DH2i DxOdyssey 21.0 Software: and SQL Edge on Docker Quick Start Guide

DH2i Company

Support: +1 (800) 380-5405 ext. 2
https://dh2i.com/support/
eFax: +1 970-295-4505
Email: support@dh2i.com
https://www.dh2i.com

DxOdyssey and SQL Edge on Docker

This quick start describes how to use DxOdyssey and Azure SQL Edge containers from Docker Hub to connect two SQL instances via tunnels. Using this guide, the user will deploy and configure DxOdyssey and Azure SQL Edge at two sites using Docker commands, create DxOdyssey tunnels so the SQL instances can communicate with each other, and connect the Azure SQL Edge instances together via Linked Server T-SQL command.

Prerequisites

- Two Linux VMs not connected to each other via any type of direct connection; examples include a VM on a local network and a VM on Azure, or two local VMs separated by subnets.
    - Both VMs must have Docker installed and an internet connection. View Docker documentation for installation instructions.
- A valid DxOdyssey license.

Configure Docker and Start Containers at Site A

1. The two containers (DxOdyssey and Azure SQL Edge) need to communicate with each other via network names, so a custom Docker network bridge must be created. Run the command `docker network create -d bridge dxo-network` to create the bridge.
2. Pull the latest DxOdyssey container image using `docker pull dh2i/dxo:latest`.
3. Start the DxOdyssey container using the `docker run` command. DxOdyssey uses TCP port 7979 for gateway group administration with the DxOdyssey Client UI. The -d flag runs the container detached so that the command line may still be used on the host machine, -h assigns the container a hostname used by DxOdyssey, --network adds the container to the dxo-network that was created, --name specifies the name within Docker for the container, -v adds the path for persistent storage, -p is for each port mapping assigned to the container, DX_LICENSE sets the license key, and DX_PASSEY sets a gateway group passkey. The following command example shows port 7979 being mapped to port 7979 on the host for the DxOdyssey Client UI connections. If multiple DxOdyssey containers are running on the same Docker host, then the mapped port for the client UI needs to be unique for each container (ex. 17979:7979, 27979:7979, etc.)
    - `docker run -d -h dxo1 --network=dxo-network --name dxo1 -v dxo1:/etc/dh2i -p 7979:7979 -e "DX_LICENSE=AAAA-BBBB-CCCC-DDDD" -e "DX_PASSKEY=P@ssw0rd" dh2i/dxo:latest`
4. Pull the latest Azure SQL Edge image using `docker pull mcr.microsoft.com/azure-sql-edge:latest`.
5. Start the Azure SQL Edge container using the `docker run` command. When starting the Azure SQL Edge container, specify the --network parameter to add the container to the dxo-network. A container name is needed so DxOdyssey tunnels can resolve the SQL Edge container, port 1433 must be mapped so DxOdyssey can connect to the instance, and the -v /opt/mssql/bin parameter will also need to be specified to persist Azure SQL Edge data. Please see Microsoft documentation for more information on Azure SQL Edge parameters.

- `docker run -d --network=dxo-network --name sqledge1 -p 1433:1433 -v /var/opt/mssql <sql-edge-parameters>`

Connect to and Configure the DxOdyssey Container at Site A

1. Connect to the container using the `docker exec` command. This command opens a new shell session in the container.
   - `docker exec -it dxo1 bash`
2. Set a One-Time PassKey for the gateway using `dxcli set-otpk`. Additional gateways will use the PassKey to join the gateway group using tunneling technology. Save the GUID output from this command.

   Syntax

   `dxcli set-otpk [ttl] [otpk]`

   Parameters

Name	Description	Required
ttl	The time to live	False
otpk	The One-Time PassKey in base64.	False

   Example

   `dxcli set-otpk`

Configure Docker and Start Containers at Site B

1. Create the Docker network bridge by running the command `docker network create -d bridge dxo-network`.
2. Pull the latest DxOdyssey Docker image using `docker pull dh2i/dxo:latest`.
3. Start the DxOdyssey container using the `docker run` command. The OTPK created in the previous section will need to be supplied as a parameter so the container can join the gateway group.
   - `docker run -d -h dxo2 --network=dxo-network --name dxo2 -v dxo2:/etc/dh2i -p 7979:7979 -e "DX_LICENSE=AAAA-BBBB-CCCC-DDDD" -e "DX_OTPK=07509e41-5c17-f931-630b-80b112759979" dh2i/dxo:latest`
4. Pull the latest Azure SQL Edge image using `docker pull mcr.microsoft.com/azure-sql-edge:latest`.
5. Start the Azure SQL Edge container using the `docker run` command. When starting the Azure SQL Edge container, make sure to specify the --network parameter to add the container to the dxo-network. A container name is needed so DxOdyssey tunnels can resolve the SQL Edge container, port 1433 must be mapped so DxOdyssey can connect to the instance, and the -v /opt/mssql/bin parameter will also need to be specified to persist Azure SQL Edge data.Please see Microsoft documentation for more information on Azure SQL Edge parameters.
   - `docker run -d --network=dxo-network --name sqledge2 -p 1433:1433 -v /var/opt/mssql <sql-edge-parameters>`

Connect to and Configure the DxOdyssey Container at Site B

1. Connect to the container using the `docker exec` command. This command opens a new shell session in the container.
    - `docker exec -it dxo2 bash`
2. Create a tunnel for Azure SQL Edge for Site A. The gateways have been automatically joined together into a gateway group using environment variables, so this command can be executed from Site B. The following command example shows the creation of a tunnel with a destination gateway of the DxOdyssey container at site B (dxo2) a destination of the Azure SQL Edge container at Site B (sqledge2) using the default port of 1433, and an origin gateway of the DxOdyssey container at Site A (dxo1) that accepts connections from any local IP (0.0.0.0) on port 11433.

    Syntax

    ```
    dxcli add-tunnel <name> <enabled [true|false]> <destination_gateway>
    <destination_address:destination_port>
    <origin_gateway,origin_address:origin_port[,address_filter]>|<origin
    _gateway,origin_address:origin_port[,address_filter]>
    ```

    Parameters

Name	Description	Required
name	The name of the tunnel.	True
enabled	Enable tunnel (true or false).	True
destination_gateway	The name of the destination gateway.	True
destination_address	The IP address or name of the destination.	True
destination_port	The port number for the destination.	True
origin_gateway	The name of the gateway where the listener is active.	True
origin_address	Set to 0.0.0.0 to allow all IP connections or 127.0.0.1 for local connections only.	True
origin_port	The port number for the origin gateway.	True
address_filter	The name of the address filter to add to the tunnel.	False

    Example

    ```
    dxcli add-tunnel SQL-TUNNEL1 true dxo2 sqledge2:1433
    dxo1,0.0.0.0:11433
    ```

3. Create a tunnel for Azure SQL Edge for Site B. The following command example shows the creation of a tunnel with a destination gateway of the DxOyssey container at site A (dxo1), a destination of the Azure SQL Edge container at site A (sqledge1) using the default port of 1433, and an origin gateway of the DxOdyssey container at site B (dxo2) that accepts connections from any local IP (0.0.0.0) on port 21433:
    - `dxcli add-tunnel SQL-TUNNEL2 true dxo1 sqledge1:1433 dxo2,0.0.0.0:21433`

Connect the Azure SQL Edge Instances via Linked Server

The following command example shows a T-SQL command run on sqledge1 to create a linked server to sqledge2 over DxOdyssey tunnels. Substitute the `<password>` parameter for the one provided in the SQL Edge parameters during container startup.

```
--Add linked server via DxO tunnel
EXEC master.dbo.sp_addlinkedserver @server = N'sqledge2', @srvproduct ='',
@datasrc='dxo1,11433', @provider='SQLNCLI'
go
EXEC sp_addlinkedsrvlogin  @rmtsrvname ='sqledge2', @useself = 'FALSE',
@locallogin = NULL, @rmtuser = 'sa', @rmtpassword ='<password>'
go
EXEC sp_serveroption 'sqledge2', 'rpc out', true;
go
--List the tables in the linked server.
EXEC sp_tables_ex [sqledge2];
go
```

(Optional) Connect Using the DxOdyssey Client UI

To connect to the containers using the DxOdyssey Client UI, the DxOdysseyClientSetup package will need to be installed on a Windows workstation at either Site A or Site B that has network access to the Docker host. Once installed, launch the DxOdyssey Client UI and use the following connection strings to connect to the gateway group:

- Server: <Docker Host VM IP>
- PassKey: <Gateway Group PassKey>

References

- DH2i Support Portal
- DxOdyssey Documentation
- DxOdyssey DxCLI Guide
- Microsoft – Deploy Azure SQL Edge with Docker
- Microsoft – Persist Azure SQL Edge Data
- Docker – Install Docker Engine
- Docker – CLI Reference Guide