ГАРДА

Программный комплекс сбора, хранения, анализа информации о маршрутах и распределении трафика в сетях передачи данных с коммутацией пакетов «АВГУР» Admin Guide

Версия ПО: 1.0

Москва, 2024 www.t-argos.ru

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Настоящий документ представляет собой руководство администратора Программного комплекса сбора, хранения, анализа информации о маршрутах и распределении трафика в сетях передачи данных с коммутацией пакетов «АВГУР» (далее – ПК «АВГУР») и содержит описание требований к системе, возможностей по развертыванию, настройке и администрированию.

1. Требования к администратору и его рабочему месту

Администратор ПК «АВГУР» (далее – Администратор) должен обладать навыками сетевого инженера, а также навыками работы с ПЭВМ и с серверами на уровне администратора.
ПЭВМ администратора по уровню производительности процессора, объему ОЗУ, объему дискового пространства должна относиться к классу «офисной» на современном этапе развития вычислительной техники. ПЭВМ должна иметь сетевую связность с сервером ПК «АВГУР».
Администратор должен иметь возможность подключения по протоколам SSH, HTTP/HTTPS к репозиторию ООО «ТехАргос» для скачивания образов Docker.
Требования к программному обеспечению ПЭВМ администратора:

- операционная система с графическим интерфейсом;
- Web-браузер.

2. Развертывание ПО на выделенном сервере

2.1. Требования к ресурсам виртуальной среды

Требования к ресурсам виртуальной среды представлены в Таблице 1.

Таблица 1 – Ориентировочные требования к ресурсам виртуальной среды (типовой настройке системы)

Количество маршрутизаторов в IP/MPLS сети	Количество ядер ЦПУ	Объем RAM (Гб)	Объем HDD (Тб)
до 100	4	32	5
до 500	16	128	20
до 1500	32	256	40

Необходимое ПО для установки ПК «АВГУР»: Git, Docker (версия не ниже 24.0), docker-compose (версия не ниже 1.29).

2.2. Порядок развертывания

Скачивание дистрибутива:

- перейти в каталог /opt

```
cd /opt
```

- клонировать репозиторий, управляющий программными модулями в папку «avgur»:

```
git clone git@gitlab.t-argos.ru:sources/avgur/andmax/compose.git avgur && cd avgur
```

Настройка дистрибутива:

- перейти в каталог «compose» и открыть файл .env в текстовом редакторе;
- значение переменной KAFKA_HOST по-умолчанию равно 1.2.3.4.

Необходимо указать значение IP-адреса интерфейса сервера (далее – адрес сервера) и сохранить файл.

ВНИМАНИЕ: Указывать имя «localhost» или адрес loobpack-интерфейса недопустимо.

Запуск программных модулей:

- запустить программные модули, выполнив команду:

```
docker-compose up -d
```

- начнется процесс скачивания Docker-образов, создания контейнеров и их запуска; время выполнения зависит от скорости сетевого соединения и занимает не менее 5 минут;
- после запуска контейнеров начнется фоновая загрузка данных в БД Whois.

Создание пользователя:

- открыть браузер и перейти по адресу http://[адрес сервера]/auth. В окне подсистемы аутентификации перейти по ссылке «Administration Console» и авторизоваться с учетной записью root/avgur;

ПРИМЕЧАНИЕ: При возникновении ошибки выполнить команду:

```
docker-compose exec postgres psql -c "UPDATE realm SET ssl_required='NONE' WHERE id='master';" && docker-compose restart keycloak
```

- перейти по ссылке «Users» и добавить нового пользователя (например, «testuser») от имени которого будут выполняться действия на сайте;
- перейти на вкладку «Credentials» и задать пароль пользователя.

Загрузка данных LSDB:

- скачать образ загрузчика тестовых данных LSDB

```
docker pull 172.17.131.236:5001/am-test-data-loader
```

- загрузить тестовые данные, для чего выполнить команды

```
docker run 172.17.131.236:5001/am-test-data-loader [адрес сервера]:9094
```

- проверить корректность загрузки согласно разделу 2 Руководства пользователя ПК «АВГУР».

3. Запуск и остановка ПО

ПО ПК «АВГУР» разработано в соответствии с микросервисной архитектурой и представляют собой набор контейнеров Docker. В Таблице 2 представлены их названия, базовые образы и краткое описание.

Таблица 2 – Перечень контейнеров Docker и их описание

Название контейнера	Базовый образ Docker	Описание
avgur_analyzer	172.17.131.236:5001/am-analyzer	Реализует аналитические функции над графами IGP
avgur_backend	172.17.131.236:5001/am-backend	Часть веб-приложения, работающая на стороне веб-сервера
avgur_clickhouse	172.17.131.236:5001/am-clickhouse	Реализует хранение данных из BGP и аналитические функции над графами
avgur_consumer_lsdb	172.17.131.236:5001/am-lsdb-loader	Загрузчик данных о структуре сети в графовую БД
avgur_downloader	172.17.131.236:5001/am-downloader	Реализует загрузку BGP-данных из открытых источников и их загрузку в clickhouse
avgur_frontend	172.17.131.236:5001/am-frontend	Часть веб-приложения, работающая на стороне браузера
avgur_graph_exporter	172.17.131.236:5001/am-graph-exporter	Обеспечивает выгрузку топологии сети из графовой БД
avgur_kafka	172.17.131.236:5001/confluentinc/cp-server:7.4.0	Реализует обмен сообщениями между микросервисами
avgur_keycloak	172.17.131.236:5001/jboss/keycloak:16.1.1	Реализует аутентификацию
avgur_management	172.17.131.236:5001/confluentinc/cp-enterprise-control-center:7.4.0	Веб-интерфейс брокера сообщений
avgur_neo4j	172.17.131.236:5001/am-neo4j	Графовая БД, реализующая темпоральную модель хранения топологии сети
avgur_nginx	172.17.131.236:5001/nginx:alpine	Веб-сервер
avgur_postgres	172.17.131.236:5001/postgres:15-alpine	Реализует хранение данных веб-приложения

avgur_zookeeper	172.17.131.236:5001/confluentinc/cp-zookeeper:7.4.0	Контролирует работоспособность брокера сообщений, «сборщик мусора»
avgur_ta_pg	172.17.131.236:5001/ta_pg	Реализует хранение БД Whois
avgur_ta_whois_mntnr	172.17.131.236:5001/ta_whois_mntnr	Управление БД Whois
avgur_ta_ks	172.17.131.236:5001/ta_ks	Реализует взаимодействие avgur_ta_pg и avgur_kafka

Для запуска и остановки ПО необходимо подключиться к серверу ПК «АВГУР» по протоколу SSH, перейти в каталог /opt/avgur и выполнить действия, представленные ниже.

Начальные условия:
В папке /opt/avgur размещена копия репозитория https://gitlab.t-argos.ru/sources/avgur/andmax/compose, который управляет контейнерами.

Подготовка (остановка контейнеров):

1) Перейти в каталог /opt/avgur
2) Выполнить команду:

```
docker-compose ps
```

3) Дополнительных действий не требуется, если вывод команды не содержит контейнеров из Таблицы 2 (рис. 1), или содержит их не со статусом **Up**. В этом случае контейнеры не запущены.

```
max@vector-avgur:/opt/avgur$
max@vector-avgur:/opt/avgur$ docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
max@vector-avgur:/opt/avgur$
max@vector-avgur:/opt/avgur$
```

Рисунок 1 – Результат выполнения команды для случая, когда контейнеры не запущены

4) Если же вывод команды содержит хотя бы один контейнер из Таблицы 2 со статусом Up (рис. 2), необходимо выполнить команду:

```
docker-compose stop
```

```
max@vector-avgur:/opt/avgur$ docker-compose ps
        Name                      Command                  State
-------------------------------------------------------------------------
avgur_analyzer         /docker-entrypoint.sh            Up
avgur_backend          /docker-entrypoint.sh --ho ...   Up
avgur_clickhouse       sudo -u clickhouse clickho ...   Up (healthy)
avgur_consumer_lsdb    python3 consumer_to_neo.py ...   Up
avgur_downloader       /docker-entrypoint.sh            Up
avgur_frontend         yarn start                       Up
avgur_graph_exporter   python3 consumer_graph_exp ...   Up
avgur_init-kafka_1     /bin/sh -c                       Exit 0
                       # blocks until ...
avgur_kafka            /etc/confluent/docker/run        Up (healthy)
avgur_keycloak         /opt/jboss/tools/docker-en ...   Up
avgur_management       /etc/confluent/docker/run        Up
avgur_neo4j            /sbin/tini -g -- /docker-e ...   Up (healthy)
avgur_nginx            /docker-entrypoint.sh ngin ...   Up
avgur_postgres         docker-entrypoint.sh postgres    Up (healthy)
avgur_ta_ks            /bin/sh -c java -Dlogback. ...   Up
avgur_ta_pg            docker-entrypoint.sh postg ...   Up (healthy)
avgur_ta_whois_mntnr   /liquibase/docker-entrypoi ...   Exit 0
avgur_zookeeper        /etc/confluent/docker/run        Up (healthy)
max@vector-avgur:/opt/avgur$
```

Рисунок 2 – Результат выполнения команды для случая, когда контейнеры запущены

По завершении команды все контейнеры должны быть остановлены (рис. 3), подготовка считается выполненной.

```
max@vector-avgur:/opt/avgur$ docker-compose stop
Stopping avgur_neo4j          ... done
Stopping avgur_graph_exporter ... done
Stopping avgur_consumer_lsdb  ... done
Stopping avgur_management     ... done
Stopping avgur_analyzer       ... done
Stopping avgur_downloader     ... done
Stopping avgur_nginx          ... done
Stopping avgur_backend        ... done
Stopping avgur_keycloak       ... done
Stopping avgur_kafka          ... done
Stopping avgur_ta_ks          ... done
Stopping avgur_ta_pg          ... done
Stopping avgur_clickhouse     ... done
Stopping avgur_frontend       ... done
Stopping avgur_zookeeper      ... done
Stopping avgur_postgres       ... done
max@vector-avgur:/opt/avgur$
```

Рисунок 3 – Результат остановки контейнеров, запущенных ранее

Запуск ПО:

Все контейнеры описаны в файлах docker-compose.yml и docker-compose.override.yml.

1) Выполнить команду:

```
docker-compose start
```

2) Дождаться запуска контейнеров, о чем свидетельствует надпись «done» напротив названия каждого контейнера в консольном выводе (рис. 4).
3) Выполнить команду:

```
docker-compose ps
```

Проанализировать вывод команды, он должен содержать названия всех контейнеров из Таблицы 2 (рис. 5).

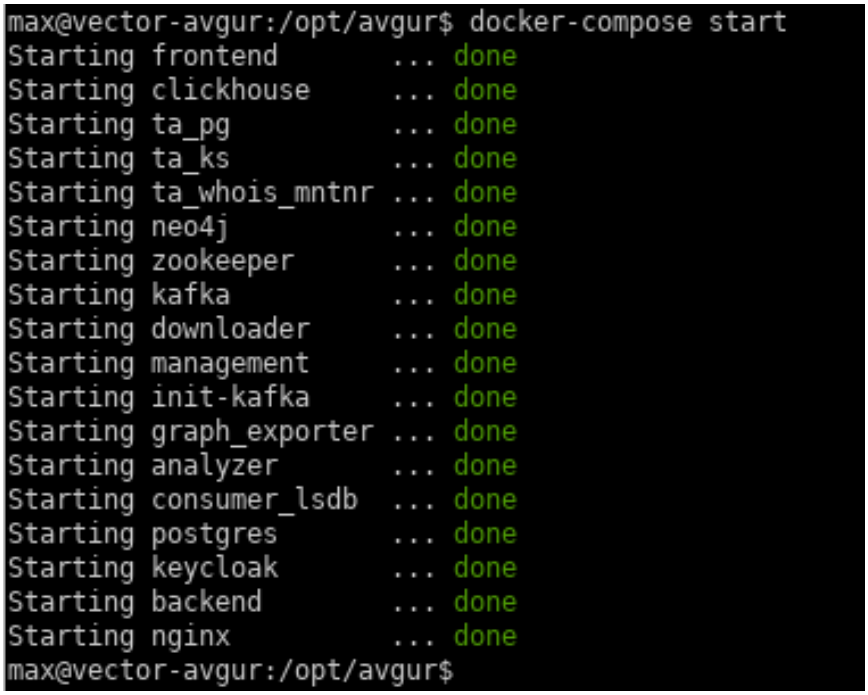

Рисунок 4 – Результат запуска контейнеров

Статус **Exited (0)** контейнеров avgur_ta_whois_mntnr и avgur_init-kafka не свидетельствует об ошибках, поскольку указанные контейнеры выполняют вспомогательные функции и завершают свою работу через несколько минут после запуска.

```
max@vector-avgur:/opt/avgur$ docker-compose ps
        Name                      Command                  State
-------------------------------------------------------------------------
avgur_analyzer          /docker-entrypoint.sh            Up
avgur_backend           /docker-entrypoint.sh --ho ...   Up
avgur_clickhouse        sudo -u clickhouse clickho ...   Up (healthy)
avgur_consumer_lsdb     python3 consumer_to_neo.py ...   Up
avgur_downloader        /docker-entrypoint.sh            Up
avgur_frontend          yarn start                       Up
avgur_graph_exporter    python3 consumer_graph_exp ...   Up
avgur_init-kafka_1      /bin/sh -c                       Exit 0
                        # blocks until ...
avgur_kafka             /etc/confluent/docker/run        Up (healthy)
avgur_keycloak          /opt/jboss/tools/docker-en ...   Up
avgur_management        /etc/confluent/docker/run        Up
avgur_neo4j             /sbin/tini -g -- /docker-e ...   Up (healthy)
avgur_nginx             /docker-entrypoint.sh ngin ...   Up
avgur_postgres          docker-entrypoint.sh postgres    Up (healthy)
avgur_ta_ks             /bin/sh -c java -Dlogback. ...   Up
avgur_ta_pg             docker-entrypoint.sh postg ...   Up (healthy)
avgur_ta_whois_mntnr    /liquibase/docker-entrypoi ...   Exit 0
avgur_zookeeper         /etc/confluent/docker/run        Up (healthy)
max@vector-avgur:/opt/avgur$
```

Рисунок 5 – Список контейнеров после запуска

4) Подождать 3 минуты и перейти в веб-интерфейс ПК «АВГУР», введя в поисковой строке браузера http://[адрес сервера]. В браузере появится страница с надписью: «Информационная система для анализа и диагностики транспортных сетей» и кнопка авторизации «Войти» справа вверху страницы.
   Нажать на кнопку авторизации, ввести учетные данные, нажать на кнопку «Войти» под полем ввода пароля.
   После авторизации (отображается имя пользователя справа вверху страницы), на странице появятся вкладки «Текущая статистика» и «События», слева появится кнопка «Действия» с изображением стрелки.

Остановка ПО:

1) Выполнить команду:

```
docker-compose stop
```

2) Дождаться остановки контейнеров, о чем свидетельствует надпись «done» напротив названия каждого контейнера в консольном выводе (рис. 3).
3) Выполнить команду:

```
docker-compose ps
```

Проверка считается пройденной, если вывод команды не содержит названия контейнеров из Таблицы 2 или содержит их не со статусом **Up**.

Перезапуск контейнера:

Для обновления образа и перезапуска контейнера выполнить команду:

```
docker pull <Базовый образ Docker> && docker-compose up -d <Название микросервиса согласно файлу docker-compose>
```