УДК 004.056.5

СТАТИЧЕСКИЙ АНАЛИЗ КОДА КАК ИНСТРУМЕНТ ПОВЫШЕНИЯ БЕЗОПАСНОСТИ ПРИЛОЖЕНИЙ

Зарипова Ч. И., Куценко С. М.

Казанский государственный энергетический университет

Аннотация: Статья посвящена значению статического анализа кода (SAST) для безопасности программного обеспечения. Рассматриваются решения Solar appScreener и SASTAV, использующие каскадную AI-валидацию и композиционный анализ. Отмечается, что внедрение SAST в SDLC повышает защищенность, снижает расходы и поддерживает цифровой суверенитет.

Ключевые слова: информационная безопасность, статический анализ кода, уязвимости, DevSecOps, SAST, AI-валидация, Solar appScreener, SASTAV.

STATIC CODE ANALYSIS AS A TOOL FOR ENHANCING APPLICATION SECURITY

Ch. I. Zaripova[1], S. M. Kutsenko[1]

[1]Kazan State Power Engineering University

Abstract: The article examines the importance of static code analysis (SAST) for software security. It discusses the Solar appScreener and SASTAV solutions, which employ cascading AI validation and compositional analysis. It is noted that integrating SAST into the SDLC enhances protection, reduces costs, and supports digital sovereignty.

Keywords: information security, static code analysis, vulnerabilities, DevSecOps, SAST, AI validation, Solar appScreener, SASTAV.

Современный этап цифровой трансформации общества характеризуется возрастанием зависимости человека и государства от программных систем. Программный код становится основой функционирования критически важных инфраструктур – от энергетики до финансов. Соответственно, каждая ошибка или недочет в коде может привести не только к сбоям, но и к масштабным инцидентам информационной безопасности [1,3].

Традиционные методы защиты (межсетевые экраны, антивирусные средства, системы шифрования) е способны противостоять уязвимостям, заложенным в самой логике

приложения. Для обеспечения комплексной защиты необходимо смещать акцент на ранние стадии разработки, когда обнаружение дефектов наиболее эффективно [2,4]. На этом фоне статический анализ кода становится ключевым инструментом предотвращения киберугроз.

Статический анализ кода представляет собой метод исследования программного обеспечения без его выполнения, основанный на изучении исходных текстов, бинарных модулей и зависимостей. Цель метода -– выявление уязвимостей, ошибок проектирования, нарушений стандартов безопасности и некорректных логических связей [5].

Анализ проводится в несколько этапов:

1. преобразование исходного кода во внутреннее представление;

2. выполнение лексического, синтаксического и семантического анализа;

3. сопоставление структур с базой известных уязвимостей;

4. формирование отчета с описанием найденных дефектов и рекомендациями.

Преимуществом статического анализа является возможность применения на ранних этапах жизненного цикла разработки, когда исправление ошибок обходится значительно дешевле, чем после релиза продукта [2].

Современные инструменты SAST эволюционируют в сторону автоматизации и интеллектуального анализа. Особое значение приобретают решения, использующие искусственный интеллект для подтверждения и приоритизации уязвимостей.

Одним из таких решений является отечественная система SASTAV, реализующая методологию каскадной AI-валидации. Данная технология позволяет автоматизировать процесс триажа, снижая количество ложноположительных срабатываний и повышая точность выявления реальных угроз. Система поддерживает более двадцати языков программирования, осуществляет композиционный анализ (SCA), формирует SBOM, выявляет транзитивные зависимости и уязвимые библиотеки, включая протестные (protestware) [4].

Интерфейс решения построен по принципам DevSecOps и включает:

- единый каталог проектов с оценкой уровня риска;
- визуальные графы уязвимостей и зависимостей;
- редактор пользовательских правил анализа;

· интеграцию с системами CI/CD и баг-трекерами.

Сходные функции реализованы и в платформе Solar appScreener, которая обеспечивает практически полное покрытие кода, поддержку 36 языков программирования, а также корреляцию результатов различных видов тестирования – SAST, DAST и OSA.

Для полноценной оценки безопасности приложений статический анализ дополняется динамическим анализом (DAST) и анализом сторонних компонентов (SCA). DAST (Dynamic Application Security Testing) исследует поведение программы в реальной среде, выявляя уязвимости, проявляющиеся только при выполнении. SCA (Software Composition Analysis) позволяет контролировать безопасность используемых библиотек и открытых компонентов, оценивать лицензионные риски и угрозы supply-chain-атак.

Комбинация SAST, DAST и SCA обеспечивает всестороннее покрытие безопасности и минимизирует вероятность пропуска критических дефектов.

Интеграция статического анализа в процесс разработки повышает уровень защищенности приложений, снижает финансовые и репутационные риски и обеспечивает соответствие требованиям государственных регуляторов в области кибербезопасности.

Особое значение внедрение отечественных SAST-решений имеет в контексте политики цифрового суверенитета и импортозамещения. Использование решений, разработанных внутри национальной юрисдикции, позволяет гарантировать контроль над данными, алгоритмами и инфраструктурой [6].

Перспективные направления развития технологий SAST включают:

· применение методов машинного обучения для повышения точности анализа;

· автоматическую корреляцию результатов между SAST, DAST и SCA;

· развитие визуальных аналитических инструментов для работы с уязвимостями;

· формирование единого пространства DevSecOps с автоматическим управлением рисками.

Статический анализ кода становится неотъемлемой частью стратегии обеспечения кибербезопасности. Он позволяет перейти от реактивной модели защиты к проактивной, где уязвимости устраняются до того, как становятся угрозами.

Решения нового поколения, такие как SASTAV и Solar appScreener, демонстрируют возможности отечественной школы ИБ-разработок и формируют основу для устойчивой цифровой независимости.

Безопасный код – это не только технический стандарт, но и элемент национальной стабильности в эпоху тотальной цифровизации.

Литература

1. Иванов А.С., Петров К.Д. Статический анализ кода мобильных приложений как средство выявления его уязвимостей [Электронный ресурс] // КиберЛенинка. – URL: https://cyberleninka.ru/article/n/staticheskiy-analiz-koda-mobilnyh-prilozheniy-kak-sredstvo-vyyavleniya-ego-uyazvimostey

2. Каширина Е.И., Власов А.М. Внедрение SonarQube: автоматизированный контроль безопасности кода в CИCD // Информационные технологии в строительных, социальных и экономических системах. 2025. DOI 10.26297/2312 9409.2025.4.8

3. Куценко, С. М. Кибербезопасность в цифровой экономике / С. М. Куценко // Экономика и предпринимательство. – 2025. – № 1(174). – С. 130-132. – DOI 10.34925/EIP.2024.174.1.021.

4. Куценко, С. М. Современные вызовы кибербезопасности и системы обнаружения вторжений / С. М. Куценко, Е. А. Салтанаева // Экономика и предпринимательство. – 2025. – № 6(179). – С. 950-954. – DOI 10.34925/EIP.2025.179.6.170.

5. Статический анализатор кода SAST для поиска уязвимостей [Электронный ресурс] // RT-Solar. – URL: https://rt-solar.ru/products/solar_appscreener/blog/2361/

6. Официальный сайт системы SASTAV [Электронный ресурс]. – URL: https://sastav.ru/