025 GMT",
                "signature_algorithm": "Unknown",
                "extended_key_usage": [],
                "crl_distribution_points": [
```


Рис. 3. Единая структура выходного JSON-файла для модуля сбора данных

Не менее важным элементом реализации стала интеграция с локальной языковой моделью (*LLM*). После завершения анализа собранные *JSON*-данные автоматически передаются *LLM*, которая формирует текстовые отчёты. Эти отчёты содержат не только сухие данные, но и осмысленные описания уязвимостей, рекомендации по их устранению и краткие резюме для ИБ-специалистов. Благодаря локальному размещению модели обеспечивается полная конфиденциальность и изоляция от внешних сервисов – это особенно важно в корпоративной среде и при работе с критичными системами.

В финальной версии реализованы два формата отчётности: простой (табличный список с краткими сводками по каждой уязвимости) и расширенный – с включением пояснительных блоков, сгенерированных языковой моделью (см. рис. 4 и 5). Это повышает читаемость отчётов и позволяет использовать результаты сразу – например, для внутренних отчётов или подготовки рекомендаций для разработчиков.



Отчёт о тестировании на проникновение

Общая информация

Дата проведения	13.06.2025 01:31:09
Целевой домен	testphp.vulnweb.com
Количество найденных уязвимостей	523
Проанализировано LLM	30
Статус	Завершено

Сводка результатов

Результаты тестирования показали следующее:

- Всего обнаружено уязвимостей: 523
- Детально проанализировано: 30
- Критичность: Высокая - 286, Средняя - 197, Низкая - 40
- SQL Injection: 227
- Cross-Site Scripting (XSS): 85
- API Security Issue: 211
- Просканировано страниц: 20
- Найдено форм: 20
- Найдено API эндпоинтов: 0

Рис. 4. Пример базового отчёта фреймворка без текстовой интерпретации

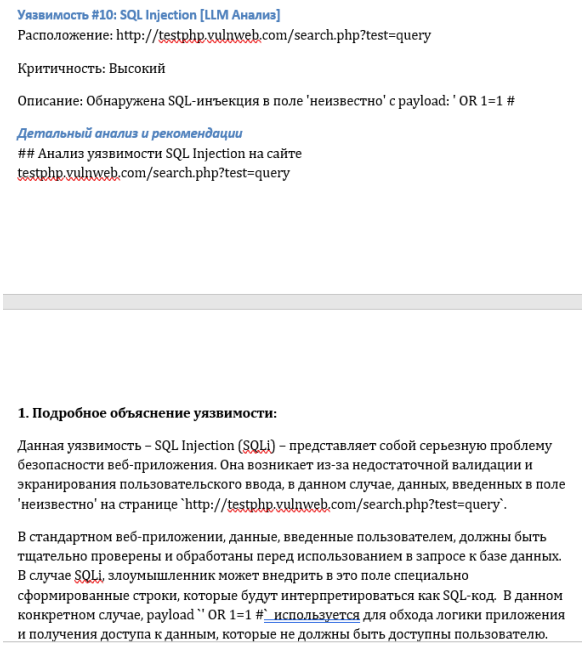


Рис. 5. Расширенный отчёт с интерпретацией уязвимостей на основе LLM

Заключение

В ходе работы был разработан модульный фреймворк для автоматизированного анализа защищённости веб-приложений. Исследование охватило весь цикл проектирования: от анализа актуальных угроз и методологии *OWASP* до создания архитектуры, реализации компонентов и проверки эффективности решения.

Проведён анализ существующих инструментов тестирования и обоснована необходимость создания гибкой и расширяемой системы, адаптированной под современные задачи аудита безопасности. Разработанный фреймворк включает модули сбора данных, поиска уязвимостей (*SQL*-инъекции, *XSS*, ошибки в *API*), генерации отчётов и интеграции с языковой моделью для интерпретации результатов.

Архитектура решения обеспечивает масштабируемость, возможность параллельной работы модулей и удобство использования благодаря унифицированному формату обмена данными (*JSON*) и командному интерфейсу. Полученные результаты подтверждают достижение поставленных целей и демонстрируют потенциал фреймворка в учебных, исследовательских и практических сценариях.

Список источников

1. OWASP Foundation, the Open Source Foundation for Application : [Open Web Application Security Project]. – OWASP Foundation, Inc., 2025. – URL: https://OWASP.org (дата обращения: 11.03.2025).

2. OWASP Top Ten | OWASP Foundation // OWASP Foundation, the Open Source Foundation for Application. – OWASP Foundation, Inc., 2025. – URL: https://owasp.org/www-project-top-ten/ (дата обращения: 11.03.2025).

3. Hidayanto B. C., Akbar I. A., Putra A. Z. D. Automated Web Security Testing Guide Mapping to Accelerate Process on Penetration Testing // Procedia Computer Science. – 2024. – Т. 234. – С. 1412-1419.

4. Loncar K., Redzepagic J., Dakic V. SECURE CODING GUIDELINES AND STANDARDS // Annals of DAAAM & Proceedings. – 2024. – Т. 35. – DOI: 10.2507/35th.daaam.proceedings.xxx.

5. WSTG - Stable | OWASP Foundation : [стабильная версия v4.2] // OWASP Foundation, the Open Source Foundation for Application. – OWASP Foundation, Inc., 2025. – URL: https://owasp.org/www-project-web-security-testing-guide/stable/ (дата обращения: 25.04.2025).

6. Мартин Р. Чистая архитектура. Искусство разработки программного обеспечения. – Санкт-Петербург: Питер, 2022. – 352 с.

7. Добрышин М. М., Шугуров Д. Е., Беляев Д. Л. Модель сетевых атак типа XSS-И SQL-инъекций на веб-ресурсы, учитывающая различные уровни сложности их реализации // Известия Тульского государственного университета. Технические науки. – 2021. – №. 2. – С. 196-204.

8. Хоффман Э. Безопасность веб-приложений. – Санкт-Петербург: Питер, 2021. – 336 с.

9. Using ollama / F.S. Marcondes, A. Gala, R. Magalhães [et al.] // Natural Language Analytics with Generative Large-Language Models: A Practical Approach with Ollama and Open-Source LLMs. – Cham: Springer Nature Switzerland, 2025. – С. 23-35.