

# Как вовлечь разработчиков в фикс уязвимостей.

Рецепт от СберМаркета

“Я триажу по **100** фолсов в день.  
И чувствую себя прекрасно”



Кашапов Нияз  
AppSec Lead @ Сбермаркет

 @niaz.kashapov

 niyaz.kashapov@sbermarket.ru

С чего начинается день AppSec?..

...с кофе и триажа найдингов



Надо успеть еще...

Отревьювить архитектуру решений

Проконсультировать команды по вопросам

Нарисовать модель угроз для новых систем

Написать требования по безопасности

Провести аудит нового сервиса

**Много чего другого!**

Проверить фиксы уязвимостей

Мониторить новости об уязвимостях

Оттриажить bug-bounty

Поресерчить новые практики

Написать статью/выступить на конфе...

# Хватит это терпеть!

- 1) Огромное количество фандингов
- 2) Отсутствие бизнес-контекста
- 3) Сложный код
- 4) Игнорирующие команды разработки
- 5) Вечно растущий техдолг



# Как мы доварились то такого?

Невкусный рецепт двухлетней давности

**200+ микросервисов**

**150 команд разработки**

**несколько OpenSource SAST**

**defectdojo с кривой  
дедупликацией**

**невозможность остановить  
деплой из-за уязвимостей**



Начинаем готовить!



Начинаем с *технического* мяса!



Отказываемся от defect dojo  
в пользу... ничего!

```
Сканер // Gosec
=====
https://gitlab/test/manual_front/compare/main.go
|| Название // Тип || - // CWE-88
|| Описание || Potential HTTP request made with variable url
|| Критичность || Medium
|| Строка // Файл || 113 // test/manual_front/compare/main.go
|| Уязвимый код || 112: func read(url string) (*Mono, error) {
113: resp, err := http.Get(url)
114: if err != nil {
|| Идентификатор || test/manual_front/compare/main.go:CWE-88:113
|| Об Уязвимости || -
|| Сканер || Gosec
=====
https://gitlab/test/manual_front/compare/compare_with_kt/main.go
|| Название // Тип || - // CWE-88
|| Описание || Potential HTTP request made with variable url
|| Критичность || Medium
|| Строка // Файл || 352 // test/manual_front/compare/compare_with_kt/main.go
|| Уязвимый код || 351: func readKT(url string) (*KT, error) {
352: resp, err := http.Get(url)
353: if err != nil {
|| Идентификатор || test/manual_front/compare/compare_with_kt/main.go:CWE-88:352
|| Об Уязвимости || -
|| Сканер || Gosec
=====
|| Название // Тип || - // CWE-88
|| Описание || Potential HTTP request made with variable url
|| Критичность || Medium
|| Строка // Файл || 334 // test/manual_front/compare/compare_with_kt/main.go
|| Уязвимый код || 333: func read(url string) (*Mono, error) {
334: resp, err := http.Get(url)
335: if err != nil {
|| Идентификатор || test/manual_front/compare/compare_with_kt/main.go:CWE-88:334
|| Об Уязвимости || -
|| Сканер || Gosec
=====
```

# Плюсы в отказе от DefectDojo

Как для appsec, так и для Dev-команд

## 01

скорость получения результатов  
(непосредственно в пайплайне)

## 03

нет необходимости управлять учетными записями (доступ по умолчанию)

## 02

отсутствие дополнительного инструмента, работа в "едином" окне

## 04

Пайплайны можно валить при обнаружении критичных уязвимостей

# Пайплайны и сканеры

Подробности в статье на Хабре



# Как устроен наш пайплайн

Шашлычок из джобов



# Ставим оценки за результаты прохождения пайплайнов

Оцениваем прохождение SAST, SCA, Secret Detection

Security **A+** 100/100

**Уязвимости в репозитории** 06.03.2024 [Как улучшить](#) [Пересчитать](#)

Данная оценка показывает максимальный уровень критичности уязвимости, найденной в репозитории, что напрямую коррелирует с уровнем риска Информационной Безопасности. Если найдена хотя бы одна уязвимость высокого уровня критичности, выставляется оценка C, если среднего - оценка B, если низкого - оценка A. Если уязвимостей нет, - оценка A+

**A+** 100/100

**Уязвимости в используемых библиотеках** 06.03.2024 [Как улучшить](#) [Пересчитать](#)

Данная оценка показывает максимальный уровень критичности уязвимости в используемых библиотеках. Если в сервисе используется библиотека, в которой найдена хотя бы одна уязвимость высокого уровня критичности, выставляется оценка C, если среднего - оценка B, если низкого - оценка A. Если в используемых библиотеках уязвимостей нет, - оценка A+

**A+** 100/100

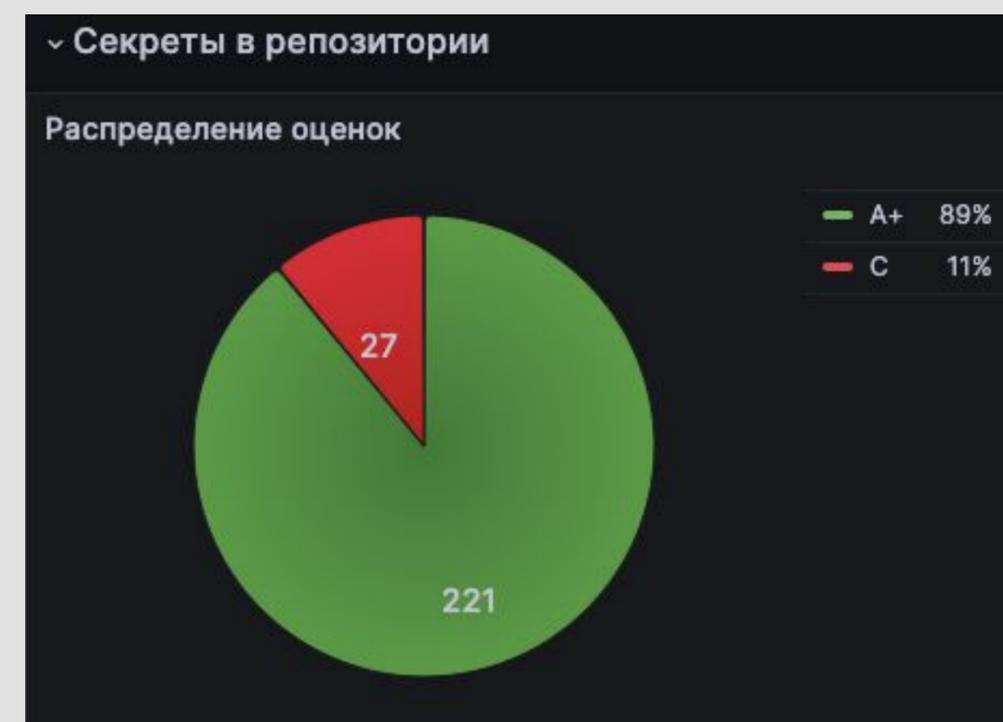
**Секреты в репозитории** 06.03.2024 [Как улучшить](#) [Пересчитать](#)

Секреты следует хранить в Vault. Если секреты найдены, будет выставлена оценка C, если нет - A+

**A+** 100/100

# Мерим общее состояние дел по всей компании

Это уже метрика для AppSec



Переходим к основе)





План кодера на день:

- 1) Код писат
- 2) МР делать
- 3) Код Пушит
- 4) Ждать (обед)
- 5) Ошибки видит
- 6) Бежат от AppSec



План разработчика на день:

- 1) Код качественный писать
- 2) MR создать, отдать на ревью
- 3) Посмотреть результаты сканов в пайплайне
- 4) Проанализировать на false positive
- 5) Добавить исключения в репозиторий

# Exclusions as a Code

The screenshot shows a GitHub pull request interface. At the top, the title is "[CS-1997] Scoring. Configuratore exceptions". There are buttons for "Edit" and "Code". Below the title, it says "Open" and "Разработчик Гигачад requested to merge cs-1997-exclude-false-secr... into master" with a merge icon and "3 days ago".

Navigation tabs include "Overview 0", "Commits 2", "Pipelines 2", and "Changes 1".

The file comparison section shows "Compare master and latest version" with "1 file +9 -0" and a settings icon. The file path is "apps/paas-platform-scoring/config.yml" with a diff range of "0 → 100644" and a "Viewed" checkbox.

```
1 + exclusions:
2 +   general_exclude_bases:
3 +     - name:
4 +       "exclude_mattermost_channel_id_con
5 +       text"
6 +       field: "Контекст"
7 +       pattern:
8 +       "mattermost_channel_id"
9 +     sast:
10 +     - general_exclude:
11 +       exclude_mattermost_channel_id_cont
12 +       ext
13 +     - finding:
14 +       "pkg/clients/odin/openapi/services
15 +       /type.go:CWE-798:7"
16 +     - finding:
17 +       "pkg/clients/odin/openapi/dependen
18 +       cies/type.go:CWE-798:7"
```

Способ готовки: экспресс-курс



до обучения



обычный разработчик скорее всего не очень сильно разбирается в безопасности, и понять false positive это или нет не сможет  
это не бро для AppSec

после обучения



после проведения обучения, разработчик будет знать про типовые уязвимости, будет уметь писать исключения для файндингов, понимать тестировать сервисы на безопасность

Способ готовки:  
специализированный-курс



# А где мотивация?

Ставим оценку сервисам за качество

Ставим минимальный уровень для команд, чтобы точно понимали что надо работать над безопасностью

Ставим OKR для тимлидов для поднятия и поддержки уровня

Делаем премию зависимой от OKR



Переходим к процессам



# Оцениваем как обучились разработчики

Оценка Security Education & Culture

## Уровень 1

- Указан состав команды в описании в Story команды
- Командой пройден онбординг курс по безопасности и сдан финальный тест

Дата назначения	Срок выполнения	Статус
18 мар. 2021 г.	–	Не начат
20 февр. 2023 г.	–	Завершен
7 мая 2021 г.	–	Не начат
19 мар. 2021 г.	–	Не начат
4 июн. 2021 г.	–	Завершен
10 мар. 2022 г.	–	Завершен

# Оцениваем как обучились разработчики

Оценка Security Education & Culture

## Уровень 2

- Выбран Security Champion, который прошел курс по своей специальности Backend или Frontend/Mobile for React Native

Domain	Team	Security champion	Team lead	Stack	
Demand	Odin	@ Разработчик Гошник Секчемповец	@ Тимлид Секчемпа Крутовец	js, java tester	QA
Demand	Odin	@ Бэкендер Секчемп Бэкендеровец	@ Тимлидов Бэкенд Разработчикович	ruby	Back

# Оцениваем как обучились разработчики

Оценка Security Education & Culture

## Уровень 3

- Security Champion провел встречу со своей командой, по изученным им материалам по безопасности и приложил материалы или запись проведенного семинара

или

- Все члены команды прошли курсы Backend или Frontend/Mobile for React Native в зависимости от специальности

# Оцениваем как разработчики взаимодействуют

Оценка Secure Development

## Уровень 1

- Прочитали статью по взаимодействию с командой AppSec (Взаимодействие с AppSec)
- Связались с командой AppSec и запросили аудит безопасности
- Получили список требований после аудита и согласовали с бизнес партнером AppSec

СУВ-13569

[Operations] Аудит  
сервиса order-service



AppSec Audit

# Оцениваем как разработчики взаимодействуют

Оценка Secure Development

## Уровень 2

- Исправили/реализовали все замечания/предложения после аудита безопасности и получили апрув у бизнес партнера AppSec
- Прочитали статьи и следуете безопасными паттернами при разработке

# Оцениваем как разработчики взаимодействуют

Оценка Secure Development

## Уровень 3

- Достигли и поддерживаете уровень метрик на уровне A+ (SAST/SCA/Secrets)
- Добавляете самостоятельно исключения в <https://gitlab/information-security/devsecops-configs> при возникновении фолсов

### [MI-881]:Retailer office front firebase

!343 · created 3 days ago by Разработчик Команда 1

✓ 8✓ 1 left 0  
updated 3 days ago

### [CS-1997] Scoring. Configure exceptions

!335 · created 4 days ago by Разработчик Красавчик 2

✓ 8✓ 1 left 0  
updated 4 days ago

# Что такое TMM?

*“Модель зрелости команд (Team Maturity Model TMM) – это список инженерных практик с описанием базового уровня для всех команд разработки в СберМаркете.*

*Цель внедрения TMM — сильная инженерная культура.”*

# Зачем TMM для команд?

- понимать, что такое "базовый уровень" в компании и как его достичь;
- диалог с экспертами – куда идет компания, что нужно делать командам, почему именно так;
- инструмент для самодиагностики и ретроспективы, чтобы найти зоны роста и создать план развития.

20.	[3 уровень] Внедрить предиктивную аналитику или способы отслеживания отклонений/аномалий в работе сервисов	TMM-1649	Observability		OPEN
21.	[2 уровень] Выбрать участника команды и назначить его Security Champion и внести в таблицу	TMM-1650	Security Ed&Culture		OPEN
22.	[2 уровень] Security Champion должен назначить себе курс самостоятельно по своей специальности и пройти его	TMM-1651	Security Ed&Culture		OPEN
23.	[3 уровень] Security Champions назначает встречу со своей командой, проводит семинар/лекцию по изученным им материалам	TMM-1652	Security Ed&Culture		OPEN
24.	[3 уровень] Подтвердить знания прохождением тестирования всеми членами команды	TMM-1653	Security Ed&Culture		OPEN
25.	[2 уровень] Исправить/реализовать все замечания/предложения после аудита безопасности и получить апрув у бизнес партнера AppSec	TMM-1654	Secure Development		CLOSED
26.	[2 уровень] Изучить и следовать безопасными паттернами	TMM-1655	Secure Development		CLOSED
27.	[3 уровень] Самостоятельно поддерживать скоринг A+ для своего сервиса (если есть)	TMM-1656	Secure Development		OPEN
28.	[3 уровень] Договориться с юнит лидом/хэдом домена о максимальном значении Lead time по PROJ для вашей команды	TMM-1657	Delivery metrics		OPEN
29.	[3 уровень] Связать задачи команды (через эпик команды или через конкретные задачи) с задачами в PROJ и делать это на регулярной основе	TMM-1658	Delivery metrics		OPEN
30.	[3 уровень] Проводить регулярные ретроспективы/обзоры потока по улучшению метрик на основании командных задач и задач из проекта PROJ	TMM-1659	Delivery metrics		OPEN
31.	[3 уровень] Попасть в согласованный лид тайм по PROJ с юнит лидом/хэдом домена	TMM-1660	Delivery metrics		OPEN
32.	[3 уровень] Достигнуть определенного значения Velocity/Throughput и для Kanban команд подготовить расчет SLA по командным типам задач	TMM-1661	Delivery metrics		OPEN

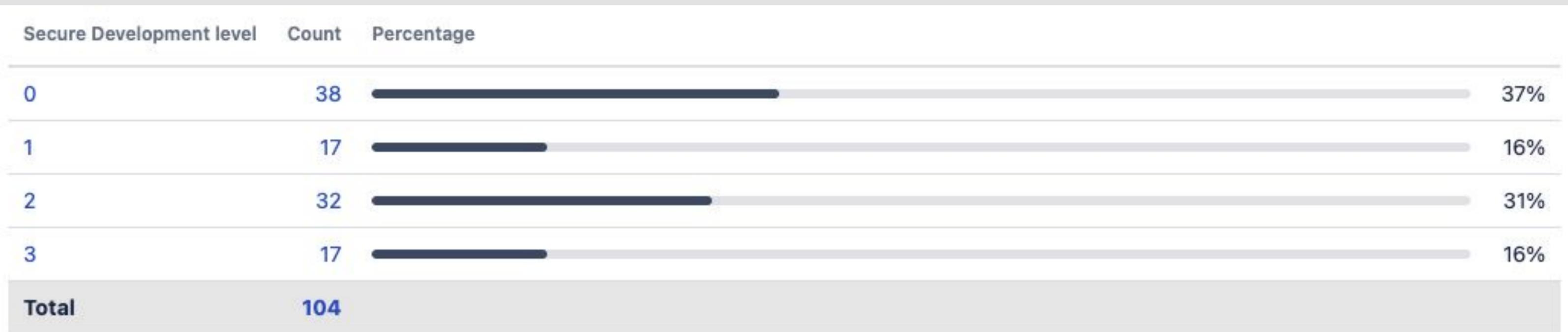
# Зачем TMM для команд?

- понимать, что такое "базовый уровень" в компании и как его достичь;
- диалог с экспертами – куда идет компания, что нужно делать командам, почему именно так;
- инструмент для самодиагностики и ретроспективы, чтобы найти зоны роста и создать план развития.

20.	[3 уровень] Внедрить предиктивную аналитику или способы отслеживания отклонений/аномалий в работе сервисов	TMM-1649	Observability		OPEN
21.	[2 уровень] Выбрать участника команды и назначить его Security Champion и внести в таблицу	TMM-1650	Security Ed&Culture		OPEN
22.	[2 уровень] Security Champion должен назначить себе курс самостоятельно по своей специальности и пройти его	TMM-1651	Security Ed&Culture		OPEN
23.	[3 уровень] Security Champions назначает встречу со своей командой, проводит семинар/лекцию по изученным им материалам	TMM-1652	Security Ed&Culture		OPEN
24.	[3 уровень] Подтвердить знания прохождением тестирования всеми членами команды	TMM-1653	Security Ed&Culture		OPEN
25.	[2 уровень] Исправить/реализовать все замечания/предложения после аудита безопасности и получить апрув у бизнес партнера AppSec	TMM-1654	Secure Development		CLOSED
26.	[2 уровень] Изучить и следовать безопасными паттернами	TMM-1655	Secure Development		CLOSED
27.	[3 уровень] Самостоятельно поддерживать скоринг A+ для своего сервиса (если есть)	TMM-1656	Secure Development		OPEN
28.	[3 уровень] Договориться с юнит лидом/хэдом домена о максимальном значении Lead time по PROJ для вашей команды	TMM-1657	Delivery metrics		OPEN
29.	[3 уровень] Связать задачи команды (через эпик команды или через конкретные задачи) с задачами в PROJ и делать это на регулярной основе	TMM-1658	Delivery metrics		OPEN
30.	[3 уровень] Проводить регулярные ретроспективы/обзоры потока по улучшению метрик на основании командных задач и задач из проекта PROJ	TMM-1659	Delivery metrics		OPEN
31.	[3 уровень] Попасть в согласованный лид тайм по PROJ с юнит лидом/хэдом домена	TMM-1660	Delivery metrics		OPEN
32.	[3 уровень] Достигнуть определенного значения Velocity/Throughput и для Kanban команд подготовить расчет SLA по командным типам задач	TMM-1661	Delivery metrics		OPEN

# Зачем ТММ для AppSec?

Картина мира и метрики!



Что в итоге?



# Рецепт:

- 1) Результаты пайплайнов безопасности вынесим в job-output
- 2) Исключения пишем в виде кода в git
- 3) Добавляем Security Quality Gate для остановки пайплайна
- 4) Внедряем оценки по Secure Development и Security Education в процесс ревью команд

# Что имеем после внедрения?

46

контрибьюторов в репозиторий с исключениями

85

Security Champion в разных командах разработки

65%

команд активно работают с AppSec BP

80%+

сервисов проходят security quality gate без проблем

# Что изменилось для AppSec-команды

- 1) Стало легче отслеживать команды в которых нужно внимание (консультация или аудит)
- 2) Разработчики сами триажают findings сканеров
- 3) Сразу видно сервисы где нужно внимание (оценка В или С)
- 4) Разработчики начали проходить курсы по безопасной разработке
- 5) Есть сформированный и актуальный список Security Champion

спасибо за внимание



Кашапов Нияз  
AppSec Lead @ Сбермаркет

 @niaz.kashapov

 niyaz.kashapov@sbermarket.ru