Инструкция по развертыванию экземпляра клиринговой системы товарного рынка АО СПВБ.

Оглавление

1. **Состав комплекса.**

Комплекс разворачивается на виртуальных или реальных машинах с предустановленной операционной системой AstraLinux, количество машин значения не имеет. Демонстрационный стенд развернут на 4х машинах, на которых развернуто:

I)
   a. RabbitMQ
   b. Redis
   c. PostgreSQL

II) FreeIPA
III) MinIO
IV) Программное обеспечение КСТР

Для развертывания сервисных подсистем RabbitMQ, Redis, PostgreSQL, FreeIPA, MinIO в документации AstraLinux существует подробное руководство. Для тестового стенда достаточно настроить один экземпляр каждого сервиса по бескластерной схеме.
Для установки инициализационных данных прилагается набор необходимых дампов. Инструкция по восстановлению данных из дампов описана в прилагаемом документе «***Резервное копирование сервисов Торгового Клиринга***»

2. **Настройка сервисных подсистем и развертывание инициализационных данных.**

2.1 Настройка RabbitMQ

В приложении Администратора требуется создать виртуальный хост, к которому будут обращаться программные модули приложения, и пользователя для доступа модулей к хосту.

Пользователь должен быть настроен для доступа к виртуальному хосту RabbitMQ.

Тестовый стенд имеет следующие настройки для обслуживания приложения:

Virtual Host=microTest
User=microTest
Password=microTest

Никаких данных для функционирования больше не требуется, каналы и очереди будут организованы динамически модулями приложения после его запуска.

2.2 Настройка Redis

После установки, требуется сохранить ***пароль*** для доступа к базе данных сервиса, который потом надо будет применить к переменной окружения RedisConnections. Для тестовых нужд никаких данных в базу данных этого сервиса загружать не надо, приложение при необходимости загрузит нужные данные динамически.

2.3 Настройка PostgreSQL

После установки, требуется создать пользователя и его пароль для доступа к данным. Для тестового стенда, достаточно использовать существующего пользователя (postgres pwd postgres), и создать две базы данных:

- TCSClearingEntitiesN
- TCSDepositEntitiesN

Для функционирования программного обеспечения, требуется инициализация данных каждой БД, **дамп прилагается**.

2.4 Настройка FreeIPA

После установки сервиса, требуется загрузка инициализационных данных, **дамп прилагается**.
Также требуется настройка имени и логина пользователя с правами Admin.

2.5 Настройка MinIO

После установки сервиса, требуется создать AccessToken для записи и чтения. Этот токен будет применен в конфигурировании комлекса программного обеспечения для модулей, которые требуют доступа к подсистеме хранения файлов. После установки сервиса, требуется загрузка инициализационных данных, **дамп прилагается**.

3. Установка и настройка модулей приложения.

3.1 Конфигурирование окружения сборки

Установка комплекса осуществляется из репозитория исходного кода, размещенного в локальном репозитории GitLab. Для получения доступа к репозиторию, требуется запросить Access Token у Администратора проекта (см. Контакты) и адрес репозирония гита.

Кроме этого, на машине, где предполагается сборка исполняемых модулей, должны быть установлены следующие программные продукты:

- git client (> sudo apt-get install git)
- NodeJS >= v.16.19 (>sudo apt install nodejs)
- Angular 14 dev kit (>sudo apt install npm)
  : npm > v9.6
  : cli > v.14.2
- dotnet v.3.1.426 (https://wiki.astralinux.ru/kb/net-core-191108093.html)

Также требуется настроенный доступ от сборочной машины к сети интернет для доступа к библиотекам.

3.2 Сборка исполняемых модулей приложения

3.2.1 Получение исходного кода

Выполнить следующую команду:
> git clone -b master https://oauth:<GIT ACCESS TOKEN>@<GIT DIRECTORY>/tvr/tcs.git <BUILD DIRECTORY>

<GIT ACCESS TOKEN> - получить у Администратора проекта

<GIT DIRECTORY> - получить у Администратора проекта

<BUILD DIRECTORY> - локальная папка, в которую будет производиться загрузка исходного кода.

3.2.2 Сборка приложения

Сборка приложений осуществляется для следующих модулей:

3.2.2.1 пользовательское WEB-приложение

1. Перейти в папку <BUILD DIRECTORY>/TCS-front

2. Выполнить следующие команды:

```
> npm i
> ng build app-clearing –output-path <FRONT DEPLOY BUILD>
```

<FRONT DEPLOY BUILD> - папка, куда будет выгружены исполняемые модули пользовательского WEB-приложения

3.2.2.2 RestAPI

Выполнить для каждого** <RestAPI module> **из списка:

RestAPI module	***Module Name***
TCSAuth	*TCSAuth*
TCSClearing	*TCSClearing*
TCSCommunicate	*TCSCommunicate*
TCSMailing	*TCSMailing*
TCSServices	*TCSServices*

1. Перейти в папку <BUILD DIRECTORY>/TCS-back/src/<RestAPI module>
2. Выполнить команду

   > dotnet publish <BUILD DIRECTORY>/TCS-back/src/<RestAPI module> -o <TARGET DIR>/<RestAPI module>-c TestDebug /p:EnvironmentName=DebugTest

   <TARGET DIR> - папка, предназначенная для формирования файлов исполняемых модулей.

3.2.2.3 Микросервисы

Выполнить для каждого** <MicroSvc module> **из списка:

MicroSvc module	***Module Name***
Microservice.Hosts/Host.Store.Actors	*Host.Store.Actors*
Microservice.Hosts/Host.Services.Audit	*Host.Services.Audit*
Microservice.Hosts/Host.Services.ScheduledTasks	*Host.Services.ScheduledTasks*
Microservice.Hosts/Host.Services.Reporting	*Host.Services.Reporting*
Microservice.Hosts/Host.Services.Deposit	*Host.Services.Deposit*
Host.Services.Clearing	*Host.Services.Clearing*
Microservice.Hosts/Host.Services.Mailing	*Host.Services.Mailing*
Microservice.Hosts/Host.Services.Business	*Host.Services.Business*

1. Перейти в папку <BUILD DIRECTORY>/TCS-back/src/<MicroSvc module>
2. Выполнить команду

   > dotnet publish <BUILD DIRECTORY>/TCS-back/src/<RestAPI module> -o <TARGET DIR>/<RestAPI module>-c TestDebug /p:EnvironmentName=DebugTest

<TARGET DIR> - папка, предназначенная для формирования файлов исполняемых модулей.

3.2 Конфигурирование приложения

3.2.1 Конфигурирование файла hosts

Для простоты настройки, конфигурационные файлы приложения ссылаются на псевдонимы адресов нод, которые настраиваются в файле /etc/hosts:

xxx.xxx.xxx.xx **infraRmq**
yyy.yyy.yyy.yy **infraRedis**
zzz.zzz.zzz.zz **infraPg**
mmm.mmm.mmm.mm **ldap**
nnn.nnn.nnn.nn **minio**

Если для RabbitMQ и PostgreSQL ранее были определены настройки доступа по умолчанию, как описано в п. 2.1 и 2.3, то не требуется изменения конфигураций.

Для настройки доступа к подсистемам инфраструктуры, требуется изменить файлыс именем appsettings.TestDebug.json для каждого построенного модуля подсистем из **3.2.2.2** и **3.2.2.3.**

Настраиваются те секции, которые уже определены в файлах, следующим образом:

```
  "RabbitMQ": {
    "IsStrongTypeNames": false,
    "IncludeTypeDescription": false,
    "ServerNameMQ": "infraRmq",
    "VirtualHostMQ": "microTest", //настроено в 2.1
    "UserNameMQ": " microTest",//настроено в 2.1
    "PasswordMQ": " microTest",//настроено в 2.1
    "RESPONSE_TIMEOUT": 15000  }
```

```
  "RedisConnections": {    "h1": "infraRedis:6379,password=<пароль из 2.2>,ssl=False,abortConnect=False,name=testN"  }
/*настройка в 2.2*/
```

```
  "ConnectionStrings": {
"ClearingEntitiesContext": "Host=infraPg;Port=5432;Database=TCSClearingEntitiesN;Username=postgres;Password=postgres",
"DepositDataContext": "Host=infraPg;Port=5432;Database=TCSDepositDataN;Username=postgres;Password=postgres"
}/*настройка в 2.3*/
```

```
  "LDAPOptions": {
    "Host": "ldap",
    "BindAdmin": "<путь к логину админа из п. 2.4>",
    "BindPassword": "<пароль админа из п. 2.4>",
    "CnBase": "cn=users,cn=accounts",
    "DnBase": "dc=test,dc=spvb",/*здесь приведено значение для тестового стенда, зависит от полученного дампа и настройки ipa*/
    "CnPermission": "cn=roles",
    "CnOrganizations": "cn=organizations,cn=groups"
  },
```

```
  "MinIO": "s3://<AccessToken из 2.5>@minio:9000",
  "StaticBaseUri": "http://minio:9000/templates/",
```

3.3 Конфигурирование доступа к приложению

3.3.1 Конфигурирование окружения исполняемых модулей

На хостах, предназначенном для размещения и функционирования исполняемых модулей RestAPI и микросервисов, должно быть установлено следующее ПО:

dotnet 3.1.426

Дополнительно, на хостах, предназначенных для RestAPI, требуется настройка роутинга на порты RestAPI - в тестовом стенде, для этого используется:

Nginx

Также допустимо устанавливать исполняемые модули RestAPI и микросервисов на один хост, как это сделано в тестовом стенде, при условии достаточности вычислительного ресурса.

3.3.2 Описание конфигурации RestAPI модулей

Для каждого модуля RestAPI в файле конфигурации appsettings.TestDebug.json существует секция **Kestrel**, который описывает адреса конечных точек приложения и их порты. Соответственно, эти порты должны быть открыты и сконфигурированы для доступа к ним извне, с тестировочной пользовательской машины. Также, должен быть настроен виртуальный роутинг на эти порты, к примеру через nginx. Пример настройки nginx тестового стенда – в приложении *Файл /etc/nginx/sites-available*

Важно понимать, что требуется настроить секции путей к портам RestAPI, разрешить все методы и открыть CORS для адреса, по которому будет расположено пользовательское WEB-приложение.

Секции RestAPI (пример из тестового стенда):

RestAPI модуль	Виртуальный путь (location)	Порт (proxy_path) должен соответствовать настройке Kestrel
TCSServices	/services/	http://127.0.0.1:5250
TCSClearing	/clearing/	http://127.0.0.1:5650
TCSAuth	/auth/	http://127.0.0.1:5050
TCSCommunicate	/signalr	http://127.0.0.1:5450
TCSMailing	Не определяется	Любой свободный

3.3.3 Установка и конфигурации WEB-приложения

Для настройки WEB-приложения надо определить папку, которая будет настроена для доступа к статическим файлам извне (с тестировочной пользовательской машины) и определить роутинг к порту 80, например:

```
location /{
#папка, предназначенная для установки WEBприложения,
# на тестовом стенде это /var/www/TCSFrontClearing
   root <WEB APP DIR>;
   index index.html index.htm index.nginx-debian.html;
      try_files $uri $uri/ =404;
  }
```

После выполнения п. 3.2.2.1, построенные файлы из папки <FRONT DEPLOY BUILD> требуется скопировать в папку <WEB APP DIR>, и убедиться что в файле <WEB APP DIR>/assets/config.json секция **hosts** сконфигурирована в соответствии с настройкой путей nginx.

3.3.4 Установка сервисов RestAPI и микросервисов.

Исполняемые на сервере модули могут функционировать как в контейнерах, так и в виде сервисов-демонов systemd. На тестовом стенде сервисы монтируются в виде демонов. Для функционирования, надо убедиться, что существует пользователь www-data и дать ему права на доступ чтения и выполнения к папкам, предназначенным для каждого модуля

Для каждой папки <TARGET DIR>, в которые были построены модули п. 3.2.2.2 и 3.2.2.3, требуется выполнить следующее:

1. Скопировать все файлы в папку, предназначенную для соответствующего модуля (или оставить в том месте куда было построено) - <MODULE WORKING DIR>.
2. Настроить сервис system в папке /etc/system/system/<имя модуля>.service
    a. User=www-data
    b. WorkingDirectory=<MODULE WORKING DIR>
    c. ExecStart=/usr/bin/dotnet <MODULE WORKING DIR>/<Module Name>.dll
3. Выполнить команду установки сервиса:

```
> sudo systemctl enable <имя модуля>.service
```

3.5 Запуск приложения

Условия запуска:

- доступность и функционирование Redis, RabbitMQ, PostgreSQL, FreeIPA, MinIO
- должны быть запущены все сервисы приложения, желательно в следующей последовательности:

TCSAuth
TCSMailing
TCSServices
TCSClearing
TCSCommunicate
Host.Services.Mailing
Host.Services.Audit
Host.Services.Clearing
Host.Services.Deposit
Host.Services.Business
Host.Store.Actors
Host.Services.Reporting
Host.Services.ScheduledTasks

Для каждого сервиса, описанного в 3.3.4, выполнить команду:

```
> sudo systemctl start <имя модуля>.service
```

3.6 Останов приложения

Останавливать модули приложения желательно в обратной описанной в 3.5 последовательности, для каждого сервиса выполняя команды

```
> sudo systemctl stop <имя модуля>.service
```

4. Проверка функционирования

Если всё настроено верно, то в админском приложении RabbitMQ для виртуального хоста, определенного при конфигурировании 2.1, приложение сгенерирует набор очередей и обменников очередей.

5. Взаимодействие с WEB-приложением.

В зависимости от установленных дампов, Администратор проекта вручает пользовательский логин и пароль для доступа к приложению через браузер с пользовательской рабочей станции. Приложение доступно для функционирования с браузерами Chrome, Yandex, FireFox, Chromium с версиями, изданными не ранее 2023 года.

6. Приложения и ссылки

6.1 Контакты

Полномочия	**Фио**	**Емейл**
Администратор проекта	Кречмар Е.А.	krechmarEA@spvb.ru
Архитектор проекта	Осипов Ю.В.	osipovYuV@spvb.ru
Тестировщик проекта	Климова Д.А.	klimovaDA@spvb.ru

6.2 Список указанных файлов

- Резервное копирование сервисов Торгового Клиринга.doc

- hosts (/etc/hosts)

- Файл /etc/nginx/sites-available