ЭНЕРГОМЕРА

ceHes. Руководство администратора

ver.1.1

Для WINDOWS/LINUX

2024

Оглавление

1. Введение

1.1. Назначение документа

Этот документ является Руководством администратора ceHes.

Для комфортной работы с ceHes пользователям необходимо:

– Знать основы работы с браузером.

– Владеть инструментами конфигурирования и администрирования OS Windows, OS Linux, системами виртуализации, контейнеризации Docker и СУБД PostgreSQL.

– Уметь пользоваться командной строкой и технической документацией к применяемым компонентам системы.

Руководство администратора предназначено для следующих целей:

– Помочь пользователю корректно установить и развернуть продукт.

– Ознакомить пользователя с процессом установки обновлений.

1.2. Термины

– **ESB** (Enterprise Service Bus) – связующее программное обеспечение, обеспечивающее централизованный и унифицированный событийно-ориентированный обмен сообщениями между различными информационными системами на принципах сервис-ориентированной архитектуры.

– **MDM** (Meter data management) – класс прикладных программ, применяемых предприятиями энергетического сектора для управления данными, полученными с приборов учёта энергии.

– **HES** (Head End System) – система, обеспечивающая коммуникацию с приборами интеллектуального учёта, для сбора, измерения, контроля параметров и предоставления доступа пользователям и внешним системам.

– **DLMS** (Device Language Message Specification) – открытый протокол для обмена данными с приборами учета.

– **СПОДЭС** – спецификация протокола обмена данными электронных счетчиков построенный на базе DLMS.

– **IEC 61968** – представляет собой серию стандартов, определяющих обмен информацией между системами распределения электроэнергии.

– **IEC 61968-100(2022)** – интеграция приложений в электроэнергетику общего пользования. Системные интерфейсы для управления распределением. Часть 100. Профили реализации.

– **CENC** – сервер канала связи, основным назначением которого является обеспечение канала связи между устройствами, имеющих не постоянный (динамический) IP-адрес и ПО верхнего уровня.

1.3. Назначение продукта

«ceHes» – коммуникационная система для организации и обеспечения взаимодействия с приборами учёта. Областью применения в рамках данной версии является серверная (облачная) платформа в виде микросервисной архитектуры в Docker-контейнерах.

Обеспечивает интеграцию с внешними MDM системами потребителя через предоставление REST-API на основе стандарта IEC 61968-100 (2022).

Система позволяет организовывать связь и обеспечивает доступ к основным функциям приборов.

Поддерживаемые приборы учёта:

– CE207 SPODES (поддержка версий 10).

– CE307 SPODES (поддержка версий 10).

– CE208 SPODES (поддержка версий 10).

– CE308 SPODES (поддержка версий 10).

Поддерживаемые функции:

– Чтение данных измерений (в том числе профилей и параметров сети).

– Чтение журналов событий.

– Чтение состояния реле.

– Изменений (управление) состоянием реле.

– Чтение и запись (синхронизация) времени.

Основными областями применения ceHes являются:

– Интеллектуальные системы учета электроэнергии (ИСУЭ).

– Розничный рынок электроэнергии для электросетевых компаний.

– Управляющие компании: СНТ, ДНТ, ТСЖ, УК и другие.

– Объекты АСКУЭ «нетребовательных потребителей» с поддержкой приборов учёта по протоколу СПОДЭС.

1.4. Основные функции программы

В ceHes существует четыре роли пользователей по умолчанию: пользователь, оператор, администратор и m2m. У каждой роли свой набор разрешений по умолчанию:

– **Пользователь**. Имеет доступ к просмотру основных форм системы и чтения архивных данных показаний, состояний и событий счетчиков.

– **Оператор**. Имеет доступ уровня пользователь и дополнительные возможности:

– Управление устройствами: добавлять, редактировать, удалять, настраивать параметры каналов связи и протоколов.

– Управление реле устройств.

– Управление расписаниями задач.

– **Администратор**. Имеет доступ уровня оператор, а также доступ к управлению системой: настройка сервера, управление пользователями, просмотр логов.

– **m2m**. Имеет доступ к чтению списка устройств, данных и возможность обращаться к REST-API интеграции на основе стандарта IEC 61968-100(2022).

1.5. Технические требования

Для корректной работы ceHes компьютер должен соответствовать следующим минимальным требованиям:

– Минимальное разрешение экрана 1280x1024.

– Оперативная память от 4ГБ.

– Подключение к интернету.

– Браузер.

Рекомендованные браузеры:

– Google Chrome v.123.

– Firefox v.124.

– Opera v.109.

Операционные системы:

– Требования к ОС для серверной части должны соответствовать актуальным требованиям для установки Docker 4.28.x (с ядром Engine 25.x). Смотрите раздел 2.1 по установке Docker.

– Требования к ОС для клиентской части должны соответствовать требованиям браузеров, характеристикам монитора и оперативной памяти из пункта выше.

При развертывании приложения на ПК, выступающем сервером и клиентом, требования выше должны быть совмещены.

Внимание! При использовании VPN и Proxy возможны сетевые проблемы или сложности у служб обеспечивающих работу Docker, WSL, Hyper-V. Ознакомьтесь с официальным руководством Docker и при необходимости обратитесь к системному администратору для консультации и решения совместного использования VPN, Proxy и Docker,

2. Установка

Для начала нужно подготовить систему, установив в ней Docker и Docker-Compose. Необходимо установить актуальную версию с официального сайта.

На текущий момент это Docker Desktop 4.28.0 (Engine 25.0.3, Compose 2.24.6)

Ниже на рисунках 1 и 2 приведена демонстрация версий для Windows и Linux.

Рисунок 1

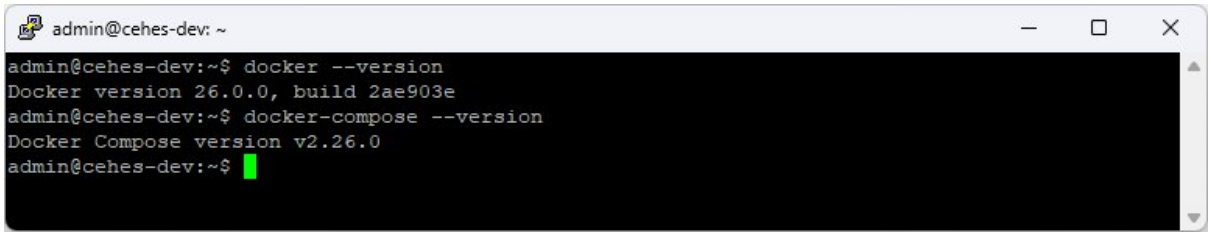

Рисунок 2

Внимание! Для установки на Windows необходимо иметь права локального администратора, а для Linux права уровня sudo/root.

2.1 Установка Docker

Актуальные системные требования, описание процесса установки и ссылки на загрузку Docker текущей версии приведены в официальной документации по ссылкам:

Для Windows: https://docs.docker.com/desktop/windows/install.

Для Linux: https://docs.docker.com/desktop/linux/install.

Внимание! Необходимо устанавливать Docker Engine не ниже версии 24.x.

2.2 Установка Docker Compose

Для Windows он будет установлен в составе Docker Desktop.

Для Linux, если установка производилась не с пакетом Docker Desktop, необходима ручная установка.

Ниже приведен пример установки для Ubuntu 22.04.

Начнем с определения последнего выпуска Docker Compose на странице выпусков (https://github.com/docker/compose/releases).

Внимание! В примерах команд ниже замените версию v2.26.0 на актуальную.

Запустите следующую команду для загрузки Docker Compose и предоставьте глобальный доступ к этому ПО в своей системе:

```
$ sudo curl -k -L "https://github.com/docker/compose/releases/download/v2.26.0/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
```

Вариант запуска команды с указанием прокси сервера:

```
$ sudo curl -x 'http:/10.5.0.9:3128' -L "https://github.com/docker/compose/releases/download/v2.26.0/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
```

Вариант запуска команды с указанием прокси сервера и игнорирование SSL сертификата (параметр -k или --insecure):

```
$ sudo curl -k -x 'http:/10.5.0.9:3128' -L "https://github.com/docker/compose/releases/download/v2.26.0/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
```

Затем необходимо задать правильные разрешения, чтобы сделать команду docker-compose исполняемой:

```
$ sudo chmod +x /usr/local/bin/docker-compose
```

Чтобы проверить успешность установки, запустите следующую команду:

```
$ sudo docker-compose --version
```

Вывод будет выглядеть следующим образом:

$ Docker Compose version v2.26.0.

2.3 Развёртывание и запуск проекта

После того как Docker и Docker-Copmpose установлены, достаточно запустить команду развёртывания проекта из репозитория Nexus (сервис Энергомера Софт).

Для этого необходимо скопировать файлы **docker-compose.yml** и **.env** в любую папку (не рекомендуется использовать длинные пути в папках или кириллические символы).

После чего, перейдя к папке с файлами в консоли, выполнить команду (см. шаг 3 в разделе 2.4 Обновление):

Ubuntu:

```
$ sudo docker-compose -p hes up -d
```

Windows:

```
docker-compose -p hes up -d
```

Внимание! Для обеспечения безопасности рекомендуется в docker-compose.yml перед развертыванием, прописать пароль системного администратора СУБД PostgreSQL (заменить root на требуемый в следующих параметрах - POSTGRES_USER=root - POSTGRES_PASSWORD=root).

Внимание! Для Windows консоль управления необходимо открыть от имени администратора.

Внимание! Для успешного выполнения всех действий необходимо наличие интернета. В случае, если интернет доступен через прокси-сервер, то необходимо настроить систему и Docker на работу через него (рекомендуется использовать интернет без прокси-сервера).

Внимание! Для исключения бесконтрольного расширения дискового пространства при внутренней процедуре логирования Docker консольного вывода контейнеров необходимо настроить ограничения на файлы логов Docker (официальная документация доступна по ссылке).

Например, добавив ограничения:

```
"log-opts": {
 "max-file": "5",
 "max-size": "10m"
}
```

Это не более 5 файлов логов архива с размером не более 10МБ.

Для Docker Desktop под Windows можно выполнить настройки, как на изображении ниже (Рисунок 3).

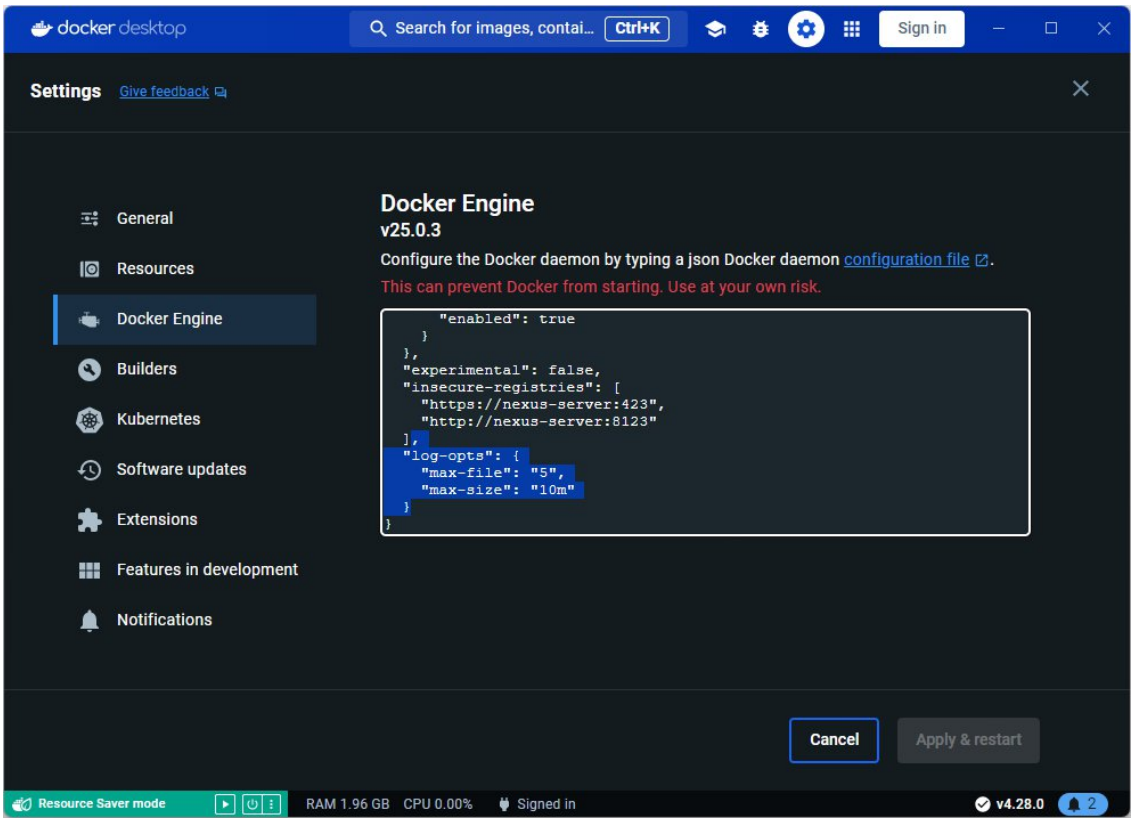

Рисунок 3

После изменения настроек необходимо нажать кнопку «Apply & restart».

Внимание! Возможна установка и настройка сертификата безопасности для подключения по HTTPS.

При необходимости его можно установить по пути хранения тома cloud_nginxssl. Путь можно узнать в параметре Mountpoint, выполнив команду:

```
docker volume inspect hes_nginxssl
```

Пример вывода (Рисунок 4):


```
[
  {
    "CreatedAt": "2023-11-27T16:24:45Z",
    "Driver": "local",
    "Labels": {
      "com.docker.compose.project": "hes",
      "com.docker.compose.version": "2.23.3",
      "com.docker.compose.volume": "nginxssl"
    },
    "Mountpoint": "/var/lib/docker/volumes/hes_nginxssl/_data",
    "Name": "hes_nginxssl",
    "Options": null,
    "Scope": "local"
  }
]
```


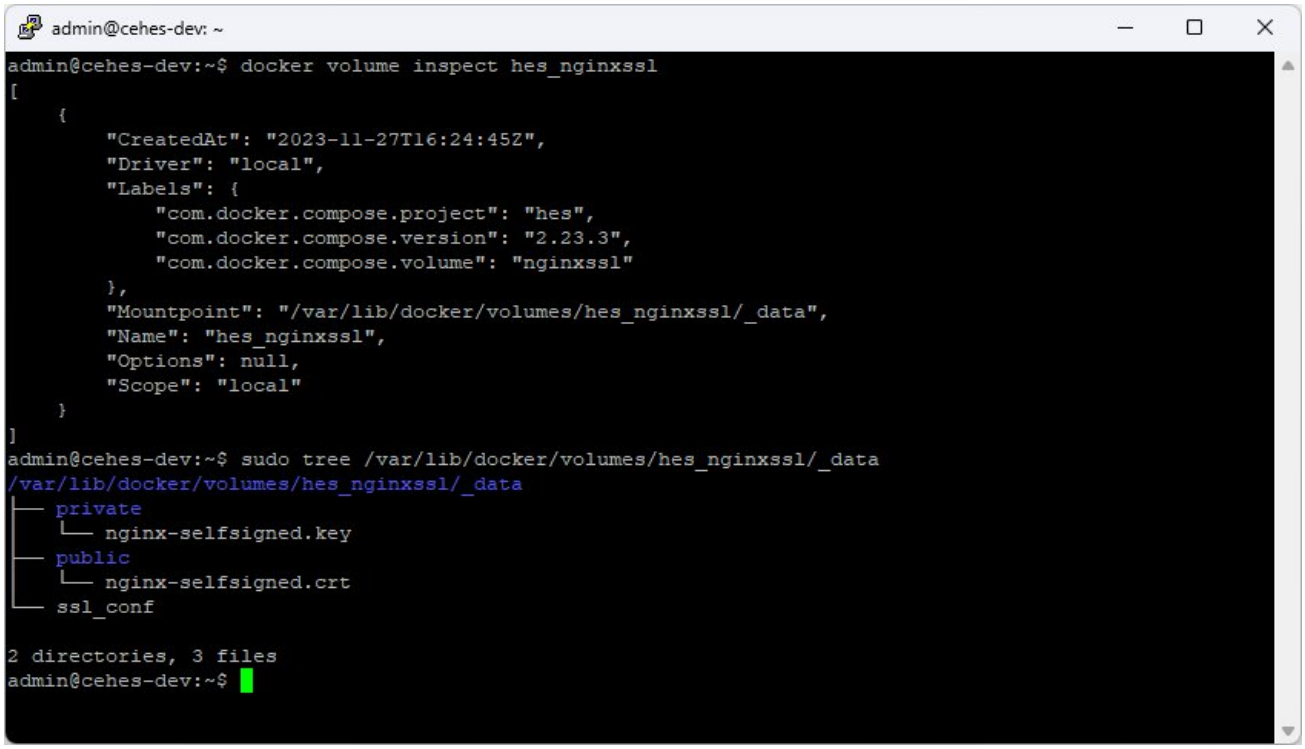

Рисунок 4

По пути /var/lib/docker/volumes/hes_nginxssl/_data в подпапках private public необходимо расположить файлы ключа (nginx-selfsigned.key) и сертификата (nginx-selfsigned.crt).

Пример подготовки самоподписанного сертификата см. далее 3.1. Настройка сертификата. Нужно обратить внимание на обязательное наличие в сертификате всех вариантов альтернативных имён DNS и IP в [alt_names].

После успешного развертывания контейнеров веб–интерфейс приложения будет доступен по ссылке https://localhost либо по адресу IP сервера, на котором оно было развёрнуто.

2.4 Обновление

Рекомендуется следующая последовательность шагов:

1) Остановить сервисы (Рисунок 5). Этот шаг является необязательным.

Ubuntu:

```
sudo docker-compose -p hes down
```

Windows:

```
docker-compose -p hes down
```

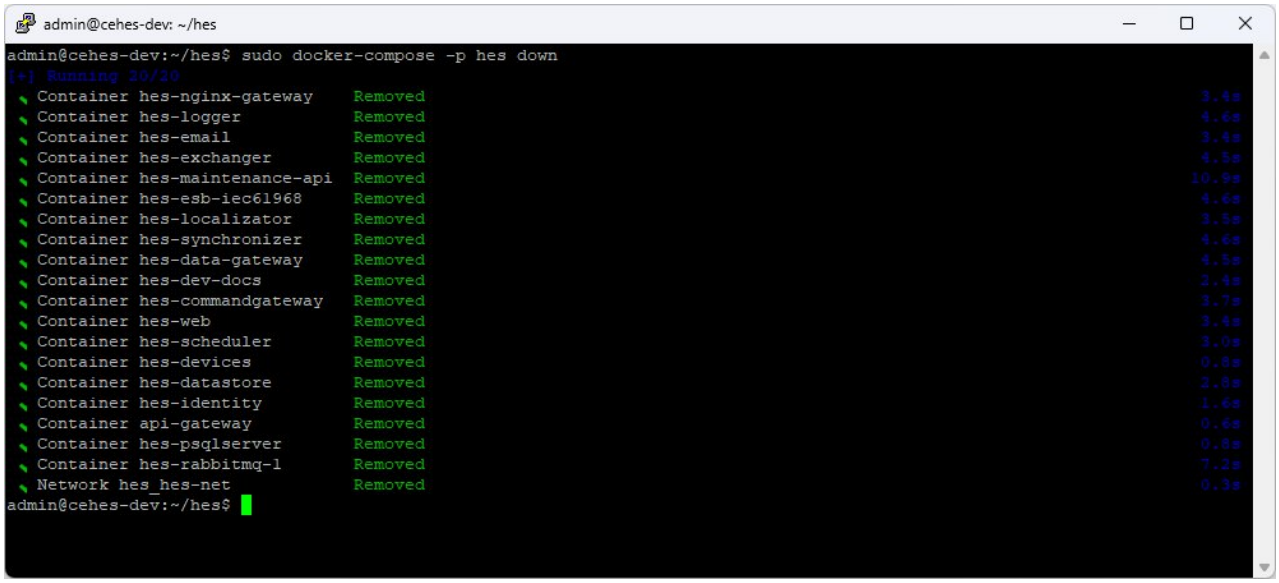

Рисунок 5

2) Обновить сервисы (Рисунок 6). Этот шаг можно выполнить повторно, чтобы убедиться, что все обновления прошли успешно и более система не обнаруживает новые версии образов.

Ubuntu:

```
sudo docker-compose -p hes pull
```

Windows:

```
docker-compose -p hes pull
```

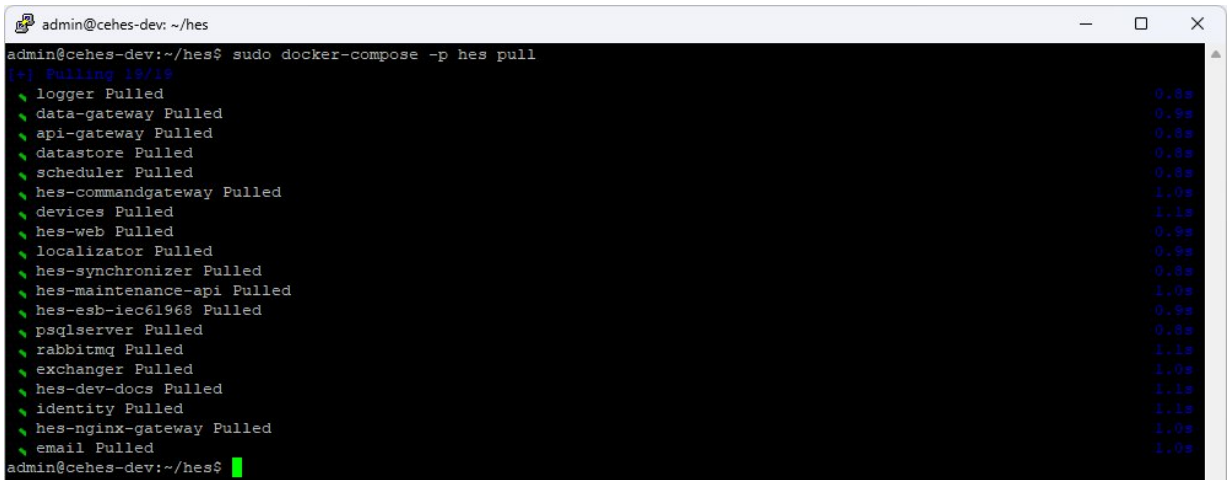

Рисунок 6

3) Запустить сервисы (Рисунок 7). Этот шаг может потребовать некоторое время, которое необходимо сервисам для инициализации настроек при их старте. Необходимо дождаться, пока у каждого сервиса будет выведено состояние Started или Healthy.

Ubuntu:

```
sudo docker-compose -p hes up -d
```

Windows:

```
docker-compose -p hes up -d
```

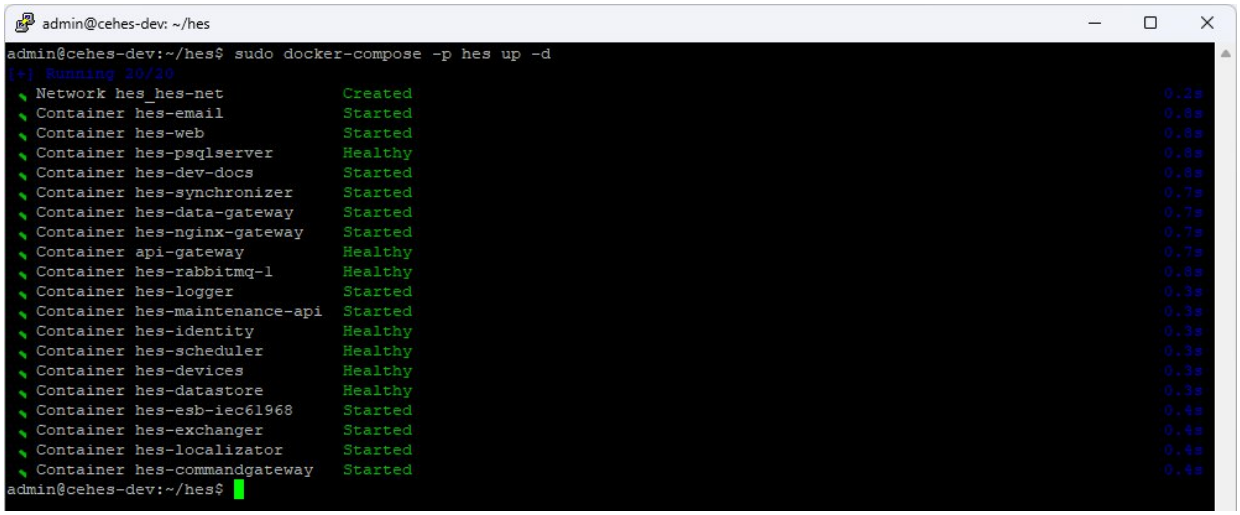

Рисунок 7

3. Конфигурирование

3.1. Настройка сертификата

Файл сертификата должен быть расположен по пути хранения тома hes_nginxssl. (см. подраздел 2.3. Развёртывание и запуск проекта, в разделе 2. Установка): /var/lib/docker/volumes/hes_nginxssl/_data. Имя файла должно быть nginx-selfsigned.crt

Для генерации самоподписанного сертификата, можно использовать openSSL.

Пример создания самоподписанного сертификата для Linux:

1. Создайте новую папку для работы и перейдите в неё:

```
mkdir /new-certs
cd /new-certs
```

2. Сгенерируйте приватный ключ:

```
openssl genrsa -out nginx-selfsigned.key 2048
```

3. Создайте текстовый файл ssl_conf следующего содержания:

```
[req]
default_bits       = 4096
prompt             = no
default_md         = sha256
x509_extensions    = v3_req
distinguished_name = dn

[dn]
C            = RU
ST           = {край/область/штат}
L            = {город/населенный пункт}
O            = {организация}
CN           = {доменное имя сервера}
emailAddress = {электронная почта}

[v3_req]
subjectAltName = @alt_names

[alt_names]
DNS.1    = {доменное имя сервера}
IP.1     = {ip-адрес сервера}
```

Внимание! Замените значения в фигурных скобках на валидные.

4. Сгенерируйте сертификат командой:

```
openssl req -new -x509 -key nginx-selfsigned.key -days 730 -out
nginx-selfsigned.crt -config <(cat ssl_conf)
```

Альтернативный способ генерирования ключа без файла конфигурации:

```
sudo openssl req -x509 -nodes -days 730 -newkey rsa:2048 -out
selfsigned.crt -keyout nginx-selfsigned.key -addext "subjectAltName =
DNS.1:{короткое доменное имя сервера}, DNS.2:{доменное имя сервера},
IP.1:{ip-адрес сервера}"
```

В данном случае все параметры, описанные в файле конфигурации, будут запрашиваться для ввода в процессе генерации ключа.

5. Переместите ключ nginx-selfsigned.key в папку private, которая была примонтирована к сервисам nginxserver.

6. Переместите сертификат nginx-selfsigned.crt в папку public, которая была примонтирована к сервисам nginxserver.

7. Перезапустите сервисы.

Подробности можно изучить в официальной документации OpenSSL.

3.2. Настройка сервиса сбора

Для настройки системы сбора возможно использовать переменные окружения.

Для этого в файле **docker-compose.yaml** в списке сервисов необходимо найти сервис exchanger и установить требуемые значения переменных окружения. Например, для установки работы сбора в 1000 потоков:

```
exchanger:
   environment:
    - exchange__concurrency=1000
```

Список переменных окружения для сбора:

Переменная окружения	Значение по умолчанию	Описание
exchange__concurrency	10	Максимальное количество параллельных потоков сбора/отправок команд
exchange__jobLifetime	00:05:00	Максимальное время жизни задачи сбора/отправки команды, после которого она будет отменена, в случае если отсутствует активность канала связи

Переменные окружения также можно настроить, введя требуемые значения в соответствующий файл **.env,** расположенный рядом с **docker-compose.yaml**.

Пример:

```
EXCHANGER_CONCURRENCY=1000
EXCHANGER_JOB_LIFETIME=00:05:00
HES_NAME=DEMO
```

3.3. Настройка сервиса DeviceMaintenance

Сервис DeviceMaintenance – содержит ряд переменных окружения, которые могут использоваться для конфигурирования поведения сервиса

Список переменных окружения:

Переменная окружения	Значение по умолчанию	Описание
HostInfo__Name	UNAVAILABLE	Имя хоста, которое будет отображаться на дашборде и странице с информацией о хосте в поле «Серийный номер».
HostInfo__IsCloud	true	Флаг того, что ПО запущено в облаке

Переменные окружения также можно настроить, введя требуемые значения в соответствующий файл **.env,** расположенный рядом с **docker-compose.yaml**.

3.4. Настройка сервиса Identity

Сервис Identity – содержит ряд переменных окружения, которые могут использоваться для конфигурирования поведения сервиса

Переменная окружения	Значение по умолчанию	Описание
HostURI		URI/IP-адрес хоста Параметр необходимо указать в соответствии с настроенной инфраструктурой, для возможности корректного восстановления пароля пользователя
AuthSettings__Signing		Ключ для подписи JWE – токена
AuthSettings__Encryption		Ключ для шифрования JWE-токена

3.5. Настройка сервиса CENC

Сервис CENC – содержит следующий ряд переменных окружения, которые могут использоваться для конфигурирования его поведения

Список переменных окружения:

Переменная окружения	Значение по умолчанию	Описание
CENC__Device__Listen__0	11001	Порт для подключения устройств по TCP/UDP
CENC__Device__Listen__1	11002	Порт для подключения устройств по TCP/UDP
CENC__AMR__Listen__0	22002	Порт для подключения ПО

Настройка отправки событий в ceHES.

Необходимо указать массив событий, которые должны отправляться в ceHES.

Список событий, которые может отправлять CENC:

Событие	Описание
SET_CENC_DEVICE_CONNECTED	Устройство подключено
SET_CENC_DEVICE_DISCONNECTED	Устройство отключено
SET_CENC_DEVICE_ACCESS_START	Начало использования CENC в качестве канала связи
SET_CENC_DEVICE_ACCESS_FINISH	Окончание использования CENC в качестве канала связи
SET_CENC_ERROR	Системная ошибка CENC
SET_CENC_AMR_USER_ADDED	AMR-агент добавлен
SET_CENC_AMR_USER_UPDATE_NAME	Обновлено имя AMR-агента
SET_CENC_AMR_USER_UPDATE_PASSWORD	Обновлен пароль AMR-агента

Для настройки событий, можно воспользоваться переменными среды и передать необходимые события из docker-compose файла.

Пример:

```
Plugins__Energomera.Hes.CENC.Plugins.CEHesIntegrationPlugin__notifications__0=SET_CENC_DEVICE_CONNECTED
Plugins__Energomera.Hes.CENC.Plugins.CEHesIntegrationPlugin__notifications__1=SET_CENC_DEVICE_DISCONNECTED
```

В данном примере – настроили передачу событий SET_CENC_DEVICE_CONNECTED, SET_CENC_DEVICE_DISCONNECTED в ceHES.

По умолчанию CENC отправляет события:

- SET_CENC_DEVICE_CONNECTED
- SET_CENC_DEVICE_DISCONNECTED
- SET_CENC_ERROR
- SET_CENC_AMR_USER_ADDED
- SET_CENC_AMR_USER_UPDATE_NAME
- SET_CENC_AMR_USER_UPDATE_PASSWORD

3.6. Настройка сервиса Logger

Сервис Logger – служит для хранения и предоставления логгов и событий системы. Так же сервис выполняет операции очистки логов, в соответствии с переданными настройками.

Базовая настройка очистки логов может быть осуществлена с помощью переменных окружения:

Переменная окружения	Значение по умолчанию	Описание
DebugLogsLifetime__Fatal	60.00:00:00	Время хранения отладочных логов уровня Fatal
DebugLogsLifetime__Error	60.00:00:00	Время хранения отладочных логов уровня Error
DebugLogsLifetime__Warning	60.00:00:00	Время хранения отладочных логов уровня Warning
DebugLogsLifetime__Information	10.00:00:00	Время хранения отладочных логов уровня Information
DebugLogsLifetime__Debug	2.00:00:00	Время хранения отладочных логов уровня Debug
DebugLogsLifetime__Trace	1.00:00:00	Время хранения отладочных логов уровня Trace

3.7. Настройка сервиса DataStore

Сервис DataStore служит для хранения и предоставления измерений, журналов событий, состояний, истории выполненных команд. Так же сервис выполняет операции очистки собранных данных, в соответствии с переданными настройками.

Некоторые настройки, которые при необходимости можно отредактировать, вынесены в переменные окружения:

Переменная окружения	Значение по умолчанию	Описание
archiveClear/launchFrequency	1.00:00:00	Периодичность запуска задачи очистки данных
archiveClear/functionDepth	365.00:00:00	Глубина хранения журнала истории выполненных команд

3.8. Настройка системы логирования

Все сервисы, по умолчанию, имеют возможность логгировать в файл, консоль, либо в СУБД.

В случае необходимости – каждому сервису можно индивидуально настроить логгирование. Для этого – необходимо подключиться к контейнеру и произвести редактирование файла nlog.config, который расположен в папке с приложением(/app).

Формат и примеры настройки файла nlog.config может быть найден по ссылке https://nlog-project.org/config/