Руководство по установке

Компонента Cost Calculator (SZUX)

Продукта Platform V Cost Calculator (SZU)

ОГЛАВЛЕНИЕ

Таблица 1. Термины и определения

Термин	Определение
DataBase DB	База данных
Система оркестрации контейнеризированных приложений	Kubernetes, а также OpenShift (опционально)
Docker images Docker-образ Образ	Исполняемый пакет, содержащий все необходимое для запуска приложения: код, среду выполнения, библиотеки, переменные окружения и файлы конфигурации
Пуллинг образа	Загрузка образа во внутренний Registry
Registry Docker registry	Ресурс для хранения образов
Deploy	Установка/развертывание
Common репозиторий	Репозиторий с configmaps стендов
Configmap	Конфигурацинный файл
Namespace	Проектная область системы оркестрации контейнеризированных приложений
UI системы оркестрации контейнеризированных приложений	Интерфейс для управления компонентами системы оркестрации контейнеризированных приложений
Installer	Инструмент для автоматизированной установки. Продукт Platform V DevOps Tools, компонент Deploy tools (CDJE).
ISTIO	Настраиваемая сервисная сеть (service mesh) с открытым исходным кодом, служащая для взаимодействия, мониторинга и

Термин	Определение
	обеспечения безопасности контейнеров в кластере Kubernetes. Рекомендуется использовать сервис istio в составе продукта Platform V Synapse Service Mesh.
Платформа	Набор продуктов Platform V, правообладателем которых является АО «СберТех». Перечень таких продуктов обозначен в документации на конкретный Продукт.

Системные требования

Требования к окружению для компонента продукта

Настройки безопасности окружения и перечень платформенных (дополнительных внешних) продуктов, используемых для установки, настройки и контроля в конечной информационной системе, выбираются при разработке конечной ИС, исходя из характера обрабатываемой в ней информации и иных требований информационной безопасности, предъявляемых к ней.

- **Системное программное обеспечение**

Для функционирования компонента Cost Calculator (SZUX) продукта Platform V Cost Calculator (SZU) необходима установка следующего программного обеспечения сторонних правообладателей.

- Операционная система

  Таблица 2. Операционная система

Наименование	Версия	Применение
ALT Linux 9	Sp8 и выше	Рекомендовано
Red Hat Enterprise Linux	7.8 и выше	Опционально

- Среда контейнеризации

  Таблица 3. Среда контейнеризации

Наименование	Версия	Применение
Kubernetes	1.21 и выше	Рекомендовано

Red Hat OpenShift	4.0 и выше	Опционально

Здесь и далее поддерживаемой системой приложений-контейнеров является Kubernetes (использование OpenShift – опционально), в именах и параметрах системы могут встречаться названия систем контейнеризации, в этом случае они одинаковы и применимы для обоих сред контейнеризации.

- Средство контейнеризации

Таблица 4. Средство контейнеризации

Наименование	Версия	Применение
Docker CE		Рекомендовано

- Java-машина

Таблица 5. Java-машина

Наименование	Версия	Применение
OpenJDK	11 и выше	Рекомендовано

- NGINX

Таблица 6. Nginx

Наименование	Версия	Применение
Nginx	1.20.1 и выше	Рекомендовано

- Система управления базами данных (СУБД)

Таблица 7. СУБД

Наименование	Версия	Применение
PostgreSQL	4.4.0 и выше	Рекомендовано (рекомендуется использование Platform V Pangolin SE)

- Сервер приложений

Таблица 8. Сервер приложений

Наименование	Версия	Применение
Tomcat Embedded	9.0.45 и выше	Рекомендовано

- Браузер

Таблица 9. Браузер

Наименование	Версия	Применение
Яндекс Браузер	21.2.0 и выше	Рекомендовано
Google Chrome		Опционально
Safari	Любая актуальная	Опционально

- Репозиторий (хранилище дистрибутива/исходного кода)

  Таблица 10. Репозиторий (хранилище дистрибутива/исходного кода)

Наименование	Версия	Применение
Nexus-Public	3.38.1 и выше	Рекомендовано
Nexus Repository Manager PRO	3.37.0-01 / 7.8.1 и выше	Опционально

- Сервис централизованного хранения репозиториев исходного код

  Таблица 11. Сервис централизованного хранения репозиториев исходного код

Наименование	Версия	Применение
GitLab Community Edition	13.7 и выше	Рекомендовано
BitBucket	7.8.1 и выше	Опционально

- Подключение к системе оркестрации

  Таблица 12. Подключение к системе оркестрации

Наименование	Версия	Применение
Клиент ОС		Рекомендовано

- Подключение к Postgres

  Таблица 13. Подключение к Postgres

Наименование	Версия	Применение
pg_admin4		Рекомендовано

- Система мониторинга сборки

  Внешние продукты не обязательны для установки.

  Таблица 14. Система мониторинга сборки

Наименование	Версия	Применение

Prometheus	2.31 и выше	Рекомендовано
Grafana	2.5.0 и выше	Рекомендовано

- **Платформенные зависимости**

Для настройки, контроля и функционирования компонента реализована интеграция с программными продуктами Платформы, правообладателем которых является АО «СберТех».

Таблица 15. Требования к окружению

Наименование продукта	Код	Версия	Код и наименование компонента	Обязательность установки	Описание
Platform V Audit SE	AUD	D-04.001.00-2265	AUDT Аудит	Опционально	Сервис для аудирования событий
Platform V Backend	#BD	4.3	OTTS One-Time Password (OTP) / OTT	Опционально	Сервис для аутентификации и авторизации и межсервисных взаимодействий
Platform V DevOps Tools	DOT		CDJE Deploy tools	Опционально	Инструмент для автоматизированной установки
Platform V Pangolin SE	PSQ	4.6.0. и выше	-	Опционально	Система управления базами данных, основанная на

					PostgreSQL

- **Аппаратные требования**

Для компонента продукта требуется минимальная конфигурация аппаратного обеспечения.

Компоненты szux-pl и szux-bh разворачиваются в единой namespace. Компонент szux-bh хранит данные в DB.

Таблица 16. Требования к окружению

Компонент	Назначение	Среда развертывания	Ресурсы CPU/RAM/DISK
szux-pl szux-bh	Front-end Back-end	Kubernetes, OpenShift (опционально)	8/16
PostgreSQL	DataBase	VM	4/16/200

Требования к наименованию элементов

Таблица 17. Требования к наименованию

Release	Тип элемента	Наименование
2.1	Схема DB Пользователь DB	sizing sizing
3.0	Схема DB Пользователь DB	szux_costcalculator sizing

Установка

Перед установкой необходима проверка параметров настройки программного и аппаратного обеспечения среды функционирования:

- операционная система - при установке используются стандартные настройки;
- иное системное ПО:
  - настройки СУБД - при установке используются стандартные настройки;
  - используемые сетевые порты - открыт сетевой порт 5432;
  - наличие прав пользователей:

- администратора для установки СУБД;
- администратора для запуска Docker;
- администратора namespace.

Установка проходит в несколько этапов:

- Установка и настройка DataBase;
- Создание Docker-образов для компонентов продукта;
- Установка в системе оркестрации контейнеризированных приложений:
  - Установка в Kubernetes (ручной способ установки);
  - Установка в OpenShift (опционально) (ручной, автоматический способ установки).

Установка DataBase

Настройка DB осуществляется через любой клиент для работы с PostgreSQL.

С помощью клиента необходимо выполнить следующие этапы настройки:

- настройка пользователя;
- настройка DB;
- настройки схемы в DB.

1. Настройка пользователя.

   Для настройки пользователя выполните команды, указанные в таблице.

   Таблица 18. Настройка пользователя

Название	Скрипт	Комментарий

Название	Скрипт	Комментарий
Создание пользователя	CREATE USER <имя пользователя> WITH PASSWORD '<пароль>';	пароль — пароль из 16 символов. Обязательно использование: букв в разном регистре, символов, цифр, отсутствие последовательных букв, цифр. Пример запрещенных комбинаций: 1234567891011Abc.

2. Настройка DB.

   Для настройки DB выполните команды, указанные в таблице.

   Таблица 19. Настройка DB

Название	Скрипт	Комментарий
Создание DB	CREATE DATABASE <имя DB> WITH OWNER = "<имя пользователя>" ENCODING = 'UTF8';	
Настройка привилегий для пользователя	GRANT ALL ON DATABASE <имя DB> TO "<имя пользователя>";	

3. Настройка схемы.

   Необходимо подключиться к DB под созданным пользователем. Для настройки схемы выполните команды, указанные в таблице.

   Таблица 20. Настройка схемы

Название	Скрипт	Комментарий
Создание схемы	CREATE SCHEMA IF NOT EXISTS <имя схемы>;	

Название	Скрипт	Комментарий
Установка схемы по умолчанию	ALTER ROLE "<имя пользователя>" IN DATABASE <имя DB> SET search_path to <имя схемы>;	

Создание структуры DB для схемы и загрузка бизнес-данных выполняются автоматически при первом запуске szux-bh.

Создание docker-образов

Необходимо создать docker-образа для компонентов:

- docker image szux-bh;
- docker image szux-pl.

Создание docker image szux-bh

1. Создайте на локальном компьютере Dockerfile.

   Укажите полный путь расположение файла **szux-bh.jar** в строке:

   ```
   COPY szux-bh.jar $APP_HOME/app.jar
   ```

   Рекомендуется szux-bh.jar и Dockerfile хранить в одной директории.

   Тело dockerfile:

```
FROM <Указать registry_path>/openjdk-11-
rhel8:latest          ENV USER=jboss     ENV
APP_HOME /home/jboss              ENV JAVA_OPTS=""
ENV EXTRA_JAVA_OPTS=""     ENV SECURITY_PROFILE=""
WORKDIR $APP_HOME      # Определяем
привилегированного пользователя для создания
директорий и назначения прав.       # Объявляем
порт, обслуживаемый приложением.     EXPOSE
8080/tcp     # Порт используемый компонентом
Actuator от Spring Boot для снятия метрик с
приложения     EXPOSE 8088/tcp       USER root
# Выполняем создание пользователя в группе root.
Контейнер всегда стартует от пользователя из группы
root.     RUN id -u $USER &>/dev/null || useradd --
```

```
no-log-init -r -g 0 $USER \      # Создаем
необходимую директорию для логирования.      &&
mkdir logs \      && mkdir config \      # Создаем
необходимую директорию для сертификатов.      &&
mkdir certs \      # Выполняем смену пользователя-
владельца рабочей директории.      && chown -R
$USER:0 . \      # Выполняем установку прав доступа
для пользователя и всей группы владельца.      &&
chmod -R u=rX,g=rX,o=r . \      # Добавляем права
на запись в директорию логирования для пользователя
и группы владельца.      && chmod -R u+w,g+w ./logs
COPY bh/szux-bh-*.jar $APP_HOME/app.jar          #
Определяем пользователя для выполнения последующих
команд: RUN, CMD, ENTRYPOINT.     USER $USER:0
CMD ["sh", "-c","java $JAVA_OPTS $EXTRA_JAVA_OPTS
$SECURITY_PROFILE -
Djava.security.egd=file:/dev/./urandom -jar
$APP_HOME/app.jar"]
```

Registry_path – внутренний или общедоступный docker registry.

2. Создайте docker image szux-bh. Для создания и размещения образа в пространстве выполните команды, указанные в таблице.

   Таблица 21. Создание docker image szux-bh

Команда	Скрипт	Параметры
create docker image szux-bh	docker build -t < your to path > /szux-bh:$	your to path — путь в docker registry distrib_version - версия дистрибутива (необходимо для версионирования образов)

Команда	Скрипт	Параметры
docker push szux-bh to registry	docker push < your to path >/szux-bh:$	your to path — путь в docker registry distrib_version - версия дистрибутива (необходимо для версионирования образов)

Создание docker image szux-pl

1. Создайте на локальном компьютере Dockerfile.

   Укажите корректное место расположение файла sizing.zip в строке:

   `COPY sizing.zip/sizing.zip`

   Рекомендуется bandle (sizing.zip) и с Dockerfile хранить в одной директории.

   Тело dockerfile

```
FROM <Укажите registry_path>/nginx-116-rhel7:latest
ENV USER=nginx                    COPY sizing.zip
/sizing.zip                   RUN useradd -ms
/bin/bash $USER \           # Распаковываем
sizing.zip             && unzip /sizing.zip -d / &&
rm -rf /siznig.zip \           # Создаем каталога
под bundle             && mkdir -p
/usr/share/nginx/html/ \           # Копируем
bundle в директорию /usr/share/nginx/
&& mv /sizing /usr/share/nginx/html/ \           #
Выполняем смену пользователя-владельца директорий и
выдачу прав             && chmod -R 755
/usr/share/nginx/ \             && touch
/var/log/nginx/nginx.error.log \             &&
chmod -R 777 /var/log/ \             && chown -R
nginx:nginx /var/spool/nginx \             && chmod
-R 755 /var/spool/nginx \             && chown -R
nginx /etc/nginx \             && chmod -R 777
/var/run/ \             && chown -R nginx /var/run/
```

```
\              && chown -R nginx:nginx
/usr/share/nginx/   USER nginx
```

Registry_path – внутренний или общедоступный docker registry.

2. Создайте docker image szux-pl.

   Для создания и размещения образа в пространстве выполните команды, указанные в таблице.

   Таблица 22. Создание docker image szux-pl

Команда	Скрипт	Параметры
create docker image szux-pl	docker build -t < your to path >/szux-pl:$.	your to path — путь в docker registry distrib_version - версия дистрибутива (необходимо для версионирования образов)
docker push szux-pl to registry	docker push < your to path >/szux-pl:$	your to path — путь в docker registry distrib_version - версия дистрибутива (необходимо для версионирования образов)

Установка в системах оркестрации контейнеризированных приложений

Доступна установка в следующих системах оркестрации контейнеризированных приложений:

- Kubernetes;
- OpenShift (опционально).

Установка в Kubernetes

Установку в Kubernetes можно проводить следующими способами:

- в ручном режиме.

Для установки необходимо иметь доступ к namespace.

Ручная установка

Для установки необходимо выполнить все этапы установки.

1. Подключение к Kubernetes

   Для подключения выполните следующие действия:

   - откройте UI Kubernetes;
   - в меню выбора namespace - выберите namespace.
2. Создание secret

   Создание secret необходимо для выгрузки докер образов из registry.

   - Создайте файл config.base64;

     Тело файла

```
{      "auths": {
"dzo.sw.sbc.space": {
"auth": "<Укакжите auth base64>"              }
}    }
```

   Auth base64 – сконвертированный в формат base64 login:password.

   - Создайте файл secret.yaml;

     Тело файла

```
kind: Secret apiVersion: v1 metadata:  name:
costcalc  namespace: <Укажите наименование
namespace> data:  .dockerconfigjson: >-  <Укажите
config base64> type:
kubernetes.io/dockerconfigjson
```

   Config base64 – сконвертированное в формат base64 содержимое файла config.base64.

   - Перейдите в UI Kubernetes и нажмите кнопку добавления нового ресурса кнопкой +;

- Выберите формат ввода информации;
- Добавьте в namespace содержимое файла secret.yaml;
- Подтвердите сохранение информации кнопкой **Upload**.

3. Настройка secrets для подключения к DB
   - Создайте файл;

     Тело файла

```
kind: Secret apiVersion: v1 metadata: name:
secret-szux namespace: c data:
szux_bh_extra_java_opts: >- <Укажите password to
base64> type: Opaque
```

     Password to base64 - сконвертированнsq в формат base64 пароль пользователя для подключения к db Postgres.

   - Перейдите в UI Kubernetes и нажмите кнопку добавления нового ресурса кнопкой +;
   - Выберите формат ввода информации;
   - Добавьте в namespace содержимое файла;
   - Подтвердите сохранение информации кнопкой **Upload**.
4. Выпуск сертификатов.

   Выпустите следующие сертификаты:

   - Istio [ingress, egress] при использовании Istio;
   - сертификаты клиентский и серверный для подключения к DB по SSL при необходимости;
   - сертификат для OTT при необходимости.
5. Использование Istio

При использовании Istio:

- используйте файлы из каталога /Istio из дистрибутива;
- актуализируйте параметры переменных, описанных в дистрибутиве config/paremetrs/;
- перейдите в UI Kubernetes и нажмите кнопку добавления нового ресурса +;
- выберите формат ввода информации;
- добавьте в namespace содержимое файлов;

- подтвердите сохранение информации кнопкой **Upload**.

6. Deploy szux-bh

Для Deploy szux-bh необходимо:

- загрузите файл ConfigMap **szux-bh-config**:
  - скопируйте файл из дистрибутива ConfigMap -> szux-bh-config;
  - отредактируйте < stand id >, < namespace >, < host DB Postgres >, < port DB Postgres >, < name DB Postgres >;
  - перейдите в UI Kubernetes и нажмите кнопку добавления нового ресурса +;
  - выберите формат ввода информации;
  - добавьте в namespace содержимое файлов;
  - подтвердите сохранение информации кнопкой **Upload**.

  Тело файла

```
kind: ConfigMap apiVersion: v1 metadata:  name:
szux-bh-config-< Укажите stand id >  namespace:
<Укажите namespace> data:  JAVA_OPTS: >-   -
Dfile.encoding=UTF-8    -
Dspring.datasource.url=jdbc:postgresql://<Укажите
host DB Postgres>:<Укажите port DB
Postgres>/<Укажите name DB Postgres>    -
Dspring.datasource.username=sizing    -
Dspring.datasource.platform=postgres    -
Dspring.datasource.driver-class-
name=org.postgresql.Driver    -
Dspring.jpa.properties.hibernate.format_sql=true
-
Dspring.jpa.properties.hibernate.default_schema=s
izing    -Dspring.jpa.database-
platform=org.hibernate.dialect.PostgreSQLDialect
-
Dspring.jpa.properties.hibernate.event.merge.enti
ty_copy_observer=allow    -
Dspring.profiles.active=dev
```

  Stand id – префикс стенда, например, dev.

- разверните **szux-bh**:
  - скопируйте файл из дистрибутива szux-bh → dc.yaml;

- отредактируйте параметры переменных, описанных в дистрибутиве config/paremetrs/;
- перейдите в UI Kubernetes и нажмите кнопку добавления нового ресурса +;
- выберите формат ввода информации;
- добавьте в namespace содержимое файлов;
- подтвердите сохранение информации кнопкой **Upload**;
- перейдите в раздел Pods и дождитесь развертывания (Status= running) pod szux-bh-<stand.id>.

- deploy **szux-bh**:
  - скопируйте файл из дистрибутива szux-bh → service.yaml;
  - отредактируйте параметры переменных, описанных в дистрибутиве config/paremetrs/;
  - перейдите в UI Kubernetes и нажмите кнопку добавления нового ресурса +;
  - выберите формат ввода информации;
  - добавьте в namespace содержимое файлов;
  - подтвердите сохранение информации кнопкой **Upload**.

7. Deploy szux-pl

Для Deploy szux-pl необходимо:

- загрузите файл ConfigMap **szux-pl-config**:
  - скопируйте файл из дистрибутива Config Map → szux-pl-config;
  - отредактируйте параметры переменных, описанных в дистрибутиве config/paremetrs/;
  - перейдите в UI Kubernetes и нажмите кнопку добавления нового ресурса +;
  - выберите формат ввода информации;
  - добавьте в namespace содержимое файлов;
  - подтвердите сохранение информации кнопкой **Upload**.
- разверните **szux-pl**:
  - скопируйте файл из дистрибутива szux-pl → dc.yaml;
  - отредактируйте параметры переменных, описанных в дистрибутиве config/paremetrs/;
  - перейдите в UI Kubernetes и нажмите кнопку добавления нового ресурса +;

- выберите формат ввода информации;
- добавьте в namespace содержимое файлов;
- подтвердите сохранение информации кнопкой **Upload**;
- перейдите в раздел Pods и дождитесь развертывания (Status= running) pod szux-pl-<stand.id>.

- deploy **szux-pl**:
  - скопируйте файл из дистрибутива szux-pl → service.yaml;
  - отредактируйте параметры переменных, описанных в дистрибутиве config/paremetrs/;
  - перейдите в UI Kubernetes и нажмите кнопку добавления нового ресурса +;
  - выберите формат ввода информации;
  - добавьте в namespace содержимое файлов;
  - подтвердите сохранение информации кнопкой **Upload**.

8. Настройка аутентификации

   Настройка подключения аутентификации в продукте описана в разделе Настройка аутентификации документа **Руководство по системному администрированию**.

9. Настройка аудита

   Настройка подключения аудита описана в разделе Настройка аудита документа **Руководство по системному администрированию**.

Установка в OpenShift (опционально)

Установку в OpenShift можно проводить следующими способами:

- в ручном режиме;
- с помощью Installer.

Для установки необходимо иметь доступ к namespace.

Ручная установка

Для установки необходимо выполнить все этапы установки.

1. Настройка пуллинга образа компонентов (szux-bh, szux-pl) в namespace:
   - откройте **UI OpenShift**;

- перейдите в раздел **Workloads → Secrets**:
- нажмите кнопку добавления Secret с типом Image Pull Secret;
- укажите следующие настройки:
  - secret name: имя в нижнем регистре;
  - registry server address: <Указать путь до registry server address>;
  - username и password: имя и пароль ТУЗа с доступом в реестр, из-под которого будут пуллиться образы из registry в проект (namespace) OpenShift;
- перейдите в раздел **User Management → Service Account**;
- выберите в списке аккаунт **default**;
- перейдите на вкладку **yaml**;
- в области **imagePullSecrets** добавьте еще одну запись:

  — name: <имя_созданного_image_pull_secret>.

2. Выпуск сертификатов.

   Выпустите следующие сертификаты:

   - Istio [ingress, egress] при использовании;
   - сертификаты клиентский и серверный для подключения к DB по SSL;
   - сертификат для OTT при необходимости.
3. Подключение к OpenShift.

   В терминале запустите команду:

   ```
   oc login https://<your_api_host_OpenShift>
   ```

   Введите login/password учетной записи, которая имеет доступ к Openshift, your_api_host_OpenShift.

4. Подключение к проекту.

   В терминале запустите команду:

   ```
   oc project <имя_namespace>
   ```

   Введите наименование проекта (namespace) в OpenShift, в котором будет развернут продукт.

5. Настройка secret для подключения к DB.

   Порядок выполнения настройки:

- откройте UI OpenShift;
- перейдите в раздел **Workloads → Secrets**;
- нажмите кнопку добавления Secret с типом Key/Value Secret;
- укажите следующие настройки:
  - Secret Name: db-secert;
  - Key: extra_java_opts;
  - Value: -Dspring.datasource.password= .
- укажите password от учетной записи для подключения к DB PostgreSQL.

6. Добавление szux-bh image в OpenShift.

   В терминале запустите команду:

   ```
   oc tag registry/szux-bh:<module_version> <namespace>/szux-bh:<module_version>
   ```

   Укажите в команде актуальные данные:

   - registry;
   - версия модуля szux-bh из registry (пример latest);
   - namespace;
   - версия модуля szux-bh для registry OpenShift.

7. Добавление szux-pl image в OpenShift.

   В терминале запустите команду:

   ```
   oc tag registry/szux-pl:<module_version> <namespace>/szux-pl:<module_version>
   ```

   Укажите в команде актуальные данные:

   - registry;
   - версия модуля szux-pl из registry (пример latest);
   - namespace;
   - версия модуля szux-pl для registry OpenShift.

8. Заполнение configmap **szux.conf**

   Пример:

```
#oc props example                STAND_ID=ift
#указываем id стенда
```

```
DOCKER_REGISTRY=registry                #указываем
registry
SZUX_PL_VERSION=$SZUX_PL_VERSION #версия image
модуля szux-pl
SZUX_BH_VERSION=$SZUX_BH_VERSION #версия image
модуля szux-bh
JDBC_URL=jdbc:postgresql://             #url database
Postgres        POSTGRES_USER=adm
#указываем учетную запись для подключения к
Postgres        SZUXBH_URL=szux-bh-<STAND_ID>-
<namespace>.apps.<url api(<указать путь до api>)>
#указываем ресурсы:
szux_bh_resources_limits_cpu=500m
szux_bh_resources_request_cpu=300m
szux_bh_resources_limits_memory=2Gi
szux_bh_resources_request_memory=1Gi
szux_pl_resources_limits_cpu=256m
szux_pl_resources_request_cpu=128m
szux_pl_resources_limits_memory=256Mi
szux_pl_resources_request_memory=128Mi
```

8.1. Настройка аутентификации

Параметры настройки описаны в разделе Настройка аутентификации документа **Руководство по системному администрированию**.

9. Установка szux-bh в OpenShift:

   В терминале запустите команду:

   ```
   oc process -n <имя namespace> -f <template> --
   param-file=<props>                --ignore-unknown-
   parameters=true | oc apply -n <namespace> -f
   ```

   Укажите в команде актуальные данные:

   - namespace;
   - template (пример szux-bh-template.yaml);
   - param-file (пример szux-ift.conf).

10. Установка szux-pl в OpenShift.

В терминале запустите команду:

```
```oc process -n <namespace> -f <template> --param-file=<props> --ignore-unknown-parameters=true | oc apply -n <namespace> -f```
```

Укажите в команде актуальные данные:

- namespace;
- template (пример szux-pl-template.yaml);
- param-file (пример szux-ift.conf).

11. Завершение работы.

Закройте сессию подключения к OpenShift в терминале для учетной записи с помощью команды:

```
```oc logout```
```

12. Настройка аудита

Параметры настройки описаны в разделе Настройка аудита документа **Руководство по системному администрированию**.

Установка через Installer

Установка дистрибутива производится через Installer. При установке используется типовая инструкция для работы Installer.

Для установки необходимо выполнить все этапы установки.

1. Миграция конфигурационных файлов.

В ПО Jenkins, перед установкой job Deploy (Installer) выберите и запустите шаг(playbook): **Миграция конфигурационных файлов ФП**. В результате выполнения шага будут мигрироваться следующие файлы:

- szux-bh.conf;
- szux-pl.conf;
- szux-all.conf;
- szux.isito.all.conf.

szux-bh.conf

Таблица 23. Параметры файла szux-bh.conf

№	Параметр файла	Описание
1	szux.bh.ose.deployment.spec.replicas	Параметр задает количество запускаемых реплик для szux-bh
2	szux.bh.ose.deployment.spec.revisionHistoryLimit	Параметр устанавливает ограничение для истории ревизий Deployment — для снижения количества репликасетов, хранящихся в кластере
3 4	szux.bh.ose.deployment.spec.strategy.activeDeadlineSeconds szux.bh.ose.deployment.spec.strategy.rollingParams.intervalSeconds	Параметры стратегии Deploy. Время ожидания между опросом состояния Deployment после обновления
5	szux.bh.ose.deployment.spec.strategy.rollingParams.maxSurge	Максимальное число POD, которые могут быть запланированы выше исходного количества POD (процент/количество)
6	szux.bh.ose.deployment.spec.strategy.rollingParams.maxUnavailable	Максимальное число POD, которые могут быть запланированы выше исходного количества POD (процент/количество)
7	szux.bh.ose.deployment.spec.strategy.rollingParams.timeoutSeconds	Время ожидания поднятия POD, прежде чем сдаться и откатится к предыдущему успешному Deployment

№	Параметр файла	Описание
8	szux.bh.ose.deployment.spec.strategy.rollingParams.updatePeriodSeconds	Время ожидания между отдельными обновлениями POD
9	szux.bh.ose.deployment.spec.template.spec.containers.readinessProbe.httpGet.path	end-point для проверки проб readness
10	szux.bh.ose.deployment.spec.template.spec.containers.readinessProbe.httpGet.port	порт для проверки проб readness
11	szux.bh.ose.deployment.spec.template.spec.containers.readinessProbe.httpGet.scheme	схема для проверки проб readness (HTTP)
12	szux.bh.ose.deployment.spec.template.spec.containers.livenessProbe.httpGet.path	end-point для проверки проб liveness
13	szux.bh.ose.deployment.spec.template.spec.containers.livenessProbe.httpGet.port	порт для проверки проб liveness
14	szux.bh.ose.deployment.spec.template.spec.containers.livenessProbe.httpGet.scheme	схема для проверки проб liveness (HTTP)

№	Параметр файла	Описание
15 16	szux.bh.ose.deployment.spec.template.spec.containers.securityContext.runAsNonRoot szux.bh.ose.deployment.spec.template.spec.containers.securityContext.readOnlyRootFilesystem	Настройки безопасности. Контейнер не должен запускаться пользователем root, файловая система должна быть в readOnly
17	szux.bh.ose.deployment.spec.template.annotations.sidecar.istio.io.inject	Переключатель для sidecar istio для szux-bh
18 19 20 21	szux.bh.ose.deployment.spec.template.annotations.sidecar.istio.io.proxyCPU szux.bh.ose.deployment.spec.template.annotations.sidecar.istio.io.proxyMemory szux.bh.ose.deployment.spec.template.annotations.sidecar.istio.io.proxyCPULimit szux.bh.ose.deployment.spec.template.annotations.sidecar.istio.io.proxyMemoryLimit	Ресурсы для sidecar istio в POD szux-bh

№	Параметр файла	Описание
22 23 24 25	szux.bh.ose.deployment.spec.template.spec.containers.resources.cpuLimit szux.bh.ose.deployment.spec.template.spec.containers.resources.memLimit szux.bh.ose.deployment.spec.template.spec.containers.resources.cpuRequests szux.bh.ose.deployment.spec.template.spec.containers.resources.memRequests	Ресурсы для szux-bh

№	Параметр файла	Описание
26 27 28 29 30 31 32 33 34 35	szux.bh.ose.deployment.spec.template.spec.containers.livenessProbe.failureThreshold szux.bh.ose.deployment.spec.template.spec.containers.livenessProbe.periodSeconds szux.bh.ose.deployment.spec.template.spec.containers.livenessProbe.initialDelaySeconds szux.bh.ose.deployment.spec.template.spec.containers.livenessProbe.successThreshold szux.bh.ose.deployment.spec.template.spec.containers.livenessProbe.timeoutSeconds szux.bh.ose.deployment.spec.template.spec.containers.readinessProbe.failureThreshold szux.bh.ose.deployment.spec.template.spec.containers.readinessProbe.periodSeconds szux.bh.ose.deployment.spec.template.spec.containers.readinessProbe.initialDelaySeconds szux.bh.ose.deployment.s	Параметры проб

№	Параметр файла	Описание
36	szux.bh.ose.configmap.data.JAVA_OPTS.profiles.active	Параметр задает профиль, с которым запускается модуль szux-bh
37	szux.bh.ose.configmap.data.datasource.username	Пользователь для подключения к DB
38	szux.bh.ose.configmap.data.datasource.url	url в виде jdbc для синхронизации с DB
39	szux.bh.ose.configmap.data.ssl.datasource.url	url в виде jdbc [ssl] для синхронизации с DB
40	szux.bh.ose.configmap.data.datasource.platform	платформа DB (postgres)
41	szux.bh.ose.configmap.data.hibernate.default_schema	Схема DB szux_costcalculator
42	szux.bh.ose.configmap.data.JAVA_OPTS.jpa.properties.hibernate.default_schema	Схема DB szux_costcalculator

№	Параметр файла	Описание
43 44 45 46 47 48 49 50	szux.bh.ose.configmap.data.JAVA_OPTS.ldap.ant.matcher.pattern szux.bh.ose.configmap.data.JAVA_OPTS.ldap.user.search.base szux.bh.ose.configmap.data.JAVA_OPTS.ldap.user.search.filter szux.bh.ose.configmap.data.JAVA_OPTS.ldap.group.search.base szux.bh.ose.configmap.data.JAVA_OPTS.ldap.group.search.filter szux.bh.ose.configmap.data.JAVA_OPTS.ldap.url szux.bh.ose.configmap.data.JAVA_OPTS.ldap.port szux.bh.ose.configmap.data.JAVA_OPTS.ldap.manager.dn	Параметры для интеграции с LDAP

№	Параметр файла	Описание
51 52 53 54 55 56 57 58 59 60 61 62 63 64	szux.bh.ose.configmap.data.JAVA_OPTS.audit.platform.core.metamodel-path szux.bh.ose.configmap.data.JAVA_OPTS.audit.platform.http.async.endpoints.metamodel szux.bh.ose.configmap.data.JAVA_OPTS.audit.platform.http.async.endpoints.event szux.bh.ose.configmap.data.JAVA_OPTS.audit.platform.http.sync.endpoints.event szux.bh.ose.configmap.data.JAVA_OPTS.security.audit.platform szux.bh.ose.configmap.data.JAVA_OPTS.audit.platform.core.retry-timeout szux.bh.ose.configmap.data.JAVA_OPTS.audit.platform.core.max-queue-size szux.bh.ose.configmap.data.JAVA_OPTS.audit.platform.http.sync.enabled szux.bh.ose.configmap.data.JAVA_OPTS.audit.platform.http.sync.connect-timeout szux.bh.ose.configmap.data	Параметры для интеграции с AUDIT

№	Параметр файла	Описание
65	szux.bh.ose.service.spec.ports.port	Параметр задания внутреннего порта для сервиса szux-bh
66	szux.bh.ose.service.spec.ports.targetPort	Параметр задания внешнего порта для сервиса szux-bh
67	szux.bh.ose.service.spec.ports.protocol	Параметр задания протокола для сервиса szux-bh

szux-pl.conf

Таблица 24. Параметры файла szux-pl.conf

№	Параметр файла	Описание
1	szux.pl.ose.deployment.spec.replicas	Параметр задает количество запускаемых реплик для szux-pl
2	szux.pl.ose.deployment.spec.revisionHistoryLimit	Параметр устанавливает ограничение для истории ревизий Deployment — для снижения количества репликасетов, хранящихся в кластере
3 4	szux.pl.ose.deployment.spec.strategy.activeDeadlineSeconds szux.pl.ose.deployment.spec.strategy.rollingParams.intervalSeconds	Параметры стратегии Deploy. Время ожидания между опросом состояния Deployment после обновления

№	Параметр файла	Описание
5	szux.pl.ose.deployment.spec.strategy.rollingParams.maxSurge	Максимальное число POD, которые могут быть запланированы выше исходного количества POD (процент/количество)
6	szux.pl.ose.deployment.spec.strategy.rollingParams.maxUnavailable	Максимальное число POD, которые могут быть запланированы выше исходного количества POD (процент/количество)
7	szux.pl.ose.deployment.spec.strategy.rollingParams.timeoutSecond	Время ожидания поднятия POD, прежде чем сдаться и откатится к предыдущему успешному Deployment
8	szux.pl.ose.deployment.spec.strategy.rollingParams.updatePeriodSeconds	Время ожидания между отдельными обновлениями POD
9	szux.pl.ose.deployment.spec.template.spec.containers.szux-pl.ports.containerPort	Порт модуля szux-pl
10	szux.pl.ose.deployment.spec.template.spec.containers.szux-pl.ports.protocol	Протокол модуля szux-pl

№	Параметр файла	Описание
11 12	szux.pl.ose.deployment.spec.template.spec.containers.securityContext.runAsNonRoot szux.pl.ose.deployment.spec.template.spec.containers.securityContext.readOnlyRootFilesystem	Настройки безопасности. Контейнер не должен запускаться по пользователем root,файловая система должна быть в readOnly
13 14 15 16	szux.pl.ose.deployment.spec.template.spec.containers.szux-pl.resources.cpuLimit szux.pl.ose.deployment.spec.template.spec.containers.szux-pl.resources.memLimit szux.pl.ose.deployment.spec.template.spec.containers.szux-pl.resources.cpuRequests szux.pl.ose.deployment.spec.template.spec.containers.szux-pl.resources.memRequests	Ресурсы для szux-pl
17	szux.pl.ose.service.spec.ports.port	Параметр задания внутреннего порта для сервиса szux-pl
18	szux.pl.ose.service.spec.ports.targetPort	Параметр задания внешнего порта для сервиса szux-pl

№	Параметр файла	Описание
19	szux.pl.ose.service.spec.ports.protocol	Параметр задания протокола для сервиса szux-pl
20 21 22	szux.pl.ose.spec.host szux.pl.ose.spec.path szux.pl.ose.spec.port.targetPort	Параметры для настройки rout (актуально для dev стенда без Istio)

szux-all.conf

Таблица 25. Параметры файла szux-all.conf

№	Параметр файла	Описание
1	szux.ose.common.openshiftRegistry	Docker Registry
2	szux.ose.common.stand.id	Суффикс стенда
3	szux.ose.common.istio.sidecar.enabled	Рубильник для istio sidecar модуля szux-bh
4	szux.ose.secret.bd.enabled	Рубильник для создания secret ssl под db Postgres
5	szux.ose.default.bd.enabled	Рубильник для синхронизации с DB по login pass
6	szux.ose.common.ott.enabled	Рубильник на включение/выключение использваония OTT
7	szux.ose.common.audit.enabled	Рубильник на включение/выключение использваония Audit
8	szux.ose.common.ldap.enabled	Рубильник для LDAP

szux.isito.all.conf

Таблица 26. Параметры файла szux.isito.all.conf

№	Параметр файла	Описание
1 2 3 4 5 6	szux.ose.istio.egress.deployment.spec.volumes.ott-certs.secret.secretName szux.ose.istio.egress.deployment.spec.volumes.egressgateway-ca-certs.secret.secretName szux.ose.istio.egress.deployment.spec.volumes.egressgateway-certs.secret.secretName szux.ose.istio.ingress.deployment.spec.volumes.ingressgateway-ca-certs.secret.secretName szux.ose.istio.ingress.deployment.spec.volumes.ingressgateway-certs.secret.secretName szux.ose.istio.egress.deployment.spec.volumes.costcalc-certs-bd.secret.secretName	Имена secret
7	szux.ose.istio.deployment.spec.template.spec.containers.istio-proxy.image	Образ Istio-proxy
8	szux.ose.istio.deployment.spec.template.spec.containers.ott-sidecar.image	Образ OTT-sidecar

№	Параметр файла	Описание
9	szux.ose.istio.egress.deployment.spec.replicas	Количество реплик egress
10	szux.ose.istio.egress.deployment.spec.strategy.rollingUpdate.maxSurge	Максимальное число POD, которые могут быть запланированы выше исходного количества POD (процент/количество)
11	szux.ose.istio.egress.deployment.spec.strategy.rollingUpdate.maxUnavailable	Максимальное число POD, которые могут быть запланированы выше исходного количества POD (процент/количество)
12 13 14 15	szux.ose.istio.egress.deployment.spec.template.spec.containers.istio-proxy.resources.limits.cpu szux.ose.istio.egress.deployment.spec.template.spec.containers.istio-proxy.resources.limits.memory szux.ose.istio.egress.deployment.spec.template.spec.containers.istio-proxy.resources.requests.cpu szux.ose.istio.egress.deployment.spec.template.spec.containers.istio-proxy.resources.requests.	Ресурсы для istio-proxy

№	Параметр файла	Описание
	memory	
16 17 18 19	szux.ose.istio.egress.deployment.spec.template.spec.containers.ott-sidecar.resources.limits.cpu szux.ose.istio.egress.deployment.spec.template.spec.containers.ott-sidecar.resources.limits.memory szux.ose.istio.egress.deployment.spec.template.spec.containers.ott-sidecar.resources.requests.cpu szux.ose.istio.egress.deployment.spec.template.spec.containers.ott-sidecar.resources.requests.memory	Ресурсы для ott-sidecar

№	Параметр файла	Описание
20 21 22 23	szux.ose.istio.egress.deployment.spec.template.spec.containers.istio-proxy.securityContext.runAsNonRoot szux.ose.istio.egress.deployment.spec.template.spec.containers.istio-proxy.securityContext.readOnlyRootFilesystem szux.ose.istio.egress.deployment.spec.template.spec.containers.ott-sidecar.securityContext.runAsNonRoot szux.ose.istio.egress.deployment.spec.template.spec.containers.ott-sidecar.securityContext.readOnlyRootFilesystem	Настройки безопасности для [istio-proxy, ott-sidecar] egress. Контейнер не должен запускаться по пользователем root, файловая система должна быть в readOnly
24 25	szux.ose.istio.egress.common.internal.port szux.ose.istio.egress.common.internal.protocol	Внутренний порт egress для http трафика

№	Параметр файла	Описание
26 27 28 29	szux.ose.istio.egress.db.common.internal.port szux.ose.istio.egress.db.common.internal.protocol szux.ose.istio.egress.db.common.ingress.port szux.ose.istio.egress.common.db.ingress.host	Внутренний порт egress для DB tcp трафика
30	szux.ose.istio.ingress.deployment.spec.replicas	Количество реплик ingress
31	szux.ose.istio.ingress.deployment.spec.strategy.rollingUpdate.maxSurge	Максимальное число POD, которые могут быть запланированы выше исходного количества POD (процент/количество)
32	szux.ose.istio.ingress.deployment.spec.strategy.rollingUpdate.maxUnavailable	Максимальное число POD, которые могут быть запланированы выше исходного количества POD (процент/количество)
33	szux.ose.istio.ingress.common.szux-bh-service-port	Service port модуля szux-bh
34	szux.ose.istio.ingress.common.szux-pl-service-port	Service port модуля szux-pl
35	szux.ose.istio.egress.deployment.spec.template.spec.containers.ott-sidecar.readinessProbe.httpGet.port	Порт OTT Sidecar для readiness пробы

№	Параметр файла	Описание
36	szux.ose.istio.egress.deployment.spec.template.spec.containers.ott-sidecar.readinessProbe.initialDelaySeconds	Количество секунд после запуска контейнера до запланированного запуска readiness пробы
37	szux.ose.istio.egress.deployment.spec.template.spec.containers.ott-sidecar.readinessProbe.timeoutSeconds	Количество секунд бездействия, по истечении которых считается, что время readiness пробы окончилось и контейнеризированное приложение не работает
38	szux.ose.istio.egress.deployment.spec.template.spec.containers.ott-sidecar.readinessProbe.periodSeconds	Задержка между выполнением readiness проб
39	szux.ose.istio.egress.deployment.spec.template.spec.containers.ott-sidecar.readinessProbe.successThreshold	Количество раз, когда readiness проба должен сообщить об успехе после того, как он начнет давать сбой, чтобы перезагрузить процесс пробы
40	szux.ose.istio.egress.deployment.spec.template.spec.containers.ott-sidecar.readinessProbe.failureThreshold	Количество раз readiness пробы прежде чем пометит под как недоступный

№	Параметр файла	Описание
41 42 43 44	szux.ose.istio.ingress.deployment.spec.template.spec.containers.istio-proxy.resources.limits.cpu szux.ose.istio.ingress.deployment.spec.template.spec.containers.istio-proxy.resources.limits.memory szux.ose.istio.ingress.deployment.spec.template.spec.containers.istio-proxy.resources.requests.cpu szux.ose.istio.ingress.deployment.spec.template.spec.containers.istio-proxy.resources.requests.memory	Ресурсы для istio-proxy

№	Параметр файла	Описание
45 46 47 48	szux.ose.istio.ingress.deployment.spec.template.spec.containers.ott-sidecar.resources.limits.cpu szux.ose.istio.ingress.deployment.spec.template.spec.containers.ott-sidecar.resources.limits.memory szux.ose.istio.ingress.deployment.spec.template.spec.containers.ott-sidecar.resources.requests.cpu szux.ose.istio.ingress.deployment.spec.template.spec.containers.ott-sidecar.resources.requests.memory	Ресурсы для ott-sidecar
49 50	szux.ose.istio.ingress.deployment.spec.template.spec.containers.istio-proxy.securityContext.runAsNonRoot szux.ose.istio.ingress.deployment.spec.template.spec.containers.istio-proxy.securityContext.readOnlyRootFilesystem	Настройки безопасности для istio-proxy ingress. Контейнер не должен запускаться по пользователем root, файловая система должна быть в readOnly

№	Параметр файла	Описание
51 52	szux.ose.istio.gw.spec.servers.tls.mode szux.ose.istio.gw.ott.spec.servers.tls.mode	Режим tls для istio ingress

№	Параметр файла	Описание
53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70	szux.ose.istio.egress.confi gmap.ott.module.id szux.ose.istio.egress.confi gmap.ott.client.cert.alias szux.ose.istio.egress.confi gmap.ott.certstore.path szux.ose.istio.egress.confi gmap.ott.sevice.hosts szux.ose.istio.egress.confi gmap.ott.trust.store.path szux.ose.istio.egress.confi gmap.ott.service.url szux.ose.istio.egress.confi gmap.ott.certstore.type szux.ose.istio.egress.confi gmap.ott.service.cert.alias szux.ose.istio.egress.confi gmap.ott.authz.version szux.ose.istio.egress.confi gmap.ott.authz.realm szux.ose.istio.egress.confi gmap.ott.client.mmt- comptibility-mode szux.ose.istio.egress.confi gmap.ott.grpc.port szux.ose.istio.egress.confi gmap.ott.http.port	Конфигурация OTT sidecar egress

№	Параметр файла	Описание
71 72 73 74	szux.ose.istio.ingress.common.cn_validation.enabled szux.ose.istio.ingress.ef.validate.bpms.cn.map szux.ose.istio.ingress.ef.validate.user.cn.map szux.ose.istio.ingress.ef.validate.ott.system.cn.map	Валидация по CN/DN
75 76 77 78	szux.ose.istio.egress.common.audit.app.config.protocol szux.ose.istio.egress.common.audit.ingress.host szux.ose.istio.egress.common.audit.app.config.port szux.ose.istio.egress.common.audit.ingress.port	Интеграция с платформенным аудитом
79 80 81 82 83	szux.ose.istio.egress.common.ott.app.config.protocol szux.ose.istio.egress.common.ott.ingress.host.1 szux.ose.istio.egress.common.ott.ingress.host.2 szux.ose.istio.egress.common.ott.app.config.port	Интеграция с платформенным сервисом OTT

№	Параметр файла	Описание
	szux.ose.istio.egress.common.ott.ingress.port	
84 85 86 87 88	szux.ose.istio.egress.common.ldap.app.config.protocol szux.ose.istio.egress.common.ldap.ingress.host szux.ose.istio.egress.ldap.common.internal.port szux.ose.istio.egress.common.ldap.ingress.port szux.ose.istio.egress.ldap.common.internal.protocol	Интеграция с LDAP
89 90	szux.ose.istio.egress.svc.external.ports.enabled szux.ose.istio.egress.svc.external.ports	Порт DB PostgreSQL

2. Выпуск сертификатов.

   Выпустите следующие сертификаты:

   - Istio [ingress, egress] при использовании;
   - сертификаты клиентский и серверный для подключения к DB по SSL;
   - сертификат для OTT при необходимости.

3. Добавление сертификатов в common репозиторий.

   В common репозиторий добавьте сертификаты, указанные в таблице.

   Таблица 27. Сертификаты

Наименование	Расположение

Наименование	Расположение
ingress, egress, ott, db	costcalculator_common_dev/ift/ansible/files/ssl/szux
сервера отт	costcalculator_common_dev/ift/ansible/files/ssl

В common репозиторий добавьте описание параметров обращения к сертификатам.

Таблица 28. Описание параметров обращения к сертификатам

Наименование	Расположение
custom_property.conf.yml	costcalculator_szux_dev/ift/conf/

Пример,

```
# просто_комментарий openshift:  project: 'tribe-
pc-ift-costcalc' # Istio # Ingress
ingressKeyStoreFile: 'ansible/files/ssl/szux/szux-
ingress.jks' ingressKeyStorePass:
'szux.ssl.ose.istio.keyStore.ingress.password'
ingressRootCertAlias: 'root' ingressCertAlias:
'szux-ingress' ingressPrivateKeyAlias: 'szux-
ingress'  # Egress egressKeyStoreFile:
'ansible/files/ssl/szux/szux-egress.jks'
egressKeyStorePass:
'szux.ssl.ose.istio.keyStore.egress.password'
egressRootCertAlias: 'root' egressCertAlias: 'szux-
egress' egressPrivateKeyAlias: 'szux-egress'  ##OTT
сертификаты # Сами кейсторы разместить в commons
среды, пути до кейсторов указать в репозитории фп в
`conf/custom_property.conf.yml`
costcalcCertStoreName: 'costcalculator.p12'
costcalcOttCertStorePath:
'ansible/files/ssl/szux/costcalculator.p12'
ottTrustStoreName: 'ift_sol_std3_ott_public.p12'
ottTrustStorePath:
'ansible/files/ssl/ift_sol_std3_ott_public.p12'
```

4. Заполнение файла _passwords.conf.

   В common репозитории в файл **_passwords.conf** добавьте параметры:

- szux_bh_extra_java_opts=password для синхронизации с DB;
- -Dcost.calculator.ldap.manager.password=password ТУЗ для синхронизации с Ldap;
- Параметры для **istio** (для создания Secrets из jks):
  - szux.ssl.ose.istio.keyStore.ingress.password=password;
  - szux.ssl.ose.istio.keyStore.egress.password=password;
- Параметры для **OTT**:
  - szux_istio_ott_pass_OTT_CERTSTORE_PRIVATE_KEY_PWD=password;
  - szux_istio_ott_pass_OTT_CERTSTORE_PWD=password;
  - szux_istio_ott_pass_OTT_TRUST_STORE_PWD=password.

5. Корректировка параметров в configmap.

   Скорректируйте в репозитории Bitbucket параметры:

   - /conf/config/parametrs/szux-bh.conf:
     - параметры [ConfigMap, LDAP, AUDIT].
   - /conf/config/parametrs/szux.all.conf:
     - url Docker Registry[alpha/sigma];
     - Суффикс стенда [dev/psi/prom].
   - /conf/config/parametrs/szux.isito.all.conf:
     - Образы [istio-proxy, ott-sidecar];
     - Конфиги егресса.

6. Корректировка переключателей в configmap.

   Скорректируйте, при необходимости, в репозитории Bitbucket состояние переключателей:

   Рубильники на включение/отключение в дистрибутиве (по умолчанию выбрано true):

   - /conf/config/parametrs/szux.isito.all.conf [audit, ott, ldap];
   - /conf/config/parametrs/szux.all.conf [istio sidecar];

7. Установка ПО.

   Выполните установку Istio, CostCalculator.

Обновление

Доступно обновление в следующих системах оркестрации контейнеризированных приложений:

- Kubernetes;
- OpenShift (опционально).

Обновление в Kubernetes

Обновление можно проводить следующими способами:

- в ручном режиме.

Ручное обновление

Для обновления необходимо выполнить следующие действия:

- Подключитесь к UI Kubernetes;
- В меню выбора namespace - выберите namespace;
- Перейдите в раздел Deployments;
- Отредактируйте Deployment **szux-bh-<stand.id>**:
  - Кликните по наименованию Deployment **szux-bh-<stand.id>**;
  - Нажмите кнопку **Редактировать**;
  - Откройте вкладку YAML;
  - В поле **image** укажите актуальную версию для szux-bh;
  - Сохраните изменения кнопкой **Update**.

```
containers:  - name: szux-bh    image: "{{
szux.ose.common.openshiftRegistry }}/szux-
bh:<Укажите new_version докер образа>"
```

  - Перейдите в раздел Pods и дождитесь развертывания (Status= running) pod szux-bh-<stand.id>.
- Отредактируйте Deployment **szux-pl-<stand.id>**:
  - Кликните по наименованию Deployment **szux-pl-<stand.id>**;
  - Нажмите кнопку **Редактировать**;
  - Откройте вкладку YAML;
  - В поле **image** указать актуальную версию для szux-pl;
  - Сохраните изменения кнопкой Update.

```
containers:  - name: szux-pl    image: "{{
szux.ose.common.openshiftRegistry }}/szux-
pl:<Укажите new_version докер образа>"
```

- Перейдите в раздел Pods и дождитесь развертывания (Status= running) pod szux-pl-<stand.id>.

Обновление в Openshift (опционально)

Обновление можно проводить следующими способами:

- в ручном режиме;
- с помощью Installer.

Ручное обновление

Для обновления необходимо выполнить последовательно все команды в терминале.

Таблица 29. Обновление

№	Наименование команды	Команда	Дополнительная информация
1	Подключение к OpenShift	`oc login https://<your_api_host_OpenShift>`	Введите login/password УЗ с доступом к Openshift, your_api_host_OpenShift.
2	Подключение к проекту	`oc project <имя_проекта(namespace)>`	Введите наименование проекта (namespace) в OpenShift, в котором будет обновлен продукт.
3	Обновление версии модуля szux-bh в registry OpenShift	`oc tag registry/szux-bh:<module_version> <namespace>/szux-bh:<module_ve`	Укажите registry, версию модуля szux-bh из registry (пример latest), namespace, версию модуля szux-bh для registry OpenShift.

№	Наименование команды	Команда	Дополнительная информация
		rsion>	
4	Обновление версии модуля szux-pl в registry OpenShift	oc tag registry/szux-pl:<module_version> <namespace>/szux-pl:<module_version>	Укажите registry, версию модуля szux-pl из registry (пример latest), namespace, версию модуля szux-pl для registry OpenShift.
5	Завершение сессии	oc logout	Закройте сессию подключения к OpenShift в терминале для УЗ с помощью команды

Обновление через Installer

Обновление дистрибутива производится через Installer с использованием Job Deploy, в которой необходимо выбрать шаг (playbook) **Установка в Openshift**. Для обновления используется типовая инструкция по развертыванию окружения под Installer job [Deploy, Service].

Удаление

Удаление проходит в несколько этапов:

- Удаление компонентов системы оркестрации контейнеризированных приложений;
- Удаление DataBase.

Удаление компонентов системы оркестрации контейнеризированных приложений

Доступно удаление в следующих системах оркестрации контейнеризированных приложений:

- Kubernetes;
- OpenShift (опционально).

Удаление компонентов Kubernetes

Необходимо удаление всех следующих типов компонентов Kubernetes:

- Deployments [egressgw, ingressgw] с префиксом namespace; deployments [szux-bh, szux-pl] с префиксом stand.id;
- Secrets [costcalc-pull-secret, costcalculator-ott-certs, egressgateway-ca-certs, egressgateway-certs, ingressgateway-ca-certs, ingressgateway-certs, istio-secret-szux, secret-szux];
- Config Maps [egressgateway-istio-basic, ingressgateway-istio-basic, sizing-bh-config-ift, szux-ott-egress-sidecar-settings];
- Services [szux-bh, szux-pl] с префиксом наименования стенда, services [egressgw, ingressgw] с префиксом namespace;
- Ingress [istio-ingressgateway-https-route] с префиксом наименования стенда.

Компоненты можно удалить несколькими способами.

- Через терминал:
  - подключитесь через терминал к Kubernetes;
  - выполните команду удаления с актуальными данными.

```
```kubectl -n <имя namespace> delete <тип компонента Kubernetes> <имя компонента Kubernetes>```
```

- Через **UI Kubernetes**:
  - зайдите в **UI Kubernetes**;
  - выберите namespace;
  - выберите тип компонентов;
  - выберите компонент;
```

- выберите в контекстном меню компонента опцию **Delete**.

Удаление компонентов OpenShift (опционально)

Необходимо удаление всех следующих типов компонентов Openshift:

- Deployment Configs [szux-bh, szux-pl] с префиксом наименования стенда;
- Deployments [egressgw, ingressgw] с префиксом namespace;
- Secrets [costcalc-pull-secret, costcalculator-ott-certs, egressgateway-ca-certs, egressgateway-certs, ingressgateway-ca-certs, ingressgateway-certs, istio-secret-szux, secret-szux];
- Config Maps [egressgateway-istio-basic, ingressgateway-istio-basic, sizing-bh-config-ift, szux-ott-egress-sidecar-settings];
- Services [szux-bh, szux-pl] с префиксом наименования стенда, [egressgw, ingressgw] с префиксом namespace;
- Routes [istio-ingressgateway-https-route], [sizing-bh, sizing-pl] с префиксом наименования стенда.

Компоненты можно удалить несколькими способами.

- Через терминал:
  - подключитесь через терминал к Openshift;
  - выполните команду удаления с актуальными данными.

```
```kubectl -n <имя namespace> delete <тип компонента Openshift> <имя компонента Openshift>```
```

- Через **UI Openshift**:
  - зайдите в **UI Openshift**;
  - перейдите в раздел **Project**;
  - выберите namespace;
  - перейдите в раздел **Workloads**;
  - выберите тип компонентов;
  - выберите компонент;
  - выберите в контекстном меню компонента опцию **Delete**.
```

Удаление DataBase

Для удаления DB:

- откройте клиент для работы с DB;
- подключитесь к серверу DB Postgres с привилегированной УЗ;
- выполните удаление DB с помощью команды.

```
```DELETE DATABASE <имя DB>```
```

Проверка работоспособности

Доступно установка/обновление в следующих системах оркестрации контейнеризированных приложений:

- Kubernetes;
- OpenShift (опционально).

Проверка работоспособности установки/обновления зависит от способа установки/обновления.

Проверка работоспособности при работе в Kubernetes

Установку/обновление можно проводить следующими способами:

- в ручном режиме.

Проверка после ручной установки/обновления

Проверка включает следующие действия:

- Проверить наличие в DB структуры и данных;
- Проверить наличие обязательных записей в таблице DATABASECHANGELOG с id из списка: v1.0_0002_ddl_create_postgres, v1.0_alter_seq, v1.1_outgoing_load_tps_double_precision;
- Получить корректный статус Pod: в **UI Kubernetes → Pods → Status** модулей szux-bh, szux-pl = **Running**;
- Выполнить вход по url szux-pl : в **UI Kubernetes → Service**:
  - без использования **Istio**:
    - выбрать **szux-pl-<stand.id>**;
    - нажать кнопку **Редактировать**;
```

        - перейти на вкладку YAML;
        - изменить **type: NodePort** (по умолчанию ClusterIP);
        - сохранить изменения с помощью кнопки **Update**;
        - перейти по ссылке `http://<Указать ip adress Kubenetes>:<Указать Internal Endpoints-node port>`;
    - с использованием Istio:
        - **Ingress** → istio-ingressgateway-https-route;
- Выполнить аутентификацию в продукт.

Проверка работоспособности при работе в OpenShift (опционально)

Установку/обновление можно проводить следующими способами:

- в ручном режиме;
- с помощью Installer.

Проверка после ручной установки/обновления

Проверка включает следующие действия:

- Проверить наличие в DB структуры и данных;
- Проверить наличие обязательных записей в таблице DATABASECHANGELOG с id из списка: v1.0_0002_ddl_create_postgres, v1.0_alter_seq, v1.1_outgoing_load_tps_double_precision;
- Получить корректный статус Pod: в **UI OpenShift → Project → Проект → Workloads → Pods → Status** модулей szux-bh, szux-pl = **Running**;
- Выполнить вход по url szux-pl: в **UI OpenShift → Project → Проект → Networking → Routes → Location**;
- Выполнить аутентификацию в продукт.

Проверка после Installer

Проверка включает следующие действия:

- Получить корректный статус при завершении Job Deploy Installer = **Success**;

- Выполнить вход по по url szux-pl: в UI OpenShift → Project → Проект → Networking → Routes → Location → istio-ingressgateway-https-routel;
- Выполнить аутентификацию в продукт.

Откат

Для отката необходимо откатить:

- компоненты системы оркестрации контейнеризированных приложений;
- DB.

Откат компонентов системы оркестрации контейнеризированных приложений

Доступен откат в следующих системах оркестрации контейнеризированных приложений:

- Kubernetes;
- OpenShift.

Откат компонентов Kubernetes

Откат компонентов Kubernetes может быть произведен следующими способами:

- ручной.

Ручной Откат

Для отката необходимо выполнить следующие действия:

- Подключитесь к UI Kubernetes;
- В меню выбора namespace – выберите namespace;
- Перейдите в раздел Deployments;
- Отредактируйте Deployment **szux-bh-<stand.id>**:
  - Кликните по наименованию Deployment **szux-bh-<stand.id>**;
  - Нажмите кнопку **Редактировать**;
  - Откройте вкладку YAML;
  - В поле **image** укажите актуальную версию для szux-bh;
  - Сохраните изменения кнопкой **Update**.

```
containers:  - name: szux-bh    image: "{{
szux.ose.common.openshiftRegistry }}/szux-
bh:<Укажите version докер образа>"
```

- Перейдите в раздел Pods и дождитесь развертывания (Status= running) pod szux-bh-<stand.id>.

- Отредактируйте Deployment **szux-pl-<stand.id>**:
  - Кликните по наименованию Deployment **szux-pl-<stand.id>**;
  - Нажмите кнопку **Редактировать**;
  - Откройте вкладку YAML;
  - В поле **image** указать актуальную версию для szux-pl;
  - Сохраните изменения кнопкой Update.

```
containers:  - name: szux-pl    image: "{{
szux.ose.common.openshiftRegistry }}/szux-
pl:<Укажите version докер образа>"
```

- Перейдите в раздел Pods и дождитесь развертывания (Status= running) pod szux-pl-<stand.id>.

Откат компонентов OpenShift (опционально)

Откат компонентов OpenShift может быть произведен следующими сопсобами:

- ручной;
- через Installer.

Ручной Откат

Для отката необходимо выполнить последовательно все команды в терминале.

Таблица 30. Откат

№	Наименование команды	Команда	Дополнительная информация

№	Наименование команды	Команда	Дополнительная информация
1	Подключение к OpenShift	`oc login https://<your_api_host_OpenShift>`	Введите login/password УЗ с доступом к Openshift, your_api_host_OpenShift.
2	Подключение к проекту	`oc project <имя_проекта(namespace)>`	Введите наименование проекта (namespace) в OpenShift, в котором будет обновлен продукт.
3	Обновление версии модуля szux-bh в registry OpenShift	`oc tag registry/szux-bh:<module_version> <namespace>/szux-bh:<module_version>`	Укажите registry, версию модуля szux-bh из registry (пример latest), namespace, версию модуля szux-bh для registry OpenShift.
4	Обновление версии модуля szux-pl в registry OpenShift	`oc tag registry/szux-pl:<module_version> <namespace>/szux-pl:<module_version>`	Укажите registry, версию модуля szux-pl из registry (пример latest), namespace, версию модуля szux-pl для registry OpenShift.

№	Наименование команды	Команда	Дополнительная информация
5	Завершение сессии	`oc logout`	Закройте сессию подключения к OpenShift в терминале для УЗ с помощью команды

Откат через Installer

Откат дистрибутива производится через Installer с использование Job Deploy, в которой необходимо выбрать шаг (playbook) **Установка в Openshift**.

Откат DateBase

Для отката DB необходимо выполнить следующие действия:

- подключитесь к СУБД Postgres;
- выберите DB;
- удалите схему:

```
DROP SCHEMA <Укажите schema_name> CASCADE;
```

- создайте новую схему:

```
CREATE SCHEMA <Укажите schema_name>
AUTHORIZATION <Укажите user>;   GRANT ALL ON
SCHEMA schema_name TO <Укажите user>;   ALTER
DEFAULT PRIVILEGES IN SCHEMA <Укажите
schema_name>  GRANT ALL ON TABLES TO <Укажите
user>;
```

Часто встречающиеся проблемы и пути их устранения

Проблемы не выявлены.

Чек-лист валидации установки

1. На сервере развернуто необходимое ПО из раздела Системные требования;
2. Развернута и настроена DB PostgreSql;
3. Созданы docker-образы szux-pl и szux-bh;
4. Выделен проект системы оркестрации контейнеризированных приложений;
5. Произведена настройка проекта системы оркестрации контейнеризированных приложений;
6. Выделена VM с DB PostgreSQL;
7. Произведена первоначальная настройка DB PostgreSQL;
8. Выпущены сертификаты [клиентские/серверные] для:
   - Istio [ingress, egress] при использовании,
   - сертификаты для подключения к DB PostgreSQL по SSL при необходимости.
   - сертификат для OTT при необходимости.
9. Произведен Deploy в системе оркестрации контейнеризированных приложений;
10. Подготовлено окружение в соответствии с типовой инструкцией по настройке Job [Deploy, Service] при установке через Installer.