СВД
Встраиваемые
Системы

Безопасная разработка
Требования и реализация

Дужак Евгений

Руководитель группы разработки СЗИ

О чем мы будем говорить?

- **Организация процесса БР на примере опыта ЗОСРВ «Нейтрино»**
- **Статический анализ и технологии, связанные с ним**
- **Динамический анализ, его виды и технологии, связанные с ним**
- **Ручной анализ кода и технологии, связанные с ним**
- **Общие практики и технологии, которые могут облегчить и обезопасить разработку ПО**

Безопасная разработка.
Что это?

«Главное в ходе следственных действий не выйти на самих себя»

Безопасная разработка — методика разработки ПО, предотвращающая случайное внедрение уязвимостей и обеспечивающая устойчивость к воздействию вредоносных программ и несанкционированному доступу.

Безопасная разработка.
С чего начать?

«Путь в тысячу ли начинается с одного шага»

- Есть стандарт — ГОСТ Р 56939-2016
- Есть проект нового стандарта — ГОСТ Р 56939-202х
- Есть стандарт — ГОСТ Р 58412-2019
- Есть стандарт СА — ГОСТ Р 71207-2024
- Есть концепция DevSecOps
- Есть центры компетенции — испытательные лаборатории, участники ТК 362, институты и частные компании

ИСП РАН

Безопасная разработка.
Это процесс!

«У самурая нет цели, только путь!»

Нельзя взять и внедрить сразу все методики.

Безопасная разработка — это вечная борьба с как со внешним, так и со внутренним.

- **Внешнее — регуляторы и злоумышленники**
- **Внутреннее — разработчики и менеджеры**

Безопасная разработка. Уязвимости

- **Для чего вообще стоит заниматься безопасной разработкой? Ответ такой: потому что существуют уязвимости.**

- **Что такое уязвимость?**
  Уязвимость — недостаток ПО, который может быть использован для реализации угрозы безопасности информации.

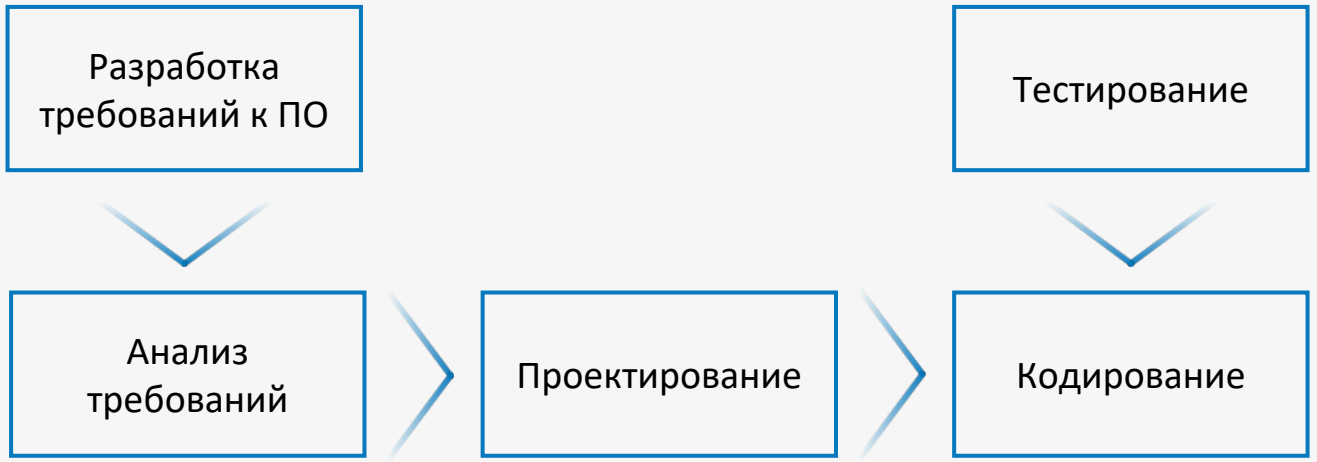
- **Недостаток — любая ошибка допущенная в рамках проектирования или реализации ПО.**

Безопасная разработка. Что входит в процесс?

Коротко — всё!

Подробно:

- Безопасность корпоративной сети
- Принципы разработки
- SAST, DAST, IAST, SCA
- DevOps/DevSecOps
- СКВ, СУП, СУТ, СУД



Безопасная разработка.
Требования к участникам процесса

«Учиться, учиться и ещё раз учиться!»

1 Продукт

2 Ресурсы

3 Компетенция

4 Здравый смысл

Безопасная разработка. Общий принцип

«Divide et impera»

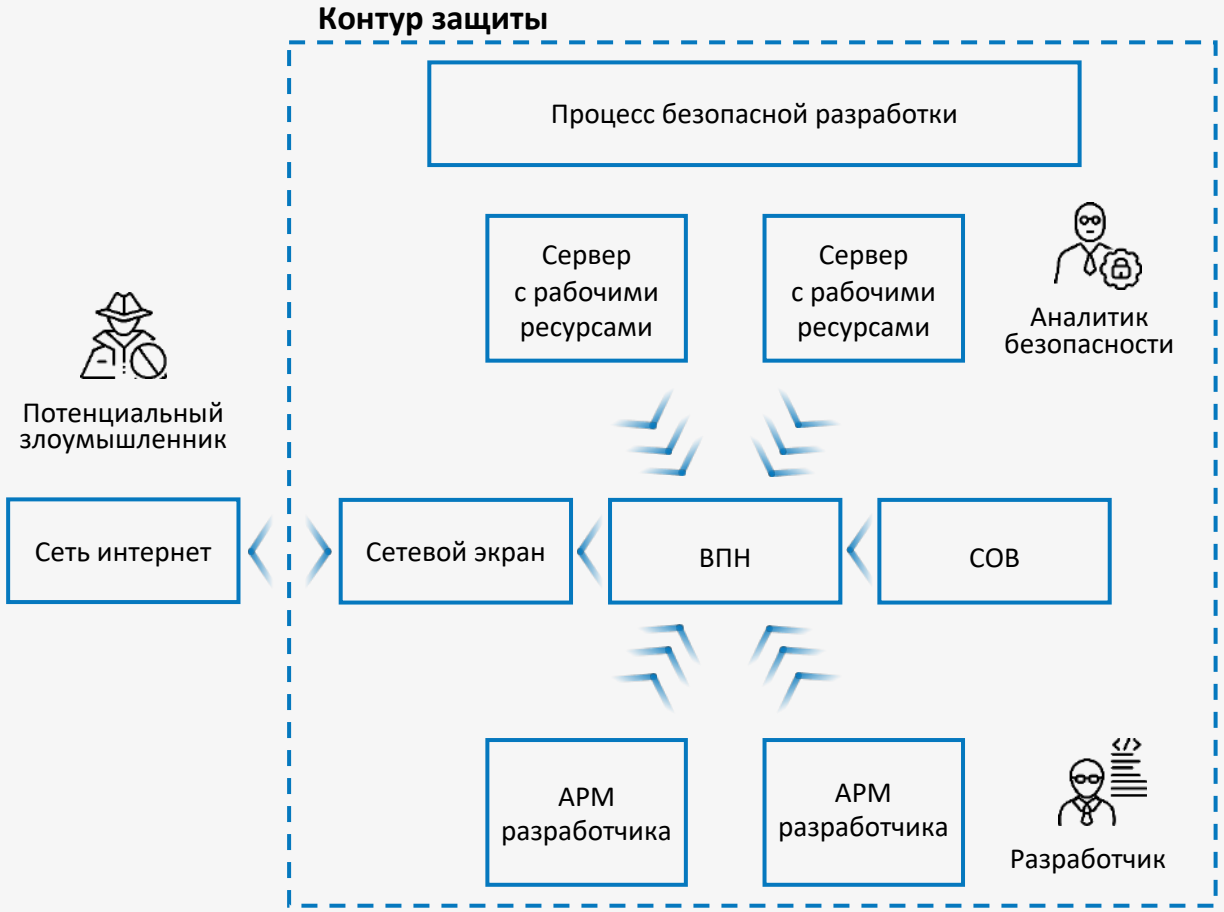
В первую очередь проект, в зависимости от масштаба, стоит разделить на части, разграничить ответственность команд (если проект большой) или конкретных участников команд (если проект небольшой).

Организации, команды, матрицы доступа и прочее.

Для чего это нужно?

- попытка избежать размытия ответственности, когда все отвечают за всё — никто не отвечает ни за что.
- в рамках команды происходит специализация, прекращается «any key»-ство, ускоряется рост компетенции разработчиков.

Безопасная разработка. Организация безопасности сети



«Если у вас нет паранойи, это еще не значит, что за вами не следят»

Процесс должен быть защищен как внутри, так и снаружи!

Не стоит забывать про правильную архитектуру командной сети.

Безопасная разработка.
Система контроля версий.

Для чего в принципе создавались системы контроля версий?

Главное их задачей является упрощение работы с изменяющейся информацией.

Что же скрывается за такой широкой формулировкой?

В первую очередь для разработчиков СКВ упрощает командную работу над сложными проектами.

СКВ фиксирует состояния исходников в необходимые моменты времени. Позволяет версионирование проекта.

Безопасная разработка.
Стиль кодирования и код-ревью

Стандарт кодирования — это набор правил и соглашений, используемых при написании исходного кода.

Бывают общепринятые и корпоративные.

Связанные вещи — линтеры (например, clang-format).

Код-ревью — дословно это рецензия на код.

(*плотно связан с СКВ)

Проверка некого этапа разработки, когда более опытный(-ые) (в идеале) разработчик(-и) просматривает(-ют) и оценивает(-ют) результаты работы исполнителя(-ей) задачи.

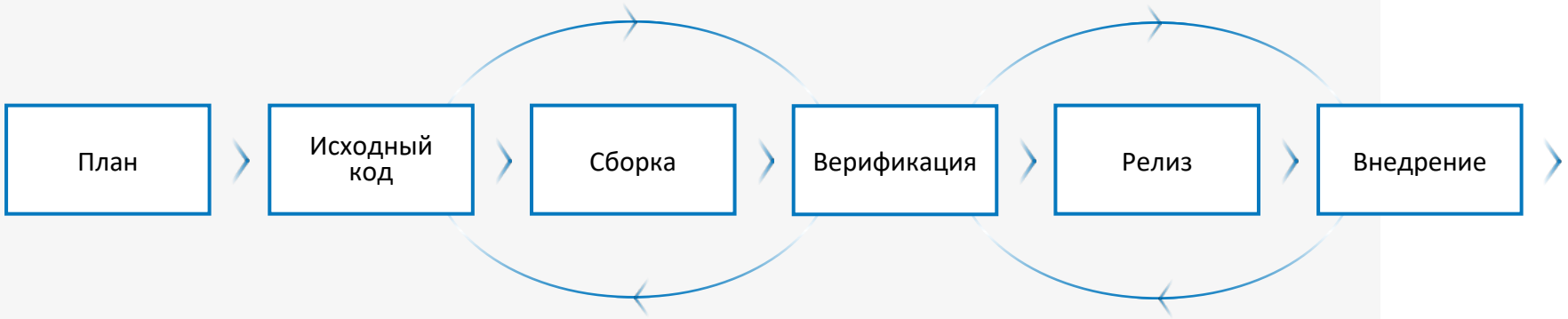
Безопасная разработка. Система CI/CD

«Безумие — повторение одного и того же действия раз за разом в надежде на изменение».

Задача систем CI/CD — ускорение и стабилизация изменений ПО, минимизация количества ошибок, повышение темпов сборки и улучшение качества выходного продукта.

Jenkins

CI/CD



Безопасная разработка.
Система управления проектами

Проекты бывают:

- Конечные (продукт, обычно аутсорс)
- Бесконечные (продукт, развиваемый «бесконечно»

Преимущества СУП:

- Продолжение концепции про разделение ответственности
- Может помочь всем и инженерам, и менеджерам
- Приближает к пониманию управляющих и управляемых

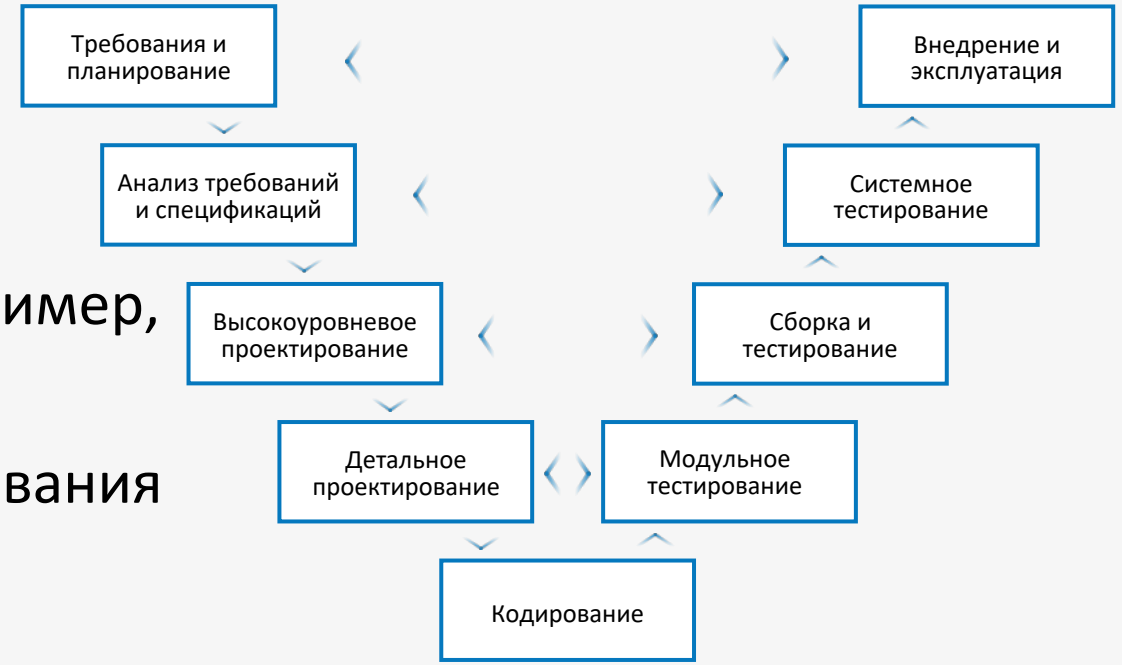
DEVPROM ALM

OpenProject

Jira

Безопасная разработка.
Система управления требованиями

- Требования бывают бизнес, бывают регуляторные
- Концепция схожа с СУПами
- Можно использовать «гибкую» СУП (например, open project)
- Связывает воедино весь процесс от требования до тестов



DEVPROM ALM

Jira

Безопасная разработка.
Система управления документацией

Да кому нужна документация?

Очевидно — заказчику/пользователю.

Неочевидно — элемент обучения собственных разработчиков.

+ Облегчает техническую поддержку.

Важно помнить про принцип разделения и здесь.

Виды документации:

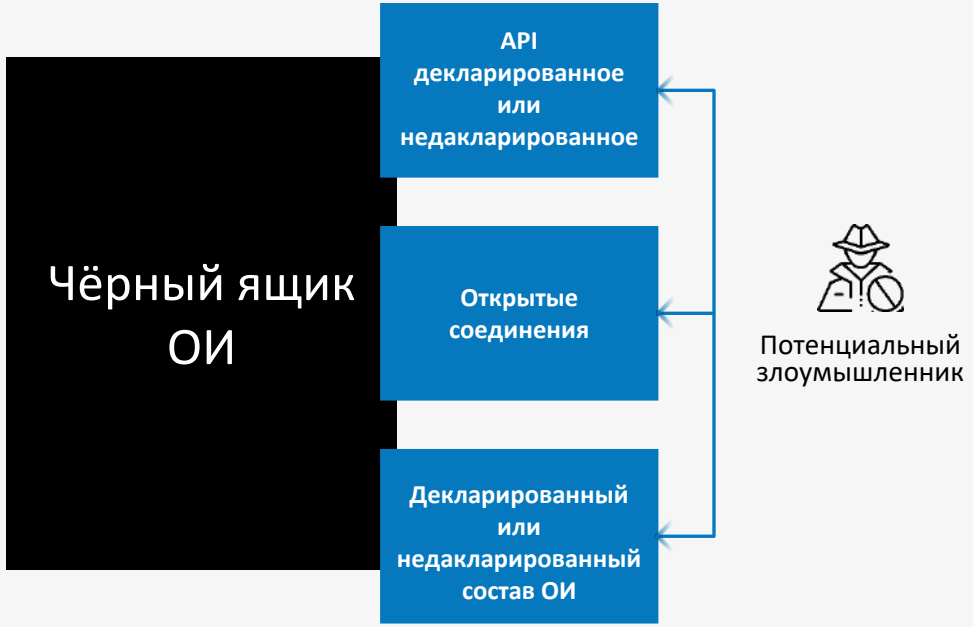
- Высокоуровневый дизайн, низкоуровневый дизайн.
- Описание API, howto-статьи.

Безопасная разработка.
Куда бить? Поверхность атаки

«Чтобы поймать преступника, нужно думать, как преступник».

Поверхность атаки -

совокупность всех возможных точек входа, через которые злоумышленник может получить несанкционированный доступ к информационной системе организации или отдельного пользователя.



Безопасная разработка. Ручная инспекция кода

Когда автоматизированные инструменты нельзя применять?

- Его величество assembler
- Нет инструмента (редкий ЯП)

Опираемся на экспертную оценку.

10000 строк кода — максимум который не стоит превышать.

Есть альтернативы: openreil/binside

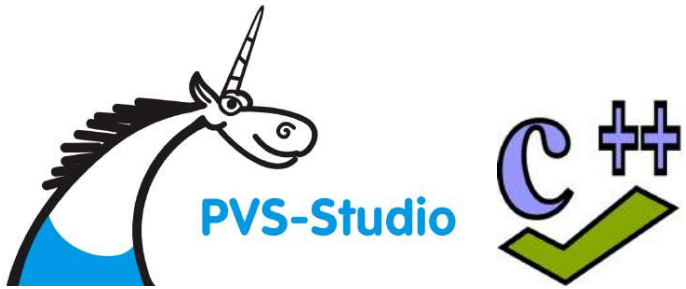
```
1   global _start
2
3   section .rodata
4       hello_world: db "Hello world!", 0xA, 0x0
5
6   section .text
7   _start:
8       mov eax, 0x04
9       mov ebx, 0x1
10      mov ecx, hello_world
11      mov edx, 14
12      int 0x80
13      mov eax, 0x01
14      mov ebx, 0
15      int 0x80
```

Безопасная разработка. Статический анализ

ГОСТ Р 71207-2024

Поиск потенциальных ошибок, уязвимостей и нарушений стандартов кодирования.

Существуют требования как к инструменту, так и к специалистам проводящим анализ результатов.



Безопасная разработка. Анализ зависимостей

SCA-анализатор — инструмент, который осуществлять поиск уязвимостей в сторонних open-source компонентах, подключенных к проекту.

Безопасная разработка. Динамический анализ

Виды:

- Трассировка маркированных данных
- Фаззинг-тестирование
  - Мутационное
  - Гибридное
- Тестирование
  - Модульное
  - Функциональное

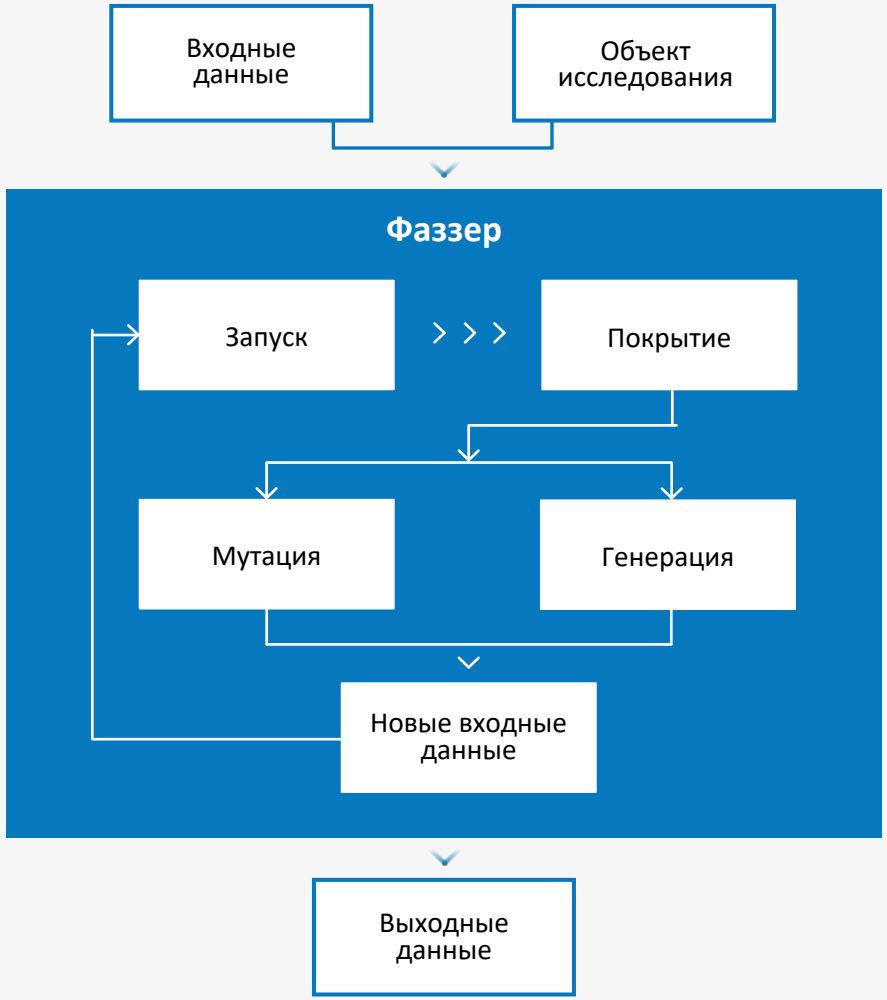
Безопасная разработка. Фаззинг-тестирование

Бывает:

- Мутационным
- Гибридным

Гибридное — мутационное + символьное или конколическое исполнение/выполнение.

Цель — внести детерминированность.



Безопасная разработка. Трассировка маркированных данных

Исследование в рамках процесса (taintgr

Valgrind — эмулятор для процесса.

Исследование в рамках системы (полносистемные эмуляторы).

Panda — эмулятор для системы.

Безопасная разработка. Тестирование

Тестирования — важная часть этапа верификации в рамках CI/CD-процесса.

Метрики:

- Увеличение количества тестов
- Увеличение количества успешно пройденных тестов
- Увеличение кодового покрытия

#expect

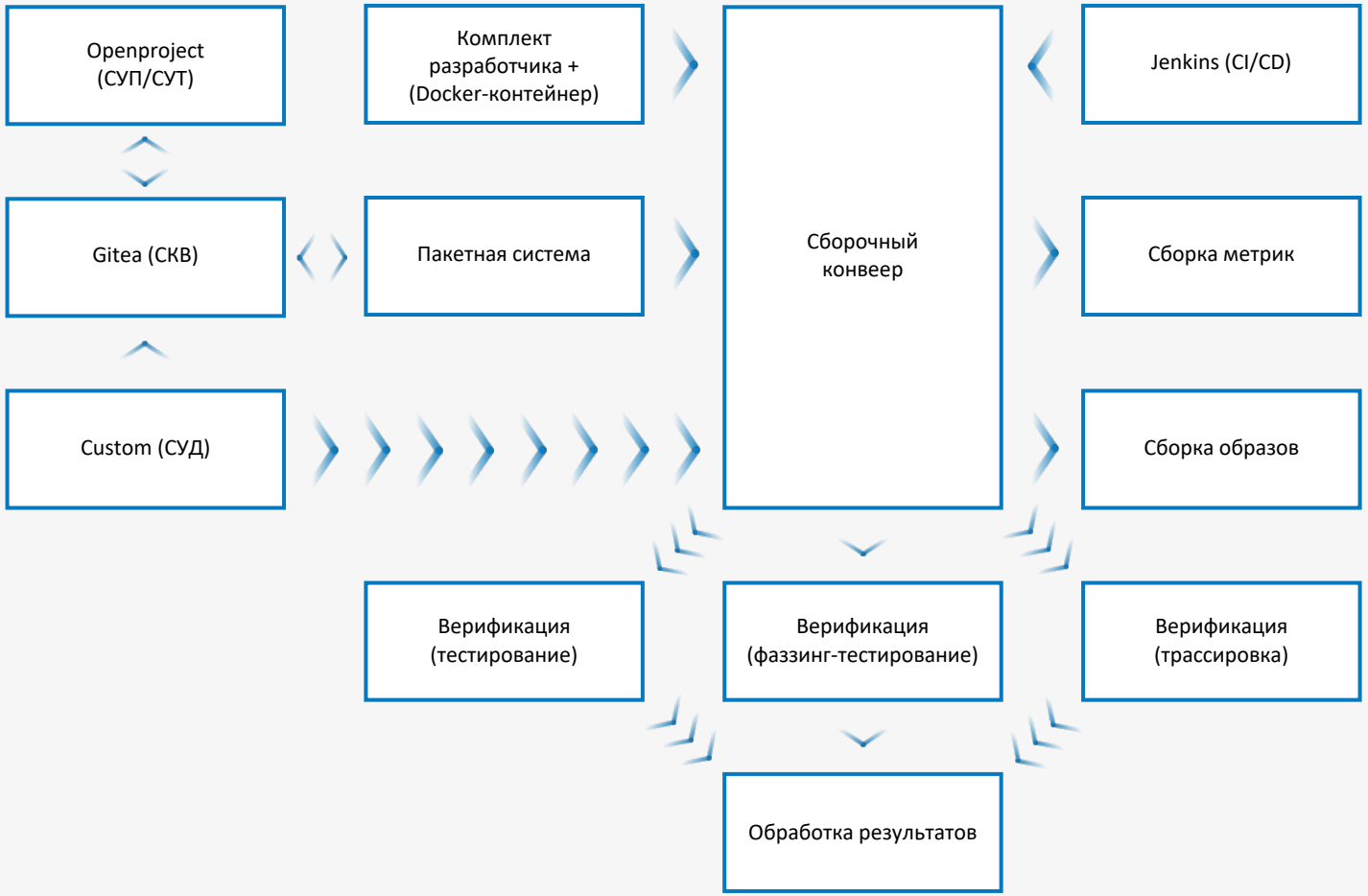
Безопасная разработка. Безопасные компиляторы

- Борьба с влиянием оптимизации на рост уязвимостей в ПО
- Борьба с влиянием версии компилятора на «старый» код
- Внедрение дополнительных средств защиты, затрудняющих реализацию угроз безопасности

Три класса безопасности, в зависимости от них замедляется код.

ИСП РАН

Безопасная разработка. Пример



Заключение

«Дорогу осилит идущий».

1 **Автоматизируй!** CI/CD, СКВ, SAST, DAST.

2 **Учись!** ГОСТы, программы обучения.

3 **Внедряй!** Стандарты кодирования, SAST, DAST.

4 **Управляй!** Проектами, требованиями, документацией.

СВД
Встраиваемые
Системы

Спасибо за внимание!

Дужак Евгений

Руководитель группы разработки СЗИ

+7 (812) 346-89-56
www.kpda.ru