Руководство по установке

Продукта Platform V Product Pricing

Продукта DYP

Поколение 3

Оглавление

Руководство по установке

Этот документ предназначен для администраторов продукта Platform V Product Pricing (далее Продукт), которые обеспечивают работоспособность Продукта.

Руководство используется в следующих случаях:

- при установке или обновлении Продукта;
- при обновлении фабрики-потребителя.

Термины

Термин	**Значение**
Graceful Shutdown	Штатная остановка сервиса программным способом
Platform V	Набор программных продуктов Platform V, предоставляющих совокупность функциональных возможностей и позволяющий обеспечить быстрое конструирование информационных систем из множества готовых компонентов
SPAS	Сервер приложений сервиса авторизации
StandIn	Режим дублирования ключевых систем (англ. Stand-In, дублирующий)
Редакция	Отдельный расчетный алгоритм, который использует зафиксированный в версии набор переменных и выходных значений

Термин	Значение
Прикладная фабрика	Подсистема, которая выполняет обслуживание группы продуктов или автоматизацию группы функций продуктов
Продукт	Platform V Product Pricing

Сокращения

Сокращение	Значение
IP	Адрес устройства, работающего в компьютерной сети по протоколу IP
Json	JavaScript Object Notation. Формат предоставления данных
OTT	Platform V One-Token (OTT). Механизм выдачи одноразовых токенов
SLA	Соглашение о качестве и объеме предоставляемых услуг (англ. Service Level Agreement)
SI	Stand-In - режим дублирования ключевых систем

Сокращение	Значение
БД	База данных
ЕКПиТ	Единый Каталог Продуктов и Тарифов. Сервис для централизованного управления продуктовым рядом, пакетами услуг и тарифами Банка и дочерних компаний
СУБД	Система управления базами данных
УЦ	Удостоверяющий центр (сертификации). Англ. Certifying Authority / CA
ЭЦП	Электронная цифровая подпись

Системные требования

Продукт состоит из следующих модулей:

- Конструктор Правил (pricing-ide-war). Предоставляет пользовательский интерфейс для работы с правилами тарификации (создание / редактирование / публикация).
- Расчетный модуль (pricing-service-war). Производит расчет правил тарификации.

Предусловия

Перед установкой Продукта убедитесь, что выполнены следующие условия:

1. Обязательной установке подлежат:

- ОС Linux (рекомендован ОС «Альт 8 СП»)
- сервер приложений WildFly;

- имеется база данных продукта, а также пользователь и пароль к ней. Также необходим пользователь-владелец схемы, через него запускается обновление БД. Предполагается, что все действия по инициализации схемы данных пользователя были сделаны заранее;
- установлен веб-браузер (Яндекс.Браузер версии не ниже 19.10.1).

2. Опциональная установка/использование следующих компонентов, производится по усмотрению потребителя:

    - доступ на чтение к базе данных справочников (ЕКПиТ), а также пользователь и пароль к ней. Предполагается, что все действия по инициализации схемы данных пользователя были сделаны заранее, а таблицы в БД не являются пустыми;
    - Компонент Прикладной журнал (APLJ) продукта Platform V Data Tools;
    - Продукт Platform V One-Time-Token (OTT);
    - Компонент Аудит (AUDT) продукта Platform V Audit SE ;
    - Компонент Журналирование (LOGA) продукта Platform V Monitor;
    - Компонент Объединенный мониторинг (MONA) продукта Platform V Monitor;
    - Компонент Авторизация (AUTZ) продукта Platform V IAM SE;
    - Компонент IAM Proxy (AUTH) продукта Platform V IAM SE (реализует аутентификацию и авторизацию).

Аппаратные требования

В качестве клиентской части Продукта используется АРМ, реализуемое по технологии "тонкого" клиента (WEB-интерфейс). Минимальные требования к составу оборудования системного блока типового АРМ по WEB-технологии и их характеристики:

- процессор - Pentium III;
- объем оперативной памяти – 512 Мб;
- сетевая карта;
- видеокарта - SVGA 32 Мб с поддержкой видео режима с глубиной цвета не хуже HiColor (65536 цветов);
- жёсткий диск – 10 Гб;
- мониторы с разрешающей способностью не менее 1024x768 пикселей.

Программные требования

Требования к ПО:

- сервер приложений WildFly (лицензия LGPL2.1) 10.1.0.Final;
- клиент liquibase (нужен только на время развертывания);
- OpenJDK рекомендуется использовать версию 1.8

Функционирование 3-го поколения обеспечивается разворачиванием приложения на сервере приложений (WildFly). При разворачивании используются платформенные сервисы. В транспорте (ММТ) выполняется регистрация API, которые обслуживают вызовы, приходящие по ММТ.

Необходимые компоненты для корректной установки Продукта

Для корректной установки Продукта требуется наличие следующих компонентов – модулей в сервере приложений, от которых зависит Продукт:

```
<dependencies>
    <module name="org.dom4j"/>
    <module name="deployment.custodian-distr-impl-ear.ear" services="import" optional="true"/>
    <module name="deployment.dpl-lite-ear-${version}.ear" meta-inf="import" optional="true"/>
    <module name="deployment.logger-platform-${version}.war" optional="true"/>
    <module name="deployment.seap-lib-${version}.ear" meta-inf="import" services="import"
optional="true" />
</dependencies>
```

Seap_lip

Для того, чтобы выполнить установку Seap_lip:

1. Скопируйте файл seap-lib*.ear, расположенный в party дистрибутиве, в папке lib-overlay на сервер WildFly.
2. Выполните разархивацию.
   unzip –q seap-lib*.ear –d <your seap-lib dir>
3. В распакованной директории выдайте разрешение на исполнение к файлам с расширением sh.
   chmod +x ${SEAP LIB DIR}/*.sh
4. Убедитесь, что в окружении заведен env JBOSS HOME. В случае отсутствия добавьте, пример: export JBOSS HOME=/opt/wildfly.
5. Вызовите скрипт.
   create-overlay.sh
6. В результате чего произойдет добавление библиотек в wildfly.
7. Подключите зависимости в инсталляцию приложения.
   . jboss-cli.sh –connect –command=”deployment-overlay link –

name=seap-lib-<version>-- deployments=<имя инсталляции приложения, пример: pricing-service-war>

Подробное описание смотрите в документации WildFly.

Установка

Подготовка к развертыванию сервера приложений

Пререквизитом для установки Продукта является установка и настройка следующих продуктов/компонент, использование которых носит опциональный характер:

- Platform V Configuration;
- компонент Журналирование (LOGA) продукта Platform V Monitor;
- Platform V Audit SE.

В рамках установки **Продукта** (Расчетного модуля) использованы конфигурационные файлы, где прописаны все интеграции, дополнительных действий администратора не предполагается. Обязательные и необязательные параметры по установке указаны ниже по документу.

Настройка сервера приложений

Настройка сервера приложений для работы с БД модуля Конфигуратор

Чтобы Продукт загружал конфигурационные настройки системных модулей Платформы и свои бизнес параметры, в настройках сервера приложений (WildFly/bin/standalone.conf) пропишите параметры подключения к БД модуля **Конфигуратор**. При этом сам модуль **Конфигуратор** должен быть заранее установлен, выполнены все скрипты БД, входящие в состав дистрибутива модуля **Конфигуратор** (в том числе скрипты, создающие учетную запись пользователя для чтения конфигурационных параметров).

Название	**Значение (пример)**	**Описание**
config-store@jdbc.url	jdbc:postgresql:thin:@10.68.24.101:1521:FILTERS	URL для подключения к БД Конфигуратора

Название	Значение (пример)	Описание
config-store@jdbc.login	CONFIG_READER	Логин пользователя с правами на чтение схемы Конфигуратора
config-store@jdbc.password	CONFIG_READER	Пароль пользователя с правами на чтение схемы Конфигуратора
config-store@crypto.password	123456	Ключ для шифрования пароля. Обязательно должен совпадать со значением ключа crypto.password Конфигуратора (задается при установке конфигуратора)
node.id	sbt-oabs-356	Идентификатор сервера для межмодульного транспорта (должен быть

Название	Значение (пример)	Описание
		уникален в рамках всей инфраструктуры back-части платформы).

Создание источников данных

В конфигурационном файле `WildFly/standalone/configuration/standalone.xml` пропишите драйвер для подключения к БД, в секцию `<datasources>` для `<subsystem xmlns="urn:jboss:domain:datasources:4.0">`.

```
<drivers>
<driver name="postgresql" module="org.postgres">
<driver-class>org.postgresql.Driver</driver-class>
</driver>
</drivers>
```

Для источника данных Platform V Product Pricing в конфигурационный файл добавьте строки с параметрами, пример которых показан в коде ниже:

```
<xa-datasource jndi-name="java:/jdbc/pricing" pool-name="IFT_PricingDS" enabled="true" use-
ccm="true">
<xa-datasource-property name="URL">jdbc:postgresql://10.53.238.164:6543/dp?prepareThreshoId=0</xa-
datasource-property>
<xa-datasource-class>org.postgresql.xa.PGXADataSource</xa-datasource-class>
<driver>postgresql</driver>
<xa-pool>
<is-same-rm-override>false</is-same-rm-override>
<no-tx-separate-pools>true</no-tx-separate-pools>
</xa-pool>
<security>
<user-name>PRICING_SERVICES_APPL</user-name>
<password>PRICING_SERVICES_APPL</password>
</security>
```

Для источника данных ЕКПиТ, должен быть добавлен конфигурационный файл со следующими параметрами:

```
<xa-datasource jndi-name="java:/jdbc/ekpit" pool-name="ekpitDataBaseDS" enabled="true" use-
ccm="true">
<xa-datasource-property name="URL">
jdbc:dbserver:thin:@grid1655.ca.sbrf.ru:1521:db18
</xa-datasource-property>
<xa-datasource-class>postgresql.jdbc.xa.client.DatabaseXADataSource</xa-datasource-class>
<driver>postgresql</driver>
<xa-pool>
<is-same-rm-override>false</is-same-rm-override>
<no-tx-separate-pools>true</no-tx-separate-pools>
</xa-pool>
<security>
<user-name>EKPIT</user-name>
<password>EKPIT</password>
</security>
</xa-datasource>
```

В конфигурации приложения (platform-config.properties) укажите следующие параметры (если они отличаются от принятых по умолчанию):

Название	**Значение (пример)**	**Описание**
jndiNameEkpit	java:/jdbc/ekpit	JNDI-имя источника данных ЕКПиТ
ownerSchemaEkpit	EKPIT	Имя владельца схемы (если отличается от текущего пользователя)
jndiNamePricing	java:/jdbc/pricing	JNDI-имя источника данных Продукта
ownerSchemaPricing	PRICING_SERVICES	Имя владельца схемы, если отличается от текущего пользователя

Если предусмотрена работа в режиме StandIn, добавьте следующие параметры:

Название	Значение (пример)	Описание
jndiNameEkpitStandIn	java:/jdbc/ekpitStandIn	JNDI-имя источника данных ЕКПиТ
jndiNamePricingStandIn	java:/jdbc/pricingStandIn	JNDI-имя источника данных

Установка веб-приложений на сервер WildFly

Перед установкой приложений установите библиотеку seap-lib (см. стр. 5). Для корректной работы Продукта необходим overlay – seap-lib-8.1. Установка приложений представляет собой развертывание *.war-пакетов на сервер приложений:

- Конструктор Стратегий: pricing-ide-war-<номер_версии>.war;
- Расчетный Модуль: pricing-service-war-<номер_версии>.war.

Инструкции о способах развертывания war-пакетов можно найти в документации WildFly.

Конфигурация развернутых приложений

В комплекте поставки в директории configuration идут следующие файлы с определением конфигурационных параметров:

- pricing-ide-config-struct.xml: файл конфигурации **Конструктора Стратегий** Продукта;
- pricing-service-config-struct.xml: файл конфигурации Расчетного Модуля Продукта.

Таблица ниже содержит конфигурационные параметры Расчетного Модуля.

Имя параметра	Описание	Пример значения
third-party-default-timeout	Значение timeout по умолчанию (в секундах)	120 (default)
logAccumulatingEnabled	Включен ли механизм накопления сообщений в	true (default)

Имя параметра	Описание	Пример значения
	логгер с уровнем DEBUG	
jndiNameEkpit	Jndi-имя источника данных схемы ЕКПиТ	java:/jdbc/ekpit
ownerSchemaEkpit	Имя схемы владельца сущностей ЕКПиТ	EKPIT
jndiNameEkpitStandIn	Jndi-имя источника данных схемы ЕКПиТ SI	java:/jdbc/ekpitStandIn
jndiNamePricing	Jndi-имя источника данных схемы Продукта	java:/jdbc/pricing
ownerSchemaPricing	Имя схемы владельца сущностей Продукта	PRICING_SERVICES
jndiNamePricingStandIn	Jndi-имя источника данных схемы Продукта	java:/jdbc/pricing
useTrace	Признак включенного механизма трассировки	true
useMonitoringOnly	Признак включенного режима *только мониторинг*	false

Имя параметра	Описание	Пример значения
enableRuleSignatureCheck	Признак включенного режима проверки ЭЦП стратегий	false
pprbac.gf.shutdown.maxpoolsize	Graceful Shutdown:	50 (default)
pprbac.gf.shutdown.timeout	Graceful Shutdown:	40000 (default)
pj.disabled	Graceful Shutdown:	false (default)

Таблица ниже содержит конфигурационные параметры Конструктора Правил.

Имя параметра	Описание	Пример значения
rule-lock-timeout	Время удержания блокировки после последнего действия пользователя (в минутах). Если правило редактируется и пользователь не нажимает кнопку **Сохранить** в течение N минут (значение параметра), то другой пользователь	10

Имя параметра	Описание	Пример значения
	может забрать блокировку данного правила и, соответственно, сохранение правила первым пользователем не произойдет.	
third-party-default-timeout	Время ожидания ответа по умолчанию	120
`tpsEnabled"	Включена ли проверка производительности стратегий	false (default)
`jndiNameEkpit"	Jndi-имя источника данных схемы ЕКПиТ	pricing-ide@jndiNamePricing=java:/jdbc/pricing_ift
`ownerSchemaEkpit"	Имя схемы владельца сущностей ЕКПиТ	pricing-ide@jndiNamePricingStandIn=java:/jdbc/pricingStandIn
`jndiNameEkpitStandIn	Jndi-имя источника данных схемы ЕКПиТ SI	pricing-ide@jndiNameEkpitStandIn=java:/jdbc/ekpitStandIn

Имя параметра	**Описание**	**Пример значения**
`jndiNamePricing	Jndi-имя источника данных схемы Продукта	pricing-ide@jndiNameEkpit=java:/jdbc/ekpit_ift
`ownerSchemaPricing	Имя схемы владельца сущностей Продукта	pricing-ide@ownerSchemaPricing=PRICING_SERVICES
`jndiNamePricingStandIn"	Jndi-имя источника данных схемы Продукта SI	pricing-ide@ownerSchemaEkpit=EKPIT
`useTrace"	Признак включенной трассировки	`false` (default)
`pj.disabled	Признак отключенной отправки в ПЖ	`false` (default)
pprbac.gf.shutdown.maxpoolsize	Максимальный размер пула выключения всех API-серверов	`50`
pprbac.gf.shutdown.timeout	Таймаут в течении которого выполняется ожидание выключения всех API-северов	`40000`

Имя параметра	Описание	Пример значения
pj.disabled	Отправка в ПЖ отключена	`false` (default)
pjDataSetPayloadSize	Максимальный размер набора данных отправляемых в ПЖ	`102400`

Для работы с **Конструктором Стратегий Продукта** требуется иметь идентифицированных пользователей с набором ролей. Роль представляет собой набор прав (грантов) на выполнение определенных действий.

Список функциональных прав, используемых в **Конструкторе стратегий**:

NAME	DISPLAY NAME
calc_export	Экспорт расчетов
calc_import	Импорт расчетов для тестов
calc_import_limited	Импорт расчетов ограниченный
data_view	Просмотр
function_delete	Удаление функций
function_edit	Редактирование функций

NAME	DISPLAY NAME
rule_bulk_publish	Массовая публикация стратегий
rule_delete	Удаление стратегий
rule_edit	Редактирование стратегий
rule_publish	Публикация стратегий
rule_update	Актуализация стратегий
test_edit	Редактирование тестовых наборов данных
test_exec	Выполнение тестов
test_perfomance	Проверка производительности

Настройка пространства WildFly

Для того чтобы установить postgresql jdbc-driver и добавить источник данных:

1. Воспользуйтесь драйвером из дистрибутива ojdbc.jar: \sbt-spb-fsrv-01\OAVSP\Distrib\Servers\ojdbc;
2. Создайте директорию [WILDFLY_HOME]/modules/system/layers/base/com/postgresql/main/;
3. Скопируйте в созданную директорию ojdbc.jar;
4. Создайте файл module.xml в той же директории и добавьте:

```
5. <?xml version="1.0" encoding="UTF-8"?>
```

```
<module xmlns="urn:jboss:module:1.3" name="org.postqres">

 <resources>
    <resource-root path="ojdbc6.jar"/>
      </resources>
      <dependencies>
         <module name="javax.api"/>
         <module name="javax.transaction.api"/>
      </dependencies>
    </module>
```

16. В конфигурационном файле standalone/configuration/standalone.xml добавьте в тег следующее описание

```
    <driver name="postgresql" module="org.postqres">
       <driver-class>org.postgresql.Driver</driver-class>
    </driver>
```

20. Добавьте источник данных в после ExampleDS:

```
<xa-datasource jndi-name="java:/jdbc/pricingDS" pool-name="pricingDS" enabled="true" use-ccm="true">
        <xa-datasource-property name="URL">
             jdbc:postgresql:@10.68.24.101:1521:filtes
        </xa-datasource-property>
        <xa-datasource-class>org.postgresql.xa.PGXADataSource</xa-datasource-class>
        <driver>postgresql</driver>
           <xa-pool>
             <min-pool-size>1</min-pool-size>
             <initial-pool-size>1</initial-pool-size>
             <max-pool-size>10</max-pool-size>
             <is-same-rm-override>false</is-same-rm-override>
             <no-tx-separate-pools>true</no-tx-separate-pools>
        </xa-pool>
        <security>
             <user-name>SERVICE_PACKAGE</user-name>
             <password>SERVICE_PACKAGE</password>
        </security>
</xa-datasource>
```

Взаимодействие с СУДИР

Интеграция с СУДИР реализована на уровне back-части платформы (модуль Технологического Ядра).

Реализованные механизмы	**Способ реализации**
Реализован механизм управления доступом и обеспечено автоматическое предоставление типового набора прав доступа в соответствии с ролевой моделью	Реализовано на уровне back-части платформы (модуль Технологического Ядра)
Реализована максимальная длина, сложность и случайность значений сессионных идентификаторов	Реализовано на уровне back-части платформы (модуль Технологического Ядра)
Реализован контроль соответствия идентификатора сессии IP-адресу авторизованного пользователя	Реализовано на уровне back-части платформы (модуль Технологического Ядра)
Реализован механизм блокирования идентификатора сессии по истечении установленного времени не активности пользователя и при выходе пользователя из системы	Реализовано на уровне back-части платформы (модуль Технологического Ядра)
Реализовано периодическое подтверждение своих прав пользователем в АС путем ввода учетных данных (пароля), перед совершением критичных операций	Требование к подтверждению публикации повторным вводом логина/пароля отложено на дальнейшее развитие
Реализовано предоставление доступа в АС только в рамках роли, получаемой	В Системе для определения доступа пользователя будет

Реализованные механизмы	**Способ реализации**
АС из СУДИР по результатам аутентификации пользователя	использоваться только наличие привилегии в роли пользователя
Реализован функционал создания и управления учетными записями пользователей	Реализовано в СУДИР
Блокировка пользователей осуществляется путем перевода их в разряд заблокированных без возможности совершения операций под их именами	Реализовано в СУДИР
Реализована авторизация доступа и инструментов общего администрирования	Реализовано на уровне back-части платформы (модуль Технологического Ядра)
Реализован механизм контроля доступа, предоставляемый СУДИР	Реализовано на уровне back-части платформы (модуль Технологического Ядра)

Метрики

Реализован сбор метрик модуля **pricing-service**. Метрики pricing-service

Название метрики	**Идентификатор метрики**	**Описание**	**Разрезы фиксации**	**Возможные значения**
Доступность модуля	pricing-service.availability	Метрика контроля доступности модуля pricing-service	–	"1" - доступен "0" - модуль отключен
Доступность базы данных	Pricing (Main) pricing-service.health.db.pricing.main	Метрика контроля доступности базы данных для модуля pricing-service	–	"1" - БД доступна "0" - БД не доступна
Доступность базы данных Pricing (SI)	pricing-service.health.db.pricing.si	Метрика контроля доступности базы данных SI для модуля pricing-service	–	"1" - БД доступна "0" - БД не доступна
Доступность базы данных	pricing-service.health.db.ekpit.main	Метрика контроля доступности базы	–	"1" - БД доступна

Название метрики	**Идентификатор метрики**	**Описание**	**Разрезы фиксации**	**Возможные значения**
ЕКПиТ (Main)		данных для модуля pricing-service		"1" - БД доступна
Доступность базы данных ЕКПиТ (SI)	pricing-service.health.db.ekpit.si	Метрика контроля доступности базы данных для модуля pricing-service	–	"1" - БД доступна "1" - БД доступна
Доступность внешних API	pricing-service.health.api	Метрика контроля доступности внешних API для модуля pricing-service	className - Java класс API;methodName - Имя метода	"1" - API доступны "0" - API не доступны
Продукт: Выполнение стратегий	pricing-service.ruleExecution	Метрика выполнения стратегий Продукта - Время выполнен	terbank - Номер ТБ из запроса;opCode - Уникальный код расчета;spNa	Время в мс выполнения стратегии

Название метрики	Идентификатор метрики	Описание	Разрезы фиксации	Возможные значения
		ия одной стратегии (одного расчета)	me - Название АС инициатора	
Продукт: Вызовы функций	pricing-service.functionCall	Название функции, которая была вызвана	functionName - DSL название функции	Время в мс выполнения функции
Продукт: Количество ошибок выполнения стратегий	pricing-service.ruleErrors	Метрика количества ошибок выполнения стратегий Продукта	terbank - Номер ТБ из запроса; opCode - Уникальный код расчета; spName - Название АС инициатора	Количество ошибок (счетчик)

Задача тестирования инсталляции

Создание джоба jenkins по запуску smoke-тестов

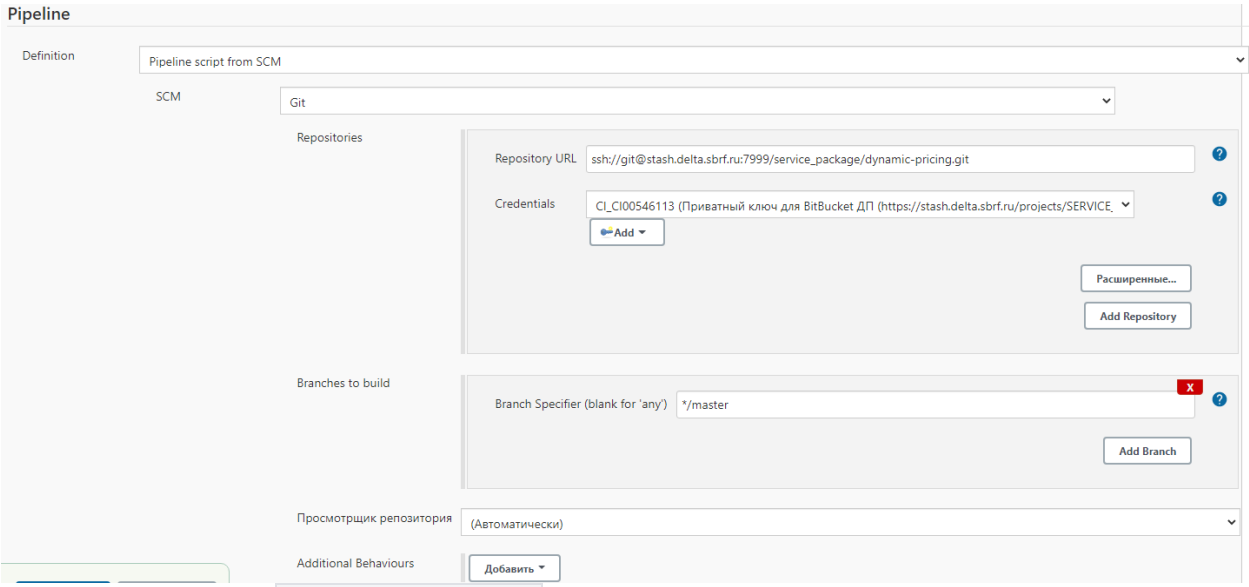
Создание джоба jenkins по запуску smoke-тестов

При установке необходимо использовать скрипт соответствующий версии дистрибутива, чтобы скрипт и дистрибутив находились в соответствии.

Для создания джоба jenkins:

1. В Jenkins создайте задачу типа "Pipeline".
2. В секции **Pipeline** в поле `Definition` выберите `Pipeline script from SCM`.

1. В поле SCM выберите Git.
2. Укажите адрес к репозиторию с сохраненной задачей, а также имя, где сохранена задача, также указываем имя Credentials в jenkins для доступа к этому репозиторию.



1. В поле Branches to build укажите ветку, в которой находится скрипт задачи.
2. В поле Script Path укажите путь до файла скрипта.
3. Сохраните изменения.

Описание скрипта запуска тестов (Jenkinsfile)

Для запуска smoke тестов нужно использовать созданную ранее задачу jenkins.

Для этого джоба нужно указать следующие параметры:

1. standType - имя стенда, для которого будут запущены тесты.
2. artifactUrl - ссылка на дистрибутив из Nexus.
3. Опциональное значение ветки репозитория, из которой будем брать параметры и секреты. Если значение не указано явным образом, будет использовано значение ветки по умолчанию (параметр secretAndParametersRepoBranch).

В процессе работы задачи будут выполнены следующие действия:

1. Запущены тесты для кластера и на geo-балансировщик с поддержкой OTT.
2. Запущены тесты для кластера и на geo-балансировщик без OTT.

Создание keystore и truststore OTT

Для доступа к серверу OTT понадобится сертификат, ключ и цепочка сертификатов.

Для выпуска сертификата OTT нужно выполнить следующие шаги:

1. Получить публичный сертификат.
2. Получить приватный сертификат. В качестве module.id указать значение: `pricing-service-cloud-test`.
3. После выполнения шагов должно быть получено два файла (имена могут отличаться от указанных ниже примеров):

   a. ott_service_truststore.p12 - публичная часть (truststore);

   b. pricing-service.p12 - приватная часть (keystore).
4. Переименовать файлы на `ott-truststore.p12` и `ott-keystore.p12`.
5. Сохранить файлы в репозитории в каталоге `test-resources` в директории с секретами `${secretConfigurationPath}`.

Создание keystore и truststore TLS

Корневой и промежуточный доверенные сертификаты выданы тестовым удостоверяющим центром банка. Для промышленных стендов удостоверяющий центр должен быть не тестовым. Поэтому часть файлов будет называться по-другому.

Для выпуска сертификата TLS:

1. Создайте конфигурацию запроса на сертификат.

```
[ req ]
default_bits       = 2048
distinguished_name = req_distinguished_name
req_extensions     = req_ext
prompt = no

[ req_distinguished_name ]
countryName                = RU
stateOrProvinceName        = MOW
localityName               = MOW
organizationName           = SBRF
```

```
commonName                  = pricing-service-cloud-test

[ req_ext ]
subjectAltName = @alt_names

[alt_names]
DNS.1   = pricing-service-cloud-test
```

2. Сформируйте request.csr.

```
openssl req -out request.csr -newkey rsa:2048 -nodes -keyout tls.key -config request.cnf
```

3. Запросите создание клиентского сертификата.
4. После получения ответа, скопируйте файл request.cer (подписанный сертификат в формате DER), и цепочку сертификатов, которым он подписан.
5. Преобразуйте полученный сертификат в формат PEM.

```
openssl x509 -inform DER -in request.cer -out tls.crt
```

6. Создайте хранилище keystore.

```
openssl pkcs12 -export -inkey tls.key -out keystore.p12 -in tls.crt
```

7. Преобразуйте keystore из формата PKCS12 в JKS.

```
keytool -importkeystore -destkeystore keystore.jks -srckeystore keystore.p12 -srcstoretype PKCS12
```

8. Измените пароль приватного ключа.

```
keytool -keypasswd -keystore keystore.jks -alias 1
```

9. Измените алиас сертификата.

```
keytool -changealias -alias 1 -destalias pricing-service-cloud-test -keystore keystore.jks
```

10. Преобразуйте полученный корневой сертификат в формат PEM.

```
openssl x509 -inform DER -in "Test Root CA 2.cer" -out "Test Root CA 2.crt"
```

11. Преобразуйте полученный промежуточный сертификат в формат PEM.

```
openssl x509 -inform DER -in "Sberbank Test Issuing CA 2.cer" -out "Sberbank Test Issuing CA 2.crt"
```

12. Создайте хранилище truststore.jks и добавить в него доверенный корневой сертификат.

```
keytool -import -keystore truststore.jks -file "Test Root CA 2.crt" -alias root-ca -noprompt
```

13. Добавьте в truststore.jks промежуточный доверенный сертификат.

```
keytool -import -keystore truststore.jks -file "Sberbank Test Issuing CA 2.crt" -alias chain-ca -noprompt
```

14. Сохраните файлы в репозитории в каталоге test-resources в директории с секретами ${secretConfigurationPath}`.

Получаем два файла:

- keystore.jks - приватный ключ и сертификат
- truststore.jks - доверенные сертификаты

Обновление файла passwords.properties паролями от хранилищ и ключей

В файле хранятся пароли к БД для запуска liquibase, а также пароли к хранилищам сертификатов и ключей для запуска джоба тестирования. Файл шифруется тем же способом, и с тем же паролем, что и секреты для среды контейнеризации Kubernetes или Red Hat OpenShift 4+ с помощью ansible-vault. Файл `passwords.properties` следует разместить в том же каталоге, в котором находятся секреты для среды контейнеризации Kubernetes или Red Hat OpenShift 4+ в зашифрованном виде — `${secretConfigurationPath}`.

В файл passwords.properties нужно добавить пароли для созданных хранилищ и ключей:

```
...
ott.certstore.pwd=111111
ott.trust.store.pwd=111111
ott.certstore.private.key.pwd=111111
transport.rest.ssl.keystore.password=2222222
transport.rest.ssl.private.key.password=111111
transport.rest.ssl.truststore.password=111111
```

Описание добавляемых параметров:

- ott.certstore.pwd - пароль для доступа к хранилищу сертификатов OTT.
- ott.trust.store.pwd - пароль доступа к хранилищу доверенных сертификатов OTT.
- ott.certstore.private.key.pwd - пароль доступа к приватному ключу OTT
- transport.rest.ssl.keystore.password - пароль для доступа к хранилищу сертификатов.
- transport.rest.ssl.private.key.password - пароль доступа к приватному ключу.

Обновление

Обновление осуществляется так же, как первоначальная установка: запускается задача установки с параметром updateMode = fullInstall. В процедуру установки включен процесс обновления базы данных с помощью утилиты liquibase. Установку новой версии продукта необходимо осуществлять согласно описанным шагам в разделе **Установка** текущего документа. Дополнительных настроек не требуется.

Проверка работоспособности установки

Для проверки корректности установки, проверьте, что задача установки завершилась со статусом SUCCESS (успешно). Проверка работоспособности осуществляется путем:

- unit теста и проверки доступности базы данных;
- выполнение автотестов и smart регрессов.

Откат

Автоматической процедуры отката нет.

Для отката необходимо выполнить установку предыдущей стабильной версии в соответствии с инструкцией, раздел **Установка** текущего документа.

Часто встречающиеся проблемы и пути их устранения

В данном разделе приведены наиболее частые проблемы и описаны пути их устранения.

Проблема	Причина	Способ устранения
Загрузка конфигурационных элементов	Некорректная загрузка	Просмотреть логи на наличие ошибок
При накате скриптов Liquibase	–	Обратиться в команду разработки
При деплое Продукта	–	Обратиться к администраторам Wildfly

Чек-лист валидации установки

После установки необходимо проверить:

- для проекта в WildFly создались объекты приложений.
- логи на отсутствие ошибок, логи можно наблюдать в централизованной системе управления.