



**POSITIVE
TECHNOLOGIES**

DevSecOps. PT AI в разработке ПО: блокировка релиза

Алексей Жуков

Эксперт отдела систем защиты приложений

Тимур Гильмуллин

Заместитель руководителя отдела технологий и процессов разработки

Дмитрий Рагулин

CI-инженер отдела технологий и процессов разработки

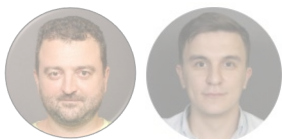
Антон Володченко

Руководитель группы обеспечения качества

ptsecurity.com

PT Application Inspector: серия вебинаров

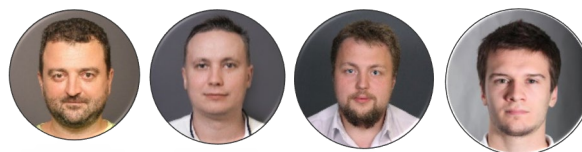
PT



22 октября 2020

PT Application Inspector:
обзор новой версии и roadmap

<https://www.youtube.com/watch?v=MSotiaSowmM>



Сегодня

**DevSecOps: процесс устранения
уязвимостей в приложениях. PT Application
Inspector в релизном цикле разработки ПО**

- Демонстрация AI Security Gates и блокировки мердж-реквестов (техническая часть)
- Типовой пайплайн сборки, в который внедрено сканирование PT AI (техническая часть)



Q4 2021

Анонс новой версии продукта PT AI

- Проблематика уязвимостей в коде приложений
- Рассказ о продукте: возможности, демонстрация
- История развития и Roadmap



3 декабря 2020

DevSecOps:
внедрение в продуктовый конвейер
и эксплуатация PT Application Inspector

<https://youtu.be/Qg-SmDpMSQ4>

План вебинара



01

Введение

- От DevOps к DevSecOps
- PT Application Inspector: основные возможности и способы интеграции с CI-системами

02

AI Security Gates и блокирование мердж-реквестов

- Настройка AI Security Gates и блокировка мердж-реквестов для сборок в GitLab CI и TeamCity
- Бот AI для работы с мердж-реквестами
- Автоматическая блокировка мердж-реквестов

03

Демонстрация интеграции

- Типовой пайплайн сборки, в который внедрено сканирование

04

Демонстрация работы с уязвимостями

- Что делают разработчики с результатами сканирования

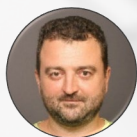
05

Q&A



Введение

PT Application Inspector для команды



Алексей Жуков

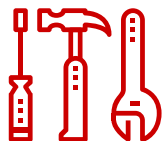
Эксперт отдела систем защиты приложений

ptsecurity.com

PT AI как часть DevSecOps

PT

От простого к сложному



ПРОСТОТА ИСПОЛЬЗОВАНИЯ

- Плагины для CI / CD
- CLI-интерфейс
- Кроссплатформенность



УДОБСТВО РАБОТЫ С РЕЗУЛЬТАТАМИ

- Отчеты в HTML / PDF
- Результаты сканирования в XML / JSON
- Фильтры



API И РАСШИРЯЕМОСТЬ

- Работа с JSON-результатами
- Реализация индивидуальной логики обработки результатов анализа

Неограниченный простор для творчества команды Dev(Sec)Ops



DevSecOps

PT AI в разработке ПО: блокировка релиза

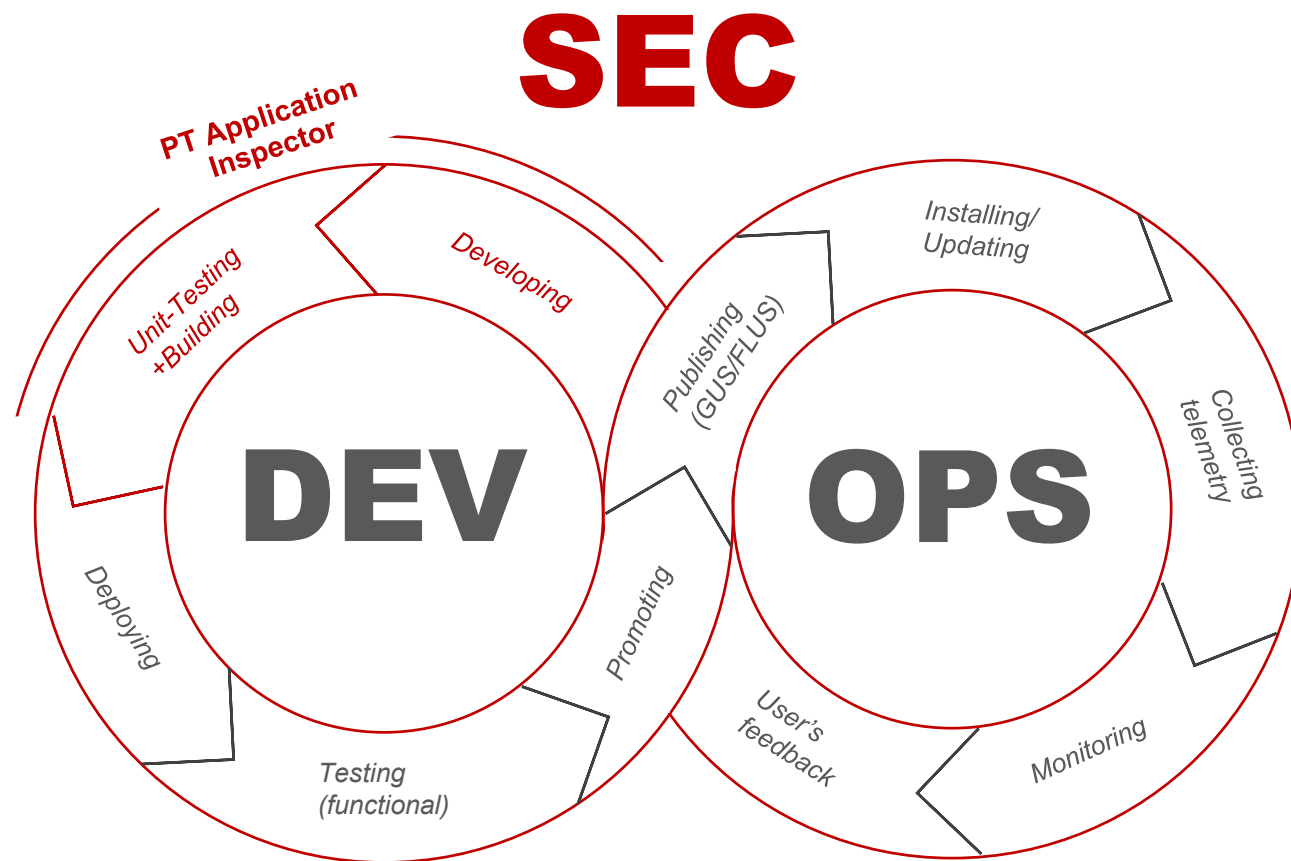


Тимур Гильмуллин

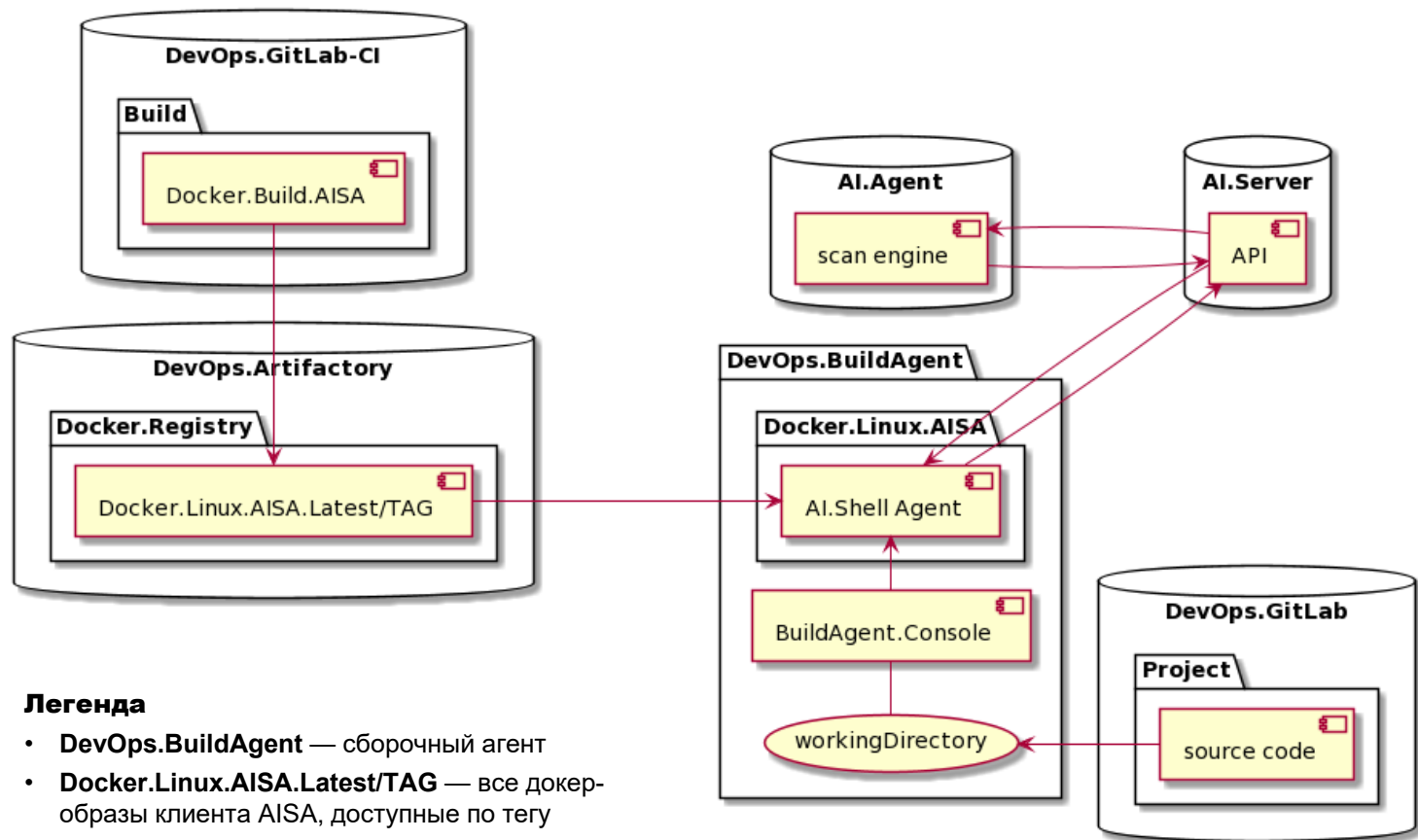
Заместитель руководителя отдела технологий и процессов разработки

ptsecurity.com

Процессы DevSecOps в PT



Архитектура и методика



Легенда

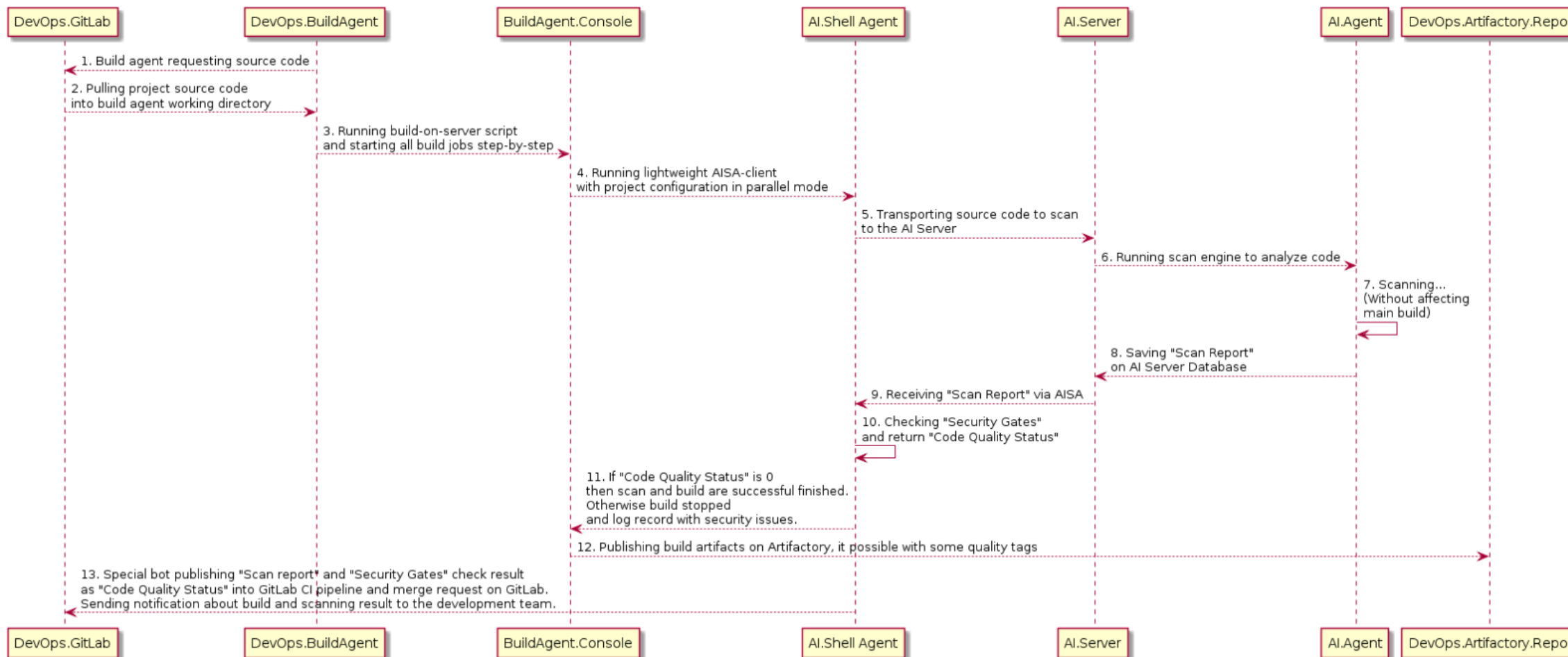
- **DevOps.BuildAgent** — сборочный агент
- **Docker.Linux.AISA.Latest/TAG** — все докер-образы клиента AISA, доступные по тегу
- **AI.Agent** — агент сканирования PT AI
- **AI.Server** — PT Application Inspector сервер
- **DevOps.GitLab** — хранилище кода
- **DevOps.GitLab-CI** — CI-система
- **DevOps.Artifactory** — хранилище артефактов
- **Docker.Registry** — хранилище докер-образов

Методика

- Подготовка серверной части
 - Установка и настройка сервера
 - Установка и настройка агентов сканирования
- Подготовка клиентской части
 - CI-цикл для клиента AISA (поставка docker-образом)
- Подготовка проекта сканирования
 - На стороне сервера
 - Через клиент AISA
- Работа с CI-системами
 - Шаблоны сканирования в GitLab CI
 - Метараннеры в TeamCity
 - Работа с CLI AISA

- **Docker.Linux.AISA** — артефакт сборки клиента AISA (рабочий докер-образ с клиентом)
- **AI.Shell Agent** — клиент AISA, запущенный внутри докер-контейнера, работающий с API AI-сервера
- **BuildAgent.Console** — системная консоль сборочного агента
- **workingDirectory** — рабочий каталог на сборочном сервере с исходным кодом, который будет сканироваться

Сборочный процесс с использованием PT Application Inspector



AI Security Gates

Правила и группировки

Пример: aisa-codequality.settings.yaml

```
1  threats mapping:
2    info: Potential
3    minor: Low
4    major: Medium
5    critical: High
6    blocker: []
7  security gates:
8    info: 0
9    minor: 0
10   major: 2
11   critical: 1
12   blocker: 0
```

Пример группировки в aisa-codequality.settings.yaml

```
1  threats mapping:
2    info:
3      - Potential
4      - Low
5      - Medium
6    minor: []
7    major: []
8    critical: []
9    blocker: High
```

AI Security Gates — это набор правил для проекта сканирования, которые указывают CI-системе, как группировать найденные уязвимости и как на них реагировать: блокировать сборку или мердж-реквест, либо работать в режиме информирования.

SonarQube codequality

PT

Overview 109 Commits 91 Pipelines 86 Changes 5

4 unresolved threads

PtRunnerRunCommandsParallel.py

```
from yaml import Loader
from yaml import FullLoader
```

builder_sonar @builder_sonar · 1 year ago

Remove this commented out code.

Developer

Reply...

builder_sonar @builder_sonar commented on commit 07e79e9c 1 year ago

SonarQube analysis reported 2 issues

- 2 major

Watch the comments in this conversation to review them.

1 extra issue

Note: The following issues were found on lines that were not modified in the commit. Because these issues can't be reported as line comments, they are summarized here:

- Remove this commented out code.

Reply...

builder_sonar @builder_sonar commented on commit 9e54eb6 1 year ago

SonarQube analysis reported 2 issues

- 2 major

Note: The following issues were found on lines that were not modified in the commit. Because these issues can't be reported as line comments, they are summarized here:

- Remove this commented out code.
- Remove this commented out code.

Reply...

builder_sonar @builder_sonar commented on commit 7f2922b4 1 year ago

SonarQube analysis reported 2 issues

- 2 major

Note: The following issues were found on lines that were not modified in the commit. Because these issues can't be reported as line comments, they are summarized here:

- Remove this commented out code.
- Remove this commented out code.

Reply...

PtRunnerRunTeamcityConfiguration31.py

```
print('custom_parameters:')
#print('custom_parameters:')
```

builder_sonar @builder_sonar · 1 year ago

Remove this commented out code.

Developer

Reply...

builder_sonar @builder_sonar commented on commit 17e50131 1 year ago

SonarQube analysis reported 4 issues

- 1 critical
- 3 major

Watch the comments in this conversation to review them.

3 extra issues

Note: The following issues were found on lines that were not modified in the commit. Because these issues can't be reported as line comments, they are summarized here:

- Refactor this function to reduce its Cognitive Complexity from 34 to the 15 allowed.
- Remove this commented out code.
- Function "main" has 14 parameters, which is greater than the 7 authorized.

Reply...

builder_sonar @builder_sonar commented on commit af25d8af 1 year ago

SonarQube analysis reported 3 issues

- 1 critical
- 2 major

Note: The following issues were found on lines that were not modified in the commit. Because these issues can't be reported as line comments, they are summarized here:

- Refactor this function to reduce its Cognitive Complexity from 34 to the 15 allowed.
- Remove this commented out code.
- Function "main" has 14 parameters, which is greater than the 7 authorized.

Reply...

builder_sonar @builder_sonar commented on commit 7b908380 1 year ago

SonarQube analysis reported 3 issues

- 1 critical
- 2 major

Note: The following issues were found on lines that were not modified in the commit. Because these issues can't be reported as line comments, they are summarized here:

- Refactor this function to reduce its Cognitive Complexity from 34 to the 15 allowed.
- Remove this commented out code.
- Function "main" has 14 parameters, which is greater than the 7 authorized.

builder_sonar @builder_sonar commented on commit 441b8dc 1 year ago

SonarQube analysis reported 3 issues

- 1 critical
- 2 major

Note: The following issues were found on lines that were not modified in the commit. Because these issues can't be reported as line comments, they are summarized here:

- Refactor this function to reduce its Cognitive Complexity from 34 to the 15 allowed.
- Remove this commented out code.
- Function "main" has 14 parameters, which is greater than the 7 authorized.

Reply...

builder_sonar @builder_sonar commented on commit 02c51002 1 year ago

SonarQube analysis reported 3 issues

- 1 critical
- 2 major

Note: The following issues were found on lines that were not modified in the commit. Because these issues can't be reported as line comments, they are summarized here:

- Refactor this function to reduce its Cognitive Complexity from 34 to the 15 allowed.
- Remove this commented out code.
- Function "main" has 14 parameters, which is greater than the 7 authorized.

Reply...

builder_sonar @builder_sonar commented on commit e345e58b 1 year ago

SonarQube analysis reported 3 issues

- 1 critical
- 2 major

Note: The following issues were found on lines that were not modified in the commit. Because these issues can't be reported as line comments, they are summarized here:

- Refactor this function to reduce its Cognitive Complexity from 34 to the 15 allowed.
- Remove this commented out code.
- Function "main" has 14 parameters, which is greater than the 7 authorized.

Reply...

builder_sonar @builder_sonar commented on commit a0f58cd4 1 year ago

SonarQube analysis reported 3 issues

- 1 critical
- 2 major

Note: The following issues were found on lines that were not modified in the commit. Because these issues can't be reported as line comments, they are summarized here:

- Refactor this function to reduce its Cognitive Complexity from 34 to the 15 allowed.
- Remove this commented out code.
- Function "main" has 14 parameters, which is greater than the 7 authorized.

AI Security Gates: проверки

PT



Application Inspector Enterprise

Maintainer



@aie_gitlab_bot · 11 hours ago

Application Inspector detect **380** vulnerabilities for [18b4c963d](#):

Found vulnerabilities:

- ▲ **0** Blocker
- ▼ **0** High
- ◆ **55** Medium
- ◆ **18** Low
- **307** Potential

► More results...



See full [AI HTML report](#) in [TeamCity build](#).

Edited by Application Inspector Enterprise just now



Reply...

Resolve thread



Application Inspector Enterprise

Maintainer



@aie_gitlab_bot · 11 hours ago

Application Inspector detect **378** vulnerabilities for [18b4c963d](#):

▼ Merge Request has been locked by current AI Security Gates

Threats Mapping:

info: *Potential*
minor: *Low*
major: *Medium*
critical: *High*
blocker: *[]*

Security Gates:

info: *0*
minor: *0*
major: *2*
critical: *1*
blocker: *0*

Found vulnerabilities:

- ▲ **0** Blocker
- ▼ **0** High
- ◆ **53** Medium
- ◆ **18** Low
- **307** Potential

► More results...



See full [AI HTML report](#) in [GitLab job](#).

Edited by Application Inspector Enterprise 1 minute ago



Reply...

Resolve thread



Интеграция с TeamCity

PT

Test projects / Dragulin / Testing / Application Inspector Enterprise / Test CodeQuality

Run ... Actions Edit Configuration Settings | ▾

✓ #10 (17 Feb 21 12:20) | ▾

Overview Changes Build Log Parameters Artifacts AIE Scan Report

◀ ✓ #9 | All history | #11 ✓ ▶

Positive Technologies Application Inspector Scan Result

17 February 2021

Project dragulin_tests_application_inspector
Website address <http://localhost>
Programming language JavaScript
Version 3.6.2.1212
Scan duration 0:00:08
Algorithm Search starting from entry points, Search starting of public and protected methods
Excluded files *.7z, *.bmp, *.dib, *.dll, *.doc, *.docx, *.exe, *.gif, *.ico, *.jif, *.jpe, *.jpeg, *.jpg, *.odt, *.pdb, *.pdf, *.png, *.rar, *.swf, *.tif, *.tiff, *.zip
Excluded modules Vulnerability search in source code, Black-box search

⚠ The report includes results that match your preconfigured filters

Specific attributes Except suppressed
Confirmation status Confirmed, Automatically approved, No status

In the report it is not designated

Specific attributes Suppressed
Confirmation status Rejected

Source code checks

Total files 955
Checked 955
Files with vulnerabilities 4

Vulnerabilities filtered (Total)

112 (116)

Confirmed 0
Rejected 0
Not processed 112 (116)

Suppressed 0 (4)

Security policy Passed

Breakdown of vulnerabilities

By severity

Medium 15

Open source

Шаблоны автоматизации для работы с PT AI доступны на GitHub:

github.com/devopshq/dohq-ai-best-practices

- Рекомендуемая методика внедрения
- Скрипты инсталляции и рекомендации по настройке серверной части PT Application Inspector
- Dockerfile для сборки AISA-клиента (Windows и Linux)
- Шаблоны шагов запуска сканирования через PT Application Inspector:
 - job для GitLab CI,
 - метараннеры для TeamCity
- Шаблоны типовых пайплайнов для GitLab CI с возможностью блокирования мердж-реквестов.

ЧТО ПОЧИТАТЬ

- [Личный опыт: как выглядит наша система Continuous Integration](#)
- [Автоматизация процессов разработки: как мы в Positive Technologies внедряли идеи DevOps](#)
- [Моделирование производственных процессов в ИТ-компаниях](#)
- [Управление хаосом: наводим порядок с помощью технологической карты](#)
- [DevSecOps: как мы внедряли PT Application Inspector в наш продуктовый конвейер](#)
- Делимся наработками в опенсорсе [DevOpsHQ](#) и на митапах Op!DevOps! ([2016](#), [2017](#))



AI Security Gates

Блокировка мердж-реквестов



Дмитрий Рагулин

СІ-инженер отдела технологий и процессов разработки

ptsecurity.com

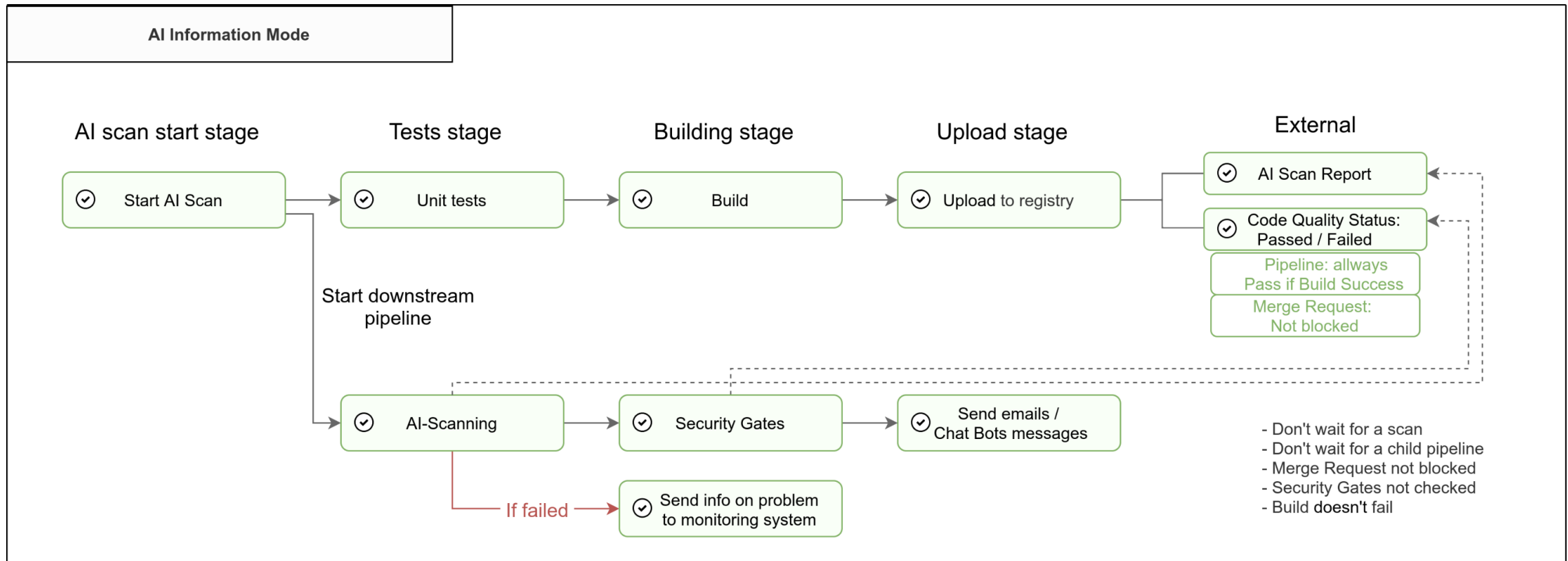
AI Security Gates:

три режима работы

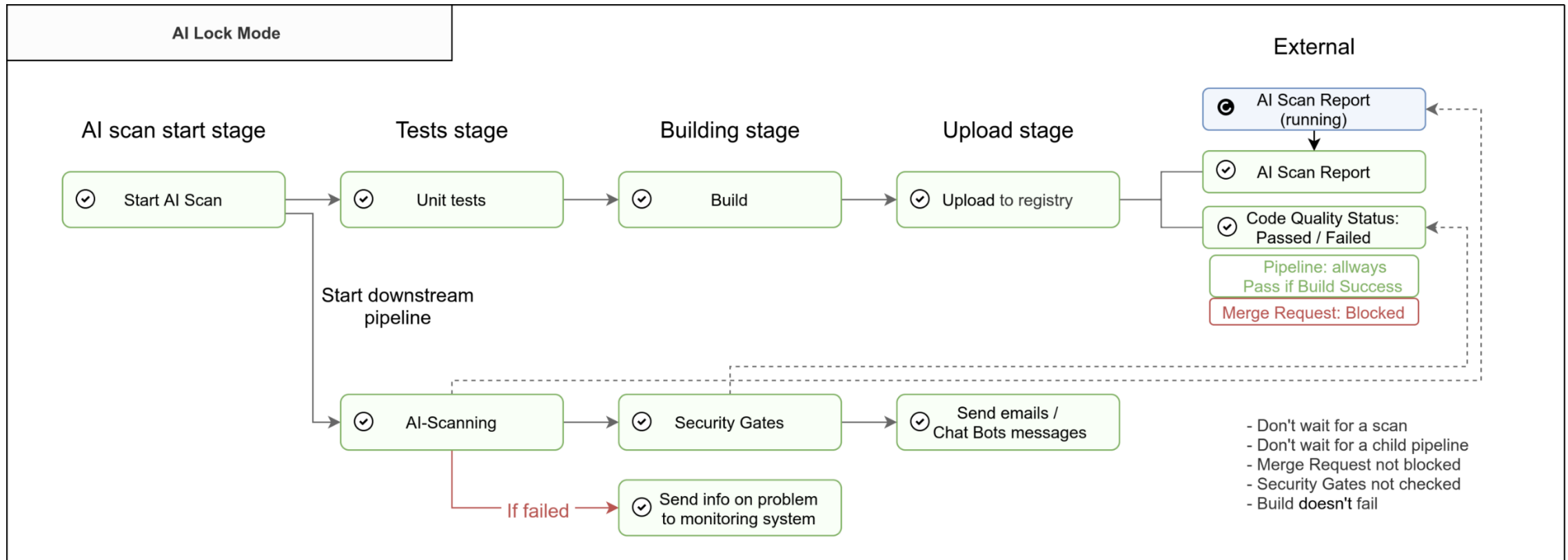
РТ

Возможности:	Режимы:	Информирования (Information mode)	Блокировки (Block mode)	Строгий (Strictest mode)
Ожидание окончания сканирования		Нет	Нет	Да
Сохранение HTML-отчета в pipeline		Да	Да	Да
Дублирование текстового отчета в merge request thread		Да	Да	Да
Проверка правил Security Gates и установка Code Quality Status		Да	Да	Да
Допускается блокировка merge request		Нет	Да	Да
Допускается блокировка pipeline		Нет	Нет	Да
Допускается блокировка шага публикации артефакта сборки в registry		Нет	Нет	Да

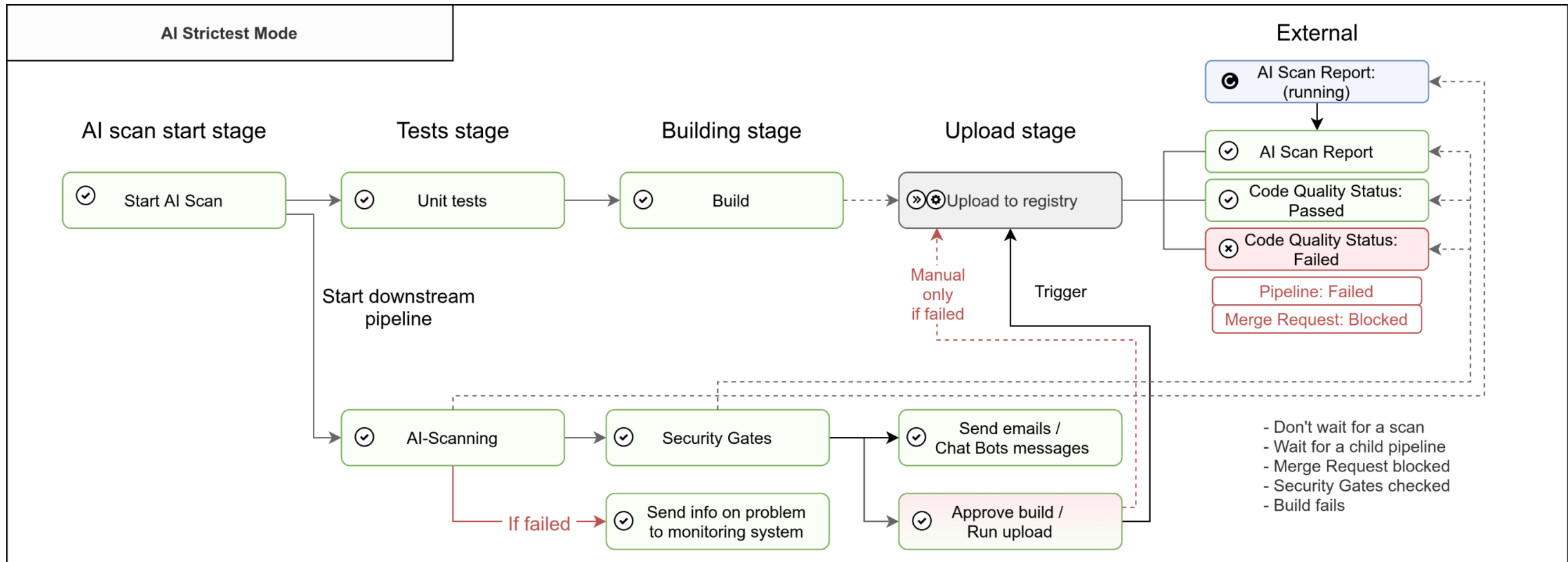
AI Security Gates: Information mode



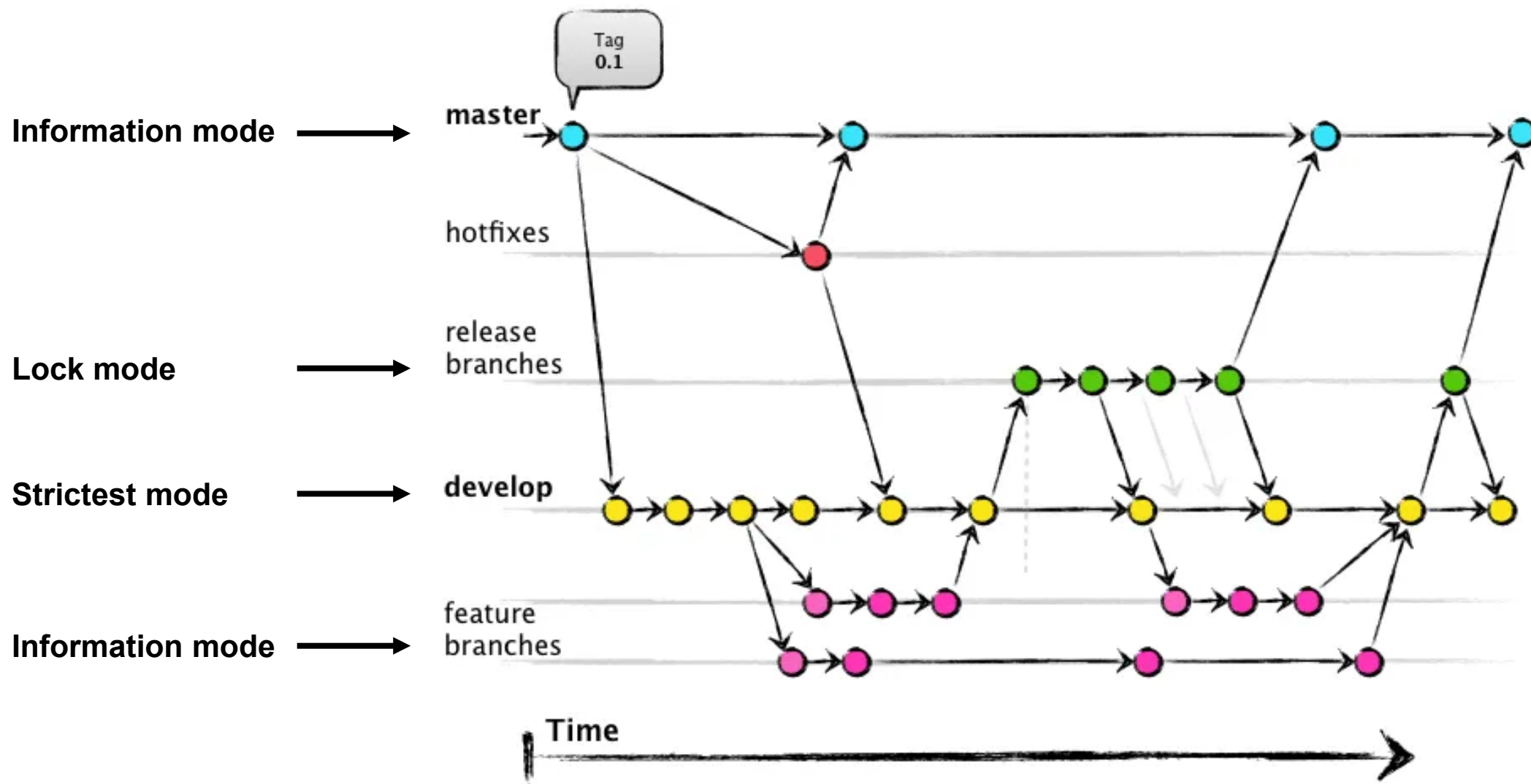
AI Security Gates: Lock mode



AI Security Gates: Strictest mode



Пример работы с ветками





Демонстрация

AI Security Gates и мердж-реквесты



Дмитрий Рагулин

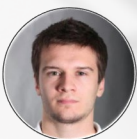
СІ-инженер отдела технологий и процессов разработки

ptsecurity.com



Демонстрация

Работа с уязвимостями. PT AI в CI



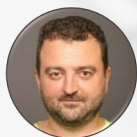
Антон Володченко

Руководитель группы обеспечения качества

ptsecurity.com



Заключение



Алексей Жуков

Эксперт отдела систем защиты приложений

ptsecurity.com

Как провести пилот PT Application Inspector



Заполните
заявку на странице
PT Application Inspector
[на сайте](#)
или свяжитесь
с персональным менеджером
Positive Technologies
или [партнера](#) компании

Подписание NDA,
заполнение анкеты
о приложениях

Развертывание PT AI:
установка, настройка,
подключение приложений

Вебинар:
работа с продуктом

Пилотный проект
при поддержке экспертов
Positive Technologies
Вебинар:
работа с уязвимостями

Отчет
о найденных
уязвимостях
Презентация
результатов

≈ 4 недели



POSITIVE
TECHNOLOGIES

Подробнее

ptsecurity.com/ru-ru/products/ai/

github.com/devopshq/dohq-ai-best-practices

АЛЕКСЕЙ ЖУКОВ

alzhukov@ptsecurity.com

ТИМУР ГИЛЬМУЛЛИН

tgilmullin@ptsecurity.com

ДМИТРИЙ РАГУЛИН

dragulin@ptsecurity.com

АНТОН ВОЛОДЧЕНКО

avolodchenko@ptsecurity.com



@BigAppSec

ptsecurity.com