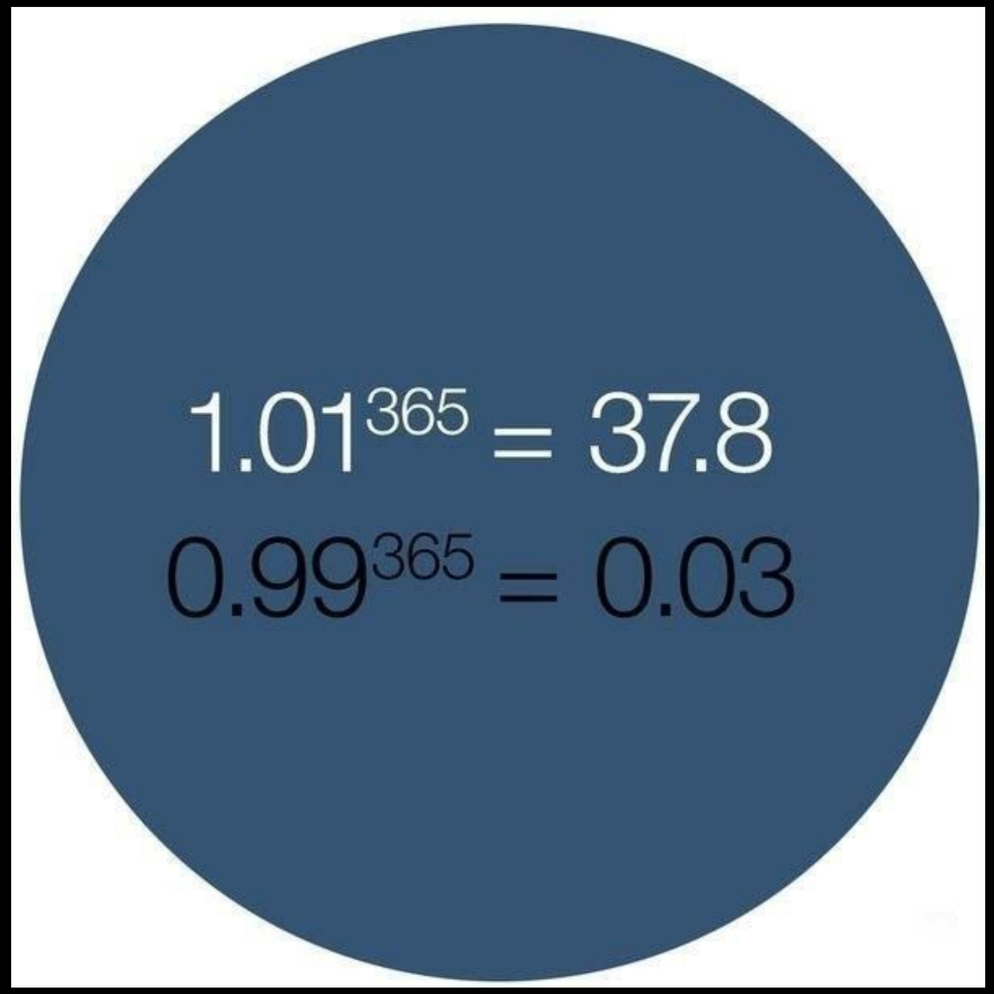
avito.tech Москва 2025

Час пик: постройка системы проверки API, которая успевает за частыми релизами



Александр Трифанов

Тех и этих лид команды Application Security

Whoami

Александр Трифанов
тех и этих лид

- Замкадыш из Новосибирска
- 8+ лет занимаюсь тестированием на проникновение и разработкой решений для продуктовой безопасности
- Неравнодушен к реверс-инжинирингу, эксплуатации бинарных уязвимостей и языку Rust

hostname

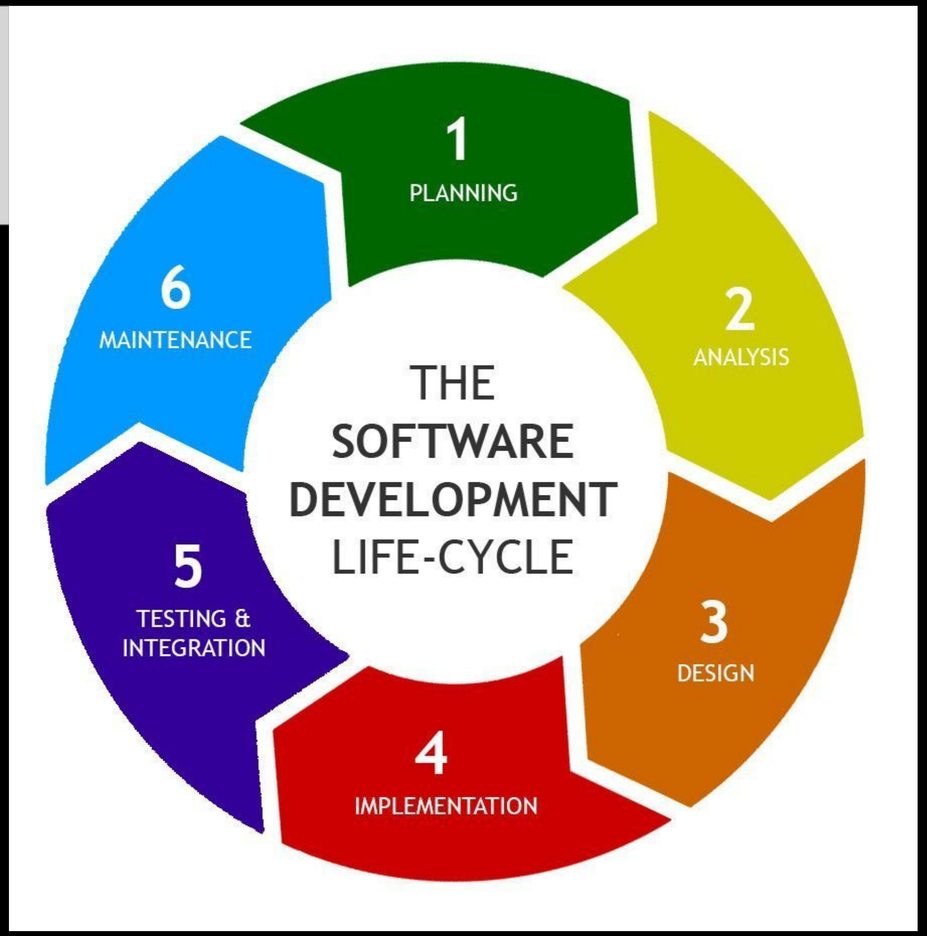
Авито

В Авито **больше четырёх тысяч микросервисов** на JavaScript, Python и Go

У нас уже **больше 2700 инженеров**

Release model

Продукт
монолит

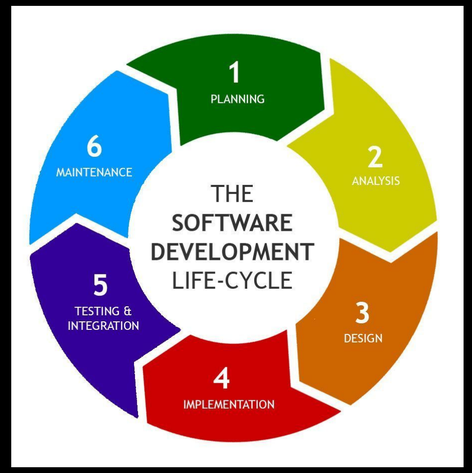


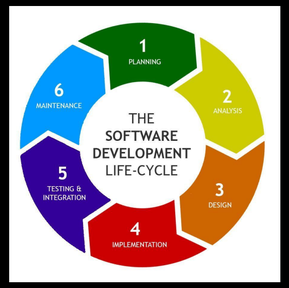
Добавляем сюда безопасность на каждый этап

Вы великолепны!

Release model

Продукт
микросервисы





S-SDLC Как построить?

Чтобы построить S-SDLC вокруг SDLC нужно применить S-SDLC для каждого процесса, который мы строим внутри S-SDLC

S-SDLC Как построить?

Чтобы построить S-SDLC вокруг SDLC нужно применить S-SDLC для каждого процесса, который мы строим внутри S-SDLC

Догфудинг

Неожиданные побочки от наших процессов

S-SDLC Планирование + анализ

Оценка рисков
Моделирование угроз

Сбор требований

Нагрузка на команду ИБ на каждую итерацию

SLA/SLO/SLI

Скорость итераций

Доступный ресурс разработки/команды ИБ

S-SDLC Планирование + анализ

Оценка рисков
Моделирование угроз

Сбор требований

Нагрузка на команду ИБ на каждую итерацию

SLA/SLO/SLI

Скорость итераций

Доступный ресурс разработки/команды ИБ

Нагрузка на ИБ - 3 часа на итерацию

1 релиз в неделю

SLA 1 рабочий день

доступный ресурс:
1 разработчик
1 app sec

S-SDLC Планирование + анализ

Стоит синхронизироваться со стратегией развития бизнеса - 1 релиз в неделю сейчас

А что планируется через год? три? пять?

У бизнеса Agile - используйте Agile в ИБ

доступный ресурс: 1 разработчик 1 app sec	сейчас 1 релиз в неделю	через год 5 релизов в неделю	через 3 года 50 релизов в неделю
	нагрузка: 3 часа в неделю	нагрузка: 15 часов в неделю	**нагрузка: 150 часов в неделю**
	SLA 1 рабочий день	**SLA 1 рабочий день**	**SLA 1 рабочий день**

S-SDLC Планирование + анализ

Стоит синхронизироваться со стратегией развития бизнеса - 1 релиз в неделю сейчас

А что планируется через год? три? пять?

У бизнеса Agile - используйте Agile в ИБ

доступный ресурс:
1 разработчик
1 app sec

автоматизируем опросник по рискам	через год 5 релизов в неделю	через 3 года 50 релизов в неделю
оптимизируем фреймворк для модели угроз	нагрузка: 7.5 часов в неделю (итерация - 1.5 часа)	**нагрузка: 75 часов в неделю**
платформенные библиотеки для соблюдения требований закона	**SLA 1 рабочий день**	**SLA 1 рабочий день**

S-SDLC Планирование + анализ

Стоит синхронизироваться со стратегией развития бизнеса - 1 релиз в неделю сейчас

А что планируется через год? три? пять?

У бизнеса Agile - используйте Agile в ИБ

доступный ресурс:
1 разработчик
3 app sec

автоматизируем опросник по рискам	автоматизируем внесение рисков в систему	через 3 года 50 релизов в неделю
оптимизируем фреймворк для модели угроз	автоматизация оценки рисков	нагрузка: 25 часов в неделю (итерация - 30 минут)
платформенные библиотеки для соблюдения требований закона	нанимаем +2 app sec	SLA 1 рабочий день

S-SDLC Собираем требования бизнеса

4000+ микросервисов

7200 релизов в неделю
(это 3 релиза в минуту)

S-SDLC Собираем требования бизнеса

4000+ микросервисов

7200 релизов в неделю
(это 3 релиза в минуту)

3 релиза в минуту:

- если на проверку каждого релиза будет уходить 30 минут AppSec инженера (на всех этапах) нам понадобится “всего” 90 человек
- нанять 90 человек - это не очень эффективно, но тоже решение проблемы
- мы попробуем в автоматизацию

Безопасность API

Знать какие риски ИБ несет API:

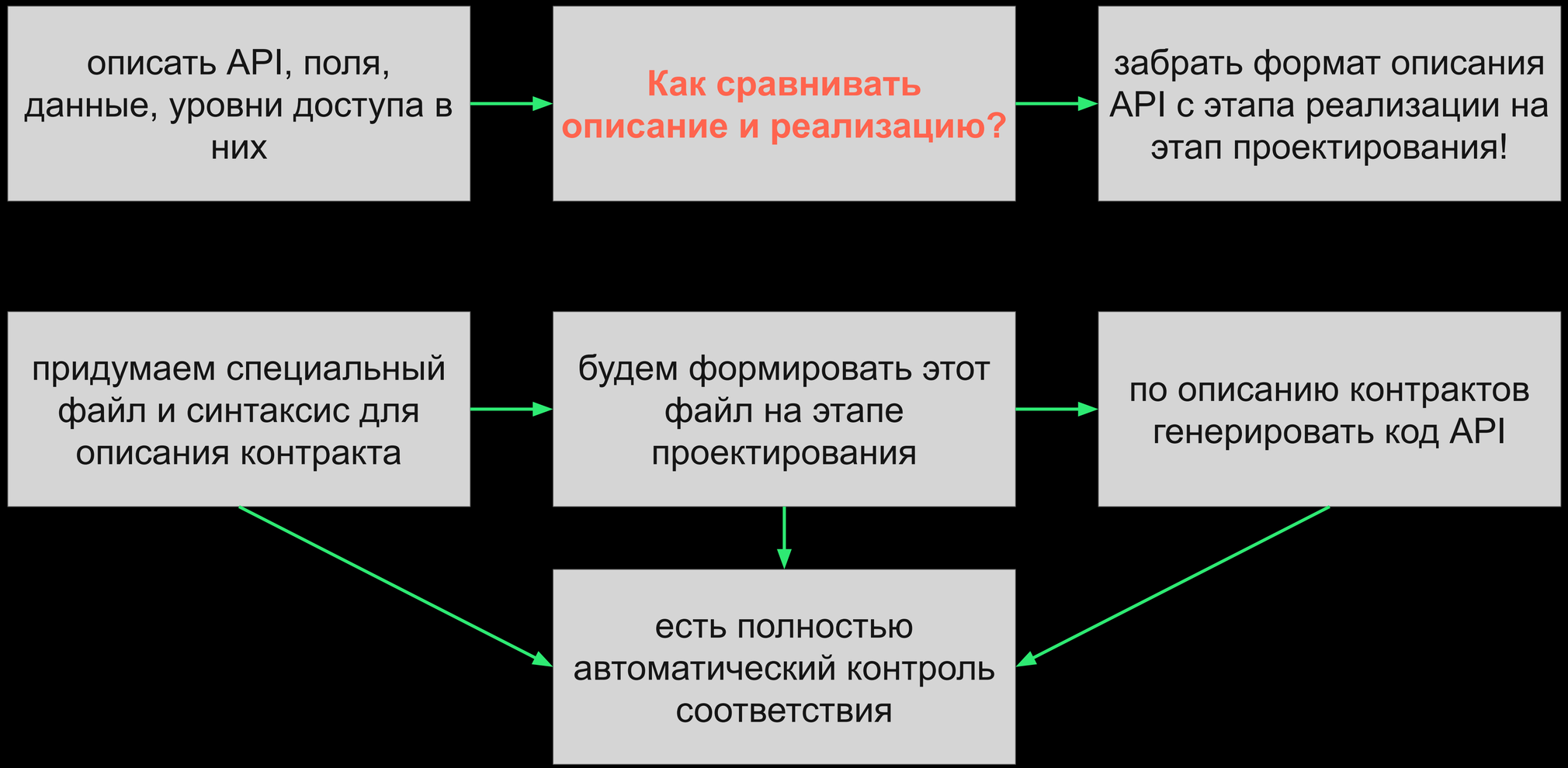
- все API, уровни доступа, все поля, виды данных в них
- все взаимодействия между компонентами
- зачем нужен каждый API и что он делает
- есть ли в API уязвимости
- все потоки данных*

* это строится из взаимодействия, поэтому получаем +- автоматически

Какие риски мы готовы принять?

S-SDLC этап планирования, анализа, проектирования

все API, уровни доступа, все поля, виды данных в них

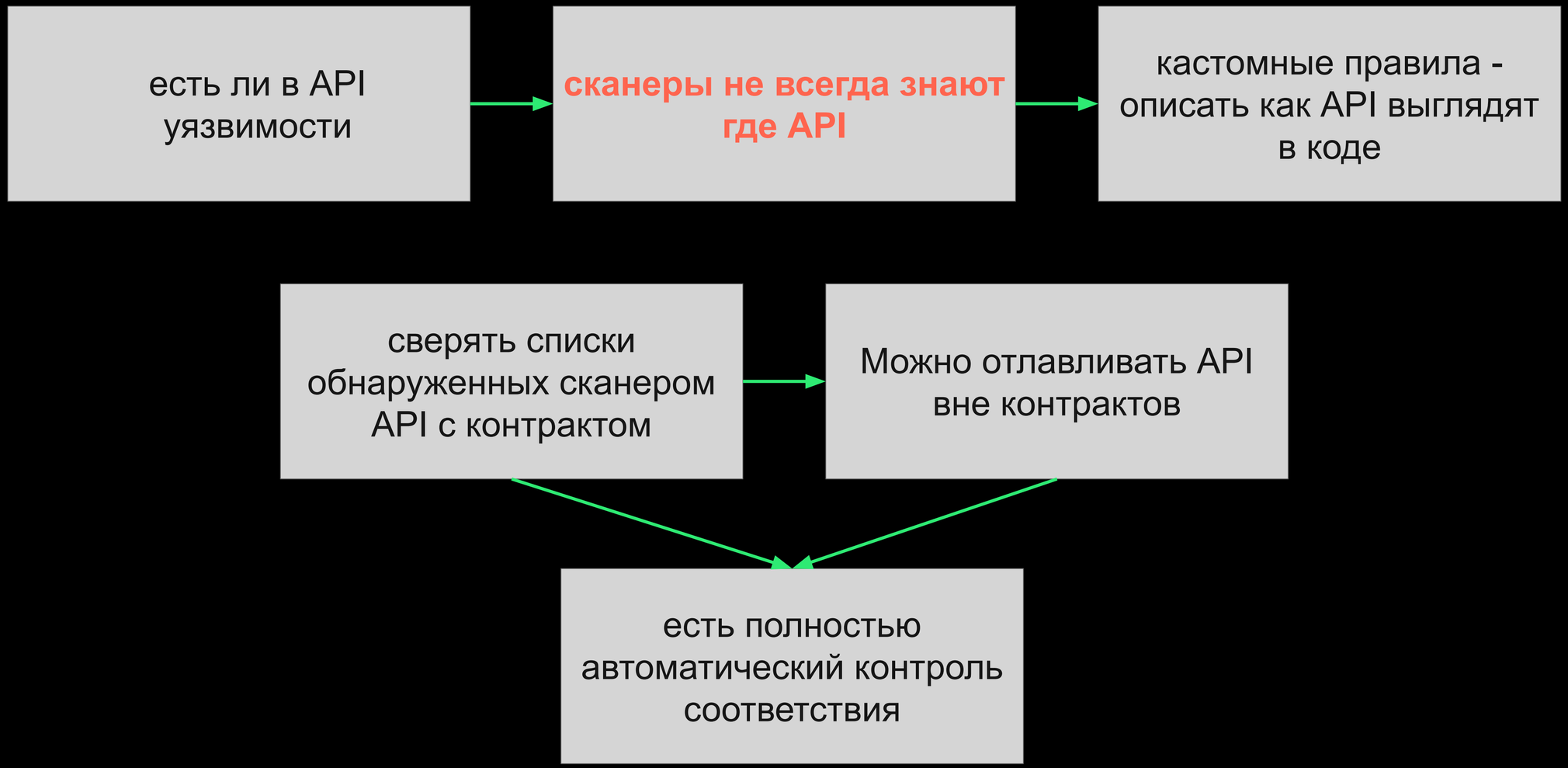


S-SDLC этап планирования, анализа, проектирования

все взаимодействия между компонентами

S-SDLC этап реализация

есть ли в API уязвимости



S-SDLC тестирование

Важно: на ручное тестирование выносить
только то, чего не может автоматика!

Критерий	Что делаем?
все API, уровни доступа, все поля, виды данных в них	ищем нет ли API опубликованного в обход
все взаимодействия между компонентами	проверить соблюдение принципа наименьших привилегий
зачем нужен каждый API и что он делает	тестируем сценарии и логику

S-SDLC Публикация

Критерий	Что делаем?	Что может пойти не так?
все API, уровни доступа, все поля, виды данных в них	единый флоу публикации сверяем что публикуется только API указанное как публичное встраиваем сюда Quality Gate	**Технически можно опубликовать API в обход** **Доступен реактивный контроль**
есть ли в API уязвимости	в дело вступает багбаунти и логирование	могут попытаться эксплуатировать, а не сообщать

S-SDLC Новый круг!

Нужно закладывать ресурсы на постоянные изменения в процессах

Это сбережёт нервы AppSec

Позволит успевать наравне с бизнесом, а не играть в роли догоняющих

Выводы

01 Закладывайте ресурсы на постоянное улучшение процессов

02 Платформенные решения драматично повышают (или понижают) безопасность

03 SDLC - гибок, если где-то нужно совместить этапы сделай это

04 Есть что-то нужно заполнять руками - там будут ошибки

05 Рекурсия и догфудинг улучшит User Experience

Безопасность должна быть гибкой

Александр Трифанов

тех и этих лид

Процессы требуют постоянного улучшения