УДК 004

А. О. Ревазова[1]✉

Методы выявления и оценки уязвимостей в системах статического анализа безопасности программного обеспечения

[1]Национальный исследовательский ядерный университет «МИФИ»,
г. Москва, Российская Федерация
e-mail: aanrevazova@gmail.com

Аннотация. В работе исследуются методы выявления и оценки уязвимостей в системах статического анализа безопасности программного обеспечения. Существующие методы статического анализа сталкиваются с рядом ограничений, включая высокое количество ложных срабатываний, недостаточную точность и сложность анализа современных языков программирования и архитектур. Рассматриваются текущие подходы, их недостатки и необходимость разработки новых методов, способных более эффективно справляться с выявлением угроз. В частности, акцентируется внимание на важности интеграции машинного обучения и искусственного интеллекта в процессы статического анализа для повышения точности и снижения числа ложных срабатываний. В работе проведено исследование и сравнение существующих решений в области статического анализа безопасности. Также предложен и протестирован метод машинного обучения, направленный на снижение количества ложноположительных срабатываний.

Ключевые слова: статический анализ, ложноположительные срабатывания, машинное обучение

A. O. Revazova[1]✉

Methods for identifying and evaluating vulnerabilities in static software security analysis systems

[1]National Research Nuclear University MEPhI (Moscow Engineering Physics Institute),
Moscow, Russian Federation
e-mail: aanrevazova@gmail.com

Annotation. The paper examines methods for identifying and evaluating vulnerabilities in static software security analysis systems. Existing static analysis methods face a number of limitations, including a high number of false positives, insufficient accuracy, and the complexity of analyzing modern programming languages and architectures. The current approaches, their disadvantages, and the need to develop new methods that can more effectively deal with threat detection are considered. In particular, attention is focused on the importance of integrating machine learning and artificial intelligence into static analysis processes to increase accuracy and reduce the number of false positives. The paper examines and compares existing solutions in the field of static security analysis. A machine learning method aimed at reducing the number of false positives has also been proposed and tested.

Keywords: static analysis, false positives, machine learning

Введение

При обеспечении безопасности программного обеспечения можно выделить два фундаментальных подхода: предотвращение уязвимостей на раннем этапе

при помощи анализа исходного кода инструментами статического анализа и выявление уязвимостей на этапе сопровождения. В данной работе рассматривается первый способ.

Актуальность этой задачи возрастает не только в контексте внешних библиотек и компонентов, но и в отношении самого программного обеспечения и его архитектуры. В настоящее время снижение количества ложноположительных срабатываний является одной из ключевых задач для разработчиков и организаций, занимающихся обеспечением качества программного обеспечения. Особенно важным аспектом является использование статического анализа кода, который позволяет выявлять потенциальные уязвимости и ошибки, но часто сталкивается с проблемой высокой степени ложных срабатываний.

С развитием технологий и появлением новых методов разработки программного обеспечения традиционные подходы статического анализа становятся недостаточно эффективными и могут привести к игнорированию реальных угроз [1]. В связи с этим возникает необходимость в разработке и исследовании новых методов интеллектуально-адаптивного снижения ложноположительных срабатываний в статическом анализе кода [2, 3].

Проектирование стенда

Для проверки системы статического анализа кода, основанной на искусственном интеллекте и нацеленной на уменьшение числа ложноположительных срабатываний, целесообразно применять тестовый стенд.

Необходим сервер для обучения и тестирования моделей, которые будут применяться в системе статического анализа кода. Он должен обладать достаточной вычислительной мощностью для обработки больших объемов кода и метрик, необходимых для обучения. К системе тестирования подключается база данных, содержащая информацию о коде, метриках качества, уязвимостях и правилах анализа. База данных будет использоваться как для обучения моделей, так и для хранения результатов их работы.

Также участвуют устройства, на которых будет выполняться статический анализ кода. Это могут быть рабочие станции разработчиков или серверы, на которых размещен анализируемый код.

Для сравнительного анализа необходимо обеспечить наличие средств статического анализа, которые помогут выявить потенциальные уязвимости и ошибки в коде. Для оценки эффективности необходимы инструменты мониторинга и анализа, которые помогут определить производительность системы статического анализа, выявить ошибки и принять решения по их устранению.

На рис. 1 представлена схема экспериментального стенда для проведения тестирования.

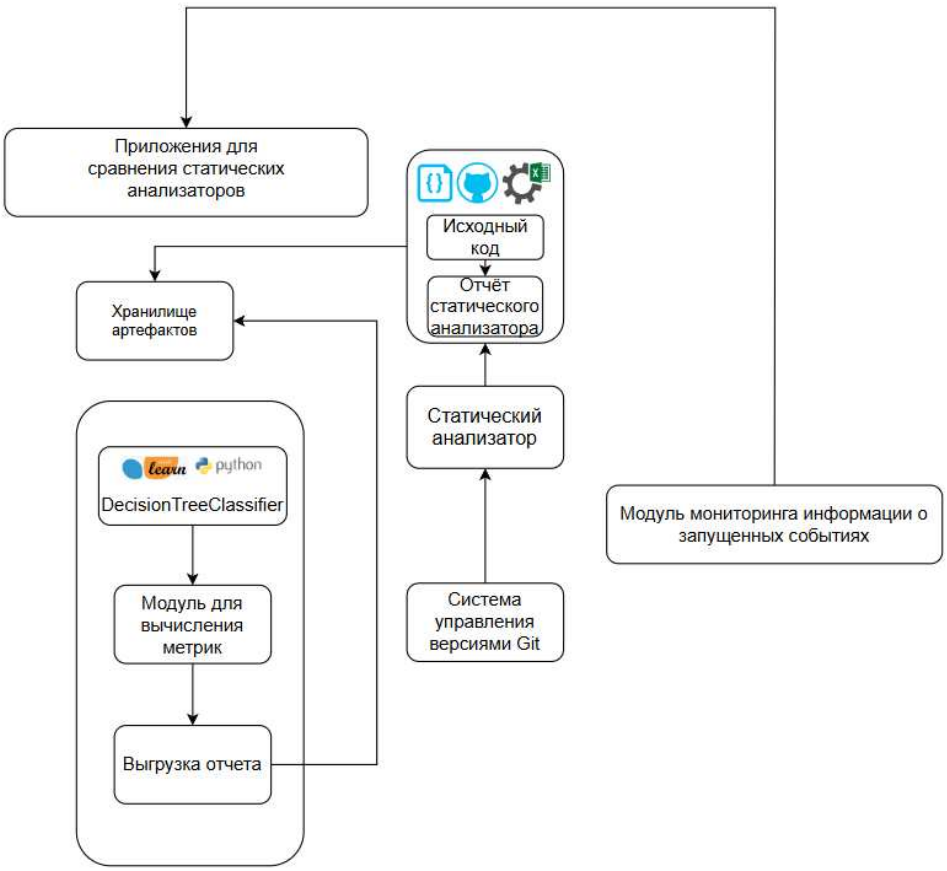

Рис. 1. Схема экспериментального стенда для проведения тестирования

Необходимо разработать разнообразные тестовые сценарии, которые будут использоваться для проверки точности и устойчивости системы статического анализа. Эти сценарии могут включать в себя различные примеры кода с известными уязвимостями, а также код, который не содержит ошибок, чтобы оценить уровень ложноположительных срабатываний.

Тестирование

В процессе проверки анализаторов исходного кода были выявлены различные проблемы, которые объединяют все анализаторы. Среди возможных направлений для исследования можно выделить разработку нового метода для обнаружения ложноположительных срабатываний.

Предлагается использовать методы на базе машинного обучения для выявления ложноположительных срабатываний с целью улучшения качества анализа исходного кода и повышения безопасности системы в целом [4–6].

На этапе обучения модели возникла проблема переобучения, связанная с использованием модели решающих деревьев, которая хорошо подходит для клас-

сификации признаков на две группы. Для устранения этой проблемы было принято решение ограничить глубину дерева, что позволило значительно снизить уровень переобучения, однако возникла новая проблема: метрики машинного обучения стали сильно варьироваться на различных тестовых наборах данных. План выбора модели машинного обучения и проведения тестирования представлен на рис. 2.

Рис. 2. План выбора модели машинного обучения

Результаты

Были получены усредненные метрики, однако следует отметить высокую зависимость от контекста кода.

Полученные метрики имеют следующие значения:

– accuracy (точность) = 26 %,
– recall (полнота) = 19 %,
– precision (точность) = 63 %.

Заключение

Для того, чтобы уверенно говорить о достижении целей работы, необходимо добиться значений accuracy и recall не менее 40 %, а precision не менее 80 %. В дальнейшем для повышения указанных метрик планируется рассмотреть подходы и методы, изложенные в [7–11].

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Ali E. A., Ammar F. M. SAST Tools and Manual Testing to Improve the Methodology of Vulnerability Detection in Web Applications // Special Issue for IJEIT on Engineering and Information Technology. 2024. № 1 (12). С. 79.

2. Hüther L., Sohr K., Berger B., Rothe H. Machine Learning for SAST: A Lightweight and Adaptable Approach // Springer Science and Business Media Deutschland GmbH, 2024. С. 85–104.

3. Machine Learning in the Context of Static Application Security Testing - ML-SAST // Federal Office for Information Security. 2021.

4. Chandra Panda K., Agrawal S. Application of AI and ML in the Field of DevSecOps // Journal of Artificial Intelligence & Cloud Computing 2022. № 4. С. 1–4.

5. Fu M., Pasuksmit J., Tantithamthavorn C. AI for DevSecOps: A Landscape and Future Opportunities // ACM Transactions on Software Engineering and Methodology. 2024. №34 (4).

6. Saurabh S. K., Kumar D. Model to Reduce DevOps Pipeline Execution Time Using SAST // DOI:10.21203/rs.3.rs-2919183/v1. 2023.

7. Jian S., Ying S. Preference Selection Index Method for Machine Selection in a Flexible Manufacturing Cell // EDP Sciences, 2017.

8. Wang P., Zhang C., Qi F., Huang Z. A Single-Shot Arbitrarily-Shaped Text Detector Based on Context Attended Multi-Task Learning // 27th ACM International Conference, Inc, 2019. С. 1277–1285.

9. Wang Y., Zhao N., Jing H., Meng B. A Novel Model of the Ideal Point Method Coupled with Objective and Subjective Weighting Method for Evaluation of Surrounding Rock Stability // Mathematical Problems in Engineering, 2016.

10. Zhu J., Li K., Chen S., Fan L. A Comprehensive Study on Static Application Security Testing (SAST) Tools for Android // Journal of LaTex class files, vol. 14, no. 8, August 2024.

11. Chen J., Xiang H., Li L., Zhang Y. Utilizing Precise and Complete Code Context to Guide LLM in Automatic False Positive Mitigation // DOI:10.48550/arXiv.2411.03079 2024.