GS LABS
GS GROUP

Публичный - L0

DRE Advanced Media Platform SCRAMBLER

Руководство по установке

Индекс	2004-SCRAMBLER-IG
Секретность	Публичный - L0
Ревизия	1.1
Статус	Согласован
Подразделение	ДПРСУД
Компания	GS Labs

Содержание

1. Аннотация

Данный документ содержит руководство по установке и первоначальной настройке DRE Advanced Media Platform SCRAMBLER, а также описание системных требований для компонентов.

Документ предназначен для технических специалистов, в обязанности которых входит установка и первоначальная настройка системы.

2. Термины и сокращения

Термин	Определение
Scrambler Instance	Один экземпляр скремблера. Данный компонент получает на вход ссылку на файл, либо сам файл с помощью multi part запроса. Ссылка также может указывать на live поток. Вместе с этим передается информация о том, с какими параметрами необходимо контент обрабатывать.
Scrambler Manager	Компонент доступный для внешних пользователей. Предназначен для получения от пользователя задания на скремблирование, выбора свободного Srambler Instance и отправки задания данному Scrambler Instance. Если пользователь указал несколько файлов для скремблирования, данный компонент осуществляет последовательную постановку заданий на скремблирования в свободный скремблер. Постановка задания в скремблер осуществляется только после того, как скремблер освобожден и не занят обработкой. При использовании нескольких экземпляров Scrambler Instance, данный компонент хранит в базе данных информацию об адресах всех Scrambler Instance. Также компонент может осуществлять выбор свободного Scrambler Instance для обработки. Scrambler Manager получает от пользователя информацию о профиле обработки, преобразует профиль в список параметров и передает их в Scrambler Instance.

Сокращение	Расшифровка
БД	База Данных

3. Введение

3.1. Назначение

Программа DRE Advanced Media Platform SCRAMBLER (далее - SCRAMBLER или Система) предназначена для подготовки Live (прямой эфир) и VOD (видео по запросу) контента для последующего вещания в OTT-платформах. Реализует функции транскодирования, пакетирования и шифрования. При транскодировании обеспечивает возможность конвертации контента в разные уровни качества, соответствующие пропускной способности канала связи. При этом пакетирование выполняет в форматы HLS или DASH, а шифрование осуществляет с поддержкой технологий GS DRM, Google Widevine или Apple FairPlay. Поддерживает как последовательную обработку файлов (VOD контента) и видеопотоков (Live контента), так и задание их списком для пакетной обработки. Программа обеспечивает возможность использования ресурсов как одного, так и нескольких физических или виртуальных серверов (распределенная обработка), а также ускорение процесса транскодирования с помощью видеокарт, поддерживающих технологию Nvidia Cuda. Тип ЭВМ IBM PC – совместимый ПК; ОС Linux.

3.2. Системные требования

- Операционная система:
  - Ubuntu 18.
- Пакеты:
  - Docker-CE версии 1.12 и позже.
  - docker-compose ver.1.19 и выше.
- Процессор - не менее 4 ядер, не менее 4GB.

Для сохранения результатов обработки требуется наличие удаленного NFS-сервера.

3.3. Требования к квалификации установщика

Для установки системы необходимо наличие навыков работы с ОС Ubuntu, в том числе:

- установка пакетов;
- создание и настройка сетевых подключений;
- установка и настройка PostgreSQL;
- создание и работа с БД под управлением PostgreSQL;
- работа с Docker и Docker-compose.

4. Установка и настройка SCRAMBLER

4.1. Состав компонентов для установки системы

Перед началом установки Системы убедитесь в наличии компонентов, перечисленных ниже.

4.1.1. Подсистема создания таблиц базы данных (SCRAMBLER_DB_SCH)

Архив ***SCRAMBLER_DB_SCH.zip*** со сборкой для создания схемы баз данных, включающий:

- папки *sql* и *common_db*, содержащие файлы скриптов создания и обновления схем баз данных, таблиц и пользователей;
- скрипты для установки и обновления подсистемы (*install_full.sh*, *install_full.bat*, *process_full.bat*).

4.1.2. Подсистема API для работы с базой данных (SCRAMBLER_DB_API)

Архив ***SCRAMBLER_DB_API.zip*** со сборкой для установки API управления базами данных, включающий:

- папку *sql* с процедурами работы с БД;
- папку *INIT_DATA* со скриптом начального наполнения БД;
- файлы *install.sh* и *install.bat* для установки подсистемы.

4.1.3. Папка с файлами для развертывания SCRAMBLER

Папка проекта по развертыванию SCRAMBLER:

https://cloud.gs-labs.tv/apps/files/?dir=/Scrambler&fileid=360918 (доступ ограничен)

В папке находится конфигурационный файл ***docker-compose.yml***, содержащий настройки, используемые для установки компонентов SCRAMBLER в контейнеры с помощью docker-compose.

⚠ Следует также проверить наличие в папке скриптов *install_full.sh* и *install.sh*, с помощью которых запускается процесс установки системы (см. ниже).

4.2. Настройка PostgreSQL

⚠ Система работоспособна с PostgreSQL версии 10 и выше. В документации настройка производится на примере PostgreSQL версии 10.

Следующие действия выполняются только после установки пакета postgresql версии 10 и выше:

1. (**Обязательно**) подключиться под пользователем *postgres* и добавить/изменить ему пароль (в противном случае не получиться подключиться к БД под пользователем *postgres* и запустить под ним скрипты):

```
sudo su postgres
psql -d postgres
ALTER USER postgres with encrypted password 'postgres';
```

2. Открыть конфигурационный файл ***postgresql.conf*** для редактирования:

```
sudo nano /etc/postgresql/10/main/postgresql.conf
```

3. В файле изменить значение параметра *listen_addresses*, как показано ниже, и раскомментировать соответствующую строку:

```
listen_addresses = '*'          # what IP address(es) to listen on;
```

> ⚠ Значение '*' означает, что будут слушаться все IP-адреса. Как альтернатива (не рекомендуется) вместо этого значения можно указать специфический IP-адрес **машины**, на которой стоит БД (будет слушаться только он).

4. Открыть конфигурационный файл ***pg_hba.conf*** для редактирования:

```
sudo nano /etc/postgresql/10/main/pg_hba.conf
```

5. Отредактируйте ***pg_hba.conf*** .
Необходимо, чтобы к postgres могли подсоединиться любые процессы с локальной машины и компьютеры из локальной сети (например, с ip 192.168.x.x). Также необходимо указать настройки IPv6. Таким образом, файл может выглядеть следующим образом (рекомендуется задавать уровень доступа менее открытый, чем *trust*):

```
# "local" is for Unix domain socket connections only
local        all
all                                                                     trust
# IPv4 local connections:
host        all                         all                         127.0.0.1
/32                                 md5
host        all                         all                         172.17.0.0
/16                                 md5
host        all                         all                         192.168.0.0
/16                                 md5
# IPv6 local connections:
host        all                         all                         ::1
/128                                                    md5
```

> ⚠ Если задается конфигурация доступа, отличная от приведенной выше, необходимо обеспечить доступ:
>
> **local scr scradmin trust**

6. Перезапустить PostgreSQL:

```
sudo /etc/init.d/postgresql restart
```

4.3. Создание БД

В SCRAMBLER используется БД "SCRAMBLER DB".

Начальное наполнение *SCRAMBLER DB* производится при установке. В дальнейшем БД заполняется в процессе работы с системой.

4.3.1. Создание табличных пространств

По умолчанию, БД будет создана в табличном пространстве *pg_default* (табличное пространство по умолчанию для PostgreSQL).

4.3.2. Распаковка архивов, изменение прав доступа к файлам

Последовательность действий:

1. Распаковать архив ***SCRAMBLER_DB_SCH_dev_X.X.X.zip***, входящий в комплект поставки, в желаемую папку на разделе диска, где установлен PostgreSQL (например, */tmp/SCRAMBLER_DB_SCH*).
2. Предоставить пользователям права на чтение, изменение и запуск содержимого папки, созданной на предыдущем шаге:

```
sudo chmod -R 777 /tmp/SCRAMBLER_DB_SCH
```

3. Распаковать архив ***SCRAMBLER_DB_API_dev_X.X.X.zip***, входящий в комплект поставки, в желаемую папку на разделе диска, где установлен PostgreSQL (например */tmp/SCRAMBLER_DB_API*).
4. Предоставить пользователям права на чтение, изменение и запуск содержимого папки, созданной на предыдущем шаге:

```
sudo chmod -R 777 /tmp/SCRAMBLER_DB_API
```

4.3.3. Создание SCRAMBLER DB

Последовательность действий:

1. Перейти в папку с распакованной сборкой *SCRAMBLER_DB_SCH* (в нашем примере это */tmp /SCRAMBLER_DB_SCH*):

```
cd /tmp/SCRAMBLER_DB_SCH
```

2. **1 вариант**.
    a. Запустить скрипт ***create_db.sh*** (скрипт лежит в подпапке *common_db*) со следующими параметрами:

```
bash create_db.sh $1 $2 $3 $4 $5 $6 $7
```

где:

i. $1 - host name
ii. $2 - port
iii. $3 - postgres user password
iv. $4 - db name
v. $5 - username to create
vi. $6 - user password to create
vii. $7 - DB scheme

b. Проверить лог-файл процесса создания БД (*create_db.log*) на отсутствие ошибок в процессе установки.
c. Установить схему БД: в папке с распакованной сборкой *SCRAMBLER_DB_SCH* запустить скрипт ***install_full.sh***:

```
bash install_full.sh
```

d. Используя утилиту pgAdmin, убедиться, что таблицы БД созданы. Также необходимо проверить лог-файл процесса установки (*...install_full.log*) на отсутствие ошибок в процессе установки.

3. **2 вариант**.
a. Задать входные параметры для БД:

```
export DB_HOST=... DB_PORT=... DB_NAME=... USER_NAME=... USER_PASSWORD=... PG_USER=...
PG_PASSWORD=... DB_SCHEME=...
```

, где
DB_HOST - IP для обращения к БД;
DB_PORT - порт для обращения к БД;
DB_NAME - имя БД;
USER_NAME - имя администратора БД;
USER_PASSWORD - пароль администратора БД;
PG_USER - имя пользователя postgres;
PG_PASSWORD - пароль пользователя postgres;
DB_SCHEME - имя схемы БД.

b. Запустить скрипт ***check_install_sch.sh*** (скрипт лежит в подпапке *common_db*):

```
bash check_install_sch.sh
```

c. Скрипт автоматически проверит, установлена ли БД, и выполнит все необходимые действия (обновление БД либо установка с нуля).
d. Проверить лог-файлы (*create_db.log, install_full.log* (при установке с нуля) или *install.log* (при обновлении)) на отсутствие ошибок в процессе установки.

⚠ Следует проверить на соответствие параметры, заданные при создании БД.

4.3.4. Установка API управления БД

⚠ Установку API управления БД (SCRAMBLER_DB_API) следует производить только после успешного создания БД (SCRAMBLER_DB_SCH) (см. выше).

Последовательность действий:

1. Перейти в папку с распакованной сборкой *SCRAMBLER_DB_API* (в нашем примере это */tmp /SCRAMBLER_DB_API*):

```
cd /tmp/SCRAMBLER_DB_API
```

2. Запустить скрипт ***install.sh*** (входит в состав распакованной сборки):

```
bash install.sh
```

3. Убедиться в том, что в лог-файле процесса установки (*.../SCRAMBLER_DB_API/install.log*) отсутствуют ошибки.

4.4. Установка пакетов Docker

Установка пакетов docker-ce выполняется при наличии доступа в Интернет. Инструкция по установке docker-ce описана на официальном сайте: https://docs.docker.com/engine/install/ubuntu/

4.5. Добавление пользователя в группу docker

При работе с Docker необходимо все команды с ним выполнять под *sudo* .

1. Чтобы этого избежать, рекомендуется добавить своего пользователя в группу docker. Для этого зайти в систему под требуемым пользователем (если это **не root**) и выполнить следующую команду:

```
usermod -a -G docker <current_user>
```

2. Перелогиньтесь либо выполните перезагрузку, с тем чтобы новые права вошли в силу.

4.6. Установка пакетов Docker-compose

Установка пакетов Docker-compose выполняется при наличии доступа в Интернет. Инструкция по установке Docker-compose описана на официальном сайте: https://docs.docker.com/compose/install/#install-compose

4.7. Установка компонентов SCRAMBLER с помощью Docker-compose

4.7.1. Подготовительные действия

1. Скопировать содержимое папки проекта по развертыванию SCRAMBLER https://cloud.gs-labs.tv/apps/files/?dir=/Scrambler&fileid=360918 (доступ ограничен) на машину, которая будет использоваться в качестве хоста для контейнеров.

2. Перейти в выбранную папку (в нашем примере это */home*):

```
cd /home
```

3. Предоставить пользователям права на чтение, изменение и запуск содержимого папки, созданной на предыдущем шаге:

```
sudo chmod -R 777 /home
```

4.7.2. Настройка docker-compose.yml

Конфигурационный файл compose содержит настройки, определяющие перечень контейнеров и параметры их запуска.

Описание возможных параметров можно посмотреть здесь:

- version 2: https://docs.docker.com/compose/compose-file/compose-file-v2/

Файлы docker-compose:

1. docker-compose.yml для развертывания ScramblerManager, Publisher(nginx). Пример файла:

docker-compose.yml

```
version: '2.0'
services:
  scrambler_manager:
    image: <link to image>:<scrambler_manager_version>
    container_name: scrambler_manager
    volumes:
     - ./output:${NFS_DIR}
    ports:
      - "8081:80"
      - "9090:9090"
    environment:
      OUTPUT_DIR: ${NFS_DIR}
      FILES_EXPIRE_TIME: 24
      DELETE_OLD_TASKS: 1
      # scrambler db params
      DATABASE_HOST: 192.168.10.43
      DATABASE_PORT: 5432
      DATABASE_USER: postgres
      DATABASE_PASSWORD: postgres
      DATABASE_DBNAME: scrambler
      # http server params
      HTTP_HOST: 0.0.0.0
      HTTP_PORT: 80
      HTTP_WRITE_TIMEOUT: 5
      HTTP_READ_TIMEOUT: 10000
      # logger params
      LOGGER_LEVEL: debug
      # publisher http server params
      PUBLISHER_HTTP_SERVER: "http://${LOCAL_ADDR}:8080"
      # prom params
      PROMETHEUS_ADDRESS: "0.0.0.0:9090"
      PROMETHEUS_WRITE_TIMEOUT: 5
      PROMETHEUS_READ_TIMEOUT: 5
      DISCOVERY_HEALTH_CHECK_TIMEOUT: 60
      CHECK_UNIQUE_RESOURCE_ID: 1
```

```
    # mds params
    MDS_ADDRESS: "http://mdsdmz.mds-dmz.test.gs-labs.tv"
    TIMEOUT: 5
    # rtsp
    RTSP_SERVER_HOST: "rtsp://${LOCAL_ADDR}:8088"
  restart: on-failure

# nginx for getting files from NFS over HTTP or HTTPS
nginx:
  image: <link to image>:nginx
  container_name: nginx
  ports:
    - "8080:8080"
  volumes:
    - ./nginx/nginx.conf:/etc/nginx/nginx.conf
    - ./nginx/start.sh:/start.sh
    - ./output:${NFS_DIR}
    # settings for TLS
    #- ./nginx/tls/nginx.conf:/etc/nginx/nginx.conf
    #- ./nginx/tls:/tls
  command: "nginx -g 'daemon off;'"
  cap_add:
    - SYS_ADMIN
  devices:
    - /dev/fuse
  security_opt: ["apparmor:unconfined"]
  restart: on-failure

# fas service
fas:
  image: <link to image>:<fas_service_version>
  container_name: fas
  ports:
    - "8087:80"
  environment:
    EXTERNAL_ADDRESS: "http://${LOCAL_ADDR}:8080"
    # prom params
    PROMETHEUS_ADDRESS: "0.0.0.0:9090"
    PROMETHEUS_WRITE_TIMEOUT: 5
    PROMETHEUS_READ_TIMEOUT: 5
    # http
    HTTP_ADDRESS: "0.0.0.0:80"
    HTTP_WRITE_TIMEOUT: 5
    HTTP_READ_TIMEOUT: 5
    # logger
    LOGGER_LEVEL: debug
    # secure link
    SECURE_LINK_ENABLED: "1"
    SECURE_LINK_SECRET: "212sdsadsaddsw"
    SECURE_LINK_EXPIRES: 6000
    # encryption key must be 32 bytes
    ENCRYPTION_KEY: "ba67c51400d104b784c53a63fb00eb7b"
  restart: on-failure
```

2. docker-compose.yml для развертывания сервиса ScramblerIntance (лежит в папке */scrambler_instance_deploy /scrambler_instance_deploy*). Пример файла:

docker-compose.yml

```
version: '2.0'
services:
  scrambler_instance:
    image: <link to image>:<scrambler_instance_version>
    network_mode: "host"
    volumes:
      - ./output:${NFS_DIR}
    environment:
      # transcoding
      instance_type: cpu
      transcoding_last_transcode_files: /home/scrambler/last_transcode
      # logger
      logger_level: debug
      # scrambler manager params
      manager_host: ${MANAGER_HOST_PORT}
      manager_available: 1
      # local output directory
      output_dir: ${NFS_DIR}
      # http server params
      http_address: 0.0.0.0
      http_port: ${HTTP_PORT}
      http_external_address: "${LOCAL_ADDR}:${HTTP_PORT}"
      # KMS params
      kms_host: "http://kms.integration-stand.dev.casdev"
      # encryptor params
      encryptor_Widevine_provider: provider_name
      encryptor_GsDrm_hls_time: 6
      encryptor_GsDrm_encryption: 1
      encryptor_GsDrm_key_rotation_time: 600
      # publisher params
      publisher_http_server: ${PUBLISHER_HOST_PORT}
      publisher_call_back_timeout: 1
      # time to live for chunks
      live_time_shift_buffer_depth: 100
      transcoding_overrun_nonfatal: 0
      # thumbs
      thumbnails_enabled: 0
      thumbnails_interval: 60
      thumbnails_resolution: "320x180"
      non_transcoding_target_bitrate: 10000
    restart: on-failure
```

3. docker-compose.yml сервиса rtsp_server. Находится в папке rtsp_server_deploy. Пример файла:

docker-compose.yml

```
version: '2.0'
services:
  rtsp_server:
    image: <link to image>:<rstp_server_version>
    ports:
      - "8556:8556"
      - "9998:9998"
    environment:
      RTSP_RTSPADDRESS: ":8556"
      RTSP_EXTERNALADDRESS: "172.22.0.1"
      RTSP_SCRAMBLERMANAGERADDRESS: "http://172.22.0.1:8081"
      RTSP_SCRAMBLERMANAGERTIMEOUT: "5s"
      RTSP_LOGLEVEL: "info"
      RTSP_METRICS: "yes"
    restart: on-failure
    volumes:
      - "./tls:/tls"
      - "./rtsp-simple-server.yml:/rtsp-simple-server.yml"
```

Формат yml/yaml имеет специфические требования к синтаксису (например, в yml/yaml файле нельзя использовать табуляцию). В связи с этим настоятельно рекомендуется проверить файл на корректность: https://codebeautify.org/yaml-validator

4.7.3. Установка и запуск Docker-контейнеров

Перед установкой необходимо загрузить переданные образы.

```
docker load < scrambler_instance
docker load < scrambler_manager
docker load < nginx
docker load < fas
```

Установка:

1. Перейти в папку, где лежит файл ***docker-compose.yml*** для развертывания системы.
2. Запустить скрипт установки:

   ```
   bash install_full.sh local_ip_addr
   ```

   , где local_ip_addr - адрес текущей машины.

3. Дождаться окончания операции. В случае успеха стек контейнеров будет успешно поднят. Для проверки можно:
   a. выполнить команду:

      ```
      docker ps -a
      ```

b. в появившемся списке должны быть контейнеры scrambler_manager, scrambler_instance, nginx, развернутые с помощью docker-compose.

4.7.4. Установка и запуск дополнительного Scrambler instance

Компонент Scrambler instance может устанавливаться отдельно от Scrambler manager - в случае самостоятельной работы, или в дополнение к существующим Scrambler Instance для увеличения производительности.

Перед установкой выполните подготовительные действия, как указано в разделе выше.

Перейдите в папку ***scrambler_instance_deploy***. Папка содержит конфигурационный файл ***docker-compose.yml*** для отдельной установки Scrambler instance, а также скрипт ***install.sh*** для запуска установки.

После редактирования ***docker-compose.yml*** (смысл параметров аналогичен описанным выше для комплексной установки) следует запустить скрипт установки:

```
bash install.sh local_addr manager_host:port publisher_host:port [scrambler_http_port,default=8083]
```

, где:

- *local_ip_addr* - адрес текущей машины;
- *manager_host:port* - адрес и порт компонента Scrambler manager (для регистрации);
- *publisher_host:port* - адрес и порт, которые будут указываться для системы публикации контента для забора результатов обработки.
- *scrambler_http_port* - порт, который будет слушать запущенный ScramblerInstance. По умолчанию 8083. Данная конфигурация необходима для запуска нескольких экземпляром ScramblerInstance на одной машине.

Для установки ScramblerInstance с GPU необходимо выполнить следующие действия:

1. Установить драйвера nvidia-cuda-toolkit https://developer.nvidia.com/cuda-downloads.
2. Пропатчить драйвера с помощью данной утилиты https://github.com/keylase/nvidia-patch. Это необходимо для работы ffmpeg в многопоточном режиме (более двух транскодирований одновременно).
3. Перезапустить машину, чтобы патч драйвера вступил в силу.
4. Установить nvidia-container-runtime https://github.com/NVIDIA/nvidia-container-runtime. Это необходимо для работы ffmpeg с cuda внутри docker контейнера.
5. Скопировать файл scrambler_instance_deploy/daemon.json в /etc/docker/ и перезапустить docker:

   ```
   sudo cp scrambler_instance_deploy/daemon.json /etc/docker/
   sudo systemctl restart docker
   ```

6. После чего можно выполнить запуск скремблера, как было описано выше. При этом необходимо поменять параметры instance_type в docker-compose на значение gpu.

   ```
   bash install.sh local_addr manager_host:port publisher_host:port [scrambler_http_port,default=8083]
   ```

4.7.5. Запуск RTSP сервера + HAProxy

Для запуска сервиса rtsp_server необходимо выполнить следующие действия:

1. cd rtsp_server_deploy
2. docker-compose up -d

Также может поднадобиться запустить балансировщик нагрузки между rtsp серверами. Его роль выполняет haproxy. Для этого необходимо выполнить следующие действия:

1. cd haproxy
2. docker-compose up -d

⚠ По умолчанию в переменных scrambler_manager указан адрес haproxy. Если балансировка между rtsp серверами не требуется, необходимо просто указать в scrambler_manager - RTSP_SERVER_ADDRES: rtsp_server_addr.

Чтобы указать адреса различных RTSP серверов нужно отредактировать секцию "backend rtsp_backend" конфига haproxy.cfg