ООО «Войс Коммьюникэйшн»

РАЗРАБОТКА СИСТЕМЫ ОБЕСПЕЧЕНИЯ БЕЗОПАСНОСТИ В ПОМЕЩЕНИЯХ SECURETRACK.AI НА БАЗЕ РЕЧЕВЫХ ТЕХНОЛОГИЙ И ИСКУССТВЕННЫХ НЕЙРОННЫХ СЕТЕЙ

Руководство по установке и настройке

Листов 24

2023

Инв. № подл.	Подп. и дата	Взамен инв. №	Инв. № дубл.	Подп. и дата

АННОТАЦИЯ

Данное руководство по установке и настройке описывает установку и первичную настройку Системы обеспечения безопасности в помещениях на базе речевых технологий и искусственных нейронных сетей SecureTrack.AI и предназначено для пояснения работы инженера или пользователя с ролью «Администратор» при развертывании данной программы.

СОДЕРЖАНИЕ

1. ВВЕДЕНИЕ

Система обеспечения безопасности в помещениях на базе речевых технологий и искусственных нейронных сетей SecureTrack.AI (далее – Система) состоит из набора сервисов, связанных друг с другом в единый комплекс. Для удобства развёртывания и последующего администрирования системы, каждый модуль выполнен в виде отдельного сервиса, упакованного в docker-контейнер. Это позволяет осуществлять быстрое развёртывание системы на базе платформы docker и простые механизмы её обновления.

2. КОНФИГУРАЦИЯ РАЗВЕРТЫВАНИЯ СИСТЕМЫ

2.1. Архитектура системы

Общая схема модулей и их взаимосвязей представлена на рисунке ниже (Рисунок 1).

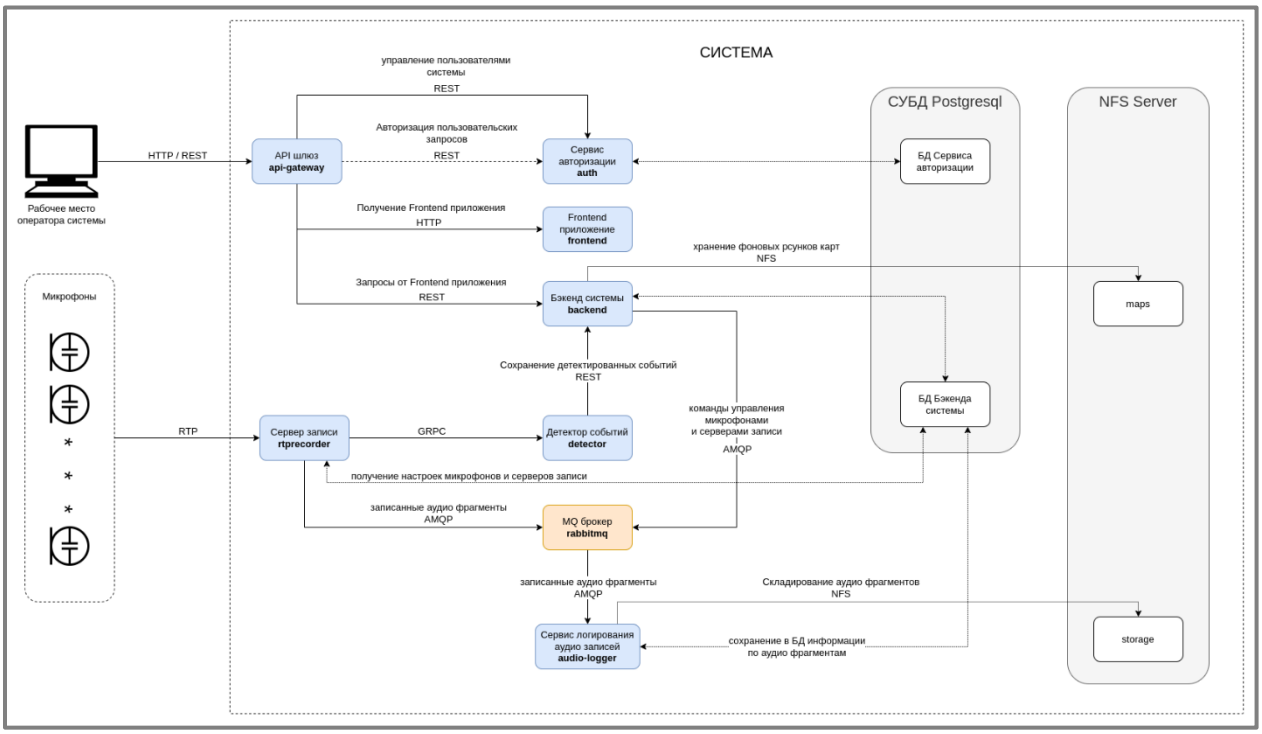

Рисунок 1 – Архитектура Системы

На данной схеме голубым цветом выделены модули Системы, входящие в дистрибутив продукта. Модули поставляются в виде образов docker-контейнеров, готовых к развёртыванию. Брокер сообщений RabbitMQ является сторонним приложением, его образ должен быть скачан самостоятельно. Для функционирования Системы также требуются дополнительные сервисы — СУБД Postgresql и NFS сервер.

Для безошибочной настройки Системы важно понимать общий принцип её работы и взаимодействие модулей между собой.

Условно, все модули Системы можно разделить на две группы:

– модули обеспечения графического интерфейса пользователя. В эту группу входят: API-шлюз, сервис авторизации, модуль визуализации (Frontend приложения) и функциональность (бэкенд) Системы.

– модули обработки данных. В эту группу входят модуль «Сервер записи», «Детектор событий», «Сервис логирования аудиозаписей» и «Бэкенд» Системы.

Как можно заметить, функциональность, отображенная в бэкенде Системы является центральным звеном и необходима и для работы графического интерфейса, и для модулей обработки данных.

Для полноценной работы Системы все модули должны иметь сетевую связность между собой, а Сервер записи должен иметь сетевую связность с микрофонами. Кроме того, с рабочего места оператора должен быть доступ к веб-интерфейсу Системы, предоставляемому через API–шлюз. Как правило это порт TCP 80 или (опционально) TCP 443.

2.2. Требования к программно-аппаратной части

2.2.1. Требования к серверам

Минимальные требования к аппаратному обеспечению:

1) в состав технических средств входят два сервера с конфигурацией:

– 4 ядра процессора, 2.4 ГГц;

– оперативная память 8 ГБ;

– жёсткий диск 60 ГБ.

2) кроме того, в состав аппаратных средств входит звуковая аппаратура:

– IP передатчик звука OCA P4LN;

– микрофоны аналоговые (4 шт.);

– сетевой коммутатор;

– аппаратура звуковоспроизведения.

Требования к ПО:

– Операционная система (ОС): Система тестировалась на ОС Debian 11, и именно данная ОС рекомендуется для установки. Однако потенциально Система может быть запущена на любой ОС, поддерживающей платформу Docker, только если данная ОС не накладывает каких-либо дополнительных ограничений на данную платформу;

– СУБД: Система требует для своей работы СУБД Postgresql версии 13 или выше.

2.2.2. Требования к рабочим станциям

Рабочая станция должна связываться с Системой по IP протоколу (TCP-порты 80 или 443, в зависимости от выбранного варианта развёртывания), и иметь актуальную версию установленного в ОС веб браузера.

Система не имеет дополнительных требований к аппаратному обеспечению рабочих станций. Рекомендуется подбирать оптимальные характеристики аппаратного обеспечения рабочих станций в соответствии с выбранной ОС на них.

2.2.3. Требования к каналам связи

К каналам связи предъявляются следующие требования:

- каналы связи от микрофонов до Системы. Канал связи от каждого микрофона до сервера записи должен обеспечивать пропускную способность не менее 256кбит/с. Пропускная способность канала связи к которому подключается сервер записи не должна быть меньше Nx256 кбит/с, где N — число микрофонов на которых осуществляется мониторинг через данный сервер записи;
- каналы связи внутри Системы. Пропускная способность между модулями Системы должны быть не менее 100Мбит/с. При этом пропускная способность канала между Системой записи и модулем детектирования должна быть не менее Nx256 кбит/с, где N — число микрофонов на которых осуществляется мониторинг событий;
- каналы от Системы до рабочих мест оператора. Для комфортной работы оператора с графическим интерфейсом системы необходим канал связи с пропускной способностью не менее 1 Мбит/с (рекомендуемая пропускная способность не менее 10Мбит/с).

2.3. Требования к квалификации персонала

Администратор Системы должен обладать квалификацией, обеспечивающей:

- навыки установки и администрирования систем на базе ОС Linux (рекомендуется ОС Debian);
- знание платформы Docker, Docker Swarm, Docker Compose;
- навыки установки, настройки и администрирования СУБД Postgresql;
- понимание принципов работы стека TCP/IP, умение настроить межсетевой экран, навыки диагностики сетевых неполадок.

3. ПОРЯДОК РАЗВЕРТЫВАНИЯ СИСТЕМЫ

3.1. Установка системы

Система спроектирована и разработана с учётом того, что она может быть развёрнута на объектах с различными масштабами (от нескольких единиц микрофонов до десятков тысяч), а соответственно и различными требованиями по производительности. В связи с этим, архитектура Системы позволяет развернуть большое число вариантов, каждый из которых будет удовлетворять конкретным требованиям на объекте мониторинга.

В данном руководстве представлены два варианта развёртывания Системы:

– минимальный вариант развёртывания в конфигурации «всё-в-одном» на одном сервере;

– масштабируемый вариант на базе кластера Docker Swarm.

Примечание – В действительности для Системы можно использовать любой другой оркестратор контейнеров, например Kubernetes. Другие оркестраторы, кроме Swarm, не рассматриваются в рамках данного руководства, однако их можно настроить, опираясь на знания архитектуры Системы.

3.1.1. Подготовка к установке

Вариант развёртывания Системы зависит от ряда функциональных требований, поэтому для начала необходимо ответить на ряд вопросов:

– количество микрофонов для единовременного мониторинга (больше 30?);

– в процессе эксплуатации системы может потребоваться значительное увеличение количества микрофонов (более чем на 20%?);

– микрофоны распределены на значительных площадях (например, по множеству помещений в пределах города?);

– требуется обеспечение отказоустойчивости 24x7 (да).

При положительном ответе хотя бы на один из этих вопросов, необходимым вариантом развёртывания является вариант на базе кластера Docker Swarm

3.1.2. Предварительные работы

Часть предварительных работ необходимо провести для обоих вариантов развёртывания системы, другая часть работ необходима только для варианта с Docker Swarm.

Примечание – Все примеры команд по установке пакетов приводятся для варианта установки Системы на ОС Linux Debian 11.

3.1.2.1. Установка Postgresql сервера

Для работы Системы необходим Postgresql сервер версии 13 или выше. Для макетов или небольших инсталляций допускается использование postgresql-сервера в docker-контейнере (из официального образа), однако для продуктивных сред данный способ не рекомендуется. Для варианта с одновременным мониторингом более чем 30 микрофонов рекомендуется вынести Postgresql сервер на выделенный физический или виртуальный сервер.

Процедура установки СУБД может отличаться в зависимости от ОС, на которой она базируется, и от версии СУБД. Для корректной установки необходимо воспользоваться детальной инструкцией в зависимости от выбранного варианта развертывания на этом сайте: **https://postgrespro.ru/docs/postgresql/13/admin.**

Если необходимо установить СУБД Postgresql13 на ОС Debian 11, нужно выполнить следующие действия:

1) обновление ОС до последней версии пакетов командой:

```
sudo apt update && sudo apt -y upgrade
```

2) добавление в ОС репозитория PostgreSQL командами:

```
sudo apt update && sudo apt -y install gnupg2
curl -fsSL https://www.postgresql.org/media/keys/ACCC4CF8.asc|sudo gpg --
dearmor -o /etc/apt/trusted.gpg.d/postgresql.gpg
echo "deb http://apt.postgresql.org/pub/repos/apt/ `lsb_release -cs`-pgdg main"
|sudo tee /etc/apt/sources.list.d/pgdg.list
```

3) установка PostgreSQL с помощью менеджера пакетов командами:

```
sudo apt update
sudo apt -y install postgresql-13 postgresql-client-13
```

4) запуск СУБД командой:

```
sudo pg_ctlcluster 13 main start
```

Для настройки отказоустойчивого кластер PostgreSQL необходимо обратиться за соответствующими инструкциями к официальному руководству по настройке PostgreSQL.

Вариант запуска postgres в докере (для макетов)

```
docker run -d -p 5432:5432 \
--restart always \
-e POSTGRES_PASSWORD=NhdrjGj3453hgSWLLERWH \
-e PGDATA=/var/lib/postgresql/data/pgdata \
-e TZ=Europe/Moscow \
-v /disk/postgres-data:/var/lib/postgresql/data \
--name postgres \
postgres:14.7
```

В данном случае инициализируется СУБД с пользователем postgres и указанным паролем. Рабочая директория СУБД подключается к контейнеру через директорию volume. При выключении/включении контейнера повторная инициализация происходить не будет.

По умолчанию, СУБД PostgreSQL устанавливается с настройками, подходящими для использования «из коробки» для небольших продуктивных стендов. Для корректной работы Системы рекомендуется изменить следующие параметры в файле postgresq.conf:

```
datestyle = 'iso, dmy'
max_connections = 1000 (только для варианта с docker swarm)
shared_buffers. Выставить равным 1/4 оперативной памяти
timezone = 'Europe/Moscow'
jit_above_cost = -1 (начиная с версии 14)
```

После установки соответствующих настроек необходимо перезапустить СУБД.

Для разрешения доступа по паролю с тех хостов, на которых будут работать сервисы Системы, необходимо отредактировать файл pg_hba.conf.

Для завершения настройки Системы необходимо добавить в нее двух пользователей: для бэкенда и сервиса авторизации, и добавить две БД, владельцами которых являются соответствующе пользователи.

Примечание – Система может работать с любой БД от имени привилегированного пользователя postgres, но в целях обеспечения корректного контроля доступа рекомендуется создать для этого специальных пользователей.

3.1.2.2. Установка NFS сервера

NFS сервер является удобным способом предоставления общего доступа к файлам и папкам. Следовательно, различные модули Системы (или их инстансы) могут совместно использовать файлы, необходимые им для работы. Например, Сервис логирования аудиозаписей складирует архивные аудиофрагменты на NFS-хранилище, а Бэкенд Системы может прочитать их для воспроизведения в интерфейсе пользователя.

В платформе Docker есть встроенная поддержка подключения к контейнерам NFS-разделов в качестве директорий volumes. Это удобный способ подключения общих директорий к контейнерам Системы, поэтому рекомендуется использовать именно этот способ.

Выбор сервера или виртуального сервера для складирования информации основывается на принципах, что сервер должен обладать соответствующим объёмом дисковых накопителей. Рекомендуется использовать RAID1 массив на сервере во избежание возможной потери информации при выходе из строя физического носителя. Хорошим вариантом будет расположение PostgreSQL и NFS-server на одном физическом сервере с хранением общих NFS-директорий и файлов СУБД на разделах, располагающихся на RAID1 массивах.

Для настройки NFS сервера первоначально необходимо:

1) установить nfs-сервер с помощью пакетного менеджера:

```
sudo apt update && sudo apt -y upgrade
sudo apt install -y nfs-kernel-server nfs-common portmap
```

2) провести конфигурирование директорий для общего доступа. Для этого указать в файле /etc/exports набор доступов для хостов и соответствующих им директорий:

```
/nfs 10.0.0.0/24(ro, fsid=0, async, no_subtree_check)
/nfs/maps 10.0.0.0/24(rw, async, no_subtree_check)
/nfs/storage 10.0.0.0/24(rw, async, no_subtree_check)
```

Здесь предполагается, что к директориям /nfs/maps и /nfs/storage доступ получают серверы из подсети 10.0.0.0/24, при этом все серверы получают права на чтение-запись.

Если развертывание Системы происходит по варианту «всё в одном», то можно предоставить доступ только для локального интерфейса сервера (не localhost). Если развертывание Системы происходит на кластере Swarm, то необходимо разрешить доступ для всех хостов, на которых расположены контейнеры сервисов Бэкенда и Сервис логирования аудиозаписей.

3) после внесения изменений необходимо перезапустить сервис

```
exportfs -ra
```

3.1.3. Установка платформы Docker

Для текущей версии Системы в качестве контейнера рекомендуется использовать Docker версии не ниже 20.10.18

Наиболее удобным вариантом установки, как и для Postgresql, является установка из внешних подключенных репозиториев. Это обеспечивает возможность использования актуальных версий пакетов и их оперативное обновление.

Для установки контейнера необходимо:

1) обновить Систему и установить необходимые дополнительные пакеты командами:

```
sudo apt-get update
sudo apt-get install ca-certificates curl gnupg
```

2) добавить репозиторий docker с помощью команд:

```
sudo install -m 0755 -d /etc/apt/keyrings
curl -fsSL https://download.docker.com/linux/debian/gpg | sudo gpg --dearmor
-o /etc/apt/keyrings/docker.gpg
sudo chmod a+r /etc/apt/keyrings/docker.gpg
echo "deb [arch="$(dpkg --print-architecture)" signed-
by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/debian \
  "$(. /etc/os-release && echo "$VERSION_CODENAME")" stable" | \
  sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
sudo apt-get update
```

3) установить последнюю версию docker с помощью менеджера пакетов командой:

```
sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-
plugin docker-compose-plugin
```

4) проверить, что docker установлен командой, результатом которой будет текущая установленная версия докера:

```
docker -v
Docker version 20.10.18, build b40c2f6
```

3.1.4. Инициализация Docker Swarm

Несмотря на то, что потенциально Систему можно запустить с помощью docker-compose или вовсе только с использованием команд docker run, рекомендуется придерживаться разворачивания Системы на базе Docker Swarm, т. к. это даст следующие преимущества:

– унифицированный способ разворачивания системы и возможность ее масштабирования;

– docker swarm — это режим, поддерживаемый сервисом docker «из коробки»;

– возможность масштабировать небольшую систему до любых размеров без необходимости перенастраивать платформу;

– встроенные механизмы балансировки запросов между экземплярами сервисов;

– встроенные механизмы обеспечения отказоустойчивости;

– множество других мелких преимуществ.

Примечание – Даже при развертывании небольшой демонстрационной или тестовой Системы в варианте «всё в одном» рекомендуется проводить развертывание на базе Docker Swarm.

Кластер Docker Swarm состоит из двух типов нод: manager (управляющей) и worker (рабочей) ноды. На manager-ноде есть возможность управления стеками (подробнее см. в руководстве пользователя docker swarm). В минимальном варианте в кластере можно иметь одну manager-ноду. Для более серьёзного кластера, используемого для продуктивных сред под большой нагрузкой, рекомендуется использовать не менее трех manager-нод и необходимое количество worker-нод.

Инициализация кластера swarm проводится с помощью команды:

```
docker swarm init
```

3.1.5. Подготовка файлов конфигурации

Ряд параметров Системы задаётся переменными окружения. Это универсальный и удобный вариант конфигурирования Системы. Если сервисы запускаются без использования контейнеров, то переменные окружения можно задать либо в .env файлах, либо с помощью команды export. В случае запуска сервисов внутри docker-контейнеров переменные окружения удобнее всего задавать в специальных файлах. При старте контейнера переменные переносятся в окружение сервиса и становятся доступны для него.

Система поставляется с типовым файлом docker-compose, позволяющим запустить её при помощи одной команды, и типовым набором переменных по умолчанию. Большинство из них не требует изменений, поэтому в данном руководстве они рассмотрены не будут. Подробнее о всех конфигурационных переменных см. в руководстве администратора системы.

Перечень переменных, которые необходимо задать перед стартом Системы, представлены в таблице ниже (Таблица 1).

Таблица 1 – Список переменных окружения Системы

Параметр	Описание	Примечания
Сервис авторизации. Файл .env-auth		
DB_HOST	Адрес хоста СУБД	
DB_PORT	Порт СУБД	
DB_LOGIN	Логин к БД сервиса авторизации	
DB_PASSWORD	Пароль к БД сервиса авторизации	
DB_NAME	Название БД сервиса авторизации	
JWT_SECRET	Ключ шифрования JWT токенов	Рекомендуется сгенерировать командой openssl rand -hex 32
Бэкенд системы. Файл .env-backend		
DB_HOST	Адрес хоста СУБД	
DB_PORT	Порт СУБД	
DB_LOGIN	Логин к БД бэкенда	
DB_PASSWORD	Пароль к БД бэкенда	
DB_NAME	Название БД бэкенда	
RABBITMQ_HOST	Адрес хоста MQ брокера	
RABBITMQ_PORT	Порт MQ брокера	

RABBITMQ_USER	Логин на MQ брокер	
RABBITMQ_PASS	Пароль на MQ брокер	
Детектор событий. Файл .env-detector		
SERVER_MAX_WORKERS	Максимальное число параллельных потоков обработки	Устанавливается в зависимости от производительности Системы и числа микрофонов
Сервер записи. Файл .env-rtprecorder		
DB_HOST	Адрес хоста СУБД	
DB_PORT	Порт СУБД	
DB_LOGIN	Логин к БД бэкенда	
DB_PASSWORD	Пароль к БД бэкенда	
DB_NAME	Название БД бэкенда	
RABBITMQ_HOST	Адрес хоста MQ брокера	
RABBITMQ_PORT	Порт MQ брокера	
RABBITMQ_USER	Логин на MQ брокер	
RABBITMQ_PASS	Пароль на MQ брокер	
NUMBER_OF_PORTS	Максимальное число параллельных потоков записи	Устанавливается в зависимости от производительности системы и числа микрофонов

Все остальные переменные в большинстве случаев можно оставить со значениями по умолчанию.

3.1.6. Импортирование образов системы

Для успешного запуска Системы docker должен иметь доступ к образам всех модулей Системы. В случае простой установки «всё в одном» достаточно импортировать все поставляемые в дистрибутиве Системы образы с помощью команды docker load

Пример команды для сборки образа бэкенда:

```
docker load -i st_backend-v1.0.tar
```

Образ будет импортирован и доступен для работы.

В случае распределённого кластера Docker Swarm образы должны быть доступны со всех worker-нод кластера. Наиболее удобным вариантом в этом случае будет организация приватного

пространства docker registry, на котором будут лежать все необходимые образы Системы, и к которому будет доступ со всех worker нод системы.

Для разворачивания репозитория образов Docker Registry необходимо обратиться к соответствующему руководству из официальной документации: **https://docs.docker.com/registry/.**

3.1.7. Запуск системы

Для запуска Системы необходимо:

1) скопировать все .env* файлы (включая отредактированные) из дистрибутива Системы и файл docker-compose.st.yaml в одну директорию. Например, это может быть директория /opt/secure_track;

2) перейти в директорию и в ней запустить Систему командой:

```
docker stack deploy --with-registry-auth -c  docker-compose.st.yaml
secure_track
```

Примечание – Директиву --with-registry-auth можно опустить, если при установке не используется docker registry и все образы есть на worker-ноде.

3) при запуске стека отобразится следующее сообщение в консоли:

```
Creating network secure_track_default
Creating service secure_track_audio-logger
Creating service secure_track_rabbitmq
Creating service secure_track_rtprecorder
Creating service secure_track_detector
Creating service secure_track_api-gateway
Creating service secure_track_frontend
Creating service secure_track_auth-service
Creating service secure_track_backend
```

4) убедиться, что все сервисы запустились: посмотреть список запущенных контейнеров командой:

```
docker stack services  secure_track
```

В результате выполнения команды отобразится список сервисов. В столбце REPLICAS напротив каждого сервиса должно быть отображено значение 1/1, что означает что каждый инстанс (один инстанс из одного) запущен.

В данном случае запущен стек в минимальном варианте, где все сервисы запущены в одном экземпляре. Если необходимо расширить производительность Системы, и кластер Swarm имеет несколько worker-нод, то необходимо отредактировать файл docker-compose.st.yaml и указать нужное количество инстансов для сервисов. Устанавливается это директивой replicas. Увеличивая число реплик для сервисов, возможно увеличить общую производительность Системы. Входящие запросы на сервисы равномерно распределяютсяя между их инстансами через Ingress сеть Swarm. Таким образом, Система может горизонтально масштабироваться под любые объёмы задач. Для более подробного изучения механизмов масштабирования кластера swarm обратитесь к соответствующим инструкциям по платформе docker.

3.2. Рекомендации по логированию

Наиболее удобный способ журналирования, если Система работает в docker, является вывод логов в консоль. В этом случае docker позволяет «из коробки» настроить перенаправление логов на сервер логирования по протоколу rsyslog. В качестве приёмника может использоваться простой rsyslog демон, либо любая из доступных систем логирования, например Elastic Stack.

Чтобы включить логирование, необходимо добавить в файл docker-compose.st.yaml следующие строки для выбранных сервисов:

```
logging:
  driver: syslog
  options:
    mode: non-blocking
    syslog-facility: "local1"
    tag: "detector"
    syslog-address: "tcp://rsyslog-server:514"
```

В данном случае подразумевается, что все логи от контейнера будут перенаправляться по протоколу syslog на сервер rsyslog-server на порт TCP 514. Параметр «tag» позволяет промаркировать логи, например, для распределения их по разным файлам в зависимости от тега.

Пример конфигурации демона rsyslog для парсинга таких логов (добавляется в файле /etc/rsyslog.conf):

```
$template secure_track,"/var/log/rsyslog/secure_track/%$YEAR%/%$YEAR%-
%$MONTH%/%$YEAR%-%$MONTH%-%$DAY%/%$YEAR%-%$MONTH%-%$DAY%-
%SYSLOGTAG:R,ERE,1,FIELD:(.*)\[--end%.log"
local1.* ?secure_track
```

В данном случае журналирование будет происходить в директорию /var/log/rsyslog/secure_track, разделяя события по дате и по используемым сервисам.

3.3. Настройка системы

По умолчанию Система устанавливается с одним пользователем с ролью «Администратор». Данный пользователь позволяет осуществлять любые действия по настройке системы, в том числе добавление других пользователей.

Логин/пароль по умолчанию:

admin@example.com / admin

Для осуществления дальнейшей настройки необходимо создать пользователя Системы и уточнить его функциональность в руководстве пользователя системы.

4. ОБНОВЛЕНИЕ СИСТЕМЫ

Все контейнеры имеют определенную версионность, поэтому обновлять необходимо только основное в версии, миграции обновляются автоматически.

Перед обновлением необходимо полностью выключить Систему, что можно сделать из предоставленного репозитория.

ПЕРЕЧЕНЬ ТЕРМИНОВ

Термин	Расшифровка
API-шлюз	Представляет собой сервис между пользователями и любым количеством внутренних сервисов системы (имеющих API), выполняющий функцию обратного прокси.
Debian	свободно распространяемая операционная система семейства Linux
Docker	Технология для автоматизации развертывания приложений в виде переносимых автономных контейнеров, выполняемых в облаке или локальной среде.
Docker Compose	Надстройка над платформой docker, приложение написанное на Python, которое позволяет запускать множество контейнеров одновременно и маршрутизировать потоки данных между ними
Docker Registry	удалённая платформа, используемая для хранения образов Docker. Является частным случаем репозитория.
docker-контейнер	формат пакетирования, который позволяет упаковать весь код и зависимости приложения в стандартный формат, чтобы приложение могло быстро и надежно запускаться в разных вычислительных средах
Frontend	Графический интерфейс приложения
Ingress сеть	Балансировщик нагрузки, распределяющий запросы между несколькими инстансами сервиса.
Kubernetes	Платформа с открытым кодом для развертывания контейнеров и управления ими в большом масштабе
localhost	имя хоста, которое ссылается на текущий компьютер, используемый для доступа к нему-же
NFS сервер	сервер, владеющий одной или более файловыми системами и делающий их доступными при работе в сети по протоколу NFS
Swarm (docker swarm)	Платформа оркестровки контейнеров с открытым исходным кодом, созданная и поддерживаемая компанией Docker. Кластер платформы обычно содержит три элемента: ноды, службы и задачи, балансировщики нагрузки.
Кластер	Совокупность серверов или сервисов, связанных между собой или объединяемых по наличию у них сходных признаков.
Менеджер пакетов	Программа в составе ОС, помогающая устанавливать и удалять приложения. Также менеджер пакетов отслеживает зависимости между программами и сохраняет систему в целостности.
Обратный прокси	прокси-сервер, логически расположенный перед набором других серверов и определяющий маршрут конкретного запроса. Эта технология позволяет начать работу с простой топологии и впоследствии перевести ее в более сложную

Термин	Расшифровка
Переменные окружения	Именованные переменные, содержащие текстовую информацию, которую могут использовать запускаемые программы.
Репозиторий	место, где хранятся и поддерживаются какие-либо данные. В частном случае репозиторий может хранить образы контейнеров Docker.
Система	Система обеспечения безопасности в помещениях на базе речевых технологий и искусственных нейронных сетей SecureTrack.AI

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ

Сокращение	Расшифровка
API	Application Programming Interface, набор способов и правил, по которым различные программы общаются между собой и обмениваются данными
IP	уникальный числовой идентификатор устройства в компьютерной сети.
NFS	Network File System, сетевая файловая система, позволяющая пользователям обращаться к файлам и каталогам, расположенным на удалённых компьютерах, как если бы эти файлы и каталоги были локальными. Главным преимуществом такой системы является то, что отдельно взятые рабочие станции могут использовать меньше собственного дискового пространства, так как совместно используемые данные хранятся на отдельной машине и доступны для других машин в сети
TCP	Transmission Control Protocol, протокол пакетной передачи информации
БД	База данных
ОС	Операционная система
ПО	Программное обеспечение
СУБД	Система управления базами данных

Лист регистрации изменений									
Номера листов (страниц)					Всего листов (страниц) в докум.	№ документа	Входящий № сопроводительного докум. и дата	Подп.	Дата
Изм.	измененных	замененных	новых	аннулированных					