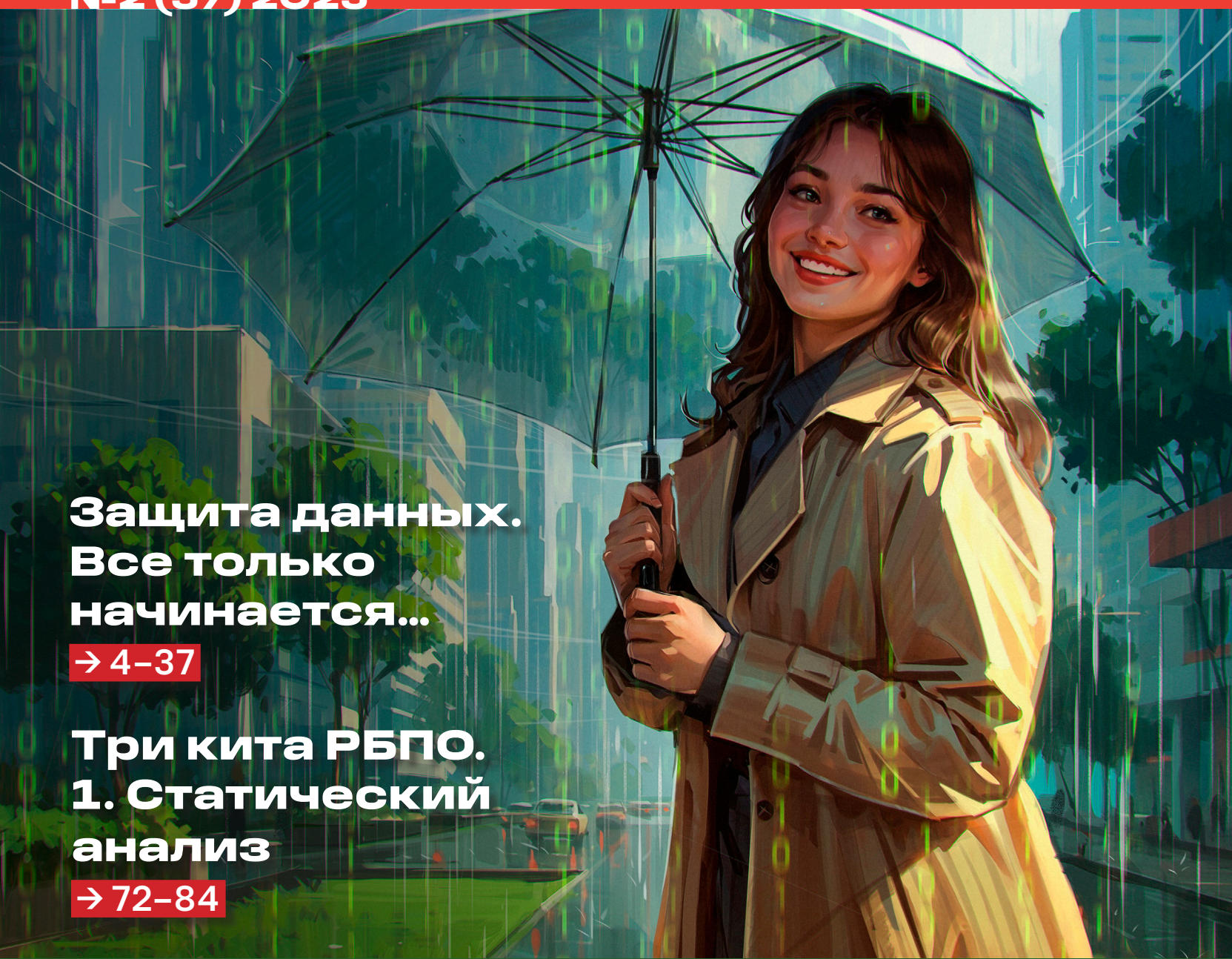
BIS JOURNAL

ИНФОРМАЦИОННАЯ БЕЗОПАСНОСТЬ БИЗНЕСА

№2 (57) 2025

Игорь ШЕРЕМЕТ
РАН
К независимости без изоляции
→ 38

Василий ПИСКАРЕВ
Госдума
«Мы работаем!»
→ 48



Защита данных. Все только начинается...
→ 4–37

Три кита РБПО. 1. Статический анализ
→ 72–84

Елена БУТКОВА
Мингосуправления Московской области
«Даю согласие на обработку...» → 8

Дмитрий ПОНОМАРЕВ
ООО НТЦ «Фобос-НТ»
Испытания статических анализаторов → 72

Андрей ЖАРКЕВИЧ
Start X
Анатомия корпоративного конфликта → 85

Сергей ПАЗИЗИН
научный редактор
BIS Journal

В связи со вступлением в действие 30 мая 2025 года закона «Об оборотных штрафах» за утечку информации (ФЗ от 30.11.2024 № 420-ФЗ «О внесении изменений в Кодекс Российской Федерации об административных правонарушениях») кардинально ужесточается ответственность за действия и бездействие, повлекшие за собой утечку персональных данных. Это серьезный повод для компаний пересмотреть свое отношение к защите данных — проанализировать слабые места и выйти на новый уровень зрелости в части обеспечения безопасности информационных ресурсов.

В данном номере мы подробно рассмотрим изменения в законодательстве о защите персональных данных и предоставим возможность экспертам высказать свое мнение по данному вопросу. Расскажем о необходимости инвентаризации данных, комплексном подходе к обеспечению безопасности информации и используемых технических средствах.

В нашей постоянной рубрике о разработке безопасного программного обеспечения проинформируем о проводимых под патронажем ФСТЭК России испытаниях открытых и коммерческих отечественных статических анализаторов исходных кодов компилируемых и динамических языков программирования. Целью указанных испытаний является оценка технологических возможностей анализаторов в соответствии с требованиями ГОСТ Р 71207–2024 в части полноты обнаружения критичных дефектов, количества ложных срабатываний, поддерживаемых методов анализа и потребности в ресурсах. Читатель познакомится с планом этих испытаний и их основными участниками. Информацию о применяемых техниках тестирования и результатах будем публиковать на страницах журнала по мере поступления.

Также в целях повышения эффективности взаимодействия и обмена опытом между научным и практическим направлениями в сфере обеспечения информационной безопасности мы начинаем сотрудничество с журналом «Вопросы кибербезопасности», занимающим ведущие позиции в области публикации научных результатов. Планируем в дальнейшем обмениваться с коллегами рефератами наиболее интересных статей и новостными материалами.

BIS JOURNAL
ИЗДАЁТСЯ С 2010 ГОДА

УЧРЕДИТЕЛЬ-ИЗДАТЕЛЬ
ООО «Медиа Группа „Авангард“»
Адрес: 127473, Россия, Москва, 3-й Самотечный переулок, д. 21
Тел.: +7 495 921 42 44
Интернет-версия: ib-bank.ru/bisjournal
E-mail: bis@ib-bank.ru

Зарегистрирован 13 августа 2010 года Федеральной службой по надзору в сфере связи, информационных технологий и массовых коммуникаций (Роскомнадзором) как «BIS Journal — Информационная безопасность банков».

Зарегистрирован 11 июля 2023 года Федеральной службой по надзору в сфере связи, информационных технологий и массовых коммуникаций (Роскомнадзором) как «BIS Journal — Информационная безопасность бизнеса». Свидетельство ПИ № ФС77–85487 от 11.07.2023 г.

Подписан в печать 21.03.2025. Тираж 7000 экз.
Цена договорная. Отпечатан в типографии «Московский Печатный Двор», 123298, г. Москва, 3-я Хорошевская 18, корп. 1, офис 201А

 16+

ИСПЫТАНИЯ СТАТИЧЕСКИХ АНАЛИЗАТОРОВ

Дмитрий ПОНОМАРЕВ
заместитель генерального директора — директор департамента внедрения и развития практик РБПО ООО НТЦ «Фобос-НТ», сотрудник ИСП РАН, преподаватель МГТУ им. Н. Э. Баумана

Испытания открытых и коммерческих статических анализаторов исходных кодов компилируемых и динамических языков программирования под патронажем ФСТЭК России [1] — это уникальное для отечественного рынка событие, свидетельствующее как о растущем уровне зрелости рынка и требований [2], так и о востребованности технологии статического анализа для организации процессов разработки безопасного и качественного ПО (далее — РБПО).

Особый интерес к испытаниям вызван формулировкой их целей (см. ниже) и антицелей. Во главу угла ставятся именно технологические возможности «движков» инструментов, определяемые и регламентируемые ГОСТ Р 71207–2024 [3].

Разумеется, конечного пользователя при выборе инструмента могут интересовать и иные критерии — наличие интеграционного функционала, наличие детекторов на актуальные «мисконфиги» (под этим термином понимается класс уязвимостей, связанных с неправильной настройкой ПО, например, установленные в настройках «по умолчанию» слабые режимы шифрования трафика веб-сервера), и даже наличие понятной маркетинговой стратегии разработчика инструмента. Естественно и желание увидеть некую совокупную метрику, сочетающую в себе «тёплое с мягким умноженное на цветовые предпочтения» — аналог точки в знаменитом «Гартнеровском квадрате».

Тем не менее ФСТЭК России сознательно фокусируется именно на испытаниях возможностей «движков» инструментов. Поиск различных актуальных «мисконфигов», в том числе снижающих защищенность системы, бесспорно одна из важных задач при обеспечении реальной безопасности. Качественно реализованный функционал интеграции с иными корпоративными сервисами и вспомогательным инструментами — такими как службы домена, корреляторы ошибок и багтрекеры — это практически безальтернативное свойство любого анализатора, планирующегося к применению в крупной организации. Наличие понятных обучающих и маркетинговых материалов, равно как и собственной «школы» внедренцев и последователей, разумеется, также способствует комфорту конечного пользователя. Однако реализация всего вышеуказанного, как правило, представляет собой всё-таки «задачи», а не «проблемы». Число инженеров и иных специалистов, умеющих решать данные задачи, в России и Мире бесспорно многократно выше, чем число узкопрофильных специалистов, умеющих создавать, развивать и поддерживать «сердца» инструментов статического анализа.

Именно поэтому в фокусе испытаний именно «движки» инструментов — как отечественных, так и подмножества «открытых» — это должно подчеркнуть важность создания и развития соответствующих отечественных «школ», способных в том числе составлять конкуренцию на мировом рынке. А одной из важнейших целей испытаний является выработка методики оценки возможностей движков, которая в дальнейшем может быть использована не только для оценки возможностей инструментов, но и в качестве актуализируемого обобщенного Технического Задания, на которое смогут ориентироваться начинающие коллективы, желающие попробовать свои силы в создании и развитии статических анализаторов.

[1] https://ib-bank.ru/bisjournal/news/21667
[2] https://ib-bank.ru/rbpo_pubs
[3] https://clck.ru/3HsNTZ

КРАТКАЯ ХАРАКТЕРИСТИКА ИСПЫТАНИЙ

Цели испытаний

- оценка технологических возможностей отечественных и открытых статических анализаторов исходного кода
- формирование публичных методик оценки и критериев применимости анализаторов в различных видах исследований и испытаний

Критерии оценки

- соответствие требованиям ГОСТ Р 71207–2024 в части:
  - полноты обнаружения критичных дефектов
  - избыточности ложных обнаружений дефектов
  - поддерживаемых методов анализа
  - ресурсозатратности анализа
- соответствие дополнительным критериям, вырабатываемым жюри

Предметная область испытаний

- компилируемые языки программирования (Си / Си++ / Java / Go / C#)
- интерпретируемые языки программирования с динамической типизацией (Python / JavaScript)

Не войдут в критерии оценки

- наличие сертификатов тех или иных регуляторов или добровольных сертификаций
- интеграционный функционал и возможности инструментов по встраиванию, взаимодействию с иными видами анализаторов
- факты использования технологий искусственного интеллекта «в движке», либо «для приоритизации»
- функционал, требующий запуска ПО и его анализа в динамическом режиме
- активность маркетологов и рекламщиков; число статей и выступлений на коммерческих конференциях.

ПЛАН ИСПЫТАНИЙ

№	Шаг	Дата начала	Дата окончания	Ответственные
1	Анонс испытаний на Открытой конференции ИСП РАН	11.12.24	12.12.24	ФСТЭК России
2	Формирование жюри и установочная встреча	10.01.25	10.02.25	ФСТЭК России
3	Повторный анонс на ТБ Форум	12.02.25	13.02.25	ФСТЭК России
4	Подбор компонентов с открытым исходным кодом и разработка комплекта тестов для основного этапа испытаний	10.04.25	01.07.25	ФСТЭК России, жюри
5	Разработка комплекта тестов «Домашнее задание»	01.01.25	25.03.25	ИСП РАН, разработчики СА, жюри
6	Согласование комплекта тестов «Домашнее задание»	25.03.25	10.04.25	ФСТЭК России, жюри
7	Проведение этапа «Домашнее задание»	10.04.25	15.05.25	разработчики СА
8	Подготовка декларативных отчётов по результатам этапа «Домашнее задание»	15.05.25	25.05.25	разработчики СА
9	Подготовка и публикация пресс релиза по результатам этапа «Домашнее задание»	25.05.25	01.06.25	ФСТЭК России, Медиа Группа «Авангард»
10	Подготовка инфраструктуры основного этапа испытаний	01.03.25	01.07.25	МГТУ им. Н. Э. Баумана, разработчики СА, жюри
11	Выполнение анализа в подготовленной инфраструктуре в ходе основного этапа испытаний	01.07.25	08.07.25	МГТУ им. Н. Э. Баумана, жюри
12	Разметка результатов выполненного анализа	08.07.25	01.11.25	жюри, разработчики СА, дополнительно привлечённые организации-эксперты
13	Кроссверификация результатов выполненного анализа	08.07.25	01.11.25	жюри, дополнительно привлечённые организации-эксперты
14	Подготовка технического отчета по результатам основного этапа испытаний	01.10.25	01.11.25	жюри
15	Подготовка итогового отчета о результатах испытаний	01.11.25	01.12.25	жюри, ФСТЭК России
16	Выступление ФСТЭК России, посвященное результатам испытаний		до 15.12.25, открытая конференция ИСП РАН	ФСТЭК России

КРИТЕРИИ ОЦЕНКИ НА ЭТАПЕ «ДОМАШНЕЕ ЗАДАНИЕ»

Комплект тестов этапа «Домашнее задание» доступен всем участникам испытаний, а оценка результатов будет носить декларативный характер. Каждая организация-участник, получив комплект тестов и, проведя испытания самостоятельно, также самостоятельно заполнит типовую анкету зафиксировав там результаты, полученные в ходе анализа комплекта тестов. Данные результаты будут переданы жюри и организаторам испытаний без какой-либо дополнительной верификации — цель данного этапа это не только верхнеуровневая оценка возможностей инструментов, но и выравнивание позиций и восприятия коллективами участников и жюри.

В критерии оценки этапа «Домашнее задание» войдут количественные показатели обнаружения заложенных в атомарные тесты дефектов кода, а также некоторые качественные характеристики инструментов, такие как маппинг детекторов на «пункты» ГОСТ 71207–2024, наличие документации по всем типам ошибок и выдача результатов разметки в открытых форматах.

КОММЕНТАРИИ ОТ ОРГАНИЗАЦИЙ-УЧАСТНИКОВ ИСПЫТАНИЙ

ИСП РАН
(участник испытаний)

Испытания статанализаторов — важная инициатива регулятора, которая решает сразу несколько задач. Во-первых, раскрывает сильные и слабые стороны анализаторов на едином наборе тестов как для пользователей-вендоров, так и для разработчиков самих анализаторов. Во-вторых, «обкатывает» положения недавно принятого ГОСТ 71207 в отношении к уровню технологий анализа и классам критических ошибок на ряде популярных языков программирования. Можно ожидать, что по итогам испытаний будут, кроме прочего, сформированы предложения по развитию требований ГОСТ. Наконец, методика испытаний в дальнейшем сможет стать основой для оценки и других инструментов, в частности, динамических анализаторов. Также важно, что все ведущие российские разработчики анализаторов откликнулись на предложение ФСТЭК. Совместно с коллегами приятно работать и обмениваться опытом, и мы уверены, что результаты испытаний будут интересны и полезны для всего сообщества.

ПВС
(участник испытаний)

Благодаря инициативе ФСТЭК у разработчиков ПО и сертификационных лабораторий появится единая методика оценки инструментов при их выборе.

В ГОСТ Р 71207–2024 (раздел 7 и 8) перечислены требования, предъявляемые к инструментам СА. Однако пользователям затруднительно самостоятельно проверить, насколько им соответствует анализатор. Нужен большой набор синтетических (малых) и реальных (больших) тестов, проверяющих, что выявляются все типы критических ошибок (п. 6.7). Более того, тесты должны проверить, что детекторы СА реализованы на должном уровне и обеспечивают глубокий анализ кода. Имеется в виду, что тесты следует делать разносторонними и иногда сложными, чтобы убедиться, что в инструменте реализованы методы анализа, перечисленные в п. 7.4., например анализ потока данных.

Всё это возможно, но ресурсозатратно. Дополнительная сложность, что пользователи инструментов просто не обладают такой же экспертизой, какая есть у команд, разрабатывающих СА. Непонятно, как убедиться, что разработанные тесты действительно проверяют все важные аспекты.

Поэтому очень важно, что ФСТЭК привлёк к созданию методики сами команды, разрабатывающие инструменты анализа. Благодаря совместным усилиям можно подготовить хорошие наборы тестов и провести их кросс-верификацию.

Ценность разрабатываемой методики в том, что затем можно взять любой инструмент и достаточно быстро, надёжно и единообразно оценить, насколько он соответствует требованиям ГОСТ Р 71207–2024.

Существование публичной методики испытаний будет стимулировать вендоров адаптировать инструменты к особенностям стандарта, что в итоге поможет пользователям. Например, для испытаний потребуется готовый набор детекторов для выявления критических ошибок. Сложно внедрять СА в процесс РБПО, если такого набора нет и пользователям самим предстоит его составлять, рискуя ошибиться.

Базальт СПО
(жюри и участник испытаний)

В ходе испытаний будут выработаны материалы и способы сравнения статических анализаторов, проявятся их преимущества и ограничения. Это позволит разработчикам, в том числе в «Базальт СПО», обоснованно выбирать инструменты в зависимости от конкретных задач.

«Базальт СПО» предложила внести в программу испытаний свободные статические анализаторы Clang Static Analyzer и CppCheck, нагрузку по сопровождению которых полностью возьмут на себя специалисты компании.

Мы также готовы способствовать открытой публикации тестов, материалов и техники тестирования не только для участников испытаний, но и для всего мирового сообщества.

SVACE

Андрей БЕЛЕВАНЦЕВ
руководитель направления анализа и оптимизации программ ИСП РАН, доктор физико-математических наук, профессор РАН

Svace — семейство статических анализаторов, разрабатываемых в ИСП РАН с 2003 года. Мы гордимся, что:

- прошли путь от фундаментальных исследований по анализу до внедрений в крупнейшие российские и зарубежные компании;
- в рамках совместной лаборатории с компанией Samsung показали, как инновационные технологии могут быть «заточены» под специфику конкретных заказчиков;
- добились результатов, которые помогли определить мировой уровень развития анализа, на который ориентируются у нас и за рубежом при создании регуляторики;
- внесли определяющий вклад в формирование научной специальности «Кибербезопасность» в России.

Svace сегодня — это:

- 10 языков, более 70 классов критических ошибок и более 1300 детекторов
- передовой уровень технологий — анализ всей программы целиком (межпроцедурность), раздельный анализ путей выполнения (символьное выполнение), анализ чувствительных данных, анализ статистики, точное построение графа вызовов (девиртуализация). И это далеко не все!
- работа с результатами анализа через графический интерфейс Svacer, в котором можно изучить код и разметить предупреждения небольшой программы, а можно вести десятки проектов различных команд с регулярным проведением анализа, переносом разметки между версиями, пользовательскими фильтрами и статистикой. Именно Svacer поддерживает работу Центра исследований безопасности системного ПО, объединяющего десятки компаний
- программный интерфейс для создания пользовательских детекторов-плагинов, который в простой декларативной манере дает доступ к части анализов из основного «движка». В итоге создание детектора для поиска несложных, специфичных для проекта ошибок сравнимо по трудоемкости с простыми шаблонами запросов для анализаторов уровня синтаксического дерева, но при этом дает значительно более точный результат. Пользователи также могут делиться своими детекторами друг с другом на площадке Svace, получать консультацию разработчиков и предоставлять свои плагины в качестве образцов (подробнее в нашем блоге)
- AI-ассистент для разметки предупреждений анализатора, позволяющий оценивать вероятность истинности предупреждения на основе машинного обучения, управляемого историей разметки (для языков C, C++, C#, Java). Базовыми признаками для оценки служат метрики исходного кода программы, которые дополнительно собираются в ходе анализа. Наилучшая точность предсказания достигается при учете данных прошлой пользовательской разметки. Если результаты анализа загружаются на сервер с такими данными, то дообучение модели для предсказания происходит автоматически (подробнее в нашем блоге).

Svace активно развивается: мы внедряем результаты научных исследований и реализовываем необходимый индустрии функционал. Новые версии Svace выходят ежегодно, а обновления текущей стабильной версии выпускаются каждые два-три месяца. Среди примеров изменений двух последних лет:

- поддержка последних версий языков (C++20, Java 21, Go 1.24, Kotlin 2.0, C# 12)
- инфраструктура универсального синтаксического дерева (UAST), которая унифицирует простые детекторы между языками и позволяет в ускоренном темпе добавлять новые. В 2024 году на ее основе добавлен язык JavaScript
- улучшение анализа чувствительных данных и расширение возможностей по пользовательским спецификациям источников и стоков
- работа с предупреждениями через комментарии в коде
- роли пользователей, гибкие фильтры предупреждений, статистика разметки, глобальный полнотекстовый поиск в интерфейсе Svacer

Svace — результат многолетних исследований научной школы системного программирования ИСП РАН, реализованных в инновационном продукте благодаря обратным связям от российской и зарубежной индустрии. Svace внедрен более чем в 200 компаниях, среди которых – ключевые игроки в области ИТ и кибербезопасности, крупнейшие финансовые организации (в том числе ведущие банки России), а также предприятия телекоммуникационного сектора. Вместе мы обеспечиваем технологическую независимость российского бизнеса и всей страны.

Будем рады видеть вас в числе наших заказчиков!

Инструменты анализа ИСП РАН

Блог Svace

СТАТИЧЕСКИЕ АНАЛИЗАТОРЫ В КОММЕРЧЕСКОЙ РАЗРАБОТКЕ

ОДНА ГОЛОВА ХОРОШО, А НЕСКОЛЬКО — ЛУЧШЕ

Дмитрий ШМОЙЛОВ
Head of Software Security, Kaspersky

Любая компания, которая начинает внедрять процессы безопасной разработки, в первую очередь задумывается об инструменте статического анализа. С него, как правило, начинается «проба пера» по внедрению практик безопасной разработки. Статические анализаторы являются пилотом и первой ласточкой в коммуникации зарождающейся роли Application Security (подразделения, ответственного за развитие безопасной разработки) с командой разработки ПО.

Это вполне объяснимо, так как данный инструмент ближе программисту. Команды разработки так или иначе всегда использовали инструментарий статического анализа (начиная с IDE плагинов и простых линтеров) самостоятельно, без запроса со стороны Application Security. Уже позже развитие отношений между Application Security и разработкой перейдёт на новый уровень доверия — использование динамических анализаторов, проведение обучений по тематике безопасной разработки, SCA и т.д., а некоторые даже попробуют фаззинг... Но оставим их и вернёмся к «статике».

Статические анализаторы существуют давно и в большом разнообразии представлены на рынке, как в open-source, так и среди коммерческих решений. При выборе подходящего решения компаниям приходится ответить на множество вопросов: что лучше, что удобнее, что эффективнее? И самое главное — как это применить к нашим реалиям, процессам, процедурам сборки (пайплайнам)?

Поразмыслив ещё, они понимают: «Не мы первые, кто задаётся таким вопросом». А что у «коллег по цеху»?

Думаем, что регулятор инициировал сравнение инструментов статического анализа (SAST), в том числе с намерением ответить и на этот вопрос.

Мы в «Лаборатории Касперского» работаем с SAST каждый день, на большом объёме кода, знаем и понимаем данную практику, ежедневно видим пользу от её применения. В нашей практике встречались случаи, когда разные инструменты находили одни и те же баги, а также когда ни один из используемых SAST не находил ошибку, которая должна обнаруживаться подобными решениями.

Использование статических анализаторов в ходе разработки наших продуктов — обязательное требование для всех коммерческих проектов, это естественная процедура, как помыть руки перед едой.

В ряде критичных проектов используют несколько анализаторов последовательно. Часто эксперты пишут свои дополнительные правила для инструментов, чтобы расширить их детектирующие возможности, адаптируя их под специфику нашего кода.

Там, где это возможно, инструменты работают в блокирующем режиме: любое срабатывание останавливает заливку кода в мастер-ветку, пока срабатывание не будет обработано.

Да, специфика работы SAST подразумевает, что предупреждения инструмента содержат немало ложноположительных срабатываний, это неизбежно. Наши эксперты хорошо знают путь к группам поддержки отечественных SAST-инструментов, сообщая им о том, как можно оптимизировать их работу для снижения ложноположительных срабатываний. Тем полезнее будет совместная работа плечом к плечу с разработчиками SAST.

Вопрос необходимости сравнения SAST-инструментов мы поднимали в процессе согласования ГОСТ по статическому анализу.

Мы всецело приветствуем инициативу и надеемся на выработку полезных критериев по оценке инструментов. Эти критерии дадут компаниям-новичкам в данной области больше информации для выбора, облегчат им «первый старт», а зрелые компании, вероятно, найдут ответы на давно волнующие их вопросы.

В любом случае подобные коллаборации ведущих разработчиков ПО приносят пользу и ускоряют развитие всей индустрии разработки.

Пожелаем нам всем удачи и будем с нетерпением ждать результатов!

ОБ АВТОРЕ

Дмитрий Шмойлов. Более 23 лет опыта в IT, из них 18 — в сфере ИБ (от специалиста до CISO) и разработки (архитектура ИБ, Application Security, SSDLC, DevSecOps). Последние 7 лет руководит AppSec-подразделением в «Лаборатории Касперского», где отвечает за совершенствование процессов безопасной разработки. Сфера профессиональных интересов: безопасная разработка, безопасная архитектура, DevSecOps, пентест, Bugbounty, data privacy (в том числе GDPR).

АНАЛИЗАТОР КОДА ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ АК-ВС 3

AKVS анализатор кода

КОМПЛЕКСНОЕ РЕШЕНИЕ ДЛЯ ОБЕСПЕЧЕНИЯ БЕЗОПАСНОСТИ КОДА В СОВРЕМЕННЫХ РАЗРАБОТКАХ

Виталий ВАРЕНИЦА
заместитель генерального директора АО «НПО «Эшелон»

В эпоху стремительного роста киберугроз защита программного обеспечения становится приоритетной задачей. Компании обязаны соответствовать требованиям регуляторов и обеспечивать безопасность своих решений. Регулярные проверки кода являются ключевым аспектом обеспечения безопасности программного обеспечения. Применение инструментов тестирования не только способствует улучшению качества программного обеспечения, но и обеспечивает соответствие актуальным требованиям регуляторов в области разработки безопасного ПО.

ВВЕДЕНИЕ

АК-ВС 3 представляет собой инструмент, предназначенный для автоматизации процесса проведения испытаний при сертификации в соответствии с требованиями различных регуляторов, а также для выполнения периодических проверок в рамках разработки безопасного программного обеспечения.

ОСНОВНЫЕ ПРЕИМУЩЕСТВА

- Все виды анализа в одном продукте: возможность проведения статического анализа кода (SAST), динамического анализа кода (DAST), полносистемного динамического анализа, фаззинг-тестирования (FAST) в одном продукте в соответствии с требованиями разных регуляторов. В рамках одной лицензии разработчик получает продукт, который комплексно способен проводить сразу несколько независимых видов тестирования.
- Интеграция с CI/CD: АК-ВС 3 легко внедряется в DevSecOps-процессы благодаря консольному клиенту, который позволяет запускать анализ автоматически при каждом коммите. Это помогает обнаруживать уязвимости ещё на этапе разработки, а также автоматизировать проверку кода.
- Работа в собственной среде разработки: встроенная IDE «Эшелониум» на базе Visual Studio Code обеспечивает удобство работы с кодом и анализом результатов, поддерживает патчи, плагины и интеграцию с системами разработки, что исключает необходимость в дополнительных редакторах (рис. 1).

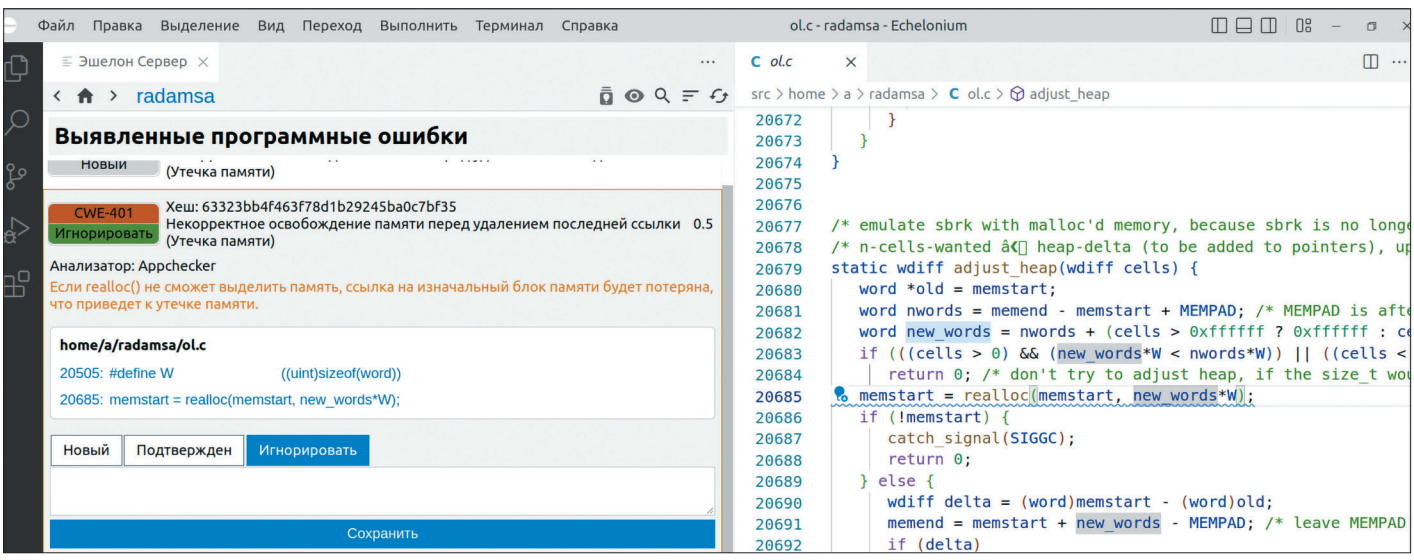


Рисунок 1. Работа в IDE «Эшелониум»

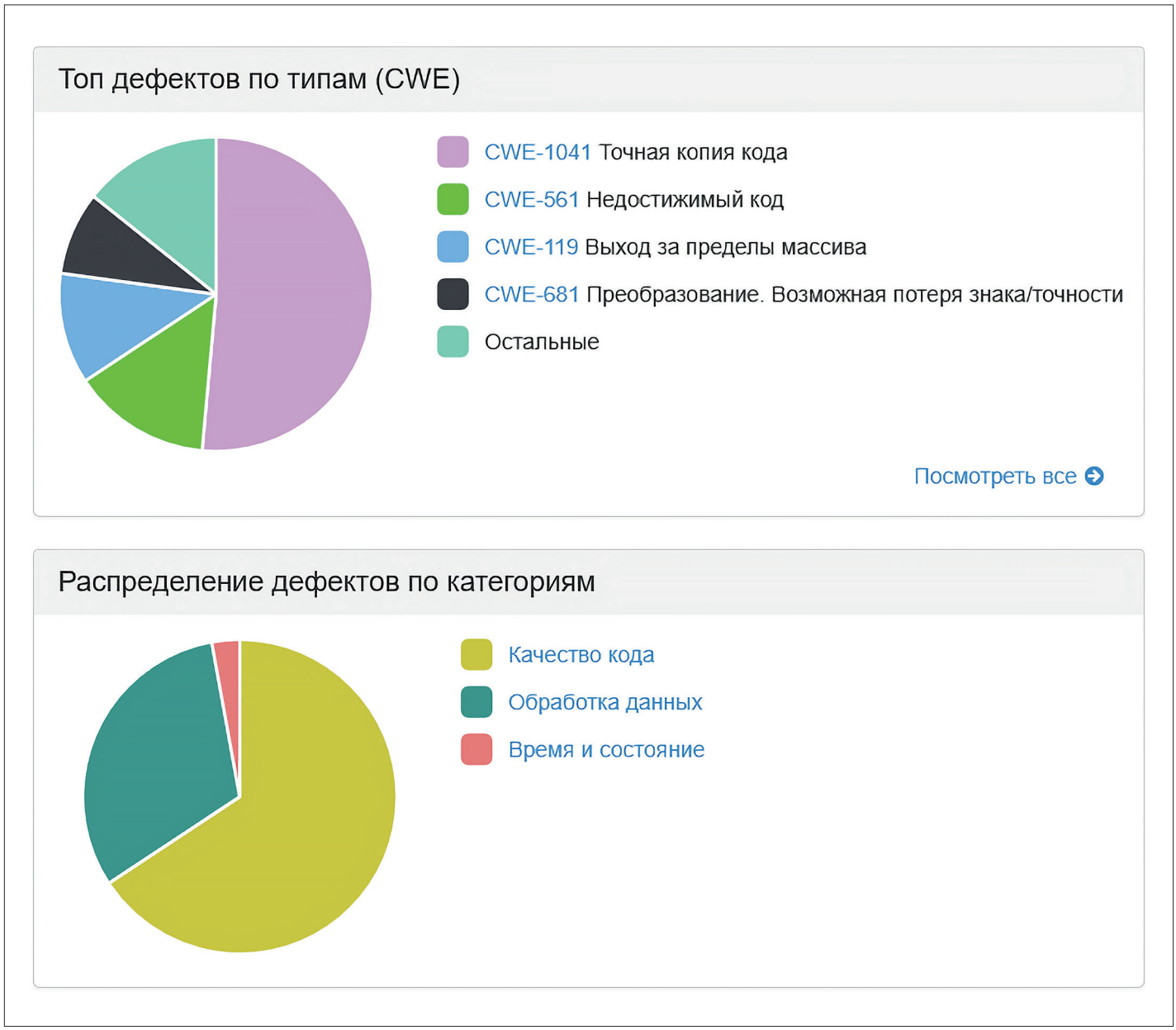


Рисунок 2. Классификация выявленных дефектов

◆ **Соответствия требованиям регулятора:** АК-ВС 3 выполняет требования:

- ФСТЭК России в части:
  - *«Статический анализ объекта оценки (САО)»;*
  - *«Фаззинг-тестирование объекта оценки (ДАО.2)»;*
  - *«Системное тестирование объекта оценки (ДАО.3)».*
- Министерства обороны Российской Федерации в соответствии **с руководящим документом «Защита от несанкционированного доступа к информации Часть 1. Программное обеспечение средств защиты информации. Классификация по уровню контроля отсутствия недекларированных возможностей»** в части:
  - *п. 3 «Статический анализ исходных текстов программ»;*
  - *п. 4 «Динамический анализ исходных текстов программ».*
- Требований по разработке безопасного ПО в соответствии с **ГОСТ Р 56939–2016 «Защита информации. Разработка безопасного программного обеспечения. Общие требования»** в части:
  - *п. 5.3.3.4 «Статический анализ исходного кода программы»;*
  - *п. 5.4.3.3 «Динамический анализ исходного кода программы»;*
  - *п. 5.4.3.4 «Фаззинг-тестирование программы».*
- Требований по разработке безопасного ПО в соответствии с **ГОСТ Р 56939–2024 «Защита информации. Разработка безопасного программного обеспечения. Общие требования»** в части:
  - *п. 5.10 «Статический анализ исходного кода»;*
  - *п. 5.11 «Динамический анализ исходного кода».*

◆ **Проведение различных видов статического анализа исходного кода.** Поддерживается статический анализ исходного кода на уровне:
- сигнатурного анализа;
- анализа потоков;
- межпроцедурного контекстно-чувствительного анализа;
- анализа, чувствительного к путям выполнения.

◆ **Описание дефектов в соответствии с международным стандартом CWE:** представляет собой не просто получение сигнатуры или фиксирование срабатывания, а включает в себя указания соответствующей категории дефекта в системе CWE (рис. 2).

◆ **Построение различных отчётов для статического анализа исходного кода:**
- данные о базовых блоках, функциональных и информационных объектах;
- отчёт по выявленным вставкам кода на низкоуровневых языках программирования;
- список предполагаемых дефектов;
- блок-схемы функциональных объектов.

◆ Построение различных отчётов для динамического анализа исходного кода:
- отработавшие функциональные объекты и связи между ними;
- отработавшие базовые блоки и связи между ними.

◆ **Соблюдение ролевой модели:** гибкое разграничение доступа по ролям позволяет чётко разделять права пользователей и предотвращать несанкционированные изменения, что важно для команд с разными уровнями ответственности и защиты данных.

◆ **Удобство развёртывания:** архитектура «клиент-сервер» позволяет масштабировать систему в крупных компаниях, снижая нагрузку на локальные машины и обеспечивая централизованное управление.

◆ **Параллельность работы:** позволяет проводить одновременный анализ нескольких проектов и координировать действия различных экспертов.

КАК АК-ВС 3 ПОМОГАЕТ БИЗНЕСУ?

Благодаря АК-ВС 3 компании могут:

◆ **повысить безопасность ПО, выявляя уязвимости до релиза:** выявление уязвимостей на ранних стадиях разработки позволяет минимизировать потенциальные риски, связанные с эксплуатацией уязвимостей злоумышленниками;

◆ **соответствовать нормативным требованиям без лишних затрат:** инструмент полностью отвечает требованиям ФСТЭК, Минобороны России, положениям Банка России и ГОСТ Р 56939, упрощая сертификацию и обеспечение регуляторного соответствия;

◆ **снизить затраты на исправление дефектов кода за счёт раннего выявления проблем:** обнаружение уязвимостей на этапе разработки снижает расходы на их устранение по сравнению с исправлением ошибок в выпущенной версии приложения;

◆ **автоматизировать проверки, экономя время команды разработчиков:** АК-ВС 3 легко интегрируется в конвейеры CI/CD, позволяя компаниям внедрять DevSecOps-подход и обеспечивать безопасность без замедления процессов разработки;

◆ **проводить анализ для широкого спектра языков программирования:** АК-ВС 3 анализирует код на C/C++, Java, C#, Python, PHP, Golang;

◆ **использовать интерактивные инструменты визуализации анализа:** механизмы построения графов зависимостей, блок-схем и маршрутов функциональных объектов упрощают понимание кода и ускоряют устранение уязвимостей;

◆ **применять гибкую настройку и расширяемость:** возможность адаптации инструментов анализа под требования конкретного проекта позволяет учитывать специфику используемых технологий и архитектурных решений.

ЗАКЛЮЧЕНИЕ

АК-ВС 3 представляет собой уникальное решение для обеспечения безопасности программного обеспечения, сочетая в себе несколько типов анализа и соответствие нормативным требованиям. Его возможности по интеграции в процессы CI/CD, поддержка множества языков программирования и высокая гибкость делают его мощным инструментом для разработчиков и компаний, стремящихся обеспечить высокий уровень безопасности своих программных продуктов.

СПРАВКА О КОМПАНИИ

Группа компаний «Эшелон» специализируется на комплексном обеспечении информационной безопасности и включает в себя множество отделов, занимающихся вопросами защиты информации. Основными направлениями деятельности Центра оценки соответствия и тестирования являются:

◆ сертификация средств защиты информации (в том числе технических);

◆ внедрение, включая выстраивание всех процессов в соответствии с ГОСТ Р 56939;

◆ оценка соответствия по требованиям ОУД в рамках проверки банковских приложений (требования ГОСТ ИСО/МЭК 15408, Профиля защиты Банка России, положений Банка России);

◆ проведение тематических исследований по требованиям ФСБ России;

◆ проведение оценки по требованиям стандарта платформы Цифрового рубля.

* * *

Для отслеживания последних новостей в сфере информационной безопасности подписывайтесь на наш telegram-канал Echelon Eyes:

КАК РАБОТАЮТ ИНСТРУМЕНТЫ СТАТИЧЕСКОГО И ДИНАМИЧЕСКОГО АНАЛИЗА КОДА

ОПЫТ ИСПОЛЬЗОВАНИЯ АК-ВС 3

Роберт АРУСТАМЯН
заместитель директора Центра оценки соответствия и тестирования АО «НПО «Эшелон»

Нелли МАГАКЕЛОВА
руководитель группы Центра оценки соответствия и тестирования АО «НПО «Эшелон»

Илья АНТИПОВ
руководитель группы Центра оценки соответствия и тестирования АО «НПО «Эшелон»

В условиях развития технологий и возрастания угроз в сфере кибербезопасности обеспечение безопасности программного обеспечения становится приоритетной задачей для многих организаций. Одним из эффективных способов выявления потенциальных уязвимостей является использование инструментов статического и динамического анализа кода. Эти методы анализа играют ключевую роль в раннем обнаружении дефектов и уязвимостей, что позволяет значительно повысить безопасность разрабатываемых программных продуктов.

СТАТИСТИКА

АК-ВС 3 активно используется в различных сферах, включая государственные учреждения, коммерческие организации и высшие учебные заведения. Решение успешно интегрировано в процессы безопасности множества организаций, что подтверждается растущей статистикой его внедрения. Университеты применяют АК-ВС 3 для обучения специалистов в области информационной безопасности, а государственные структуры используют его для анализа критически важных программных решений.

С 2024 года по настоящее время 118 организаций, включая государственные учреждения, высшие учебные заведения и испытательные лаборатории, успешно приобрели продукт АК-ВС 3. Из данного числа 110 организаций представляют коммерческий сектор, что свидетельствует о растущем интересе бизнеса к современным технологиям анализа кода. С начала 2025 года около 10–15 организаций воспользовались возможностью протестировать демоверсию АК-ВС 3.

АРХИТЕКТУРА АК-ВС 3:

1. Модуль «Клиент» — IDE «Эшелониум»: осуществляет работу с проектами, предоставляет возможность для просмотра, анализа и разметки срабатываний.

2. Модуль «Сервер»: обеспечивает выполнение статического анализа кода, динамического анализа кода, полносистемного динамического анализа (Код2), фаззинг-тестирования, а также предоставляет веб-интерфейс для удобной работы.

3. Модуль сборки: включает утилиту для подготовки исходных текстов, необходимых для выполнения статического анализа в рамках проведения подконтрольной сборки исследуемого программного обеспечения.

4. Модуль фаззера: менеджер фаззеров, который выполняет фаззинг-тестирование.

5. Модуль «Код2»: отвечает за реализацию полносистемного динамического анализа и подготовку полносистемных трасс для последующего анализа (рис. 1).

ИСПОЛЬЗУЕМЫЕ МЕТОДЫ АНАЛИЗА

Статический анализ исходного кода программы (SAST).

Статический анализ представляет собой процесс анализа исходного кода программы с целью выявления дефектов кода, в частности потенциально опасных конструкций. В соответствии с актуальной нормативной базой данный анализ также следует проводить в отношении компонентов, заимствованных у сторонних разработчиков, при условии доступности их исходного кода.

Первые версии статических анализаторов были схожи с линтерами — утилитами, которые выполняют анализ кода, ищут совпадения с заданными шаблонами и формируют выводы о наличии или отсутствии определённых конструкций в исходном коде. Например, это может касаться жёстко заданных паролей или использования небезопасных функций. Изначально линтеры разрабатывались с целью содействия разработчикам в написании качественного кода и не имели непосредственного отношения к вопросам безопасности.

Цели статического анализа кода программы:

- поиск дефектов кода;

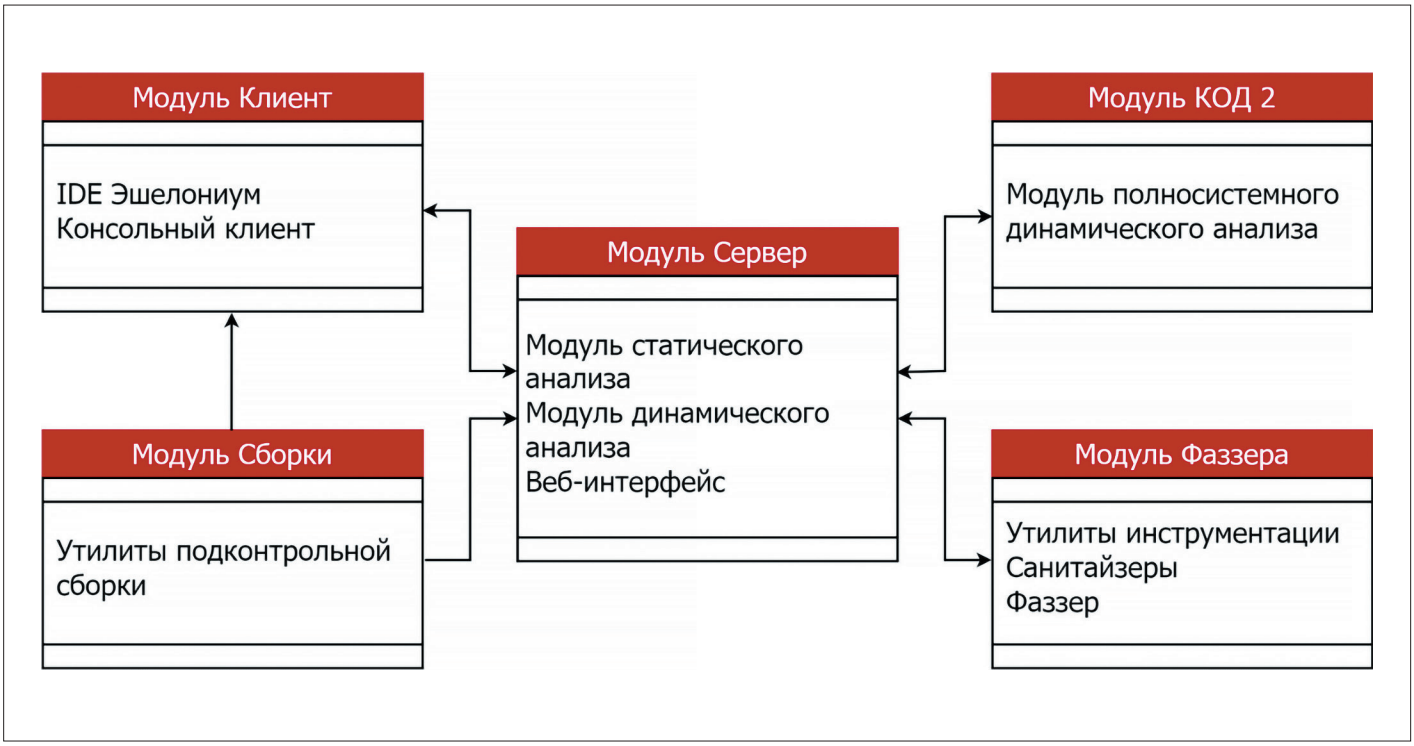


Рисунок 1. Состав и структура АК-ВС 3

- подсчёт метрик;
- рекомендации по оформлению кода;
- внедрение в рамках цикла безопасной разработки (SSDLC);
- обеспечение выполнения нормативных требований заказчика и регуляторов.

Основные применяемые методы статического анализа кода программы:

1. Синтаксический анализ: включает использование базы данных сигнатур, где ошибки выявляются путём поиска заранее известных антипаттернов в исходном коде.

2. Межпроцедурный анализ: позволяет выявлять сложные ошибки, возникающие из-за взаимодействия различных функций и методов. В этом случае анализируется не только отдельная строка кода, но и совокупность всех файлов и функций с учётом контекста и взаимосвязей.

3. Контекстно-чувствительный анализ: сохраняет контекст переходов между функциями и их состояниями. Например, если для входа в определённую функцию необходимо, чтобы переменная имела значение 0, статический анализатор будет отслеживать, что в дальнейшем эта переменная не может принимать иное значение.

Дополнительно могут быть использованы межмодульный анализ, анализ потока управления, анализ потока данных и ряд других методов. Для более детального ознакомления с различными видами статического анализа рекомендуется обратиться к статье, в которой представлен сравнительный анализ работы статических анализаторов в соответствии с требованиями ФСТЭК России, Министерства обороны Российской Федерации, а также зарубежных стандартов, таких как NIST SP 500–268 и SATEC[1].

[1] Марков А. С., Антипов И. С., Арустамян С. С., Магакелова Н. А. Сравнительный анализ статических анализаторов кода. Современная методология // Вопросы кибербезопасности. 2024. № 5 (63). С. 79–88.

В ходе статического анализа кода акцент делается на выявление дефектов, однако уязвимости могут быть обнаружены исключительно в процессе динамического анализа. Значительная часть проблем, связанных с безопасностью программного обеспечения, проявляется лишь во время выполнения программы. Таким образом, динамический анализ играет ключевую роль в обеспечении безопасности программных продуктов, позволяя выявить риски, которые невозможно обнаружить, работая лишь с исходными текстами программы.

Динамический анализ кода (DAST). Динамический анализ кода представляет собой процесс выявления уязвимостей в режиме выполнения (run-time) программного обеспечения. Основные методы динамического анализа включают общую сборку путей выполнения, полносистемный анализ и фаззинг-тестирование.

Общая сборка путей выполнения включает проведение сборки исходных текстов, в которые будут интегрированы специальные датчики. Эти датчики отслеживают прохо-

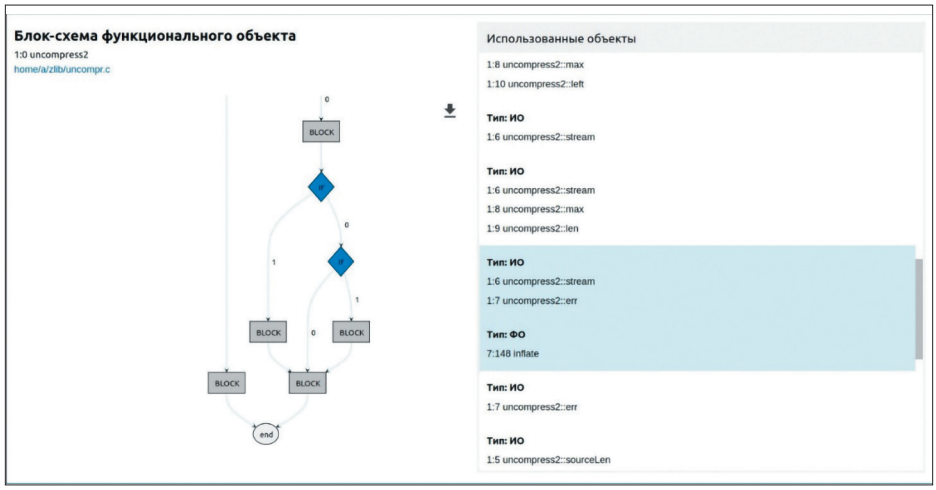


Рисунок 2. Построение блок-схемы для конкретного функционального объекта

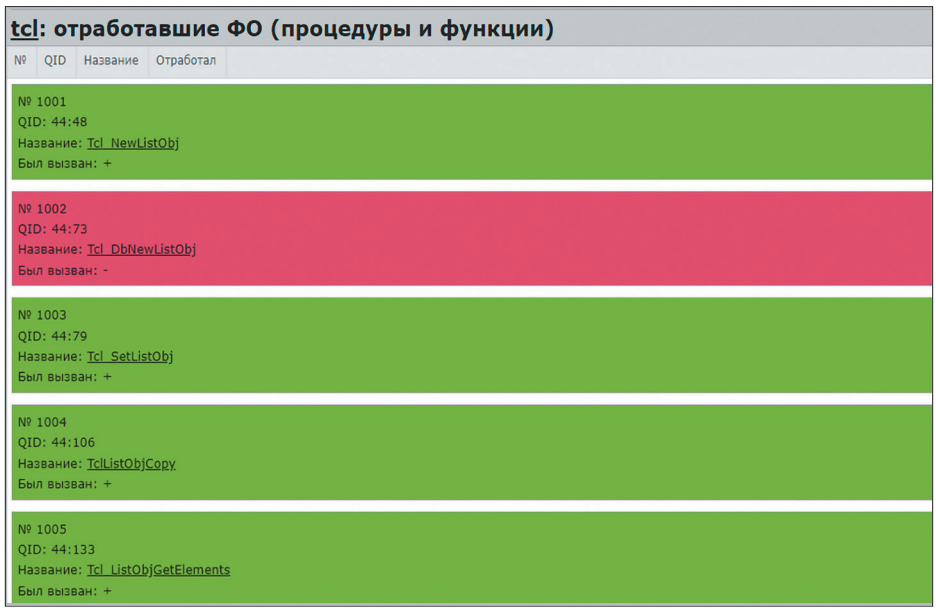


Рисунок 3. Анализ отработавших функциональных объектов, которые были проинструментированы датчиками при динамическом анализе

дящие через них вызовы функций и осуществляют запись логов в файл, на основе которых впоследствии осуществляется анализ и формируется трасса выполнения. Это позволяет определить, какие блоки кода были вызваны, а какие остались неиспользованными, а также проследить за прохождением по веткам выполнения (рис. 2).

Таким образом, этапы динамического анализа выглядят следующим образом:

1. Сбор логов по динамическому выполнению проекта.
2. Обнаружение путей ветвления и исполнения проекта на основе реального выполнения.
3. Определение модулей и функций, через которые проходило выполнение.
4. Выявление мест возникновения ошибок и сбоев (рис. 3).

Полносистемный динамический анализ (Код2). Полносистемный анализ представляет собой анализ с полной трассировкой выполнения программы, что позволяет отслеживать помеченные данные, а также выполнение и пересечение различных элементов. Этот метод основан на использовании QEMU, в рамках которого осуществляется функциональное тестирование, трассировка и непосредственный анализ.

Этапы полносистемного анализа:

- полносистемный анализ трассы выполнения кода;
- запись данных всей системы с отслеживанием каждого элемента;
- поиск утечек данных и критической информации.

Фаззинг-тестирование (FAST). Фаззинг-тестирование представляет собой методику, при которой на вход программы подаются сгенерированные или мутационные данные. В рамках фаззинга с обратной связью (feedback-based fuzzing) осуществляется сбор логов о том, как обработанные данные влияют на систему. Входные данные генерируются с использованием как мутационных, так и словарных (генерационных) методов.

- Мутационный фаззинг: данные могут принимать любые формы, включая изображения, видео, байты и произвольные наборы данных.
- Генерационный фаззинг: этот метод, хотя и имеет частичное совпадение с мутационным, позволяет генерировать входные данные, которые являются валидными для первого фильтра при взаимодействии с интерфейсом системы.

Фаззинг-тестирование в АК-ВС 3 реализовано на основе AFL++ с учётом оригинальной технологии разработки.

ВЫВОД

Инструменты статического и динамического анализа кода играют важнейшую роль в обеспечении безопасности программного обеспечения, позволяя выявить уязвимости на разных этапах разработки. Статический анализ помогает обнаружить дефекты на уровне исходного кода, прежде чем они могут быть использованы злоумышленниками. Динамический анализ, в свою очередь, позволяет выявить проблемы, которые могут проявиться лишь во время выполнения программы, такие как утечки памяти или переполнения буфера. Оба метода являются неотъемлемой частью процесса безопасной разработки и позволяют предотвратить угрозы безопасности на ранних стадиях.

СЕРТИФИКАЦИЯ ПО

НОВЫЕ ВЫЗОВЫ И ВОЗМОЖНОСТИ

Михаил ГЕЛЬВИХ
руководитель отдела технического сопровождения ООО ПВС

В эпоху цифровизации и растущих киберугроз сертификация программного обеспечения становится критически важной для бизнеса. Особую актуальность этот вопрос приобрёл после введения новых стандартов (таких как ГОСТ Р 71207–2024 и ГОСТ Р 56939–2024), направленных на обеспечение процесса разработки безопасного программного обеспечения (РБПО).

В то время как ГОСТ Р 56939–2024 охватывает более широкий спектр процессов и мероприятий по обеспечению безопасности на всех этапах жизненного цикла ПО, ГОСТ Р 71207–2024 фокусируется именно на инструментальных методах анализа, позволяющих повысить качество и безопасность создаваемых программных продуктов.

СТАТИЧЕСКИЙ АНАЛИЗ: ЧТО ЭТО?

SAST (Static Application Security Testing) — это методология тестирования, при которой код программы анализируется на наличие потенциальных уязвимостей без его фактического выполнения. Согласно стандарту, современные анализаторы должны обнаруживать наиболее распространённые критические ошибки, возникающие при разработке ПО:

- ошибки непроверенного использования чувствительных данных;
- ошибки некорректного использования системных процедур и интерфейсов;
- ошибки целочисленного переполнения и переполнения буфера;
- проблемы с управлением динамической памятью;
- ошибки в многопоточных приложениях.

ПРОБЛЕМЫ, РЕШАЕМЫЕ СТАТИЧЕСКИМ АНАЛИЗОМ

Соответствие требованиям ГОСТ Р 56939–2024 в рамках процессов разработки безопасного ПО подразумевает использование специализированных инструментов для систематического выявления и устранения уязвимостей. Как раз здесь на помощь приходят статические анализаторы, такие как PVS-Studio, которые позволяют решать ряд ключевых задач:

1. Раннее выявление уязвимостей. Статический анализ кода позволяет обнаруживать потенциальные уязвимости ещё на этапе разработки, что значительно снижает риск их использования злоумышленниками.

2. Экономия на исправлении ошибок. Исследования подтверждают: чем раньше будет обнаружена уязвимость, тем меньше будут затраты на её исправление. Это позволяет компаниям не только экономить деньги, но и сосредоточиться на разработке новых функций, а не на исправлении старых.

3. Защита корпоративной репутации. Репутационные потери от инцидентов безопасности часто превышают прямые финансовые убытки. Регулярное использование SAST-инструментов помогает минимизировать риски и защищает вашу репутацию.

4. Автоматизация и интеграция в CI/CD. Современные статические анализаторы легко интегрируются в процессы CI/CD. Это позволяет автоматически проверять код на наличие потенциальных уязвимостей после каждого изменения, что ускоряет процесс разработки и улучшает её качество.

5. Увеличение доверия со стороны клиентов. Наличие сертификации по ГОСТ Р 56939–2024 и использование инструментов статического анализа, таких как PVS-Studio, могут повысить доверие со стороны клиентов к разработчикам программного обеспечения или даже являться обязательным требованием, если речь заходит о проектах в области КИИ.

ПОЧЕМУ ЭТО ВАЖНО ДЛЯ БИЗНЕСА

Таким образом, внедрение статического анализа в процессы разработки и сертификация ПО по новым стандартам становятся необходимыми для компаний, стремящихся к повышению уровня своей информационной безопасности. Инструменты, такие как PVS-Studio, соответствующие требованиям ГОСТ Р 71207–2024, помогают разработчикам создавать более безопасные и надёжные программные решения, сокращая затраты на исправление ошибок и повышая эффективность разработки.

* * *

Предложите вашей команде разработки возможность бесплатно попробовать PVS-Studio в течение месяца, используя промокод BIS, и на практике оценить возможности инструмента по выявлению дефектов безопасности в коде программ, написанных на языках C, C++, C#, Java.

ПИЛОТ, ЧТОБЫ SAST «ЛЕТЕЛ» В ПЯТЬ РАЗ БЫСТРЕЕ

ПРИМЕНЕНИЕ ТЕХНОЛОГИЙ ИИ ДЛЯ ЗАДАЧ СТАТИЧЕСКОГО АНАЛИЗА КОДА

Антон БАШАРИН
старший управляющий директор AppSec Solutions

Бизнес всё чаще начинает уделять внимание информационной безопасности, и компании неизбежно сталкиваются с ситуаций перегрузки ИБ-инженеров. Топ-менеджмент встает перед выбором: либо кратно увеличивать штат, либо пренебрегать безопасностью. И то и другое — дорого, поскольку найм редких специалистов затратен, а утечки данных могут привести к серьезным убыткам.

Сегодня бизнес, запустивший внедрение инструментов безопасной разработки и практику РБПО, ожидает сокращение рисков и времени на разработку в несколько раз. Однако быстро сталкивается с объективными трудностями. Внедрение SAST-инструментов не ускоряет разработку, а наоборот замедляет её, поскольку 80% срабатываний — ложные. Это показало исследование, которое провела команда AppSec Solutions на данных полутора десятков компаний-клиентов.

Статистика по автоматизации обработки результатов сканирований ПО в России улучшилась: в 2023 году такому разбору поддавалось лишь 35% срабатываний, в 2024 — уже 64%. Это значительно облегчает и ускоряет работу ИБ-команд. Без таких инструментов инженерам приходится тратить недели на разбор срабатываний, большинство из которых ложные.

Команда из 10 000 разработчиков при интенсивной нагрузке может обрабатывать до 1000 сканов по 1000 записей в сутки. В сумме мы получаем ежедневно до 1 млн записей, которые необходимо разобрать и определить, в каких случаях требуется личное участие квалифицированного ИБ-инженера, а в каких ошибки типичны или вовсе не существуют. Для команды из 10 специалистов по кибербезопасности это работа приблизительно на 200 дней. Впечатляющие цифры, не так ли? Если учесть, что 80% срабатываний ложные, можно считать, что это время потрачено напрасно, при том, что каждый ИБ-инженер на вес золота.

Как сэкономить это время? Одно из решений — использовать искусственный интеллект. Мы разработали модуль AppSec.CoPilot. Это DevSecOps-помощник на основе ИИ, который в автоматическом режиме анализирует и сортирует уязвимости, определяя критические с точностью 96%. Он устраняет ложные срабатывания и сортирует уязвимости по критичности, позволяя команде сосредоточиться на самых серьезных недостатках кода и освобождая время инженеров для более важных задач.

Исследования команды AppSec Solutions на данных более 10 компаний показали, что применение ИИ-помощника сокращает время работы на 85%. Если в 2023 году обработка одной уязвимости занимала 26 дней, то сейчас — всего 1–2 дня, что позволяет сократить Time-to-Market более чем втрое.

Аналогов AppSec.CoPilot в России пока нет, поскольку мы обучали его конкретно под сканирование и приоритизацию уязвимостей. В отличие от крупных универсальных нейросетей, которые обладают широкой специализацией, наш инструмент создан и обучен именно для обеспечения информационной безопасности. Он легко встраивается в любой SAST и адаптируется на стороне заказчика. Модель защищена от лишней информации и обучаема на коде клиента, что позволяет эффективно внедрять ее в работу.

AppSec Solutions запустила AppSec.CoPilot осенью 2024 года. Изначально инструмент был «заточен» под финтех и защиту коммерческой тайны, но к настоящему моменту мы адаптировали его для удобной интеграции в работу промышленных компаний, будь то добывающая или обрабатывающая отрасли, где активы и секреты значительно отличаются от финансового сектора. Это особенно важно при решении проблем параллельного импорта и работы софта по управлению иностранным промышленным оборудованием, где каждое обновление ПО может привести к остановке процесса без возможности «восстановиться» из бэкапа.

До конца 2025 года мы планируем усовершенствовать модуль, добавив автоматическое закрытие уязвимостей, обучая его на исходном коде. Так AppSec.CoPilot сможет облегчать жизнь и ИБ-инженерам, и разработчикам, которым назначаются дефекты и тикеты по устранению подтвержденных уязвимостей.