Инструкция по установке Polyflow

Содержание

Требования к системе

Серверная часть

Минимальные требования к серверному оборудованию следующие:

- CPU 2 vCPU (2.8 ГГц и выше);
- RAM 8 ГБ,
- HDD 10 ГБ

Ориентировочная формула для подсчета конфигурации в зависимости от количества процессов при использовании Local Executor: дополнительно к минимальным системным требованиям необходимо RAM 256-512МБ CPU 0.1 vCPU в среднем на каждый процесс.

Операционная система: Astra Linux Special Edition 1.6 (Смоленск) или аналог.

Права пользователя, разворачивающего приложение: user - non-root with sudo privileges.

Дополнительные требования к установленным приложениям: Docker версии 20.10.0 и до 25, Docker -compose версия 1.29 и выше.

Локальная сеть

Все компоненты платформы должны находиться в одной подсети или должна обеспечиваться прозрачная маршрутизация. Не рекомендуется использовать NAT. В рамках ознакомления рекомендуется отключить брандмауэры. Внутри локальной сети между всеми компонентами не должно быть ограничений по передаче данных. Для доступа из внешней сети достаточно открыть порт, используемый Polyflow (порт задается при установке). При использовании системы с установленными антивирусами или комплексными системами защиты необходимо обеспечить свободную работу, сетевую активность и взаимодействие компонентов.

Установка версии

Установка и настройка Docker

1. Установить Docker в соответствии с инструкцией: https://docs.docker.com/install/linux/docker-ce/ubuntu/.
   Примечание - Рекомендуемая версия 20.10.12.
2. Установить Docker Compose в соответствии с инструкцией: https://docs.docker.com/compose/install/.
   Примечание - Рекомендуемая версия 1.29.2.

Установка компонентов на один сервер

- Создать директорию для файлов образов

```
cd ~ && mkdir install
```

- Скопировать из полученного дистрибутива в созданную директорию архивы базовых образов
  - df_operator .tar
  - df_airflow .tar
  - df_postgres13 .tar
- выполнить команды:

```
docker load -i ~/install/df_operator.tar

docker load -i ~/install/df_airflow.tar

docker load -i ~/install/df_postgres13.tar
```

- Создать директории сервиса

```
cd ~ && mkdir dataflow && cd dataflow
```

- Скопировать из полученного дистрибутива в созданную директорию инициализирующие компоненты
  - manage.py
  - __init__.py
  - docker-compose.yml.tmpl
  - airflow.cfg.tmpl
- Разместить в созданной директории файлы ключей (key.pem) и сертификатов (cert.pem) (путь до файлов может быть задан при установке).
- Развернуть сервис

```
python3 manage.py --deploy
```

Создание и заполнение системных таблиц

Для корректной работы модулей системные таблицы необходимо создавать в отдельной схеме df, установленной по умолчанию для пользователя df. Для текущей версии приложения системная схема поддерживается только для СУБД PostgreSQL 10 и выше.

Создание и заполнение системных таблиц выполняется последовательным запуском DAG'ов df_upgrade_datadb и df_populate_datadb.

Для работы необходимо добавить объект Connection:

```
Conn Id: df

Conn Type: <Необходимый тип базы данных хранилища>

Host: <Хост базы данных>

Login: <Логин>

Password: <Пароль>

Extra: {"schema": "<Схема df>"}
```

Переопределить наименование схемы df возможно через переменную `'DATAFLOW_SCHEMA'`.

Если пользователь базы хранилища отличается от пользователя df, ему требуется явная выдача разрешений на системные таблицы, пример запроса:

```
grant select, insert, update, delete on all tables in schema df to dwh;

grant select, usage on all sequences in schema df to dwh;

grant execute on all functions in schema df to dwh;

grant execute on all procedures in schema df to dwh;

grant usage on schema df to dwh;

revoke select, insert, update, delete on df.alembic_version from dwh;
```

Указанные запросы можно выполнить с помощью команды:

```
docker exec -it --user postgres df-database psql -U sa -d appdb -c '<SQL-запрос>'
```

Описание утилиты управления сервисом Polyflow manage.py

Автоматизирует процесс установки, обновления и управления Polyflow.

Принимает следующие аргументы:

- `-i` или `--interactive` интерактивный режим обновления
- `--reconfigure` сгенерировать файлы конфигураций
- `--deploy` развернуть сервисы с нуля
- `--update` обновить сервисы
- `--compile` подготовка структуры артефактов
- `--run-dags` запуск системных dag'ов `df_datadb_upgrade` и `df_datadb_populate` (для запуска необходимо наличие системного подключения)
- `--sync-perm` синхронизации разрешений и добавление ролей
- `--df-conn-id` идентификатор системного подключения, по умолчанию: `df`
- `--backup` бэкап бд и конфигурации
- `--restore` восстановление бд и конфигурацию
- `--backup-ws` бэкап рабочего пространства, содержащего системный исходный код и артефакты проектов
- `--restore-ws` восстановление рабочего пространства, содержащего системный исходный код и артефакты проектов
- `--init-project` создание структуры проекта
- `--no-rebuild` не пересобирать локальные образы

- `--no-restart` не запускать (перезапускать) сервисы
- `--no-upgrade` не обновлять базу данных
- `--docker-repo` задать репозиторий docker-образов, например: `test.polymedia.ru/r5/`
- `--timeout` задать таймаут выполнения шагов утилиты в секундах (по умолчанию 10 секунд)
- `--localize` пересобрать библиотеки интерфейса (в том числе появляется возможность скомпилировать пользовательские файлы переводов)
- `--rm-default-conns` удалить примеры подключений Airflow
- `--rm-temp-tables` удалить временные таблицы Airflow.

Примеры запуска:

- `python3 manage.py --backup --no-restart --no-rebuild --no-upgrade` создать бэкап базы и конфигурации сервиса (при обновлении)
- `python3 manage.py --update --backup --no-upgrade --compile` запустить обновление
- `python3 manage.py --deploy` развернуть сервис
- `python3 manage.py --run-dags --no-restart --no-rebuild --no-upgrade` запустить системные dag'и
- `python3 manage.py --no-restart --no-upgrade` пересобрать образы
- `python3 manage.py --no-restart --no-upgrade --localize` пересобрать образы с компиляцией библиотек интерфейса
- `python3 manage.py --add-df-conn --no-restart --no-rebuild --no-upgrade` создать подключение `df` к базе данных со схемой с системными объектами.

Порядок обновления системы

При обновлении необходимо обеспечить наличие базовых образов в системе, а также инициализирующих компонентов (manage.py, __init__.py, docker-compose.yml.tmpl, airflow.cfg.tmpl), аналогично как при установке.

Ниже указан общий порядок обновления:

1. запустить обновление

```
python3 manage.py --update --backup --no-upgrade --compile
```

2. перезапустить сервис

```
docker-compose down && docker-compose up -d
```

Описание файлов и структуры проекта

Вся работа происходит в папке projects, содержимое которого линкуется с папкой workspace.

Для инициализации структуры проекта необходимо выполнить команду `python3 manage.py --init-project <Имя проекта>`. Если имя проекта не задано, то будет использовано значение default. После выполнения команды будут созданы следующие папки:

- dataflow/volumes/projects/<Имя проекта>/cache
- dataflow/volumes/projects/<Имя проекта>/dags
- dataflow/volumes/projects/<Имя проекта>/metadata
- dataflow/volumes/projects/<Имя проекта>/plugins
- dataflow/volumes/projects/<Имя проекта>/share
- dataflow/volumes/projects/<Имя проекта>/source

Дополнительно после установки сервиса доступны следующие конфигурационные файлы:

- airflow.env - переменные окружения, используемые airflow;
- operator.env - переменные окружения, используемые в operator (например, схемы БД);
- airflow.cfg - конфигурация шедулера и веб-интерфейса.

Тома:

- /home/osuser/docker/dataflow/volumes/projects - том df_projects;
- /home/osuser/docker/dataflow/volumes/workspace - том df_workspace;
- /home/osuser/docker/dataflow/volumes/data - том df_data;
- /home/osuser/docker/dataflow/volumes/logs - том df_logs.

Прикладная разработка и оператор работают в df_projects, airflow работает с df_workspace.

Поддержка локализации

В приложении реализована интернационализация и поддержан русский язык. Для переключения языка в airflow.cfg для настройки babel_default_locale необходимо указать значение ru.

Поддерживаются пользовательские словари перевода. Для их подключения в локальном Dockerfile airflow необходимо скопировать папку с переводами в соответствующую папку сервиса. Пример команды:

```
COPY --chown=airflow:airflow translations home/airflow/.local/lib/python3.9/site-packages/airflow/www/translations.
```

Папка с переводами должна иметь следующую структуру translations/<locale>/LC_MESSAGES/messages.po, где:

- <locale> - код локали, например: ru, zh, en;
- messages.po - файл перевода, который необходимо создать на основе шаблона (можно получить командой `docker cp df-webserver:/home/airflow/.local/lib/python3.9/site-packages/airflow/www/translations/messages.pot ./messages.pot`).

Настройки в airflow.cfg

При разворачивании приложения на тестовых серверах допустимо использовать http. В этом случае настройку cookie_secure можно установить в False.

Требования к СУБД

Поддерживаются СУБД PostgreSQL 9.6+ и MS SQL Server 2008 R2 SP3 10.50.6560.0+.