GS LABS
GS GROUP

Публичный - L0

Сервис защиты цифрового контента DREMARK

Руководство по установке

Индекс	DREMARK-IG
Конфиденциальность	Публичный - L0
Ревизия	1.0
Статус	Согласован

Содержание

1. Аннотация

Данный документ содержит руководство по установке и первоначальной настройке сервиса защиты цифрового контента DREMARK (далее - система), интегрируемого с системой DRE Advanced Media Platform SCRAMBLER (далее - SCRAMBLER).
Документ предназначен для технических специалистов, в обязанности которых входит установка и первоначальная настройка системы.
Перед установкой системы рекомендуется изучить ее технические особенности построения и функционирования.
Данная информация содержится в документах "DREMARK. Общее описание" и "DREMARK. Техническое описание", входящих в комплект поставки. Данное описание является документом для внутреннего пользования, т.е. распространяется среди сотрудников GS Labs и партнеров компании.

⚠ **Данный документ опубликован исключительно с целью изучения системных требований для установки продукта, а также ознакомления с последовательностью и деталями процесса установки. Реальная установка продукта производится с использованием внутренних репозиториев ООО "Цифра", доступ к которым предоставляется заказчику по запросу.**

2. Требования к квалификации установщика

Для установки системы сотрудник обязан:

- иметь навыки работы с ОС Ubuntu, а именно:
  - установка пакетов;
  - создание и настройка сетевых подключений;
  - запуск служб, настройка автозапуска служб;
  - установка и настройка PostgreSQL;
  - создание и работа с БД под управлением PostgreSQL.
- иметь базовые представления и практические навыки работы с Git.
- иметь базовые представления и практические навыки работы с Docker.

3. Системные требования

Для установки необходимо предварительно выполнить следующие требования:

- Установлен и настроен кластер Kubernetes.
    - Так как развертывание производится в кластере k8s, то необходим config file для доступа к кластеру.
        1. Если пользователь выполнял развертывание Kubernetes самостоятельно, то он сам должен создать config file (см. документацию Kubernetes).
        2. Если Kubernetes был развернут сторонними людьми, то необходимо получить config file у администратора кластера.
- Установлен kubectl.
- Установлен helm.
- Развернут DNS-сервер, преобразование имен dns зоны настроено на мастера k8s (созданы A записи на зону dns).
- Для корректной работы системы требуется Redis база данных(устанавливается в кластере);
- Для корректной работы системы требуется развернуть кластер БД.
- Для корректной работы системы необходим доступ к следующим ресурсам:
    - chartmuseum cas-dep (ссылка предоставляется по запросу заказчика)
    - chartmuseum svc-dep (ссылка предоставляется по запросу заказчика)
    - gitlab (ссылка предоставляется по запросу заказчика)
- Необходим доступ к репозиторию, содержащему helmfile для развертывания системы. Helm файл содержит инструкции, с помощью которых осуществляются настройки устанавливаемых компонентов (Manager, web, ags и т.д.). Сами компоненты поставляются в виде образов (images), из которых разворачиваются Docker-контейнеры. Данные берутся из репозитория git.

Для сохранения результатов обработки требуется наличие удаленного NFS-сервера.

Также для корректной работы системы необходимо наличие минимум одного сервера для запуска задач:

- Операционная система ubuntu-20.04-server-amd64 (с установленным пакетом sudo).
- Docker версии 20.10.17 и выше.
- Процессор - не менее 4 ядер, не менее 4GB ОЗУ.
- При необходимости транскодирования на GPU, требуется так же наличие видеокарты nvidia не младше 600 серии и драйвер не ниже версии 520.56.06.

Сервер(-а) необходимо устанавливать в локальной сети, защищенной от доступа извне.

3.1. Предварительные действия

3.1.1. Установка PostgreSQL

> Если установка БД производится "с нуля", то необходимо развернуть кластер БД (ссылка на документ предоставляется по запросу заказчика).

Ниже приведен пример установки PostgreSQL на сервер без развертывания и настройки кластера БД.

1. (Рекомендуется) обновить текущие системные пакеты, если это новый экземпляр сервера:

```
sudo apt update
sudo apt -y install vim bash-completion wget
sudo apt -y upgrade
```

Установите дополнительные пакеты (локаль):

```
locale -a
sudo locale-gen ru_RU.UTF-8
sudo dpkg-reconfigure locales
```

Выполните перезагрузку:

```
sudo reboot
```

2. Добавьте репозиторий PostgreSQL 15:
    a. Перед настройкой репозитория APT импортируйте ключ GPG, используемый для подписи пакетов:

    ```
    sudo apt update
    sudo apt -y install gnupg2
    wget --quiet -O - https://www.postgresql.org/media/keys/ACCC4CF8.asc | sudo apt-key add -
    ```

    b. После импорта ключа GPG добавьте содержимое репозитория в ОС:

    ```
    echo "deb http://apt.postgresql.org/pub/repos/apt/ `lsb_release -cs`-pgdg main" |sudo tee  /etc
    /apt/sources.list.d/pgdg.list
    ```

    c. Добавленный репозиторий содержит много различных пакетов, включая сторонние дополнения. Они включают:

        i. PostgreSQL-клиент.
        ii. PostgreSQL.
        iii. libpq-DEV.
        iv. PostgreSQL-сервер-DEV.
        v. Пакеты pgadmin.

    d. Cat файл, созданный для проверки его содержимого:

    ```
    $ cat /etc/apt/sources.list.d/pgdg.list
    deb http://apt.postgresql.org/pub/repos/apt/ buster-pgdg main
    ```

3. Установка пакетов PostgreSQL 15:
    a. Обновите список пакетов и установите серверные и клиентские пакеты PostgreSQL 15:

    ```
    sudo apt update
    sudo apt -y install postgresql-14 postgresql-client-15
    ```

    b. Запустите сервер базы данных, используя следующую команду:

    ```
    sudo pg_ctlcluster 15 main start
    ```

c. Подтвердите статус службы и используемый файл конфигурации:

```
$ sudo pg_ctlcluster 15 main status
pg_ctl: server is running (PID: 4209)
/usr/lib/postgresql/15/bin/postgres "-D" "/var/lib/postgresql/15/main" "-c" "config_file=/etc
/postgresql/15/main/postgresql.conf"
```

d. Можно использовать команду *systemctl* для проверки статуса службы. В случае успешной установки выводится сообщение примерно следующего вида:

```
$  systemctl status postgresql.service
  postgresql.service - PostgreSQL RDBMS
    Loaded: loaded (/lib/systemd/system/postgresql.service; enabled; vendor preset: enabled)
    Active: active (exited) since Sun 2019-10-06 10:23:46 UTC; 6min ago
  Main PID: 8159 (code=exited, status=0/SUCCESS)
     Tasks: 0 (limit: 2362)
    CGroup: /system.slice/postgresql.service
 Oct 06 10:23:46 debian systemd[1]: Starting PostgreSQL RDBMS…
 Oct 06 10:23:46 debian systemd[1]: Started PostgreSQL RDBMS.

$ systemctl status [email protected]
  [email protected] - PostgreSQL Cluster 15-main
    Loaded: loaded (/lib/systemd/system/[email protected]; indirect; vendor preset: enabled)
    Active: active (running) since Sun 2019-10-06 10:23:49 UTC; 5min ago
  Main PID: 9242 (postgres)
     Tasks: 7 (limit: 2362)
    CGroup: /system.slice/system-postgresql.slice/[email protected]
            9242 /usr/lib/postgresql/15/bin/postgres -D /var/lib/postgresql/15/main -c
config_file=/etc/postgresql/15/main/postgresql.conf
            9254 postgres: 15/main: checkpointer
            9255 postgres: 15/main: background writer
            9256 postgres: 15/main: walwriter
            9257 postgres: 15/main: autovacuum launcher
            9258 postgres: 15/main: stats collector
            9259 postgres: 15/main: logical replication launcher
 Oct 06 10:23:47 debian systemd[1]: Starting PostgreSQL Cluster 15-main…
 Oct 06 10:23:49 debian systemd[1]: Started PostgreSQL Cluster 15-main.

$ systemctl is-enabled postgresql
enabled
```

e. Во время установки автоматически создаётся пользователь postgres. Это пользователь со статусом superadmin, который имеет полный доступ ко всему PostgreSQL.

4. Проверка соединения с PostgreSQL, настройка пользователя:
   a. Во время установки пользователь postgres создается автоматически. Этот пользователь имеет полный доступ superadmin ко всему экземпляру PostgreSQL.

```
sudo su - postgres
```

   b. (Необязательно) замените пароль пользователя на более надежный:

```
psql -c "alter user postgres with password 'NEW_PASSWORD'"
```

   c. Запускаем PostgreSQL с помощью команды:

```
$ psql
```

d. Получить информацию о подключении, как показано ниже:

```
$ psql
psql (15.0 (Ubuntu 14.0-1.pgdg18.04+1))
Type "help" for help.

postgres=# \conninfo
You are connected to database "postgres" as user "postgres" via socket in "/var/run/postgresql"
at port "5432".
```

e. Убедиться, что сервис PostgreSQL запускается при загрузке системы, можно с помощью команд:

```
$ systemctl status postgresql.service
$ systemctl status postgresql@15-main.service
$ systemctl is-enabled postgresql
```

3.1.2. Настройка PostgreSQL

> Данный раздел следует использовать только в случае установки БД в режиме Standalone.

Следующие действия выполняются на сервере, где будут развернуты базы данных, только после установки пакета postgresql-15.

Открыть конфигурационный файл postgresql.conf для редактирования:

```
sudo nano /etc/postgresql/15/main/postgresql.conf
```

Изменить значение параметра listen_addresses, как показано ниже, и раскомментировать соответствующую строку:

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

Открыть конфигурационный файл pg_hba.conf для редактирования:

```
sudo nano /etc/postgresql/15/main/pg_hba.conf
```

Необходимо, чтобы к postgres могли подключиться 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
/0                                       md5
host        all                       all                       172.17.0.0/0                                md5
host        all                       all                       192.168.0.0/0
md5
# IPv6 local connections:
host        all                       all                       ::1
/128                                             md5
```

При работе системы требуются подключения к базам данных, приведенным в таблице ниже. Необходимо настроить к ним доступ:

Название БД	Администратор БД
screrrmap	errmap
scr	scradmin
mgs	mgsadmin

После внесения изменений перезапустить PostgreSQL:

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

4. Установка и настройка системы

4.1. Процедура установки

4.1.1. Развёртывание внутри Kubernetes

Ссылка на проект для развертывания в kubernetes предоставляется по запросу заказчика.

4.1.1.1. Необходимые доступы

Для развёртывания проекта dremark потребуется получить доступы к проектам dremark, dremark-username-stand, cd-templates. Ссылка на проекты предоставляется по запросу заказчика.

4.1.1.2. Состав репозитория

- helmfile.yaml - конфиг helmfile
- values - конфигурационные шаблоны helmfile для генерации values.yaml файлов для каждого helm chart
- default.yaml - значения по умолчанию
- versions.gen.yaml - файл, содержащий последние стабильные версии сервисов

4.1.1.3. Описание helmfile

Helm файл содержит инструкции, с помощью которых осуществляются настройки устанавливаемых компонентов системы. Сами компоненты поставляются в виде образов (images), из которых разворачиваются Docker-контейнеры.

В helmfile.yaml содержится следующая информация:

```
repositories:
  - name: <название репозитория>
    url: <url репозитория, в котором хранятся chart'ы>

releases:
  - name: <название chart'a>
    namespace: <название namespace'a>
    chart: <путь до chart'a из репозитория>
    version: <версия chart'a>
    condition: <условие деплоя chart'a>
    labels:  # Список меток для деплоя chart'a
      stage: <стадия, на которой происходит развертка chart'a>
    values: # Указывается путь до конфигурационных шаблонов
      - "./values/<название chart'a>/values.yaml.gotmpl"
```

4.1.1.4. Описание templates

В данной директории хранятся конфигурационные шаблоны helmfile для генерации values.yaml файлов для каждого helm chart.

Фактические значения для шаблонов хранятся в файле default.yaml, что позволяет адаптировать развертывание под конкретное окружение, используя одинаковые щаблоны.

Если значение отсутствует в default.yaml, то оно может быть задано в качестве дефолтного внутри шаблона.

4.1.2. Настройка CD для продукта, опубликованного в Releases

Ссылка на документ с описанием настроек предоставляется по запросу заказчика.

4.1.3. Настройка и развертывание системы

4.1.3.1. CD для артефактов БД

При развертывании системы происходит установка SCH и API для БД через механизм Kubernetes Jobs. В процессе установки сохраняется лог в контейнере.

```
scr_db_sch:
  enabled: true

scr_db_api:
  enabled: true


# You can optionally override database address and port here:
db:
  db_mng: scr
  address: 11.111.11.111
  master_port: 5432
```

Этот режим поддерживают scr_db_* , error_mapper_db_* .

4.1.3.2. CD для бновления support_server

При развертывании dremark происходит обновление support_server на всех ранее созданных серверах для запуска задач через механизм Kubernetes Jobs. В процессе установки сохраняется лог в контейнере.

```
support_server_init:
  enabled: true
```

4.1.3.3. CD для KeyDB

Для хранения состояний задач система использует keydb-ha-cluster, доступ до которого осуществляется через Nodeport. В качестве стандартного порта используется "31379".

```
key_db:
  enabled: true
  system:
    password: password
    timeout: 10 #         KeyDB.
    NodeIp: 22.222.22.222 #         KeyDB       master-node.
    NodePort: 31379
```

4.1.3.4. Описание параметров scr_manager

Ниже представлена таблица, которая описывает параметры, используемые scr_manager сервером при деплое и работе.

Параметр	Описание
scr_manager.enabled	Добавление сервиса в деплой.
scr_manager.replicas	Кол-во реплик пода.
scr_manager.hpa.enabled	Включение hpa.
scr_manager.hpa.minReplicas	Минимальное число hpa реплик.
scr_manager.hpa.maxReplicas	Максимальное число hpa реплик.
scr_manager.hpa. targetCPUUtilizationPercentage	Процент целевого использования CPU для hpa.
scr_manager.resources.limits.cpu	Ограничение cpu для пода
scr_manager.resources.limits.memory	Ограничение памяти для пода
scr_manager.resources.requests.cpu	Запрос cpu для пода
scr_manager.resources.requests.memory	Запрос памяти для пода
scr_manager.ingress.hostname	Адресс ingress'a сервиса.
scr_manager.ingress.enabled	Добавление ingress в деплой.
scr_manager.ingress.annotations	Аннотации для ингресса
scr_manager.udpNodePort.enabled	Использовать NodePort для сервиса.
scr_manager.udpNodePort.portNumber	Номер NodePort'a.
scr_manager.configEnabled	Использование конфига.
scr_manager.config.logger.level	Степень логирования событий.Возможные значения:0 - trace, 1 - debug, 2 - info (значение по умолчанию),3 - warning, 4 - error, 5 - fatal.
scr_manager.config.serialization.emit_defaults	Серилизация значений по умолчанию.
scr_manager.config.cache.db	Номер базы данных для доступа scr_manager.
scr_manager.config.cache.expire_time	TTL записей в кеше.
scr_manager.config.cache.cid_ttl	Время длительности хранения cid задачи в redis.
scr_manager.config.instances_settings. create_pipes	Создание пайпов/файлов во время работы инстансов.

scr_manager.config.instances_settings.status_check	Период опроса статусов работы инстансов.
scr_manager.config.instances_settings.fail_after	"Время для перевод инстанса в статус ""Failed"" при отсутствии на нем обработки данных."
scr_manager.config.instances_settings.proxy_settings.host	Хост прокси эллемента.
scr_manager.config.instances_settings.proxy_settings.timeout	Таймаут прокси.
scr_manager.config.instances_settings.proxy.output_type	Тип прокси эллемента.
scr_manager.config.instances_settings.output_dir	Папка для публикации.
scr_manager.config.instances_settings.docker_dir	Стандартная папка для хранения файлов, результатов работы docker контейнеров.
scr_manager.config.instances_settings.timeout	Таймаут.
scr_manager.config.instances_settings.ffprobe_timeout	Таймаут при работе с ffprobe.
scr_manager.config.instances_settings.skip_instance_deletion	Включение удаления инстансов с сервера после завершения задачи.
scr_manager.config.transcoder.sync_every	Период синхронизации транскодера.
scr_manager.config.packager.encryption_scheme	Схема шифрования.
scr_manager.config.packager.segment_duration	Время на которое делятся сегменты видео.
scr_manager.config.packager.chunk_list_size	Предельное кол-во сохраняемых чанков для live потока.
scr_manager.config.packager.key_rotation_period	Период ротации ключей.
scr_manager.config.packager.kms_timeout	Таймаут kms сервера.
scr_manager.config.packager.kms_url	Адрес kms сервера.
scr_manager.config.packager.ffmpeg_join_a_v	Флаг, отвечающий за сбор видео-, аудио-потоков в 1 файл. Актуален для профилей шифрования gs_aes и plain_hls.
scr_manager.config.publisher.update_period	Период обновления данных паблишера.
scr_manager.config.publisher.status_timeout	Таймаут работы паблишера.

scr_manager.config.publisher. max_workers_vod	Максимальное число процессов для публикации VOD контента.
scr_manager.config.publisher. max_workers_live	Максимальное число процессов для публикации LIVE контента.
scr_manager.config.publisher. delete_after_copy	Флаг, отвечающий за удаление уже скопированных чанков в исходных папках.
scr_manager.config.rtsp_process. segment_duration	Время на которое делятся сегменты видео.
scr_manager.config.rtsp_process. chunk_list_size	Предельное кол-во сохраняемых чанков для rtsp потока.
scr_manager.config.rtsp_process.wait_timeout	Таймаут для перехода rtsp задачи в статус In progress.
scr_manager.config.downloader.attempts	Кол-во попыток загрузки.
scr_manager.config.downloader. attempt_period	Время между попытками загрузки.
scr_manager.config.downloader.max_parallel	Кол-во одновременного загружаемых чанков.
scr_manager.config.downloader. playlist_timeout	Таймаут при скачивании плейлиста.
scr_manager.config.downloader.file_timeout	Таймаут при скачивании видеофайла.
scr_manager.config.downloader. segment_duration	Время сегментов загрузки.
scr_manager.config.support_server. ffprobe_timeout	Таймаут ffprobe
scr_manager.config.operator	Используемый оператор.
scr_manager.config.process_monitor. update_duration	Время обновлений.
scr_manager.config.pipeline.ping_interval	Интервал проверки состояния.
scr_manager.config.pipeline.attempts_number	Попыток проверки.
scr_manager.config.playlist_url.mounted_dir	Префикс примаунченной дирректории для составления итогово url мастер-плейлиста.
scr_manager.config.playlist_url.use_plm	Флаг отвечающий за переформирование ссылки на мастер-плейлист сразу для пользователя plm
scr_manager.config.playlist_url.plm_url	Адрес plm сервера для использования в ссылке на мастер-плейлист

scr_manager.config.tasks_consumer.buffer_size	максимальное количество задач в очереди
scr_manager.config.tasks_consumer.max_routines	максимальное количество отслеживаемых задач
scr_manager.config.tasks_consumer.sleep_time	время ожидания между запросами в базу данных для получения новых задач
scr_manager.config.watermark.injector.clear_output_dir	Флаг отчистки рабочей диррекории после его перемещения в папку для публикации.
scr_manager.config.watermark.injector.concurrent_frames	Количество одновременно обрабатываемых кадров.
scr_manager.config.watermark.injector.max_workers	Максимальное число одновременно запущенных процессов инжектирования в рамках задачи.
scr_manager.config.watermark.injector.attempts_num	Количество попыток инжектирования чанка.
scr_manager.config.watermark.injector.max_frames_per_group	максимальное количество кадров в одной группе кадров. (Группа кадров - пачка кадров между двумя ключевыми кадрами). Параметр нужен для оптимизации потребления памяти
scr_manager.config.watermark.postprocessor.obfuscation_enabled	Флаг включения обфускации.
scr_manager.config.watermark.postprocessor.obfuscation_secret	Параметр обфускации.
scr_manager.config.watermark.postprocessor.delete_unused	Флаг отвечающий за удаление побочных файлов из рабочей директории.
scr_manager.config.watermark.preprocessor.hls_time	Длительность чанков.
scr_manager.config.mgs.host	адресс ingress'a сервиса
scr_manager.config.mgs.timeout	таймаут
scr_manager.config.mgs.protocol	протокол для взаимодействия с сервисом (http или grpc)
scr_manager.config.detection.parameters_definition_duration	длительность контента в секундах, используемое при определении исходных параметров контента
scr_manager.config.detection.user_determination_duration	длительность контента в секундах, используемое для определения пользователя
scr_manager.config.detection.min_length_for_success_parameters_definition	минимальная длина AB последовательности, чтобы признать определение параметров удачным и перейти к определению пользователя

scr_manager.config.detection.detector.skip_container_deletion	флаг, чтобы не удалять контейнер после работы
scr_manager.config.detection.detector.packet_size	длительность чанка в секундах при разбиении контента
scr_manager.config.detection.detector.concurrent_chunks	количество одновременно обрабатываемых чанков
scr_manager.config.detection.detector.concurrent_frames	количество одновременно обрабатываемых кадров для одного чанка
scr_manager.config.detection.detector.skip_artifacts_deleting	флаг пропуска удаления служебных файлов
scr_manager.config.detection.transformer.skip_container_deletion	флаг,чтобы не удалять контейнер после работы
scr_manager.config.detection.transformer.crf	Constant Rate Factor, параметр отвечающий за качество видео. Задаётся в диапазоне от 0 до 51, где 0 - кодирование без потерь, а 51 - максимальное сжатие
scr_manager.config.detection.classifier.skip_container_deletion	флаг необходимый, чтобы не удалять контейнер после работы
scr_manager.config.detection.classifier.debug_mode	флаг, отвечающий за отключение ограничения min_labels_for_detect
scr_manager.config.detection.classifier.thold	порог ниже которого метка считается не обнаруженной
scr_manager.config.detection.classifier.step	количество значений для усреднения
scr_manager.config.detection.classifier.min_seq_size	минимальное количество одинаковых групп фреймов с одной и той-же меткой
scr_manager.config.detection.classifier.shortest_seq_len	минимальная последовательность чанков которая считается корректной
scr_manager.config.detection.classifier.min_labels_for_detect	количество меток которое достаточно для детектирования пользователя
scr_manager.config.detection.classifier.window_size	количество значений, используемых в методе скользящего окна
scr_manager.config.detection.default_params.hadamard_matrix_size	размер матрицы Адамара
scr_manager.config.detection.default_params.secret1	секретное слово 1
scr_manager.config.detection.default_params.secret2	секретное слово 2

4.1.3.5. Описание параметров key_db

Ниже представлена таблица, которая описывает параметры, используемые key_db сервером при деплое и работе.

Параметр	Описание
key_db.enabled	Добавление сервиса в деплой.
key_db.ingress.enabled	Добавление ingress в деплой.
key_db.ingress.host.admin	Адрес ingress администратора.
key_db.system.password	Пароль, используемый для доступа.
key_db.system.timeout	Таймаут.
key_db.system.NodeIp	Ip адрес ноды, через которую будет осуществляться доступ.
key_db.system.NodePort	Порт ноды, через который будет осуществляться доступ.

4.1.3.6. Описание параметров ags

Ниже представлена таблица, которая описывает параметры, используемые ags сервером при деплое и работе.

Параметр	Значение по умолчанию	Описание
ags.enabled	true	Добавление сервиса в деплой
ags.replicas	2	Кол-во реплик пода
ags.hpa.enabled	true	Включение hpa
ags.hpa.minReplicas	2	Минимальное число hpa реплик
ags.hpa.maxReplicas	4	Максимальное число hpa реплик
ags.hpa. targetCPUUtilizationPercentage	25	Процент целевого использования CPU для hpa
ags.ingress.hostname	ags.testdrp.tz.cas	Адресс ingress'a сервиса
ags.ingress.enabled	true	Добавление ingress в деплой
ags.ingress.useHttps	true	Включение https ингреса
ags.ingress.annotations		Аннотации для ингресса
ags.resources.default_limits.cpu	128m	Ограничение cpu для пода
ags.resources.default_limits. memory	128Mi	Ограничение памяти для пода
ags.resources.default_requests. cpu	96m	Запрос cpu для пода
ags.resources.default_requests. memory	64Mi	Запрос памяти для пода
ags.system.mds.mr_address	am.testmds.cas.local	Адрес account manager'a
ags.system.mds.mr_protocol	http	Протокол связи с account manager'a
ags.system.mds.service_name	configManagerService	Название сервиса для работы с account manager'ом
ags.system.manager_host	scr-manager-svc	"Адрес manager сервиса. Если используется manager в качестве docker-контейнера, то требуется указать его адрес в офрмате ""<ip-сервера>:<port из SYSTEM_HTTP_ADDRESS>"""
ags.env.GOGC	'400'	Определяет отношение между процессором GC и памятью в GO

ags.env.GOMAXPROCS	'1'	Максимальное количество потоков GO
ags.configEnabled	true	Использование конфига
ags.config.logger.level	debug	Степень логирования событий. Возможные значения: trace, debug, info, warning, error, fatal
ags.config.errors.api_not_allowed.code	4009	Код ошибки api_not_allowed
ags.config.errors.api_not_allowed.title	''	Заголовок ошибки api_not_allowed
ags.config.errors.api_not_allowed.detail	'API Not Found'	Детали об ошибке api_not_allowed
ags.config.errors.method_not_allowed.code	4010	Код ошибки method_not_allowed
ags.config.errors.method_not_allowed.title	Method Not Allowed	Заголовок ошибки method_not_allowed
ags.config.errors.method_not_allowed.detail	''	Детали об ошибке method_not_allowed
ags.config.errors.internal.code	4008	Код ошибки internal
ags.config.errors.internal.title	Внутренняя ошибка	Заголовок ошибки internal
ags.config.errors.internal.detail	''	Детали об ошибке internal

4.1.3.7. Описание параметров ags_ui

Параметры, используемые scrambler_ui сервером при деплое и работе:

Параметр	Значение по умолчанию	Описание
scrambler_ui.enabled	true	Добавление сервиса в деплой
scrambler_ui.replicas	1	Кол-во реплик пода
scrambler_ui.ingress.host	web.testscrambler.tz.cas	Адрес ingress
scrambler_ui.ingress.annotations		Аннотации для ингресса
scrambler_ui.resources.limits.cpu	400m	Ограничение cpu для пода
scrambler_ui.resources.limits.memory	256Mi	Ограничение памяти для пода
scrambler_ui.resources.requests.cpu	128m	Запрос cpu для пода
scrambler_ui.resources.requests.memory	192Mi	Запрос памяти для пода

4.1.3.8. Описание параметров ags_web

Описание используемых параметров ags_web аналогично описанию, приведенному в разделе *Описание параметров ags* (вместо префикса ags используется префикс ags_web).

4.1.3.9. Описание параметров ems

Ниже представлена таблица, которая описывает параметры, используемые ems сервером при деплое и работе.

Параметр	Значение по умолчанию	Описание
ems.enabled	false	Добавление сервиса в деплой
ems.replicas	1	Кол-во реплик пода
ems.hpa.enabled	true	Включение hpa
ems.hpa.minReplicas	1	Минимальное число hpa реплик
ems.hpa.maxReplicas	3	Максимальное число hpa реплик
ems.hpa. targetCPUUtilizationPercentage	25	Процент целевого использования CPU для hpa
ems.ingress.hostname	ems.testshield.tz.cas	Адресс ingress'a сервиса
ems.ingress.enabled	false	Добавление ingress в деплой
ems.ingress.annotations		Аннотации для ингресса
ems.resources.limits.cpu	24m	Ограничение cpu для пода
ems.resources.limits.memory	54Mi	Ограничение памяти для пода
ems.resources.requests.cpu	15m	Запрос cpu для пода
ems.resources.requests. memory	48Mi	Запрос памяти для пода
ems.env.GOGC	'400'	Определяет отношение между процессором GC и памятью в GO
ems.env.GOMAXPROCS	'1'	Максимальное количество потоков GO
ems.configEnabled	true	Использование конфига
ems.config.logger.level	debug	Степень логирования событий. Возможные значения: trace, debug, info, warning, error, fatal
ems.config.errors. api_not_allowed.code	4004	Код ошибки api_not_allowed
ems.config.errors. api_not_allowed.title	''	Заголовок ошибки api_not_allowed
ems.config.errors. api_not_allowed.detail	'API Not Found'	Детали об ошибке api_not_allowed

ems.config.errors.method_not_allowed.code	4005	Код ошибки method_not_allowed
ems.config.errors.method_not_allowed.title	''	Заголовок ошибки method_not_allowed
ems.config.errors.method_not_allowed.detail	''	Детали об ошибке method_not_allowed
ems.config.errors.internal.code	4102	Код ошибки internal
ems.config.errors.internal.title	Нет связи с сервером. Попробуйте позже (код 20)	Заголовок ошибки internal
ems.config.errors.internal.detail	'Method Not Allowed'	Детали об ошибке internal

4.1.3.10. Описание параметров scrambler_ui

Ниже представлена таблица, которая описывает параметры, используемые scrambler_ui сервером при деплое и работе.

⚠ Внимание! При установке Dremark необходимо, чтобы значение параметра VITE_HAS_WATERMARK было установлено в "true".

Параметр	Значение по умолчанию	Описание
scrambler_ui.enabled	true	Добавление сервиса в деплой
scrambler_ui.replicas	1	Кол-во реплик пода
scrambler_ui.ingress.host	web.testscrambler.tz.cas	Адрес ingress
scrambler_ui.ingress.annotations		Аннотации для ингресса
scrambler_ui.resources.limits.cpu	400m	Ограничение cpu для пода
scrambler_ui.resources.limits.memory	256Mi	Ограничение памяти для пода
scrambler_ui.resources.requests.cpu	128m	Запрос cpu для пода
scrambler_ui.resources.requests.memory	192Mi	Запрос памяти для пода
scrambler_ui.env.VITE_API_URL	http://ags-web.scrambler.dev.int	Адрес сервиса точки входа для web запросов
scrambler_ui.env.VITE_HAS_WATERMARK	true	Использование версии scrambler_ui для dremark

4.1.3.11. Описание параметров scrambler_web

Описание используемых параметров scrambler_web аналогично описанию, приведенному в разделе Описание параметров ags. Данный сервис требуется для взаимодействия со scrambler_ui.

4.1.3.12. Описание параметров секции db

В секции db представлены параметры, необходимые для установки kob с БД.

Параметр	Значение по умолчанию	Описание
pg_db.db_mng	scr	Наименование базы данных системы
pg_db.db_route	route	Наименование базы данных сервиса route
pg_db.db_mgs	mgs	Наименование базы данных сервиса mgs
pg_db.address	11.111.11.111	Адрес БД
pg_db.master_port	5432	Порт доступа до БД
pg_db.async_port	5432	Порт доступа до асинхронной реплики БД
pg_db.sync_port	5432	Порт доступа до синхронной реплики БД
pg_db.timeout	50	Таймаут соединения к БД
pg_db.max_conns	10	Максимальное количество соединений к БД
pg_db.max_conn_lifetime	10	Максимальная длительность соединения с БД
pg_db.max_conn_idle_time	10	Максимальная длительность idle клиента БД

4.1.3.13. Описание параметров секции mgs

Параметры, используемые mgs сервисом при деплое и работе представлены в таблице. Предназначен для хранения и взаимодействия с абонентами в системе DREMARK:

Параметр	Значение по умолчанию	Описание
mgs.enabled	true	Добавление сервиса в деплой
mgs.replicas	1	Кол-во реплик пода
mgs.hpa.enabled	true	Включение hpa
mgs.hpa.minReplicas	1	Минимальное число hpa реплик
mgs.hpa.maxReplicas	5	Максимальное число hpa реплик
mgs.hpa. targetCPUUtilizationPercentage	35	Процент целевого использования CPU для hpa
mgs.resources.limits.cpu	102m	Ограничение cpu для пода

mgs.resources.limits.memory	128Mi	Ограничение памяти для пода
mgs.resources.requests.cpu	96m	Запрос cpu для пода
mgs.resources.requests.memory	64Mi	Запрос памяти для пода
mgs.ingress.hostname	mgs.scrambler.test.ru	Адресс ingress'a сервиса
mgs.ingress.enabled	false	Добавление ingress в деплой
mgs.ingress.annotations		Аннотации для ингресса
mgs.env.GOGC	400	Определяет отношение между процессором GC и памятью в GO
mgs.env.GOMAXPROCS	1	Максимальное количество потоков GO
mgs.configEnabled	true	Использование конфига
mgs.config.logger.level	debug	Степень логирования событий. Возможные значения: trace, debug, info, warning, error, fatal

4.1.3.14. Описание параметров секции plm

Сервис Playlist Manager выпускается как отдельный продукт. Документ с описанием параметров предоставляется заказчику по требованию. Сервис подготавливает необходимые для проигрывания ссылки на плейлисты.

4.1.3.15. Описание параметров секции apf_server

Параметры, используемые apf_server сервисом при деплое и работе, представлены в таблице нмже. Данный сервис является частью сервиса AGS. Используется для маршрутизации входящих запросов при создание пользователей в DREMARK, является входной точкой для внешних клиентов.

Параметр	Значение по умолчанию	Описание
apf_server.enabled	true	Добавление сервиса в деплой.
apf_server.replicas	1	Количествово реплик пода.
apf_server.hpa.enabled	true	Включение hpa
apf_server.hpa.minReplicas	1	Минимальное число hpa реплик.
apf_server.hpa.maxReplicas	2	Максимальное число hpa реплик
apf_server.hpa.targetCPUUtilizationPercentage	25	Процент целевого использования CPU для hpa
apf_server.ingress.hostname	apf.testdrp.tz.cas	Адресс ingress'a сервиса

apf_server.ingress.enabled	true	Добавление ingress в деплой
apf_server.ingress.useHttps	false	Включение https ингреса
apf_server.ingress.annotations		Аннотации для ингресса
apf_server.resources.default_limits.cpu	102m	Ограничение cpu для пода
apf_server.resources.default_limits.memory	128Mi	Ограничение памяти для пода
apf_server.resources.default_requests.cpu	96m	Запрос cpu для пода
apf_server.resources.default_requests.memory	64Mi	Запрос памяти для пода
apf_server.env.GOGC	'400'	Определяет отношение между процессором GC и памятью в GO
apf_server.env.GOMAXPROCS	'1'	Максимальное количество потоков GO
apf_server.configEnabled	true	Использование конфига
apf_server.config.logger.level	debug	Степень логирования событий. Возможные значения: trace, debug, info, warning, error, fatal
apf_server.route_server.host	"route-server"	Адрес route сервера
apf_server.route_server.timeout	5	Таймаут запроса к route серверу
apf_server.route_server.protocol	http	Протокол взаимодействия с toute сервером
apf_server.route_server.health_check	false	Проверка доступности route сервера
apf_server.route_server.sync_period	300	Период синхронизации данных с route сервером
apf_server.route_server.errors.api_not_allowed.code	4009	Код ошибки api_not_allowed
apf_server.route_server.errors.api_not_allowed.title	''	Заголовок ошибки api_not_allowed
apf_server.route_server.errors.api_not_allowed.detail	'API Not Found'	Детали об ошибке api_not_allowed
apf_server.route_server.errors.method_not_allowed.code	4010	Код ошибки method_not_allowed

apf_server.route_server.errors.method_not_allowed.title	'Method Not Allowed'	Заголовок ошибки method_not_allowed
apf_server.route_server.errors.method_not_allowed.detail	''	Детали об ошибке method_not_allowed
apf_server.route_server.errors.internal.code	4008	Код ошибки internal
apf_server.route_server.errors.internal.title	Внутренняя ошибка	Заголовок ошибки internal
apf_server.route_server.errors.internal.detail	''	Детали об ошибке internal

4.1.3.16. Описание параметров секции route_server

Параметры, используемые route_server сервисом при деплое и работе, представлены в таблице ниже. Данный сервис является частью сервиса AGS. Используется для настройки маршрутизации для apf_server.

Параметр	Значение по умолчанию	Описание
route_server.enabled	true	Добавление сервиса в деплой.
route_server.replicas	1	Количествово реплик пода.
route_server.pullPolicy	Always	
route_server.imagePullPolicy	Always	
route_server.hpa.enabled	false	Включение hpa
route_server.hpa.minReplicas	2	Минимальное число hpa реплик.
route_server.hpa.maxReplicas	4	Максимальное число hpa реплик
route_server.hpa.targetCPUUtilizationPercentage	25	Процент целевого использования CPU для hpa
route_server.ingress.hostname	rs.testdrp.tz.cas	Адресс ingress'a сервиса
route_server.ingress.enabled	true	Добавление ingress в деплой
route_server.ingress.annotations		Аннотации для ингресса
route_server.env.GOGC	'400'	Определяет отношение между процессором GC и памятью в GO
route_server.env.GOMAXPROCS	'1'	Максимальное количество потоков GO
route_server.configEnabled	true	Использование конфига

route_server.config.logger.level	debug	Степень логирования событий. Возможные значения: trace, debug, info, warning, error, fatal
route_server.constraints.route_add_request_max_links	300	Максимальное число дополнительных запросов в рамках одного route
route_server.resources.limits.cpu	102m	Ограничение cpu для пода
route_server.resources.limits.memory	128Mi	Ограничение памяти для пода
route_server.resources.requests.cpu	96m	Запрос cpu для пода
route_server.resources.requests.memory	64Mi	Запрос памяти для пода

4.1.3.17. Настройка переменных окружения

Настройка переменных осуществляется в gitlab.

В боковом меню выбрать **Settings** (на панели слева) -> **CI/CD** -> **Environment variables**. Отредактировать переменные. Для корректной работы и развертывания системы должны быть заданы следующие переменные:

1. ERRMAPDB_LOGIN - логин для подключения к ErrMap DB.
2. ERRMAPDB_PASSWORD - пароль для подключения к ErrMap DB.
3. SCRDB_LOGIN - логин для подключения к SCR DB.
4. SCRDB_PASSWORD - пароль для подключения к SCR DB.
5. POSTGRES_LOGIN - имя администратора PostgreSQL БД.
6. POSTGRES_PASSWORD - пароль администратора PostgreSQL БД.
7. SCRAMBLER_CRT - сертификат для использования в scrambler_ui.
8. SCRAMBLER_KEY - ключ для использования scrambler_ui.
9. MGSDB_LOGIN - логин для подключения к MGS DB.
10. MGSDB_PASSWORD - пароль для подключения к MGS DB.
11. ROUTEDB_LOGIN - логин для подключения к ROUTE DB.
12. ROUTEDB_PASSWORD - пароль для подключения к ROUTE DB.

ⓘ Параметры _LOGIN и _PASSWORD задаются пользователем и используются при подключении к соответствующим базам данных.

4.2. Подготовка локального сервера для запуска задач

4.2.1. Установка пакетов Docker

Способ установки docker-ce определяется установщиком.

4.2.2. Добавление пользователя в группу docker

В базовом варианте установки Docker, команды для управления docker-cli выполняются от имени суперпользователя. Чтобы этого избежать необходимо добавить текущего пользователя(отличного от **root**) в группу docker:

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

Чтобы изменения вступили в силу требуется выполнить перезагрузку.

4.2.3. Обновление драйверов

Для корректной работы транскодирования на GPU на сервере для запуска задач необходим драйвер видеокарты nvidia не ниже версии 520.56.06.

При необходимости удалить все пакеты для видеокарты NVIDIA можно воспользоваться следующей последовательностью команд:

- Поиск всех пакетов в системе

  ```
  sudo dpkg -l | grep -i nvidia
  ```

- Далее необходимо удалить все пакеты, кроме common(он используется для работы с графическим интерфейсом, если в нем нет необходимости и работа производится только из терминал, то его тоже можно удалить)

  ```
  sudo apt remove --purge packet1 packet2
  ```

- Рекомендуется выполнить перезагрузку

  ```
  sudo reboot 0
  ```

Чтобы уточнить версию уже установленного драйвера используется команда:

```
nvidia-smi
```

Пункт Driver Version. Если на сервере установлен драйвер младшей версии, то его необходимо обновить.

> Способ выполнения этой операции остается **на усмотрение заказчика/установщика**.

Ниже представлен пример обновления драйвера nvidia из PPA-репозитория.

Для начала необходимо добавить PPA-репозиторий в исходники системы:

```
sudo add-apt-repository ppa:graphics-drivers/ppa
```

Далее необходимо обновить его кеш используя команду apt:

```
sudo apt update
```

Затем необходимо установить драйвер для графики nvidia:

```
sudo apt install nvidia-driver-550
```

Рекомендуется выполнить перезагрузку%

```
sudo reboot 0
```

4.2.4. Установка nvidia-container-toolkit

Полный User guide представлен здесь. Ниже будет представлена краткая последовательность действий:

```
distribution=$(. /etc/os-release;echo $ID$VERSION_ID) \
curl -s -L https://nvidia.github.io/nvidia-docker/gpgkey | sudo apt-key add - \
curl -s -L https://nvidia.github.io/nvidia-docker/$distribution/nvidia-docker.list | sudo tee /etc/apt/sources.
list.d/nvidia-docker.list \
sudo apt-get update \
sudo apt-get install -y nvidia-docker2 \
sudo systemctl restart docker
```

При появлении ошибки вида:

```
# Unsupported distribution!
# Check https://nvidia.github.io/nvidia-docker
```

Необходимо посетить страницу с полным User guide.

Перейти к пункту Linux Distributions. И найти необходимый для установленной ОС ***Identifier.***

После чего выполнить представленный выше команды, заменив ***$distribution*** на ***Identifier*** установленной OS.

После установки nvidia-container-toolkit необходимо проверить, что в файле docker демона по пути /etc/docker /daemon.json изменились настройки и они соотвествуют:

```
{
    "default-runtime": "nvidia",
    "runtimes": {
        "nvidia": {
            "path": "/usr/bin/nvidia-container-runtime",
            "runtimeArgs": []
        }
    }
}
```

Рекомендуется выполнить перезагрузку:

```
sudo reboot 0
```

4.2.5. Патч драйвера nvidia

Для работы ffmpeg в многопоточном режиме(более двух транскодирований одновременно) необходимо произвести патч драверов nvidia по актуальной инструкции, представленной в https://github.com/keylase/nvidia-patch.

4.2.6. Запуск docker daemon

Docker-daemon используется для работы с docker через REST API. Данный сервис автоматически устанавливается вместе с другими пакетами Docker.

Для запуска docker-daemon на 2375 порту необходимо выполнить следующую команду:

```
sudo systemctl stop docker
sudo dockerd -H unix:// -H 0.0.0.0:2375 &
```

Далее необходимо проверить, что docker-daemon на нужной машине запущен, путем выполнения curl запроса к host-машине для получения списка запущенных контейнеров:

```
curl --location 'http://<host-ip>:2375/containers/json'
```

В ответ должен вернуться пустой список "[]".

После запуска dockerd необходимо убедиться, что максимальное количество открытых >файлов для него составляет 1048576. Находим процесс dockerd%

```
pidof dockerd
```

Проверяем установленные лимиты:

```
cat /proc/$(pidof dockerd)/limits
```

Должна появится следующая таблица:

```
Limit                     Soft Limit           Hard Limit           Units
Max cpu time              unlimited            unlimited            seconds
Max file size             unlimited            unlimited            bytes
Max data size             unlimited            unlimited            bytes
Max stack size            8388608              unlimited            bytes
Max core file size        0                    unlimited            bytes
Max resident set          unlimited            unlimited            bytes
Max processes             127781               127781               processes
Max open files            1048576              1048576              files
Max locked memory         67108864             67108864             bytes
Max address space         unlimited            unlimited            bytes
Max file locks            unlimited            unlimited            locks
Max pending signals       127781               127781               signals
Max msgqueue size         819200               819200               bytes
Max nice priority         0                    0
Max realtime priority     0                    0
Max realtime timeout      unlimited            unlimited            us
```

Где "Soft Limit" для "Max open files" будет равен 1048576. Если это не так, то необходимо обновить docker до актуальной версии. Или изменить этот лимит вручную на усмотрение оператора, ниже будет приведен один из способов.

Для увеличения лимита на количество открытых файлов для сервиса Docker можно > отредактировать файл конфигурации systemd. В CentOS/RHEL это может быть файл > docker.service, в Ubuntu/Debian — docker.service.d/override.conf.

1. Отредактируйте файл конфигурации. Например, для CentOS/RHEL:

```
sudo systemctl edit docker.service
```

Для Ubuntu/Debian:

```
sudo mkdir -p /etc/systemd/system/docker.service.d
sudo nano /etc/systemd/system/docker.service.d/override.conf
```

2. Добавьте или отредактируйте параметр LimitNOFILE

```
[Service]
LimitNOFILE=infinity
```

3. Перезапустите сервис Docker:

```
sudo systemctl daemon-reload
sudo systemctl restart docker
```

После этих шагов лимит на количество открытых файлов для сервиса Docker должен быть увеличен.

4.2.7. Mount папки output_dir

В процессе работы система сохраняет результаты работы в папке, наименование которой настраивается при помощи параметра scr_manager.system.instances_settings.output_dir, по умолчанию используется значение "publ".

Данная папка стандартно находится по пути "/var/lib/docker/volumes/publ/". В дальнейшем рекомендуется "маунтить" папку сразу со стандартной директорией "/_data/".

> Данная директория на момент создания сервера может не существовать.

Чтобы артефакты работы системы сохранялись в ***CDN directory***. Необходимо произвести mount папки "publ" на host машине с папкой на удаленном CDN.

> Способ выполнения этой операции остается **на усмотрение заказчика/установщика**.

Ниже приведен пример команды для mount'a сетевой папки без использования аутентификации.

```
sudo mount -t cifs //<DOMAIN>/<MOUNT_DIR_PATH> /var/lib/docker/volumes/publ/_data/ -o users,sec=none
```

5. Настройка взаимодействия системы с Account Manager

Для создания разветвленной системы прав и доступов в UI системы, начиная с версии 1.1, необходимо прописать права пользователей (permissions) в сервисе Account Manager.

Процедура выполняется в следующих случаях:

- при установке системы "с нуля";
- в случае обновления/добавления/удаления прав (permissions).

5.1. Добавление сервиса 'scrambler' в Account Manager

Данная процедура выполняется однократно, до установки/обновления scrambler-permissions в Account Manager. В случае сбоя/переустановки Account Manager "с нуля" этот сервис может быть удалён - в этом случае его придётся создавать заново.

1. Войдите в web-интерфейс Account Manager.
2. В левой части окна выберите Администрирование -> Сервисы. Раздел Services доступен только суперпользователю Account Manager.
3. Проверьте, что в списке сервисов присутствует сервис 'scrambler'. Если его нет, добавьте его. Подробное описание работы с сервисами приведено в документе "Руководство пользователя" Account Manager в разделе "Сервисы".

5.2. Добавление permissions в Account Manager

Добавление permissions происходит с помощью скрипта (предоставляется заказчику по запросу). Подтяните актуальную версию scrambler_routing_init:

```
git clone --recursive git@gitlab.gs-labs.tv:scrambler/scrambler_routing_init.git
```

Загрузите permissions.json (предоставляется заказчику по запросу) в Account Manager с помощью скрипта (предоставляется заказчику по запросу). Для этого используется команда, где host - ACM host name.

```
cp -rf scrambler_routing_init/permissions.json scrambler_routing_init/routing_db_scripts/scripts
cd scrambler_routing_init/routing_db_scripts/scripts
bash create_acm_permissions.sh -h [acm_host]
```

Если запускать скрипт через Linux/Mac, то он работает корректно. На Windows нужно конвертировать файл через dos2unix командой "dos2unix create_acm_permissions.sh", после чего запускать файл.

5.3. Проверка загруженных permissions

1. Войдите в web-интерфейс ACM.
2. В левой части окна выберите Администрирование -> Роли.
3. В появившейся форме, в поле "Название" введите имя административной роли (например, ACM).

4. На экране отобразится список разрешений, доступных для этой роли. Необходимо, чтобы в правый список были добавлены все разрешения для работы с Scrambler (начинаются с scrambler). Если это не так, добавьте их к административной роли. Список разрешений см. в документе Scrambler.Permissions в Confluence.

Если web-интерфейс Scrambler уже запущен, то сбросьте кеш страницы (в браузере) и перелогиньтесь. Если требуется новый пользователь web-интерфейса Scrambler, тогда передайте администратору Account Manager список прав, которые должны быть у этого пользователя.