

SASTAV

Российское решение
для статического анализа исходного кода
с применением методологии каскадной AI-валидации
найденных дефектов

В ЧЕМ, СОБСТВЕННО, ПРОБЛЕМА?

Основная задача, стоящая перед разработчиками, - создание функционирующего кода в сжатые сроки, ведь время на стороне конкурента! Подобный вызов вынуждает даже высококвалифицированных и компетентных специалистов пренебрегать аспектами безопасности кода или, как минимум, отодвигать их на второй план.

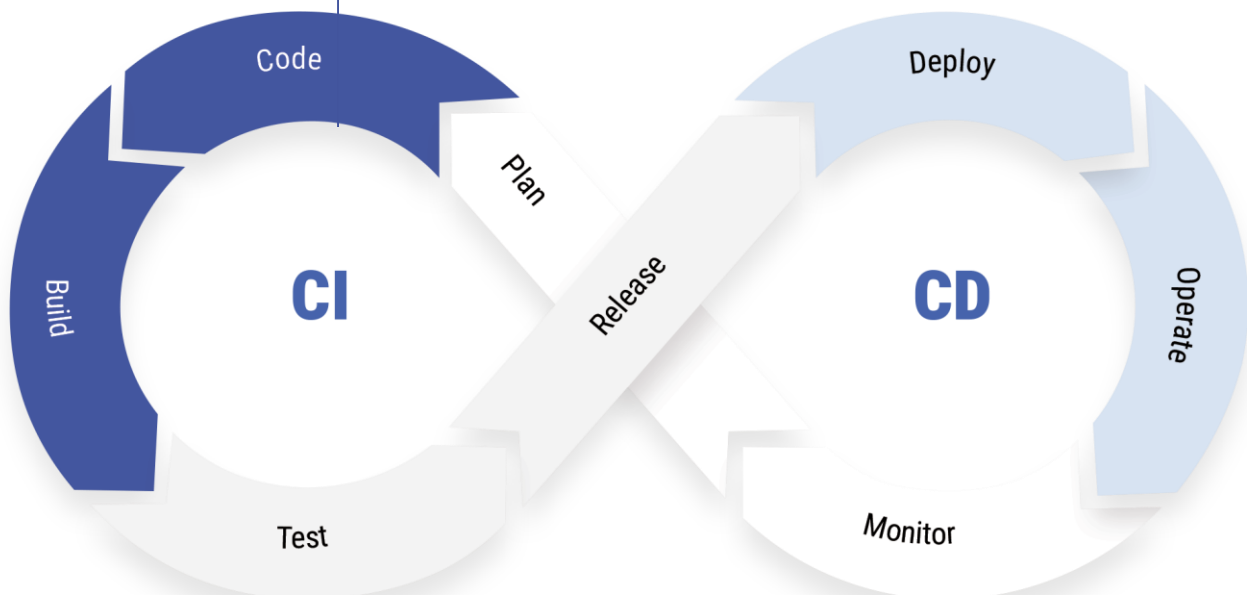
НЕСВОЕВРЕМЕННОЕ ОБЕСПЕЧЕНИЕ БЕЗОПАСНОСТИ КОДА МОЖЕТ СТАТЬ ПРИЧИНОЙ ФИНАНСОВЫХ И РЕПУТАЦИОННЫХ ПОТЕРЬ.

Выявление уязвимостей в коде на поздних этапах цикла разработки имеет целый ряд недостатков, которые, в конечном итоге, сказываются на сроках вывода программного продукта на рынок. Например, код-ревью, проводимый для уже скомпилированного кода, слишком сильно подвержен влиянию человеческого фактора

а его скорость и качество ограничены человеческими возможностями. Тесты же на проникновение, которые проводятся для уже готового приложения, выдают лишь частичные результаты, причем с большой задержкой с точки зрения цикла разработки.

РЕШЕНИЕ ЕСТЬ!

SAST 



SASTAV

УСТАНАВЛИВАЕТСЯ ЗА 20 МИНУТ И ТРЕБУЕТ МИНИМАЛЬНОГО ОБСЛУЖИВАНИЯ

Встраивается в SDLC на самом раннем этапе, не требуя ни сборки, ни компиляции, ни полноты логики сканируемого кода

- Позволяет ускорить вывод на рынок цифровых продуктов без привлечения высококвалифицированных специалистов
- Применяется на самых ранних этапах разработки и легко встраивается в цикл для оптимизации и автоматизации процесса SDLC
- Достигает рекордных скоростей сканирования, не компрометируя качества результатов
- Указывает на оптимальную точку для устранения дефекта, чем эффективно снижает трудозатраты разработчиков
- Группирует репозитории в проекты для улучшения навигации и отмечает в каждом репозитории уже отсканированные ветки
- Легко масштабируется путём подключения дополнительных ядер сканирования для распараллеливания процесса при больших нагрузках
- Помогает визуально ориентироваться в коде и указывает на уязвимую строку кода путём графического представления векторов атак
- Тиражирует результаты триажа, эффективно предотвращая повторение однажды вычисленных векторов уязвимости в последующих сканированиях
- Предоставляет описание уязвимости и рекомендации по ее устранению для каждой сработки

ПОДДЕРЖИВАЕМЫЕ ЯЗЫКИ



КАК ЭТО РАБОТАЕТ

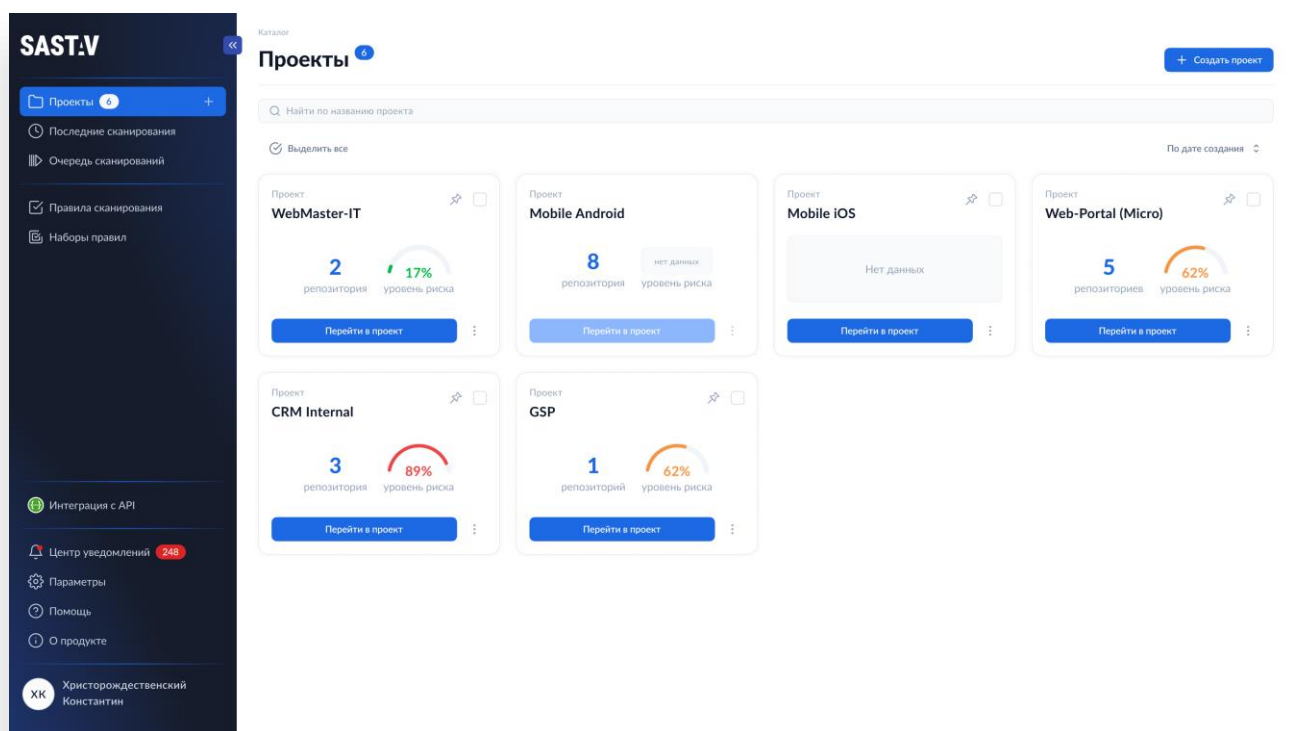


ТЕХНИЧЕСКИЕ ПАРАМЕТРЫ

Версия Java: 17
Версия Spring Boot: 3.2.1

Вариант поставки: docker-образ
Среда развертывания: Linux, K8s

ЕДИНЫЙ КАТАЛОГ ПРОЕКТОВ С ОЦЕНКОЙ РИСКОВ



На главном экране отображается сводный каталог всех проектов с индикаторами уровня риска для каждого. Непрерывный Health Check дает четкое понимание текущей ситуации. Мониторинг в реальном времени позволяет предпринимать своевременные действия при изменениях.

ИНТЕРФЕЙС «ПРОЕКТ»

The screenshot displays the SAST.V web interface. On the left is a dark sidebar with navigation options: 'Проекты' (Projects), 'Последние сканирования' (Last scans), 'Очередь сканирований' (Scan queue), 'Правила сканирования' (Scanning rules), and 'Наборы правил' (Rule sets). Below these are sections for 'Интеграция с API' (API integration), 'Центр уведомлений' (Notification center with 248 items), 'Параметры' (Parameters), 'Помощь' (Help), and 'О продукте' (About product). The main content area is titled 'Проект WebMaster-IT' and shows a '+ Добавить репозиторий' (Add repository) button. Below is a search bar and a 'Выделить все' (Select all) checkbox. A table lists repositories with columns for 'Репозиторий', 'Ветки' (Branches), 'Набор правил' (Rule set), 'Команда' (Team), 'Последнее сканирование' (Last scan), and 'Уровень риска' (Risk level). The table contains 12 rows of data, each with a checkbox, repository name, branch count, rule set, team, scan date, and risk percentage. A pagination bar at the bottom indicates 'Показано 1-15 из 20' (Showing 1-15 of 20) and 'Показывать по 15' (Show 15).

Репозиторий	Ветки	Набор правил	Команда	Последнее сканирование	Уровень риска
Название репозитория	2	XSS and SQLi only	ХК	Нет данные	0%
Название репозитория	4	Default	ХК Т	15.08.2024, 14:32	100%
Очень_длинное_название_р	12	Default	ХК Т Д И +999	15.08.2024, 14:32	12%
Название репозитория	2	Default	ХК Т Д	15.08.2024, 14:32	37%
Очень_длинное_название_р	12	Default	ХК Т Д И +999	15.08.2024, 14:32	12%
Название репозитория	2	Default	ХК Т Д	15.08.2024, 14:32	37%
Название репозитория	5	Default	ХК Т	15.08.2024, 14:32	25%
Название репозитория	5	Default	ХК Т	15.08.2024, 14:32	25%
Очень_длинное_название_р	12	Default	ХК Т Д И +999	15.08.2024, 14:32	12%
Название репозитория	2	Default	ХК Т Д	15.08.2024, 14:32	37%

Проект – это логическая группа репозитория, объединенных общей целью.

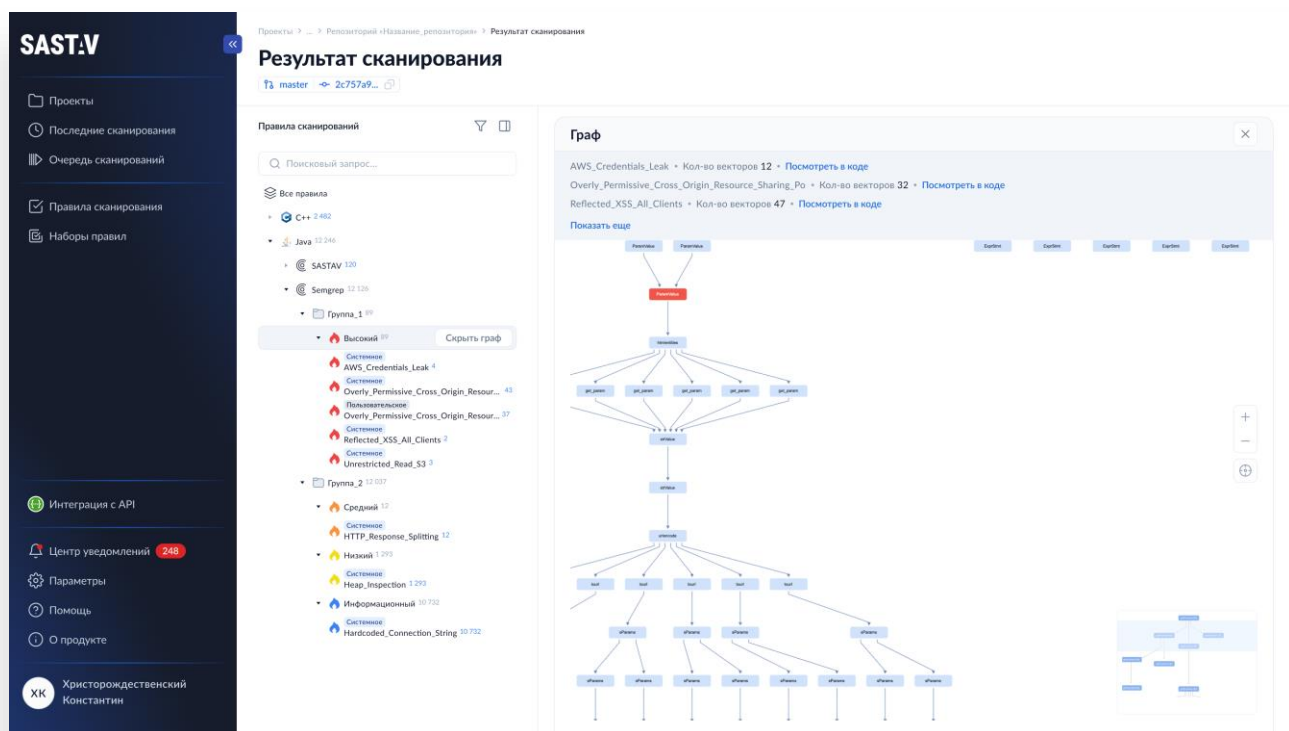
В разделе отображаются наборы правил сканирования, применяемые к конкретному репозиторию, ответственная команда и актуальная оценка рисков. Доступны функции множественного выбора и дальнейшей детальной настройки непосредственно из списка репозитория.

ИНТЕРФЕЙС «РЕПОЗИТОРИЙ»

The screenshot displays the SAST.V interface. On the left is a dark sidebar with navigation items: 'Проекты' (5), 'Последние сканирования', 'Очередь сканирований', 'Правила сканирования', 'Наборы правил', 'Интеграция с API', 'Центр уведомлений' (248), 'Параметры', 'Помощь', 'О продукте', and 'ХристоРождественский Константин'. The main area shows the 'Репозиторий Form service' with a 'Начать сканирование' button. Below is a 'Список сканирований' table with columns for Status, Risk Level, Start Date, End Date, Scan Time, Branch, Commit, Languages, and Strs. The table lists five scans with varying risk levels and completion statuses.

Статус	Уровень риска	Дата начала	Дата завершения	Время сканирования	Ветка	Коммит	Языки	Стр.
47% В процессе	High	29.05.2024	Нет данных	Нет данных	ms020	e2e5585416...	+2	1 47
82% Флажка сканирования	High	29.05.2024	15.08.2024	13 ч. 24 мин. 08 сек.	ms021	e2e5585416...	+8	1 47
100% Завершено	100%	29.05.2024	15.08.2024	9 мин. 08 сек.	ms012	e2e5585416...	+1	1 47
100% Завершено	17%	29.05.2024	15.08.2024	8 сек.	ms022	e2e5585416...	+5	1 47
100% Завершено	32%	29.05.2024	15.08.2024	38 мин.	ms023	e2e5585416...	+12	1 47

Раздел предоставляет данные о проводимых сканированиях. Здесь отображается список всех проверок с четкой цветовой индикацией, позволяющей мгновенно оценить уровень риска. Для каждого сканирования указан применяемый пресет - готовый набор правил проверки. Доступны гибкие фильтры, функция множественного выбора для групповых операций и возможность перехода к детальным настройкам. Вся ключевая информация собрана на одной странице.



В разделе «Результат сканирования» доступно представление в виде графов. Граф визуализирует цепочки уязвимостей, выделяя критические точки пересечения. Исправление таких узловых элементов автоматически устраняет все зависимые уязвимости ниже по графу, что даёт максимальный эффект при минимальных усилиях. Граф — это не просто красивая картинка, а мощный аналитический инструмент, меняющий подход к устранению уязвимостей.

ИНТЕРФЕЙС «ДЕФЕКТ»

The screenshot displays the SAST.V interface for a specific defect. The sidebar on the left contains navigation elements such as 'Проекты', 'Последние сканирования', 'Очередь сканирований', 'Правила сканирования', and 'Наборы правил'. The main area shows the defect details for #7653493, including a rule name 'Incorrect_Permission_Assignment_For_Critical...', a status 'На проверку', and a list of affected files. The central code editor displays Java code for 'HttpServletResponseHttpServlet.java', highlighting a vulnerability in the 'doGet' method. The right sidebar shows a 'Вектор уязвимости' (Vulnerability Vector) with a list of affected components, including 'getParameter [32]', 'InputStream [21]', and 'parse [48]'. The interface is designed for easy navigation and status management of defects.

На этой странице отображается конкретный дефект, обнаруженный в процессе сканирования и открытый из его результатов. Интерфейс разработан для удобной работы с проблемным участком кода. Справа - вектор уязвимости с возможностью мгновенного перехода по связанным участкам программы. В верхней части экрана - мультивкладки, позволяющие осуществлять навигацию по коду. Для ускорения workflow прямо на странице доступна функция смены статуса дефекта в рамках процесса триажа.

ИНТЕРФЕЙС «НАСТРОЙКА ПРАВИЛ СКАНИРОВАНИЯ»

Правила сканирований

Настройка правил сканирований

Редактор правил

Правила сканирований

Java Список правил

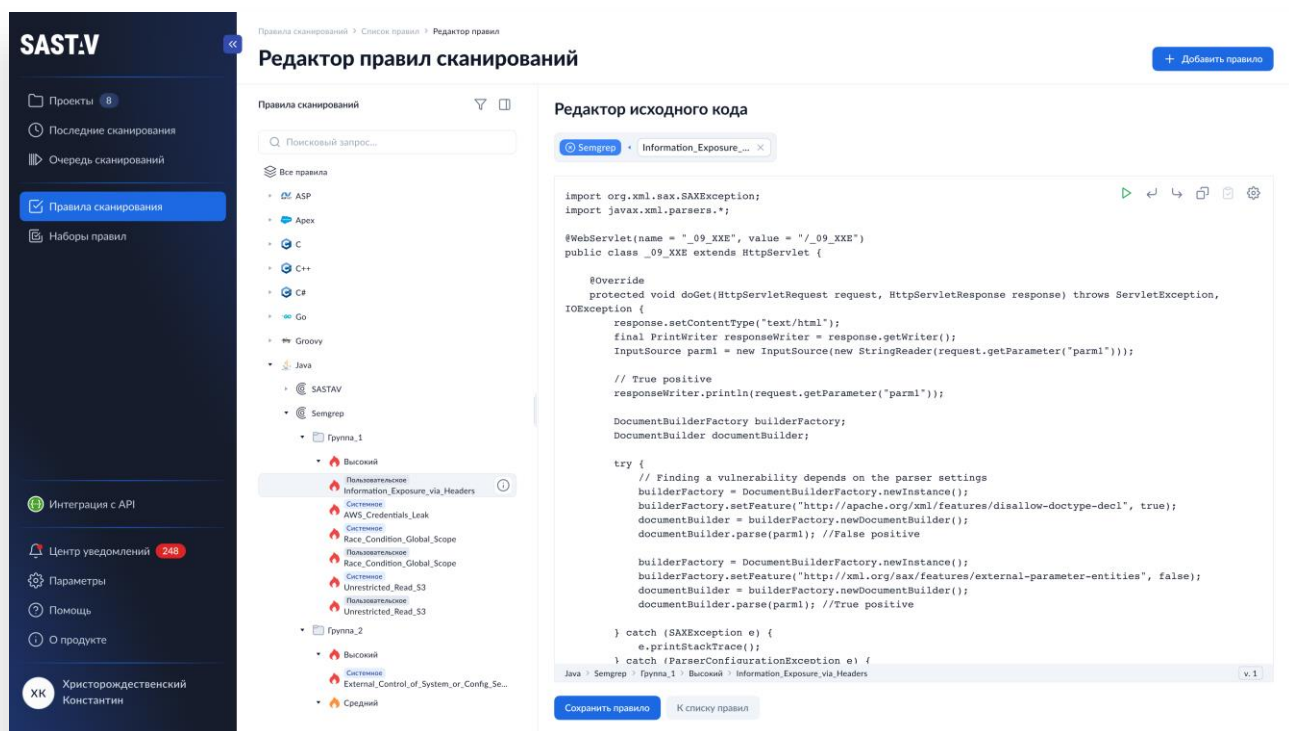
Найти по названию правила

Правило	Уровень критичность	Тип правила	Тип ядра	Режим валидации AI
Reflected_XSS_All_Clients	Высокий	Системное	SASTAV	<input type="checkbox"/>
Incorrect_Permission_Assignment_For_Critical_Res...	Высокий	Системное	SASTAV	<input type="checkbox"/>
Information_Exposure_Through_an_Error_Message	Высокий	Системное	SASTAV	<input type="checkbox"/>
Unrestricted_File_Upload	Средний	Системное	Semgrep	<input checked="" type="checkbox"/>
Unrestricted_File_Upload	Средний	Пользовательское	Semgrep	<input checked="" type="checkbox"/>
Information_Exposure_Through_an_Error_Message	Низкий	Системное	Semgrep	<input checked="" type="checkbox"/>
Missing_Content_Security_Policy	Низкий	Системное	Semgrep	<input checked="" type="checkbox"/>
Information_Exposure_Through_an_Error_Message	Низкий	Пользовательское	Semgrep	<input checked="" type="checkbox"/>
Incorrect_Permission_Assignment_For_Critical_Res...	Информационный	Системное	Semgrep	<input checked="" type="checkbox"/>
Reflected_XSS_All_Clients	Информационный	Системное	Semgrep	<input checked="" type="checkbox"/>
Incorrect_Permission_Assignment For Critical Res...	Информационный	Системное	Semgrep	<input checked="" type="checkbox"/>

Показано 1-15 из 1 201 Показывать по 15

В разделе представлен набор доступных правил сканирования, где каждое правило идентифицируется как пользовательское (созданное вручную) или системное (предустановленное в решение). Для каждого правила указан соответствующий движок анализа, для которого оно предназначено, что позволяет пользователям быстро ориентироваться в доступных возможностях и понимать, какие именно технологии анализа будут задействованы при применении конкретных правил.

ИНТЕРФЕЙС «РЕДАКТОР ПРАВИЛ СКАНИРОВАНИЯ»



Инструмент для создания и редактирования пользовательских правил статического анализа с поддержкой мультивкладок в верхней части интерфейса позволяет одновременно работать с несколькими правилами для разных движков анализа. Редактор оснащен системой версионирования, которая сохраняет историю изменений каждого правила, обеспечивая возможность отслеживания правок и возврата к предыдущим версиям. В нижней части страницы отображается полный путь к правилу с указанием его владельца.

КАИВ

Каскадная AI-валидация дефектов кода

Каскадная AI-валидация решает две главные проблемы в статическом анализе исходного кода: проблему большого количества ложных срабатываний (False Positive) и проблему трудозатрат.

При статическом анализе кода часто генерируются большие объемы данных, включая ложные срабатывания и нерелевантные ошибки. Чем масштабнее проверяемые проекты и чем больше их количество, тем больше будет таких ошибок. Это неизбежно. Неудивительно, что триаж большого числа сработок инструмента отнимает у разработчиков и профильных инженеров много времени. Для компаний это выливается в значительные финансовые затраты.

Проблема большого количества False Positive решается за счет использования в КАИБ AI-модели для автотриаж дефектов (в рамках поддерживаемых языков), точность работы которого довольно высока: менее 20% ложных сработок. Иными словами, AI-валидация снижает количество False Positive до 20% от исходного объёма, устраняя необходимость ручного анализа и отсеивания нерелевантных данных в 80% случаев. Представьте, что КАИБ – это что-то вроде адвоката, который пытается оправдать каждую сработку SAST, собирая её «алиби». В нашей концепции изначально «виновны» все. Благодаря такому подходу мы смогли достичь показатель в 0% допущенных False Negative.

Применяя подход с использованием векторов и точек пересечения на практике, можно заметить, что из большого количества сработок - после построения графов - выявляются пересечения в ключевых точках нескольких уязвимостей. Эти пересечения представляют собой связанные точки или узлы, позволяя дополнительно проанализировать изменения данных вдоль вектора: что с ними происходило и где исправление дефекта будет оптимально. Сосредоточив усилия на исправлении корневой проблемы в такой точке, можно устранить сразу несколько уязвимостей. Это значительно сокращает объем трудозатрат, позволяет сосредотачиваться на ключевых проблемах, глубже анализировать их и предотвращать их появление в будущем.

Таким образом, проблема трудозатрат решается за счёт каскадного механизма, построения векторов уязвимостей, дальнейшего формирования узлов как точек их пересечения. Исправление корневых дефектов приводит непосредственно к уменьшению трудозатрат на исправление, исключая необходимость в исправлении побочных дефектов.

Подобный подход имеет еще и психологическое воздействие на специалиста: когда ты знаешь, что тебе не придётся делать бесполезную работу, что все твои действия эффективны, мотивация устранить проблеме выше. Всегда.

Методология каскадной валидации с использованием искусственного интеллекта основывается на поэтапной работе системы, в ходе которой на каждом этапе выявляются ключевые точки, содержащие векторы потенциальных уязвимостей. Эти критические области становятся основным фокусом последующего анализа. Такой подход не только акцентирует внимание на наиболее значимых аспектах, но и позволяет глубже погрузиться в контекст, отслеживая путь данных в коде. Это включает анализ ограничений, этапов преобразований и их взаимодействий.

Преимущество метода заключается в способности автоматически проводить триаж, эффективно отфильтровывая нерелевантные сигналы и достигая высокой точности в определении проблемных областей. В результате минимизируется вероятность пропуска значимых уязвимостей (False Negatives), что позволяет обеспечить надежность и устойчивость системы.

Основными критериями для AI-ассистента при приоритизации найденных уязвимостей являются следующие:

1. Уровень критичности уязвимости (Severity).

Это непосредственно то, насколько серьезно уязвимость может повлиять на систему. Оценка выносится на основе стандартов, таких как CVSS (Common Vulnerability Scoring System).

2. Вероятность эксплуатации - вероятность того, что уязвимость будет использована злоумышленником.

Является важным фактором при приоритизации. Определяется наличием публичных эксплоитов и сложностью их эксплуатации.

3. Зависимости и каскадный эффект.

Определяется тем, может ли одна уязвимость вызвать цепную реакцию, приводящую к другим проблемам.

В результате приоритизации по этим критериям получаем фокус на наиболее критичных уязвимостях (разработчики устраняют проблемы, которые могут нанести наибольший вред, вместо того чтобы тратить время на исправление менее значимых ошибок). За счёт корреляции этих параметров уменьшаем ложные и нерелевантные срабатывания, а также сокращаем каскадные проблемы.

AI позволяет не только автоматизировать триаж, но и генерирует рекомендации по изменению кода. Алгоритмы, предназначенные для рекомендаций по изменению кода, в своей основе используют подход, схожий с методологией валидации. Они фокусируются на сборе максимально полного контекста в точках сработок, детально анализируют используемые классы, библиотеки и отслеживают все трансформации данных в рамках конкретного сценария.

На основе собранных данных формируется узконаправленное задание для искусственного интеллекта, который, опираясь на собранный контекст, предлагает оптимальные изменения. Этот метод отличается высокой степенью адаптивности и универсальности, что делает его применимым к проектам любого уровня сложности и масштаба.

Внедрение AI-ассистента в процесс статического анализа кода является важным шагом в направлении Shift Left Security, акцентируя внимание на проактивном подходе к безопасности и снижению Time-to-Market (TTM). Такой инструмент не просто выполняет анализ кода, а предоставляет глубоко проработанный и детализированный отчет. Вместо типичного набора сработок AI предлагает уже отфильтрованные, приоритизированные и агрегированные данные, включая целые векторы уязвимостей, сопровождаемые конкретными рекомендациями по их исправлению. Это существенно повышает качество устранения уязвимостей, снижает время на их обнаружение и обработку, а также уменьшает затраты ресурсов.

Глобально это приводит к сокращению затрат на устранение дефектов, которые могли бы быть обнаружены только на поздних этапах разработки или внедрения, где цена исправлений возрастает многократно.

КЛЮЧЕВЫЕ ПРЕИМУЩЕСТВА ПОДХОДА:

Повышение качества исправлений

Конкретные рекомендации и анализ критичности позволяют команде сосредоточиться на наиболее важных аспектах.

Ускорение реакции

Снижение времени от момента обнаружения проблемы до ее устранения.

Оптимизация трудозатрат

Значительное уменьшение усилий команды на ручной триаж и анализ.

Разработка компании ShiftLeft Security

115114, г. Москва, ул. Дербеневская,
д. 15Б, помещ. 2/1, 3 этаж, офис 306

+7 (499) 502-13-75
support@sastav.ru



sastav.ru