Инструкция по развертыванию

СберРост Про

Оглавление

[Установка docker и docker-compose 3](#_Toc64288557)

[Развертывание ИСУ 3](#_Toc64288558)

[Установка 3](#_Toc64288559)

[2. Установка isu\_gateway 4](#_Toc64288560)

[1.1. Настройка БД 4](#_Toc64288561)

[1.2. Установка приложения isu\_gateway 5](#_Toc64288562)

[6.1. Добавление клиентских сертификатов в isu\_gateway 6](#_Toc64288563)

[6.2. Настройка SSL-соединения приложения isu\_gateway 6](#_Toc64288564)

[6.3. Настройка keystore приложения isu\_gateway 8](#_Toc64288565)

[2.1. Добавление серверный сертификата FrontApp 8](#_Toc64288566)

[2. Установка isu\_app 8](#_Toc64288567)

[2.1. Настройка БД 8](#_Toc64288568)

[2.2. Настройка серверного сертификата Ingress Proxy 9](#_Toc64288569)

[1.3. Установка приложения isu\_app 9](#_Toc64288570)

[1. Настройка сервера PostgreSQL 11](#_Toc64288571)

[Развертывание Toolkit 13](#_Toc64288572)

[1. Установка docker-images и запуск nginx (gateway) 13](#_Toc64288573)

[1.1. Установка и сборка начальных образов 13](#_Toc64288574)

[1.2. Сборка конечных образов 14](#_Toc64288575)

[1.3. Выпуск серверного сертификата для запуска gateway 14](#_Toc64288576)

[1.4. Запуск gateway 14](#_Toc64288577)

[1.5. Установка библиотеки 15](#_Toc64288578)

[2. Создание нового клиента 15](#_Toc64288579)

[2.1. Установка конвейера и репозитория развертки клиента 15](#_Toc64288580)

[2.2. Установка БД для библиотеки и конвейера 15](#_Toc64288581)

[2.3. Создание сертификатов для взаимодействия между Gateway и Ingress Proxy 16](#_Toc64288582)

[2.4. Редактирование конфигурационных файлов 16](#_Toc64288583)

[2.5. Настройка Gateway 17](#_Toc64288584)

[2.6. Запуск контейнеров 17](#_Toc64288585)

[2.7. Создание супер-пользователя для библиотеки 17](#_Toc64288586)

[2.8. Перезапуск gateway 18](#_Toc64288587)

[Получение сертификатов TLS 18](#_Toc64288588)

[1. Генерация приватного ключа 18](#_Toc64288589)

[2. Генерация файла-запроса на получение сертификата 19](#_Toc64288590)

[3. Отправка запроса в удостоверяющий центр 20](#_Toc64288591)

[Приложение 20](#_Toc64288592)

[1. Создание приватного и публичного rsa ключей 20](#_Toc64288593)

[2. Список сертификатов TLS 21](#_Toc64288594)

# Установка docker и docker-compose

1. Запустить установку docker:

**sudo yum install docker**

1. Запускаем docker:

**sudo systemctl start docker**

1. Запустить обновление git:

**sudo yum install git-all**

1. Загружаем библиотеки Python3-pip:

**sudo yum install python3-pip**

1. Запустить установку docker-compose:

**sudo pip3 install --index-url http://mirror.sigma.sbrf.ru/pypi/simple --trusted-host mirror.sigma.sbrf.ru docker-compose**

1. Создать ссылку на скрипты docker-compose

**sudo ln -s /usr/local/bin/docker-compose /usr/bin/docker-compose**

1. Для установки Openssl11 поменять в файле **/etc/yum.repos.d/mirror.repo** параметр "**enabled**" с «0» на «**1**» в поле [**EPEL7**]
2. Установить Openssl11:

**sudo yum install openssl11**

# Развертывание ИСУ

# Установка



Рисунок 1

Клиентская часть СберРост ПРО представлена следующими подсистемами:

 Isu\_gateway;

 Isu\_app;

 Isu\_gateway\_db.

**Isu\_gateway** – фильтрация запросов пользователей и перенаправление на Back services клиента, к которому относится пользователь. Общий для всех клиентов.

Предустановленное ПО:

1. docker
2. docker-compose
3. В docker загружены образы:
* node:12.20.0-alpine
* nginx:1.18.0-alpine
* openjdk:11.0

**Isu\_app** – бизнес функции приложения. Для каждого клиента создается свой независимый экземпляр.

Предустановленное ПО:

1. docker и docker-compose
2. В docker загружены образы:
* node:12.20.0-alpine
* nginx:1.18.0-alpine
* openjdk:11.0

**Isu\_gateway\_db** – Instance БД или кластер, в зависимости от конфигурации, используемый для размещения независимых баз данных под каждого клиента.

Предустановленное ПО:

1. PostgreSQL DB версии 11.9

# Установка isu\_gateway

Isu\_gateway разворачивается единожды на своих отдельных серверах (gateway сервер и сервер gateway базы данных) (см. Рисунок 1).

## Настройка БД

Авторизоваться в БД под пользователем с административными привилегиями;

Создать пользователя isu\_gateway, указав пароль соответственно требованиям безопасности к паролям;

**create role isu\_gateway with noinherit login encrypted password 'Password';**

 Создать базу данных с доступом для пользователя из п.2;

**create database isu\_gateway with owner isu\_gateway encoding 'UTF-8';**

 Установить временную зону размещения системы, по умолчанию – Москва. При необходимости заменить на требуемую;

**alter database isu\_gateway set timezone to 'Europe/Moscow';**

Подключиться к созданной базе данных и создать схему, указав права на неё для пользователя из п.2;

**create schema isu\_gateway authorization isu\_gateway;**

## Установка приложения isu\_gateway

isu\_gateway разворачивается единожды на отдельном сервере (gateway сервер и сервер gateway базы данных из схемы 1).

1. Скачать архив **sber-growth-gateway.tar.gz** который содержит нужные исходники и базовую структуру приложения.
2. Разархивировать **sber-growth-gateway.tar.gz** в любом месте. Архив имеет следующую структуру:

**/sber\_growth\_gateway**

**build\_gateway\_images.sh**

**/src**

**/external\_ingress**

**/gateway**

**/isu\_gateway**

**docker-compose-gateway.yml**

**/env**

**database.env**

**/logs**

1. Собрать образы **external\_ingress** и **gateway** запустив:

**./sber\_growth\_gateway/build\_gateway\_images.sh**

1. Переместить директорию **./sber\_growth\_gateway/isu\_gateway** в любое место, где будет находиться приложение.
2. Установить права на доступ к папке **./gateway/env** только пользователю под которым будет работать приложение и администраторам

Отредактировать файл **./gateway/env/database.env** с данными для подключения к базе данных.

Пример database.env: Желтым указаны комментарии

**DATABASE\_HOST=12.12.123.12** ip адрес сервера с базой данных

**DATABASE\_PORT=5432** порт базы данных

**DATABASE\_NAME=isu\_gateway** имя базы данных из пункта п.1.1. п.3

**DATABASE\_OPTIONS='ssl=true&sslfactory=org.postgresql.ssl.LibPQFactory&sslmode=require'**

**DATABASE\_USER=isu\_gateway** имя пользователя из пункта п.1.1. п.2

1. Выпустить клиентский сертификат для сервера БД (Сертификат 5.1 см. Получение сертификатов TLS) в Сommon Name указать значение, которое содержится в файле **database.env** в поле **DATABASE\_USER**.
	1. Преобразовать полученный сертификат в PEM формат:

**openssl x509 -inform der -in [Имя файла сертификата].cer -out postgresql.crt**

* 1. Поместить преобразованный сертификат в **./gateway/cert/db/**
	2. Приватный ключ на основании которого был сгенерирован файл запроса на сертификат, необходимо поместить в **./gateway/cert/db/postgresql.key**
	3. Преобразовать приватный ключ **./gateway/cert/db/postgresql.key** в формат **pk8** командой:

**openssl pkcs8 -topk8 -inform PEM -outform DER -in postgresql.key -out postgresql.pk8 –nocrypt**

* 1. Поместить корневой и промежуточный сертификаты (формат **PEM**) в файл **./gateway/cert/db/root.crt**
1. Перейти в папку **./gateway** и выполнить команду:

**docker-compose -f docker-compose-gateway.yml up -d**

## Добавление клиентских сертификатов в isu\_gateway

1. Для нового клиента необходимо запросить **свой** клиентский сертификат Ingress Proxy (Сертификат 2 см. Получение сертификатов TLS) в Сommon Name указать значение Host-name сервера Ingress Proxy.
	1. Преобразовать полученный сертификат в PEM формат:

**openssl x509 -inform der -in [Имя файла сертификата].cer -out [Host-name сервера Ingress Proxy].pem**

* 1. Поместить преобразованный сертификат в **./gateway/cert/client/**

## Настройка SSL-соединения приложения isu\_gateway

1. Создание закрытого ключа:
	1. В командной строке выполните команду:

**openssl genrsa -des3 -out private.key 2048**

* 1. В ответ на запрос «Enter pass phrase for private.key» введите пароль для защиты закрытого ключа.
	2. После запроса «Verifying - Enter pass phrase for private.key» - повторите ввод пароля.
	3. Ваш закрытый ключ будет создан и сохранен в файле **private.key**.
1. Создание CSR:
	1. В командной строке выполните команду:

**openssl req -new -key private.key -out [Доменное имя сервера].csr**

* 1. Введите пароль закрытого ключа в ответ на запрос «Enter pass phrase for private.key».
	2. Следующие поля заполняются латинскими символами:
* Country Name - двухсимвольный код страны. Должно быть «RU»;
* State or Province Name: название области или региона без сокращений. Не заполняется;
* Locality Name: название города или населенного пункта (необязательное поле). "Moscow";
* Organization Name: название организации в латинском эквиваленте;
* Organizational Unit Name: название подразделения (необязательное поле)
* Common Name: полностью определенное (FQDN) доменное имя сервера.
* Email Address: контактный e-mail адрес (необязательное поле);
* A challenge password: не заполняется;
* An optional company name: альтернативное имя компании (не заполняется).
	1. Запрос на сертификат будет сохранен в файле **[Доменное имя сервера]**.csr в виде закодированного текста.
1. Оправляем созданный запрос на подпись в удостоверяющий центр.
2. Получаем подписанные сертификаты и сертификат удостоверяющего центра сертификации (CA Signer certificate) в формате DER encoded X.509 (.CER).
3. конвертируем сертификат в PEM формат:

**openssl x509 -in [Имя сертификата].cer -inform DER -out [Имя сертификата].cer -outform PEM**

1. Формируем pem**:**

**cat private.key [Имя сертификата].cer > keyсert.pem**

1. Формируем pkcs12**:**

**openssl pkcs12 -export -in keycert.pem -out keystore.pkcs12 -name [Имя сертификата] -noiter -nomaciter**

1. Проверяем что в хранилище:

**keytool -list -keystore ./keystore.pkcs12 -storetype pkcs12 -storepass [Пароль к приватному ключу]**

1. Конвертируем в jks**:**

**keytool -importkeystore -srckeystore ./keystore.pkcs12 -destkeystore ./keystore.jks -srcstoretype pkcs12 -srcstorepass [Пароль к приватному ключу] -deststorepass [Пароль к приватному ключу] пароль одинаковый, иначе пароль ключа будет отличаться от пароля хранилища, и может сбоить**

1. Импорт сертификатов (промежуточный/корневой):

**keytool -importcert -keystore ./keystore.jks -storepass [Пароль к приватному ключу] -alias**[**root.ca**](http://test.sbrf.root.ca/)**-file ./Test\ Root\ CA.cer -v**

1. Импорт сертификатов (промежуточный/корневой):

**keytool -importcert -keystore ./keystore.jks -storepass [Пароль к приватному ключу] -alias**[**issuer.ca**](http://test.sbrf.issuer.ca/)**-file ./Test\ Issuing\ CA.cer –v**

1. Проверяем что в хранилище:

**keytool -list -keystore ./keystore.jks -storepass [Пароль к приватному ключу] Должно быть 3 объекта (два сертификата удостоверяющих и наш с приватным ключом)**

1. Поместить созданный **keystore.jks в папку keystore**

## Настройка keystore приложения isu\_gateway

1. Создать файл с настройками **keystore.env** в папке **./gateway/env**
2. Добавить созданный файл настройки в:

**KEY\_STORE='/usr/app/keystore/keystore.jks'**

**KEY\_STORE\_PASSWORD=[Пароль к приватному ключу]**

**KEY\_PASSWORD=Qq123456=[Пароль к приватному ключу]**

## Добавление серверный сертификата FrontApp

1. Необходимо выпустить серверный сертификат FrontApp (Сертификат 3 см. Получение сертификатов TLS) в Сommon Name указать значение Host-name сервера FrontApp.
	1. Преобразовать полученный сертификат в PEM формат:

**openssl x509 -inform der -in [Имя файла сертификата].cer -out server.pem**

* 1. Поместить преобразованный сертификат и приватный ключ в **./gateway/cert/server/** (Папка должна содержать файлы **server.pem** и **private.key**)

# Установка isu\_app

Для каждого клиента разворачивается отдельный экземпляр isu\_app на общем клиентском сервере isu\_app и сервере клиентских баз данных (см. Рисунок 1).

## Настройка БД

Авторизоваться в БД под пользователем с административными привилегиями;

Создать пользователя isu\_app\_[**Имя клиента**] (например, isu\_app\_[**Имя клиента**]), пароль указать соответственно требованиям безопасности к паролям;

**create role isu\_app\_**[**Имя клиента**] **with noinherit login encrypted password 'Password';**

 Создать базу данных с таким же именем, как у пользователя из п.2;

**create database isu\_app\_**[**Имя клиента**] **with owner isu\_app\_**[**Имя клиента**] **encoding 'UTF-8';**

 Установить временную зону размещения системы, по умолчанию – Москва. При необходимости заменить на требуемую;

**alter database isu\_app\_**[**Имя клиента**] **set timezone to 'Europe/Moscow';**

Подключиться к созданной базе данных и создать схемы, указав права на них для пользователя из п.2;

**create schema isu\_auth authorization isu\_app\_**[**Имя клиента**]**;**

**create schema isu\_common authorization isu\_app\_**[**Имя клиента**]**;**

**create schema isu\_notifications authorization isu\_app\_**[**Имя клиента**]**;**

## Настройка серверного сертификата Ingress Proxy

1. Необходимо выпустить серверный сертификат Ingress Proxy (Сертификат 4 см. Получение сертификатов TLS) в Сommon Name указать значение Host сервера Ingress Proxy.
	1. Преобразовать полученный сертификат в PEM формат:

**openssl x509 -inform der -in [Имя файла сертификата].cer -out server.pem**

* 1. Поместить преобразованный сертификат и приватный ключ в **./client/cert/** (Папка должна содержать файлы **server.pem** и **private.key**)

## Установка приложения isu\_app

1. Скачать архив **sber-growth-app.tar.gz** который содержит нужные исходники и базовую структуру приложения.
2. Разархивировать **sber-growth-app.tar.gz** в любом месте. Архив имеет следующую структуру:

**/sber\_growth\_app**

**build\_app\_images.sh**

**/src**

**/internal\_ingress**

**/auth**

**/common**

**/notifications**

**/isu\_app**

**docker-compose-app.yml**

**/env**

**database.env**

**jwt.env**

**general.env**

**/logs**

1. Собрать образы **internal\_ingress**, **auth, common** и **notifications** запустив:

**./sber\_growth\_app/build\_app\_images.sh**

1. Переместить директорию **./sber\_growth\_app/isu\_app** в любое место, где будет находиться приложение.
2. Установить права на доступ к папке **./client/env** только пользователю под которым будет работать приложение и администраторам;
3. Отредактировать файл **./client/env/database.env** с данными для подключения к базе данных:

Пример database.env: Желтым указаны комментарии

**DATABASE\_HOST=12.12.123.12** ip адрес сервера с базой данных

**DATABASE\_PORT=5432** порт базы данных

**DATABASE\_NAME=isu\_app\_**[**Имя клиента**]имя базы данных из пункта п.2.1. п.3

**DATABASE\_OPTIONS='ssl=true&sslfactory=org.postgresql.ssl.LibPQFactory&sslmode=require'**

**DATABASE\_USER=isu\_app\_**[**Имя клиента**]имя пользователя из пункта п.2.1. п.2

1. Выпустить клиентский сертификат для сервера БД (Сертификат 5.2 см. Получение сертификатов TLS) в Сommon Name указать значение, которое содержится в файле **database.env** в поле **DATABASE\_USER**.
	1. Преобразовать полученный сертификат в PEM формат:

**openssl x509 -inform der -in [Имя файла сертификата].cer -out postgresql.crt**

* 1. Поместить преобразованный сертификат в **./client/cert/db/**
	2. Приватный ключ на основании которого был сгенерирован файл запроса на сертификат, необходимо поместить в **./client/cert/db/postgresql.key**
	3. Преобразовать приватный ключ **./client/cert/db/postgresql.key** в формат **pk8** командой:

**openssl pkcs8 -topk8 -inform PEM -outform DER -in postgresql.key -out postgresql.pk8 –nocrypt**

* 1. Поместить корневой и промежуточный сертификаты (формат **PEM**) в файл **./client/cert/db/root.crt**
1. Установить любой незанятый порт, на котором будет работать isu\_app (на одном сервере может работать несколько приложений isu\_app для разных клиентов). Для этого необходимо отредактировать файл ./client/docker-compose-client.yml. Место для правки отмечено желтым.

 **version: "3.8"**

 **services:**

 **internal\_ingress:**

 **image: internal\_ingress**

 **ports:**

 **- 9001:80**

 **depends\_on:**

 **- auth**

 **- common**

 **volumes:**

 **- type: bind**

 **source: ./logs**

 **target: /etc/nginx/logs**

 **auth:**

 **image: auth**

 **environment:**

 **…**

1. Перейти в базу данных приложения isu\_gateway. Вставить новую запись в таблицу isu\_gateway.domains



Рисунок 2

 Значения полей:

 **id** – идентификатор клиента;

 **name** – название клиента. Является уникальным;

 **host** – ip адрес сервера на котором устанавливается приложением isu\_app;

 **port** – порт которое слушает приложение isu\_app из п.5;

 **is\_active** – доступность приложения для пользователей.

1. Отредактировать файл **./client/env/jwt.env** с данными по rsa ключам для создания и проверки токенов.

Создание ключей описано в пункте III.1. Ключи должны быть указаны только в jwt.env, все временные файлы с приватными ключами должны быть удалены (например, private\_key.pem или private\_key\_pkcs8.pem). У каждого клиента должна быть своя пара приватного и публичного ключей.

Пример jwt.env: Желтым указаны комментарии

**JWT\_PRIVATE\_KEY=MIIJQwIBADANBgkqhkiG9w0BAQEFAASCCS0w…** приватный ключ в pkcs8 формате

**JWT\_PUBLIC\_KEY=MIIJQwIBADANBgkqhkiG9w0BjJFmdfjJDFJFJSEm…** публичный ключ

1. Отредактировать файл **./client/env/general.env**

Пример genera.env: Желтым указаны комментарии

 **GENERAL\_DOMAIN\_NAME=**[**Имя клиента**]имя клиента(name) в isu\_gateway.domains из п.6

1. Перейти в папку **./client** и выполнить команду

**docker-compose -f docker-compose-client.yml up -d**

# Настройка сервера PostgreSQL

1. Выпустить серверный сертификат для сервера БД (Сертификат 6 см. Получение сертификатов TLS) в Сommon Name указать адрес сервера, на котором развернута БД (Доменное имя).
	1. Преобразовать полученный сертификат в PEM формат:

**openssl x509 -inform der -in [Имя файла сертификата].cer -out server.pem**

* 1. Переименовать файл приватного ключа в **server.key**
	2. Преобразованный сертификат **server.pem** и приватный ключ (**server.key**) поместить **[Путь к PostgreSQL]/** (там же где находится **postgresql.conf**)
	3. Поместить корневой и промежуточный сертификаты в **[Путь к PostgreSQL]/root.pem**
1. Внести изменения в настройки SSL-соединения в файле **[Путь к PostgreSQL]/postgresql.conf** (см. Рисунок 3):



Рисунок 3

1. Внести изменения в файл **[Путь к PostgreSQL]/pg\_hba.conf** (см. Рисунок 4):



Рисунок 4

# Развертывание Toolkit

# Установка docker-images и запуск nginx (gateway)

## Установка и сборка начальных образов

* + 1. Локально зайти в терминал
		2. Скопировать репозиторий с docker-файлами:

**git clone ssh://git@10.21.25.60:8878/isu\_ivb/isu-toolkit-devops.git**

**cd isu-toolkit-devops/base\_images**

* + 1. Создать последовательно образы:

**cd nginx**

**sudo docker build -t nginx .**

**cd ../python\_for\_library**

**sudo docker build .**

**cd ../jupyterhub**

**sudo docker build -t jupyterhub\_base .**

**cd ../uwsgi**

**sudo docker build -t uwsgi\_base .**

* + 1. Создать на рабочем столе папку, в которую положим образы:

**mkdir ~/Desktop/base\_images**

* + 1. Сохраняем образы zip-формате:

**sudo docker save nginx | gzip >~/Desktop/base\_images/nginx.tar.gz**

**sudo docker save python | gzip >~/Desktop/base\_images/python.tar.gz**

**sudo docker save jupyterhub\_base | gzip >~/Desktop/base\_images/jupyterhub\_base.tar.gz**

**sudo docker save uwsgi\_base | gzip >~/Desktop/base\_images/uwsgi\_base.tar.gz**

* + 1. Скопировать папку с образами на сервер:

**scp -r ~/Desktop/base\_images [Имя пользователя]@[Имя сервера]:/home/[Имя пользователя]**

* + 1. Зайти на сервер

**ssh [Имя пользователя]@[Имя сервера]**

* + 1. Перейти в папку:

**cd base\_images**

В случае недостаточности места для образов docker необходимо перенести образы на другой диск (**/opt/nginx**) с последующим создание символической ссылки:

**sudo systemctl stop docker**

**cd /var/lib**

**sudo cp -r docker/ /opt/nginx/**

**sudo rm -r docker/**

**sudo ln -s /opt/nginx/docker/**

**sudo systemctl start docker**

*Завершив перенос не забудьте вернуться в каталог с образами (/home/isu\_test/base\_images)*

* + 1. Скачать начальные образы:

**sudo gunzip -c nginx\_tar.gz | sudo docker load**

**sudo gunzip -c python\_tar.gz | sudo docker load**

**sudo gunzip -c jupyterhub\_base\_tar.gz | sudo docker load**

**sudo gunzip -c uwsgi\_base\_tar.gz | sudo docker load**

## Сборка конечных образов

* + 1. Собрать конечные образы для развертки продукта:

**sudo cd /opt/toolkit**

**sudo git clone ssh://git@10.21.25.60:8878/isu\_ivb/isu-toolkit-devops.git**

**cd isu-toolkit-devops/last\_images/jupyterhub**

**sudo docker build -t jupyterhub .**

**cd ../scheduler/**

**sudo docker build -t scheduler .**

**cd ../uwsgi/**

**sudo docker build -t uwsgi .**

## Выпуск серверного сертификата для запуска gateway

* + 1. Выпустить серверный сертификат для gateway (Сертификат 9 см. Получение сертификатов TLS) в Сommon Name указать адрес сервера gateway (Доменное имя).
		2. Создать папки для хранения сертификатов

**sudo mkdir /opt/toolkit /opt/toolkit/cert /opt/toolkit/cert/ssl\_nginx**

* + 1. Положить выпущенные сертификат и закрытый ключ в папку: **/opt/toolkit/cert/ssl\_nginx**
		2. Преобразовать полученный сертификат в PEM формат:

**sudo openssl x509 -inform der -in cert.cer -out cert.pem**

* + 1. Добавить сертификат верификации:

**sudo cp /opt/toolkit/isu-toolkit-devops/base\_images/verify.pem /opt/toolkit/cert/ssl\_nginx/**

## Запуск gateway

* + 1. Удалить старый конфигурационный файл nginx:

**sudo rm -r /opt/nginx/conf/nginx.conf**

* + 1. Скопировать конфигурационный файл из репозитория в рабочую область nginx:

**sudo cp /opt/toolkit/isu-toolkit-devops/base\_images/nginx.conf /opt/nginx/conf/nginx.conf**

* + 1. Перезапустить nginx:

**sudo systemctl restart nginx**

## Установка библиотеки

* + 1. Клонирем билиотеки из git-репозитория:

**cd /opt/toolkit**

**sudo git clone ssh://git@10.21.25.60:8878/isu\_ivb/isu-toolkit\_library.git**

* + 1. Произвести компиляцию образа библиотеки:

**cd isu-toolkit\_library/toolkit\_library**

**sudo docker build -t library .**

# Создание нового клиента

## Установка конвейера и репозитория развертки клиента

* + 1. Создать папку с названием [**Имя клиента**]

**mkdir /opt/toolkit/[Имя клиента]**

* + 1. Установить конвейер из git-репозитория:

**cd /opt/toolkit/[Имя клиента]**

**sudo git clone ssh://git@10.21.25.60:8878/isu\_ivb/isu-toolkit\_conveyer.git**

* + 1. Установить репозиторий развертки клиента:

**git clone ssh://git@10.21.25.60:8878/isu\_ivb/isu\_create\_client.git**

## Установка БД для библиотеки и конвейера

* + 1. Зайти в БД под администратором
		2. Создать БД клиента:

**CREATE DATABASE [Имя клиента]**

* + 1. Раздать права клиентскому пользователю сервера Postgry

**GRANT ALL PRIVILEGES ON DATABASE [Имя клиента] TO [Имя клиента]**

* + 1. Внести изменения в конфигурационный файл **/opt/toolkit/[Имя клиента]/isu\_create\_client/database/database.ini**
* name - **[Имя клиента]**;
* database - **[Имя клиента];**
* user\_client - **[Имя клиента];**
* password\_client – **[Пароль клиента БД];**
* host – **[Host сервера БД].**
	+ 1. Внести изменения в конфигурационный файл **/opt/toolkit/[Имя клиента]/isu\_create\_client/database/docker-compose.yml:**

**<client\_name> - [Имя клиента]**

* + 1. Создать БД конвейера:

**cd /opt/toolkit/[Имя клиента]/isu\_create\_client/database**

**sudo docker-compose up**

*Как результат работы команда отобразится сообщение* ***database\_lib\_1 exited with code 0***

* + 1. Внести изменения в конфигурационный файл /opt/toolkit/client\_ift/isu\_create\_client/lib\_database/docker-compose.yml:

- environment, Параметр ISU\_LIB\_NAME - **[Имя клиента];**

- environment, Параметр ISU\_LIB\_USER – **[Имя клиента];**

- environment, Параметр ISU\_LIB\_PASSWORD - **[Пароль клиента БД];**

**-** extra\_hosts, Параметр ISU\_LIB\_HOST - **[Host сервера БД];**

- volume, Параметр <client\_name> - **[Имя клиента].**

* + 1. Запустить миграционный скрипт БД:

**cd /opt/toolkit/[Имя клиента]/isu\_create\_client/lib\_database**

**sudo docker-compose up**

*Как результат работы команда отобразится сообщение* ***lib\_database\_lib\_1 exited with code 0***

## Создание сертификатов для взаимодействия между Gateway и Ingress Proxy

* + 1. Измените в файле **/opt/toolkit/[Имя клиента]/isu\_create\_client/create\_cert.sh следующие атрибуты:**

- <server\_name> - **[Доменное имя сервера]**

* + 1. Создать сертификаты для взаимодействия между Gateway и Ingress клиента:

**cd /opt/toolkit/[Имя клиента]/isu\_create\_client**

**sudo bash create\_cert.sh**

## Редактирование конфигурационных файлов

* + 1. Внести изменения в конфигурационный файл **/opt/toolkit/[Имя клиента]/isu\_create\_client.repo.conf**

- в разделе [isu\_url] изменить атрибут adress = http://10.53.125.35:8080 [Адрес gateway АС ИСУ]

- в разделах [default\_db], [lib\_db], [core\_db], [journal\_db] - изменить везде атрибут connection\_string = host='<client\_host>' port='5432' dbname='<client\_name>' user='<client\_name>' password='<user\_passwd>';

<client\_host> - **[Host сервера БД]**

<client\_name> - **[Имя клиента]**

<user\_passwd> - **[Пароль клиента БД]**

- в разделе [tool\_db] изменить атрибут connection\_string = host='<client\_host>' port='5432' dbname='<client\_name>' user='<client\_name>' password='<user\_passwd>'

<client\_host> - **[Host сервера БД ИСУ]**

<client\_name> - **[Имя клиента toolkit БД ИСУ]**

<user\_passwd> - **[Пароль клиента БД ИСУ]**

* + 1. Внести изменения в конфигурационный файл **/opt/toolkit//[Имя клиента]/scheduler.ini:**

db\_database = **[Имя клиента]**

db\_user = **[Имя клиента]**

db\_password = **[Пароль клиента БД]**

db\_host = **[Host сервера БД]**

* + 1. Внести изменения в конфигурационный файл Jupyterhub **/opt/toolkit/client\_ift/jupyterhub.py /opt/toolkit/[Имя клиента]/jupyterhub.py**

c.Authenticator.admin\_users – пользователь-администратор jupyterhub

* + 1. Внести изменения в конфигурационный файл **/opt/toolkit/[Имя клиента]/docker-compose.yml**

- во всех разделах volume скорректировать путь к клиентским репозиториям <client\_name> на [Имя клиента]

- в разделе lib:enveronment и extra\_hosts заменить параметры аналогично пункту 2.2.7.

- в разеделе jupyterhub:enveronment параметр <client\_name> = **[Имя клиента]**

- в разеделе nginx:enveronment параметры <client\_name> = **[Имя клиента]**, <client\_port> = [**Номер свободного для связи порта с gateway**]

- в разеделе nginx:port <client\_port> = [**Номер свободного для связи порта с gateway**]

## Настройка Gateway

* + 1. Добавить в конец файла **/opt/nginx/conf/nginx.conf** до символов «}}»:

location /**[Имя клиента]**/lib/static/ {

 alias /opt/toolkit/**[Имя клиента]**/isu\_create\_client/static/;}

 location /**[Имя клиента]** {

 proxy\_pass https:// **[Доменное имя сервера]**:[**Номер свободного для связи порта с gateway**];

 proxy\_ssl\_certificate\_key /opt/toolkit/**[Имя клиента]**/isu\_create\_client/cert/gateway/priv.key;

 proxy\_ssl\_certificate /opt/toolkit/**[Имя клиента]**/isu\_create\_client/cert/ingress/cert.pem;

 proxy\_ssl\_trusted\_certificate /opt/toolkit/**[Имя клиента]**/isu\_create\_client/cert/gateway/cert.pem;

 proxy\_ssl\_verify on;

 proxy\_http\_version 1.1;

 proxy\_set\_header Host $host;

 proxy\_set\_header X-Real-IP $remote\_addr;

 proxy\_set\_header Upgrade $http\_upgrade;

 proxy\_set\_header Connection $connection\_upgrade;

 proxy\_set\_header SberPDI $ssl\_client\_s\_dn;

 client\_max\_body\_size 100M;

 }

## Запуск контейнеров

* + 1. Выполнить скрипт:

**cd /opt/toolkit/[Имя клиента]/isu\_create\_client**

**sudo docker-compose up**

*Убедитесь, что все пять контейнеров запустились*

## Создание супер-пользователя для библиотеки

* + 1. Запросить список запущенных контейнеров

**sudo docker ps**

* + 1. Скопировать идентификатор контейнера библиотеки, запущенного для текущего клиента [**CONTAINER ID**] ([**IMAGE**] = library)
		2. Зайти в пространство контейнера библиотеки:

**sudo docker exec -i -t [CONTAINER ID] bash**

* + 1. Запустить команду создания нового супер-пользователя библиотеки:

**python3 manage.py createsuperuser**

* + 1. В ответ на запросы системы ввести следующие данные:

**Имя пользователя: admin**

**Адрес электронной почты Alpha:** **Demin.A.And@sberbank.ru**

**Password: [Пароль]**

**Password (again): [Пароль]**

* + 1. Выйти из **конвейера** Cntrl+D

## Перезапуск gateway

* + 1. Перезапустить nginx:

**sudo systemctl restart nginx**

# Получение сертификатов TLS

Получение сертификата делится на три этапа:

1. Генерация приватного ключа
2. Генерация файла-запроса на получение сертификата
3. Отправка запроса в удостоверяющий центр

Перечень сертификатов необходимых для развертывания системы, отражен в Приложении 2

## Генерация приватного ключа

1. Необходимо зайти на сервер, для которого необходимо выпустить сертификат;
2. Запустить командную строку;
3. Ввести и запустить команду:

**openssl genrsa -des3 -out [*Название ключа*].key 2048.** Система выведет сообщение с запросом пароля для приватного ключа (см. Рисунок 3);



Рисунок 5

1. Придумать и ввести пароль. Система выведет сообщение с просьбой подтвердить пароль повторным вводом (см. Рисунок 4);



Рисунок 6

1. Ввести ранее введенный пароль повторно;
2. Приватный ключ сформирован.

## Генерация файла-запроса на получение сертификата

1. Для серверного сертификата FrontApp в командной строке ввести и запустить команду:

**sudo openssl11 req -nodes -newkey rsa:2048 -keyout priv.key -out [XX].csr -subj "/C=RU/ST=Moscow/L=Moscow/O=[Название организации]/OU=[Название подразделения]/CN=[Доменное имя сервера]" -addext "subjectAltName = DNS: [Доменное имя сервера]"**

1. Для всех остальных - в командной строке ввести и запустить команду:

**openssl req -new -key [*Название ключа*].key -out SERT\_[XX]\_[Host\_Name].csr**. Система запросит пароль от приватного ключа (см. Рисунок 5).

XX: BD (для сервера БД), IP (для Ingress Proxy), GT (для Gateway сервера)



Рисунок 7

1. Ввести пароль от приватного ключа. Система запросит информацию для формирования запроса (см. Рисунок 6);



Рисунок 8

1. Необходимо последовательно ввести следующие данные (см. Рисунок 7):
	1. Страна;
	2. Штат;
	3. Город;
	4. Организация;
	5. Common Name:

- Клиентский сертификат для сервера БД – Наименование пользователя, от имени которого приложения обращаются к серверу БД;

- Серверный сертификат для сервера БД – Адрес сервера на котором БД (Доменное имя);

- Серверный сертификат для Ingress Proxy - значение Host сервера;

- Клиентский сертификат Ingress Proxy - значение Host-name сервера Ingress Proxy.

- Серверный сертификат FrontApp - значение Host-name сервера FrontApp.

- Пользовательский сертификат FrontApp – значение Host-name сервера Front App.

- Клиентский сертификат для Toolkit.core - значение Host-name сервера Toolkit.core

- Серверный сертификат для /pushTasks - значение Host-name сервера /pushTasks

- Серверный сертификат gateway – Адрес сервера gateway (Доменное имя)

* 1. E-mail адрес;
	2. Придумать и ввести Challenge-пароль;
	3. Альтернативное имя компании не заполняется;



Рисунок 9

1. Файла-запроса на получение сертификата сгенерирован.

## Отправка запроса в удостоверяющий центр

1. Сформировать заявку для получения сертификата в удостоверяющий центр;
2. Приложить сформированный на предыдущем этапе файл-запрос на получение сертификата.

# Приложение

## Создание приватного и публичного rsa ключей

* 1. Сгенерировать приватный ключ

**openssl genrsa -out private\_key.pem 4096**

* 1. На основе приватного - сгенерировать публичный ключ

**openssl rsa -pubout -in private\_key.pem -out public\_key.pem**

* 1. Переконвертировать приватный ключ в pkcs8 формат

**openssl pkcs8 -topk8 -in private\_key.pem -inform pem -out private\_key\_pkcs8.pem -outform pem -nocrypt**

В файле **./client/env/jwt.env** для переменной **JWT\_PRIVATE\_KEY** указывается содержимое файла **private\_key\_pkcs8.pem** без строчек -----BEGIN PRIVATE KEY----- и -----END PRIVATE KEY-----, а для переменной **JWT\_PUBLIC\_KEY** – содержимое файла **public\_key.pem** без строчек -----BEGIN PUBLIC KEY----- и -----END PUBLIC KEY-----. После редактирования файла **./client/env/jwt.env** все \*.pem файлы необходимо удалить.

## Список сертификатов TLS

|  |  |  |  |
| --- | --- | --- | --- |
| **№** | **Тип сертификата** | **Формат имени** | **Комментарий** |
| Сертификат 1 | Пользовательский сертификат FrontApp |   |  |
| Сертификат 2 | Клиентский сертификат для Ingress Proxy | [Host Name сервера].cer |  |
| Сертификат 3 | Серверный сертификат FrontApp | [Host Name сервера].cer |  |
| Сертификат 4 | Серверный сертификат для Ingress Proxy | [Host Name сервера].cer |  |
| Сертификат 5.1 | Клиентский сертификат для сервера БД | [Имя пользователя БД].cer |  |
| Сертификат 5.2 | Клиентский сертификат для сервера БД | [Имя пользователя БД].cer |  |
| Сертификат 6 | Серверный сертификат для сервера БД | [Host Name сервера].cer |  |
| Сертификат 7 | Клиентский сертификат для Toolkit.core |   |  |
| Сертификат 8 | Серверный сертификат для /pushTasks |   |  |
| Сертификат 9 | Серверный сертификат Gateway | cert.cer |  |
| Сертификат 10 | Клиентский сертификат Gateway |   |  |
| Сертификат 11 | Серверный сертификат для Ingress Proxy |   |  |
| Сертификат 12 | Клиентский сертификат для сервера БД |   |  |