UNLIMITED PRODUCTION

eXpress
Система коммуникаций

Руководство администратора

Обновление

Сборка 3.14
03.04.2024

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Руководство предназначено для администраторов изделия «Система коммуникаций «Express» (далее – СК «Express», Express, система). В нем содержатся сведения, необходимые для обновления и настройки системы.

Служба технической поддержки. Связаться со службой технической поддержки можно по электронной почте support@express.ms. Страница службы технической поддержки на сайте компании «Анлимитед продакшен» https://express.ms/faq/.

Сайт в интернете. Информацию о продукте компании «Анлимитед продакшен» представлена на сайте https://express.ms/.

ОБНОВЛЕНИЕ ОС

Для обновления ОС серверных компонентов Express:

1. Остановите контейнеры ПО Express. Для этого выполните команду:

   ```
   cd /opt/express && DPL_PULL_POLICY=never dpl --dc stop
   ```

   Затем выполните команду:

   ```
   cd /opt/express-voice && DPL_PULL_POLICY=never dpl --dc stop
   ```

2. Убедитесь, что все контейнеры остановлены (находятся в статусе «exited»). Для этого на каждом сервере выполните команду:

   ```
   docker ps -a
   ```

3. Выключите сервис Docker при помощи команды:

   ```
   systemctl stop docker.service docker.socket
   ```

4. Выполните резервное копирование каталога /var/lib/docker или сделайте резервную копию образа виртуальной машины.

5. Обновите компоненты ОС.

   Примечание. Процедура обновления зависит от вида ОС. При обновлении следуйте инструкциям производителя по обновлению ОС.

   Пример команды для Debian/Ubuntu/Astra linux:

   ```
   apt-get update && apt-get upgrade
   ```

   Пример команды для Red Hat/Centos/AlmaLinux/Rocky Linux:

   ```
   yum update
   ```

6. После обновления ОС перезагрузите систему и убедитесь, что ОС загрузилась.

7. Проверьте, запустились ли все контейнеры, с помощью команды:

   ```
   docker ps -a
   ```

   Если контейнеры не запустились, для просмотра журнала событий выполните команду:

   ```
   dpl --dc logs --tail=200  <не_запускаемый_контейнер>
   ```

   Если процедура обновления сервера выполнена правильно, через пять минут будет доступна консоль администратора (веб-интерфейс) https://ccs_host/admin.

   Проверьте, отображаются ли пользователи в разделе «Пользователи» в консоли администратора.

8. Проверьте, функционируют ли в СК «Express»:

   - сообщения в чатах;
   - персональные звонки;
   - групповые звонки.

РУЧНОЕ ОБНОВЛЕНИЕ СЕРВЕРА

Внимание! Выполните резервное копирование перед выполнением процедуры обновления! Дальнейший процесс обновления актуален для версии eXpress 3.x и выше. Если версия eXpress ниже 3.0 перейдите в раздел «Обновление eXpress до версии 3.0».

SINGLE CTS

Обновление сервера VoEx:

1. Перейдите в директорию Express cd /opt/express-voice/.
2. Остановите сервисы из директории Express:

```
DPL_PULL_POLICY=never dpl --dc stop
```

3. Запустите обновление:

```
dpl -d
```

4. Проверьте логи на наличие ошибок командой:

```
dpl --dc logs --tail=200 -f
```

Для обновления сервера CTS:

Примечание:

- 22.07.2022 Начиная с версии 2.4 изменились минимальные требования по версии postgres. Теперь поставляется образ postgresql версии 14.4. Процедура обновления встроенной БД описана на стр. 12.
- 15.12.2022. Перед обновлением сервера на версию 2.6 настройте Kafka. Процедура настройки описана на стр. 20.

1. Перейдите в директорию Express cd /opt/express/.
2. Остановите сервисы из директории Express:

```
DPL_PULL_POLICY=never dpl --dc stop
```

3. Выполните резервное копирование /var/lib/docker/volumes (или где они лежат в этой системе).

   При выполнении обновления сервера с версии 1.28 измените хозяина файлов для сервисов (делается один раз):

```
docker volume inspect --format '{{ .Mountpoint }}'
cts_ccs_admin_public cts_file_service_uploads
cts_messaging_cache cts_messaging_uploads cts_phonebook_uploads
| xargs sudo chown -R 888:888
```

   Если предыдущая версия nginx меньше, чем 1.20.1, и используются letsencrypt сертификаты:

   - Очистите хранилище letsencrypt (один раз):

```
rm -rf cts/letsencrypt
dpl cadvinstall && dpl nxinstall
```

4. Обновите node exporter и container advisor:

```
dpl cadvinstall && dpl nxinstall
```

5. Запустите обновление:

```
dpl -d
```

После запуска обновления требуется время на проведение внутренних процедур (ориентировочное время 10-15 минут).

6. Проверьте логи на наличие ошибок командой:

```
dpl --dc logs --tail=200 -f
```

Внимание! С версий 2.2 и 2.3 откатываться назад нельзя!

Для отката обновления поправьте файл **settings**, указав параметр, например:

```
images:
     trusts: ccs/trusts:1.28.0
```

FRONT CTS И BACK CTS

Обновление сервера VoEx:

1. Перейдите в директорию Express cd /opt/express-voice/.
2. Остановите сервисы из директории Express:

```
DPL_PULL_POLICY=never dpl --dc stop
```

3. Запустите обновление:

```
dpl -d
```

4. Проверьте логи на наличие ошибок командой:

```
dpl --dc logs --tail=200 -f
```

Внимание! Перед началом процедуры обновления проверьте изменения по таблице сетевого взаимодействия!

Примечание:

- 05.03.2022 Для установок с разделением на frontend и backend нужно убедиться, что с frontend сервера доступны tcp порты 2379, 5432, 6379, 9092 на сервере backend. Также желательно закрыть доступ к этим портам отовсюду кроме frontend.
- 22.07.2022 **Начиная с версии 2.4 изменились минимальные требования по версии postgres. Теперь поставляется образ postgresql версии 14.4.** Процедура обновления встроенной БД описана на стр. 12.
- 15.12.2022. Перед обновлением сервера на версию 2.6 настройте Kafka. Процедура настройки описана на стр. 20.

Первым обновляется сервер Front CTS, затем сервер Back CTS.

Для обновления севера Front CTS:

1. Перейдите в директорию Express cd /opt/express/.
2. Остановите работу приложения командой:

```
DPL_PULL_POLICY=never dpl --dc stop
```

3. Выполните резервное копирование файлов /var/lib/docker/volumes (после нескольких дней эксплуатации резервные копии файлов можно удалить).

Если предыдущая версия nginx меньше, чем 1.20.1, и используются letsencrypt сертификаты:

- Очистите хранилище letsencrypt (один раз):

```
rm -rf cts/letsencrypt
```

4. Запустите обновление:

```
dpl -d
```

После запуска обновления требуется время на проведение внутренних процедур (ориентировочное время 10-15 минут).

5. Проверьте логи на наличие ошибок командой:

```
dpl --dc logs --tail=200 -f
```

Для обновления севера Back CTS:

1. Перейдите в директорию Express cd /opt/express/.
2. Остановите работу приложения командой:

```
DPL_PULL_POLICY=never dpl --dc stop
```

3. Выполните резервное копирование файлов /var/lib/docker/volumes (после нескольких дней эксплуатации скопированные файлы можно удалить).
4. Если версия сервера ниже 1.28, выполните:

```
docker volume inspect --format '{{ .Mountpoint }}'
cts_ccs_admin_public \ cts_file_service_uploads
cts_messaging_cache cts_messaging_uploads \
cts_phonebook_uploads | xargs sudo chown -R 888:888
dpl cadvinstall && dpl nxinstall
```

5. Запустите обновление:

```
dpl -d
```

После запуска обновления требуется время на проведение внутренних процедур (ориентировочное время 10-15 минут).

6. Проверьте логи на наличие ошибок командой:

```
dpl --dc logs --tail=200 -f
```

ОБНОВЛЕНИЕ ДО EXPRESS 3.0

Внимание! Версия Docker должна быть 20.10.23 или выше.

Обновление сервера VoEx + Front (либо Single):

1. Перейдите в директорию Express cd /opt/express-voice/.
2. Остановите сервисы из директории Express:

```
DEPLOYKA_SKIP_UPDATE=true dpl --dc down
```

3. Установите новую версию deployka:

```
docker run --rm registry.public.express/dpl:cts-release dpl-
install | bash
```

4. Выполните миграцию конфигураций командой:

```
dpl dpl-upgrade
```

5. Запустите обновление командой:

```
dpl -p && dpl -d
```

6. Перейдите в директорию Express cd /opt/express
7. Выполните миграцию конфигураций командой:

```
dpl dpl-upgrade
```

8. Запустите обновление командой:

```
dpl -p && dpl -d
```

Обновление сервера Back:

1. Перейдите в директорию Express cd /opt/express
2. Остановите сервисы из директории Express:

```
DEPLOYKA_SKIP_UPDATE=true dpl --dc stop
```

3. Установите новую версию deployka:

```
docker run --rm registry.public.express/dpl:cts-release dpl-
install | bash
```

4. Выполните миграцию конфигураций командой:

```
dpl dpl-upgrade
```

5. Запустите обновление командой:

```
dpl -p && dpl -d
```

ОТКАЗОУСТОЙЧИВАЯ КОНФИГУРАЦИЯ

В случае невозможности использования скриптов автоматического обновления выполните обновление в ручном режиме.

Для обновления отказоустойчивой конфигурации:

1. Скопируйте новые версии образов docker и скрипт загрузки образов в каталог /tmp/images и загрузите образы с помощью скрипта load.sh:

```
cp *.tar /tmp/images/
cp /opt/deploy/script/load.sh /tmp/images/
cd /tmp/images/
./load.sh
```

2. Подключитесь к консоли сервера Back и Front кластера с индексом 01 и 02.
3. Выполните команду для остановки антивируса:

```
systemctl stop kesl klnagent64
```

Примечание. В случае зависания сервера при остановке антивируса, перезагрузите оба узла кластера через систему виртуализации.

4. Выполните команду:

```
pcs status
```

5. Убедитесь, что ресурсы кластера запущены согласно списку ниже:
   - dlm-clone [dlm] (back кластер) – запущен на обоих узлах кластера;
   - clvmd-clone [clvmd] (back кластер) – запущен на обоих узлах кластера;
   - clusterfs-clone [clusterfs] (back кластер) – запущен на обоих узлах кластера;
   - cluster_ip – запущен на одном узле кластера;
   - dockerd – запущен на одном узле кластера;
   - node_exporter (back кластер) – запущен на одном узле кластера;
   - cadvisor (back кластер) – запущен на одном узле кластера;

- vmfence (back кластер) – запущен на одном узле кластера.

Если статус ресурсов кластера не соответствует перечисленным выше, выполните команду, заменив *resource_name* на имя проблемного ресурса:

```
pcs resource cleanup resource_name
```

6. На узлах кластера Back с индексом 01 и 02 выполните команду ниже:

```
ls -la /opt/ex_data/files
```

Примечание. В случае зависание вывода списка директорий необходимо перезагрузить оба узла кластера через систему виртуализации.

7. Подключитесь к консоли сервера Back кластера с индексом 01 или 02 и выполните команду:

```
pcs status | grep dockerd
```

Примечание. Команда выполняется для определения текущего первичного узла, на котором запущены ресурсы кластера.

8. Подключитесь к консоли текущего первичного узла кластера Back и последовательно выполните команды:

```
cd /opt/express
dpl -g
```

9. Подключитесь к консоли сервера Front кластера с индексом 01 или 02 и выполните команду:

```
pcs status | grep dockerd
```

Примечание. Команда выполняется для определения текущего первичного узла, на котором запущены ресурсы кластера.

10. Подключитесь к консоли текущего первичного узла кластера Front и последовательно выполните команды:

```
cd /opt/express
dpl -g
cd /opt/express-voice
dpl -g
```

11. Подключитесь к консоли вторичного узла кластера Back и последовательно выполните команды, заменив full_fqdn_slave_server на полное доменное имя вторичного узла кластера:

```
pcs resource move cluster_ip full_fqdn_slave_server
pcs resource move dockerd full_fqdn_slave_server
```

12. Дождитесь переключения ресурсов кластера dockerd и cluster_ip на вторичный узел кластера Back. Для мониторинга состояния ресурсов периодически выполняйте команду:

```
pcs status
```

13. После переключения ресурсов на вторичный узел кластера Back последовательно выполните команды:

```
cd /opt/express
dpl -g
```

14. Подключитесь к консоли вторичного узла кластера Front и последовательно выполните команды, заменив full_fqdn_slave_server на полное доменное имя вторичного узла кластера:

```
pcs resource move cluster_ip full_fqdn_slave_server
pcs resource move dockerd full_fqdn_slave_server
```

15. Дождитесь переключения ресурсов кластера dockerd и cluster_ip на вторичный узел кластера Front. Для мониторинга состояния ресурсов периодически выполняйте следующую команду:

```
pcs status
```

16. После переключения ресурсов на вторичный узел кластера Front последовательно выполните команды:

```
cd /opt/express
dpl -g
cd /opt/express-voice
dpl -g
```

17. Подключитесь к консоли текущего первичного узла кластера Back и последовательно выполните команды:

```
cd /opt/express
dpl --dc stop
dpl nxinstall && dpl cadvinstall
dpl -d
```

18. После завершения обновления сервера откройте вывод логов работы контейнеров:

```
dpl --dc logs --tail=100 -f
```

19. Дождитесь остановки вывода логов контейнеров кроме контейнера nginx.
20. Подключитесь к консоли сервера Front кластера с индексом 01 или 02 и выполните команду:

```
pcs status | grep dockerd
```

Примечание. Команда выполняется для определения текущего первичного узла, на котором запущены ресурсы кластера.

21. Подключитесь к консоли текущего первичного узла кластера Front и последовательно выполните команды:

```
cd /opt/express
dpl --dc stop
dpl -d
cd /opt/express-voice
dpl --dc stop
dpl -d
```

22. После обновления первичных узлов кластеров Front и Back проверьте функционирование системы, выполните проверку логов на наличие ошибок командой:

```
dpl --dc logs --tail=200 -f
```

23. Подключитесь к консоли вторичного узла кластера Back и последовательно выполните команды, заменив full_fqdn_slave_server на полное доменное имя вторичного узла кластера:

```
pcs resource move cluster_ip full_fqdn_slave_server
pcs resource move dockerd full_fqdn_slave_server
```

24. Дождитесь переключения ресурсов кластера dockerd и cluster_ip на вторичный узел кластера Back. Для мониторинга состояния ресурсов периодически выполняйте команду:

```
pcs status
```

25. После переключения ресурсов на вторичный узел кластера Back последовательно выполните команды:

```
cd /opt/express
dpl --dc stop
dpl nxinstall && dpl cadvinstall
dpl -d
```

26. После завершения обновления сервера, откройте вывод лога работы контейнеров и дождитесь остановки вывода логов контейнеров кроме контейнера nginx:

```
dpl --dc logs --tail=100 -f
```

27. Подключитесь к консоли вторичного узла кластера Front и последовательно выполните команды, заменив full_fqdn_slave_server на полное доменное имя вторичного узла кластера:

```
pcs resource move cluster_ip full_fqdn_slave_server
pcs resource move dockerd full_fqdn_slave_server
```

28. Дождитесь переключения ресурсов кластера dockerd и cluster_ip на вторичный узел кластера Front. Для мониторинга состояния ресурсов периодически выполняйте команду:

```
pcs status
```

29. После переключения ресурсов на вторичный узел кластера Front последовательно выполните команды:

```
cd /opt/express
dpl --dc stop
dpl -d
cd /opt/express-voice
dpl --dc stop
dpl -d
```

30. Подключитесь к консоли сервера Back и Front кластера с индексом 01 и 02, выполните команду для запуска антивируса:

```
systemctl start kesl klnagent64
```

ОБНОВЛЕНИЕ POSTGRESQL

Важно! Существует риск потери данных в момент обновления БД. Процедуру обновления рекомендуется выполнять во время минимальной пользовательской активности или остановки всех сервисов. Перед процедурой строго необходимо выполнить полное резервное копирование базы данных.

ПРОЦЕДУРА ОБНОВЛЕНИЯ

Перед обновлением встроенной БД:

1. Убедитесь в наличии свободного места на диске.

   Для обновления требуется дисковое пространство, равное по размеру существующей базе. Текущий размер базы можно узнать с помощью команды:

   ```
   docker system df -v | grep postgres_data
   ```

2. Уточните текущую версию PostgreSQL.

   Если после обновления возникнут ошибки, может потребоваться откат версии PostgreSQL. Поэтому желательно знать, на какой версии БД работала до обновления. Важно выполнить команду вывода текущей версии БД до обновления образа PostgreSQL:

   ```
   DPL_PULL_POLICY=never dpl --dc exec postgres cat
   /var/lib/postgresql/data/PG_VERSION
   ```

3. Убедитесь, что имеется доступ к apt.postgresql.org. Доступ требуется для загрузки исполняемых файлов старой версии. При отсутствии доступа см. раздел «Обновление без доступа к apt.postgresql.org».

   Чтобы откатить версию базы, нужно перенести файлы из директории с резервной копией в директорию уровнем выше.

4. В конец файла settings добавьте настройку, запускающую обновление контейнера БД при старте:

   ```
   postgres_upgrade: true
   ```

   Примечание. Если данной настройки не будет, новые версии образа PostgreSQL при запуске будут выдавать ошибку «postgres_upgrade disabled, exiting».

5. Выполните обновление образа PostgreSQL и его запуск с помощью команды:

   ```
   dpl -d postgres
   ```

6. Для отслеживания процесса обновления в логах используйте команду:

   ```
   dpl --dc logs -f --tail=1 postgres
   ```

   Когда обновление будет закончено в логах появится сообщение:

   ```
   DB upgrade is done, please disable postgres_upgrade in the
   settings
   ```

7. Отключите настройку postgres_upgrade, иначе контейнер с PostgreSQL не запустится.
8. Уберите настройку postgres_upgrade и загрузите стандартный образ с помощью команды:

   ```
   DPL_PULL_POLICY=never dpl -d postgres
   ```

Для автоматического обновления воспользуйтесь настройкой postgres_auto_upgrade, с ней база будет автоматически обновляться при выпуске образа с новой версией.

Если версию БД по каким-то причинам нужно оставить без изменений, остаются доступны образы postgres без скрипта обновления. Их можно включить через параметр images, например:

```
images:
  postgres: postgres:9.5.24
```

РЕЗЕРВНАЯ КОПИЯ

После обновления базы старая версия сохраняется для возможности отката версии. Копия расположена в том же volume, который использует контейнер postgres. В директории с именем upgrade_backup_<timestamp>.

Если по итогам обновления все сервисы работают нормально, и откат не требуется, удалите резервную копию командой:

```
 dpl --dc exec postgres find /var/lib/postgresql/data -type d -name
upgrade_backup_* -exec rm -r {} \;
```

ОТКАТ ВЕРСИИ

Чтобы откатить версию базы, нужно перенести файлы из директории с резервной копией в директорию уровнем выше.

Для работы с файловой системой volume запустите временный контейнер:

```
DPL_PULL_POLICY=never dpl --dc run --rm --entrypoint=/bin/bash
postgres
```

Или выполните операцию непосредственно с хоста. Docker volumes обычно хранятся в /var/lib/docker/volumes.

После переноса файлов выставите образ postgres предыдущей версии в файле в settings, например:

```
images:
  postgres: postgres:9.5.24
```

ОБНОВЛЕНИЕ БЕЗ ДОСТУПА К APT.POSTGRESQL.ORG

Для обновления без доступа к apt.postgresql.org собран специальный образ 14.4-from-9.5.

Примечание. Поддерживается только обновление с версии 9.5.

Для обновления БД до версии 14.4:

1. В файл settings добавьте секцию images с соответствующим тэгом:

   ```
   images:
     postgres: postgres:14.4-from-9.5
   ```

2. В конец файла settings добавьте настройку, запускающую обновление контейнера БД при старте:

   ```
   postgres_upgrade: true
   ```

Примечание. Если данной настройки не будет, новые версии образа PostgreSQL при запуске будут выдавать ошибку «postgres_upgrade disabled, exiting».

3. Выполните обновление образа PostgreSQL и его запуск с помощью команды:

```
dpl -d postgres
```

4. Чтобы отследить процесс обновления в логах, используйте команду:

```
dpl --dc logs -f --tail=1 postgres
```

Когда обновление будет закончено, в логах появится сообщение:

```
DB upgrade is done, please disable postgres_upgrade in the
settings
```

5. Уберите добавленную секцию images и настройку postgres_upgrade и загрузите стандартный образ с помощью команды:

```
DPL_PULL_POLICY=never dpl -d postgres
```

ОБНОВЛЕНИЕ С ИСПОЛЬЗОВАНИЕМ ANSIBLE-СЦЕНАРИЕВ

SINGLE CTS, BACK CTS И FRONT CTS

Для выполнения обновления всех контейнеров выполните следующие шаги:

1. Запустите ansible playbook:

```
ansible-playbook --ask-pass -v 05-update_cts.yaml
```

2. Введите пароль учетной записи root после ввода команды.

Для обновления ПО, установленного в контейнерах docker, на сервере Registry выполните:

Внимание! При выполнении скриптов обновления нельзя пропускать паузы, заложенные в него, т. к. их пропуск может привести к ошибкам обновления.

1. Скопируйте новые версии образов docker и скрипт загрузки образов в каталог /tmp/images и загрузите образы с помощью скрипта download.sh (скрипт доступен по ссылке):

```
cp *.tar /tmp/images/
cp /opt/deploy/script/load.sh /tmp/images/
cd /tmp/images/
./ download.sh
```

2. Подключитесь к консоли сервера Back и Front кластера с индексом 01 и 02, выполните команду для остановки антивируса:

```
systemctl stop kesl klnagent64
```

Примечание. В случае зависания сервера при остановке антивируса перезагрузите оба узла кластера через систему виртуализации.

3. Выполните команду:

```
pcs status
```

4. Убедитесь, что ресурсы кластера запущены согласно списку ниже:
   - dlm-clone [dlm] (back кластер) – запущен на обоих узлах кластера;
   - clvmd-clone [clvmd] (back кластер) – запущен на обоих узлах кластера;
   - clusterfs-clone [clusterfs] (back кластер) – запущен на обоих узлах кластера;
   - cluster_ip – запущен на одном узле кластера;
   - dockerd – запущен на одном узле кластера;
   - node_exporter (back кластер) – запущен на одном узле кластера;
   - cadvisor (back кластер) – запущен на одном узле кластера;
   - vmfence (back кластер) – запущен на одном узле кластера;
5. В случае если статус ресурсов кластера не соответствует перечисленным выше, выполните следующую команду, заменив *resource_name* на имя проблемного ресурса:

```
pcs resource cleanup resource_name
```

6. На узлах кластера Back с индексом 01 и 02 выполните команду ниже:

```
ls -la /opt/ex_data/files
```

Примечание. В случае зависания вывода списка директорий перезагрузите оба узла кластера через систему виртуализации.

7. Подключитесь к консоли сервера Back и Front кластера с индексом 01 либо 02 и выполните команду для определения текущего первичного узла, на котором запущены ресурсы кластера:

```
pcs status | grep dockerd
```

8. Подключитесь к консоли текущего первичного узла кластера Back и Front и последовательно выполните команду:

```
docker ps -a > current_version.txt
```

9. Сохраните полученный файл. Он потребуется для процедуры отката на предыдущую версию.
10. Запустите скрипт автоматического обновления всех контейнеров первичных серверов и введите пароль учетной записи root после ввода команды:

```
ansible-playbook --ask-pass -v 05-master_update_cts.yaml
```

11. После обновления первичных серверов проверьте функционирование системы, выполните проверку логов на наличие ошибок командой:

```
dpl --dc logs --tail=200 -f
```

12. Запустите скрипт автоматического обновления всех контейнеров вторичных серверов:

```
ansible-playbook --ask-pass -v 06-slave_update_cts.yaml
```

13. Введите пароль учетной записи root.
14. Подключитесь к консоли сервера Back и Front кластера с индексом 01 и 02, выполните команду для запуска антивируса:

```
systemctl start kesl klnagent64
```

Для отката на предыдущую версию ПО, установленного в контейнерах docker:

1. Подключитесь к консоли узлов кластера Back с индексами 01 и 02.
2. Добавьте в файл /opt/express/settings следующие строки, заменив в них версию ПО на значения из файла current_version.txt (файл получен на шаге 3 описания автоматического обновления с помощью скриптов ansible):

```
images:
  messaging: messaging:1.39.6
  settings: settings:1.39.0
  audit: audit:1.39.0
  admin: admin:1.39.1
  file_service: file_service:1.39.0
  voex: voex:1.39.0
  ad_phonebook: ad_phonebook:1.39.1
  email_notifications: email_notifications:1.39.0
  botx: botx:1.39.1
  ad_integration: ad_integration:1.39.0
  kdc: kdc:1.39.0
  routing_schema_service: routing_schema_service:1.39.0
```

3. Подключитесь к консоли узлов кластера Front с индексами 01 и 02.
4. Добавьте в файл /opt/express/settings следующие строки, заменив в них версию ПО на значения из файла current_version.txt (получен на шаге 3 описания автоматического обновления с помощью скриптов ansible):

```
images:
  trusts: trusts:1.39.0
```

5. Запустите скрипт автоматического обновления всех контейнеров первичных серверов и введите пароль учетной записи root после ввода команды:

```
ansible-playbook --ask-pass -v 05-master_update_cts.yaml
```

6. После обновления первичных серверов проверьте нормальное функционирование системы, выполните проверку логов на наличие ошибок командой:

```
dpl --dc logs --tail=200 -f
```

7. Запустите скрипт автоматического обновления всех контейнеров вторичных серверов:

```
ansible-playbook --ask-pass -v 06-slave_update_cts.yaml
```

8. Введите пароль учетной записи root после ввода команды.

Для обновления ПО, установленного в контейнерах docker Web client кластера, на сервере Registry выполните:

Важно! При выполнении скриптов обновления нельзя пропускать паузы, заложенные в него, т.к. их пропуск может привести к ошибкам обновления.

1. Скопируйте новые версии образов docker и скрипт загрузки образов в каталог /tmp/images сервера Registry.
2. Загрузите образы с помощью скрипта load.sh:

```
cp *.tar /tmp/images/
cp /opt/deploy/script/load.sh /tmp/images/
cd /tmp/images/
./load.sh
```

3. С помощью команды ниже уточните новую версию образа контейнера web client:

```
docker images | grep web_client
```

4. Измените параметр web_client_image на актуальную версию, полученную на предыдущем шаге (параметр локализован в файле настроек group_vars/all.yaml в каталоге сценариев ANSIBLE web client (/opt/deploy/playbook-webclient)).
5. Запустите скрипт автоматического обновления всех контейнеров первичного узла кластера Web Client:

```
ansible-playbook --ask-pass -v 05-master_update_web.yaml
```

6. Введите пароль учетной записи root.
7. После обновления первичного узла кластера Web Client проверьте нормальное функционирование системы, выполнив проверку логов и функции отправки сообщений.
8. Запустите скрипт автоматического обновления всех контейнеров вторичного узла кластера Web Client аналогично пп. 5-6.

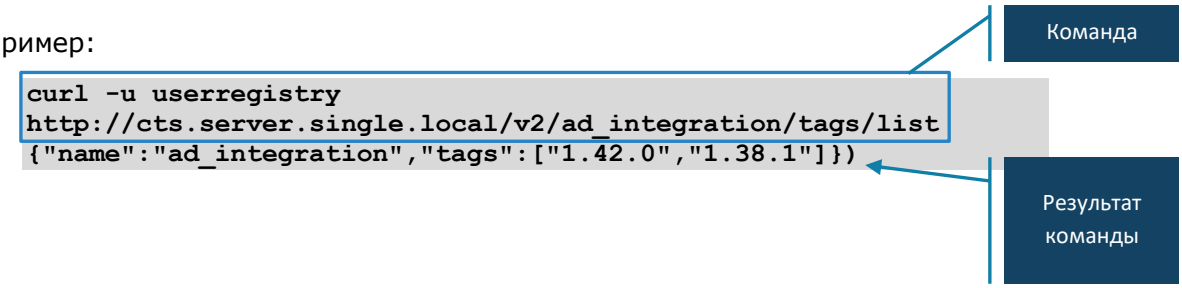
АВАРИЙНЫЕ СИТУАЦИИ ПРИ ОБНОВЛЕНИИ ИЗ ЛОКАЛЬНОГО РЕПОЗИТОРИЯ REGISTRY

Аварийные ситуации, перечисленные ниже, могут произойти в том случае, если имеется локально развернутый сервер Registry.

Ситуация 1. Отсутствия доступа к сети интернет с узла с репозиторием.

1. С узла, имеющего доступ в интернет, скачайте актуальные контейнеры с помощью скрипта по ссылке (вложение download.sh).
2. Запустите второй скрипт по ссылке (вложение upload.sh) и дождитесь окончания загрузки.
3. Сделайте тестовый запрос из консоли при помощи обращения к URL и получите версии, находящиеся в репозитории.

(Пример:



Ситуация 2. Если в п.3 предыдущей операции результат команды – "no basic auth credentials".

1. Удалите файл .docker/config.json.
2. Пройдите повторную авторизацию в Docker registry.

ПРОЦЕДУРА ОБНОВЛЕНИЯ СЕРТИФИКАТА

Для работы изделия требуется оформить сертификат на внешнее имя сервиса Express (FQDN или wildcard), выпущенный публичным доверенным центром сертификации и удовлетворяющий следующим требованиям:

- версия 3 и не ниже TLS 1.2;
- длина ключа не меньше 2048 бит;
- алгоритм подписи SHA 256;
- версия синтаксиса X.509 3;
- незашифрованный закрытый ключ.

Файл должен содержать в себе сертификат сервера, сертификаты промежуточного центра сертификации и корневого центра сертификации. Формат сертификатов должен соответствовать кодировке Base64. Файл закрытого ключа должен содержать нешифрованный закрытый ключ кодировки Base64.

Примерная структура файла сертификата изображена на рисунке ниже (Рисунок 1).

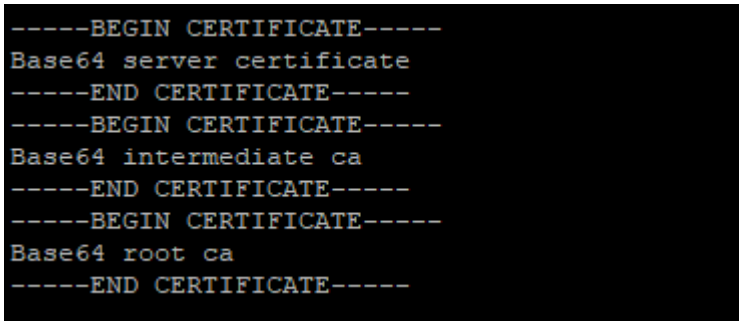


Рисунок 1

Поддерживается использование бесплатного сертификата от компании Let`s Encrypt.

Для обновления сертификата:

1. Подготовьте сертификат согласно требованиям выше.
2. Обновите файлы сертификатов, расположенные в папке /opt/express/certs/. Для разделенного сервера файлы обновляются на Front CTS.
3. Выполните команду в консоли:
   - если версия сервера ниже 3.5:

```
cd /opt/express && dpl -d && dpl --dc restart nginx
```

   - если версия сервера 3.5 и выше:

```
cd /opt/express && dpl -p && dpl --dc restart traefik
```

ОБНОВЛЕНИЕ KAFKA

Для настройки Kafka перед обновлением сервера на версию 2.6:

1. Ограничьте подключения к серверу снаружи.

   Добавьте правила в IPTABLES для цепочки DOCKER-USER, выполнив следующие команды:

```
sudo iptables -A DOCKER-USER  -p tcp --dport 80 -j DROP
sudo iptables -A DOCKER-USER  -p tcp --dport 443 -j DROP
sudo iptables -A DOCKER-USER  -p tcp --dport 5001 -j DROP
```

   Если цепочки DOCKER-USER нет, то уберите параметр «'bridge': none» из файла /etc/docker/daemon.json и перезапустить службу docker.

2. Выполните проверку Kafka Lag и дождитесь ответа всех offset:

```
docker exec -t cts_kafka_1 /opt/kafka/bin/kafka-consumer-
groups.sh --describe --all-groups --bootstrap-server
localhost:9092
```

   В выводе будут присутствовать consumer-groups:

```
trusts
audit
file_service
ad_integration
conference_bot
admin
ad_phonebook
events
botx
messaging
```

   Пример вывода по consumer-group admin:

```
GROUP  TOPIC           PARTITION  CURRENT-OFFSET  LOG-END-OFFSET
LAG  CONSUMER-ID
HOST          CLIENT-ID
admin  retry-30m       0          -               0
-    'admin@172.21.0.27'/<0.3836.0>-a0399c9b-a8a7-46c3-a7ab-
85fb1dcd  /172.21.0.27  'admin@172.21.0.27'/<0.3836.0>
admin  retry-5m        0          -               0
-    'admin@172.21.0.27'/<0.3815.0>-eb08d50c-4207-4c7b-870f-
0080d344  /172.21.0.27  'admin@172.21.0.27'/<0.3815.0>
admin  system-events   0          18              18
0    'admin@172.21.0.27'/<0.3880.0>-58d2f6fe-840b-412e-ae41-
39f999ba  /172.21.0.27  'admin@172.21.0.27'/<0.3880.0>
admin  retry-15m       0          -               0
-    'admin@172.21.0.27'/<0.3807.0>-0326ec57-bb39-4636-8639-
eb942ee5  /172.21.0.27  'admin@172.21.0.27'/<0.3807.0>
```

   Параметр LAG должен = 0.

   Если LAG ≠ 0, подождите 10 минут. Если значение параметра не меняется на 0, перезапустите сервис с LAG.

3. Удалите Kafka volume.

   Внимание! Предварительно создайте резервную копию /var/lib/docker/volumes/cts_kafka_logs.

Выполните:

- остановку и удаление контейнера Kafka:

```
docker stop cts_kafka_1 && docker rm $_
```

- удаление volume:

```
docker volume rm cts_kafka_logs
```

4. Обновите версии и запустите сервисы.

   Из директории СК «Express» запустите сервисы:

```
dpl -d
```

5. Выполните проверку контейнера Kafka:

```
docker ps -a | grep kafka
```

   Вывод:

```
CONTAINER ID   IMAGE                                COMMAND
CREATED       STATUS          NAMES
70c2fd5b3922   registry_name/kafka:2.13-3.3.1   "start-kafka"
4 days ago   Up 3 minutes   cts_kafka_1
```

   Просмотр логов Kafka:

```
docker logs cts_kafka_1
```

6. Удалите правила IPTABLES:

```
sudo iptables -D DOCKER-USER -p tcp -m tcp --dport 80 -j DROP
sudo iptables -D DOCKER-USER -p tcp -m tcp --dport 443 -j DROP
sudo iptables -D DOCKER-USER -p tcp -m tcp --dport 5001 -j DROP
```

Приложение 1

МИГРАЦИЯ MESSAGING ВЕРСИИ < 3.10 НА 3.10+

Внимание! Процедура миграции обязательна для обновления с версии ниже 3.10. После миграции на 3.10+ откатывать messaging на предыдущие версии не рекомендуется! Возвращение на прошлые версии приведет к сложной процедуре ручного удаления новых данных и повторения миграции. В случае возникновения проблем при миграции, необходимо обратиться в службу технической поддержки для получения патча.

Примечание. Время простоя в момент миграции может достигать от 5 до 30 минут в зависимости от объема базы.

ОПРЕДЕЛЕНИЕ РАЗМЕРА ТАБЛИЦ

Для определения размера таблиц:

1. Войдите в консоль СУБД PostgreSQL и подключитесь к базе данных **messaging_prod**:
   - если база данных реализована в контейнере Docker выполните команды:

```
docker exec -ti cts_postgres_1 psql -U postgres
\c messaging_prod
```

   - если база данных внешняя (без кластера) выполните команды:

```
su postgres
psql
\c messaging_prod
```

   - если база данных внешняя (кластер на базе Patroni) выполните команды:

```
psql -h внутренний_ип_ноды -p 5432 -U express -d postgres
\c messaging_prod
```

2. Выполните запрос размера таблиц базы данных:

```
select schemaname as table_schema,
    relname as table_name,
    pg_size_pretty(pg_total_relation_size(relid)) as total_size,
    pg_size_pretty(pg_relation_size(relid)) as data_size,
    pg_size_pretty(pg_total_relation_size(relid) -
pg_relation_size(relid))
      as external_size
from pg_catalog.pg_statio_user_tables
where relname = 'user_chat_events' or relname = 'chat_events'
order by pg_total_relation_size(relid) desc,
         pg_relation_size(relid) desc;
```

Внимание! Если total_size user_chat_events и/или chat_events превышает 20 Гб, то время простоя при миграции в автоматическом режиме может составлять > 1 часа. В этом случае рекомендуется провести миграцию по специальной инструкции, из раздела «Миграция больших баз данных». Для удобства рекомендуем заранее подготовить файл со всеми таймстемпами и запросами.

Примечание. Если таблицы в вашей базе данных меньше 20 Гб, вы можете мигрировать данные в автоматическом режиме (дополнительных действий не требуется).

Некоторые запросы, выполняющиеся при подготовке к миграции, могут обрабатываться до десяти часов в зависимости от объемов базы данных. Например, миграция старой истории сообщений. В этом случае рекомендуется использовать специальные утилиты для выполнения долгих задач **screen** или **nohup**.

Пример использования:

1. Добавьте запрос в текстовый файл с расширением .sql, например update.sql
2. Добавьте права на чтение файла пользователю от имени которого запускается psql.

Для утилиты screen (рекомендуется):

1. В терминале на сервере с базой данных выполните команду screen, откроется сессия терминала.
2. Выполните следующую команду (замените значение флага -u и -f на свой):

```
sudo -u postgres psql -d messaging_prod -f
/home/postgres/update.sql
```

3. Выйдите из сессии нажав сочетания клавиш **Ctrl-A** + **Ctrl-D**. Ни в коем случае не нажимайте **Ctrl-C** и **Ctrl-D** пока находитесь в screen сессии и запрос не завершился.
4. Нажмите screen -r чтобы вернуться к screen сессии и убедиться, что сессия работает и запрос выполняется. После чего можно выйти из сессии нажав **Ctrl-A + Ctrl-D.**
5. Повторите процедуру из п.4, чтобы проверять состояние задачи.

Для утилиты nohup (если знакомы с утилитой или нету возможности использовать screen):

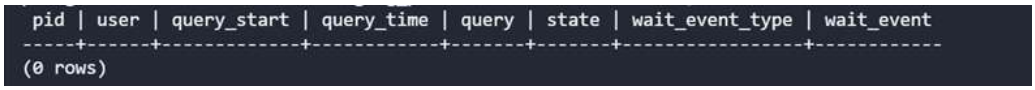
1. В терминале на сервере с базой данных выполните следующую команду (команда не должна требовать никакого input, например пароля от root юзера):

```
nohup sudo -u postgres psql -d messaging_prod -f
/home/postgres/update.sql &
```

2. Проверьте статус задачи через команду jobs -l
3. Проверьте вывод в файле honup.out
4. Проверьте статус выполнения запроса (после минуты выполнения):

```
SELECT
  pid,
  user,
  pg_stat_activity.query_start,
  now() - pg_stat_activity.query_start AS query_time,
  query,
  state,
  wait_event_type,
  wait_event
FROM pg_stat_activity
WHERE (now() - pg_stat_activity.query_start) > interval '1
minutes'
AND datname = 'messaging_prod' AND state = 'active';
```

5. В результате отображается запрос. Screen сессия или nohup процесс должен быть активным.



ПОДГОТОВКА К МИГРАЦИИ

Внимание! Перед началом операции убедитесь в достаточном количестве свободного места. Рекомендуется, чтобы объем свободного пространства в 2-3 раза превышал объем базы данных.

Примечание. Данные запросы выполняются при запущенных сервисах. В случае если база данных очень большая, подготовку можно начинать за несколько дней до миграции и выполнять этапы в технические окна.

Все запросы выполняются в psql, база данных messaging_prod.

1. Добавьте новые колонки в таблицу chat_events:

```
ALTER TABLE chat_events
ADD COLUMN events_history_scope BOOLEAN,
ADD COLUMN event_info_scope BOOLEAN;
```

2. Запишите текущий timestamp. Этот timestamp необходимо сохранить до момента будущей миграции:

```
messaging_test=# SELECT now();
              now
-------------------------------
 2023-12-25 11:06:06.037283+03
(1 row)
```

3. Запишите timestamp самого раннего сообщения:

```
messaging_test=# SELECT inserted_at FROM chat_events ORDER BY
inserted_at ASC LIMIT 1;
        inserted_at
----------------------------
 2018-02-05 10:21:44.191786
(1 row)
```

4. Выполните миграцию историй сообщений от даты, полученной из таймстемпа в п. 2 + 1 минута до даты самого раннего события п. 3 с шагом в 1 год. Каждый запрос будет выполняться длительное время, если по каким-то причинам запрос выполнить не получается, можно уменьшить шаг до полугода/трех месяцев.

```
UPDATE chat_events ce
SET events_history_scope = uce.events_history_scope,
event_info_scope = uce.event_info_scope
FROM user_chat_events uce
WHERE ce.sync_id = uce.event_sync_id
AND ce.event_type != 'routing_changed'
AND ce.inserted_at >= '2023-01-01' AND ce.inserted_at <= '2023-
12-25 11:07';

UPDATE chat_events ce
SET events_history_scope = uce.events_history_scope,
event_info_scope = uce.event_info_scope
FROM user_chat_events uce
WHERE ce.sync_id = uce.event_sync_id
AND ce.event_type != 'routing_changed'
```

```
AND ce.inserted_at >= '2022-01-01' AND ce.inserted_at <= '2023-
01-01';

UPDATE chat_events ce
SET events_history_scope = uce.events_history_scope,
event_info_scope = uce.event_info_scope
FROM user_chat_events uce
WHERE ce.sync_id = uce.event_sync_id
AND ce.event_type != 'routing_changed'
AND ce.inserted_at >= '2021-01-01' AND ce.inserted_at <= '2022-
01-01';

...

UPDATE chat_events ce
SET events_history_scope = uce.events_history_scope,
event_info_scope = uce.event_info_scope
FROM user_chat_events uce
WHERE ce.sync_id = uce.event_sync_id
AND ce.event_type != 'routing_changed'
AND ce.inserted_at >= '2018-02-05' AND ce.inserted_at <= '2019-
01-01';
```

5. Если total_size ваших таблиц больше 200 Гб, то быстрее будет выполнить один долгий общий запрос без временных шагов:

```
--- запрос для миграции без временного шага,
--- выполнять только в случае очень объемных баз,
--- обязательно с использованием nohup или screen,
--- время выполнения может достигать > 10 часов
UPDATE chat_events ce
SET events_history_scope = uce.events_history_scope,
event_info_scope = uce.event_info_scope
FROM user_chat_events uce
WHERE ce.sync_id = uce.event_sync_id
AND ce.event_type != 'routing_changed';
```

6. Создайте новые индексы для таблицы chat_events:

```
CREATE INDEX CONCURRENTLY
chat_events_group_chat_id_inserted_at_stealth_index ON
chat_events USING btree (group_chat_id, inserted_at) INCLUDE
(sync_id, stealth) WHERE (NOT (stealth IS NULL));

CREATE INDEX CONCURRENTLY
chat_events_group_chat_id_inserted_at_shared_index ON
chat_events USING btree (group_chat_id, inserted_at) WHERE
((events_history_scope = true) AND (shared = true));

CREATE INDEX CONCURRENTLY
chat_events_group_chat_id_inserted_at_non_shared_index ON
chat_events USING btree (group_chat_id, inserted_at) WHERE
((events_history_scope = true) AND (shared = false));

CREATE INDEX CONCURRENTLY
chat_events_sync_id_event_info_scope_index ON chat_events USING
btree (sync_id) WHERE (event_info_scope = true);

CREATE INDEX CONCURRENTLY chat_events_epoch_migration_index_1 ON
chat_events (event_type) WHERE event_type='added_to_chat' OR
event_type='user_joined_to_chat';
```

```
CREATE INDEX CONCURRENTLY chat_events_epoch_migration_index_2 ON
chat_events (event_type) WHERE event_type='left_from_chat' OR
event_type='deleted_from_chat' OR
event_type='kicked_by_cts_logout';
```

7. Отслеживание прогресса создания индексов

```
--- Отслеживание прогресса создания индексов
SELECT phase, blocks_total, blocks_done, (blocks_total -
blocks_done) as blocks_rest, tuples_total, tuples_done,
(tuples_total - tuples_done) as tuples_rest FROM
pg_stat_progress_create_index;
```

МИГРАЦИЯ

Внимание! Если этап подготовки к миграции был выполнен несколько дней назад, то необходимо мигрировать историю от текущего времени до времени начала этапа подготовки к миграции. Если вы выполняете этап подготовки к миграции и основную миграции в один день, то пропустите п. 1 и п. 2.

1. Запишите текущий timestamp:

```
messaging_test=# SELECT now();
              now
-------------------------------
 2023-12-27 12:14:01.026154+03
(1 row)
```

2. Обновите историю от времени полученного в разделе «Подготовка к миграции» п. 2

```
UPDATE chat_events ce
SET events_history_scope = uce.events_history_scope,
event_info_scope = uce.event_info_scope
FROM user_chat_events uce
WHERE ce.sync_id = uce.event_sync_id
AND ce.event_type != 'routing_changed'
AND ce.inserted_at >= '2023-12-25 11:06:00';
```

3. Остановите все сервисы, например для Single CTS выполните команду:

```
dpl --dc stop
```

4. Зайдите в psql, подключитесь к базе данных messaging_prod.
5. С помощью следующей команды пропустите миграции, которые выполняли (текст команды вводится как указано, без изменений):

```
INSERT INTO
    schema_migrations (version, inserted_at)
VALUES
    (20231031201400, now()),
    (20231031201640, now()),
    (20231101121946, now()),
    (20231101134329, now()),
    (20231116000454, now()),
    (20231129210158, now()),
    (20231205114132, now()),
    (20231205115340, now()),
    (20231205123846, now()),
    (20231211072757, now());
```

6. Запишите текущий timestamp:

```
messaging_test=# SELECT now();
              now
-------------------------------
 2023-12-27 12:34:39.258821+03
(1 row)
```

7. Мигрируйте историю от **момента подготовки миграции**. Используйте дату и время с точностью до минуты:
   - из п. 1 если был выполнен процесс дополнительной миграции истории:

```
---- обновление с временем из пункта 1
UPDATE chat_events ce
SET events_history_scope = uce.events_history_scope,
event_info_scope = uce.event_info_scope
FROM user_chat_events uce
WHERE ce.sync_id = uce.event_sync_id
AND ce.event_type != 'routing_changed'
AND ce.inserted_at >= '2023-12-27 12:14:00';
```

   - из п. 2 раздела «Подготовка к миграции», если дополнительной миграции истории не было:

```
---- обновление с датой из раздела "Подготовка к миграции" пункт
2
UPDATE chat_events ce
SET events_history_scope = uce.events_history_scope,
event_info_scope = uce.event_info_scope
FROM user_chat_events uce
WHERE ce.sync_id = uce.event_sync_id
AND ce.event_type != 'routing_changed'
AND ce.inserted_at >= '2023-12-25 11:06:00';
```

8. Запустите сервисы, дождитесь завершения автоматических миграций и запуска сервиса messaging.

9. Мигрируйте следующие данные используя таймстемп полученный в п. 6:

```
INSERT INTO user_chat_events_v2 (user_huid, group_chat_id,
event_sync_id, inserted_at)
SELECT jsonb_array_elements_text((event_params-
>>'recipients')::jsonb)::uuid as user_huid, group_chat_id,
sync_id, ce.inserted_at
FROM chat_events ce
INNER JOIN group_chats gc ON gc.id = ce.group_chat_id
WHERE global = true AND
shared = false AND
chat_type = 'global' AND
ce.inserted_at <= '2023-12-27 12:34:39';

INSERT INTO user_chat_events_v2 (user_huid, group_chat_id,
event_sync_id, inserted_at)
SELECT jsonb_array_elements_text((event_params-
>>'recipients')::jsonb)::uuid as user_huid, group_chat_id,
sync_id, inserted_at
FROM chat_events ce
WHERE jsonb_array_length((event_params->>'recipients')::jsonb) >
0 AND
event_type = ANY('{message_new,call_end}') AND
```

```
ce.inserted_at <= '2023-12-27 12:34:39';

INSERT INTO user_chat_events_v2 (user_huid, group_chat_id,
event_sync_id, inserted_at)
SELECT jsonb_array_elements_text((event_params-
>>'recipients')::jsonb)::uuid as user_huid, group_chat_id,
sync_id, inserted_at
FROM chat_events ce
WHERE jsonb_array_length((event_params->>'recipients')::jsonb) =
1 AND
event_type = 'app_event' AND
payload->>'event_type' = 'bot_notification' AND
ce.inserted_at <= '2023-12-27 12:34:39';
```

10. Выполните дамп системных событий из каналов. Его можно будет удалить спустя время, когда релиз проверится временем:

```
CREATE TABLE channel_events_dump AS
    SELECT ce.* FROM chat_events ce INNER JOIN group_chats c ON
c.id = ce.group_chat_id WHERE chat_type = 'channel'
    AND event_type =
ANY('{added_to_chat,deleted_from_chat,left_from_chat,admin_added
_to_chat,kicked_by_cts_logout,user_joined_to_chat}');
```

11. Удалите лишние системные события из каналов:

```
DELETE FROM chat_events ce
USING group_chats c
WHERE c.id = ce.group_chat_id
AND chat_type = 'channel'
AND event_type =
ANY('{added_to_chat,deleted_from_chat,left_from_chat,admin_added
_to_chat,kicked_by_cts_logout,user_joined_to_chat}');
```

12. Если на вашем сервере использовался notifications_bot от BS (не внутренний бот) вам необходимо мигрировать его сообщения, для этого выполните следующий запрос и подставьте в качестве bot_id идентификатор notifications_bot используя таймстемп полученный в п. 6:

```
INSERT INTO user_chat_events_v2 (user_huid, group_chat_id,
event_sync_id, inserted_at)
SELECT jsonb_array_elements_text((event_params-
>>'recipients')::jsonb)::uuid as user_huid, group_chat_id,
sync_id, now()
FROM chat_events ce
WHERE jsonb_array_length((event_params->>'recipients')::jsonb) >
0 AND
event_type = 'app_event' AND
payload->>'event_type' = 'bot_notification' AND
sender = 'bot_id' AND
ce.inserted_at <= '2023-12-27 12:34:39';
```

13. Обновите статистику (см. раздел «Обновление статистики»).
14. Удалите миграционные индексы:

```
DROP INDEX chat_events_epoch_migration_index_1;
DROP INDEX chat_events_epoch_migration_index_2;
```

ОБНОВЛЕНИЕ СТАТИСТИКИ

После автоматической или ручной миграции необходимо обновить статистику. В psql выполните следующие запросы:

```
VACUUM ANALYZE user_chat_events_v2;
VACUUM ANALYZE chat_event_meta;
VACUUM ANALYZE chat_member_epochs;
VACUUM ANALYZE chat_events;
```

ИСТОРИЯ ИЗМЕНЕНИЙ

Раздел «История изменений» содержит список изменений в документе, связанных с изменениями/доработками СК «Express».

Сборка 2.8.0

№	Раздел	Изменение	Сервер	Ссылка
1.	Обновление без доступа к apt.postgresql.org	Актуализирован порядок обновления	CTS	стр. 13

Сборка 3.0

№	Раздел	Изменение	Сервер	Ссылка
1.	Обновление Kafka	Обновлен п.2		стр. 20
2.	Обновление ОС	Добавлен раздел		стр. 3
3.	Обновление Deployka	По всему документу исправлена операция DEPLOYKA_SKIP_UPDATE=true на DPL_PULL_POLICY=never		
4.	Обновление до Express 3.0	Добавлен подраздел		стр. 7
5.	Процедура обновления сертификата	Изменение директории хранения сертификата /opt/express/certs		стр. 19
6.	Ручное обновление сервера	Замена En-Dash на Hyphen-minus в примерах с командной строкой		

Сборка 3.5

№	Раздел	Изменение	Сервер	Ссылка
1.	Процедура обновления сертификата	Изменение команды при обновлении сертификата для сервера версии 3.5 и выше		стр. 18

Сборка 3.10

№	Раздел	Изменение	Сервер	Ссылка
1.	Миграция messaging на 3.10+	Добавлен раздел		стр. 22