Обновление Агредатора

Скрипты обновления:

Состав обновления:

Сбор статистики для поиска "потеряшек" сбор текущей статистики stats1.zip

переотправка запросов переотправка ответов повторный сбор статистики stats2.zip

Скрипты обновления:

Для создания "пакета обновления" и удобной доставки сделали 2 скрипта save.sh и load.sh:



save.sh

##################################################

# Данный скрипт создает архив актуальных образов

# для дальнейшей передачи в виде архива в ДМЗ

# Принцип работы:

# 1. смотрим во всех docker-compose.yml файлы и пулим новейшие образы

# 2. тэгируем их текущей датой и сохраняем всё в архив

#

# Пример:

# ./save.sh

# Результат:

# otr\_2021-04-14\_16.tar.gz - около 2.5GB

# Примерное время: 6 минут

#

##################################################



load.sh

###########################################################################

# Данный скрипт загружает образы из архива и перетегирует

# из для дальнейшего обновления запущенных контейнеров

# Принцип работы:

# 1. смотрим во всех docker-compose.yml файлы и тегиеруем ТЕКУЩИЕ образы как old

# 2. загружаем образы из файла

# 3. тегиурем загруженные образы как otr

# 4. обновляем только те контейнеры, которые изменились

# 5. infra НЕ перезапускаются автоматически

#

# Предусловия:

# - папки infra, core, service

# - каждая папка содержит .env файлы в которых SERVICE\_TAG, CORE\_TAG, CONSUL\_TAG и INFRA\_TAG строго указано как otr

#

# Пример:

# ./load.sh otr\_2021-04-14\_16.tar.gz

# Результат:

# Старые образы имеют название old, новые otr

# Примерное время: 3 минуты

#

##################################################

Сам файл с архивом (результат запуска save.sh) можно выполнить как на площадке OTR так и скачать заранее подготовленный нами и загруженный на owncloud:

Далее на продуктовой площадке необходимо запустить load.sh и в качестве аргумента ему передать файл с архивом.

При таком подходе обновляться будут все образы/контейнеры входящие в комплект Агредатор, но перезапускаться будут только те, в которых действительно были изменения.

Поскольку объем диска на продуктовой машине ограничен (в разделе root) да и вообще для очистки внутреннего хранилища docker необходимо регулярно перед обновлением запускать очистку:

docker system prune -a

Данная команда будет удалять все остановленные контейнеры и образы не используемые в данный момент на машине.

Предусловия:

* папки infra, core, service
* каждая папка содержит .env файлы в которых SERVICE\_TAG, CORE\_TAG, CONSUL\_TAG и INFRA\_TAG строго указано как "otr"

Состав обновления:

в данное обновление вошло:

обновление core (админка агредатора) - реализована выгрузка отладочной информации для поиска участков, теряющих сообщения обновление otr-gate - переотправка запросов 3-7, переотправка ответов 3-7, сбор статистики по потерям smev3 - обработка множественных статусов для incoming сообщений

Сбор статистики для поиска "потеряшек"

Этот процесс необходимо сделать в 3 этапа (ПОСЛЕ текущего обновления):

* 1. сбор текущей статистики stats1.zip
  2. переотправка запросов
  3. переотправка ответов
  4. небольшое ожидание минут 5-10
  5. повторный сбор статистики stats2.zip

# сбор текущей статистики stats1.zip

Сначала необходимо настроить параметры сбора статистики: в consul в разделе aggredator/otrtest/services/env/otr-gate/ необходимо создать ключ STAT\_CONFIG с таким содержимым:

kafka:

in\_raw:

topic: sx.smev.aggredator.request

envelope\_elements:

- orderId

out:

topic: smev.out

extra\_body\_fields:

- payload/data\_request/order\_id

extra\_headers\_fields:

- message\_id

services:

smev3:

db\_url: postgresql://postgres:otrtest@db:5432/aggredator-service-smev3

incomings:

epgu-child-allowance:

db\_url: postgres://postgres:otrtest@db:5432/aggredator-service-epgu-child-allowance

extra\_fields:

- data\_request/order\_id # search in response or metadata # json pointer

requests:

zags-inform:

db\_url: postgres://postgres:otrtest@db:5432/aggredator-service-zags-inform

extra\_fields:

- ags\_type

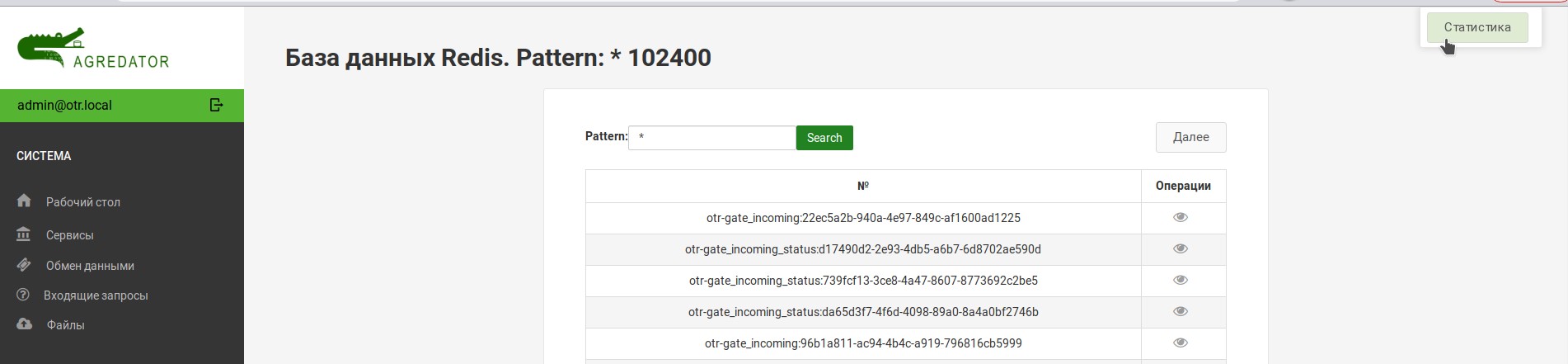
- request\_id

в consul в разделе aggredator/otrtest/services/env/otr-gate/ необходимо создать ключ EPGU\_DB\_URL с таким содержимым: postgresql://postgres:otrtest@db:5432/aggredator-service-epgu-child-allowance?pool=5 Затем в папке services необходимо запустить следующую команду:

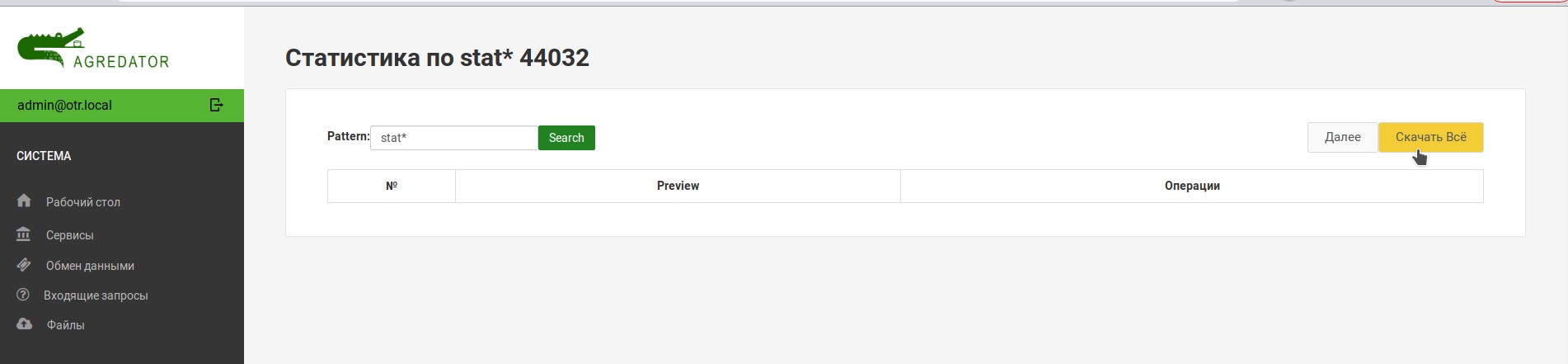
docker-compose run --rm -e AGGREDATOR\_COMPONENT=collect\_stat otr-gate

Она выполняется некоторое время - около 5ти минут. В случае непредвиденного завершения требуется передать нам выхлоп ошибки.

Затем через адмнистративную панель Агредатора зайти в раздел "Redis" (слева внизу) и после загрузки перейти в "Статистика" - справа вверху:



и далее, убедившись что в поле Pattern написано "stat\*" нажать кнопку "Скачать всё":



Этот процесс также занимает некоторое время - около минуты. В результате на компьютер пользователя будет скачен zip-файл, содержащий статистику включающую в себя состояние топиков kafka, статусы записей в БД сервисов и пр. никаких ПД в данный файл не попадает - его можно открыть штатными средствами любой ОС и просмотреть текстовые файлы внутри.

# переотправка запросов

В папке services необходимо запустить следующую команду:

docker-compose run --rm -e AGGREDATOR\_COMPONENT=smev\_request\_resender otr-gate

# переотправка ответов

В папке services необходимо запустить следующую команду:

docker-compose run --rm -e AGGREDATOR\_COMPONENT=resend\_responses otr-gate

# повторный сбор статистики stats2.zip

Затем через 5-10 минут в папке services необходимо запустить следующую команду:

docker-compose run --rm -e AGGREDATOR\_COMPONENT=collect\_stat otr-gate

Дождаться завершения и опять в админке выполнить "redis" → "Статистика" → "Скачать всё"