Тестирование программно-аппаратных решений в рамках SSDLC

Корнеева Светлана
Руководитель группы тестирования ЗНИ

Бауткин Антон
Руководитель отдела обеспечения безопасной разработки и автоматизации тестирования

АО "Аладдин Р.Д.

О выступающих

Корнеева Светлана

Руководитель группы тестирования ЗНИ

- ✓ 7 лет в тестировании

Бауткин Антон

Руководитель отдела обеспечения безопасной разработки и автоматизации тестирования

- ✓ 11 лет в безопасности

О компании

Компания «Аладдин Р.Д.» – ведущий российский разработчик и поставщик средств аутентификации, продуктов и решений для обеспечения информационной безопасности и защиты конфиденциальных данных.

- ✓ №1 на рынке аутентификации
- ✓ 28 лет успешной работы в сфере ИБ
- ✓ Разработчик национальных стандартов по идентификации и аутентификации (ГОСТ Р 58833-2020)
- ✓ Один из ведущих экспертов ФСТЭК России
- ✓ Уже 2 года проводит работы в рамках центра исследования безопасности ядра Linux

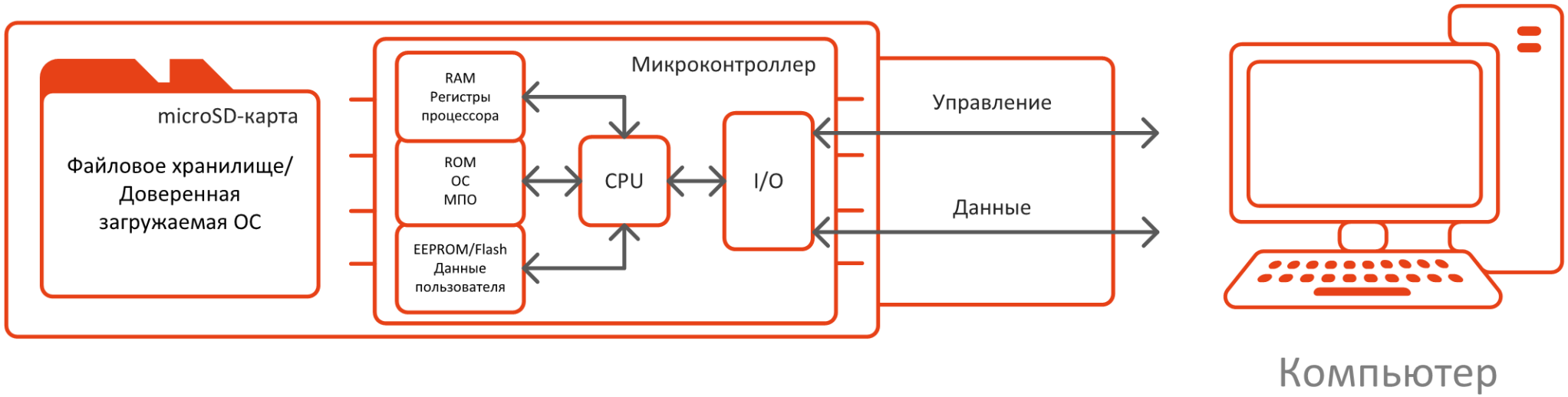
Жизненный цикл разработки ПО

- ✓ Жизненный цикл программного обеспечения (SDLC - Software Development Life Cycle) — концепция создания информационных систем, включающая их планирование, разработку, тестирование и развертку информационных систем. Применяется к аппаратным, программным или комбинированным информационным системам

Основные понятия

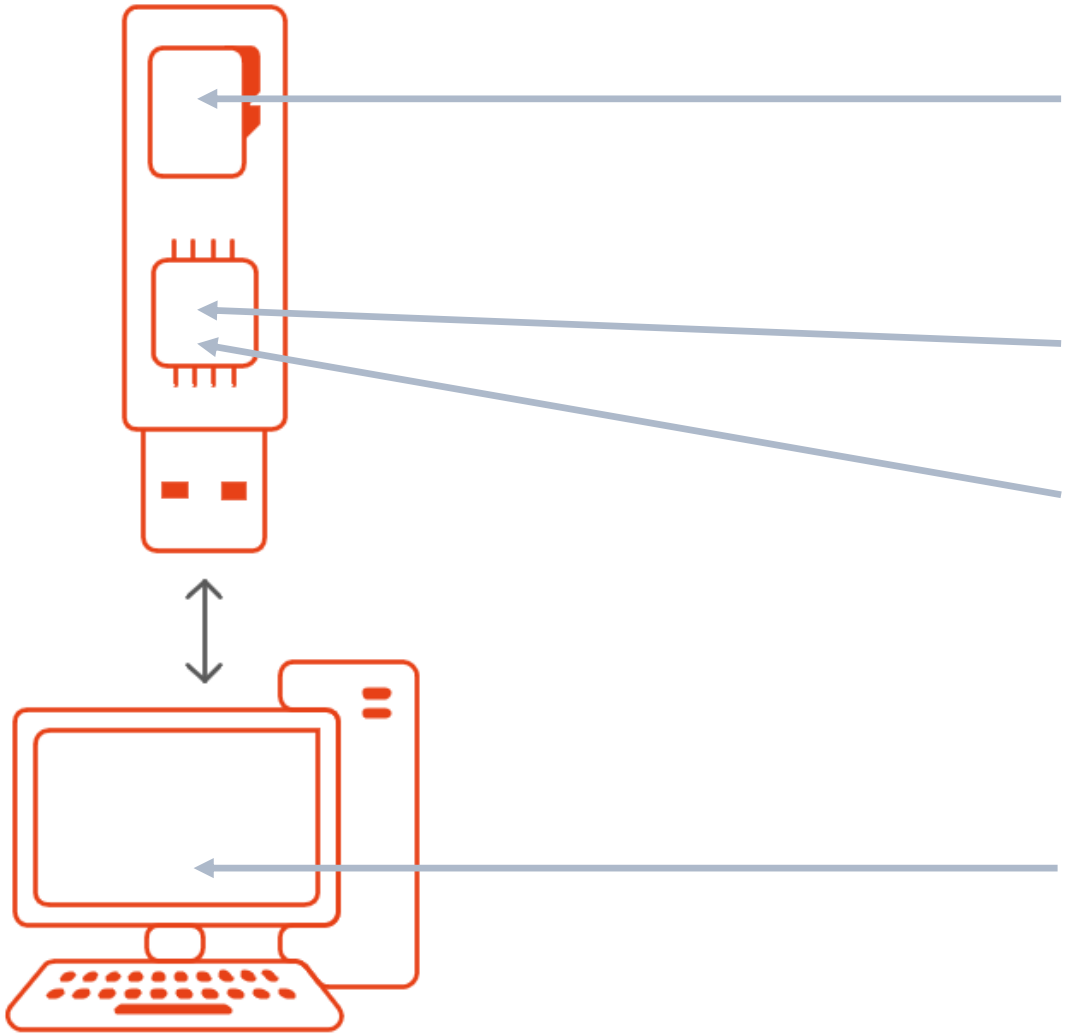
USB-токен – это смарт-карточное устройство, позволяющее пользователю сохранить ключевой контейнер или бинарные данные, используемые в качестве параметра доступа

Устройство USB-токена:



Как мы планируем тестирование для USB-токенов

Объекты тестирования



Определение тестов для каждого объекта тестирования

✓ SD-карта
- Температурные испытания
- Нагрузочное тестирование
- Тестирование стабильности и надёжности
- Объёмное тестирование

✓ Микроконтроллер
- Температурные испытания
- Совместимость с МПО

✓ Микропрограммное обеспечение
- Юнит-тесты
- Нагрузочное тестирование
- Стресс-тестирование
- Тестирование стабильности и надёжности
- Объёмное тестирование

✓ Прикладное ПО (программы управления, драйверы, библиотеки и др.)
- Различные виды функционального и нефункционального тестирования

Тестирование микроконтроллера

✓ Тестирование образцов микроконтроллеров на соответствие требуемым техническим характеристикам в специальной лаборатории

- Безопасность
- Производительность
- Стабильность работы в различных условиях (резкие перепады температуры, влажность, вибрации)

✓ Совместимость с МПО

- Запись МПО, работоспособность МПО (в рамках тестов для МПО)

Тестирование SD-карты

✓ Нагрузочное тестирование (тестирование на ресурс, на износ)

- Исследование максимального количества циклов перезаписи одного сектора памяти до выхода его из строя

✓ Тестирование стабильности и надёжности

- Исследование сохранения работоспособности SD-карты в условиях постоянной перезаписи данных в течение длительного времени (до выхода из строя)

✓ Объёмное тестирование

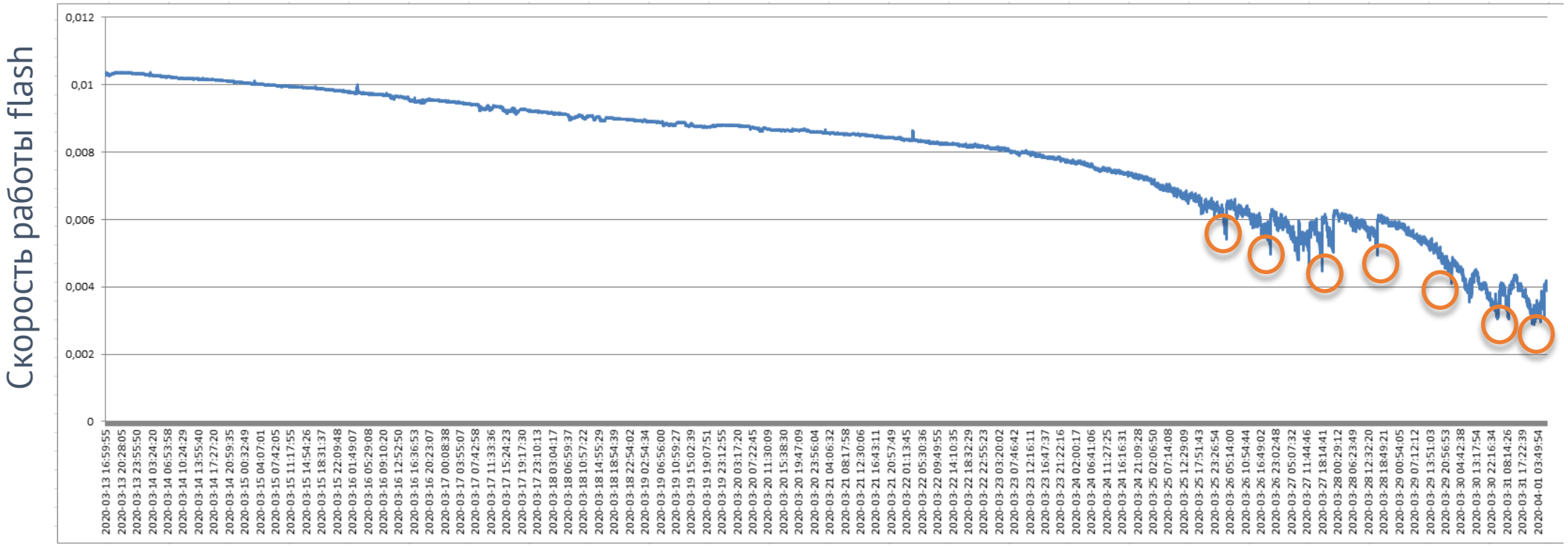
- Замеры скорости работы SD-карты при записи/чтении файлов различного размера

✓ Температурные испытания

- Исследование влияния на работоспособность резких температурных перепадов

Пример выхода из строя flash памяти

- *выведенный из строя сектор*



Продолжительность исследования 4 месяца

Тестирование МПО. Прочие виды тестирования

✓ Юнит-тестирование

- Тестирование на основе требований к продукту и функциональных спецификаций разработчиков с описанием реализованных функций и APDU-команд

✓ Нагрузочное тестирование

- Замеры скорости выполнения операций на токене (создание, изменение, удаление объектов разного размера)

✓ Тестирование стабильности и надёжности

- Выполнение на токене операций в течение длительного времени до выхода из строя

✓ Объёмное тестирование

- Исследование, какое максимальное количество объектов (ключевых контейнеров) можно создать на токене

✓ Стресс-тестирование

- Выполнение на токене операций, в ходе которых происходит прерывание электропитания (имитируем извлечение токена из USB-порта, не дожидаясь завершения операции)

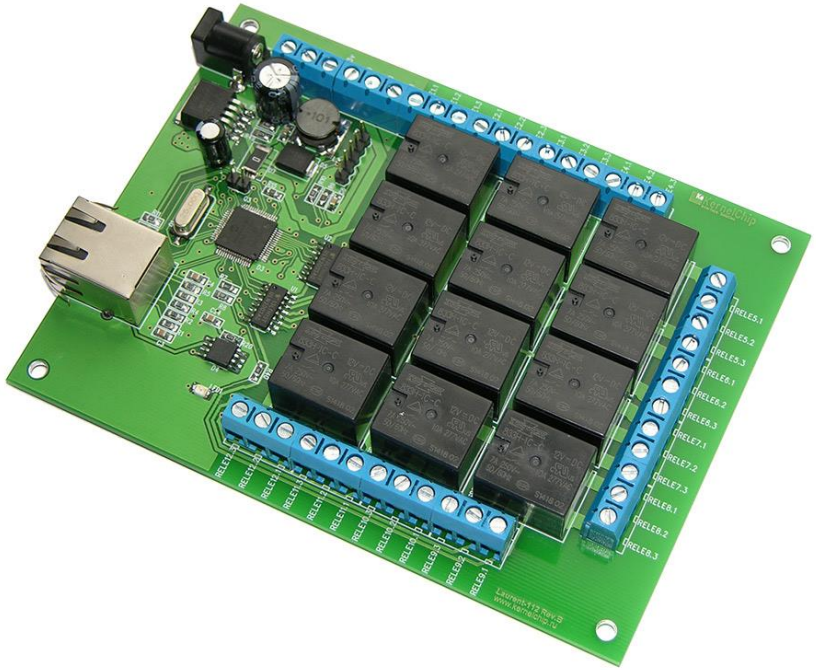
Тестирование отказоустойчивости

Тестирование отказоустойчивости на примере специализированного модуля сетевого реле Laurent-112

✓ Тестирование надёжности электронных компонентов
- Циклическая подача питания на токен

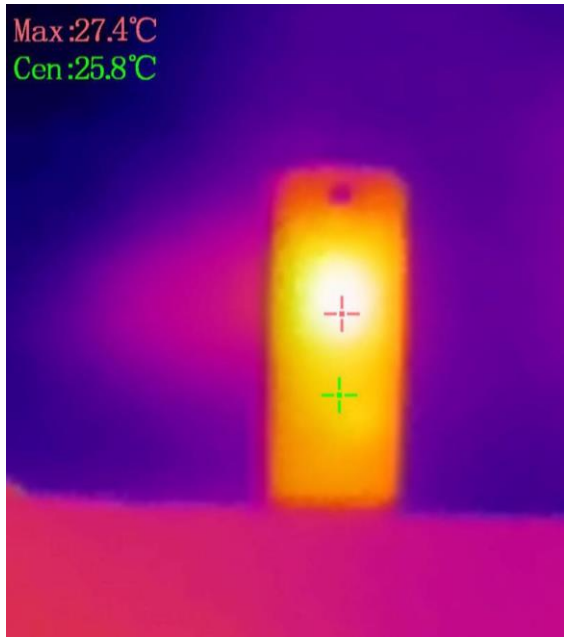
✓ Тестирование работоспособности ОС в токене
- Отключения питания при критических операциях

✓ Тестирование работы приложений в токене
- Отключения питания при выполнении вычислений



Зачем тестировщику тепловизор?

Для проверки норм климатических условий эксплуатации, а именно – рабочей температуры и температуры хранения носителя – проводятся испытания с замером t°C



Тестирование прикладного ПО

Для взаимодействия с токеном необходимо прикладное ПО – программы управления, библиотеки, драйверы

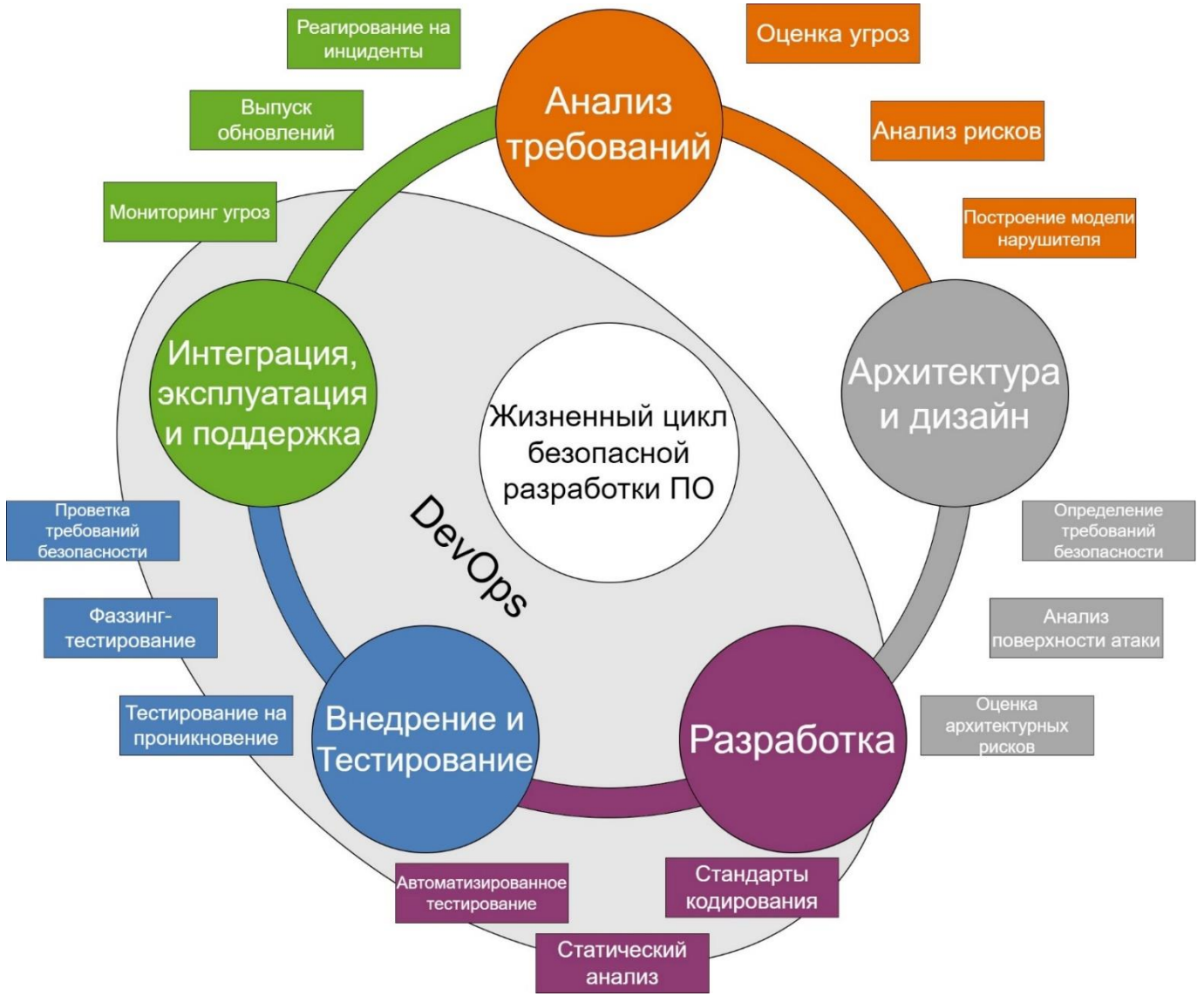
- ✓ Функциональное тестирование:
  - тестирование на основе требований и сценариев использования
  - интеграционное, тестирование взаимодействия прикладного ПО и МПО
  - и др.
- ✓ Нефункциональное тестирование:
  - тестирование установки
  - конфигурационное тестирование
  - тестирование безопасности – статистический, динамический анализы, фаззинг
  - тестирование удобства использования

Критерии завершения тестирования USB-токена

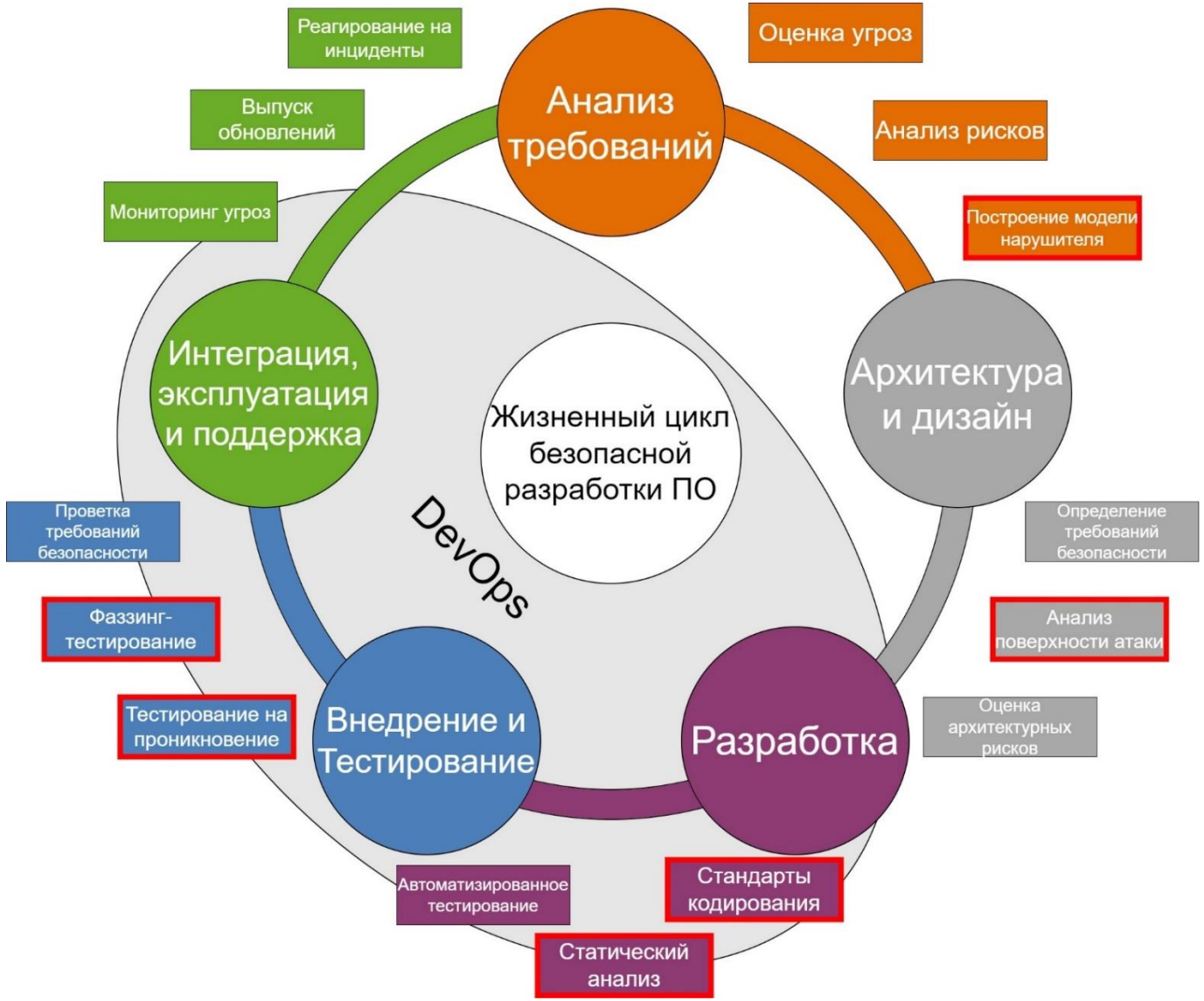
- ✓ Продукт полностью соответствует требованиям
- ✓ Обеспечена совместимость с прикладным ПО
- ✓ Проведены все запланированные тесты
- ✓ Отсутствуют блокирующие и критические ошибки

Безопасная разработка ПО (SSDLC)

Безопасная разработка ПО (SSDLC)



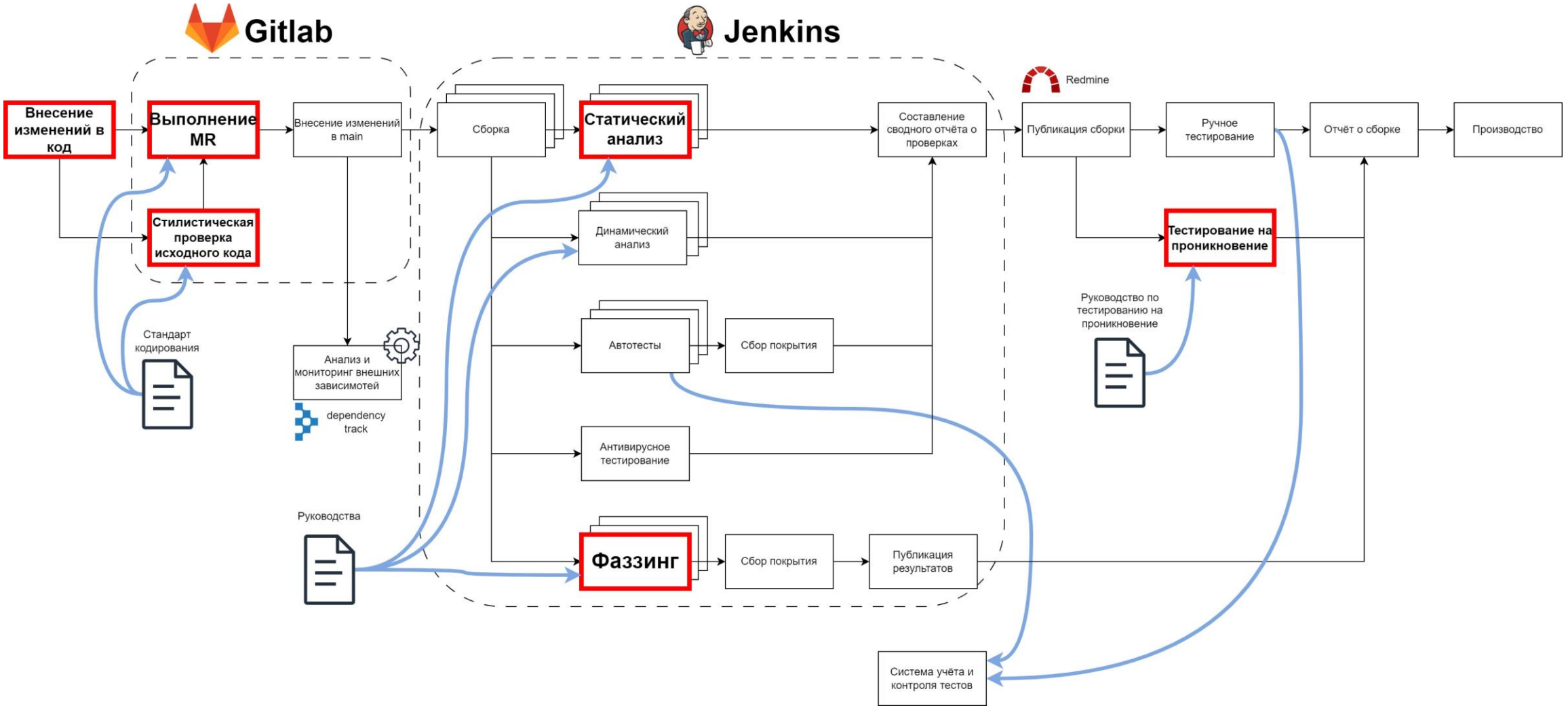
Безопасная разработка ПО (SSDLC)



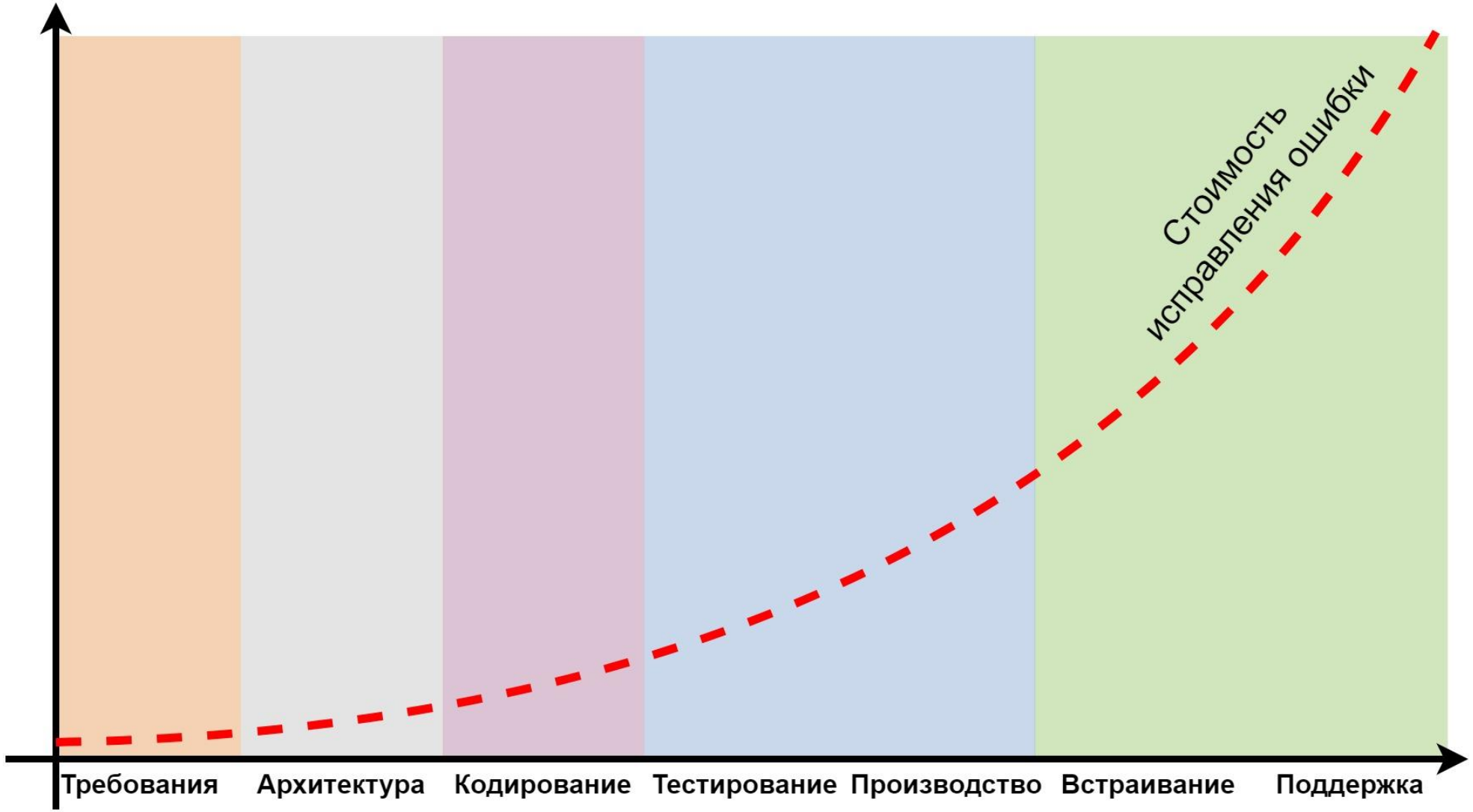
Темы презентации:

- Построение модели нарушителя
- Анализ поверхности атаки
- Статический анализ
- Стандарты кодирования
- Фаззинг-тестирование
- Тестирование на проникновение

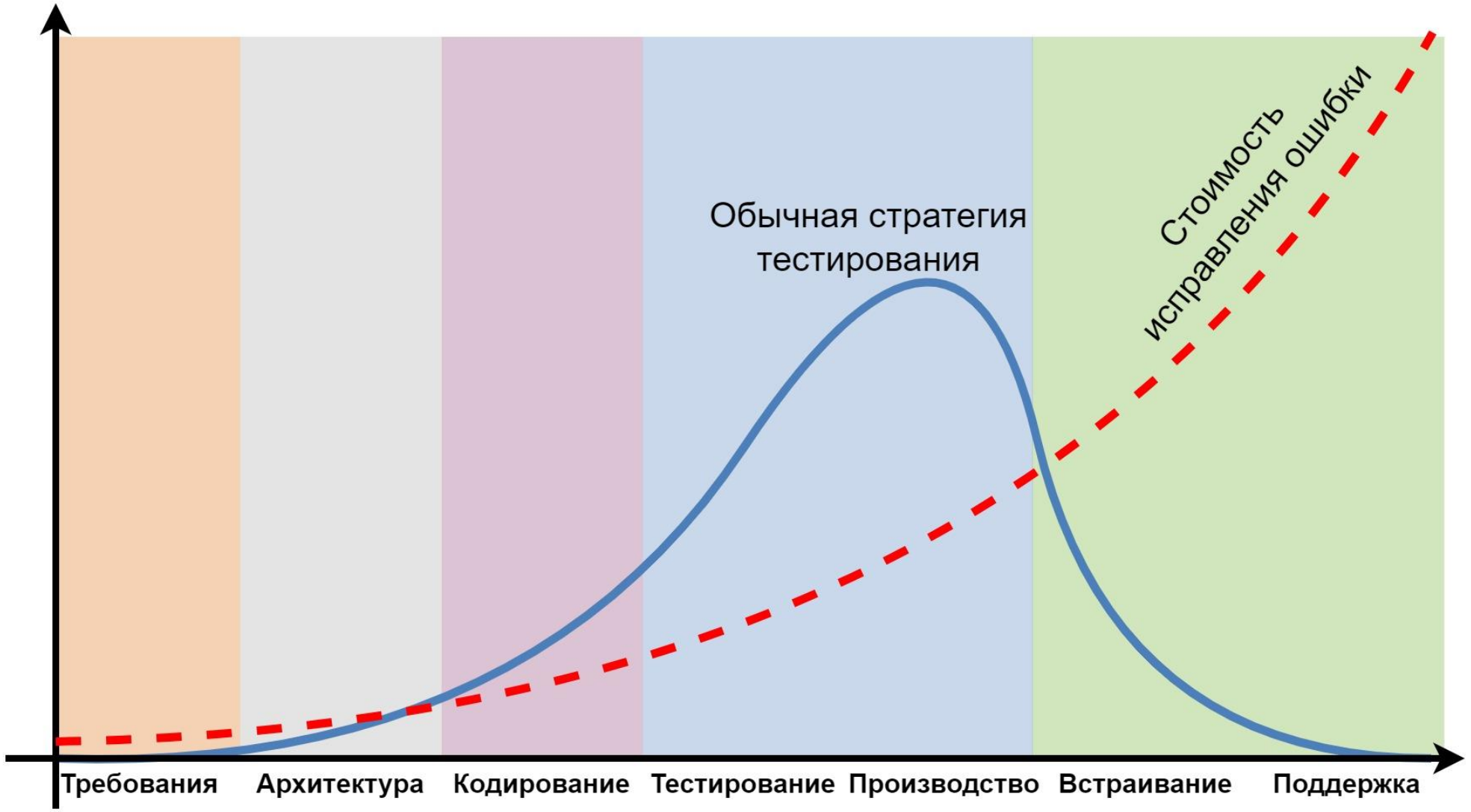
Обобщённый процесс



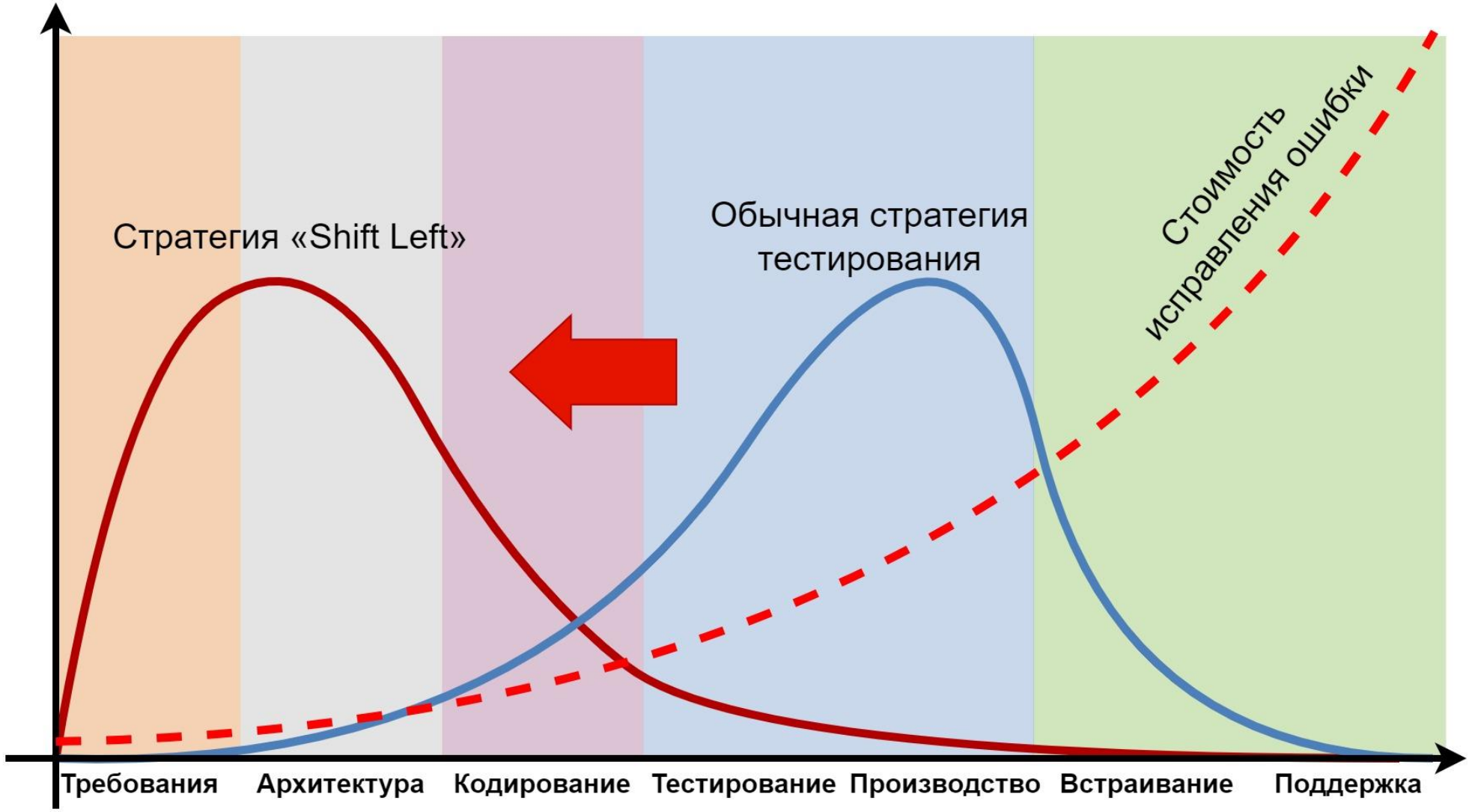
Стратегия «Shift Left»



Стратегия «Shift Left»

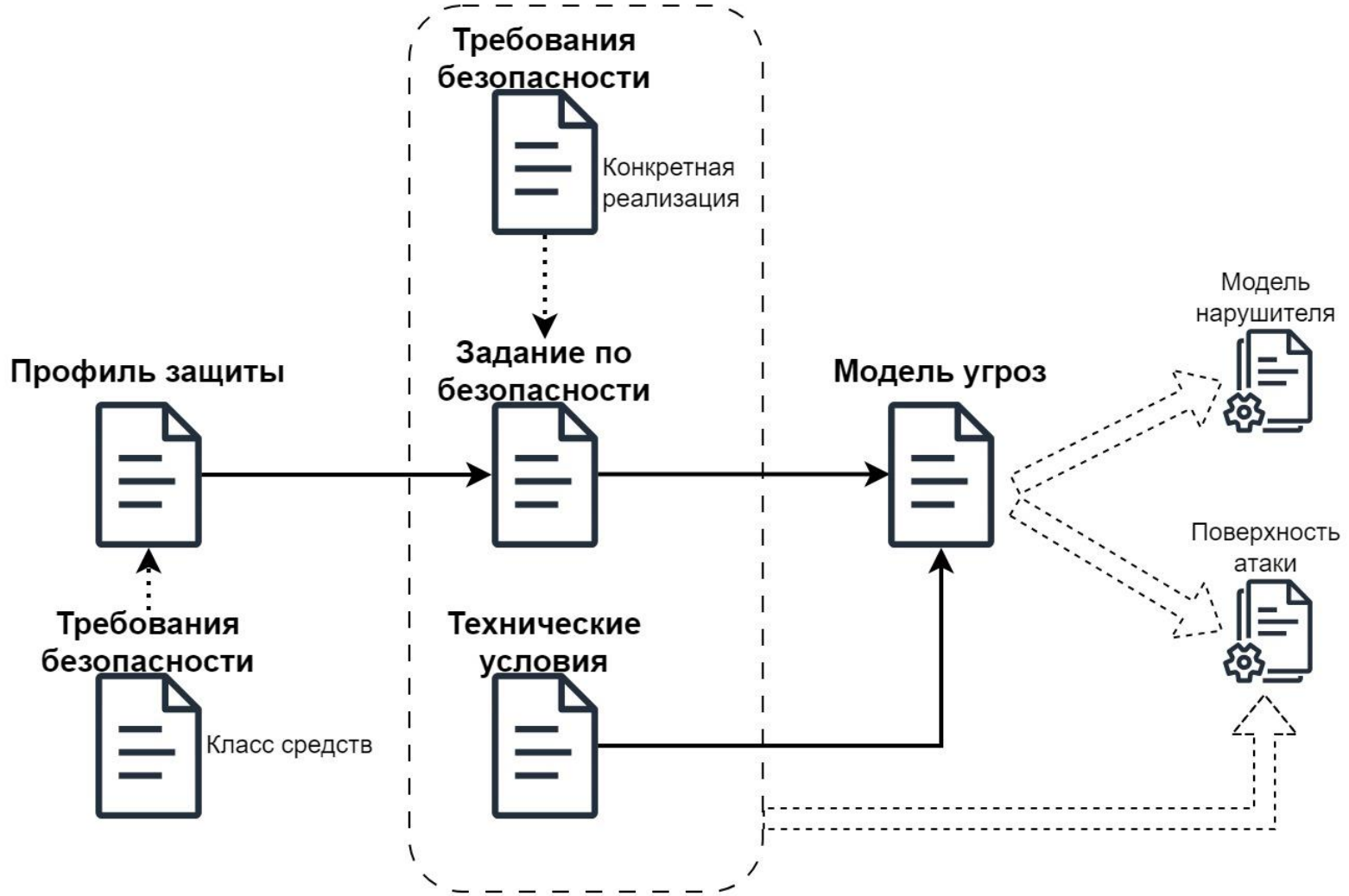


Стратегия «Shift Left»



Модель нарушителя и поверхность атаки

Модель нарушителя и поверхность атаки



Модель нарушителя и поверхность атаки

Ресурсы для разработки документов

Задание по безопасности:

- ✓ ГОСТ Р 57628-2017

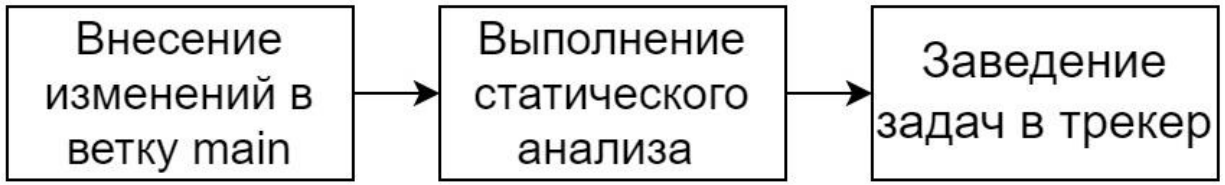
Технические условия:

- ✓ ГОСТ 2.114-2016
- ✓ ПНСТ 818-2023
- ✓ ПНСТ 819-2023
- ✓ *Конструктивная безопасность (Secure by Design)*

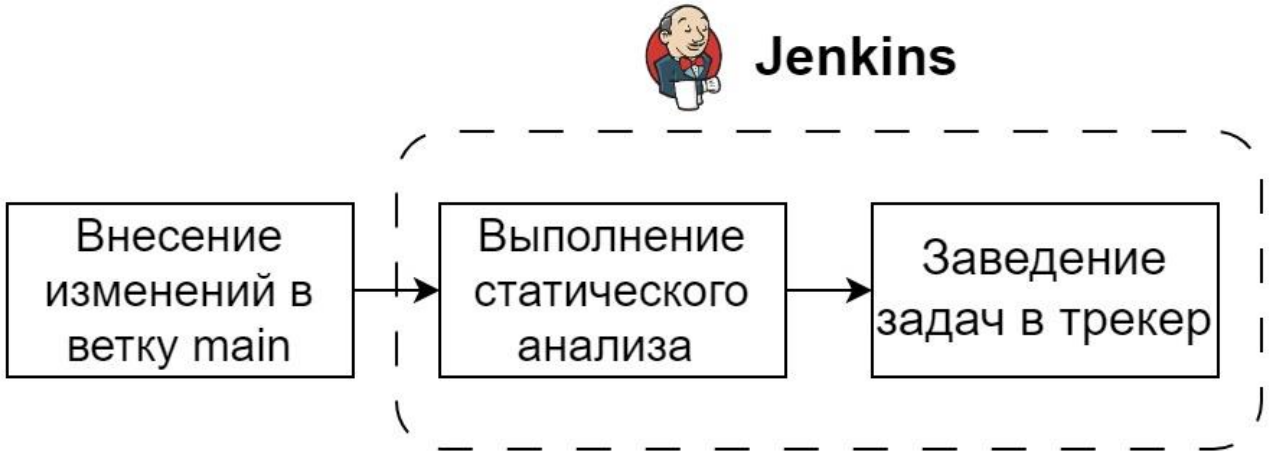
Модель угроз:

- ✓ ГОСТ Р 56939-2016
- ✓ Методика оценки угроз безопасности информации ФСТЭК России
- ✓ БДУ
- ✓ MITRE ATT&CK

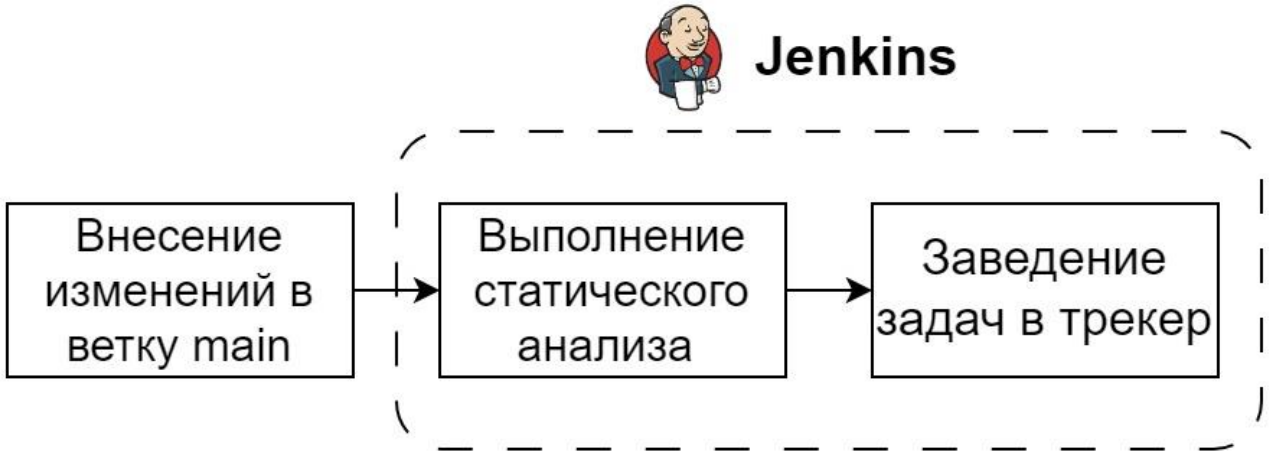
Статический анализ



Статический анализ

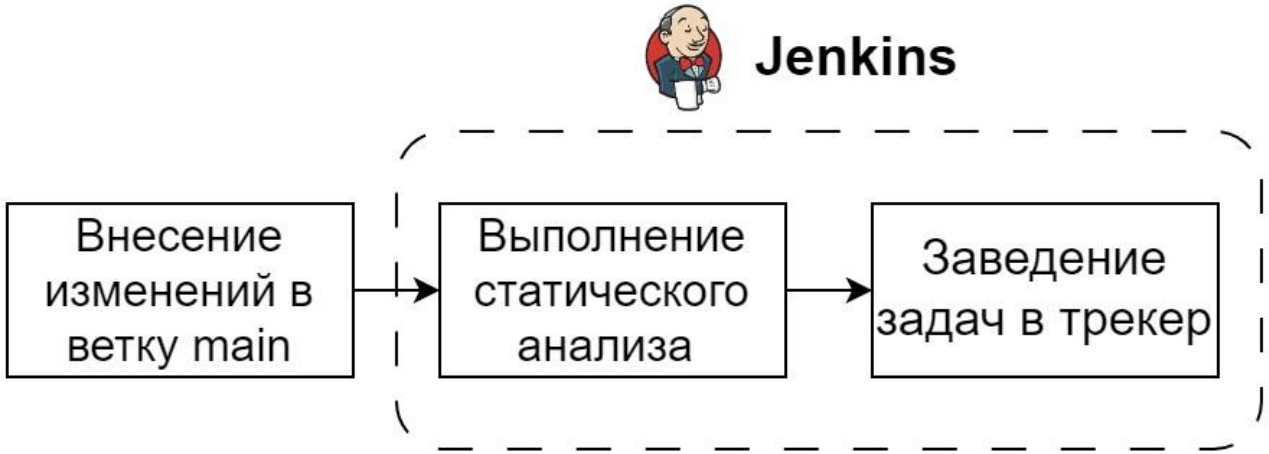


Статический анализ

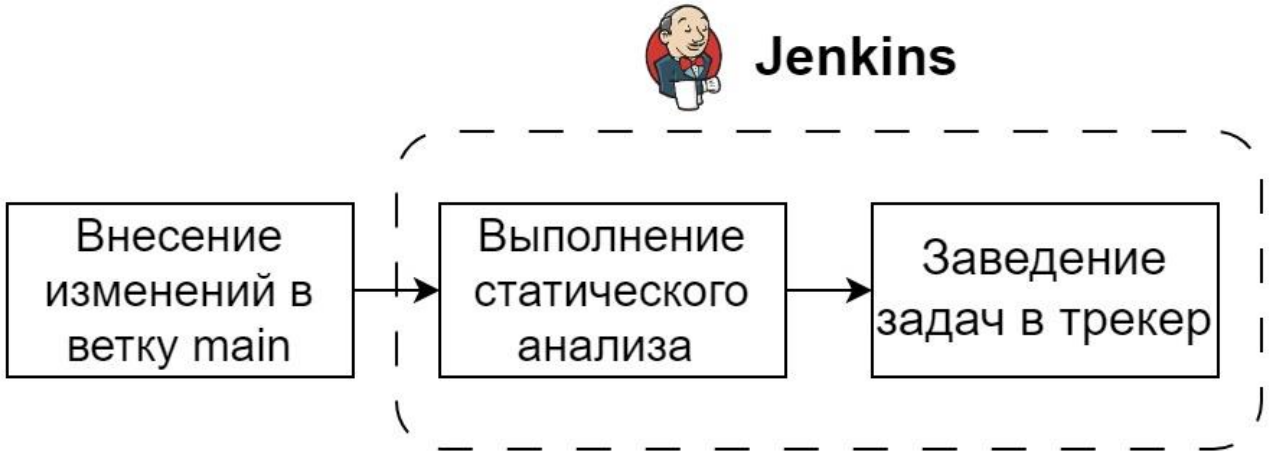


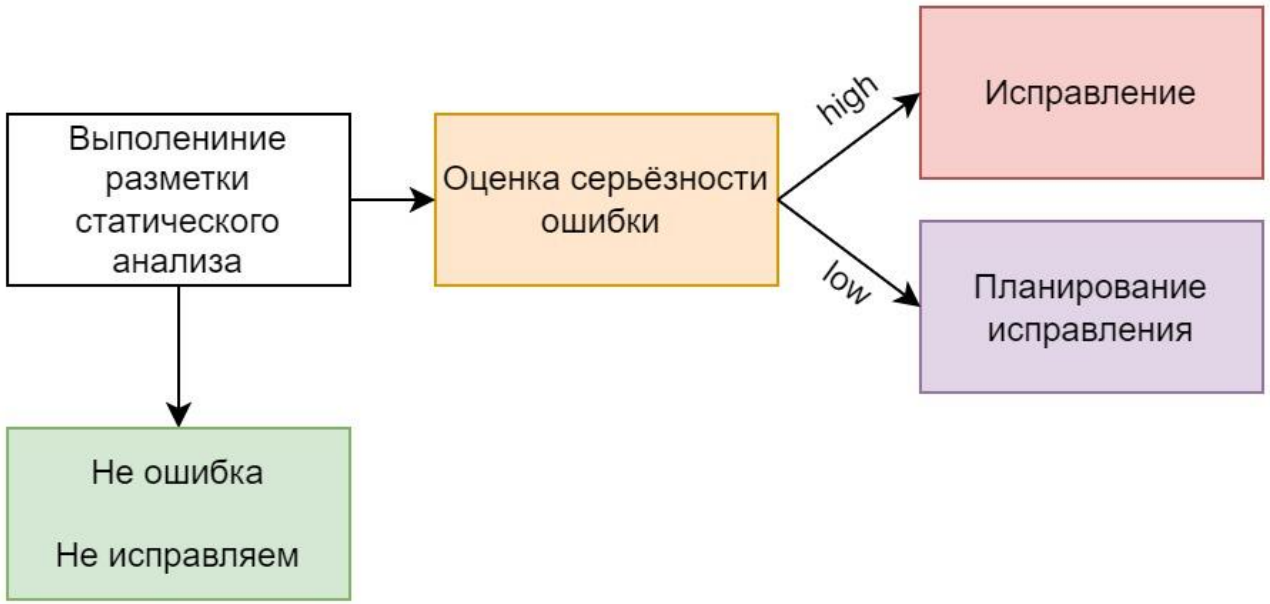
Выполениние разметки статического анализа

Статический анализ

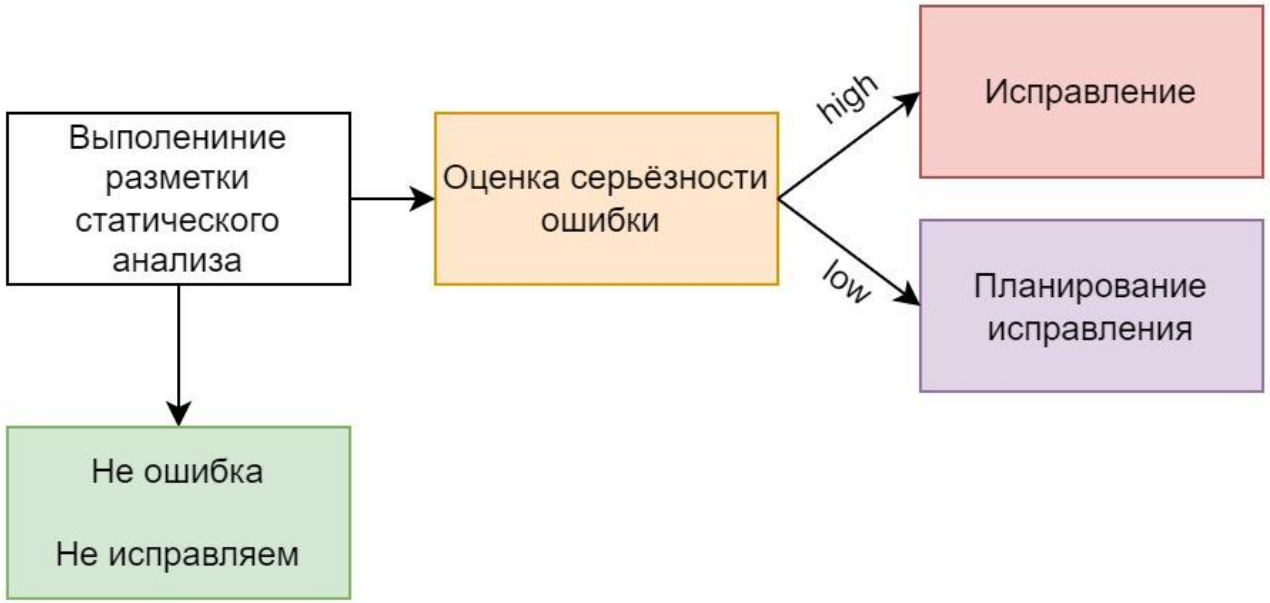


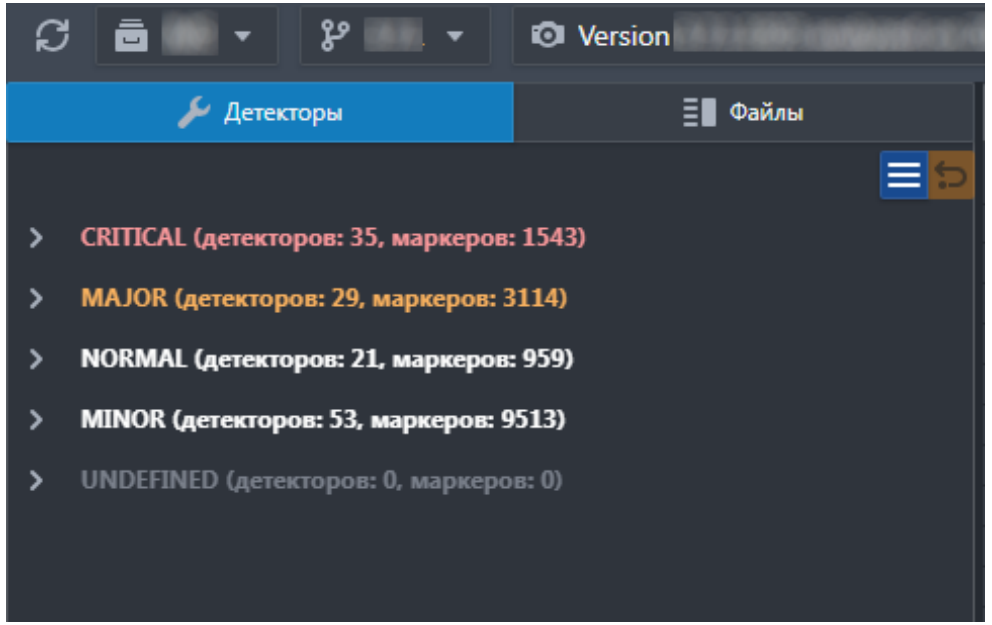
Статический анализ





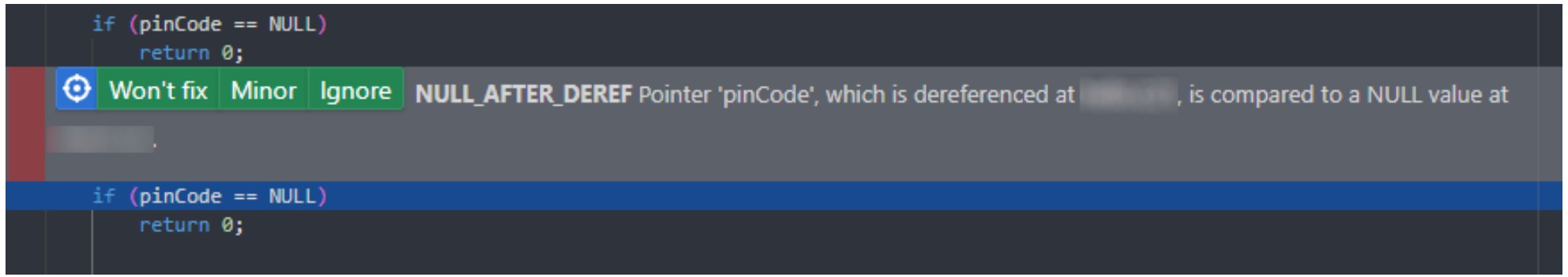
Статический анализ





Статический анализ МПО

- ✓ Условия в критичном коде ДОЛЖНЫ проверяться более одного раза



Статический анализ

Svace

C#

Java

Go

C/C++

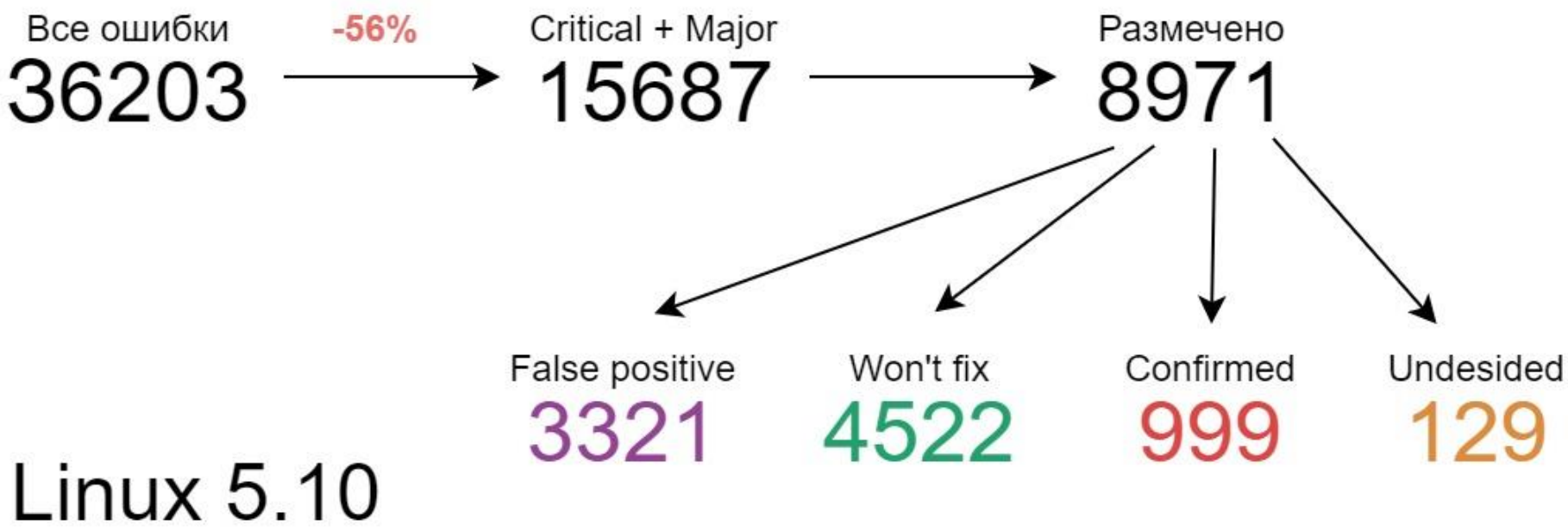
sonarqube

JavaScript

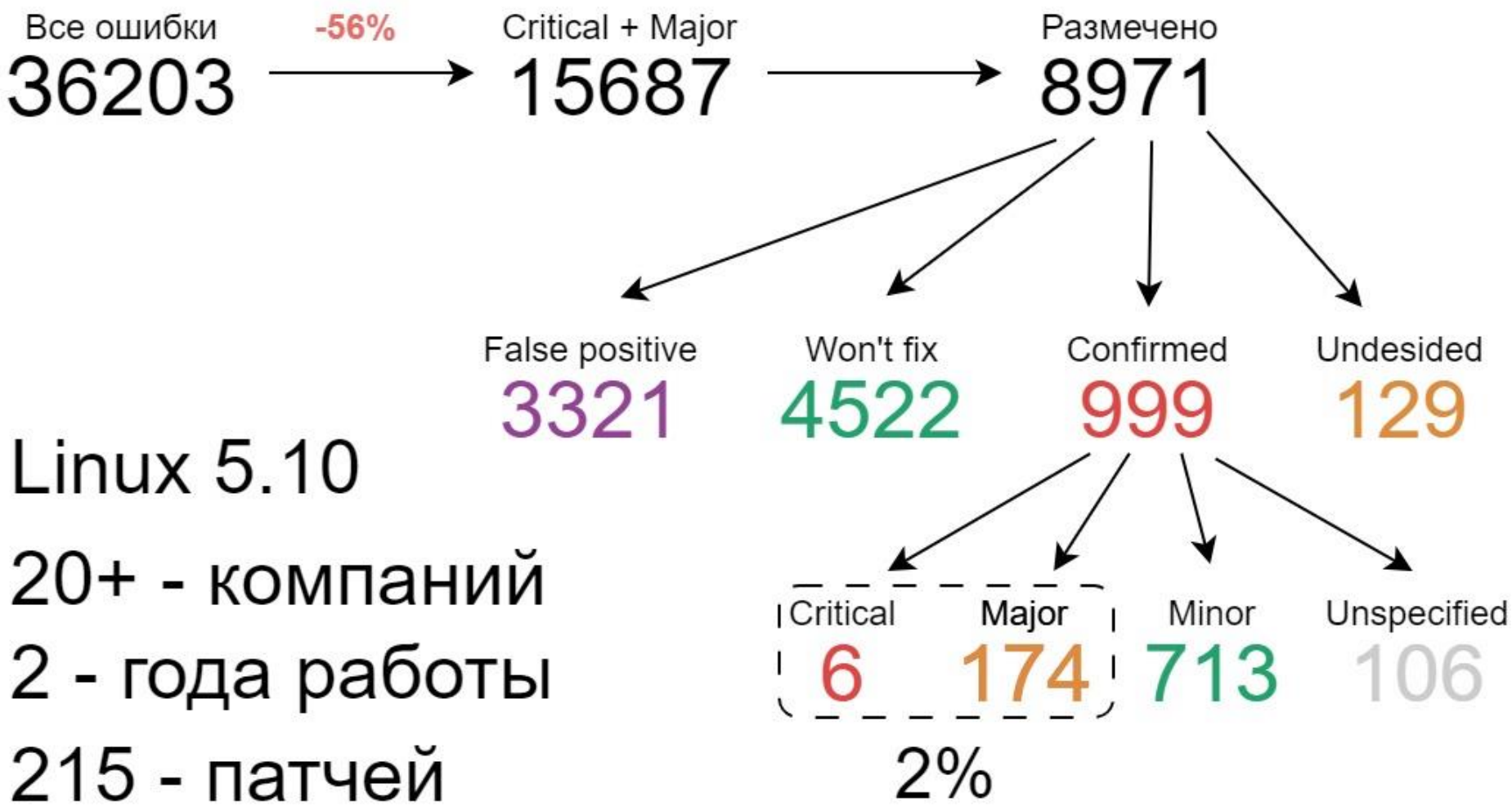
Результаты статического анализа

Linux 5.10

Результаты статического анализа



Результаты статического анализа



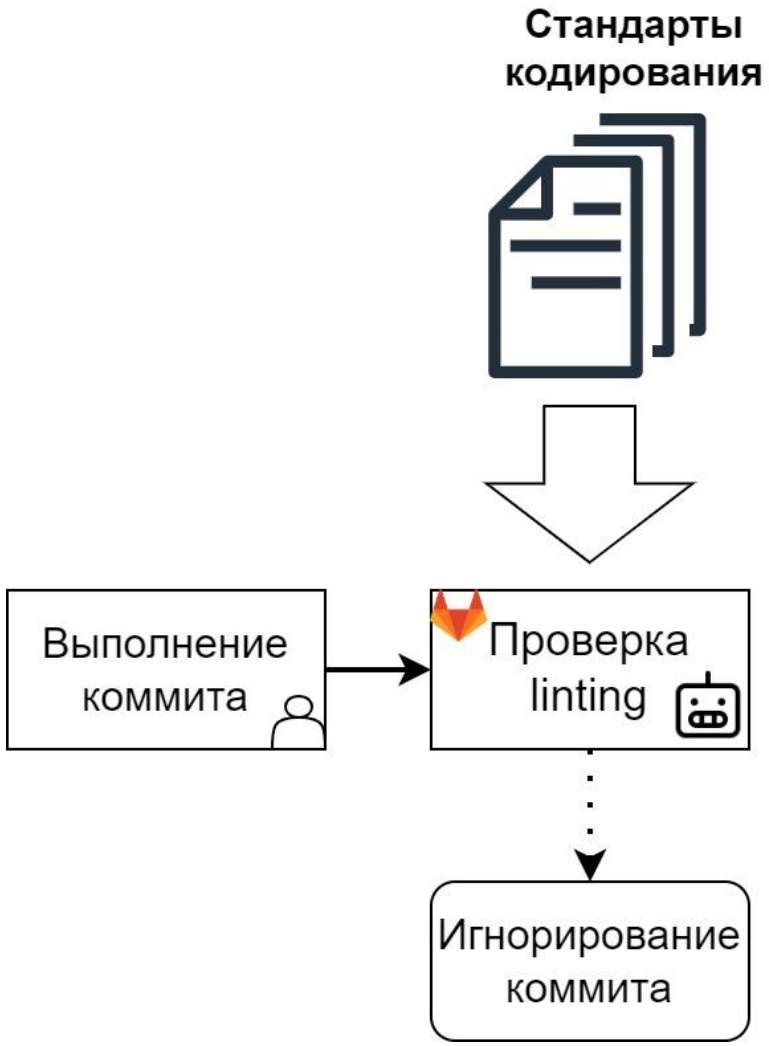
Linux 5.10

20+ - компаний

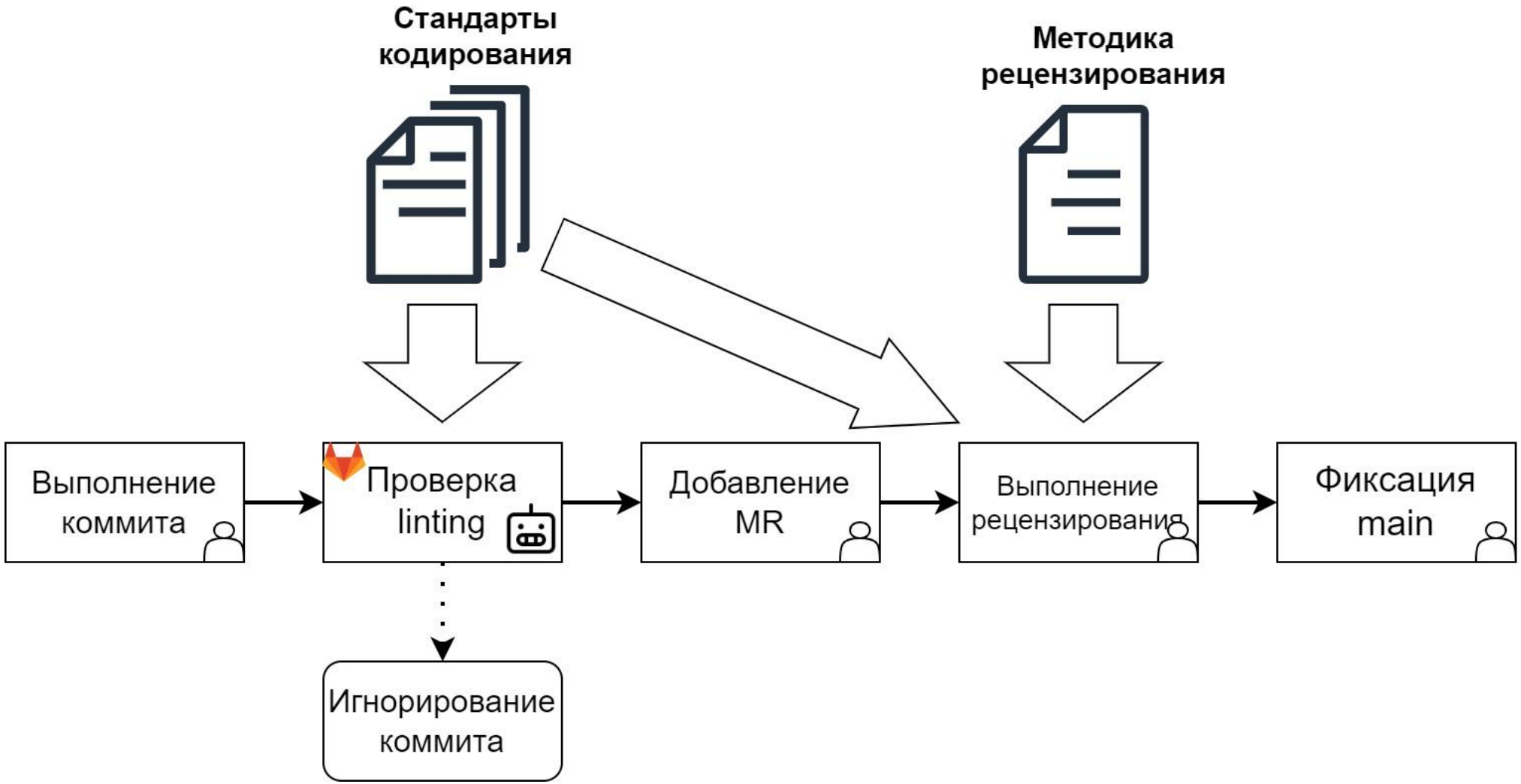
2 - года работы

215 - патчей

Рецензирование кода



Рецензирование кода



Рецензирование кода

Стандарты кодирования:

- ✓ https://google.github.io/styleguide/

Инструменты:

- ✓ https://github.com/analysis-tools-dev/static-analysis

- ✓ ***https://github.com/super-linter/super-linter***

Примеры стандартов кодирования МПО

- ✓ Условия в критичном коде ДОЛЖНЫ проверяться более одного раза
- ✓ Вычисления в критичном коде ДОЛЖНЫ быть проведены более одного раза
- ✓ Используемые константы НЕ ДОЛЖНЫ иметь тривиальные значения

Примеры стандартов кодирования МПО

- ✓ Условия в критичном коде ДОЛЖНЫ проверяться более одного раза
- ✓ Вычисления в критичном коде ДОЛЖНЫ быть проведены более одного раза
- ✓ Используемые константы НЕ ДОЛЖНЫ иметь тривиальные значения

- ✓ Области памяти между процессами ДОЛЖНЫ быть разделены на аппаратном уровне

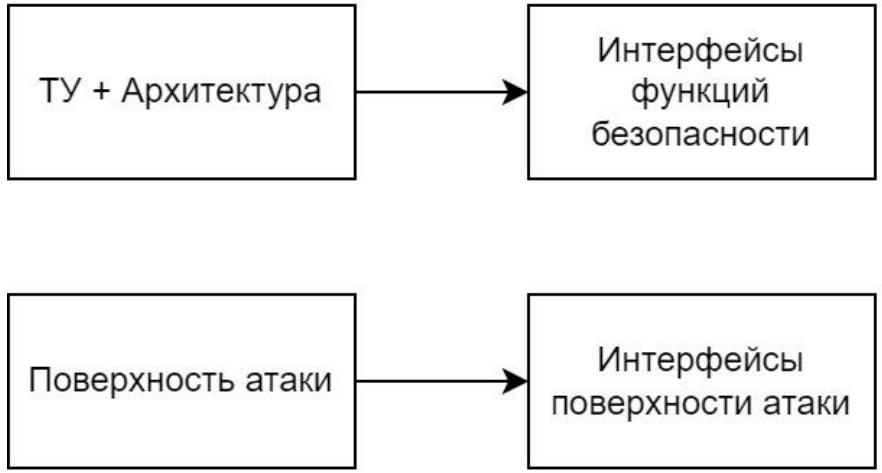
Примеры стандартов кодирования МПО

- ✓ Условия в критичном коде ДОЛЖНЫ проверяться более одного раза
- ✓ Вычисления в критичном коде ДОЛЖНЫ быть проведены более одного раза
- ✓ Используемые константы НЕ ДОЛЖНЫ иметь тривиальные значения

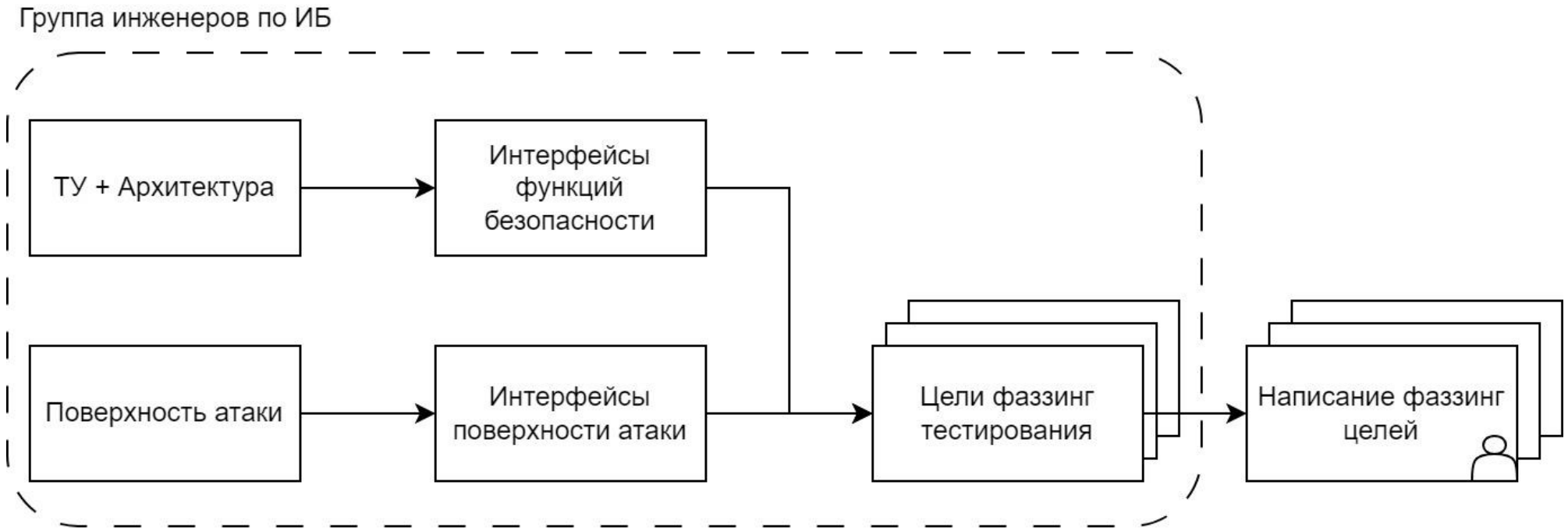
- ✓ Области памяти между процессами ДОЛЖНЫ быть разделены на аппаратном уровне

- ✓ Критичный код НЕ ДОЛЖЕН подвергаться автоматической оптимизации без повторной проверки мер безопасности

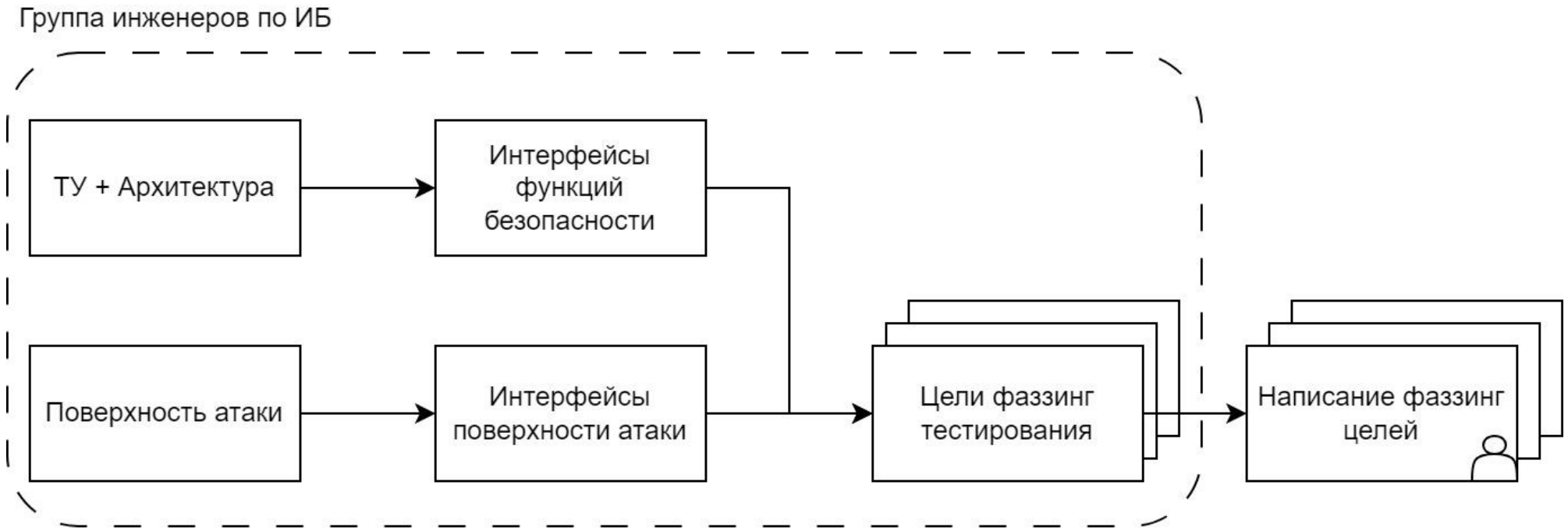
Фаззинг-тестирование



Фаззинг-тестирование

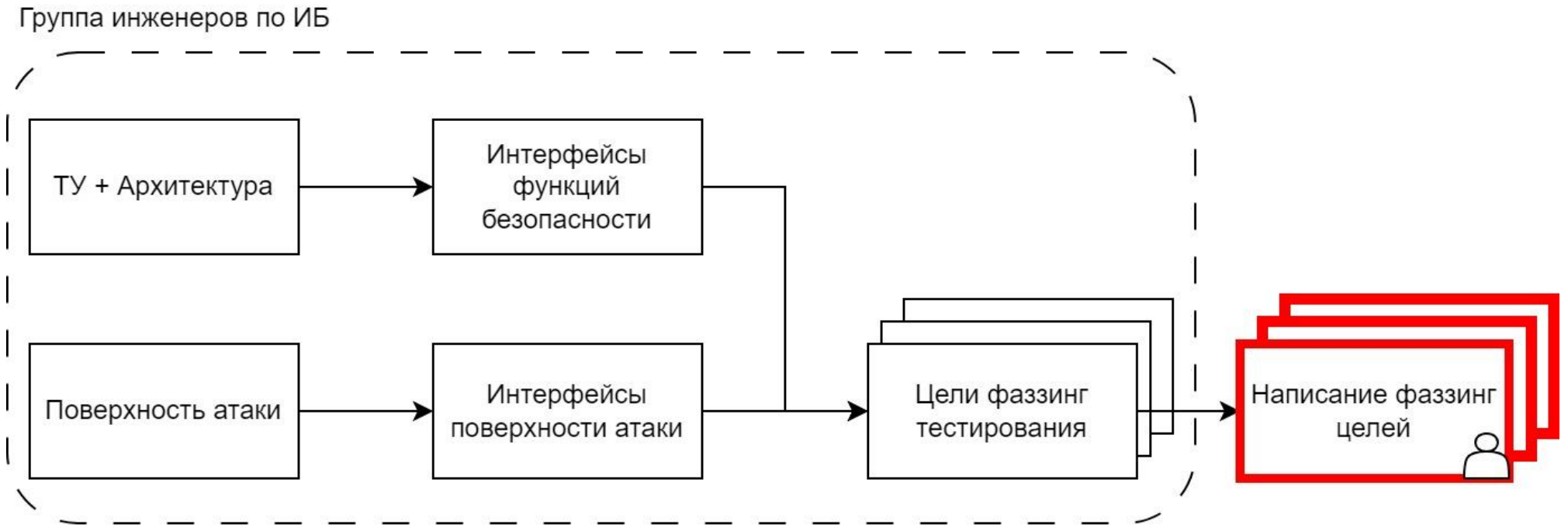


Фаззинг-тестирование



- ✓ AFL++ – C/C++
- ✓ Crusher – C/C++)
- ✓ Sharpfuzz – C#
- ✓ Jazzer – Java
- ✓ Native Go fuzz tests - Go

Фаззинг-тестирование



- ✓ AFL++ – C/C++
- ✓ Crusher – C/C++)
- ✓ Sharpfuzz – C#
- ✓ Jazzer – Java
- ✓ Native Go fuzz tests - Go

https://www.youtube.com/watch?v=fbhuzSpyUJc&ab_channel=Heisenbug

Фаззинг-тестирование (advanced)

Фаззинг-тестирование МПО:

- ✓ Способность проводить фаззинг-тестирование конкретного состояния ПО
- ✓ Эмуляция работы с внешними устройствами

Фаззинг-тестирование (advanced)

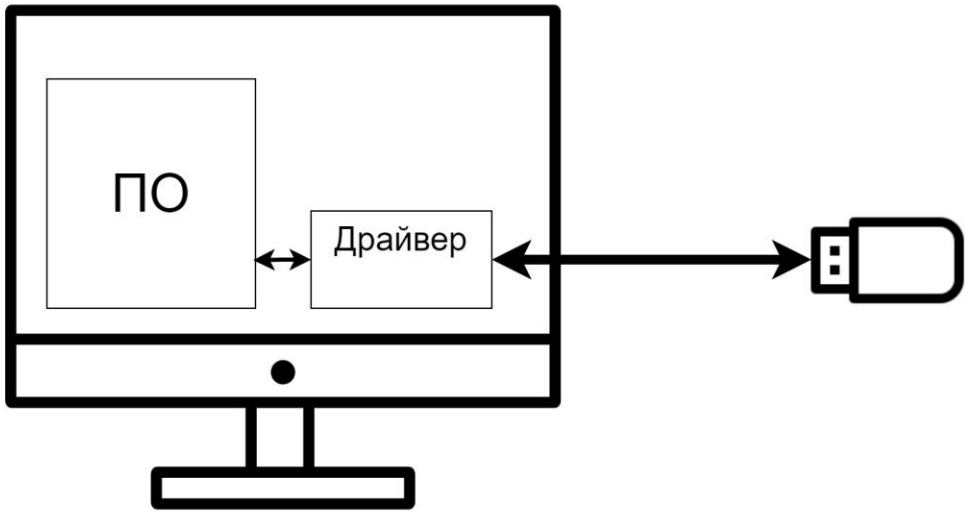
Фаззинг-тестирование МПО:

- ✓ Способность проводить фаззинг-тестирование конкретного состояния ПО
- ✓ Эмуляция работы с внешними устройствами

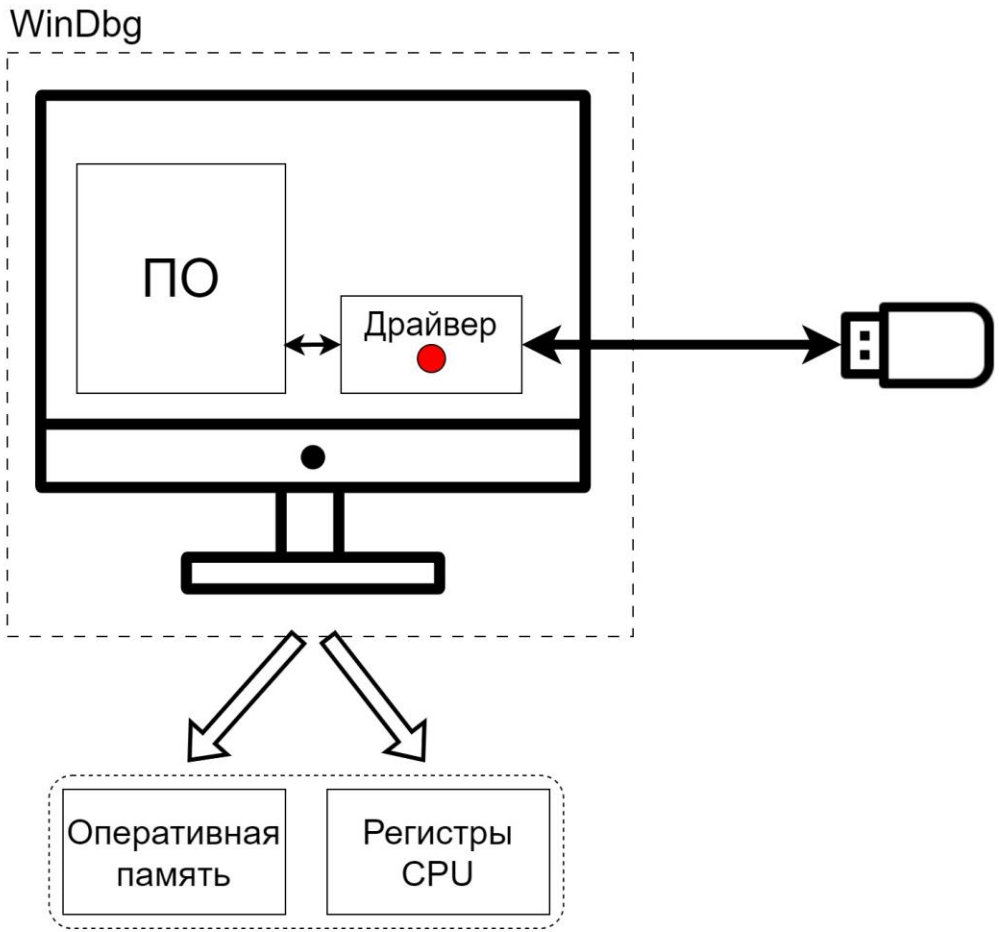
Что можно использовать:

- ✓ what the fuzz - https://github.com/0vercl0k/wtf
- ✓ Crusher - https://www.ispras.ru/technologies/crusher/
  - ✓ LuaQemu
  - ✓ Unicorn/Qiling
  - ✓ DualEmu
- ✓ Renode - https://renode.io/

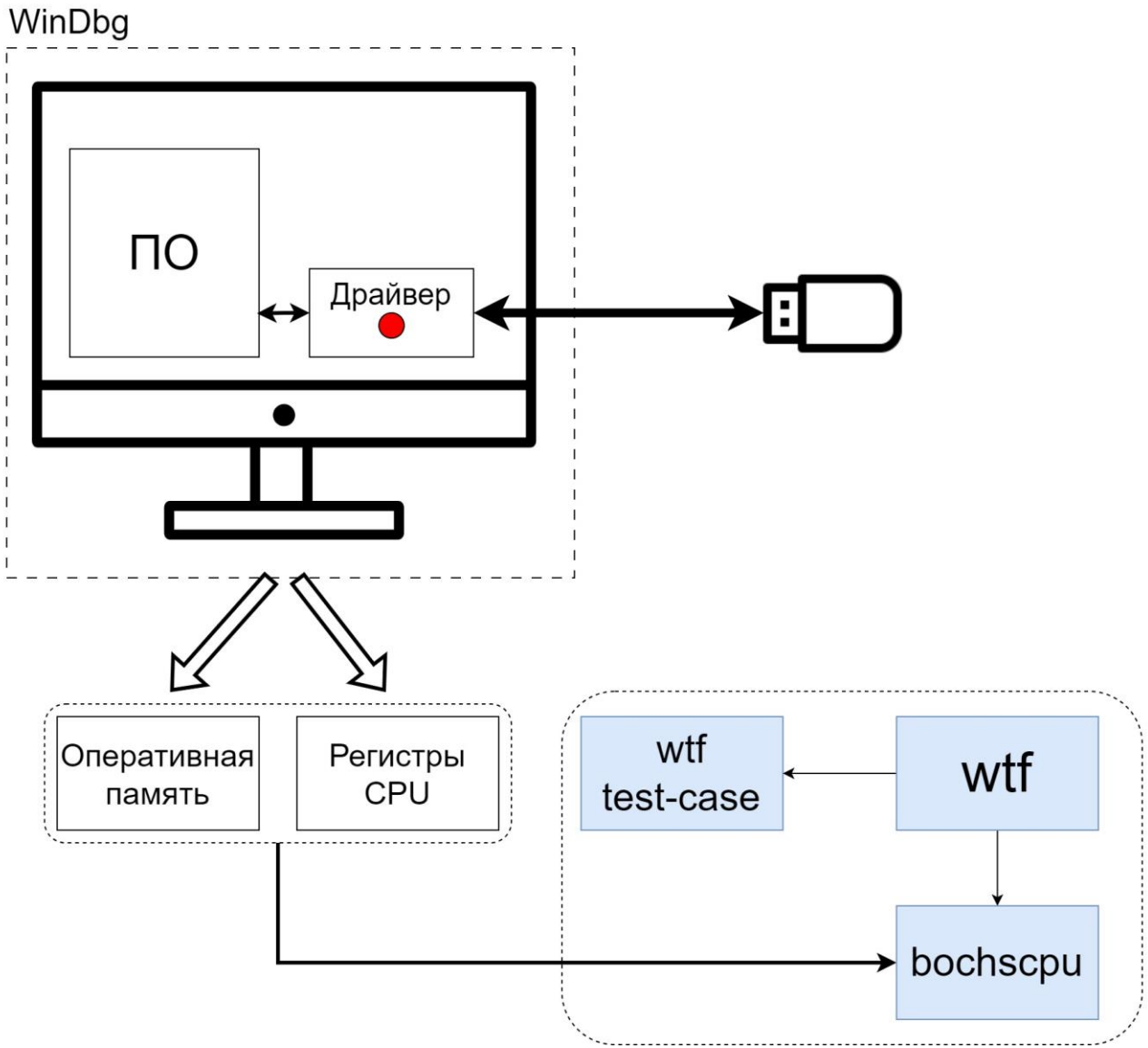
Пример использования what the fuzz



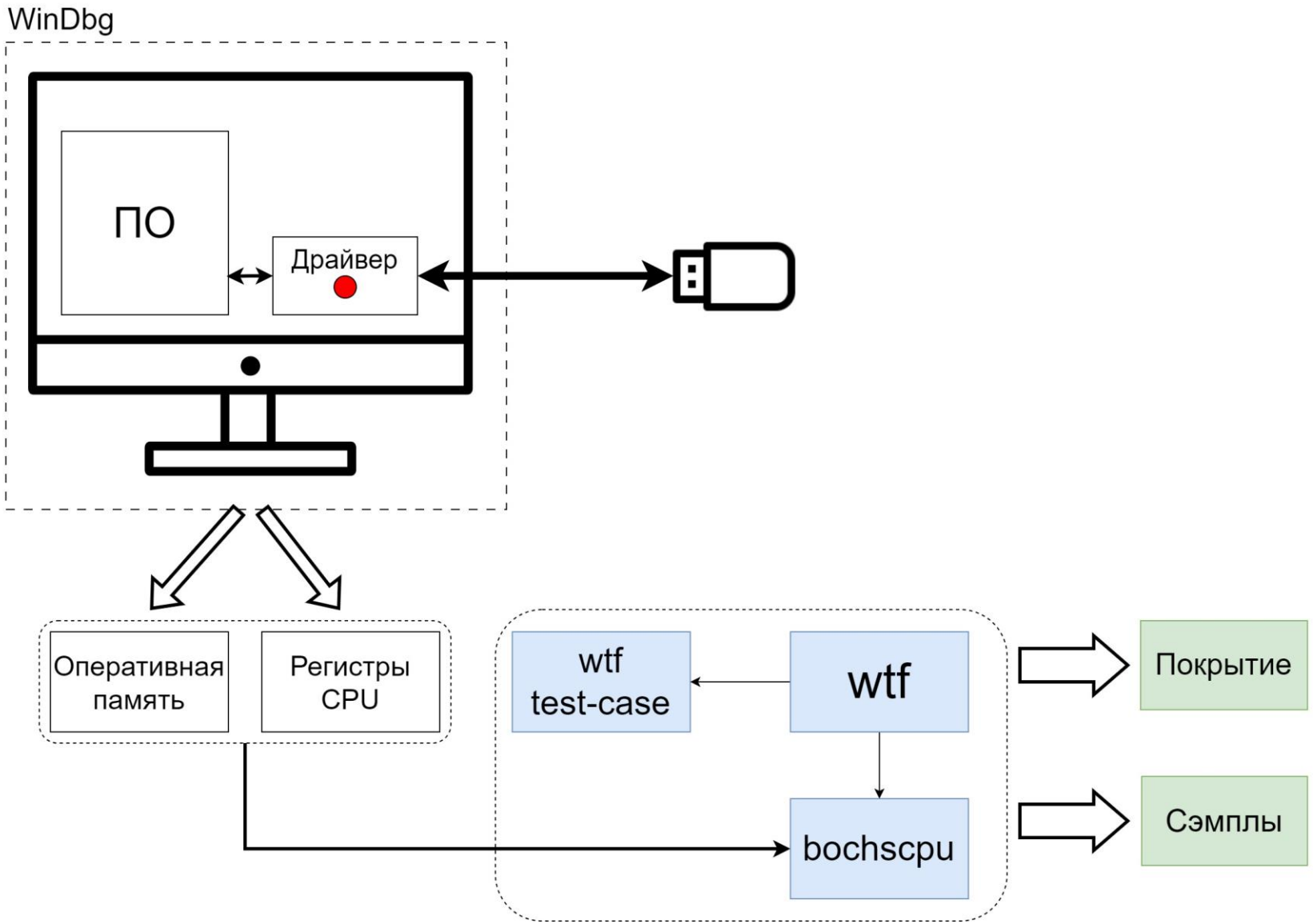
Пример использования what the fuzz



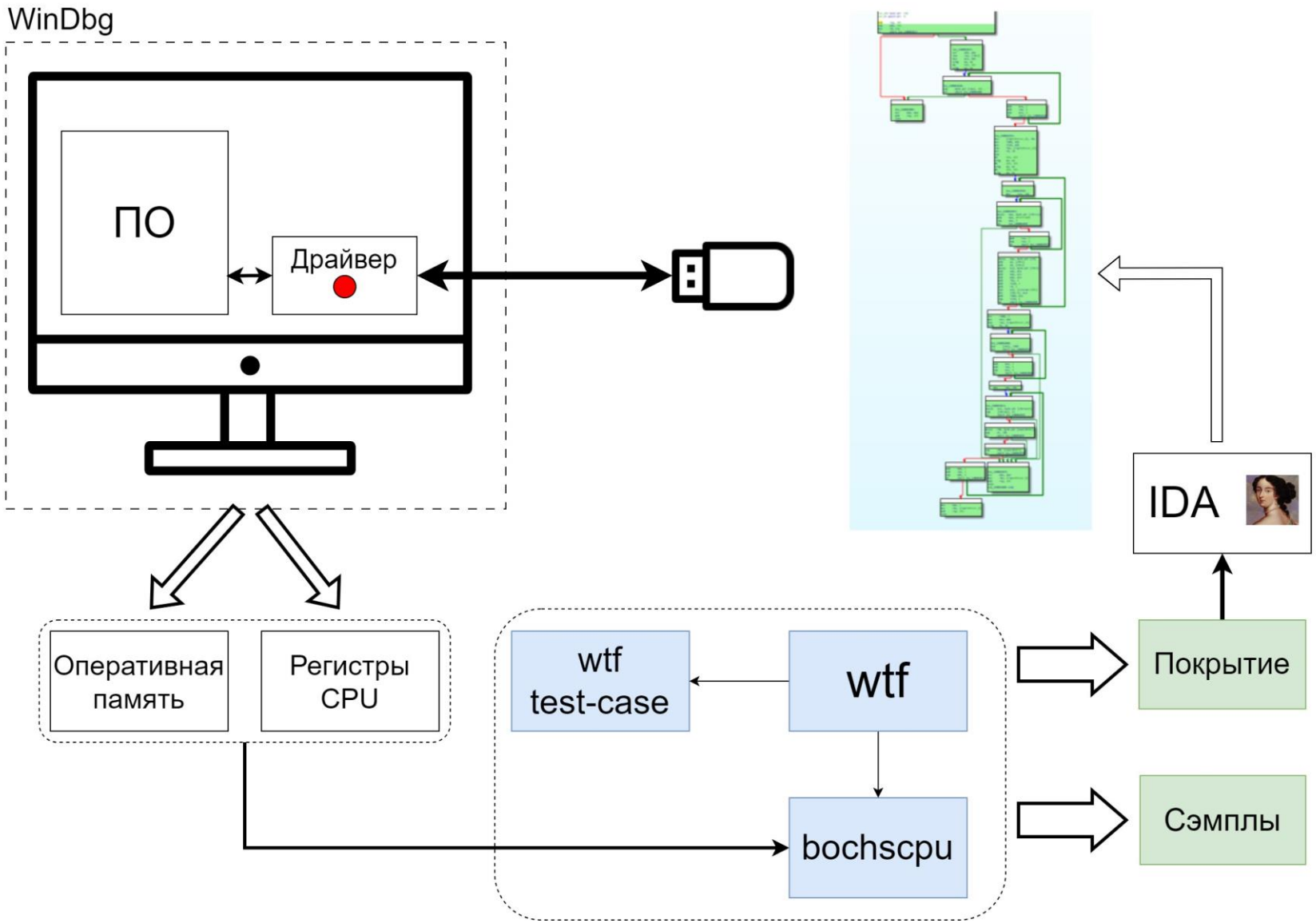
Пример использования what the fuzz



Пример использования what the fuzz



Пример использования what the fuzz

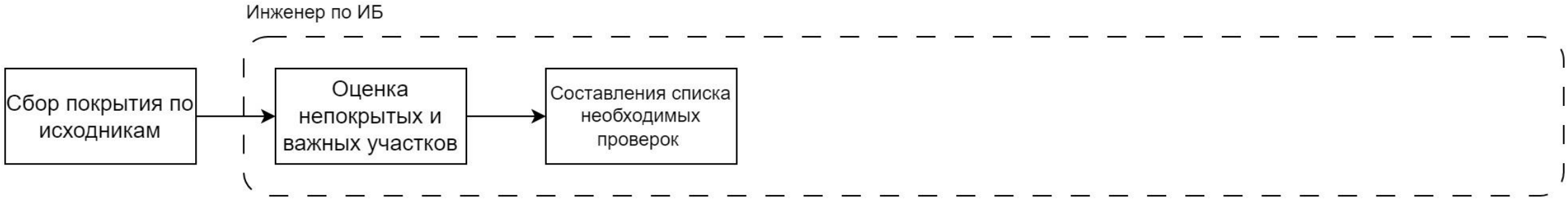


Тестирование на проникновение

Сбор покрытия по исходникам

Тестирование на проникновение

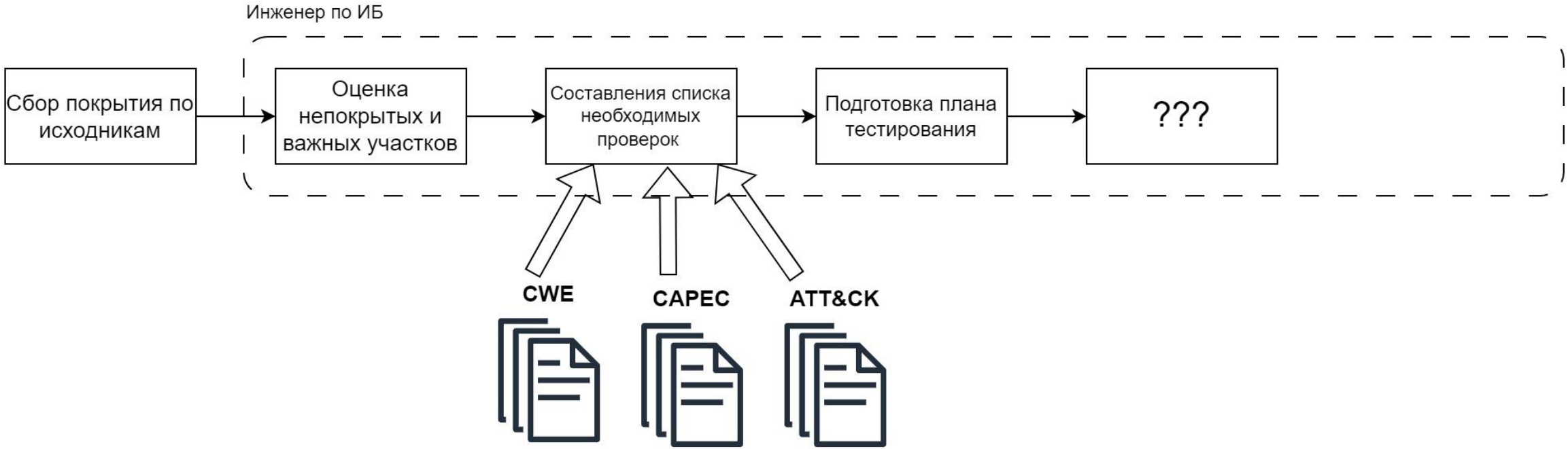
Тестирование на проникновение



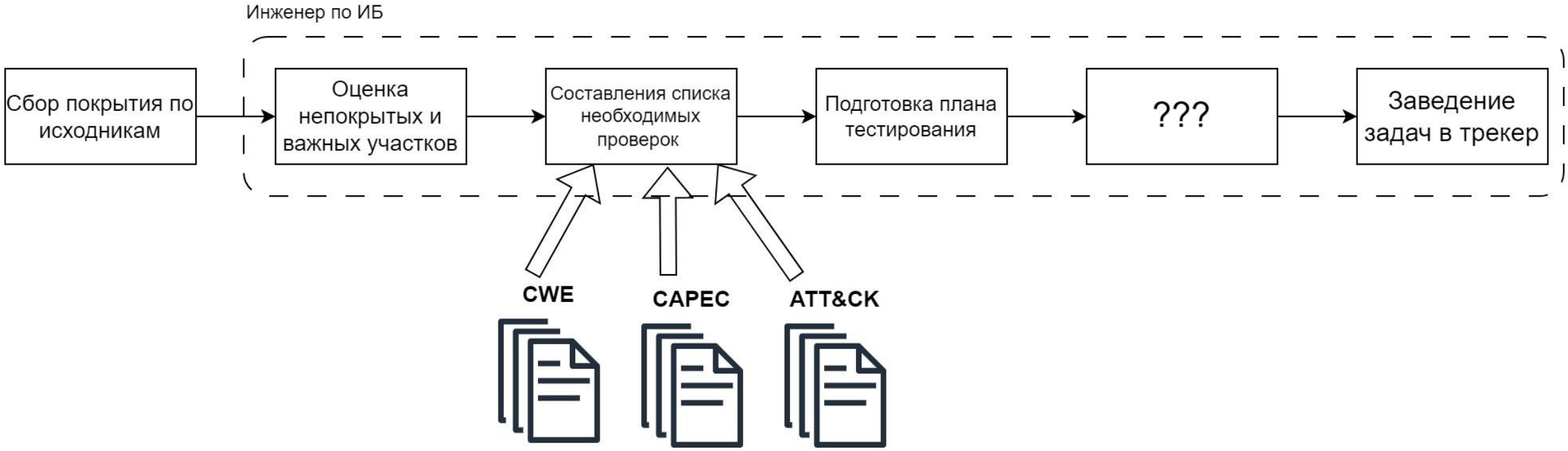
Тестирование на проникновение

Тестирование на проникновение

Тестирование на проникновение



Тестирование на проникновение



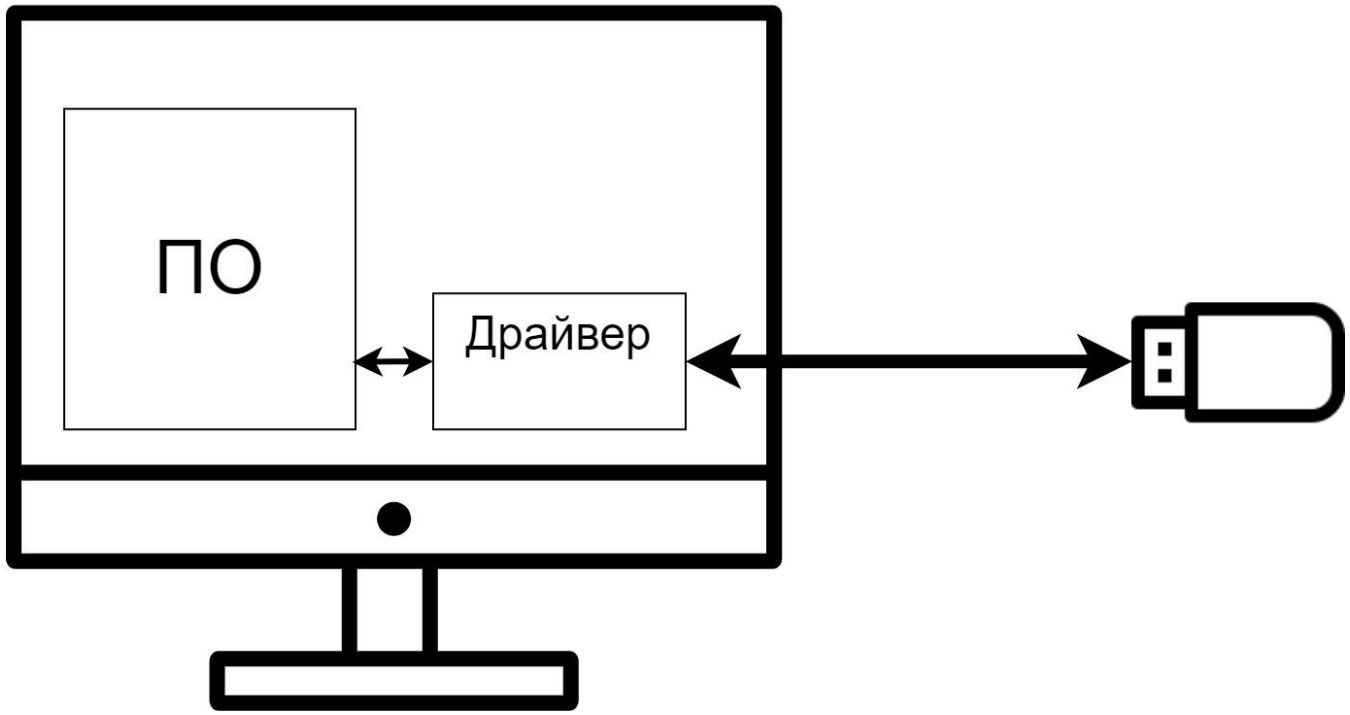
Пример найденной ошибки

CAPEC:

- ✓ CAPEC-212: Functionality Misuse
- ✓ CAPEC-124: Shared Resource Manipulation

ATT&CK:

- ✓ T1587: Develop Capabilities
- ✓ T1055: Process Injection



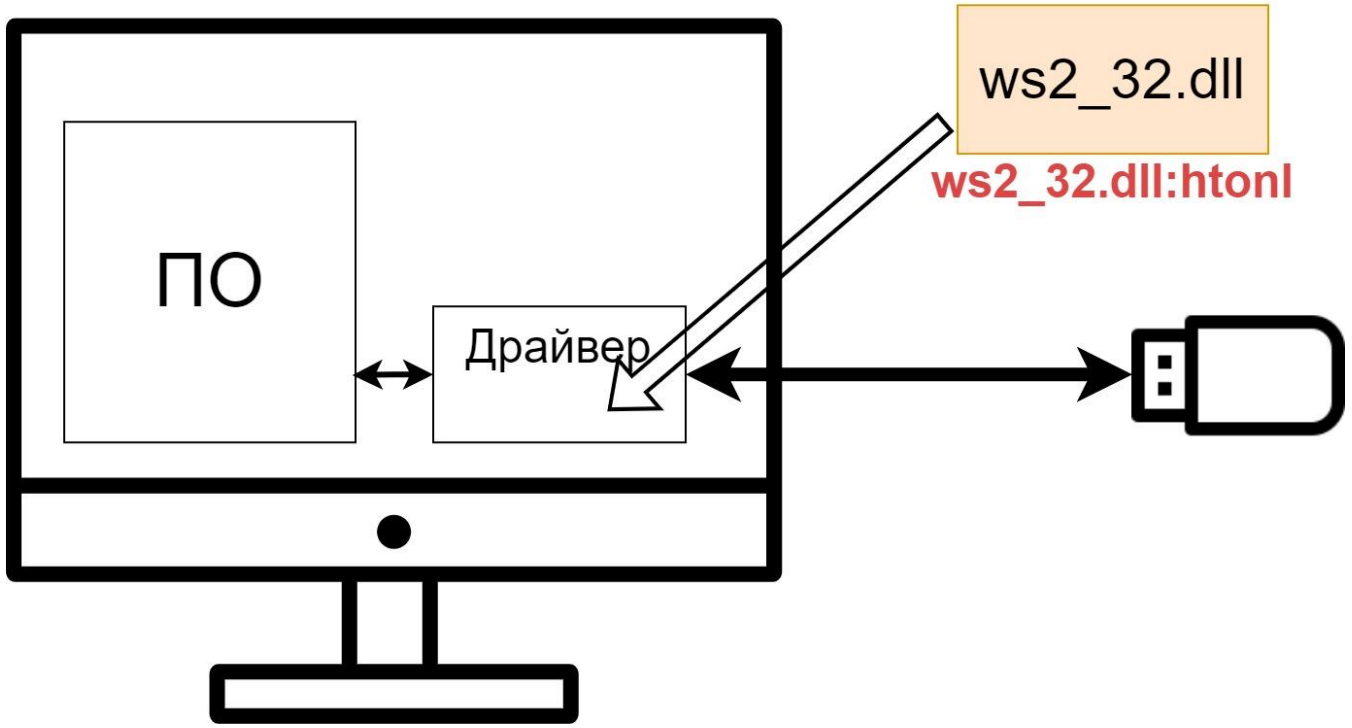
Пример найденной ошибки

CAPEC:

- ✓ CAPEC-212: Functionality Misuse
- ✓ CAPEC-124: Shared Resource Manipulation

ATT&CK:

- ✓ T1587: Develop Capabilities
- ✓ T1055: Process Injection



Пример найденной ошибки

[redacted] импортирует одну функцию из ws2_32.dll

Добавил(а) [redacted] Обновлено [redacted]

Статус:	Closed
Приоритет:	Нормальный
Назначена:	[redacted]
Версия:	[redacted]
Дата начала:	[redacted]
Срок завершения:	
Готовность:	100%
ОВЗ:	8.00 ч
Трудозатраты:	2.00 ч
Вид деятельности:	
Обнаружено в сборке:	
Закрыто в сборке:	

Описание

Я сканировал все исполняемые файлы из дистрибутива [redacted] при помощи Yara.
Набор правил был вот этот: [redacted]

На файлах [redacted] был выдан вердикт [redacted]
Но ведь [redacted] НЕ работает с сетью!

Причина оказалась в том, что из ws2_32.dll импортируется одна-единственная функция (по ординалу 0x0E): htonl.
Функция эта конвертирует big-endian dword в нативный.
Этого импорта (в сочетании со строчками "connect" и "socket" из рантайма STL) оказалось достаточно, чтобы получить [redacted]

Предлагаю заменить функцию htonl на свою. Это позволит выкинуть импорт Winsock / ws2_32.dll из [redacted]

Сертификация ФСТЭК России

Сертификация:

- ✓ Срок - 4-10 месяцев
- ✓ Участники – лаборатория, орган по сертификации, ФСТЭК России
- ✓ Документы – 76 приказ, ГОСТы

Сертификация ФСТЭК России

Сертификация:

- ✓ Срок - 4-10 месяцев
- ✓ Участники – лаборатория, орган по сертификации, ФСТЭК России
- ✓ Документы – 76 приказ, ГОСТы

Мотивация:

- ✓ Поставки ПО в КИИ, государственные органы и государственные предприятия
- ✓ Формальная верификация процессов и процедур в компании безопасной разработки

Что почитать

- ✓ Secure Software Development Framework (SSDF) Version 1.1

https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-218.pdf

- ✓ Supply-chain Levels for Software Artifacts

https://slsa.dev/

- ✓ MILS Architectural Approach Supporting Trustworthiness of the IIoT Solutions

https://www.iiconsortium.org/pdf/MILS_Architectural_Approach_Whitepaper.pdf

Аладдин - будь собой в электронном мире!

Спасибо!

Корнеева Светлана
Руководитель группы тестирования ЗНИ

Бауткин Антон
Руководитель отдела обеспечения безопасной разработки и автоматизации тестирования

АО "Аладдин Р.Д.