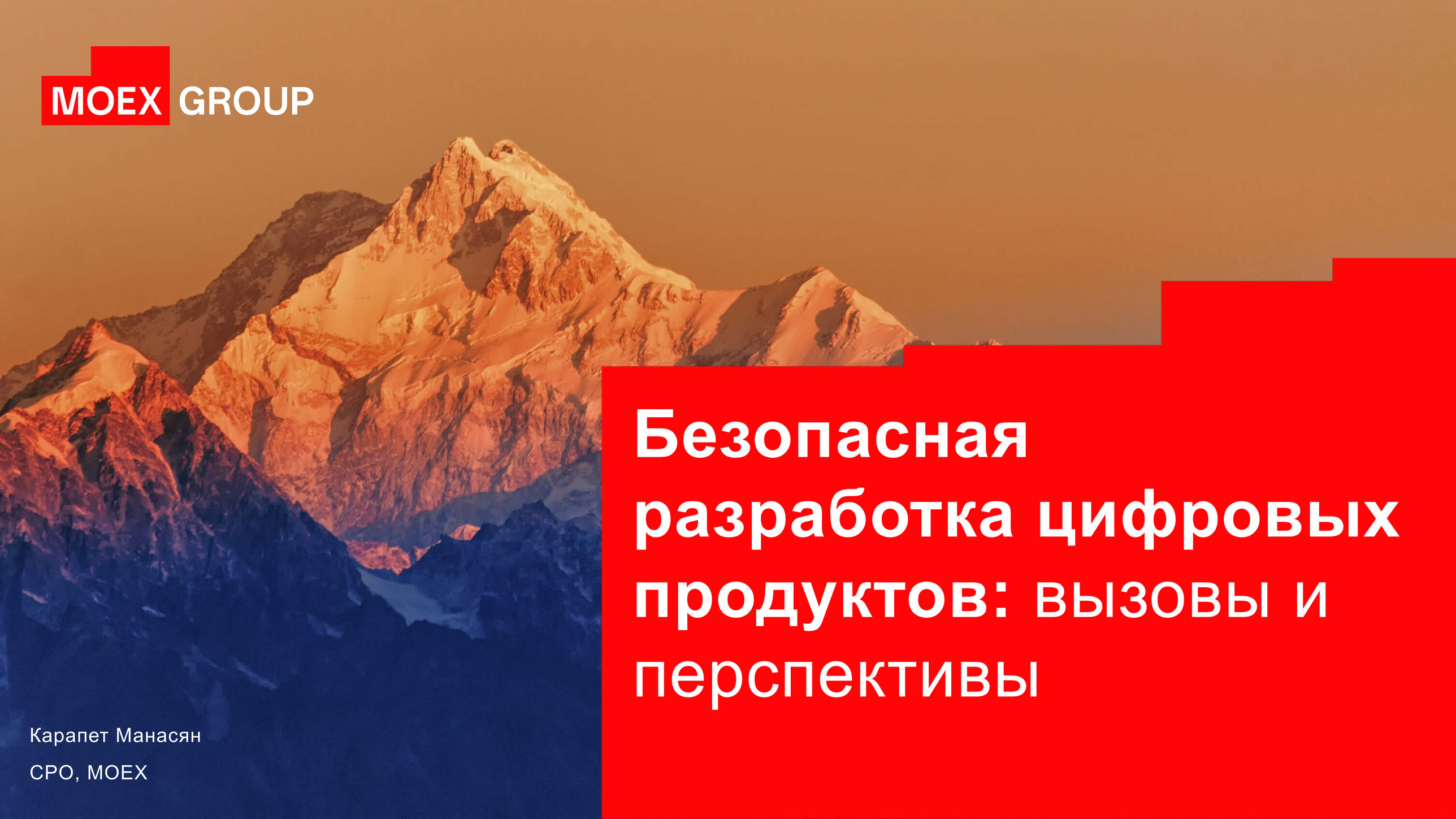
MOEX GROUP



Безопасная разработка цифровых продуктов: вызовы и перспективы

Карапет Манасян

CPO, MOEX

Основные вызовы в безопасной разработке в финтехе сегодня

Безопасность как часть ДНК всей разработки

Open-source vs closed source

Проверка внешних библиотек и OSS на уязвимости

Нехватка квалифицированных кадров

Влияние на TTM

Низкий уровень развития профильных сообществ

Основные вызовы в безопасной разработке в финтехе сегодня

Безопасность как часть ДНК всей разработки

Open-source vs closed source

Проверка внешних библиотек и OSS на уязвимости

Нехватка квалифицированных кадров

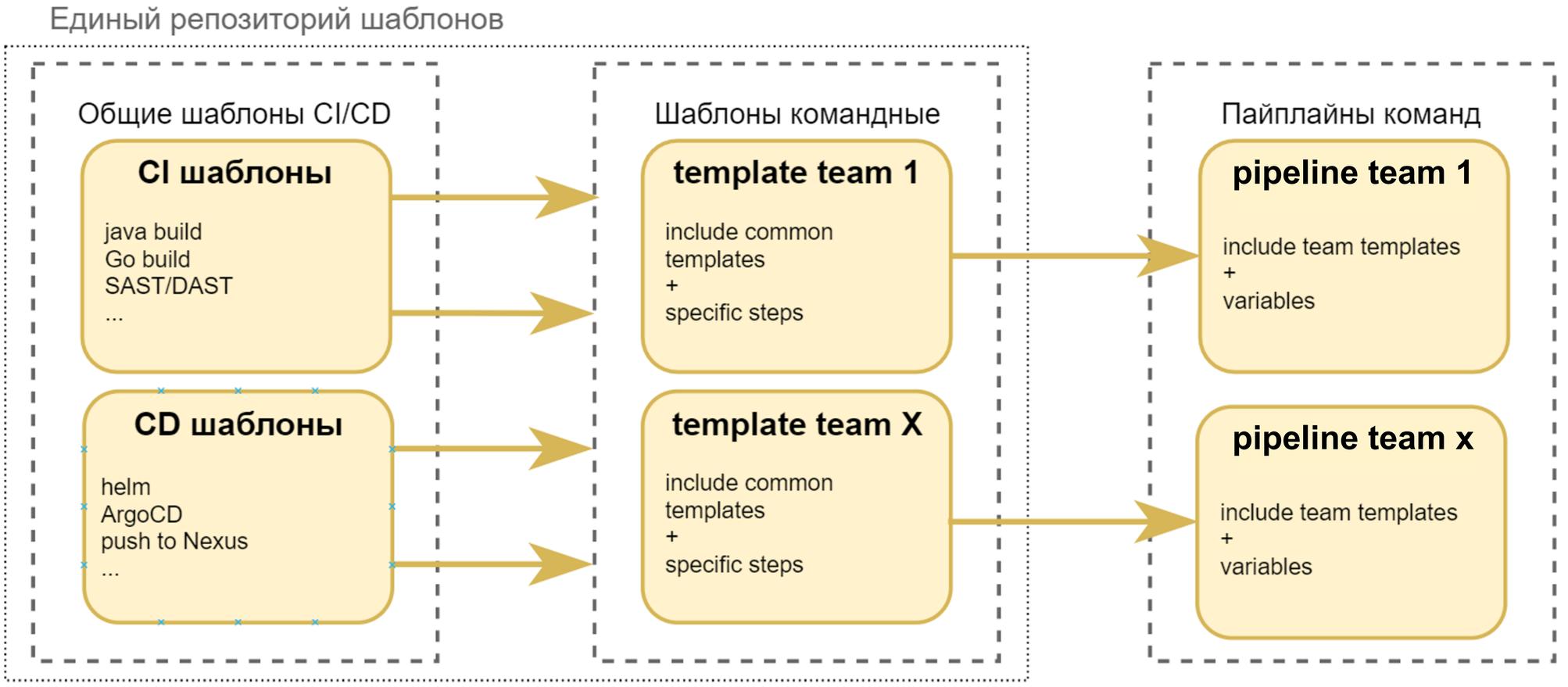
Влияние на TTM

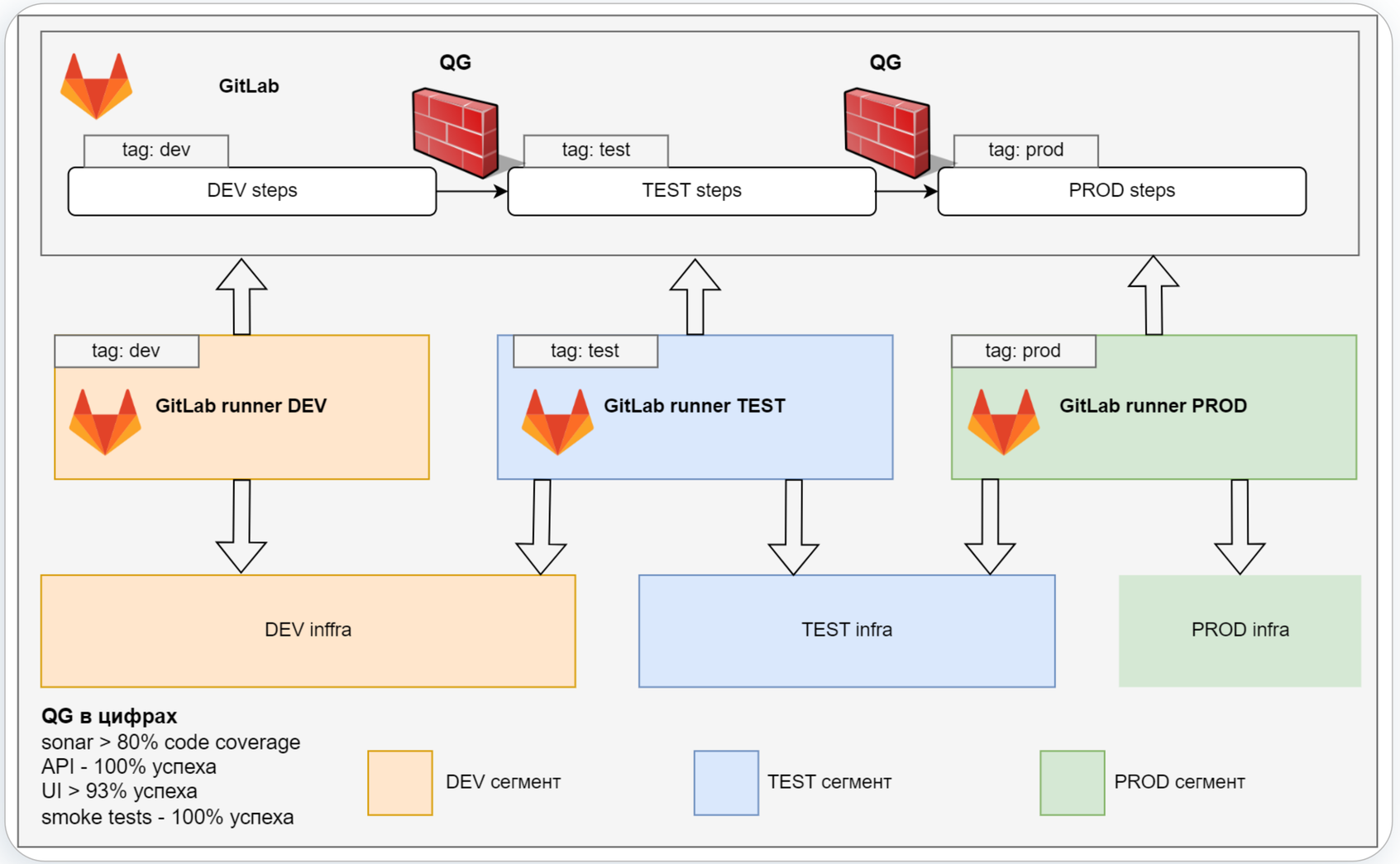
Низкий уровень развития профильных сообществ

Больше вопросов = правильная архитектура

Какие последствия компрометации?	Работает с сетью? С интернет?	Open source или бинарник?	Взаимодействует с пользователями?
Важная цель?	Обрабатывает внешние данные?	Доверенный разработчик?	Взаимодействует с СКЗИ?
Использует API или работает само?	Легко изолировать?	Был аудит/сертификация?	Есть подходящие средства анализа?
Обрабатывает защищаемую информацию?	Доверенный канал получения?	Уже применяется в компании?	Есть встроенные средства безопасности?
Есть требования регулятора?	Регулярно обновляется?	Есть аналоги или замены?	Много зависимостей?

Нельзя сделать универсальную методику


Единый репозиторий шаблонов
Общие шаблоны CI/CD
CI шаблоны
java build
Go build
SAST/DAST
...
CD шаблоны
helm
ArgoCD
push to Nexus
...
Шаблоны командные
template team 1
include common
templates
+
specific steps
template team X
include common
templates
+
specific steps
Пайплайны команд
pipeline team 1
include team templates
+
variables
pipeline team x
include team templates
+
variables

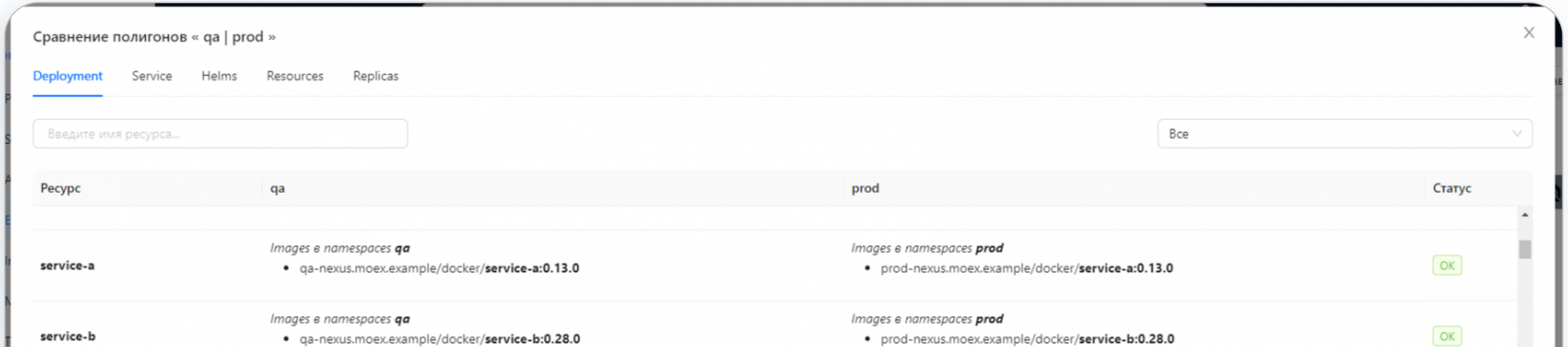


QGs/SGs (quality and security gates) - проверки и тесты на различных этапах конвейера. Такие как SAST, юнит, UI, API, регрессионные, контрактные, ручные, интеграционные и т.д. Несоответствие значения QGs/SGs эталонному блокирует конвейер до исправления ошибок.

Что входит в SGs?

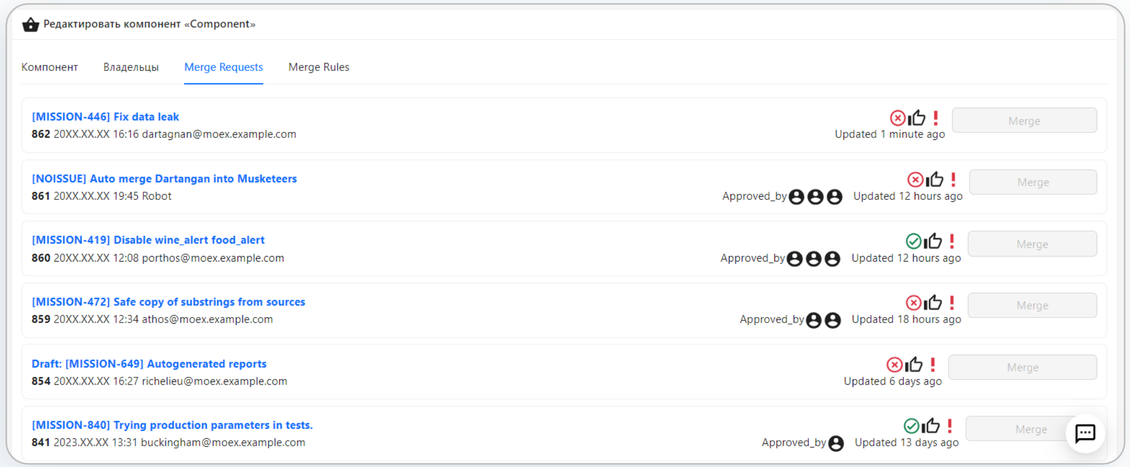
SCM и контроль версий

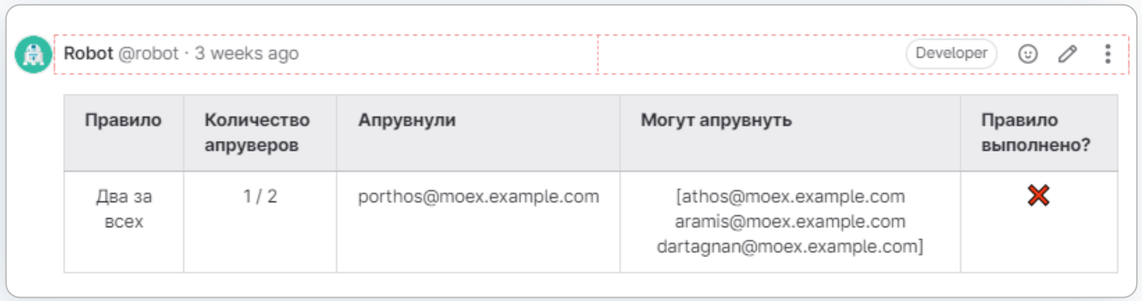
1. Единый инстанс хранения исходного кода
2. Использование GitOps-ориентированных инструментов (пр. ArgoCD)
3. Запуск процессов проверки на MR
4. Использование практик автодокументирования



Что входит в SGs?

Сборка и CI





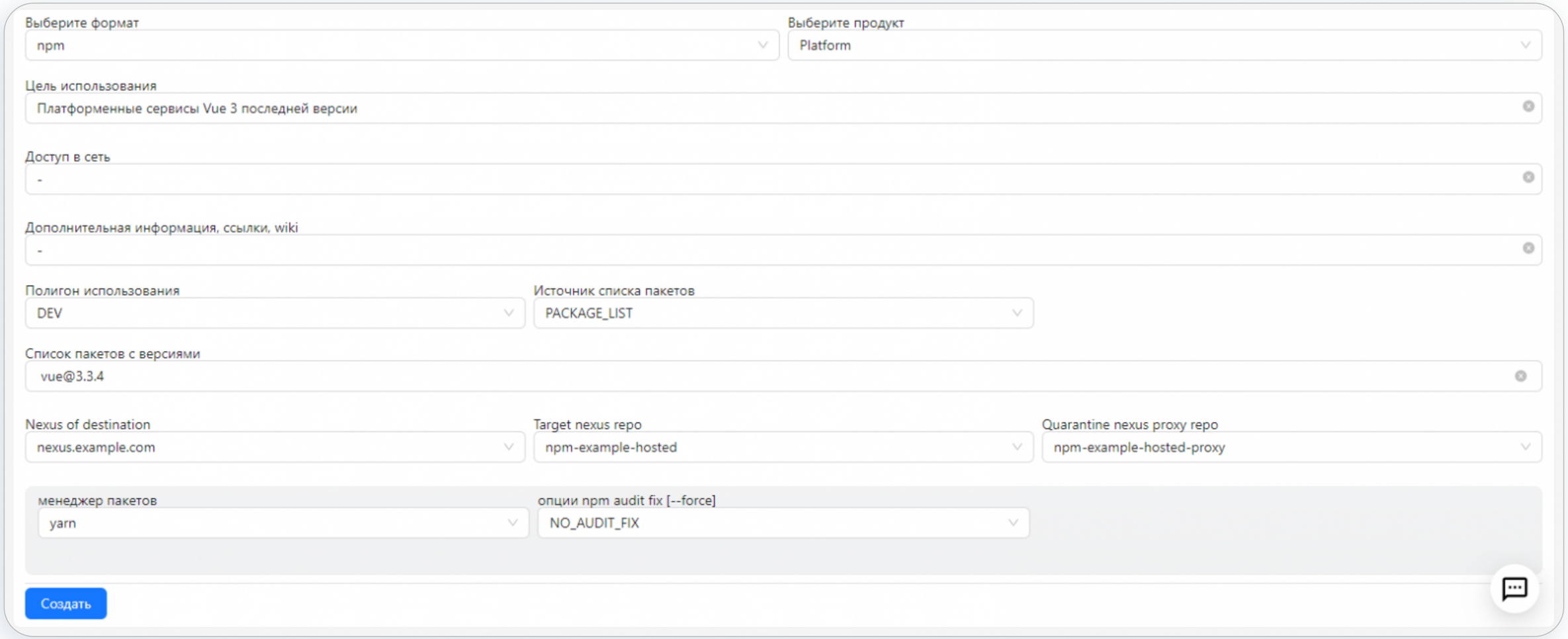
1. Разделение окружений (dev/test/prod)
2. Инструменты SAST с четкими метриками прохождения для команд (~~default~~)
3. Функционал MR-approvers в DOP
4. Инструменты генерации SBOM и SCA-анализ
5. Подпись артефактов

Что входит в SGs?

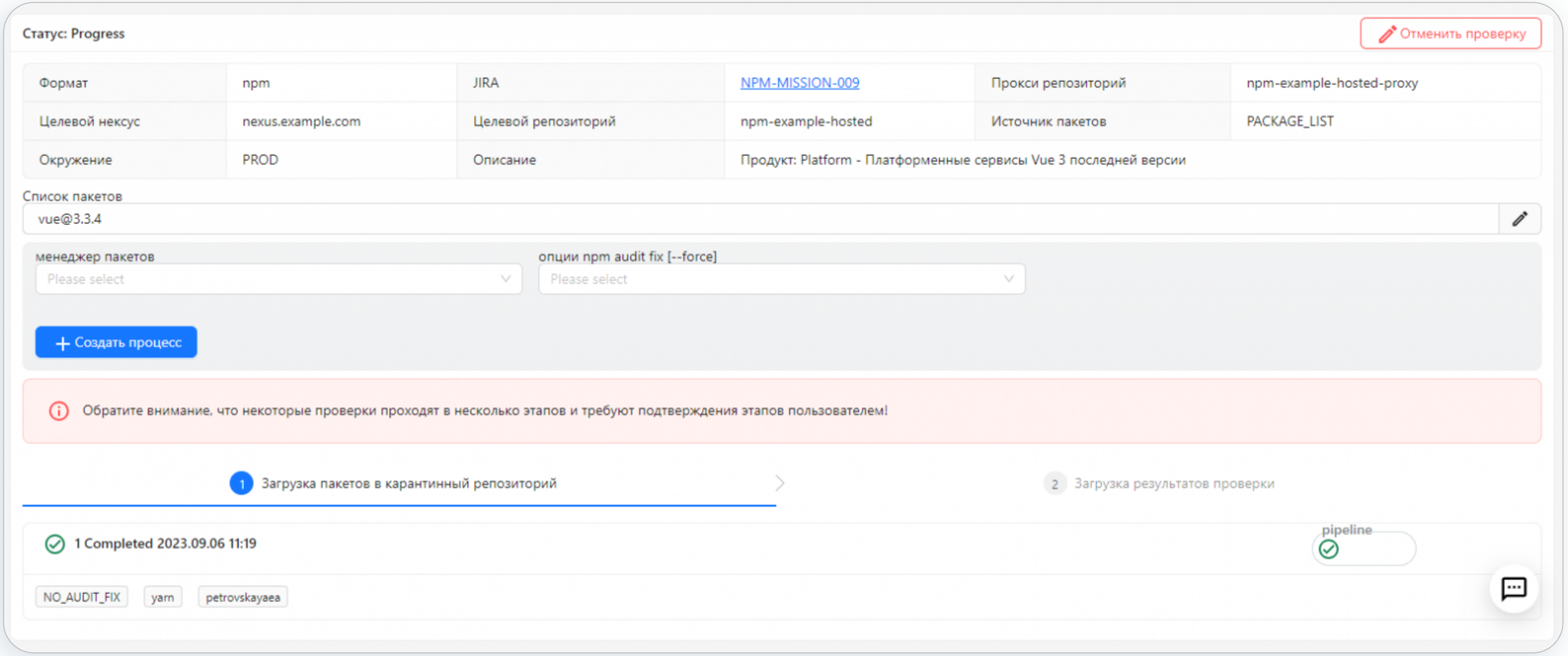
Хранение артефактов и библиотек

1. Миграция артефактов только через изолированные контуры Dev -> Test -> Prod
2. Разделение инстансов хранения артефактов по уровняем безопасности
3. Процесс переноса во внутренний контур внешних библиотек
4. Мониторинг внешних репозиториев

Как это выглядит в DOP

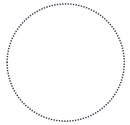


Как это выглядит в DOP



Open-source тоже разный

Высокие риски

Модель	Полный open-source	Skinny	Thin	Lean	Think	Proprietary
Визуализация						
Состав	100% OSS 0% закрытое ПО	~90% OSS ~10% закрытое ПО	~70% OSS ~30% закрытое ПО	~50% OSS ~50% закрытое ПО	~10% OSS ~90% закрытое ПО	0% OSS 100% закрытое ПО
Модель доставки	On-prem	On-prem	On-prem / cloud	On-prem / cloud	On-prem / cloud	On-prem / cloud
Уровень контроля кода	Сверхмаксимальный	Максимальный	Средний	Низкий	Минимальный	Сверхминимальный

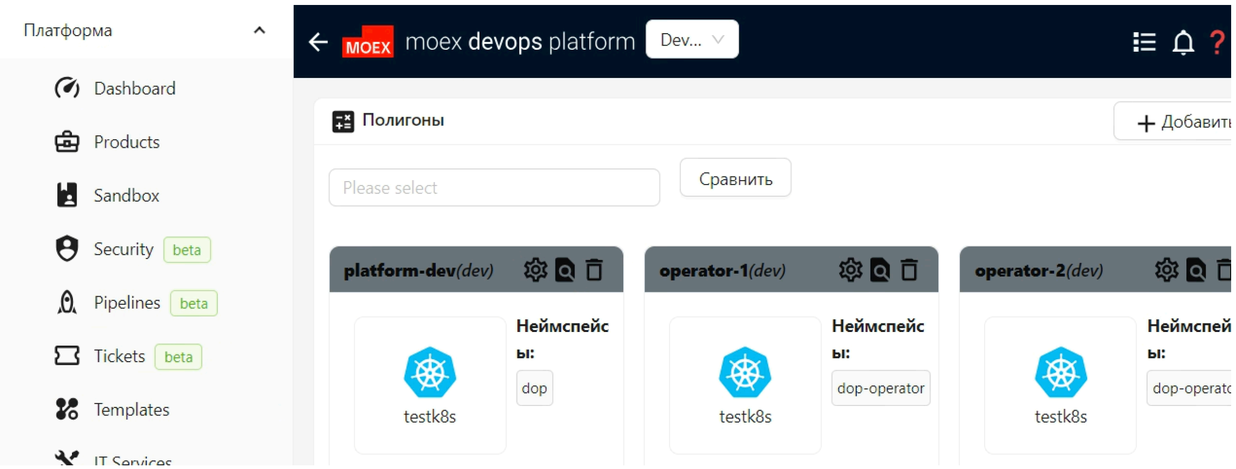
Что входит в SGs?

Тестирование и анализ

1. Система управление требованиями и тестами
2. Линтеры и SAST-ы начиная с ПК разработчика, а не только в CI/CD
3. Fuzzing тестирование
4. Тестирование изолированных окружений в песочнице
5. CDD или разработка на базе контрактов

Что входит в SGs?

Доставка, CD и мониторинг



1. Контроль целостности конфигураций и YAMLиков
2. Управление окружениями на базе MR в deploy-maps
3. Конфигурация ≠ секреты: стендо-зависимые переменные отдельно, чувствительные данные отдельно
4. Мониторинг аномалий и изменений в окружениях
5. Обеспечение полной видимости и observability всех контуров

Довериться конвейеру

Постоянно проверять гипотезы относительно devsecops

Основные вызовы в безопасной разработке в финтехе сегодня

Безопасность как часть ДНК всей разработки

Open-source vs closed source

Проверка внешних библиотек и OSS на уязвимости

Нехватка квалифицированных кадров

Влияние на TTM

Низкий уровень развития профильных сообществ

Команды и культура

Внутреннее обучение и митапы

1. ИБ + ИТ = ❤️
2. Проведение аудита и анализа
3. Внутренние курсы для разработчиков и инженеров
4. Внутренние сообщества по DevOps и DevSecOps: единая точка правды
5. Выступления внешних спикеров во внутренних митапах
6. Создание понятных how-to

Команды и культура

ИТ-сообщества

1. Создание сообщества FinDevSecOps: community по безопасной разработке в финсекторе
2. Создание совместных проектов и how-to
3. Унифицированная имплементация решений и практик
4. Шаринг знаний с помощью митапов

Выводы

- SGs не должны мешать команде
- Централизованный CI/CD-flow и gitops – наше все!
- Чем больше вопрос относительно дальнейшего использования open-source компонента, тем ниже риски связанные с ним
- Внедрил SG – определи метрики!
- Компетенции devsecops должны быть распределены на всех участников команды
- ИБ + ИТ = ❤️: безопасность как часть ДНК всего цикла разработки, security by design

DEVSECOPS НАЧИНАЕТСЯ ЗДЕСЬ