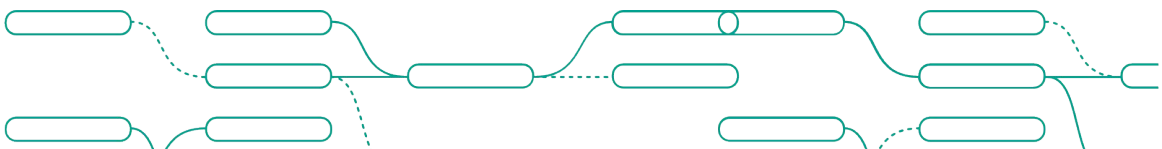
Arenadata Catalog

Arenadata Catalog

Инструкция по установке программного обеспечения

Москва 2025



Arenadata Catalog

Оглавление

1 Журнал изменений

Дата	Версия	Комментарий
25.10.2022	0.1	Начальная версия документа
16.01.2023	0.2	Актуализация с точки зрения инфраструктуры и способа поставки
25.01.2023	0.3	Добавление способа развертывания с учетом внешней БД
01.02.2023	0.4	Актуализация способа развертывания с учетом релиза v 0.2
13.07.2023	0.5	Добавлены версии поддерживаемых ОС и БД
17.07.2023	0.6	Добавлен п. 8 Обновление ПО, отредактирована часть с env файлами
21.07.2023	0.7	Актуализирован п. 8 Обновление ПО, добавлен п. 9 – Бэкап и восстановление БД, добавлено примечание для обновления с 0.3.1 до релиза 0.4.0
09.08.2023	0.8	Добавлено пояснение по предоставлению прав на ingestion-dags, способ изменения docker-compose для связи с airflow, способ загрузки докер образов из ya-storage в docker
31.08.2023	0.9	Обновлен п. Бэкап и восстановление
13.09.2023	1.0	п. Обновление версии в части индекса ES
11.10.2023	1.1	Дополнен п.9 по обновлению ПО, актуализированы поддерживаемые ОС
22.10.2023	1.2	Добавлен пункт Просмотр логов приложения
28.11.2023	1.3	Добавлен конфиг БД для увеличения кол-ва пользователей, конфиг открытия портов, описание настройки скриптов бэкапирования
13.02.2024	1.4	Keycloak & https, Monitoring
28.06.2024	1.5	Дополнен разделе 11.1 Keycloak ,изменен путь к сертификату в разделе 11.2
02.10.2024	1.6	Обновлен раздел Мониторинг, Компоненты системы.
05.11.2024	1.7	Обновлен раздел 4.Требования к установке Изменил версии компонентов Изменил команды бэкапы в 9.1 Дополнен раздел 9.2
06.02.2025	1.8	Актуализация Дополнен раздел 9.2 консольными командами Дополнен раздел 12

Arenadata Catalog

2 Введение

2.1 Термины и определения

Термин	Значение
База данных	Совокупность данных, хранимых в соответствии со схемой, манипулирование которыми выполняют в соответствии с правилами средств моделирования данных
Бизнес-глоссарий	Словарь для бизнес-пользователей. Словарь состоит из бизнес-терминов, которые могут быть связаны друг с другом, и позволяет распределять их по предметным областям, чтобы их можно было понимать в разных контекстах
SQL	Декларативный язык программирования, применяемый для создания, модификации и управления данными в реляционной базе данных, управляемой соответствующей СУБД
Data Lineage	Информация, которая описывает движение данных от источника их происхождения по точкам обработки и применения
Keycloak	Инструмент с открытым исходным кодом, используемый для аутентификации пользователей в организации, с возможностью настройки single sign-on и управления доступом

2.2 Сокращения и обозначения

Сокращение	Наименование
ПО	Программное обеспечение
СУБД	Система управления базами данных
БД	База данных
AD.C	Arenadata Catalog
УЗ	Учетная запись
ТП	Техническая поддержка

2.3 Общие положения

Инструкция предназначена для должностных лиц, осуществляющих установку программного обеспечения (ПО) Arenadata Catalog (ADC). Документ содержит пошаговое описание действий для установки специального программного обеспечения системы.

3 Назначение ПО

Основным предназначением ПО ADC является загрузка метаданных из различных систем обработки и анализа данных по всей компании с ведением и тесной интеграцией корпоративного Бизнес-Глоссария.

Программное обеспечение Arenadata Catalog обладает следующими характеристиками:

- Автоматический сбор метаданных из различных источников, включая возможность профилирования и получения примеров данных;
- Установка критериев качества данных и назначение проверок;
- Обогащение метаданных описанием и указанием владельцев;
- Визуализация происхождения данных — data lineage;
- Создания гибко настраиваемого Бизнес-глоссария;
- Управление объектами каталога метаданных и Бизнес-глоссария посредством настраиваемых рабочих процессов;
- Связывание объектов метаданных и Бизнес-глоссария;
- Полнотекстовый поиск информации в ADC.

4 Требования к установке

Для установки ADC по данной инструкции необходимо выполнение следующих требований:

1. Операционная система– Ubuntu 22.04.4 LTS, CentOS 7.9.2009, Ред ОС 7.3, ALT Server 10.1 (Mendelevium), Astra Linux Воронеж 1.7.
2. ПО Docker version 25.0.0, и выше
3. ПО Docker compose version 1.29.2, и выше
4. Пользователь с правами sudo.(или пользователь добавленный в группу docker)

Следует проверить версию установленного Docker Compose на соответствие требованиям.

В неподходящей версии Docker compose сборка не произойдет, так как используется версия контейнера 3.9.

В случае отсутствия необходимо произвести установку согласно инструкции по ссылке.

5 Подготовка к установке

Необходимо выполнить команду аутентификации на сервере репозитория, используя реквизиты полученные в технической поддержке:

```
docker login repo.arenadc.io -u login -p password
```

Далее необходимо загрузить файл Docker compose и enviroment файлы по ссылке предоставленной технической поддержкой.

В environment файлах:

- data-sec.json – отвечает за наполнение realm хранилища Keycloak модуля аутентификации(заполнять при использовании docker-compose-keycloak.yml), необходимо изменить поля связанные с host, на котором развернуто приложение (искать - localhost в секции "clientId" : "open-metadata"),
- adc.env – отвечает за параметризацию переменных docker-compose-keycloak.yml, необходимо изменить IP адреса в переменных окружения, отвечающих за аутентификацию через keycloak:

```
AUTHORIZER_CLASS_NAME=org.openmetadata.service.security.DefaultAuthorizer
AUTHORIZER_REQUEST_FILTER=org.openmetadata.service.security.JwtFilter
AUTHORIZER_ADMIN_PRINCIPALS=[admin]
AUTHORIZER_PRINCIPAL_DOMAIN=open-metadata.org
AUTHENTICATION_PROVIDER=custom-oidc
CUSTOM_OIDC_AUTHENTICATION_PROVIDER_NAME=KeyCloak
AUTHENTICATION_PUBLIC_KEYS=[http://host:8081/realms/data-
sec/protocol/openid-connect/certs,http://host/api/v1/system/config/jwks]
AUTHENTICATION_AUTHORITY=http://host:8081/realms/data-sec
AUTHENTICATION_CLIENT_ID=open-metadata
AUTHENTICATION_CALLBACK_URL=http://host:8585/callback
```

- adc_noauth.env – отвечает за параметризацию переменных docker-compose.yml необходимо изменить переменные окружения связанные с подключением к базе данных Postgres, при использовании внешней БД.

Указанные выше файлы необходимы для параметризации Docker compose и интеграции Keycloak с Data Catalog. Есть альтернативный способ настройки интеграции через web-интерфейс Keycloak, который находится на порту 8081. Также есть облегченная сборка compose без использования Keycloak.А параметризация нужна в первую очередь для порядка в переменных среды и развертывания и избежания хардкода в Docker compose файлах.

6 Установка ПО

Для работы приложения рекомендуется использовать директорию “/opt/adc/”.

Загруженную конфигурацию для docker compose поместить в директорию “/opt/adc/” и перейти в эту директорию.

```
cd /opt/adc/
```

Чтобы запустить ПО выполните в консоли одну из следующих команд:

```
sudo docker compose -f docker-compose.yml --env-file adc_noauth.env up -d
#basic authentication
#or
sudo docker compose -f docker-compose-keycloak.yml --env-file adc.env up -
d #keycloak authentication
```

Необходимо ожидание инициализации базы данных в пределах 10 минут.

В случае использования внешней БД PostgreSQL (версия 14.x) в выбранном из вышеперечисленных environment – файлов надо переопределить значения блока подключения к БД, а именно:

```
#Database configuration for server container
SERVER_DB_DRIVER_CLASS=org.postgresql.Driver
SERVER_DB_SCHEME=postgresql
SERVER_DB_USE_SSL=false
SERVER_DB_USER=openmetadata_user
SERVER_DB_USER_PASSWORD=openmetadata_password
SERVER_DB_HOST=postgresql
SERVER_DB_PORT:-5432
SERVER_DB_DATABASE=openmetadata_db
```

Перед установкой системы необходимо создать сущности в внешней БД, выполнив init.sql скрипт, что поставляется в комплекте с docker compose environment. Использовать скрипты необходимо только при условии выделенной PostgreSQL.

Примечание: в случае необходимости выполнения миграций по БД нужно учитывать факт наличия volume, и при наличии использовать команды: docker cp, docker exec, psql >.

Для контейнера БД также предусмотрено автоматизированное бекапирование с ротацией дампов (хранение за последние 3 дня), после установки приложения нужно выполнить настройку скриптов cron_backup.sh, backup_pg.sh:

```
cron_backup.sh #add/delete job with backup_pg.sh to cron
croncmd_backup="/opt/adc/backup_pg.sh"

backup_pg.sh   #create pg_dump with timestamp and rotate
# Директория для бекапов
backup_dir="/opt/adc/adc/db_backups"
# Название контейнера postgres
container_name='adc_postgresql_1'
```

Для контейнерной БД выставить лимит соединений в контейнере, в файле /var/lib/postgresql/data/postgresql.conf в зависимости от количества предполагаемых пользователей:

```
max_connections = 999
```

После установки приложения, для корректной загрузки DAG Airflow выставить права на папку с дагами:

```
sudo chmod 777 –R /opt/adc/ingestion-dags
```

Для корректного перехода на DAG airflow из Web UI ADC изменить docker-compose файлы, изменить ingestion на IP хоста с ADC:

```
# ADC Server Airflow Configuration
PIPELINE_SERVICE_CLIENT_ENDPOINT: ${PIPELINE_SERVICE_CLIENT_ENDPOINT:-
http://ingestion:8080}
```

6.1 Установка без доступа к репозиторию

Скачать образы приложения по предоставленным ссылкам.

Или выгрузить образы в архив локально:

```
docker save –o your_archive_name.zip repo_image_name:image_tag
```

Загрузить скаченные образы в docker images:

```
docker load < adc-db-vx.x.x.zip
```

После загрузки всех образов выполнить установку приложения согласно шагам, описанным в п. 6.

6.2 Открытие портов для доступа к airflow, db, camunda:

В целях безопасности в общей конфигурации docker compose закрыты порты для доступа из вне к БД, airflow, camunda. Открытие порта после установки может понадобится, например для работы через DBeaver с БД, просмотра времени выполнения импорта метаданных в airflow.

Необходимо для секции сервиса в docker compose добавить следующую инструкцию (на примере postgres):

```
services:
  postgresql:
    ...
    ports:
      - 5432:5432
    ...
```

6.3 Установка приложения без контейнера:

В случае невозможности использовать Docker контейнер,ядро приложения можно установить непосредственно в ОС Astra Linux 1.7.

В случае использования внешней БД PostgreSQL (версии между 14.x) в выбранном из вышеперечисленных .env – файлов надо переопределить значения блока подключения к БД, а именно:

```
#Database configuration for server container
SERVER_DB_DRIVER_CLASS=org.postgresql.Driver
SERVER_DB_SCHEME=postgresql
SERVER_DB_USE_SSL=false
SERVER_DB_USER=openmetadata_user
SERVER_DB_USER_PASSWORD=openmetadata_password
SERVER_DB_HOST=postgresql
SERVER_DB_PORT:-5432
SERVER_DB_DATABASE=openmetadata_db
```

Перед установкой системы необходимо создать сущности в внешней БД, выполнив init.sql скрипт, что поставляется в комплекте. Использовать скрипты необходимо только при условии выделенной PostgreSQL.

Для БД выставить лимит соединений в файле /var/lib/postgresql/data/postgresql.conf в зависимости от количества предполагаемых пользователей:

```
max_connections = 999 #recommended
```

Для контейнерной БД также предусмотрено автоматизированное бекапирование (если необходимо) с ротацией дампов (хранение за последние 3 дня), после установки приложения нужно выполнить настройку скриптов cron_backup.sh, backup_pg.sh:

```
cron_backup.sh #add/delete job with backup_pg.sh to cron
croncmd_backup="/opt/adc/backup_pg.sh"
backup_pg.sh   #create pg_dump with timestamp and rotate
# Директория для бекапов
backup_dir="/opt/adc/adc/db_backups"
# Название контейнера postgres
container_name='adc_postgresql_1'
```

После установки приложения, для корректной загрузки DAG Airflow содержимое папки указанной ниже в папку с дагами вашего Airflow:

```
# Prepare test Airflow DAGs if this need
# cp -a /ingestion_dags/. /opt/airflow/dags
```

Для корректного перехода на DAG airflow из Web UI ADC изменить .env файлы, изменить ingestion на IP хоста с ADC:

```
# ADC Server Airflow Configuration
PIPELINE_SERVICE_IP_INFO_ENABLED=false
PIPELINE_SERVICE_CLIENT_HOST_IP=""
PIPELINE_SERVICE_CLIENT_HEALTH_CHECK_INTERVAL=300
PIPELINE_SERVICE_CLIENT_VERIFY_SSL="no-ssl"
PIPELINE_SERVICE_CLIENT_SSL_CERT_PATH=""
PIPELINE_SERVICE_CLIENT_SECRETS_MANAGER_LOADER="noop"
AIRFLOW_USERNAME=admin
AIRFLOW_PASSWORD=admin
```

```
AIRFLOW_TIMEOUT=10
AIRFLOW_TRUST_STORE_PATH=""
AIRFLOW_TRUST_STORE_PASSWORD=""
```

Для корректной работы Camunda BPM platform из Web UI ADC изменить .env файлы, изменить IP хоста :

```
# ADC Server Camunda Configuration
CAMUNDA_API_URL="http://10.80.0.3:7070/engine-rest"
CAMUNDA_API_LOCK_DURATION="1000"
CAMUNDA_TENANT_ID="datacatalog"
DB_URL="jdbc:postgresql://10.80.0.3:5432/camunda_db"
DB_DRIVER="org.postgresql.Driver"
DB_USERNAME="camunda_db_user"
DB_PASSWORD="ZCV580mfkf(&)%$!?apsn"
DB_CONN_MAXACTIVE=20"
DB_CONN_MINIDLE="5"
DB_CONN_MAXIDLE="20"
DB_VALIDATE_ON_BORROW="false"
DB_VALIDATION_QUERY="SELECT"
```

Для запуска приложения в приложенных .env файлах окружения необходимо заполнить значения относящиеся к инфраструктуре вокруг приложения, как указанно выше инструкции(заменяем значения НЕ локальных сервисов портов и логинов и паролей), далее необходимо запустить скрипт инициализации приложения:

```

# extract application
cd /opt && tar zxvf adc-*.tar.gz
# change enviroment variables before run
# keycloak authentication
./bare-deploy-adc.sh adc.env
# or
# basic authentication
./bare-deploy-adc.sh adc-noauth.env
```

7 Проверка доступности сервиса

Понять о состоянии сервисов можно посредством команд:

```
docker ps –a
docker compose ps -a
docker compose logs <container name>
```

После инициализации БД запускаем Arenadata Catalog в web-браузере по адресу:

```
http://[server_address]:8585
```

В случае технической необходимости остановить сервисы и/или переустановить сервисы используем следующие команды:

```
cd /opt/adc                                  #переход в рабочую дирректорию
docker compose “chosen env files” stop   #остановить все контейнеры
docker volume prune                          #очистить все тома приложения
docker container prune                       #удалить остановленные контейнеры
```

8 Обновление ПО

Обновление приложения может быть произведено несколькими способами, в зависимости от наличия внешней базы данных, необходимости восстановления новой версии из бэкапа. Дополнительно рекомендуется перед обновлением ПО выполнить бэкап сервера или виртуальной машины, на которой установлено приложение, а также выполнить бэкап базы данных (п. 9).

Важно: перед обновлением все термины должны быть утверждены, мы не гарантируем корректное обновление, если какой-то из терминов находится в статусе «Кандидат».

Важно: перед обновлением необходимо удалить индекс Elasticsearch предыдущей версии (удалить volume ES).

8.1 Обновление при наличии внешней базы данных Postgres (не контейнерной из комплекта поставки)

Важно: во избежание потери данных - обязательно выполнить бэкапирование внешней БД доступными средствами или по п. 9 данной инструкции.

- Отключить старую версию программы выполнив:

```
sudo docker-compose -f docker-compose.yml --env-file adc_noauth.env down

или

sudo docker-compose -f docker-compose-keycloak.yml --env-file adc.env down
```

- Загрузить новую версию ПО по предоставленной ссылке.
- Изменить конфигурационные файлы acd.env или adc_noauth.env для подключения к внешней базе (п. 6 инструкции).
- Включить приложение (п. 6 инструкции).

8.2 Обновление с использованием бэкапа базы предыдущей версии ПО

- **Во избежание потери данных - обязательно сделать бэкап БД (п. 9.1 инструкции);**
- Остановить приложение в соответствие с п. 8.1;
- Установить новую версию по (п. 6 инструкции);
- Включить только postgres;
- Выполнить восстановление базы данных из бэкапа (п. 9.2 инструкции).

8.3 Настройка ротации логов Loki при установке, обновлении.

Важно: при обновлении приложения с версий 0.4.x, 0.5.0, для настройки ротации логов в Loki выполнить: бэкап вольюма adc-loki-data предыдущей версии приложения.

- После бэкапирования вольюма adc-loki-data – удалить его

- В файле adc.env (adc_noauth.env) установить переменной окружения – значение

```
LOKI_CONFIG_FILE="loki-config.yaml"
```

- По умолчанию ротация логов настроена на период - 30 дней. Параметры loki-config.yaml отвечающие за период ротации:

```
limits_config:  retention_period: 30d
reject_old_samples_max_age: 30d
max_look_back_period: 30d
retention_period: 30d
```

- Запустить приложение (п. 6)

8.4 Troubleshooting. Ошибка при старте сервера после обновления связанная с short_name.

Важно: В случае если после обновления приложения при старте контейнера server в логах возникает ошибка связанная с short name

```
org.openmetadata.service.exception.SecretsManagerUpdateException:
Unrecognized field "shortName" (class
org.openmetadata.schema.type.EntityReference), not marked as ignorable (10
known properties: "version", "deleted", "type", "id", "description",
"fullyQualifiedName", "payload", "name", "displayName", "href"])
 at [Source: (String)"{"id": "1ffec450-6553-48b3-b9f6-ac1924f1d5ff", "name":
"ogranovskaya_vtb", "owns": [{"id": "54dd8a4e-4705-441f-92e6-f70a3198802c",
"name": "111", "type": "adcSubjectArea", "deleted": false, "shortName":
"111", "description": "111", "displayName": "111", "fullyQualifiedName":
"Глоссарий_ВТБ.Информационные_системы.111"}, {"id": "dba78c57-6df1-43e0-b18d-
62612fd28aa8", "name": "PROFILE", "type": "glossaryTerm", "deleted": false,
"shortName": "PROFILE", "description": "Profile", "displayName":
"Profi"[truncated 3301 chars]; line: 1, column: 205] (through reference
chain:
org
.openmetadata.schema.entity.te
ams.User["owns"]->j
ava.util.ArrayList[0]->org.openmetadata.schema.type.Ent
ityReference["shortName"])
   at
org.openmetadata.service.secrets.SecretsManagerUpdateService.retrieveBotUsers
(SecretsManagerUpdateService.java:211)
   at
org.openmetadata.service.secrets.SecretsManagerUpdateService.updateBotUsers(S
ecretsManagerUpdateService.java:94)
   at
org.openmetadata.service.secrets.SecretsManagerUpdateService.updateEntities(S
ecretsManagerUpdateService.java:76)
   at
org.openmetadata.service.OpenMetadataApplication.run(OpenMetadataApplication.
java:256)
   at
org.openmetadata.service.OpenMetadataApplication.run(OpenMetadataApplication.
java:168)
   at io.dropwizard.cli.EnvironmentCommand.run(EnvironmentCommand.java:67)
```

```
   at io.dropwizard.cli.ConfiguredCommand.run(ConfiguredCommand.java:98)
   at io.dropwizard.cli.Cli.run(Cli.java:78)
   at io.dropwizard.Application.run(Application.java:94)
   at
org.openmetadata.service.OpenMetadataApplication.main(OpenMetadataApplication
.java:765)
```

- Необходимо выполнить следующий скрипт на служебной БД adc (openmetadata_db)

```
UPDATE user_entity SET json = jsonb_set
   (
         json,
         '{owns}',
         (select json_agg(individual_object - 'shortName') from
jsonb_array_elements(json->'owns') as individual_object)::jsonb
   )
where json -> 'owns' is not null
and jsonb_array_length(json->'owns') > 0
and json->>'owns' like '%shortName%';
```

Перезапустить приложение (п. 6)

9 Бэкап и восстановление данных

9.1 Для бэкапа данных приложения выполнить шаги:

- Создать бэкап базы данных (вручную):

```
sudo docker exec adc-postgresql-1 pg_dump -U postgres -Fc -d
openmetadata_db > /opt/adc/db_backups/dump_openmetadata.dump

sudo docker exec adc_postgresql_1 pg_dump -U postgres -Fc -d airflow_db >
/opt/adc/db_backups/dump_air.dump

sudo docker exec adc_postgresql_1 pg_dump -U postgres -Fc -d camunda_db >
/opt/adc/db_backups/dump_camunda.dump
```

- Настроить и запустить скрипт для создания бэкапов: ./backup_pg.sh
- (Необязательный шаг) При необходимости сделать бэкапы сгенерированных DAGs для ingestions и их config JSON-файлов (после восстановления DAG можно переустановить в UI Arenadata Catalog):

```
sudo docker cp adc_ingestion_1:/opt/airflow/dags /opt/adc/db_backups/dags

sudo docker cp adc_ingestion_1:/opt/airflow/dag_generated_configs
/opt/adc/db_backups/dag_generated_configs
```

9.2 Для восстановления состояния приложения из бэкапа выполнить:

- Поднять только postgres:

```
sudo docker-compose -f docker-compose-keycloak.yml --env-file adc.env up -
d postgresql
```

- Удалить и создать схемы public для баз openmetadata_db, camunda_db:

Docker exec –it adc-postgresql-1 bash
psql -U postgres -d openmetadata_db

```
DROP SCHEMA IF EXISTS public CASCADE;
CREATE SCHEMA IF NOT EXISTS public;
ALTER SCHEMA public OWNER TO openmetadata_user (camunda_db_user);

\c camunda_db

DROP SCHEMA IF EXISTS public CASCADE;
CREATE SCHEMA IF NOT EXISTS public;
ALTER SCHEMA public OWNER TO camunda_db_user;
```

- Загрузить файл с бэкапом базы данных в контейнер postgres:

```
sudo docker cp /opt/adc/db_backups/dump_openmetadata.dump <postgres container>:/home
sudo docker cp /opt/adc/db_backups/dump_camunda.dump <postgres container>:/home
```

Выполнить загрузку данных в базу данных из файла бэкапа:

-
```
sudo docker exec adc-postgresql-1 pg_restore -C -U postgres -d openmetadata_db /home/dump_openmetadata.dump
```
-
```
sudo docker exec adc-postgresql-1 pg_restore -C -U postgres -d camunda_db /home/dump_camunda.dump
```

- (Необязательный шаг) При необходимости восстановить загрузить бэкапы DAGs и их config JSON-файлы в контейнер ingestion.

```
sudo docker cp /opt/adc/db_backups/dags adc_ingestion_1:/opt/airflow/dags

sudo docker cp /opt/adc/db_backups/dag_generated_configs adc_ingestion_1:/opt/airflow/dag_generated_configs
```

- Включить контейнеры Arenadata Catalog

- **ОБЯЗАТЕЛЬНО !!!**

  Выполнить переиндексацию; кнопка “Переиндексировать все” в разделе Настройки-Поиск.В противном случае поиск работать не будет

- Создать новый токен для ingestion bot. Зайти в Настройки/Боты выбрать и нажать кнопку Отозвать токен.

- В Настройки/Сервисы, Аналитика, Поиск (переиндексация по расписанию зайти на все существующие сервисы, выполнить переустановку процессов загрузок (ingestions).

9.3 Перенос / бэкапирование данных Keycloak и других данных приложения.

Перенос других данных приложения таких как данные пользователей Keycloak, логи, ДАГи Airflow их их конфигурации, может быть произведен путем подстановки volume docker. Ниже представлен пример по переносу docker volume Keycloak.

Важно: НЕ переносить БД таким образом, выполнять только через бэкап и восстановление БД.

- Войти под пользователем с правами sudo
- Скопировать папку с данными предыдущего инстанса Keycloak:

```
cp -r /var/lib/docker/volumes/adc{old_folder_name}_keycloak-data/_data /your_folder
```

- Запустить новую версию приложения, включится новый инстанс Keycloak, если запуск происходил из папки отличной от предыдущей версии приложения, создастся новый volume.
- Скопировать папку _data в новый volume:

```
cp -r /your_folder/_data
/var/lib/docker/volumes/adc{new_folder_name}_keycloak-data/
```

- Перезапустить adc:

```
sudo docker-compose -f docker-compose-keycloak.yml --env-file adc.env down
sudo docker-compose -f docker-compose-keycloak.yml --env-file adc.env up -
d
```

- Проверить перенос пользователей в UI Keycloak.

10 Мониторинг логов приложения в Grafana

10.1 Мониторинг логов

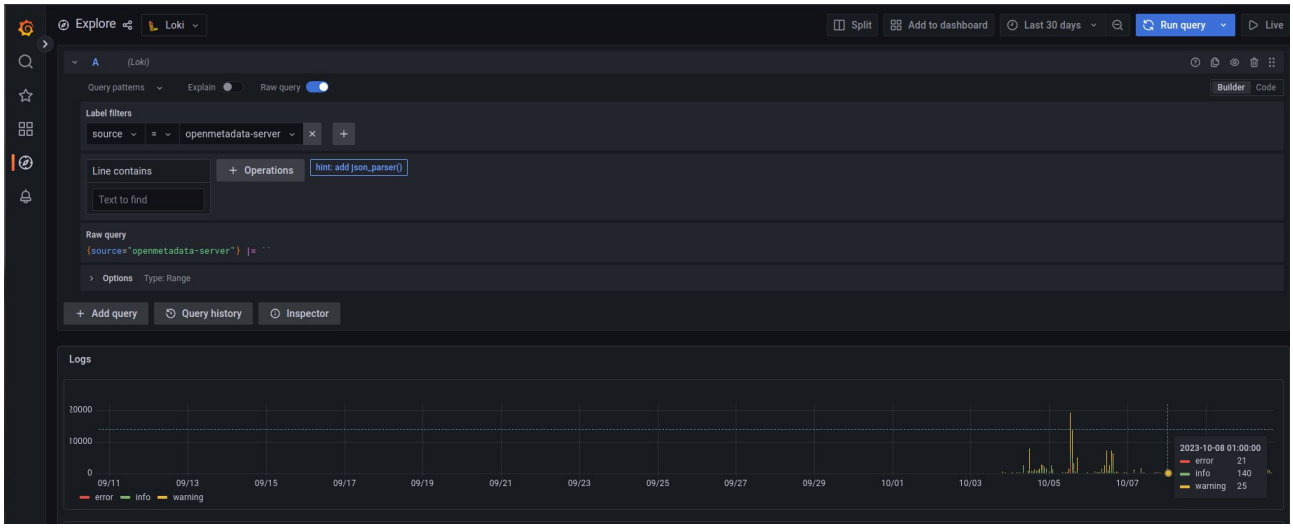
1. Перейти в grafana по адресу: <ip-address/host-name>:3000, в меню слева выбрать вкладку Explore, по умолчанию логин и пароль: admin/admin (при первом входе необходимо сменить пароль).

2. В выпадающем меню выбрать Loki.

3. В кладке Label filters выбрать “app”.

4. После условия “=” выбрать adc-rest-server.

5. Запустить запрос нажав в левом верхнем углу кнопку “Run Query”, при необходимости выставить временной интервал.

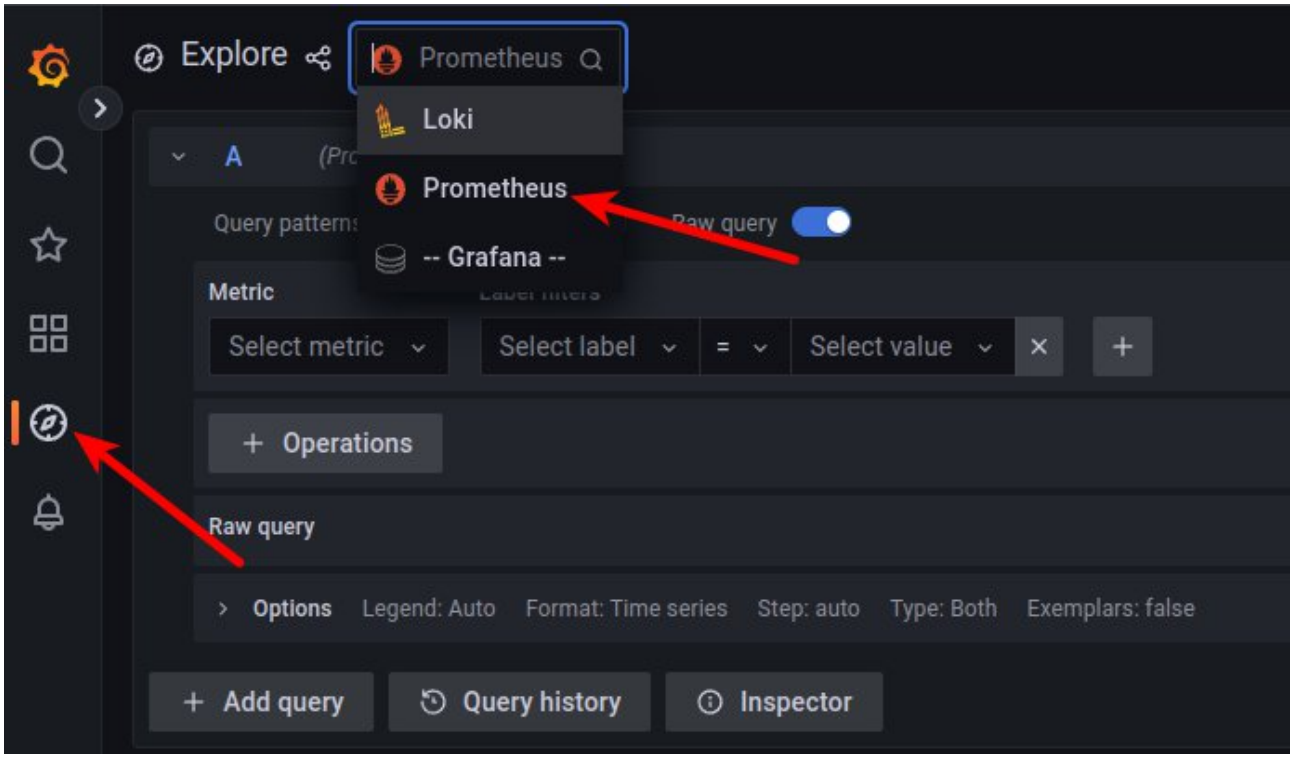


6. Дополнительно: для просмотра логов можно воспользоваться командой: docker logs <container_name> в терминале.

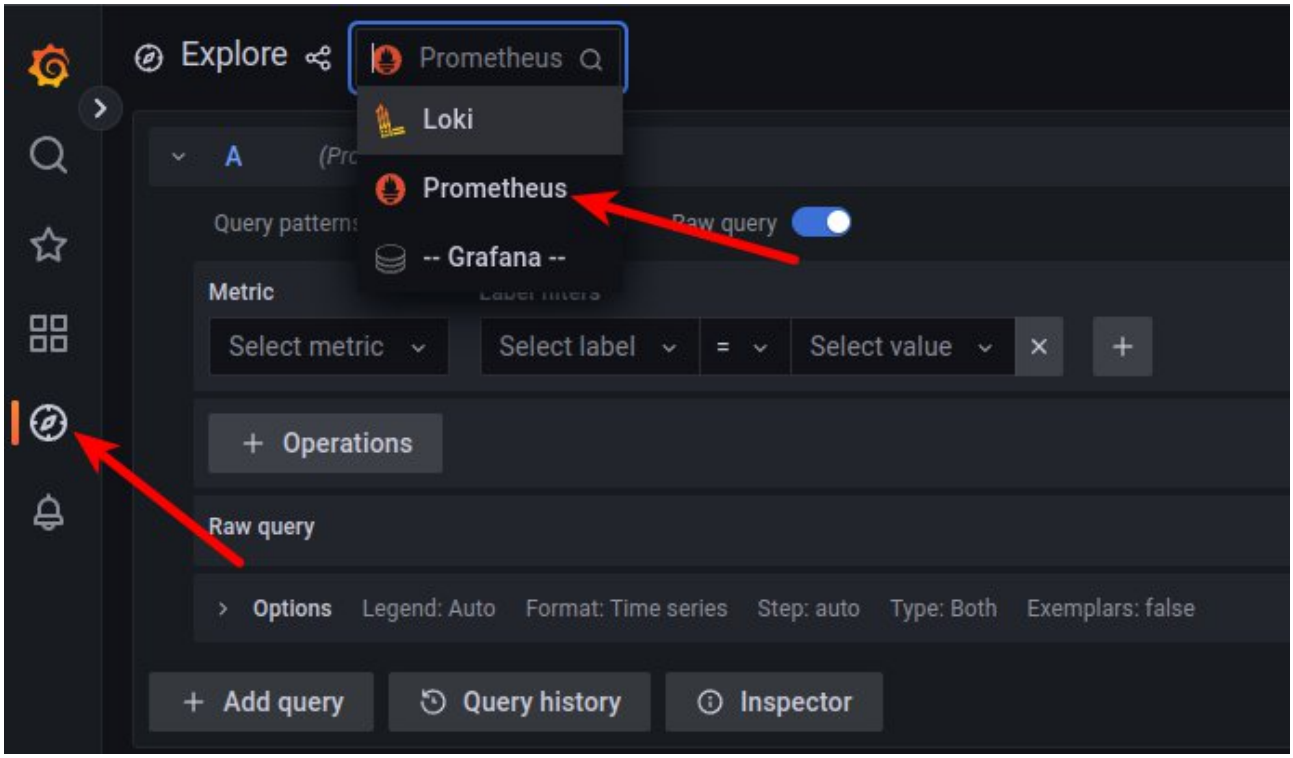
10.2 Метрики

С релиза 0.6.0 в графане доступны метрики (если они включены в compose файле - переменная ADC_CONFIGURATION_EXCLUSIONS не должна включать “conf-metrics-prod.yaml“).

Метрики доступны через раздел Explore в источнике Prometheus



Так же в Графане есть одна витрина с метриками по SQL запросам:



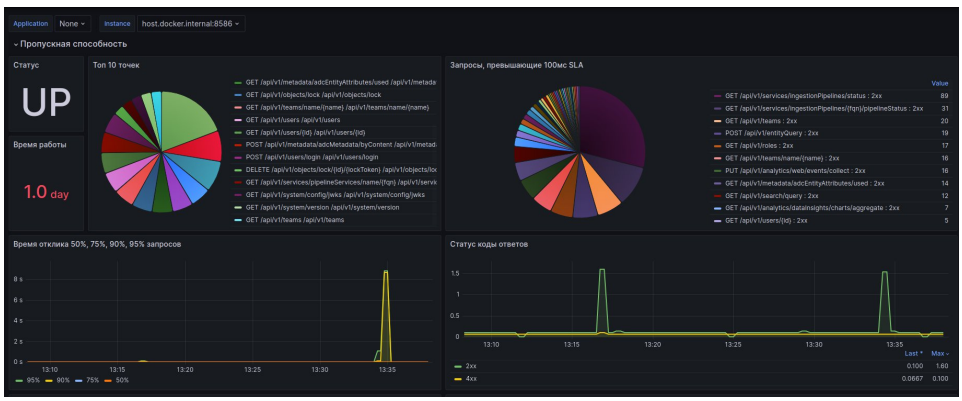
Описание некоторых метрик:

adc_sql_read_*	Семейство метрик по запросам на чтение (в метках к метрикам указана дополнительная информация, такая как класс и метод в коде которые были источником запроса)
adc_sql_write_*	Семейство метрик по запросам на запись (в метках к метрикам указана дополнительная информация, такая как класс и метод в коде которые были источником запроса)
adc_sql_unknown_*	Семейство метрик по запросам, которые не удалось идентифицировать как “на чтение“ или “на запись“ (в метках к метрикам указана дополнительная информация, такая как класс и метод в коде которые были источником запроса)
adc_jvm_memory_*	Семейство метрик памяти JVM
system_cpu_usage process_cpu_usage	Утилизация процессора - системой и ADC сервисом соответственно.
adc_health_aggregate_healthy	Статус работы сервера: 1 - Всё хорошо 0 - Одна или несколько подсистем не работает (Camunda, БД) Наименования метрик, как и сам набор отсылаемых метрик, будет меняться в ближайшем будущем.

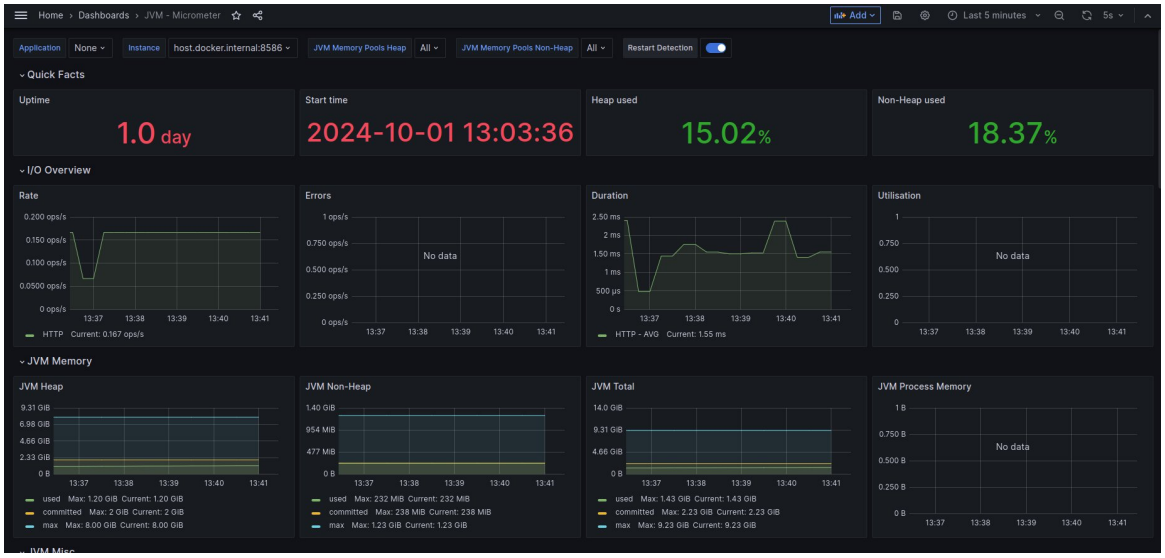
Наименования метрик, как и сам набор отсылаемых метрик, будет меняться в ближайшем будущем.

Доступны дашборды Grafana

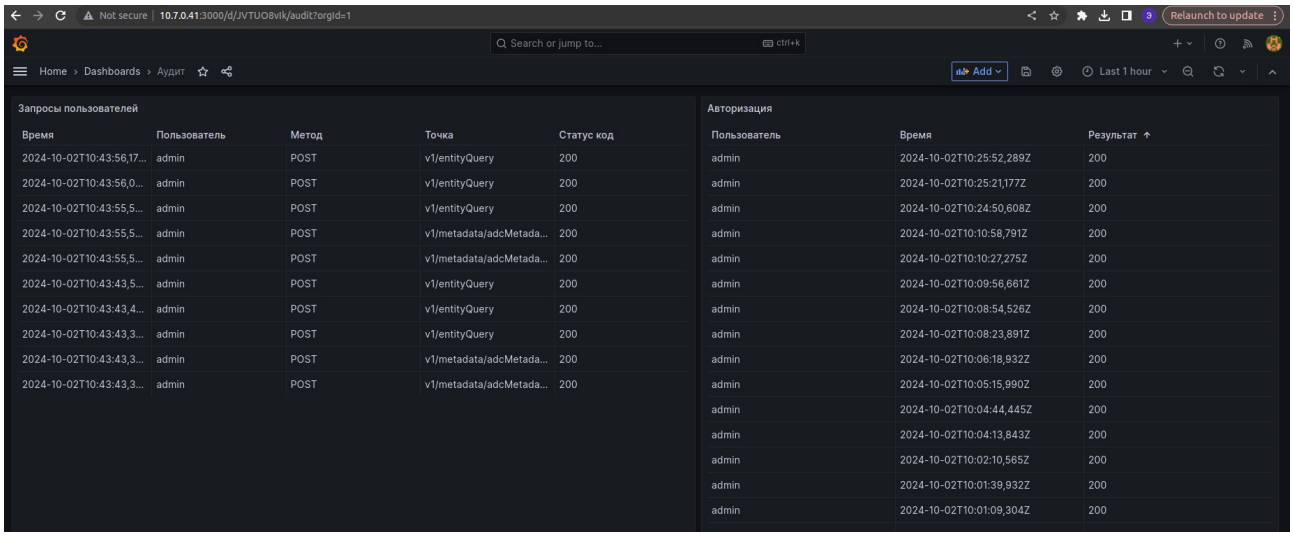
1. ADC REST Server – статус работы приложения up/down, время работы, метрики по статистке запросов.



2. JVM - Micrometer – метрики JVM – heap memory, GC, операции ввода/вывода и тд.



3. Дашборд Аудита – выведена информация о каждом действии пользователя, логин пользователя, время действия. Авторизация пользователей.

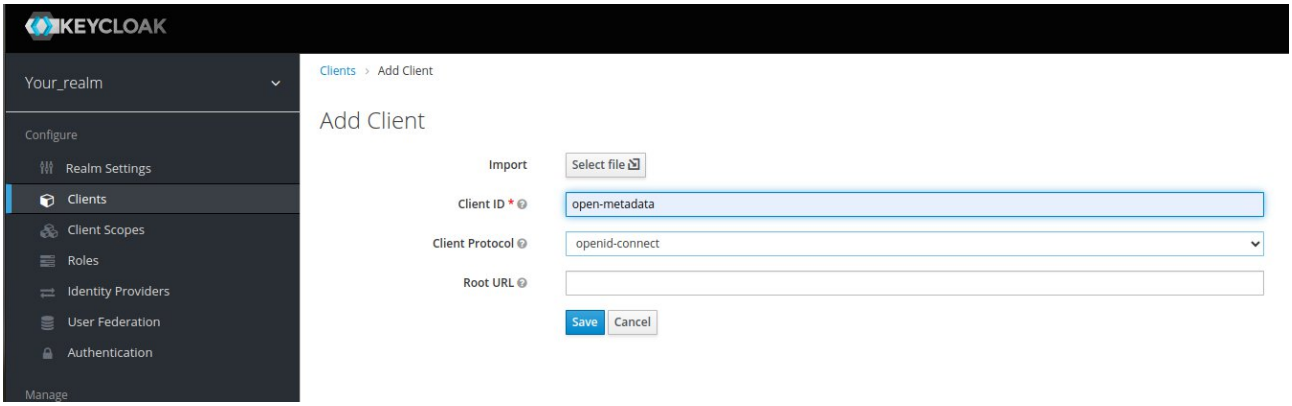


11 Keycloak

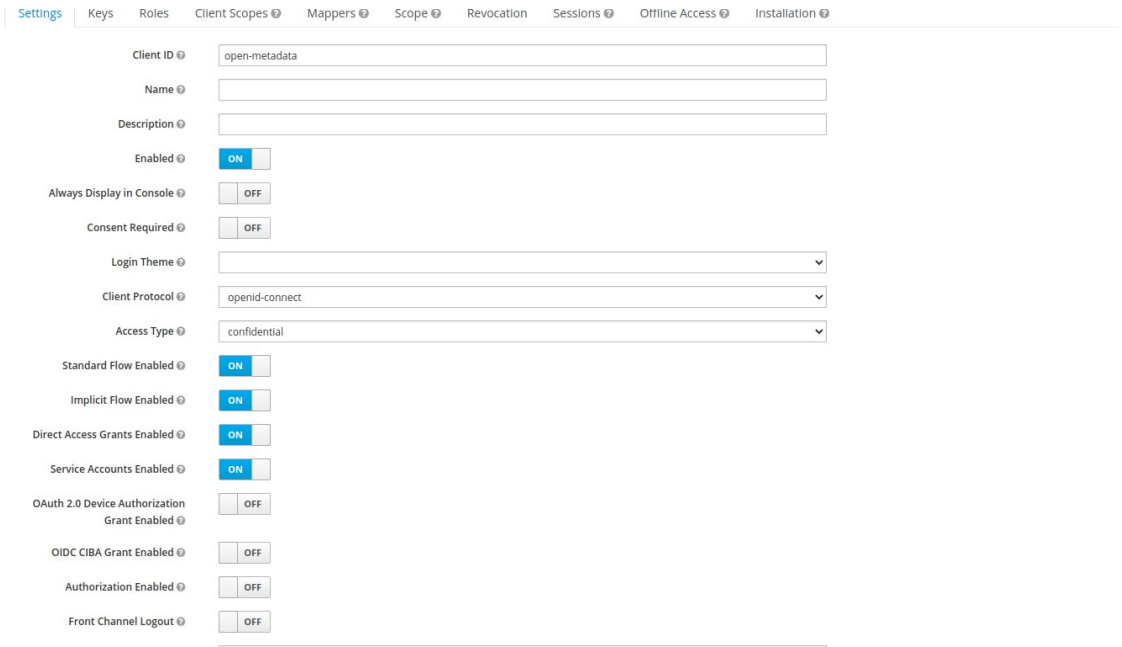
11.1 Интеграция ADC и Keycloak

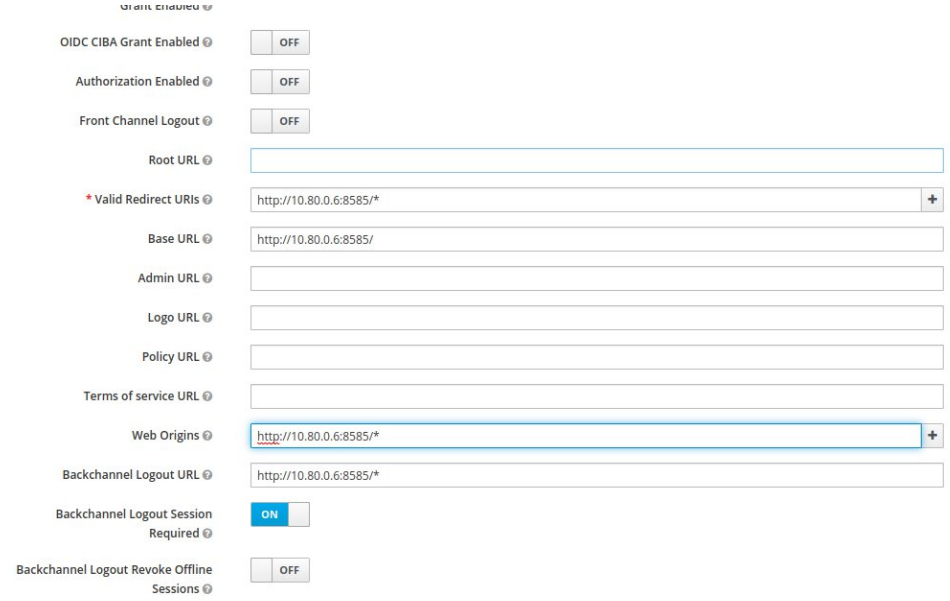
Создать realm:

Добавить Client:

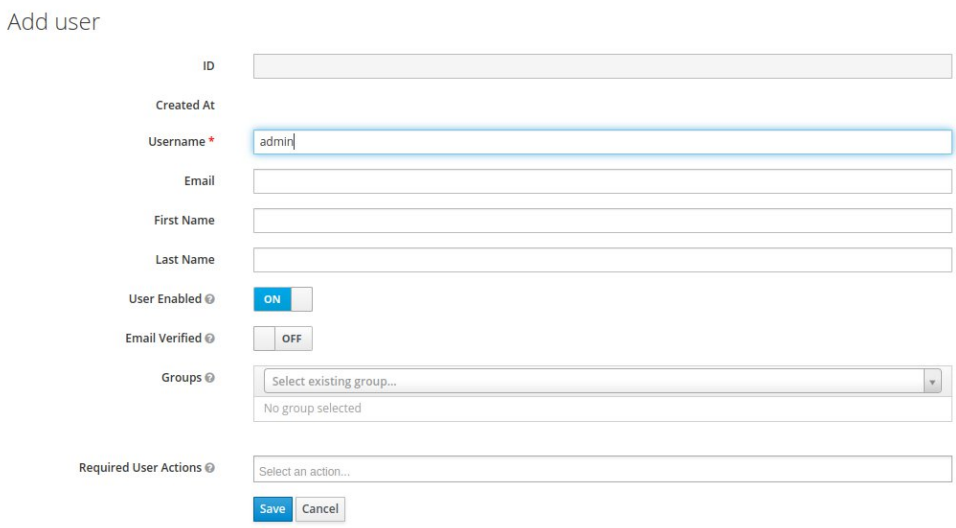


Задать и сохранить настройки клиента:

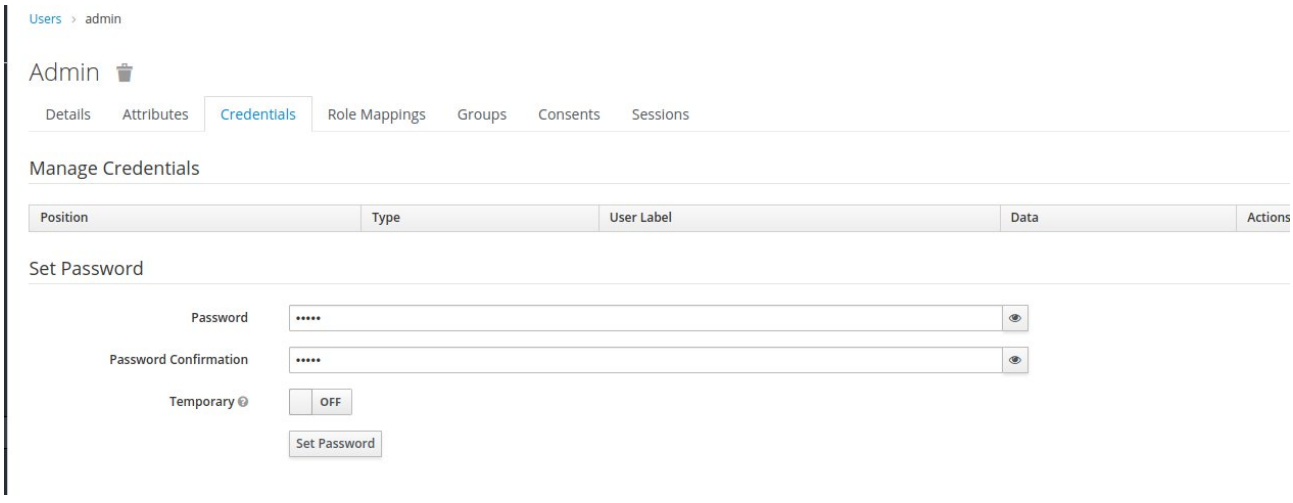




Создать user:



Задать пользователю пароль, снять флажок Temporary:



Поменять переменные окружения в docker-compose-keycloak.yml и в adc.env, указав хост и порт к своему keycloak и новому relam
(заменить 10.80.0.6:8081 и your_realm)

AUTHENTICATION_PUBLIC_KEYS=[http://10.80.0.6:8081/realms/your_realm/protocol/openid-connect/certs,http://10.80.0.6:8585/api/v1/config/jwks]
AUTHENTICATION_AUTHORITY= http://10.80.0.6:8081/realms/your_realm

Перезапустить контейнеры ADC.

Дополнительная информация по настройкам интеграции:

```
 AUTHORIZER_ADMIN_PRINCIPALS=[admin]
 AUTHORIZER_PRINCIPAL_DOMAIN=@your_domain.ru
 AUTHENTICATION_PROVIDER=custom-oidc
 CUSTOM_OIDC_AUTHENTICATION_PROVIDER_NAME=KeyCloak

AUTHENTICATION_PUBLIC_KEYS=[http://192.168.128.94:8081/realms/YOUR_REALM_NAME/protocol/openid-
connect/certs,http://192.168.128.94:8585/api/v1/system/config/jwks]

AUTHENTICATION_AUTHORITY=   http://192.168.128.94
:8081/realms/YOUR_REALM_NAME
 AUTHENTICATION_CLIENT_ID=open-metadata
 AUTHENTICATION_CALLBACK_URL=   http://192.168.128.94:8585/callback
```

AUTHORIZER_ADMIN_PRINCIPALS - это параметр конфигурации для, который предоставляет приложению список начальных администраторов.
Это значение списка, и оно обычно соответствует вашей первой половине адреса электронной почты (например ivan@adc.ru, — тогда авторизованый пользователь с правами админа будет [ivan] , то есть все, что предшествует @<your_domain_ >).
AUTHORIZER_PRINCIPAL_DOMAIN - это ваш домен из адреса электронной почты (пример adc.ru)
AUTHENTICATION_PROVIDER - Это указывает, что будет использоваться пользовательский OIDC провайдер (в данном случае, Keycloak) для аутентификации.
CUSTOM_OIDC_AUTHENTICATION_PROVIDER_NAME=KeyCloak: Это имя вашего пользовательского OIDC провайдера.
AUTHENTICATION_PUBLIC_KEYS=[...]: Это список URL-ов для получения открытых ключей, используемых для проверки JWT токенов.
AUTHENTICATION_AUTHORITY= http://192.168.128.94:8081/realms/data-sec: Это URL-адрес Keycloak, который будет использоваться в качестве источника аутентификации.
AUTHENTICATION_CLIENT_ID=open-metadata: Это идентификатор клиента, который будет использоваться для аутентификации с Keycloak. То есть клиент - это наше приложение.
AUTHENTICATION_CALLBACK_URL= http://192.168.128.94:8585/callback: Это URL-адрес обратного вызова, на который Keycloak будет перенаправлять пользователя после успешной аутентификации.

11.2 HTTPS для Keycloak

При использовании IP адресов соответствующих маске внешних IP, Keycloak для работы требует подключение по зашифрованному протоколу https. Для этого в контейнер Keycloak нужно передать

TLS сертификаты. В данном примере используется самоподписный сертификат, также можно использовать доверенные сертификаты.

1. Обновить docker-compose файл - раздел с контейнером кейклок следующим образом:

```
keycloak:
    image: ${IMG_REPO}/keycloak:${KEYCLOAK_ARTIFACT_VERSION}
    command:
      - start-dev
      - --import-realm
    environment:
      KEYCLOAK_IMPORT: /tmp/realm-export.json -
Dkeycloak.profile.feature.upload_scripts=enabled
      KEYCLOAK_ADMIN: ${KEYCLOAK_CONSOLE_ADMIN}
      KEYCLOAK_ADMIN_PASSWORD: ${KEYCLOAK_CONSOLE_ADMIN_PASSWORD}
      KC_HTTPS_CERTIFICATE_FILE: /etc/x509/https/certificate.crt
      KC_HTTPS_CERTIFICATE_KEY_FILE: /etc/x509/https/private.key
      KC_PROXY: edge – при использовании прокси
    ports:
      - "8081:8080"
      - "8843:8443"
    volumes:
      - ./config/data-sec.json:/opt/keycloak/data/import/data-sec.json
      - ./keycloak/keycloak.conf:/opt/keycloak/conf/keycloak.conf
      - keycloak-data:/opt/keycloak/data:rw
      - ./certificate.crt:/etc/x509/https/certificate.crt
      - ./private.key:/etc/x509/https/private.key
    networks:
      local_app_net:
```

2. Создать конфигурационный файл для сертификата (файл req.conf):

```
[req]
distinguished_name = req_distinguished_name
x509_extensions = v3_req
prompt = no
[req_distinguished_name]
C = US
ST = VA
L = SomeCity
O = MyCompany
OU = MyDivision
CN = YOUR_IP
[v3_req]
```

```
keyUsage = keyEncipherment, dataEncipherment, digitalSignature
extendedKeyUsage = serverAuth
subjectAltName = @alt_names
[alt_names]
IP.1 = YOUR_IP
```

3. Создать ключ и сертификат следующей командой:

```
openssl req -newkey rsa:2048 -nodes -keyout private.key -x509 -days 365 -
out certificate.crt -config req.conf -extensions 'v3_req'
```

4. Если сертификат не является доверенным добавть в JAVA Trusted Store приложения:

- Скопировать рутовый TLS сертификат в контейнер adc_server:

```
docker cp certificate_file adc_server_1:/adc/conf
```

- Зайти в контейнер adc_server:

```
docker exec -it adc_server bash
```

- Перейти в директорию с Java Trusted Store

```
cd /usr/lib/jvm/jdk-17.0.9-bellsoft-x86_64/lib/security/
```

- Там будет файл: cacerts это и будет Java Trusted Store.
- Загрузить сертификат в Java Trusted Store

```
keytool -import -alias CHOOSE-YOUR-ALIAS -file
/adc/conf/YOUR_CERTIFICATE_FILE -keystore /usr/lib/jvm/jdk-17.0.9-
bellsoft-x86_64/lib/security/
Пароль от trusted store: changeit
```

- Проверить наличие сертификата в Trusted Store по заданому ранее алиасу

```
keytool -list -keystore /usr/lib/jvm/jdk-17.0.9-bellsoft-
x86_64/lib/security/ , так же потребуется ввод пароля: changeit
```

- Перезапусть java приложение - server через docker stop/start adc_server.

12 Компоненты дистрибутива

Функциональный блок	Компонент платформы
Сервер приложения Http|API	Arenadata DC Platform
Служебная база данных	PostrgreSQL - 14.6
Поисковая система	ElasticSearch - 7.10.2 | OpenSearch - 2.16
Сервис захвата метаданных	Arenadata DC Ingestion framework Airflow-2.6.3
Сервис управления рабочими процессами Workflow	Camunda 7-19-0
Сервис уведомлений	ADC Notofication Service
	Apache Kafka, Apache Zookeeper - 7.7.0
Сервис согласований	ADC Approval Service
Перенаправление логов	Vector - 0.28.0
Хранение и поиск логов	Loki - 2.6.0, Grafana - 9.5.5
Сбор метрик	Prometheus - 2.41.0, Prometheus Graphite Exporter - 0.13.1
Средство аутентификации (опционально)	KeyCloak - 19.0.1

adc.env — файл с переменными окружения для аутентификации в приложении с keycloak
adc_noauth.env — файл с переменными окружения для базовой аутентификации по логину и паролю
alert.yml — настройки на будущее для алертов в prometheus
backup_pg.sh — скрипт для бэкапирования БД приложения/camunda/airflow
cron_backup.sh — запуск скрипта бэкапирования по расписанию (хранение 3 полных бэкапов за последние 3 дня)
config/data-sec.json — конфигурация realm в keycloak
docker-compose-keycloak.yml — docker compose для запуска приложения с keycloak
docker-compose.yml — docker compose для запуска приложения без keycloak
grafana — настройки grafana, дашборд
ingestion-dags — папка с DAGs Airflow
init.sql — скрипт инициализации БД, если используется внешняя БД
keycloak — конфигурационные файлы для логирования Keycloak
loki/loki-config.yaml — конфиг loki — сборщика логов приложения и компонентов (airflow, keycloak, vector)
prometheus — конфигурационный файл prometheus
prometheus-graphite-exporter/graphite_mapping.yml — конфигурация сборщика метрик
vector.toml — конфигурационный файл vector — агента по сбору логов
Карта портов (tcp):

- 8585, 8586 - application и его health check
- 3000 - Grafana
- 8081, 8843 - при необходимости установики Keycloak
- 8080 - Camunda – при необходимости пробросить на порт хоста: 7070
- 9090 - Prometheus
- 5432 - БД
- 8080 — Airflow

13 Контакты технических специалистов

В случае возникновения трудностей при установке программного обеспечения, свяжитесь с технической поддержкой, используя электронный адрес info@arenadc.io.