**Hyperledger Explorer Setup**

[konda.kalyan@gmail.com](mailto:konda.kalyan@gmail.com)

Reference: <https://github.com/hyperledger/blockchain-explorer>

### First, setup YOUR Fabric Network and get it up and running

#### Setup Fabric network

1. **Clone YOUR fabric repository (in this example, I am referring to my below network)**

*git clone* [*https://github.com/konda-kalyan/HLF-Multi-Host-Docker-Swarm*](https://github.com/konda-kalyan/HLF-Multi-Host-Docker-Swarm)

*cd HLF-Multi-Host-Docker-Swarm*

*git checkout* ***HL-Explorer-Setup***

1. **IMPORTANT STEP:** Make sure that **‘CORE\_PEER\_GOSSIP\_BOOTSTRAP’** and **CORE\_PEER\_GOSSIP\_EXTERNALENDPOINT’** are properly set on docker-compose files for all 3 Peers.

Example, for peer0.org1.exampe.com

CORE\_PEER\_GOSSIP\_BOOTSTRAP=peer**1**.org1.example.com:**8**051

CORE\_PEER\_GOSSIP\_EXTERNALENDPOINT=peer**0**.org1.example.com:**7**051

Likewise, for peer1.org0.example.com

CORE\_PEER\_GOSSIP\_BOOTSTRAP=peer**0**.org1.example.com:**7**051

CORE\_PEER\_GOSSIP\_EXTERNALENDPOINT=peer**1**.org1.example.com:**7**051

Repeat this step for **ALL 3 Orgs**

1. **Clone ‘HLF-Chaincode-Java-CouchDB-Rich-Queries’** (which is used as reference in ‘*~/HLF-Multi-Host-Docker-Swarm/chaincodes*’)

*cd ~/HLF-Multi-Host-Docker-Swarm/chaincodes*

*rm -rf HLF-Chaincode-Java-CouchDB-Rich-Queries*

*git clone https://github.com/konda-kalyan/HLF-Chaincode-Java-CouchDB-Rich-Queries*

1. **~~In HL Explorer repository, update Fabric network references~~**

~~In this case, ‘HLF-Multi-Host-Docker-Swarm’ is repository and ‘hlf\_multi\_host\_network’ is network name.~~

~~Update~~ *~~‘/blockchain-explorer/app/platform/fabric/config.json~~*~~’ with fabric network details~~

*~~vim ~/blockchain-explorer/app/platform/fabric/config.json~~*

~~Replace ‘first-network’ with ‘hlf\_multi\_host\_network’~~

#### Configure Operations Service Environment Variables per each Peer and Orderer

Reference: <https://github.com/hyperledger/blockchain-explorer/blob/master/CONFIG-OPERATIONS-SERVICE-HLEXPLORER.md>

##### Per each Orderer

For **Org1** Orderer

*vim org1/docker-compose-orderer.yml*

*- ORDERER\_OPERATIONS\_LISTENADDRESS=0.0.0.0:8443 # operation RESTful API*

*- ORDERER\_METRICS\_PROVIDER=prometheus # prometheus will pull metrics from orderer via /metrics RESTful API*

*- ORDERER\_OPERATIONS\_TLS\_ENABLED=false*

*- ORDERER\_OPERATIONS\_TLS\_PRIVATEKEY=/var/hyperledger/tls/server.key*

*- ORDERER\_OPERATIONS\_TLS\_CERTIFICATE=/var/hyperledger/tls/server.crt*

*- ORDERER\_OPERATIONS\_TLS\_CLIENTROOTCAS=[/var/hyperledger/tls/ca.crt]*

*- ORDERER\_OPERATIONS\_TLS\_CLIENTAUTHREQUIRED=false*

*- 8443:8443*

For **Org2** Orderer

*vim org2/docker-compose-orderer.yml*

*- ORDERER\_OPERATIONS\_LISTENADDRESS=0.0.0.0:844****4*** *# operation RESTful API*

*- ORDERER\_METRICS\_PROVIDER=prometheus # prometheus will pull metrics from orderer via /metrics RESTful API*

*- ORDERER\_OPERATIONS\_TLS\_ENABLED=false*

*- ORDERER\_OPERATIONS\_TLS\_PRIVATEKEY=/var/hyperledger/tls/server.key*

*- ORDERER\_OPERATIONS\_TLS\_CERTIFICATE=/var/hyperledger/tls/server.crt*

*- ORDERER\_OPERATIONS\_TLS\_CLIENTROOTCAS=[/var/hyperledger/tls/ca.crt]*

*- ORDERER\_OPERATIONS\_TLS\_CLIENTAUTHREQUIRED=false*

*- 844****4****:844****4***

For **Org3** Orderer

*vim org3/docker-compose-orderer.yml*

*- ORDERER\_OPERATIONS\_LISTENADDRESS=0.0.0.0:844****5*** *# operation RESTful API*

*- ORDERER\_METRICS\_PROVIDER=prometheus # prometheus will pull metrics from orderer via /metrics RESTful API*

*- ORDERER\_OPERATIONS\_TLS\_ENABLED=false*

*- ORDERER\_OPERATIONS\_TLS\_PRIVATEKEY=/var/hyperledger/tls/server.key*

*- ORDERER\_OPERATIONS\_TLS\_CERTIFICATE=/var/hyperledger/tls/server.crt*

*- ORDERER\_OPERATIONS\_TLS\_CLIENTROOTCAS=[/var/hyperledger/tls/ca.crt]*

*- ORDERER\_OPERATIONS\_TLS\_CLIENTAUTHREQUIRED=false*

*- 844****5****:844****5***

##### Per each Peer

**(HL Explorer is connecting to ‘peer0.org1.example.com’ (you can see in *‘~/blockchain-explorer/app/platform/fabric/connection-profile/hlf\_multi\_host\_network.jon*’ file) and hence private key and certificates info has to be provided at this peer configurations. At rest of the peer’s configs, it is NOT required)**

For **Org1** Peer**0**

*vim org1/docker-compose-peer.yml*

*- CORE\_OPERATIONS\_LISTENADDRESS=0.0.0.0:944****3*** *# operation RESTful API*

*- CORE\_METRICS\_PROVIDER=prometheus # prometheus will pull metrics from orderer via /metrics RESTful API*

*- CORE\_OPERATIONS\_TLS\_ENABLED=false*

***- CORE\_OPERATIONS\_TLS\_PRIVATEKEY=/var/hyperledger/tls/server.key***

***- CORE\_OPERATIONS\_TLS\_CERTIFICATE=/var/hyperledger/tls/server.crt***

***- CORE\_OPERATIONS\_TLS\_CLIENTROOTCAS=[/var/hyperledger/tls/ca.crt]***

*- CORE\_OPERATIONS\_TLS\_CLIENTAUTHREQUIRED=false*

*- 944****3****:944****3***

For **Org1** Peer**1**

*- CORE\_OPERATIONS\_LISTENADDRESS=0.0.0.0:944****4*** *# operation RESTful API*

*- CORE\_METRICS\_PROVIDER=prometheus # prometheus will pull metrics from orderer via /metrics RESTful API*

*- CORE\_OPERATIONS\_TLS\_ENABLED=false*

*- 944****4****:944****4***

For **Org2** Peer**0**

*vim org****2****/docker-compose-peer.yml*

*- CORE\_OPERATIONS\_LISTENADDRESS=0.0.0.0:944****5*** *# operation RESTful API*

*- CORE\_METRICS\_PROVIDER=prometheus # prometheus will pull metrics from orderer via /metrics RESTful API*

*- CORE\_OPERATIONS\_TLS\_ENABLED=false*

*- 944****5****:944****5***

For **Org2** Peer**1**

*- CORE\_OPERATIONS\_LISTENADDRESS=0.0.0.0:944****6*** *# operation RESTful API*

*- CORE\_METRICS\_PROVIDER=prometheus # prometheus will pull metrics from orderer via /metrics RESTful API*

*- CORE\_OPERATIONS\_TLS\_ENABLED=false*

*- 944****6****:944****6***

For **Org3** Peer**0**

*vim org****3****/docker-compose-peer.yml*

*- CORE\_OPERATIONS\_LISTENADDRESS=0.0.0.0:944****7*** *# operation RESTful API*

*- CORE\_METRICS\_PROVIDER=prometheus # prometheus will pull metrics from orderer via /metrics RESTful API*

*- CORE\_OPERATIONS\_TLS\_ENABLED=false*

*- 944****7****:944****7***

For **Org3** Peer**1**

*- CORE\_OPERATIONS\_LISTENADDRESS=0.0.0.0:944****8*** *# operation RESTful API*

*- CORE\_METRICS\_PROVIDER=prometheus # prometheus will pull metrics from orderer via /metrics RESTful API*

*- CORE\_OPERATIONS\_TLS\_ENABLED=false*

*- 944****8****:944****8***

#### Bring up Fabric network with 3 nodes and make sure that it is working end to end

*cd ~/HLF-Multi-Host-Docker-Swarm/network*

***./MASTER\_SCRIPT\_TO\_RUN\_ALL.sh***

### Install prerequisites for HL Explorer and clone HL Explorer repository

* 1. Make sure that all the SWs that are mentioned at ‘<https://github.com/hyperledger/blockchain-explorer#30-dependencies---->’ section.
  2. Clone HL Explorer repository

*git clone* [*https://github.com/hyperledger/blockchain-explorer.git*](https://github.com/hyperledger/blockchain-explorer.git)

### Database (PostgreSQL) Setup

* + 1. *cd blockchain-explorer/app*
    2. Modify *explorerconfig.json* to update PostgreSQL database settings. By default, these settings already configured. In case, you don’t find then update as follows:

*"postgreSQL": {*

*"host": "127.0.0.1",*

*"port": "5432",*

*"database": "fabricexplorer",*

*"username": "hppoc",*

*"passwd": "password"*

*}*

* + 1. Update permissions to ‘*db*’ dir; This dir already has 755 permissions. Do below steps, in case it don’t have.

*cd persistence/fabric/postgreSQL; chmod -R 755 db;*

* + 1. **Run create database script**
  1. *cd db;* (which is ‘blockchain-explorer/app/persistence/fabric/postgreSQL/db’)
  2. ***sudo -u postgres ./createdb.sh;*** (You may have to provide password, in case, you set password for postgres user while installing postgresql SW)
  3. **Optional**: Just to check, connect to the PostgreSQL database and run DB status commands

*psql -U postgres*

*\l*

*\c fabricexplorer* (This is to switch to fabricexplorer database (where our tables/data gets stored))

*\d* (You should see all tables (such as peer, blocks etc.) that were created by created.sh script)

### Authorization Configuration (I don’t see any importance of this modification)

*cd blockchain-explorer/app*

* 1. Modify explorerconfig.json to update Authorization (JWT) settings

*"jwt": {*

*"secret": "****Secret23&****",*

*"expiresIn": "2h"*

*}*

Set secret string as highlighted in bold above

### Setup Prometheus server and Grafana with using Docker

Reference: <https://github.com/hyperledger/blockchain-explorer/blob/master/CONFIG-OPERATIONS-SERVICE-HLEXPLORER.md#setup-prometheus-server-and-grafana-with-using-docker>

* 1. **Update/Configure prometheus.yml file**

*vim app/platform/fabric/artifacts/operations/balance-transfer/prometheus.yml*

Update following fields/info:

*- job\_name:* ***'hlf\_multi\_host\_network****'*

*static\_configs*

*- targets: ['ip-172-31-8-85:8443','ip-172-31-8-85:9443','localhost:9444','ip-172-31-39-243:8444','ip-172-31-39-243:9445','ip-172-31-39-243:9446','ip-172-31-32-21:8445','ip-172-31-32-21:9447','ip-172-31-32-21:9448']*

*- targets: ['orderer1.example.com:9443','peer0.org1.example.com:9443','peer1.org1.example.com:9443','orderer2.example.com:9443','peer0.org2.example.com:9443','peer1.org2.example.com:9443','orderer3.example.com:9443','peer0.org3.example.com:9443','peer1.org3.example.com:9443']*

* 1. **Update/Configure Grafana dashboard json file**

*vim app/platform/fabric/artifacts/operations/balance-transfer/balance-transfer-grafana-dashboard.json*

*"title": "****HLF Multi Host Setup****, Quick Summary"*

* 1. **Update/Configure Grafana dashboard json file**

*vim app/platform/fabric/artifacts/operations/grafana\_conf/provisioning/dashboards/dashboard.yaml*

No change is required here

* 1. **Update/Configure Grafana datasource.yaml file**

*vim app/platform/fabric/artifacts/operations/grafana\_conf/provisioning/datasources/datasource.yaml*

No change is required here until unless you are running Prometheus is running on different host and/or port (default set to localhost:9090)

* 1. **\*\* Bring up Explorer, Prometheus and Grafana**

**TODO**

### ~~Configure interested Fabric Network~~

* 1. **~~Modify ‘blockchain-explorer/app/platform/fabric/config.json’ file~~**

~~Change network name and profile file as required~~

*~~"~~****~~hlf\_multi\_host\_network~~****~~": {~~*

*~~"name": "~~****~~hlf\_multi\_host\_network~~****~~",~~*

*~~"profile": "./connection-profile/~~****~~hlf\_multi\_host\_network.json~~****~~"~~*

*~~}~~*

* 1. **~~Modify ‘blockchain-explorer/app/platform/fabric/connection-profile/hlf\_multi\_host\_network.json’ file~~**

*~~"name": "hlf\_multi\_host\_network",~~*

*~~"adminPrivateKey", "signedCert", "tlsCACerts"~~* ***~~for peer0.org1.example.com~~***

### Run Hyperledger Explorer **Using Docker Compose**

Reference: <https://github.com/hyperledger/blockchain-explorer/tree/master#122-run-hyperledger-explorer-using-docker-compose---->

**Note that HL Explorer is connecting to ‘peer0.org1.example.com’ node to get Fabric network info and stats.**

* 1. **IMPORTANT STEP: Update ‘blockchain-explorer/docker-compse.yaml’ file**

*networks:*

*mynetwork.com:*

*external:*

*name:* ***hlf\_multi\_host\_network***

*-* ***/home/ubuntu/HLF-Multi-Host-Docker-Swarm/network/crypto-config****:/tmp/crypto*

***You need to change ALL files that are referenced in volumes of different services in this compose file as done in following steps.***

* 1. **Update ‘blockchain-explorer/examples/net1/config.json’**

*"****hlf\_multi\_host\_network****": {*

*"name": "****hlf\_multi\_host\_network****",*

*"profile": "./connection-profile/****hlf\_multi\_host\_network.json****"*

*}*

* 1. **Create a new file ‘blockchain-explorer/examples/net1/connection-profile/hlf\_multi\_host\_network.json’ as copy of ‘blockchain-explorer/examples/net1/connection-profile/first-network.json’ and update as follows**

*"name": "hlf\_multi\_host\_network",*

*"adminPrivateKey", "signedCert", "tlsCACerts"* ***for peer0.org1.example.com***

* 1. **FINALLY, Bring up HL Explorer, Prometheus and Grafana containers**

***docker-compose up -d***

after running above command, 4 containers (explorer, explorerdb, promethes and grafana) should be up and running

Now, you can access,

* + - * HL Explorer at ‘localhost:**8090**’ (authentication is skipped)

Additionally,

* + - * Premetheus at ‘localhost:**9090**’
      * Grafana at ‘localhost:**3000**’ (Note: when you login for first time, you need to change the password)