PT
Информационная безопасность

Зачем аналитику SOC СиДеть и СIять или как решить проблемы обновления наборов экспертизы в SOC

Федоров Андрей
Аналитик направления мониторинга и реагирования RT Protect SOC

План доклада

1. Почему обновление наборов экспертизы — это больно?
2. Как мы попытались решить наши проблемы?
   - Создание отдельного репозитория для правил
   - Pipeline, который решает проблемы
3. Итоги

PT
Информационная безопасность

Почему обновление наборов экспертизы — это больно?

Как обновляли наборы экспертизы раньше:

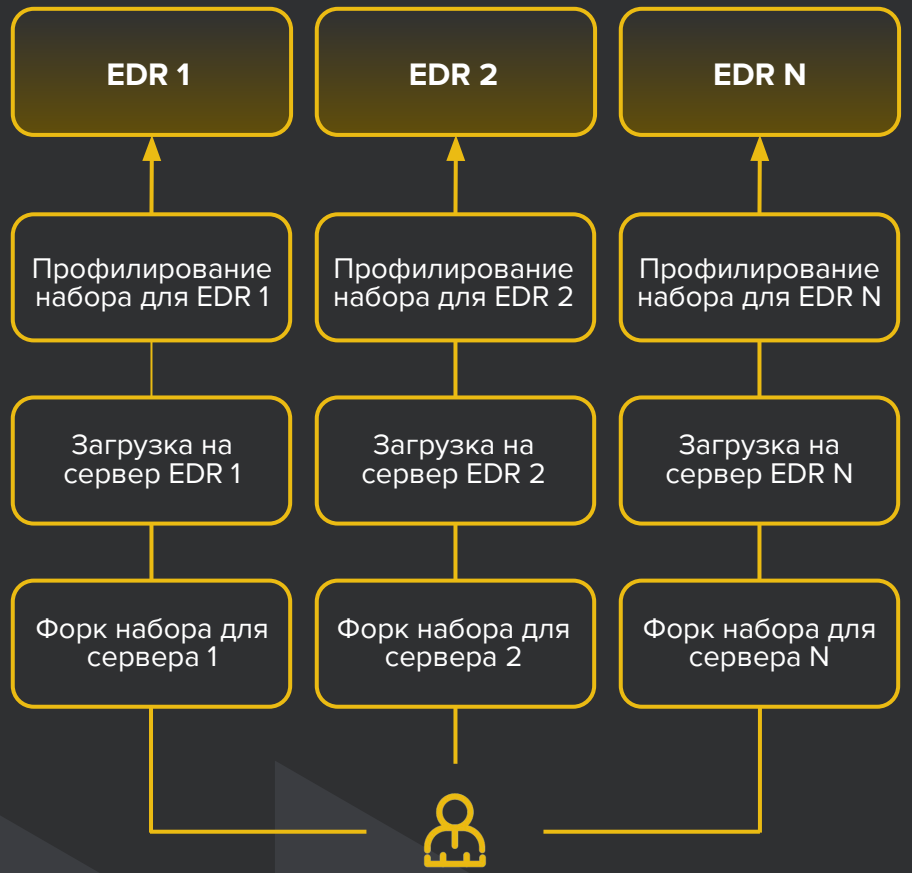
Вводные:

- 600+ Хостов
- Вторая половина дня пятницы
- Часть правил не тестировалась на реальной инфраструктуре

Полученный результат:

- Куча FP инцидентов
- Недовольная первая линия
- Нестабильная работа IRP

Все хорошо, но как это выглядит на схеме?



- Полностью ручная работа
- Профилирование после загрузки
- Высокая вероятность ошибки из-за человеческого фактора

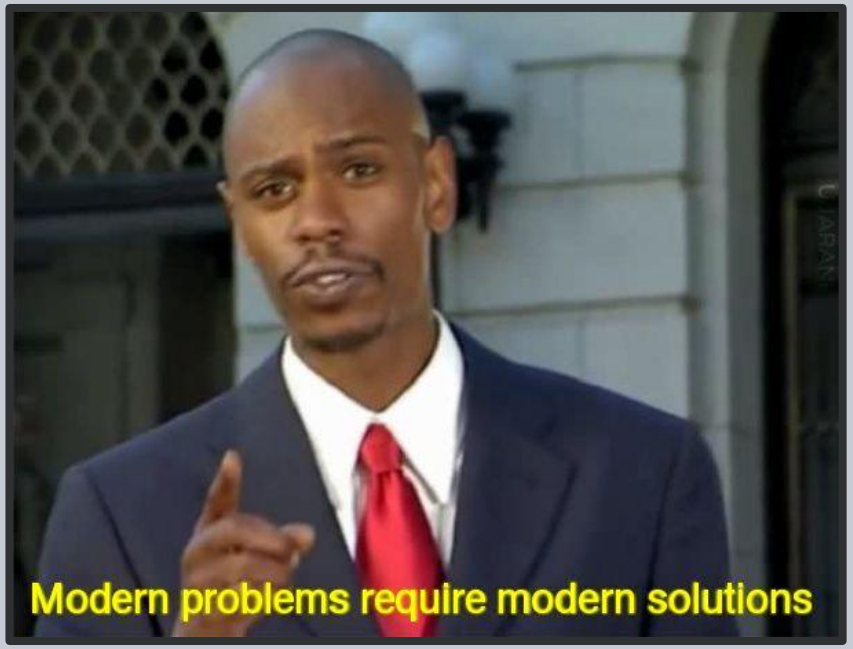
Недостатки данного способа

- ✖ Постоянное перепрофилирование обновленного набора экспертизы
- ✖ Отсутствие системы контроля версий для обновленного набора экспертизы
- ✖ Постоянная ручная загрузка нового набора экспертизы
- ✖ Должен быть эталонный набор экспертизы, который постоянно клонируется и модифицируется
- ✖ Правило из набора экспертизы может потеряться
- ✖ Отсутствие информации о том, кто и когда модифицировал набор экспертизы

Detection as Code как решение проблем

В рамках Detection as a Code решаются следующие проблемы:

- Знаем кто, когда и с каким PoC модифицировал набор экспертизы;
- Автоматизированное тестирование правил из наборов экспертизы;
- Реализация контроля версий для правил;
- Автоматизированная сборка, профилирование, проверка и загрузка обновленных наборов экспертизы.



Как мы попытались решить наши проблемы?

Первичная реализация процесса разработки и тестирования правил в SOC

1. Постановка задачи
по написанию правила

2. Разработка
правила и написание отчета

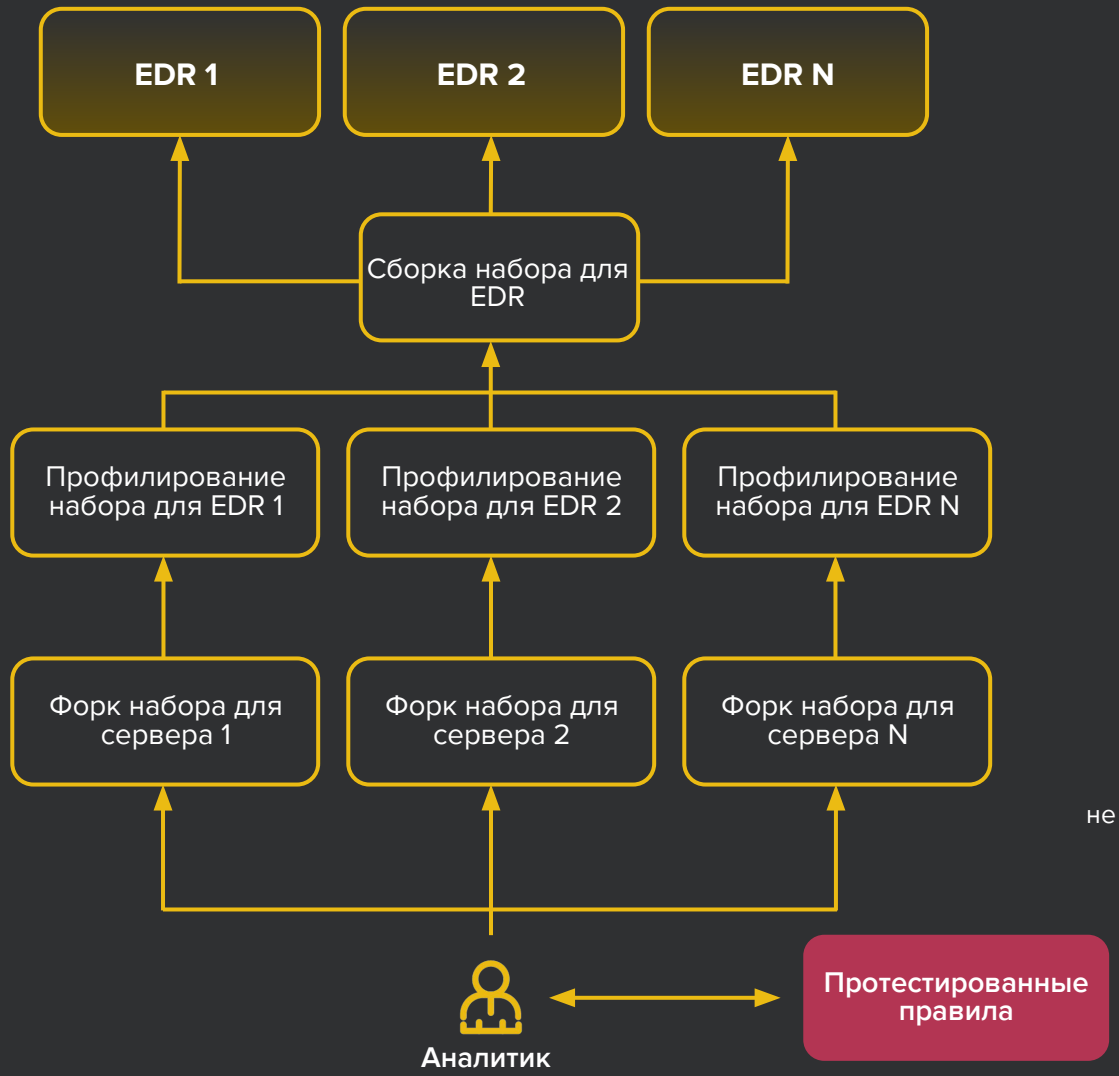
3. Загрузка
отчета и правил в GitLab

4. Тестирование
правил и написание отчета

5. Загрузка
правил, не прошедших тестирование

6. Добавление правил
не прошедших тестирование для переразработки

Все хорошо, но как это выглядит на схеме?



- Полностью ручная работа
- Профилирование перед загрузкой
- Все еще высокая вероятность ошибки из-за человеческого фактора

Основные задачи в рамках обновления наборов экспертизы при помощи GitLab CI/CD

- **Создание и регулярное обновление репозитория для правил;**
- **Создание идентификаторов для правил – соблюдение уникальности;**
- **При деплое для конкретного Заказчика:**
  - Сборка правил из репозитория;
  - Сравнение наборов экспертизы. Применение или отказ от изменений в наборах;
  - Тестирование правил на корректность;
  - Получение обновленного набора экспертизы;
  - Автоматическое профилирование правил.

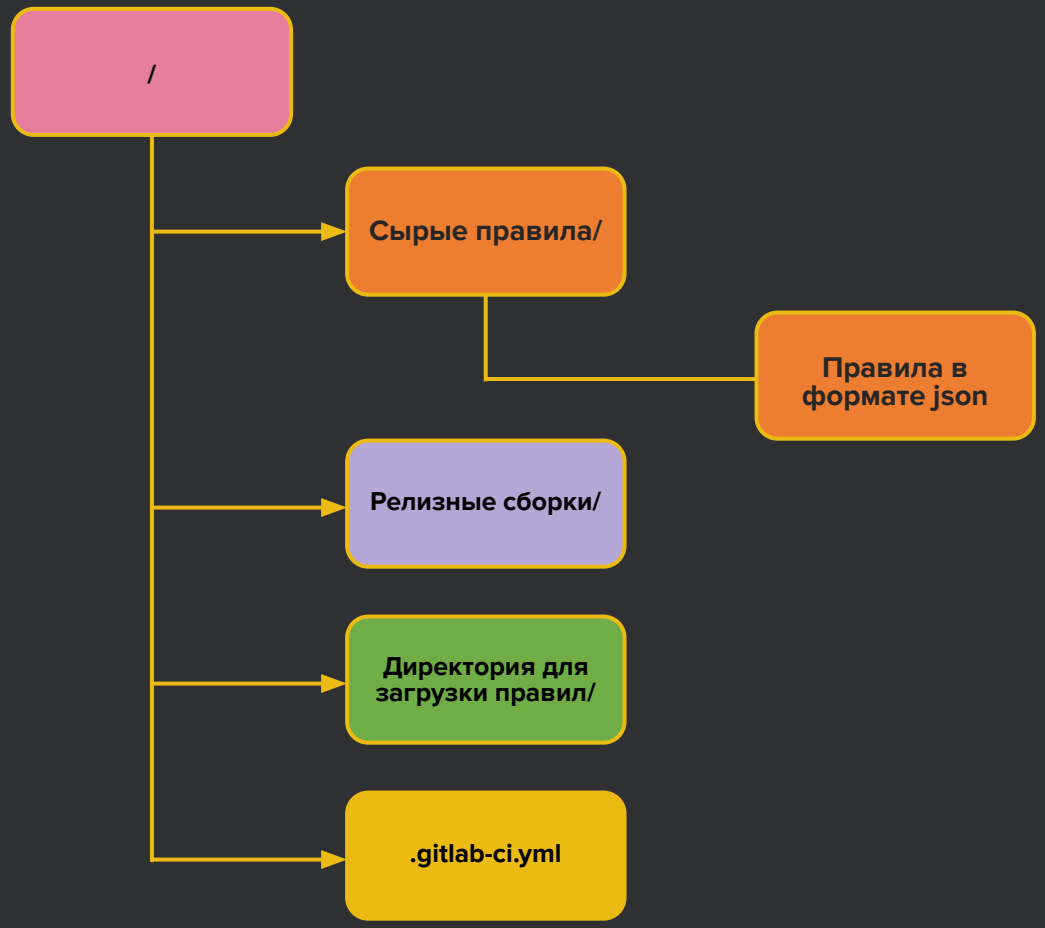
Создание отдельного репозитория для правил

Создание и регулярное обновление репозитория для правил

В рамках Detection as a Code решаются следующие проблемы:

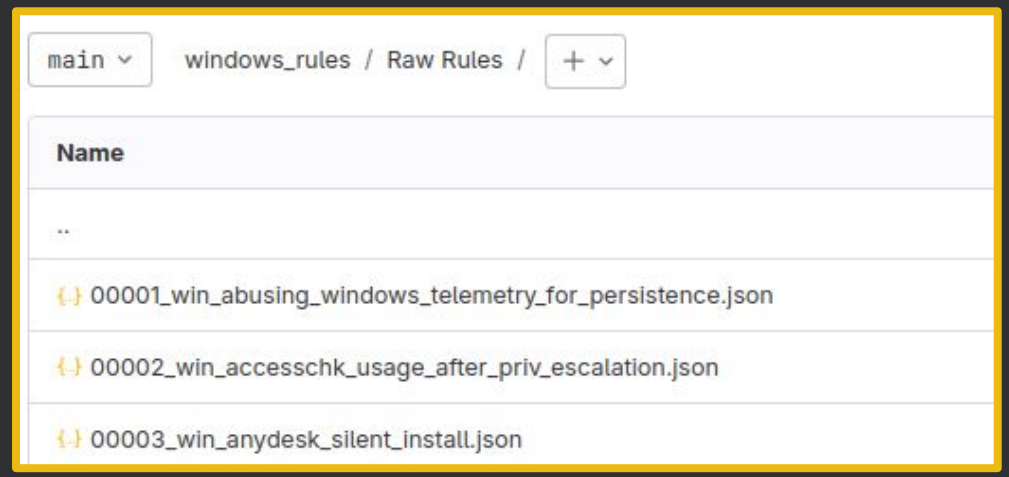
- В рамках DaaC были созданы репозитории для правил под каждую поддерживаемую платформу;
- Каждое правило при добавлении получает префикс, который является идентификатором правила;
- В данный репозиторий сохраняются все новые наборы экспертизы.

Немного про структуру репозиториев: Репозиторий с правилами

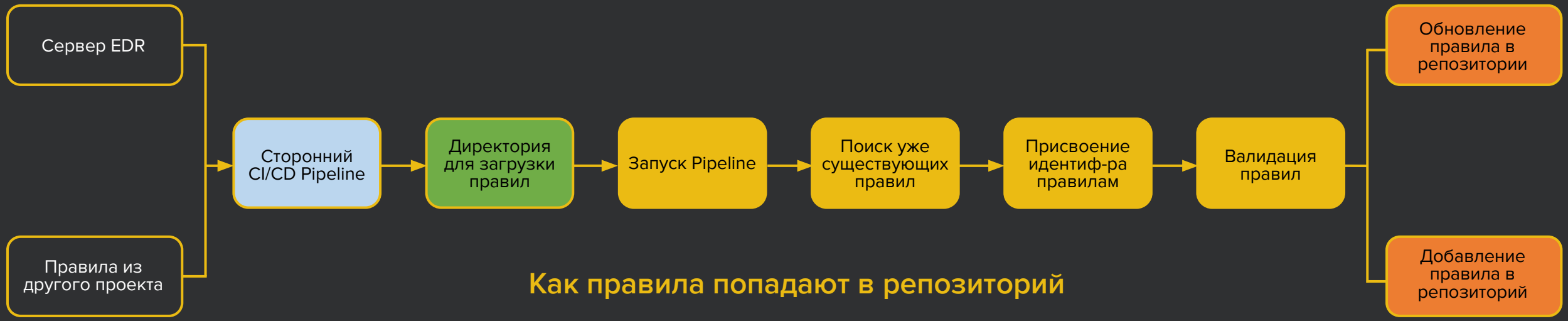


Структура репозитория для правил

Исходные правила в репозитории

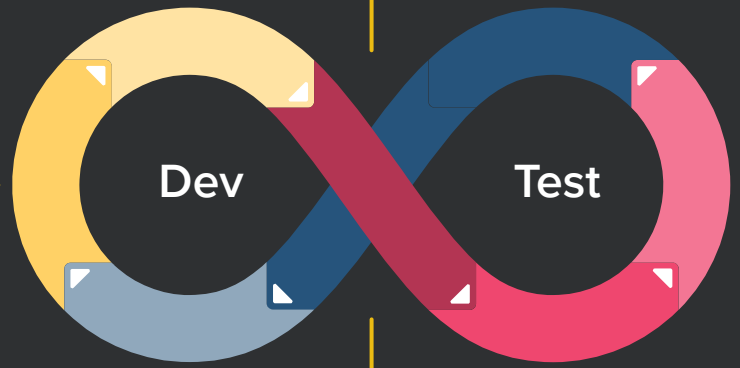


Немного про структуру репозиториев: Как правила попадают и обновляются в репозитории



Как правила попадают в репозиторий

Петля замкнулась



1. Постановка задачи
по написанию правила

2. Разработка
правила и написание отчета

3. Загрузка
отчета и правил в GitLab

4. Тестирование
правил, написание отчета и автотестирование*

5. Отправка правил
успешно прошедших тестирование в репозиторий GitLab

6. Загрузка
правил, не прошедших тестирование

7. Добавление правил
не прошедших тестирование для переразработки

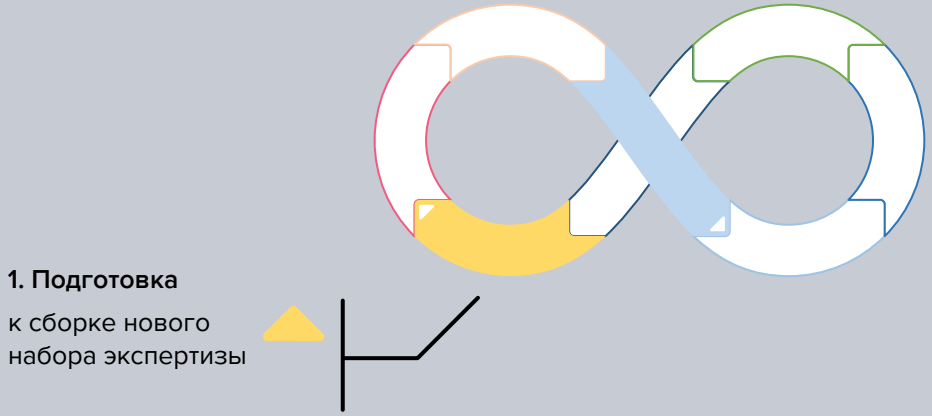
* На текущий момент данный этап пропускается, но все готово для интеграции с автотестами.

Pipeline, который решает проблемы

Основные элементы: Get Input

В рамках данного этапа выполняется:

- Выгрузка правил с сервера EDR*
- Сохранение правил в формате json*
- Получение опросного листа заказчика в формате markdown

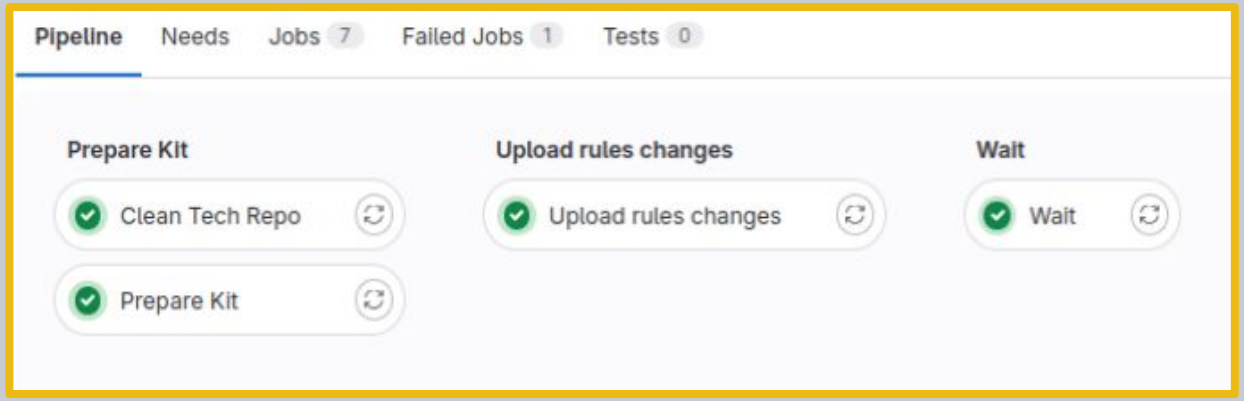


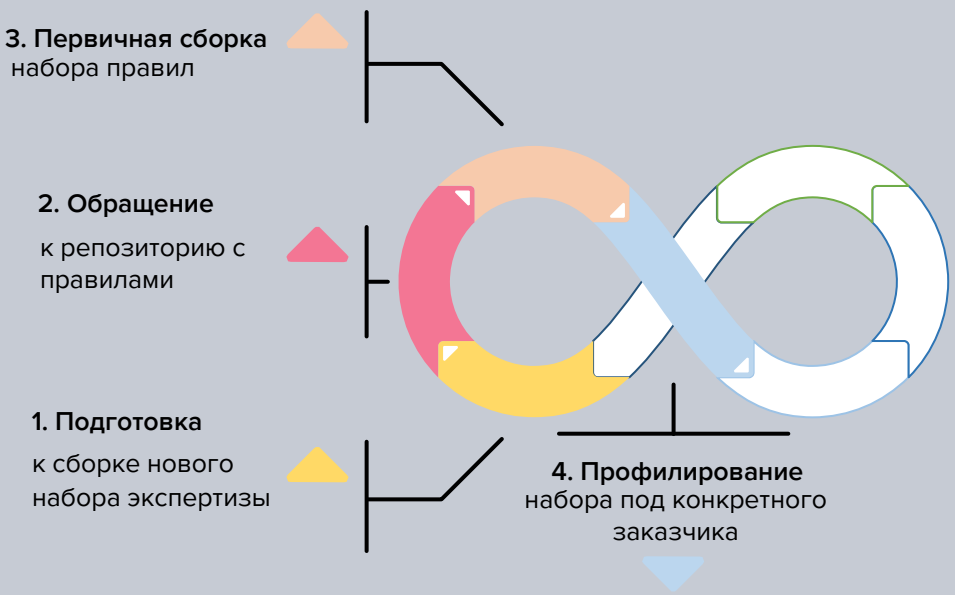
* Если это указано в конфигурационном файле.

Основные элементы: Prepare Kit, Upload rules changes и Wait

В рамках данных этапов выполняется:

- Первичная сборка набора экспертизы
- Поиск конфликтов, их загрузка в техническую ветку
- Уведомление пользователя
- Ожидание устранения пользователем конфликтов в правилах

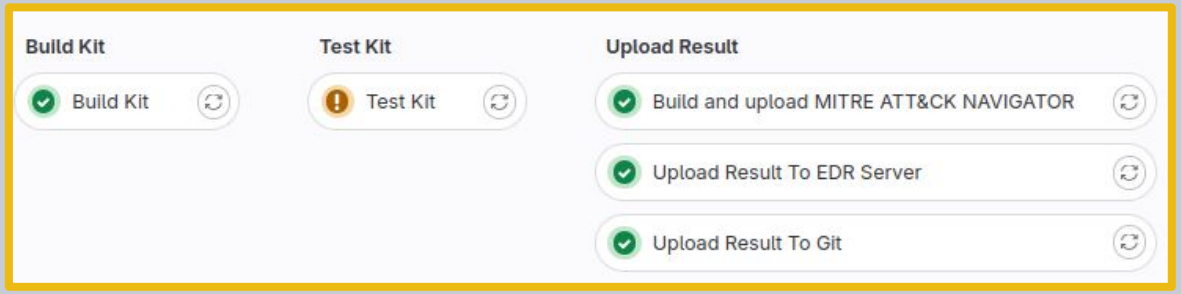




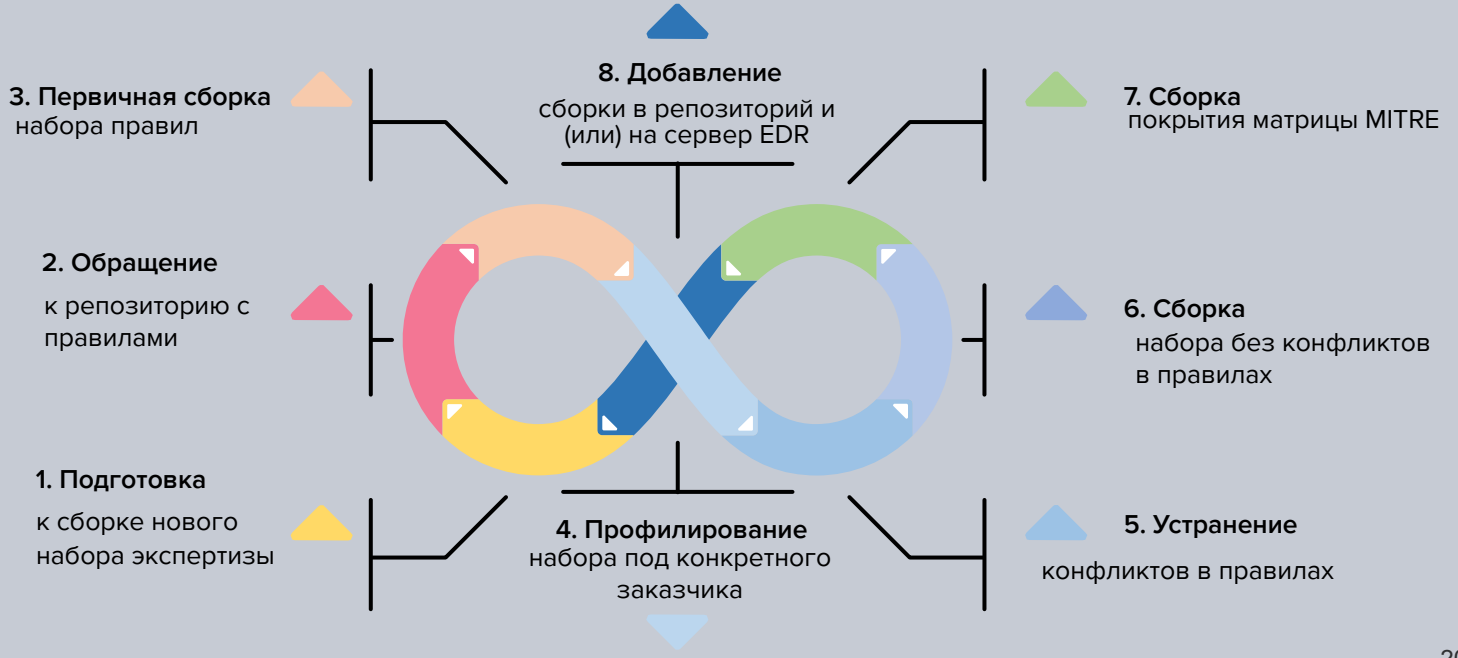
Основные элементы: Build Kit, Test Kit и Upload Result

В рамках данных этапов выполняется:

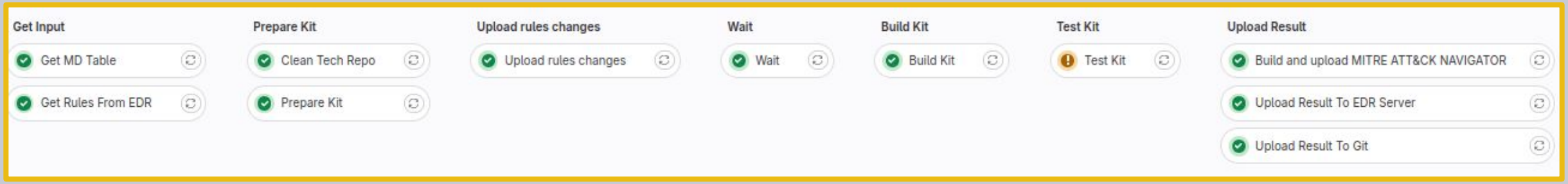
- Итоговая сборка набора экспертизы
- Проверка корректности правил
- Загрузка набора в репозиторий с правилами* и (или) на сервер EDR*



* Если это указано в конфигурационном файле.



Итоговый pipeline для сборки правил



Pipeline в GitLab CI

Петля замкнулась

Что нужно для запуска данного pipeline?

- Заполненный файл конфигурации RunConfig.yml
- В файл можно добавить несколько конфигураций

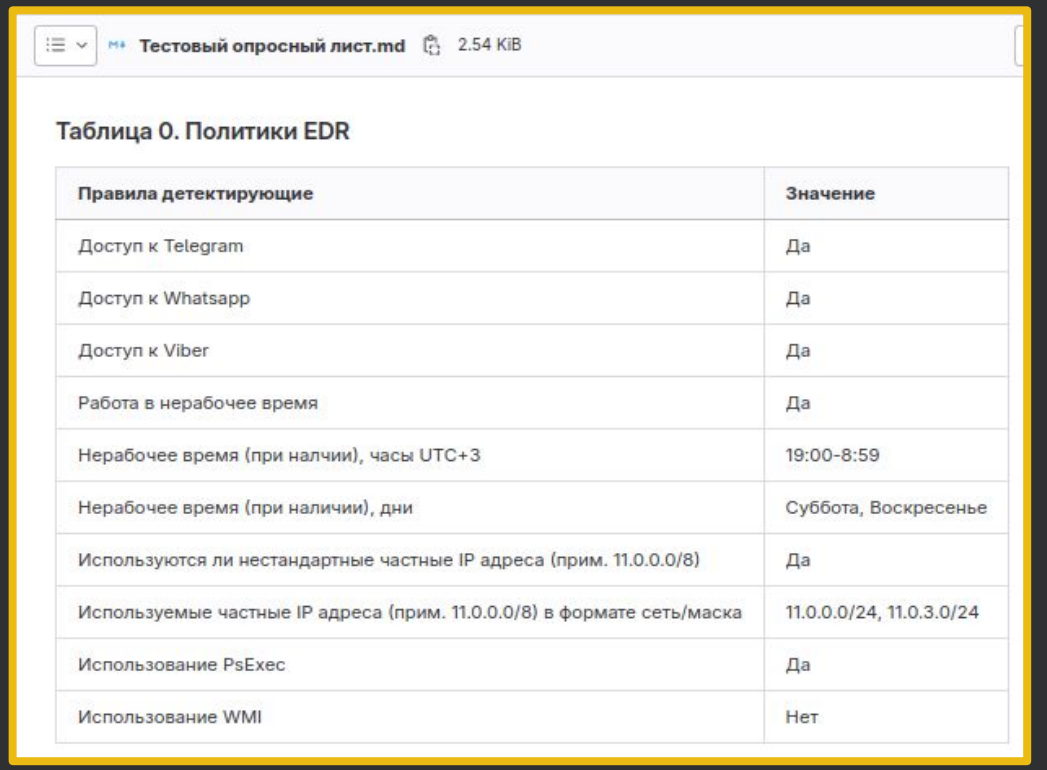
```yaml
RunConfig: TEST

TEST:
  GetOldRulesByAPI: True
  AddRulesToServerByAPI: True
  SaveRulesToGit: False
  Platform: "Windows"
  OldRules: "WindowsV1.json"
  RulesConf: "Configs/RulesConf.yml"
  OutputFileName: "WindowsWithCICD.json"
  EDRServerIP: 10.11.12.13
  EDRTokenVaultKey: "edr-test"
  QuestionnaireProject: test-gitlab.example.local/customers/questionnaire.git
  QuestionnairePath: "customers/test/Тестовый опросный лист.md"

Customer1:
  GetOldRulesByAPI: True
  AddRulesToServerByAPI: True
```

Пример RunConfig.yml

Что нужно сделать пользователю

Пользователю необходимо:

- Заполнить или отредактировать файл конфигурации RunConfig.yml и загрузить его GitLab;
- Заполнить опросный лист (если его еще нет);
- Дождаться письма о наличии конфликтов (при их наличии):
  - Перейти в технический репозиторий и заполнить 2 файла для решения конфликтов;
- Дождаться письма об успешной загрузке набора на сервер.



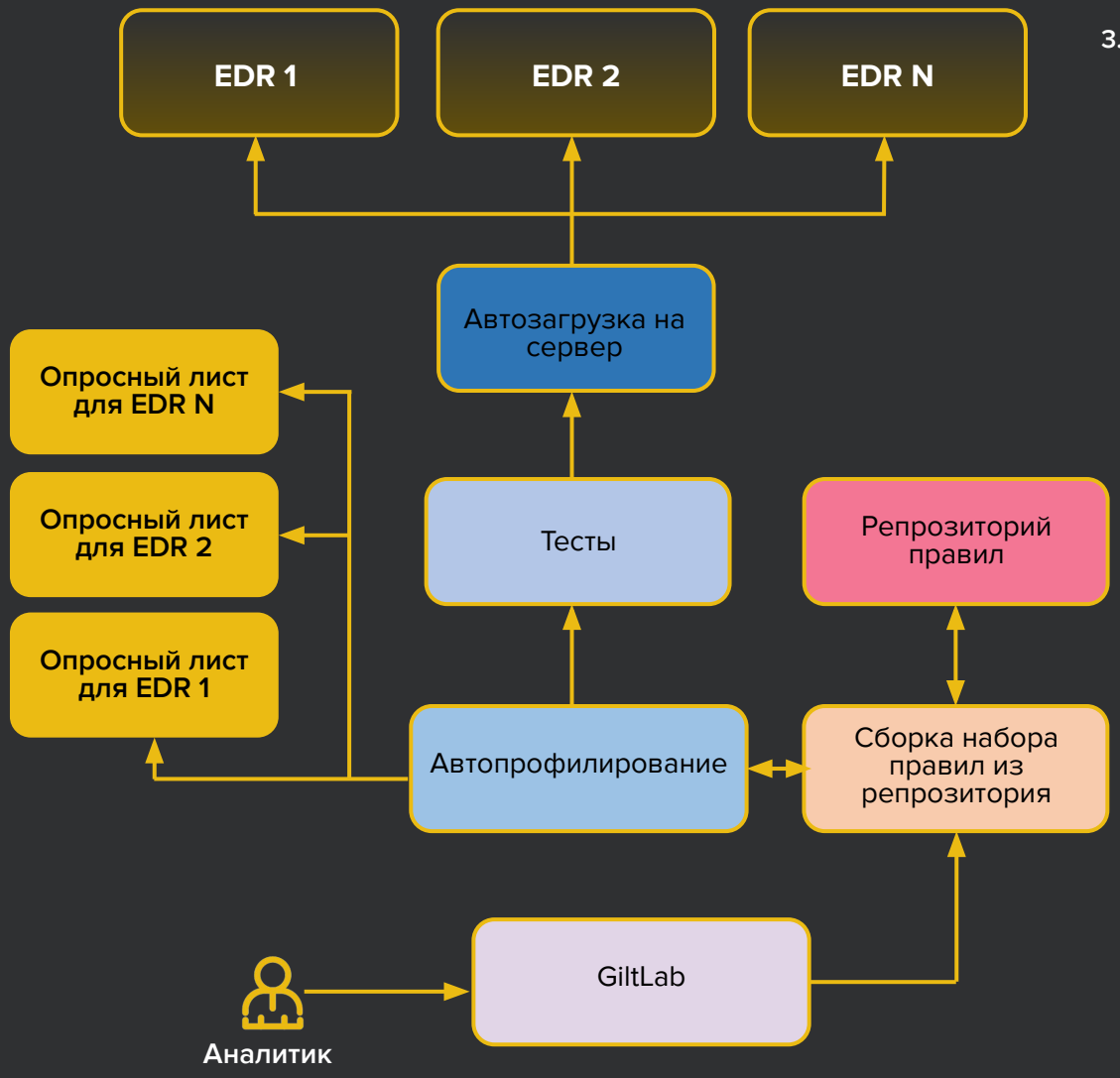
Тестовый опросный лист.md 2.54 KiB

Таблица 0. Политики EDR

Правила детектирующие	Значение
Доступ к Telegram	Да
Доступ к Whatsapp	Да
Доступ к Viber	Да
Работа в нерабочее время	Да
Нерабочее время (при налчии), часы UTC+3	19:00-8:59
Нерабочее время (при наличии), дни	Суббота, Воскресенье
Используются ли нестандартные частные IP адреса (прим. 11.0.0.0/8)	Да
Используемые частные IP адреса (прим. 11.0.0.0/8) в формате сеть/маска	11.0.0.0/24, 11.0.3.0/24
Использование PsExec	Да
Использование WMI	Нет

Пример опросного листа в формате markdown

Все хорошо, но как это выглядит на схеме?



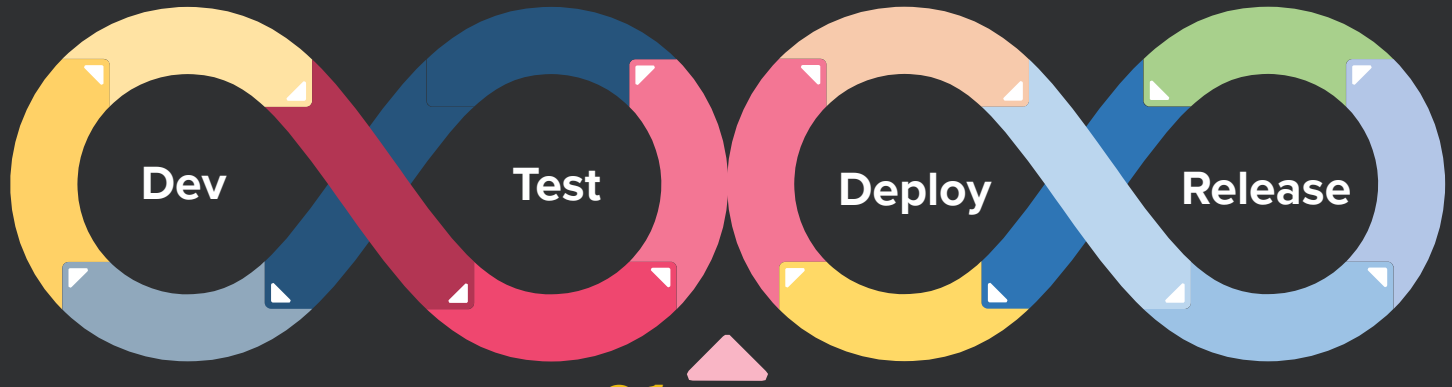
- Почти полностью автоматизированная работа;
- Вероятность ошибки из-за человеческого фактора сведена к минимуму;

Что на счет безопасности при работе pipeline`ов?

- В репозитории не хранятся аутентификационные данные
- Токены для доступа к серверу EDR хранятся на отдельном ресурсе, доступ к которому предоставляется по API
- В GitLab Variables содержатся только следующие данные:
    1. Токен технической учетной записи для работы с git
    2. Токен для доступа к хранилищу паролей по API
    3. Ссылка на хранилище с паролями (токенами)
    4. Аутентификационные данные для отправки сообщений через электронную почту

Итоги

Была 1 петля – теперь их 2



Обращение
к репозиторию с правилами

1. Постановка задачи
по написанию правила

2. Разработка
правила и написание отчета

3. Загрузка
отчета и правил в GitLab

4. Тестирование
правил, написание отчета и автотестирование

6. Загрузка
правил, не прошедших тестирование

7. Добавление
правил не прошедших тестирование для переразработки

1. Подготовка
к сборке нового набора экспертизы

3. Первичная сборка
набора правил

4. Профилирование
набора под конкретного заказчика

5. Устранение
конфликтов в правилах

6. Сборка
набора без конфликтов в правилах

7. Сборка
покрытия матрицы MITRE

8. Добавление
сборки в репозиторий и (или) на сервер EDR

Детализация работы pipeline

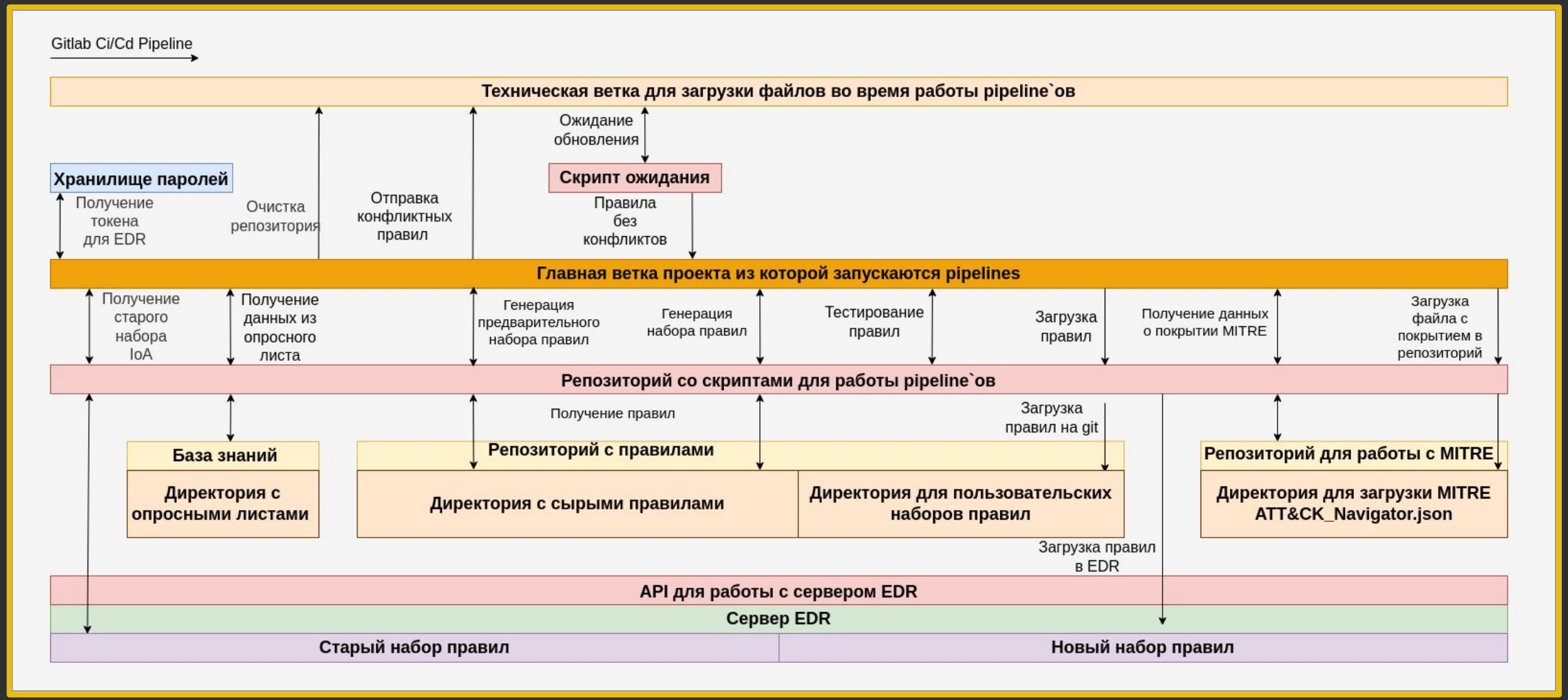
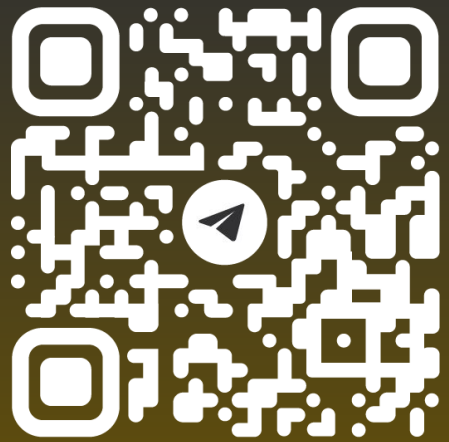


Схема работы Pipeline

Планы по дальнейшему развитию

- Добавление новых метрик
- Визуализация метрик в Grafana/Kibana
- Переход к полностью автоматизированным тестам
- Поддержка мультинаборности
- Реализация представленной схемы для правил SIEM
- Немного рефакторинга

Самое интересное из мира кибербезопасности у нас в Telegrem-канале!



@RTINFOBEZ

Нам доверяют

Ростех
РТ-Энергоэффективность
РТ
Развитие бизнеса
Ростех
Техприемка
Ростех
KAMAZ
ОАК
Объединенная авиастроительная корпорация
ОДК
КБП
СИБЕР
Нацимбио
Высокоточные комплексы
УВЗ
НОВИКОМБАНК

Контакты

Адрес: 117587, г. Москва, Варшавское шоссе, дом 118, корпус 1

Tel.: +7 (499) 390-79-05

E-mail: info@rt-ib.ru

Сайт: rt-ib.ru

РТ
Информационная безопасность