СВД
Встраиваемые
Системы

РБПО реального времени

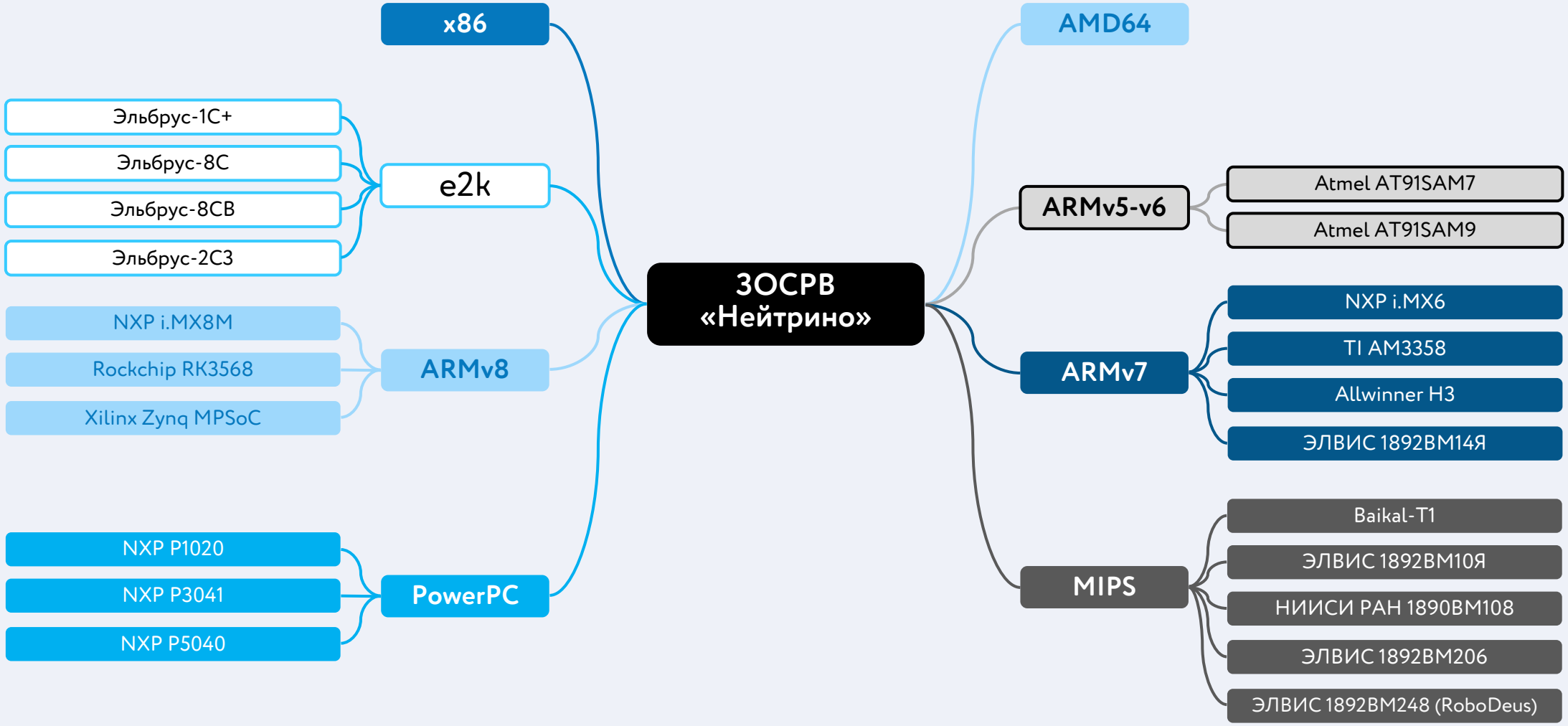
Дужак Евгений

Руководитель отдела аналитики и технологий безопасности

Привет! Мы «СВД ВС»

1. Петербургская компания (70+ сотрудников, из них 44 разработчика).
2. Основной продукт: ЗОСРВ «Нейтрино»
3. На рынке встраиваемых ОС с 2002 года
4. Честный “fork” ОСРВ QNX 6.5
5. Обеспечиваем полный цикл разработки ОСРВ от прикладного ПО до ядра
6. СЗИ, глубоко интегрированные в компоненты ОС
7. Сертифицируем ЗОСРВ в различных системах сертификации. МО, ФСТЭК, ФСБ и ФБ.
8. Комплект разработчика разрабатываем и сопровождаем сами.

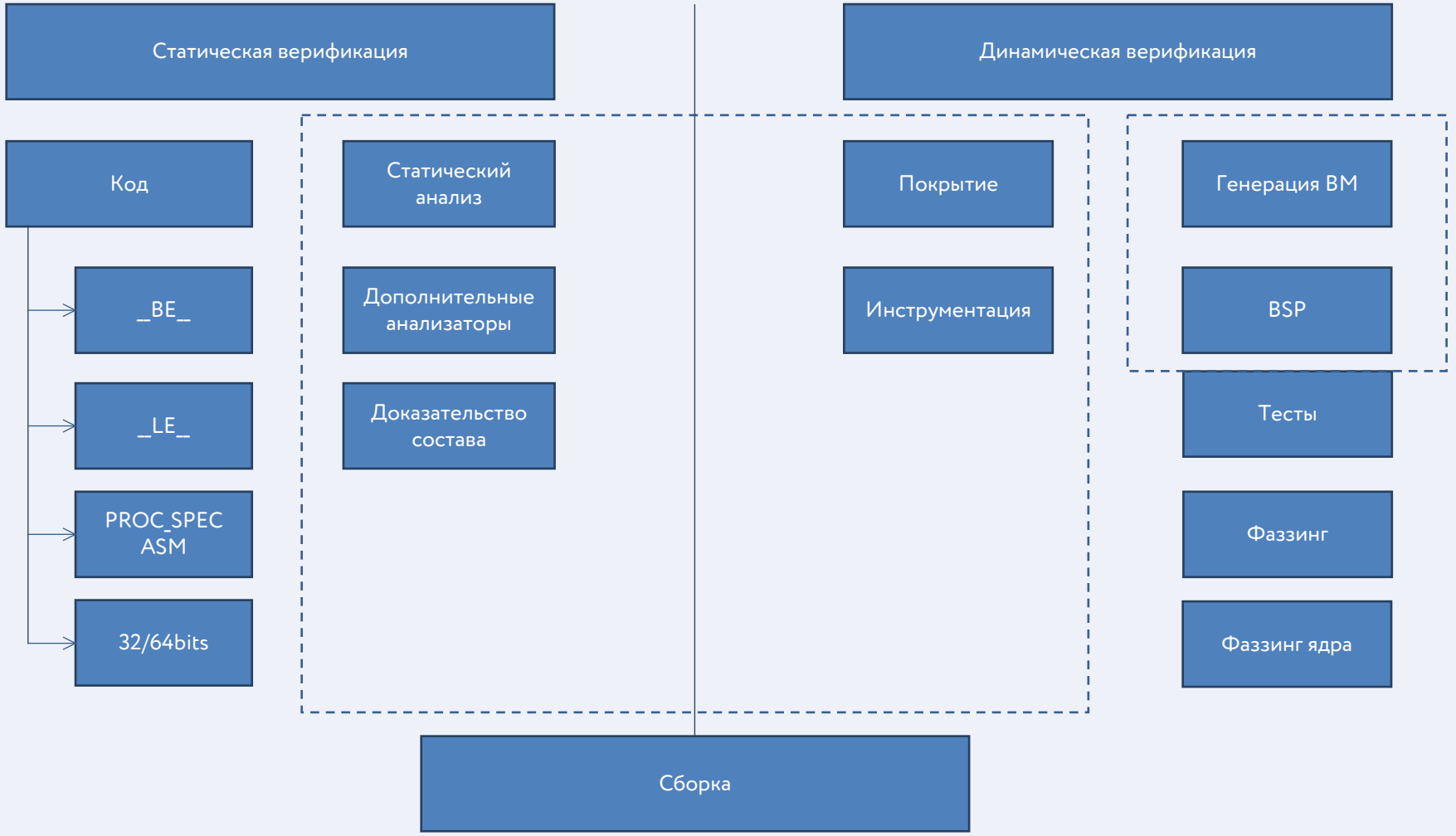
Зоопарк для ЗОСРВ.
Архитектуры, процессоры, устройства.



Зоопарк для ЗОСРВ. Состав.

- Хранение
  - Git-репозиторий
  - Мета
  - Лицензия
  - Принадлежность
  - Идентификатор хранения
  - Связи
  - Код
- Организация использования
  - Пакет
  - Инструментации
  - Архитектура
  - Цель

Зоопарк для ЗОСРВ. Верификация.



Как выглядят пакеты?

Дистрибутив:

– base/extended/optional

Пакеты:

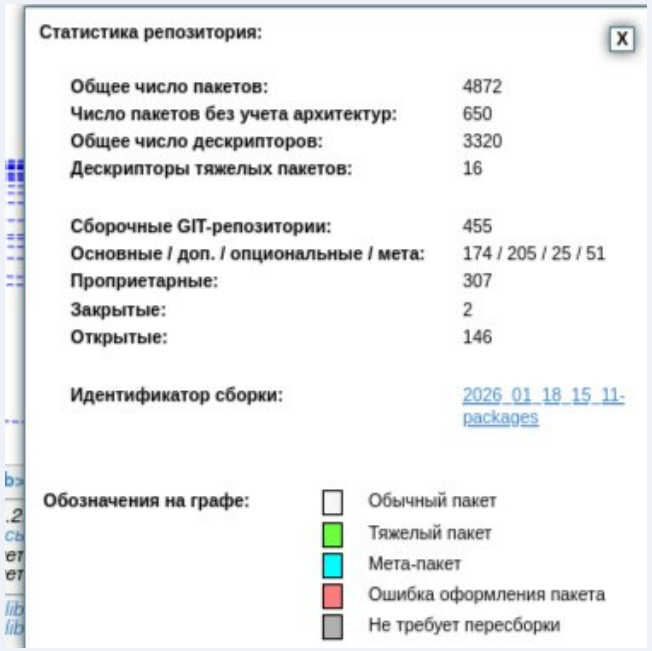
– Мета/тяжелые/обычные

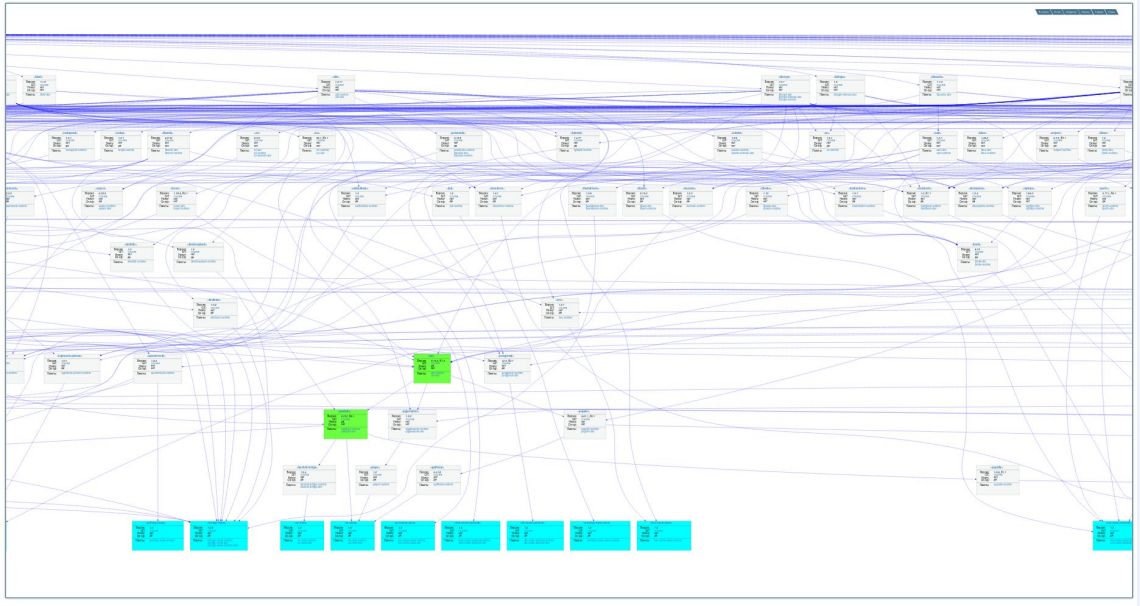
Сборки бывают трех видов:

– верификационная

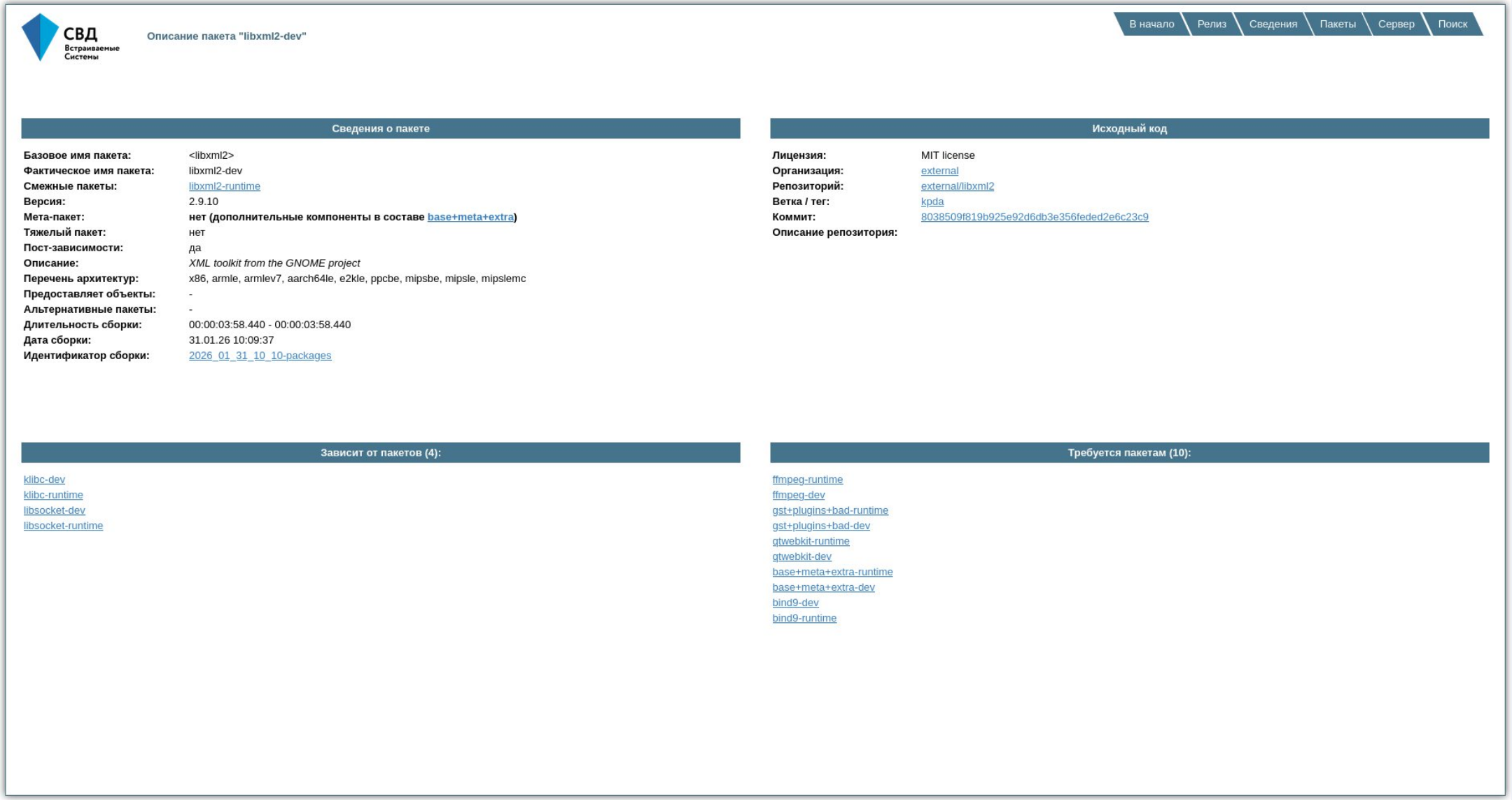
– инкрементальная

– полная





Как выглядят пакеты?



СВД
Встраиваемые Системы

Описание пакета "libxml2-dev"

В начало | Релиз | Сведения | Пакеты | Сервер | Поиск

Сведения о пакете

Базовое имя пакета:	<libxml2>
Фактическое имя пакета:	libxml2-dev
Смежные пакеты:	libxml2-runtime
Версия:	2.9.10
Мета-пакет:	**нет (дополнительные компоненты в составе base+meta+extra)**
Тяжелый пакет:	нет
Пост-зависимости:	да
Описание:	*XML toolkit from the GNOME project*
Перечень архитектур:	x86, armle, armlev7, aarch64le, e2kle, ppcbe, mipsbe, mipsle, mipslemc
Предоставляет объекты:	-
Альтернативные пакеты:	-
Длительность сборки:	00:00:03:58.440 - 00:00:03:58.440
Дата сборки:	31.01.26 10:09:37
Идентификатор сборки:	2026_01_31_10_10-packages

Исходный код

Лицензия:	MIT license
Организация:	external
Репозиторий:	external/libxml2
Ветка / тег:	kpda
Коммит:	8038509f819b925e92d6db3e356feded2e6c23c9
Описание репозитория:	

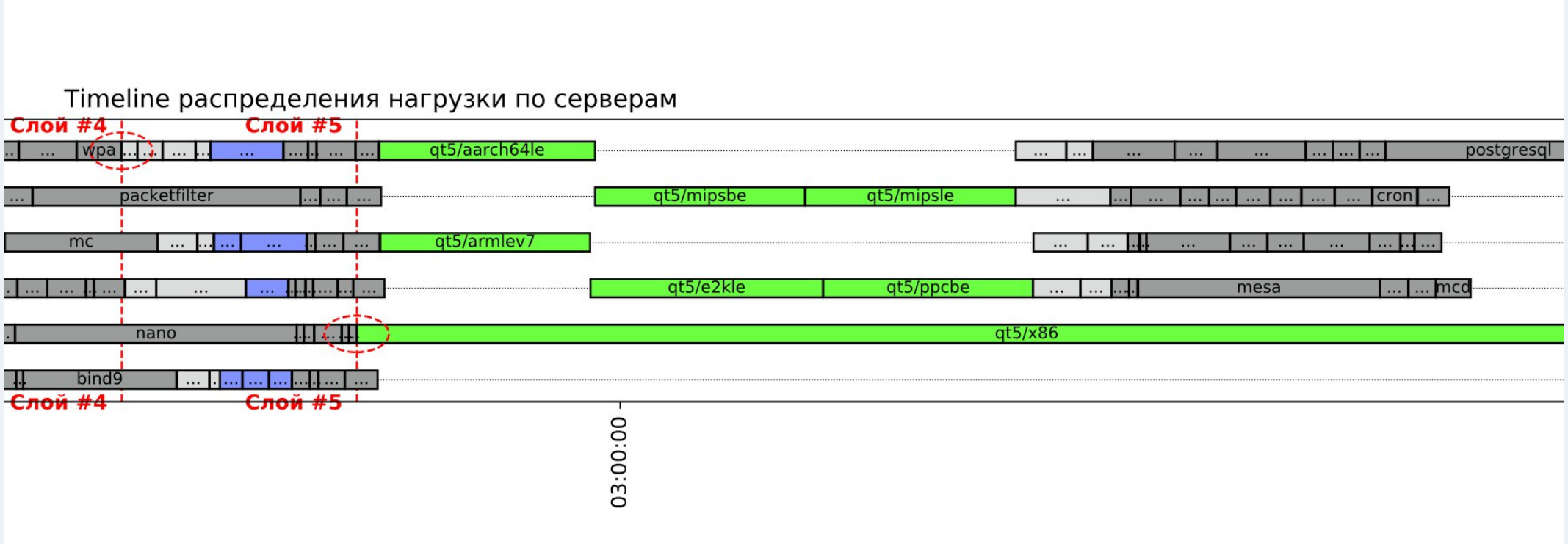
Зависит от пакетов (4):

klibc-dev
klibc-runtime
libsocket-dev
libsocket-runtime

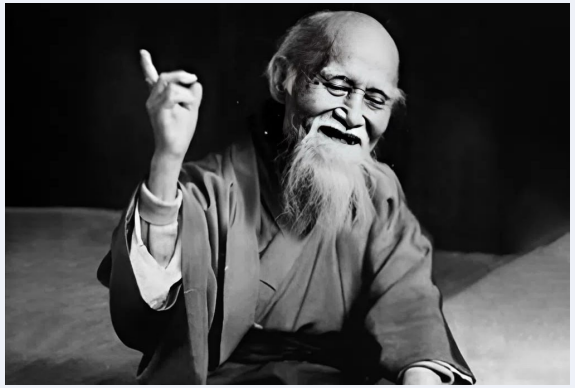
Требуется пакетам (10):

ffmpeg-runtime
ffmpeg-dev
gst+plugins+bad-runtime
gst+plugins+bad-dev
qtwebkit-runtime
qtwebkit-dev
base+meta+extra-runtime
base+meta+extra-dev
bind9-dev
bind9-runtime

Сборка пакетов



Выстраивание процессов.
У самурая нет цели, есть только путь



1. **Начальный**
   Процессы непредсказуемы
   Слабо контролируются
   Появляются как ответ на внешнее

2. **Управляемый**
   Процессы определены на уровне проектов
   Появляются как ответ на внешнее

3. **Определенный**
   Процессы определены на уровне организации и формируют события

4. **Управляемый на основе количественных данных**
   Процессы измеримы и контролируются

5. **Оптимизирующий**
   Фокус на постоянном улучшении процессов

Аудит - зло!

Аудит - зло?

Аудит - добро!

см. SWCMMI / ISO/IEC 33061

Выстраивание процессов. Ловушка уровня зрелости. Мотивация к изменению.

Симптомы ловушки уровня зрелости:
– «Мы всегда так делали, и это работало»
– Страх экспериментов с новыми технологиями и подходами
– Ориентация на стабильность в ущерб инновациям
– Процессы становятся самоцелью, а не инструментом

Есть несколько путей выхода из ловушки, делятся они на два пути:
– **«Внешний»**
– **«Внутренний»**

«Внешний» – столкновение интересов с:
– лабораторией
– регулятором
– аудитором

«Внутренний» – столкновение интересов у:
– сотрудников по вертикали
– сотрудников по горизонтали

Уговор дороже денег!

1. Процесс = регламент
2. Регламент скрывает за собой субрегламенты, методики

Можно ли описать идеальный процесс?

«Можно, а зачем?»

Регламент должен быть описанием реального положения дел.

Все участники процесса должны осознавать свою ответственность за свою часть и уметь договариваться о ее границах

Проблемы. САО.

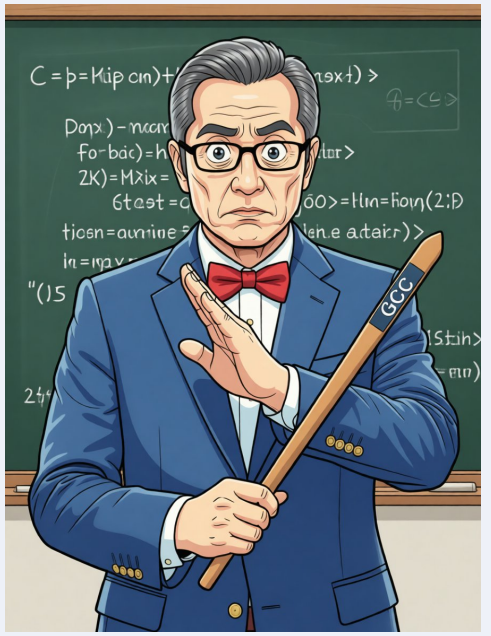
Исторически следили за качеством кода.

Этапы становления САО в «СВД ВС»:

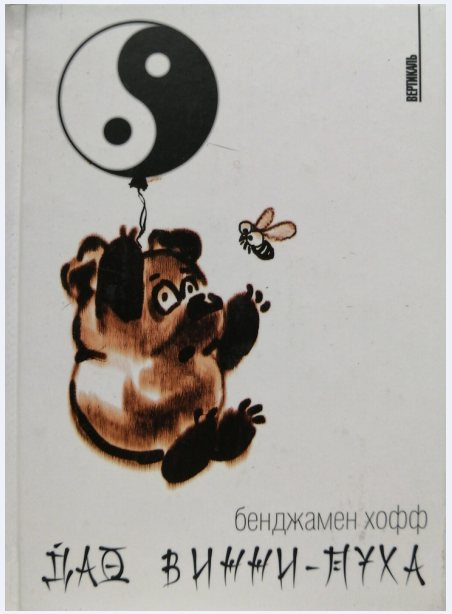
1) Строгий компилятор (-Wall –Werror –pedantic)

2) Opensource-анализатор (интеграция с build системой на уровне отдельного вызова)

3) Коммерческий анализатор с интеграцией на уровне сборки пакетов



Проблемы. ДАО.



Дао Винни-Пуха

Динамический анализ ОО включает:
а) тестирование модулей ОО (ДАО.1)
б) фаззинг-тестирование ОО (ДАО.2)
в) системное тестирование ОО (ДАО.3)

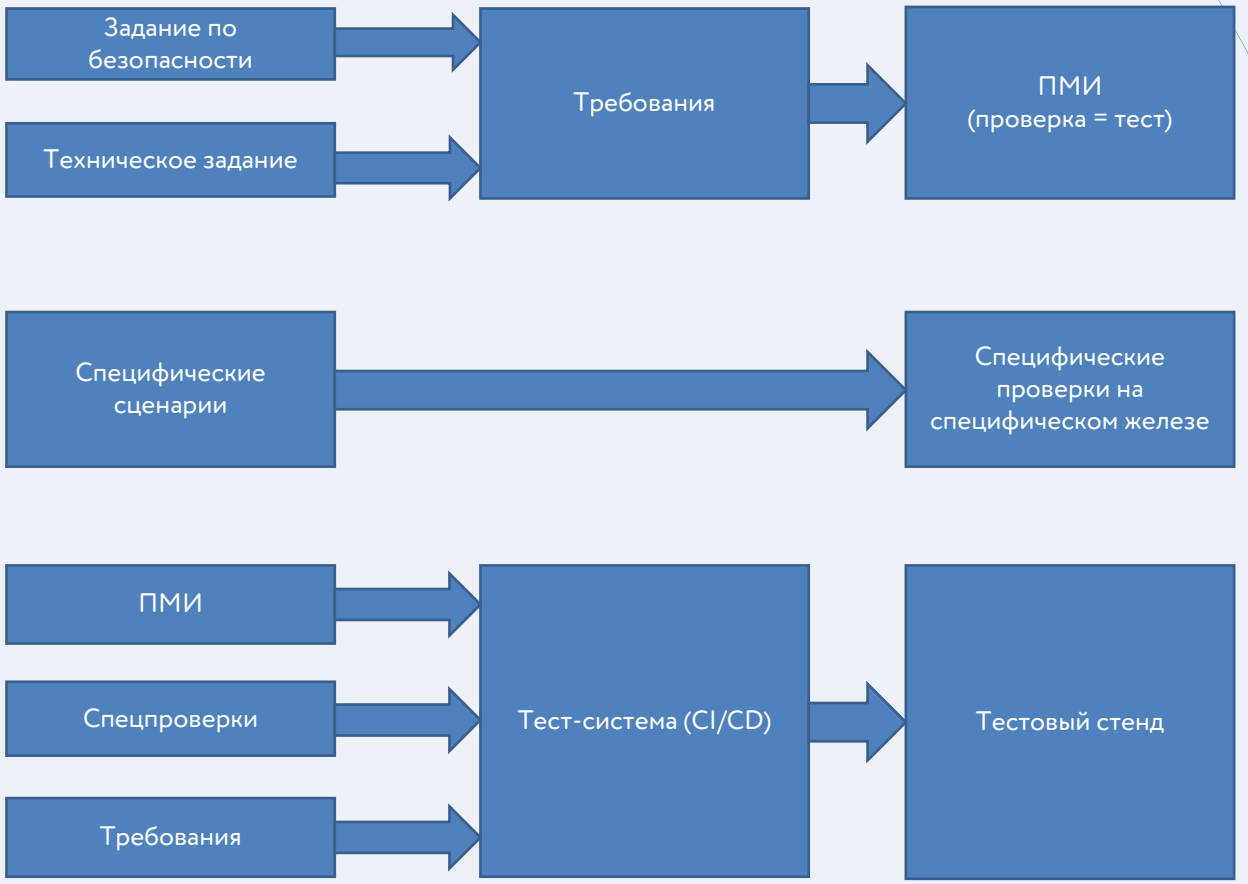
Дао здорового человека

Проблемы. ДАО.1

Тестирование

Этапы становления процесса:

1) **ПМИ + наследие QNX**
2) **Графические технологии независимо начинают требовать специфических проверок на различных архитектурах**
3) **Систематизация опыта, вырождение обобщенных ПМИ в тесты, объединение разрозненных несистемных проверок в тест-комплекты, организуется тест-стенд**



Проблемы. ДАО.1. Тестовый стенд.

Состав стенда:

Intel NUC6i3SYB (x86)
Vortex A9120 (x86)
Intel Ivy Bridge (x86)
Intel Coffee Lake (x86)
Freescale i.MX6 SABRE Smart Platform (armv7)
Atmel AT91SAM9260-EK (arm)
Orange Pi PC (armv7)
Эльбрус 4C (e2k)
Freescale P3041DS (ppcbe)
НИИСИ РАН 1890ВМ8Я «КОМДИВ64» (mipsbe)
INMYS NMS-SM-RK3568 v2 (aarch64)
Xilinx Zynq UltraScale+ ZCU106 (aarch64)
NMS-UQ7-IMX8MM (aarch64)

ДАО.2 и ДАО.3. Проблемы.

Фаззинг-тестирование? Что это?

А что вообще такое системное тестирование?

Первый этап:

1. **Их нет.**

2.Вообще...

3.Что делать?

ДАО.2 и ДАО.3. Ответы.

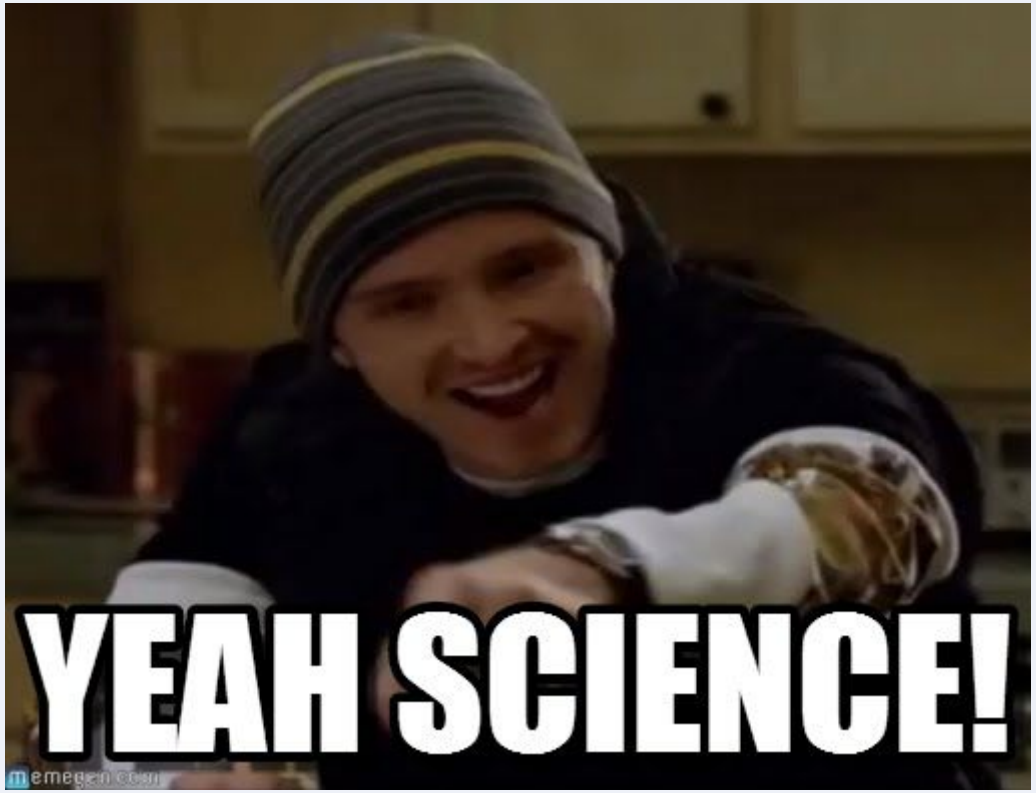
Этап 2. Поиск информации.

Публикации ИСП РАН:

https://www.ispras.ru/groups/se/publications.php

Google Академия

https://scholar.google.com/



Проблемы. ДАО.2 и ДАО.3. Реализация. Первая итерация.

Этап 3. Кросс-методики

1) Фаззинг:
   afl++ / QSym (on Linux)
2) Системное тестирование:
   valgrind/taintgrind (on Linux)
   tracelogger+gdb (on ЗОСРВ)

Можно лучше? Можно, а зачем?

Проблемы. ДАО.2 и ДАО.3. Реализация. Вторая итерация.

Этап 4. Нативные методики

1) Фаззинг:
   klee (переезд на clang on Linux)
   afl++ (on ЗОСРВ)
2) Фаззинг ядра ОС:
   syzkaller (фаззинг-кластер ядра)
3) Системное тестирование:
   Полносистемная эмуляция (в разработке)

РБПО-резюме 2025 года:

Завершено:

Состав дистрибутива:

– Подружили пакетную систему со SBOM и SCA

Процессы:

– Устаканены вопросы ответственности

– Приблизили к прозрачности

ДАО.1:

– Внедрено покрытие ядра (аналог Linux kcov)

ДАО.2:

– Фаззинг ядра

– Нативный фаззинг для ЗОСРВ

В процессе:

Процессы:

– Пройти аудит с «Фобос-НТ»

ДАО.2:

– генерация входных данных (нативная) KLEE

– полноценная фаззинг-ферма

ДАО.3:

– полносистемная эмуляция

СВД
Встраиваемые
Системы

Спасибо за внимание!

Дужак Евгений Дмитриевич

Руководитель отдела аналитики и технологий безопасности

ул. Кузнецовская, д. 19, г. Санкт-Петербург,
8-812-317-55-56
www.kpda.ru
e.duzhak@kpda.ru