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 публичным документом.

⚠ **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. Подробная информация о настройках предоставляется по требованию заказчика.

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 конфигурации сервисов

Параметры переменных окружения сервисов системы предоставляются заказчику по требованию.

4.1.3.5. 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 переменные, список которых предоставляется заказчику по требованию.

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. 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.

4.2.6. 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.

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).

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 (предоставляется по требованию заказчика).

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.