整体环境

- 172.27.3.41 CentOS Linux 7 (Core)
- 172.27.3.42 Windows Server 2012 R2 Datacenter

IP	服务名	Port	提供服务	依赖
172.27.3.41	redis	6379	缓存 NoSQL	
172.27.3.41	MySQL	3306	数据库服务	
172.27.3.41	RabbitMQ	5672	MQ 消息队列	
172.27.3.42	exchange	8733	rcp 服务	redis、MySQL(engine)、RabbitMQ
		2012	ws 服务	
172.27.3.42	engine_handle		消费MQ 写入数据到MySQL	MySQL(engine)、RabbitMQ
172.27.3.41	exchange proxy	9393	Node.js代理 ws服务	exchane(2012)
172.27.3.41	exchange proxy rest	9400/3000	Node.js代理 rest 服务	exchane(8733)
172.27.3.41	backend	8000/ 443	Python 提供 HTTP服务	redis、MySQL(maneki)、exchange(8733)、RabbitMQ(engine)

- exchange admin 集成在 backend 中

基础服务

docker 安装

内核64位必须不小于3.10

```
uname -r
#使用root权限登陆系统。
#更新系统包到最新。
yum -y update

sudo yum install -y yum-utils \
  device-mapper-persistent-data \
  lvm2
```

```

sudo yum-config-manager \
    --add-repo \
    https://download.docker.com/linux/centos/docker-ce.repo

sudo yum install docker-ce

sudo systemctl enable docker.service
sudo systemctl start docker
docker info
```

Docker Compose 安装

```
sudo curl -L https://github.com/docker/compose/releases/download/1.18.0/docker-compose-`uname -s`-`uname -m` -o /usr/local/bin/docker-compose
sudo chmod +x /usr/local/bin/docker-compose
docker-compose --version
```

相关命令

```
docker-compose stop 容器id      #停止服务
docker-compose up -d            #启动
docker-compose ps               #查看运行状况
docker-compose kill             #杀容器
docker-compose logs -f 容器id   #查看日志
docker-compose down -v          #关闭容器
docker-compose build            # 构建镜像
docker-compose up              # 编排启动容器 -d 以守护模式启动
docker-compose start            # 启动容器
docker-compose stop             # 停止容器
docker-compose restart          #重启相关容器
```

redis

```
docker run --name my-redis -p 6379:6379 -v /opt/redis/data:/data -d redis redis-server --appendonly yes
```

MySQL 服务

```
docker run --name my-mysql -p 3306:3306 -v /opt/mysql/conf:/etc/mysql/conf.d -v /opt/mysql/data:/var/lib/mysql -v /opt/mysql/logs:/logs -e MYSQL_ROOT_PASSWORD=123456 -d mysql:5 --character-set-server=utf8mb4 --collation-server=utf8mb4_unicode_ci
# 进入 docker
docker exec -it my-mysql /bin/bash
```

RabbitMQ Cluster

```
cd /opt
git clone https://github.com/bijukunjummen/docker-rabbitmq-cluster
cd docker-rabbitmq-cluster/
./build-images.sh
cd /opt/docker-rabbitmq-cluster/cluster
# docker-compose.yml 中可以修改 用户名密码 myuser mypass
docker-compose build #构建基础镜像
docker-compose up -d #启动

```

安装成功后，登录管理界面http://172.27.3.41:15672/#/vhosts，默认用户名myuser，密码mypass

Add a new virtual host

Name `engine`

默认端口记录

4369 – erlang发现口

5672 –client端通信口

15672 – 管理界面ui端口

25672 – server间内部通信口

exchange

Windows 环境

安装 MySQL 驱动及.Net Framework 4.5.1

- mysql-connector-net-6.9.11.exe
- NDP451-KB2858728-x86-x64-AllOS-ENU.exe

数据库初始化

Navicat Premium 工具连接 MySQL

root 123456

mysql创建引擎库engine，字符集 `utf-8`

```
CREATE DATABASE `engine`
```

运行SQL文件 `engine_futures.sql` 初始化数据

修改项目配置

`exchange` 项目配置文件 `MyExchange.exe.config.xml`

```
<appSettings>
    <add key="ConnectionIp" value="172.27.3.41" />
    <add key="ConnectionPortNum" value="6379" />
    <add key="User" value="myuser"/>
    <add key="PassWord" value="mypass"/>
    <add key="MqIp" value="172.27.3.41"/>
    <add key="MqPort" value="5672"/>
    <add key="ApiIP" value=""/>
    <add key="BotAccount" value="987961,btcc-574205" />
    <add key="IsOpenFreeze" value="1" />
  </appSettings>
<applicationSettings>
    <MyExchange.Properties.Settings>
      <setting name="Withdraw_API_Token" serializeAs="String">
        <value>Btc_chin@_Engine_exg_2018</value>
      </setting>
      <setting name="Redis_IP_Config" serializeAs="String">
        <value>172.27.3.41</value>
      </setting>
    </MyExchange.Properties.Settings>
</applicationSettings>
```

ConnectionIp redis的ip

ConnectionPortNum是redis的端口

User和PassWord是 MQ 的用户名和密码，MqIp和MqPort是 MQ 的ip和端口

```
<add name="MyExgEntities"  connectionString="server=172.27.3.41;Port=3306;Character Set=utf8;User Id=root;password=123456;Persist Security Info=True;database=engine;default command timeout=300"
providerName="MySql.Data.MySqlClient" />
```

connectionStrings.connectionString 中修改为MySQL服务具体ip、端口、用户、密码、数据库名称。

启动 exchange

双击 `exchange/MyExchange.exe`，点击 Worker 中的 `Start` 按钮 DatabaseService、BitcoinPriceService、WorkerService、ConnectionService、LiveService的状态为Started，即启动成功。

Service	Port
Websocket Port	2012
Rest Port	8733

engine_handle

环境同 exchange

消费 RabbitMQ 写入数据到 MySQL

修改配置文件

appsettings.json

```
"SettingRoute": "Test",
"Test": {
    "ConnectionStrings": {
      "MySqlContext": "server=172.27.3.41;Port=3306;Character Set=utf8;User Id=root;password=123456;Persist Security Info=True;database=engine;default command timeout=300"
    },
    "RabbitMqSettings": {
      "Host": "172.27.3.41",
      "Username": "myuser",
      "Password": "mypass",
      "Port": "5672",
      "VirtualHost": "engine"
    }
}
```

启动 engine handle

管理员运行命令行cmd

```
# binPath根据实际情况修改
sc create EngineHandle binPath= "d:\btcc\engine_handle\Engine_Handle.exe"
sc start EngineHandle
```

backend

交易引擎服务器 exchange-server

交易所项目

核心入口: `exchange-server/maneki/apps`, 业务代码都在这里

```
├── maneki
|   ├── __init__.py
    ├── apps  / 所有业务单元
    |   ├── __init__.py
    |   ├── api_key       / API-key相关
    |   ├── constants.py
    |   ├── engine        / 与交易引擎通信
    |   ├── market        / 市场相关
    |   ├── transaction   / 交易相关
    |   ├── user_assets   / 用户资产
    |   ├── user_auth     / 用户登录授权
    |   └── user_kyc      / 用户KYC审核
```

环境需求

CentOS 7 下安装 docker和docker_compose

数据库初始化

Navicat Premium 工具连接 MySQL

root 123456

mysql创建引擎库 `maneki` 字符集 `utf-8`

```
CREATE DATABASE `maneki`
```

运行SQL文件 `maneki.sql` 初始化数据

修改配置文件

```
tar zxf backend.tar.gz
cd exchange-server
vi .env
```

```
# 修改引擎的ip
ENGINE_URL=http://172.27.3.42:8733/secure/rpc

# 修改mysql配置
DATABASE_URL=mysql://root:123456@172.27.3.41:3306/maneki

# 保持所有的 ENGINE_KEY 一样
ENGINE_KEY=Btc_chin@_Engine_exg_2018

# 修改redis
REDIS_URL_SERVER=redis://172.27.3.41:6379
REDIS_URL_COMMON=redis://172.27.3.41:6379
CELERY_BROKER_URL=redis://172.27.3.41:6379/11

# 修改mq
RABBITMQ_URL=amqp://myuser:mypass@172.27.3.41:5672/engine
```

构建容器并启动

```
make prod.api.all CMD="build" && make prod.api.all CMD="up -d" && make prod.worker.all CMD="build" && make prod.worker.all CMD="up -d"

make prod.admin CMD="down --remove-orphans"  && make prod.api.all CMD="build" && make prod.api.all CMD="up -d" && make prod.worker.all CMD="build" && make prod.worker.all CMD="up -d"
```

启动成功端口：8000

访问：`http://172.27.3.41:8000/api/v1/auth/login/email/`

exchange proxy 安装

交易引擎 ws 代理

Node.js 提供的 websocket 服务代理

需要和 `exchange-server` 使用相同的 `redis` 服务

修改配置文件

通过工具 Xftp 5 复制文件 `exchange-proxy-master.zip` 到 `/opt/src`

```
cd /opt/src
unzip exchange-proxy-master.zip
cd exchange-proxy-master
cp .env.example .env
vim .env
```

```
BACKEND_REDIS=redis://172.27.3.41:6379/1
ENGINE_WS=ws://172.27.3.42:2012
PORT=9393
DEBUG=ip,time
CONNS=10
RECONNECT_INTERVAL=10000
RATE_LIMIT_REDIS=redis://172.27.3.41:6379/1
RATE_LIMIT=600
RATE_LIMIT_INTERVAL=60
SENTRY_DSN=
NODE_ENV=production
TRAEFIK_DOMAIN=ws.freesnow.com
TRADES_REDIS=redis://172.27.3.41:6379/1
# 当前服务器IP
ALLOWED_ORIGIN=172.27.3.41
IS_API_SERVER=true
CRID_REDIS=redis://172.27.3.41:6379/11
CRID_EXPIRE=86400
LOG_MONGO=mongodb://172.19.0.42:27017/engine_log
TIMESTAMP_EXPIRE=3600000
IS_LOCAL=true
TZ=UTC
FOR_API_USER=false
```

构建容器并启动

```
docker-compose build
docker-compose up -d
```

exchange proxy rest

Node.js 提供交易引擎 rest 代理

修改配置文件

将 `exchange-proxy-rest-master.zip` 拷贝到 `/opt/src`

```
cd /opt/src
unzip exchange-proxy-rest-master.zip
cd exchange-proxy-rest-master
cp .env.example .env
vim .env
```

```
ENGINE_RPC_URL= http://172.27.3.42:8733/secure/rpc
ENGINE_KEY=Btc_chin@_Engine_exg_2018
LOG_MONGO=mongodb://mongo:27017/proxyLogs
ENGINE_RPC_TIMEOUT=3000
CRID_REDIS=redis://172.27.3.41:6379/11
SIGN_REDIS=redis://172.27.3.41:6379
MYSQL_HOST=172.27.3.41
MYSQL_USER=root
MYSQL_PASSWORD=123456
MYSQL_DATABASE=engine
CRID_EXPIRE=86400
```

构建容器并启动

```
docker-compose build
docker-compose up -d
```

附加

Node.js & yarn

```
# nodejs 自带 npm
curl --silent --location https://rpm.nodesource.com/setup_8.x | sudo bash -
sudo yum -y install nodejs
# yarn
curl --silent --location https://dl.yarnpkg.com/rpm/yarn.repo | sudo tee
/etc/yum.repos.d/yarn.repo
sudo yum -y install yarn
yarn --version
```

注意 `ENGINE_KEY=Btc_chin@_Engine_exg_2018` 所有地方一样

- backend
- exchange proxy rest

app 配置

```
{
  "ENV": "dev",
  "BACKEND_URL": "http://172.27.3.41:8000",
  "CHART_URL": "https://chart.freesnow.com",
  "EXCHANGE_API_URL": "http://172.27.3.41:3000",
  "SOCKET_URL": "ws://172.27.3.41:9393",
  "PRI_SOCKET_URL": "ws://172.27.3.41:9393",

  "IX_BACKEND_URL": "http://119.28.142.152:10110",
  "FUND_SOCKET_URL": "ws://119.28.142.152:10110/ws_book"
}
```