CODE SCORING

DevOops 2023

Прорастить или насадить?

Практическое руководство по выращиванию политик безопасной разработки

Алексей Смирнов
Основатель CodeScoring

Мы делаем CodeScoring

Решение для безопасной работы с OpenSource & качеством кода:

Защита цепочки поставки /OSA
Композиционный анализ /SCA
Оценка качества кода /Quality Intelligence

> 5 лет обучения аналитических моделей и разработки анализаторов

> 200 млн. проанализированных OSS-проектов и библиотек

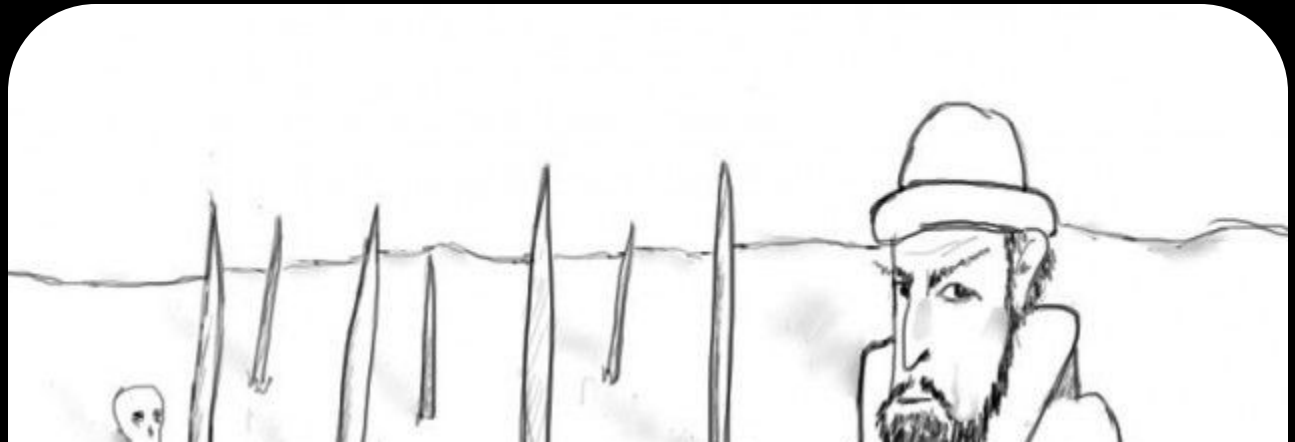
19 подключенных фидов уязвимостей, ~2000 открытых лицензий

Кто сталкивался с насаживанием политик?

Насаждение контроля

Внезапно установив жесткий выходной контроль, последуют риски:

- срыв сроков из-за обсуждений (споров)
- срыв сроков из-за устранения проблем
- последующие конфликты и «недолюбливание»



Немного про растения

Фикус — очень консервативное растение и ненавидит любые перемены.

Прямо как наша разработка.

Источник идеи:
Усадьба Багиры, www.bagira.cz

За фикусом нужно ухаживать

00	Подготовиться к уходу
01	Ознакомиться с условиями
02	Направлять к росту
03	Отсекать ненужное
04	Жестить, когда нужно
05	Пересаживать

∞	**Здоровый фикус**

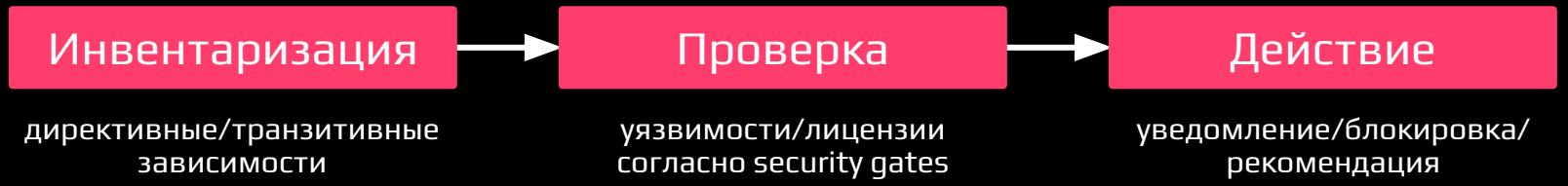
На примере работы с OpenSource

/вы можете применить и к другим доменам безопасной разработки

Композиционный анализ ПО

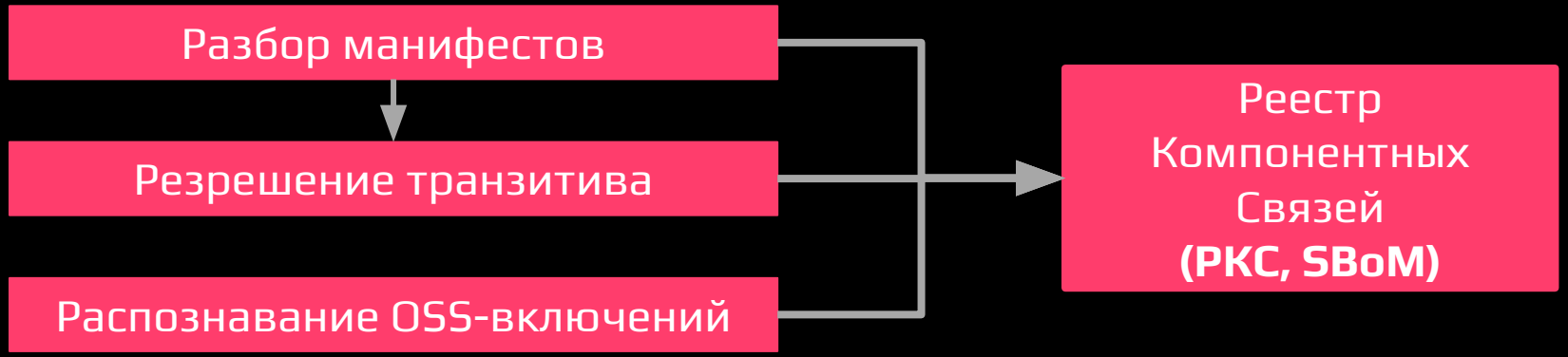
Композиционный анализ ПО (Software Composition Analysis, SCA) — процесс определения компонентов, составляющих ПО.

SCA-инструменты помогают в управлении рисками, связанными с безопасной разработкой.



Инвентаризация ПО!

Найти манифесты, разобрать их, идентифицировать компоненты.
Нужно не забыть про «включения» Open Source в ваш код.



“shazam” для кода

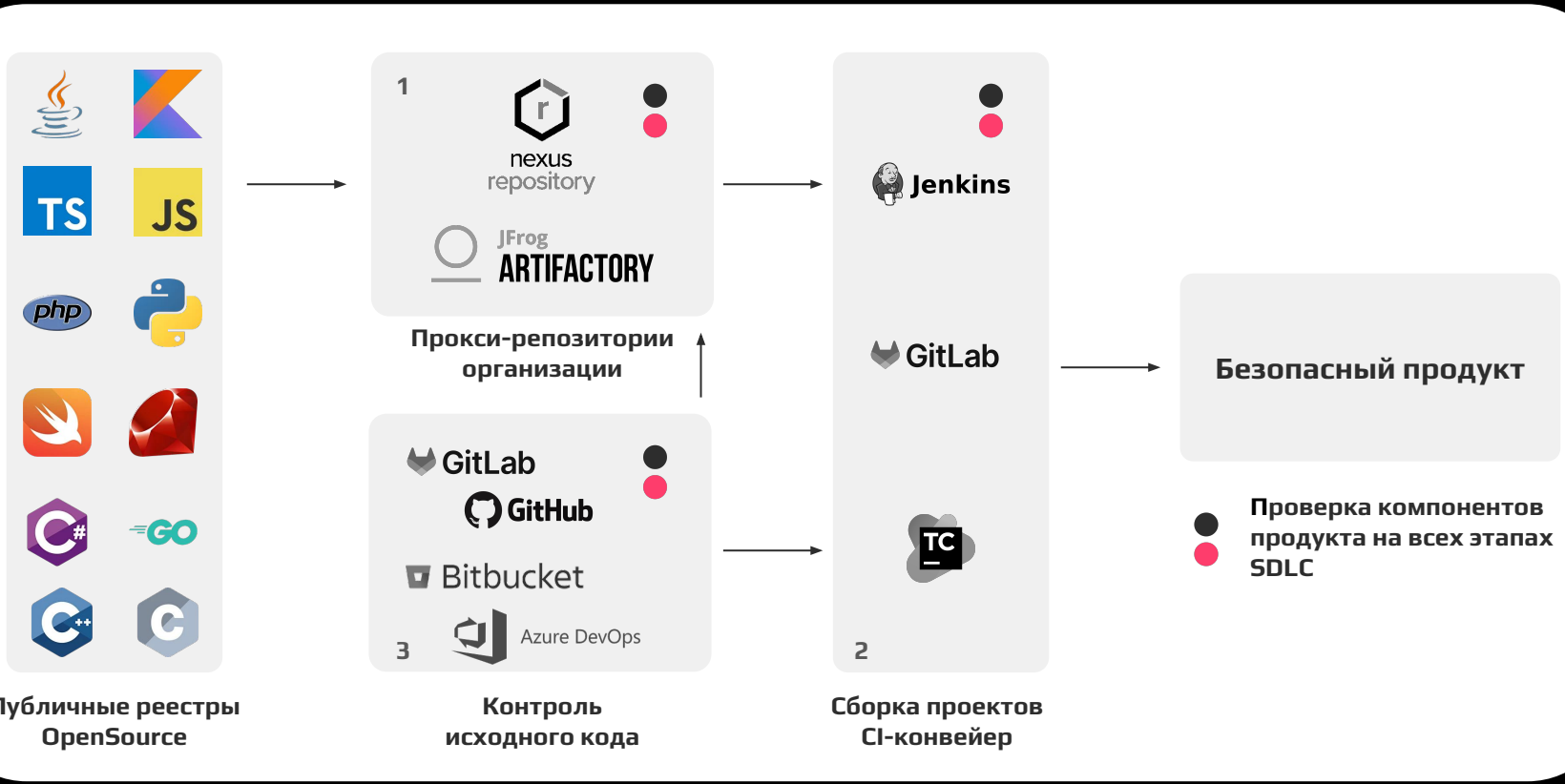
Реестр компонентных связей (SBoM)

Содержит необходимую информацию о составе ваших продуктов.

Объектная модель одного из стандартов обмена данными: CycloneDX.

Ещё есть SPDX, SWID, etc.

Общая схема внедрения проверок стороннего ПО (**разные этапы — разные политики**)



Рекомендуемая цепочка выполняемых проверок для **выпускаемых** продуктов

CODE SCORING

проверка на вредонос в прокси-репо	регулярное сканирование исходных кодов	сканирование исходных кодов в CI-конвейере	сканирование артефактов сборки (CI)	сканирование образов сборки (CI)
1	2	3	4	5
проверка **всех компонентов** попадающих в разработку на **наличие вредоносного кода** и особо опасные уязвимости, подозрительные факторы если есть — блокируем и в песочницу	проверка и **заблаговременное предупреждение** разработчиков об использовании небезопасных компонентов уведомляем ответственных: дефект-трекеры, почта	**предварительная** проверка исходных кодов поступающих на конвейер валим сборку, если нарушена политика исправление по рекомендациям	проверка того, **что вошло в сборку артефакта**, иногда в неё входит не только то, что написали разработчики валим сборку, если нарушена политика исправление по рекомендациям	проверка собранных образов на небезопасные компоненты. в них появляется знание о **системных пакетах** валим сборку, если нарушена политика исправление по рекомендациям
	фиксация SBoM	**фиксация SBoM**	**фиксация SBoM**	**фиксация SBoM**
OSS firewalling	**design**	**pre-build**	**build**	**build**

Рекомендуемая цепочка выполняемых проверок для **выпускаемых** продуктов

CODE SCORING

проверка на вредонос в прокси-репо	регулярное сканирование исходных кодов	сканирование исходных кодов в CI-конвейере	сканирование артефактов сборки (CI)	сканирование образов сборки (CI)
1	2	3	4	5
проверка **всех компонентов** попадающих в разработку на **наличие вредоносного кода** и особо опасные уязвимости, подозрительные факторы если есть — блокируем и в песочницу	проверка и **заблаговременное предупреждение** разработчиков об использовании небезопасных компонентов уведомляем ответственных: дефект-трекеры, почта	**предварительная** проверка исходных кодов поступающих на конвейер валим сборку, если нарушена политика исправление по рекомендациям	проверка того, **что вошло в сборку артефакта**, иногда в неё входит не только то, что написали разработчики валим сборку, если нарушена политика исправление по рекомендациям	проверка собранных образов на небезопасные компоненты. в них появляется знание о **системных пакетах** валим сборку, если нарушена политика исправление по рекомендациям
	фиксация SBoM	**фиксация SBoM**	**фиксация SBoM**	**фиксация SBoM**
OSS firewalling	**design**	**pre-build**	**build**	**build**

Рекомендуемая цепочка выполняемых проверок для **выпускаемых** продуктов

CODE SCORING

проверка на вредонос в прокси-репо	регулярное сканирование исходных кодов	сканирование исходных кодов в CI-конвейере	сканирование артефактов сборки (CI)	сканирование образов сборки (CI)
1	2	3	4	5
проверка **всех компонентов** попадающих в разработку на **наличие вредоносного кода** и особо опасные уязвимости, подозрительные факторы если есть — блокируем и в песочницу	проверка и **заблаговременное предупреждение** разработчиков об использовании небезопасных компонентов уведомляем ответственных: дефект-трекеры, почта	**предварительная** проверка исходных кодов поступающих на конвейер валим сборку, если нарушена политика исправление по рекомендациям	проверка того, **что вошло в сборку артефакта**, иногда в неё входит не только то, что написали разработчики валим сборку, если нарушена политика исправление по рекомендациям	проверка собранных образов на небезопасные компоненты. в них появляется знание о **системных пакетах** валим сборку, если нарушена политика исправление по рекомендациям
	фиксация SBoM	**фиксация SBoM**	**фиксация SBoM**	**фиксация SBoM**
OSS firewalling	**design**	**pre-build**	**build**	**build**

Рекомендуемая цепочка выполняемых проверок для **выпускаемых** продуктов

CODE SCORING

проверка на вредонос в прокси-репо	регулярное сканирование исходных кодов	сканирование исходных кодов в CI-конвейере	сканирование артефактов сборки (CI)	сканирование образов сборки (CI)
1	2	3	4	5
проверка **всех компонентов** попадающих в разработку на **наличие вредоносного кода** и особо опасные уязвимости, подозрительные факторы если есть — блокируем и в песочницу	проверка и **заблаговременное предупреждение** разработчиков об использовании небезопасных компонентов уведомляем ответственных: дефект-трекеры, почта	**предварительная** проверка исходных кодов поступающих на конвейер валим сборку, если нарушена политика исправление по рекомендациям	проверка того, **что вошло в сборку артефакта**, иногда в неё входит не только то, что написали разработчики валим сборку, если нарушена политика исправление по рекомендациям	проверка собранных образов на небезопасные компоненты. в них появляется знание о **системных пакетах** валим сборку, если нарушена политика исправление по рекомендациям
	фиксация SBoM	**фиксация SBoM**	**фиксация SBoM**	**фиксация SBoM**
OSS firewalling	**design**	**pre-build**	**build**	**build**

Рекомендуемая цепочка выполняемых проверок для **выпускаемых** продуктов

CODE SCORING

проверка на вредонос в прокси-репо	регулярное сканирование исходных кодов	сканирование исходных кодов в CI-конвейере	сканирование артефактов сборки (CI)	сканирование образов сборки (CI)
1	2	3	4	5
проверка **всех компонентов** попадающих в разработку на **наличие вредоносного кода** и особо опасные уязвимости, подозрительные факторы если есть — блокируем и в песочницу	проверка и **заблаговременное предупреждение** разработчиков об использовании небезопасных компонентов уведомляем ответственных: дефект-трекеры, почта	**предварительная** проверка исходных кодов поступающих на конвейер валим сборку, если нарушена политика исправление по рекомендациям	проверка того, **что вошло в сборку артефакта**, иногда в неё входит не только то, что написали разработчики валим сборку, если нарушена политика исправление по рекомендациям	проверка собранных образов на небезопасные компоненты. в них появляется знание о **системных пакетах** валим сборку, если нарушена политика исправление по рекомендациям
	фиксация SBoM	**фиксация SBoM**	**фиксация SBoM**	**фиксация SBoM**
OSS firewalling	**design**	**pre-build**	**build**	**build**

Рекомендуемая цепочка выполняемых проверок для **выпускаемых** продуктов

CODE SCORING

проверка на вредонос в прокси-репо	регулярное сканирование исходных кодов	сканирование исходных кодов в CI-конвейере	сканирование артефактов сборки (CI)	сканирование образов сборки (CI)
1	2	3	4	5
проверка **всех компонентов** попадающих в разработку на **наличие вредоносного кода** и особо опасные уязвимости, подозрительные факторы если есть — блокируем и в песочницу	проверка и **заблаговременное предупреждение** разработчиков об использовании небезопасных компонентов уведомляем ответственных: дефект-трекеры, почта	**предварительная** проверка исходных кодов поступающих на конвейер валим сборку, если нарушена политика исправление по рекомендациям	проверка того, **что вошло в сборку артефакта**, иногда в неё входит не только то, что написали разработчики валим сборку, если нарушена политика исправление по рекомендациям	проверка собранных образов на небезопасные компоненты. в них появляется знание о **системных пакетах** валим сборку, если нарушена политика исправление по рекомендациям
	фиксация SBoM	**фиксация SBoM**	**фиксация SBoM**	**фиксация SBoM**
OSS firewalling	**design**	**pre-build**	**build**	**build**

Рекомендуемая цепочка выполняемых проверок для **выпущенных** продуктов

[рекуррентные проверки SBoM]

6

регулярная перепроверка компонентов ранее выпущенных версий продуктов

в них уязвимости не появляются,
а обнаруживаются!

нашли проблему — отзывная кампания и выпуск коррекции

перепроверка SBoM

operations

[вывод версии продукта из эксплуатации]

7

выпущенные продукты заканчивают свою жизнь, вместе с ними и SBoM-спецификация

проводим ревью компонентного состава и, при необходимости, — **выводим компоненты из эксплуатации**

архивация SBoM

post-operations

Ключ к успеху — отслеживание качества безопасности в ранее выпущенных релизах и управление процессом вывода из эксплуатации.

Рекомендуемая цепочка выполняемых проверок для **выпущенных** продуктов

рекуррентные проверки SBoM

6

регулярная перепроверка компонентов ранее выпущенных версий продуктов

в них уязвимости не появляются,
а обнаруживаются!

нашли проблему — отзывная кампания и выпуск коррекции

перепроверка SBoM

operations

вывод версии продукта из эксплуатации

7

выпущенные продукты заканчивают свою жизнь, вместе с ними и SBoM-спецификация

проводим ревью компонентного состава и, при необходимости, — **выводим компоненты из эксплуатации**

архивация SBoM

post-operations

Ключ к успеху — отслеживание качества безопасности в ранее выпущенных релизах и управление процессом вывода из эксплуатации.

Рекомендуемая цепочка выполняемых проверок для **выпущенных** продуктов

рекуррентные проверки SBoM

вывод версии продукта из эксплуатации

6

7

регулярная перепроверка компонентов ранее выпущенных версий продуктов

в них уязвимости не появляются, а обнаруживаются!

нашли проблему — отзывная кампания и выпуск коррекции

выпущенные продукты заканчивают свою жизнь, вместе с ними и SBoM-спецификация

проводим ревью компонентного состава и, при необходимости, — **выводим компоненты из эксплуатации**

перепроверка SBoM

архивация SBoM

operations

post-operations

Ключ к успеху — отслеживание качества безопасности в ранее выпущенных релизах и управление процессом вывода из эксплуатации.

Выращивание политик

/последовательные действия «садовника»

00. Подготовительный этап

00	**Подготовительный этап**
01	Инвентаризация
02	Направление к росту
03	Отсечение ненужного
04	Жестить, когда нужно
05	Пересаживать

Определить:

- объекты оценки
- команды-разработчики
- технологии & процессы
- целевые метрики качества (параметры политик)
- выбрать инструменты анализа

Ответить на вопрос:

- сколько у вас времени до цели?

01. Инвентаризация

00	Подготовительный этап
01	**Инвентаризация**
02	Направление к росту
03	Отсечение ненужного
04	Жестить, когда нужно
05	Пересаживать

1. провести предварительный аудит компонентной базы
2. проверить корректность аудита
3. скорректировать состав технологий и свое понимание о текущих процессах

02. Направление к росту

00	Подготовительный этап
01	Инвентаризация
02	**Направление к росту**
03	Отсечение ненужного
04	Жестить, когда нужно
05	Пересаживать

1. уведомить о результатах аудита
   => отследить реакцию
   (проверка осознанности)
2. N раз отправить отчеты
   о ~~не~~изменяемости ситуации
   => отследить реакцию
3. Показать простые точечные
   проблемы, предложить решить
   => отследить реакцию

02. Направление к росту /пример #1

Не хотим слишком новых/старых пакетов?

=> смотрим сколько таких, строим политики ступеньками

Например:

- фильтруем всё моложе 3 / 7 / 14 / 30 дней
- усиляем политику ~помесячно, чтобы привыкали

02. Направление к росту /пример #2

Хотим подтолкнуть к исправлению критических проблем?

1. Проверяем инвентарь: выбираем с чего начать
2. Короткое время уведомляем об уязвимостях
   => следим за реакцией
3. Блокируем сборку только на уязвимостях которые понятны и чинятся просто:
   directive + critical + has_exploit + has_fixed_version
4. Усложняем условия и следим как прорастает культура

03. Отсечение ненужного

00	Подготовительный этап
01	Инвентаризация
02	Направление к росту
03	**Отсечение ненужного**
04	Жестить, когда нужно
05	Пересаживать

1. Предупредить, что будете блокировать
2. Поставить простую политику с широким диапазоном значений
3. Сужать ворота безопасности со временем по мере привыкания и приведения кодовой базы в порядок

03. Отсечение ненужного /пример

Избавляемся от зависимостей, которые используются мало, а тянут много (и съедают время на триаже)

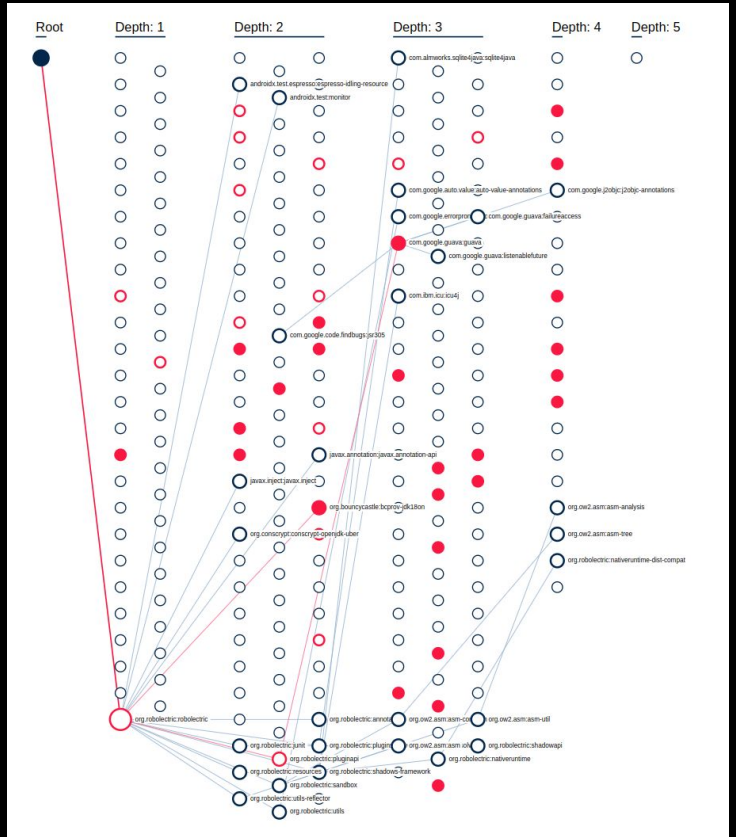
1. Выделяем самые яркие и нетрудоемкие кейсы
   => отдаем в разработку

2. Отслеживаем скорость исправлений
   => готовим следующую пачку (разумную)

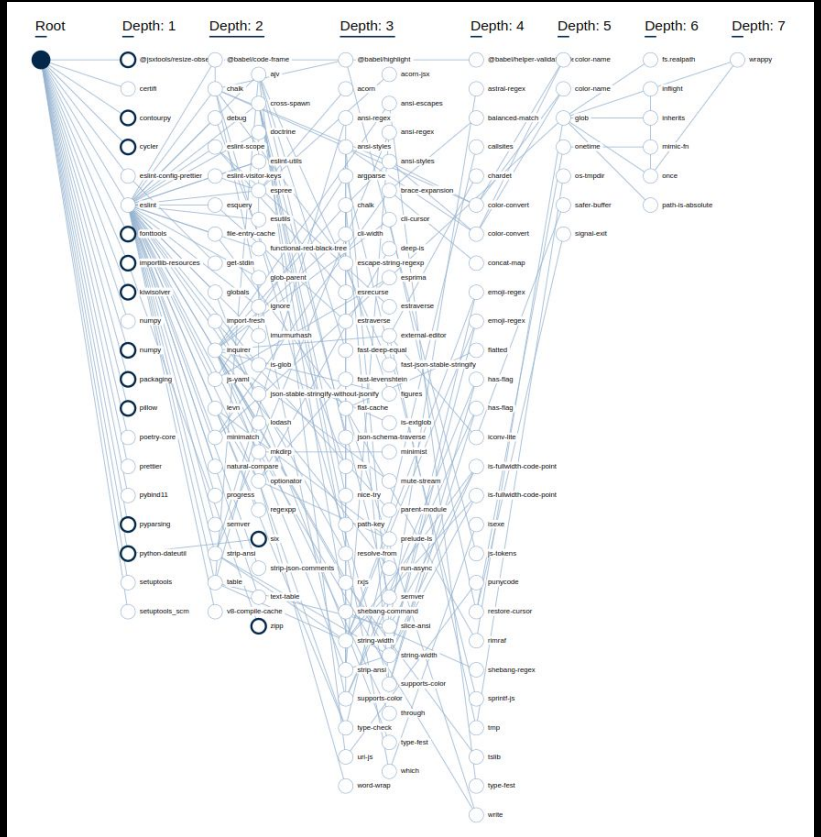
3. Ставим на регулярное уведомление о проблематике на сборке

03. Отсечение ненужного /пример-картинка

#1. Пакет тянет много уязвимого



#2. Великое множество dev-пакетов



04. Жестить, когда нужно

00	Подготовительный этап
01	Инвентаризация
02	Направление к росту
03	Отсечение ненужного
04	**Жестить, когда нужно**
05	Пересаживать

- всегда помним про целевые параметры качества
- при грубых нарушениях установленных политик не стесняемся **на примерах** объяснять зачем всё это нужно
- не пропускаем релизы пока не исправлено

05. Пересаживать

00	Подготовительный этап
01	Инвентаризация
02	Направление к росту
03	Отсечение ненужного
04	Жестить, когда нужно
05	**Пересаживать**

Как правило, разработка растет:

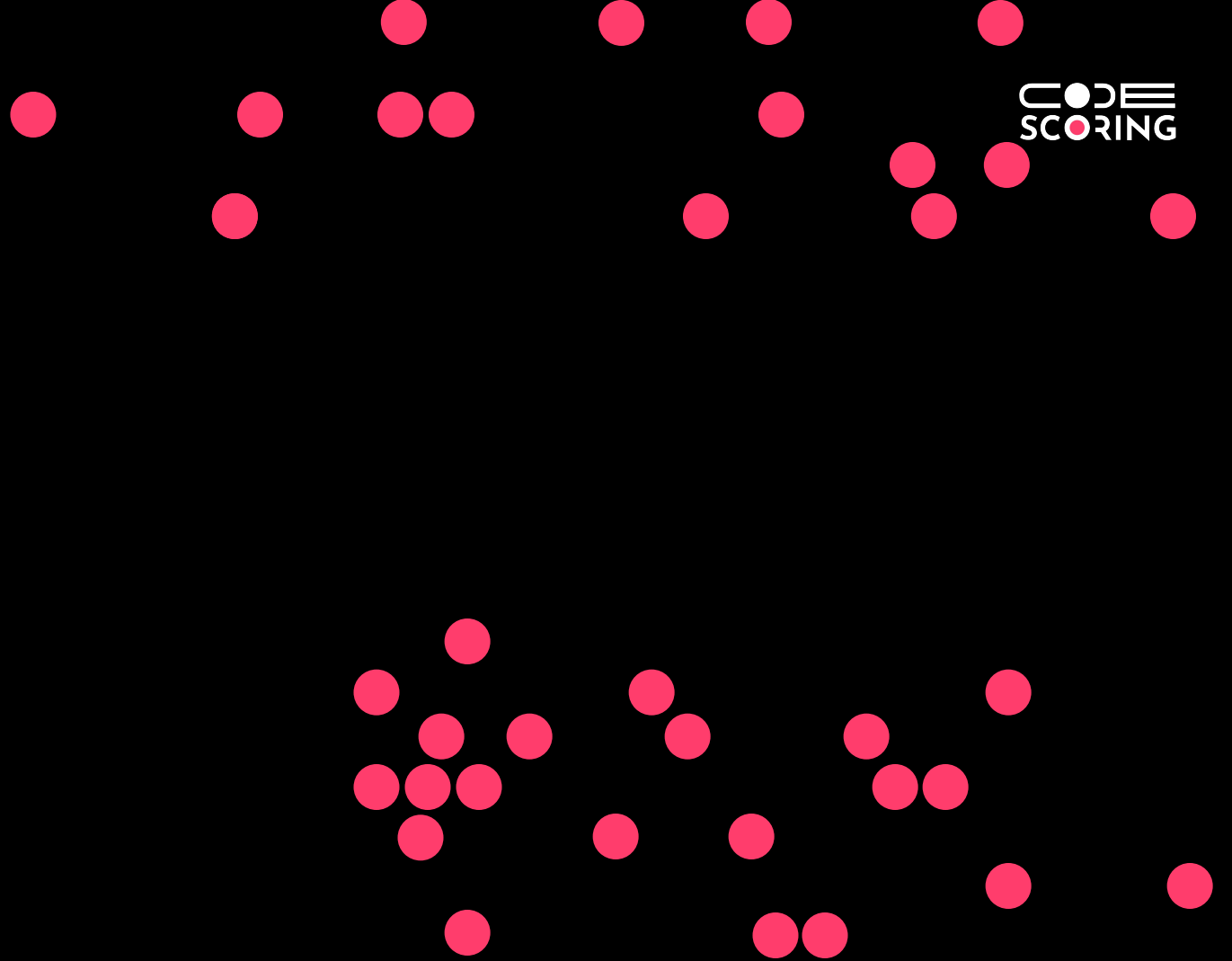
- адаптируем политики: обновление, разделение, масштабирование
- адаптируем процессы
- обновляем инструменты

∞. Повторять и размножать

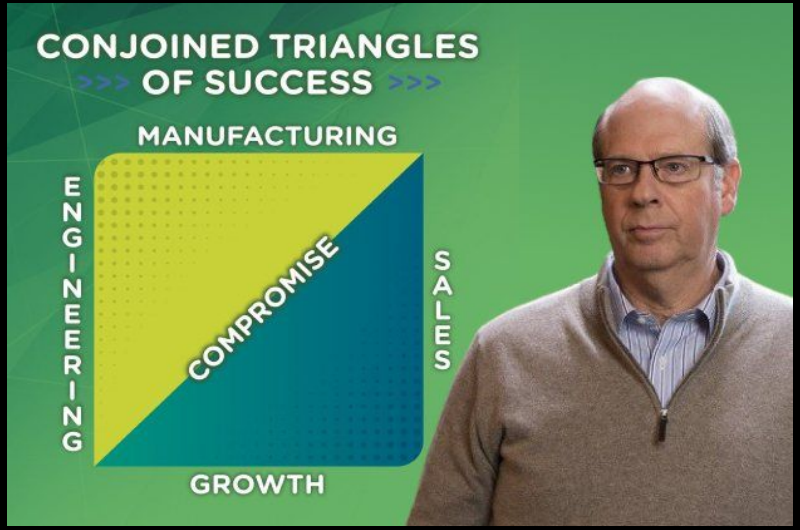
00	Подготовительный этап
01	Инвентаризация
02	Направление к росту
03	Отсечение ненужного
04	Жестить, когда нужно
05	Пересаживать

Выводы

/так о чем это всё?

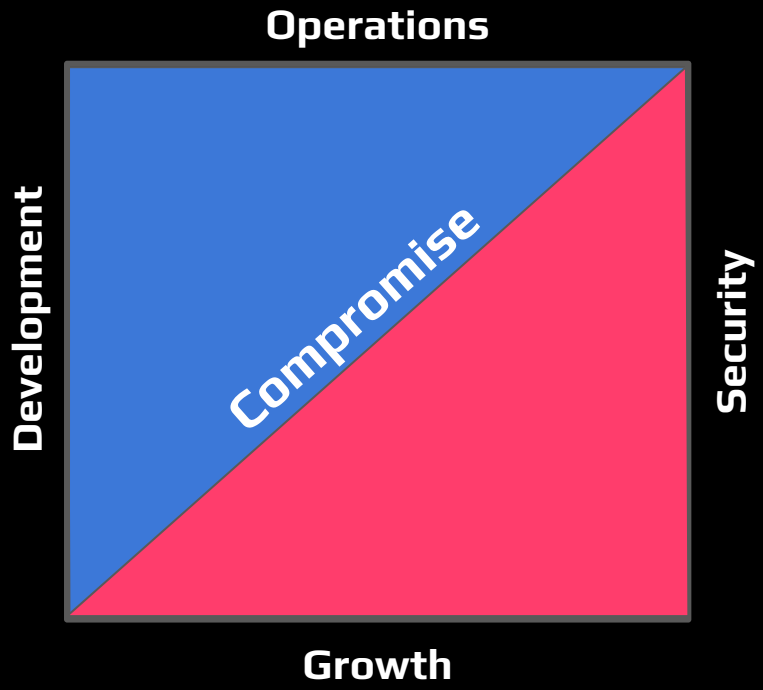


Объединенные треугольники успеха



Conjoined triangles of succes (c) Silicon Valley

Объединенные треугольники успеха AppSec



Главное — выдержать баланс адекватности

Насадить или прорастить политики

¯_(ツ)_/¯

Насадить политики
1. Быстрый эффект
2. Пропуск важного в политиках
3. Борьба с последствиями, непереносимость знаний

Прорастить политики
1. Дольше внедрять
2. Точность выше
3. Надежность и **культура** => **экономия времени**

Так насадить или прорастить? ;)

Спасибо за внимание!

DevOops 2023

Алексей Смирнов,
основатель CodeScoring,
решения композиционного
анализа (SCA)

alexey@codescoring.ru

@alsmirn — докладчик

@codescoring — новости продукта

О — Образование:

youtube.com/@codescoring