УДК 004.4'23

РАЗРАБОТКА ИНТЕГРИРОВАННОГО ПРОГРАММНОГО СРЕДСТВА ДЛЯ АВТОМАТИЗАЦИИ ПРОЦЕССОВ РУЧНОГО ТЕСТИРОВАНИЯ ВЕБ-ПРИЛОЖЕНИЙ

Войнилович Н.Ю., Галяк И.П., Ефименко Д.Д.

Белорусский государственный университет информатики и радиоэлектроники, г. Минск, Республика Беларусь

Научный руководитель: Коваленко И.В. – ст. преподаватель кафедры ПИКС

Аннотация. В работе рассматривается проблема низкой эффективности ручного тестирования веб-приложений, связанная с выполнением повторяющихся операций. Предлагается решение в виде комплексного программного средства (ПС) – браузерного расширения, которое интегрирует инструменты для генерации тестовых данных, аудита безопасности, перехвата сетевого трафика и автоматизированного формирования отчетов об ошибках с использованием большой языковой модели (LLM).

Ключевые слова: обеспечение качества (QA), автоматизация тестирования, безопасность веб-приложений, LLM, браузерное расширение, генерация отчетов, фаззинг, XSS, SQL-инъекция.

Введение. Современный цикл разработки программного обеспечения, основанный на методологиях Agile и DevOps, характеризуется короткими итерациями и частыми релизами. Это предъявляет повышенные требования к скорости и качеству работы отделов обеспечения качества [1]. Значительная часть времени специалистов по ручному тестированию расходуется не на поиск бизнес-логических ошибок, а на рутинные, повторяющиеся задачи. К ним относятся заполнение регистрационных форм, сбор сетевых логов, создание и оформление баг-репортов.

Актуальность данной работы обусловлена необходимостью сокращения временных затрат на вспомогательные операции в процессе ручного и исследовательского тестирования. Существующие инструменты, как правило, узкоспециализированы, что заставляет тестировщика постоянно переключать контекст, снижая общую производительность.

Целью работы является разработка программного средства, выполненного в виде браузерного расширения, для повышения эффективности ручного тестирования за счет объединения наиболее востребованных инструментов в едином пользовательском интерфейсе.

Основная часть Программное средство (далее ПС) направлено на решение ряда проблем, с которыми сталкиваются QA-инженеры.

Первой из них является трудоемкость подготовки тестового окружения. Для проверки функциональности форм тестировщику необходимо многократно вводить различные данные, что отнимает значительную часть времени. ПС решает эту проблему путем автозаполнения, анализируя DOM-атрибуты полей и сопоставляя их с внутренними словарями для заполнения форм за одну операцию.

Вторая проблема заключается в сложности сбора артефактов для отчета об ошибке. Качественный отчет об ошибке требует от инженера выполнения нескольких несвязанных действий для сбора скриншотов, URL и сетевых логов [2]. Предлагаемое ПС автоматизирует этот процесс, осуществляя фоновый перехват сетевого трафика и объединяя все необходимые артефакты в единый, готовый к отправке HTML-файл.

Третья проблема – это высокий порог входа в тестирование безопасности для функциональных тестировщиков. Проверка уязвимостей, таких как XSS или SQL-инъекции, требует специфических знаний. ПС снижает этот порог за счет интеграции с LLM [3]. Когда тестировщик описывает задачу, ПС передает нейросети не только запрос, но и полный

контекст поля ввода, что позволяет LLM сгенерировать узконаправленный и потенциально более эффективный вектор атаки.

Программное средство реализовано в виде браузерного расширения для браузеров на базе Chromium, реализующих архитектуру Manifest V3 [4]. Архитектура ПС является модульной и включает в себя несколько компонентов. Первым компонентом является Content Script, который отвечает за рендеринг пользовательского интерфейса, обработку действий пользователя и считывание атрибутов DOM-элементов. Второй компонент, Injected Script, внедряется в глобальный контекст веб-страницы для переопределения методов XMLHttpRequest и window.fetch, что позволяет перехватывать сетевые запросы и передавать их копии в Content Script. Третьим элементом является Background Service Worker, выполняющий логические операции. Особая роль данному компоненту отведена в подсистеме интеллектуальной генерации атак. Ввиду действия политики Cross-Origin Resource Sharing (CORS), блокирующей прямые запросы к сторонним API со страниц сайтов, взаимодействие с сервисом Groq реализовано через Service Worker. Он принимает от Content Script контекст целевого поля ввода, выполняет авторизованный запрос к модели llama-3.3-70b-versatile и возвращает сгенерированный вектор атаки обратно в интерфейс расширения. Взаимодействие компонентов при выполнении данного сценария представлено на диаграмме последовательности (рисунок 1).

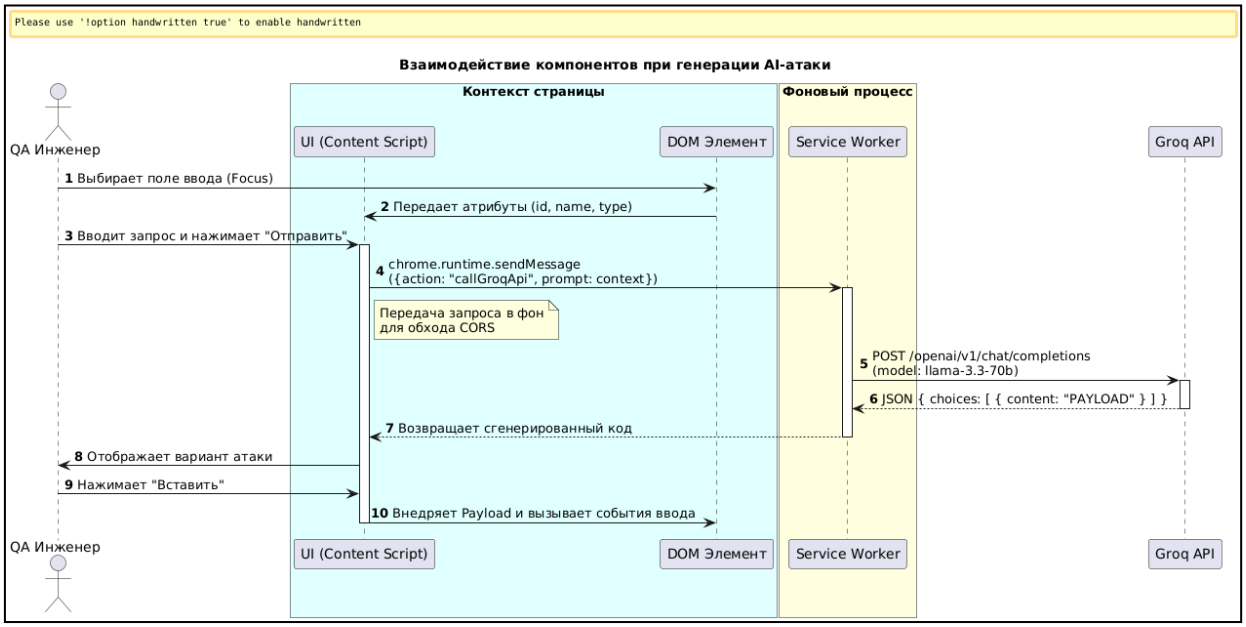

Рисунок 1 – Взаимодействие элементов при генерации атаки

Внедрение разработанного программного средства в технологический процесс обеспечения качества позволяет изменить подход к проведению исследовательского и регрессионного тестирования. Основной эффект достигается за счет минимизации времени, затрачиваемого на проведение тестов. В традиционном сценарии тестировщик вынужден оперировать набором разрозненных инструментов, переходя из браузера в текстовый редактор, консоль разработчика и графический редактор. Интеграция всех необходимых функций в единый интерфейс позволяет проводить проверки без прерывания пользовательской сессии, что, согласно предварительным оценкам, сокращает время обработки одного тестового сценария на 30-40%.

Особую практическую ценность представляет модуль интеллектуальной поддержки на базе LLM, который трансформирует расширение из простого инструмента автоматизации в систему поддержки принятия решений. Генерация векторов атак нейросетью, осведомленной о контексте конкретного DOM-элемента, повышает вероятность выявления уязвимостей типа

XSS и SQL Injection на ранних этапах жизненного цикла ПО, снижая стоимость их последующего исправления [5].

Использование модуля генерации отчетов обеспечивает стандартизацию выходных артефактов тестирования. Автоматическое прикрепление снимка экрана и таблицы сетевых логов к отчету исключает человеческий фактор, при котором разработчики часто получают неполные данные о дефекте. Это упрощает коммуникацию внутри команды разработки и ускоряет локализацию ошибок на стороне сервера, так как отчет уже содержит точные данные о статусах HTTP-ответов и теле запросов.

Заключение. В ходе выполнения работы спроектировано и реализовано программное средство в формате расширения для браузеров на базе Chromium. Созданный инструмент успешно решает задачи автоматизации рутинных операций ввода данных, перехвата сетевого трафика и документирования дефектов, подтверждая гипотезу о целесообразности объединения инструментов в единый программный комплекс.

Результаты разработки могут быть рекомендованы к внедрению в процессы ручного функционального тестирования веб-приложений для повышения производительности труда QA-инженеров. Дальнейшее развитие проекта предполагает интеграции с внешними системами управления проектами, в частности, реализацию прямой отправки сгенерированных отчетов в системы управления проектами посредством API, а также расширение базы сценариев для автоматического фаззинга без участия нейросети.

Список литературы

1. Hoffman, A. Web Application Security: Exploitation and Countermeasures for Modern Web Applications / Andrew Hoffman. – Sebastopol: O'Reilly Media, 2020. – 340 p. – ISBN 978-1-4920-5311-8.

2. Wang, J. Software Testing with Large Language Models: Survey, Landscape, and Vision / Junjie Wang, Yuchao Huang, Chunyang Chen // IEEE Transactions on Software Engineering. – 2024. – Vol. 50, N 4. – Pp. 911–936.

3. Иванов, А. И. Интеграция больших языковых моделей в процессы автоматизированного тестирования защищенности веб-приложений / А. И. Иванов, В. С. Петров, Д. А. Смирнов // Вопросы кибербезопасности. – 2023. – № 5 (57). – С. 45–58.

4. Фрисби, М. Building Browser Extensions: Create Modern Extensions for Chrome, Safari, Firefox, and Edge / Matt Frisbie. – Berkeley: Apress, 2022. – 332 p. – ISBN 978-1-4842-8724-8.

5. Large Language Models for Cyber Security: A Systematic Literature Review / Y. Fang [и др.] // ACM Computing Surveys. – 2024. – Vol. 56, N 12. – Art. 302. – Pp. 1–34.

UDC 004.4'23

DEVELOPMENT OF THE INTEGRATED SOFTWARE TOOL FOR AUTOMATION OF MANUAL TESTING PROCESSES OF WEB APPLICATIONS

Voynilovich N.Yu., Galyak I.P., Efimenko D.D.

Belarusian State University of Informatics and Radioelectronics, Minsk, Republic of Belarus

Kovalenko I.V. – Senior Lecturer at the Department of Marketing

Annotation. This paper addresses the challenge of low efficiency in manual web application testing caused by the execution of repetitive operations. We propose a comprehensive software solution implemented as a browser extension that integrates tools for test data generation, security auditing, network traffic interception, and automated bug report generation leveraging Large Language Models (LLMs).

Keywords: quality assurance (QA), test automation, web application security, LLM, browser extension, report generation, fuzzing, XSS, SQL injection.