Deploying and Running Oracle GoldenGate 19c Microservices on Docker

By: Bobby L. Curtis, MBA

RheoData

Introduction

Enterprises today are quickly transitioning to flexible service-based architectures that can be rapidly and dynamically deployed. Many of these architectures rely on software that performs virtualization of their underlying hardware stack. Docker is the premier service product that uses OS-level virtualization to build and deliver software in containers. Docker containers make deploying software packages easy and simple.

There are many ways to procure a pre-built docker container that can rapidly be deployed; however, there are times when greater flexibility and customization is needed or desired. This paper will explore items that need to be considered when deploying Oracle GoldenGate 19c Microservices within a container.

Software

Docker

Docker is a container platform that allows an organization to package code and/or applications with all dependences into a lightweight, standalone, executable package of software called images. Images can then be executed against the Docker Engine running on either Linux or Windows. Enabling a consistent and repeatable execution of software between development, testing, and production environments.

Docker containers that run against Docker Engine are:

- Standard: Docker is the industry standard for containers. Ensuring that they are portable between environments.

- Lightweight: Containers use the OS kernel and do not require an OS per application. Making servers more efficient and reducing overall licensing costs.

- Secure: Docker provides a safer environment to run applications due to containerization and provides the strongest isolation capabilities in the industry

Containers powered by Docker are running everywhere and on every major platform. This makes a great platform to build and run Oracle GoldenGate 19c Microservices and leverage advanced options like remote capture and remote apply within a hub-n-spoke architecture.

Oracle GoldenGate 19c Microservices

Oracle GoldenGate 19c is an enterprise real-time data integration and replication software. It enables real-time data integration and replication, high-availability solutions, transactional change data capture (CDC) between operational and analytical enterprise systems.

Using Oracle GoldenGate 19c, you can move transactional data across multiple systems in the enterprise in real-time; providing data where it is needed when it is needed. The heterogenous capabilities of Oracle GoldenGate 19c allow you to capture and distribute data between many different data sources, filter transactions within the enterprise, and migrate databases in near zero-downtime.

The Microservices architecture within Oracle GoldenGate 19c builds on top of the existing Oracle GoldenGate framework and makes implementation and usage easier. By enabling microservices on the various Oracle GoldenGate components and providing REST API end-points, administration becomes easier. The enhanced integration within Oracle GoldenGate 19c Microservices open up administration and execution without having to have server access.

Nginx

Nginx is a free open-source or purchased HTTP/Reverse Proxy utility that is recommend by Oracle to use in front of the Oracle GoldenGate 19c Microservices architecture. The purpose of the reverse proxy is to consolidate the port numbers that are established with Oracle GoldenGate 19c Microservices for all deployments on the host.

Deployment Model

Oracle GoldenGate 19c Microservices can be deployed in many use-cases or architecture models. The primary goal of Oracle GoldenGate 19c Microservices is to enable users to build solid data integration environments on a large scale, quickly establish active/active environments, or scale out mesh architectures.

One of the popular deployment models is the hub-n-spoke model. This model centralizes the management of Oracle GoldenGate 19c and enables the usage of remote capture and remote apply. There is no need to ship trail files from source to target because everything resides on the central hub of the architecture.

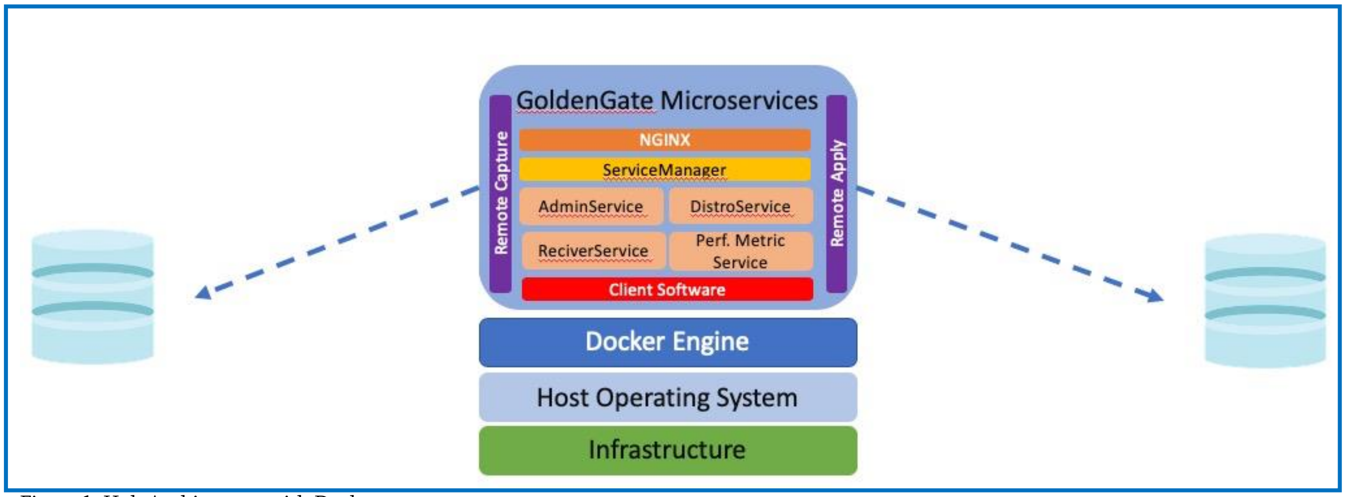


Figure 1: Hub Architecture with Docker

Connections/Client Software

Connections to the source and target databases are done over standard Oracle TNS Names connections or EZConnect connections. For this to be enabled, the hub has to have all the Oracle Client software needed per database platform connecting to. Depending on the Oracle Client packages used, this can bloat the Docker container. It is best to use a minimal version of the Oracle Client software.

Networks

The standard Docker network works out of the box; however, if there is a need to string a few Docker containers together the data will not route correctly without IP addresses. To get around this issue Docker provides a way to build additional networks. Build a defined network for all containers to use. This will allow communication between all Docker containers on this defined network.

Volumes

There are multiple ways of sharing data between the host and the Docker Engine, Bind Mounts and Volumes. Volumes are an internal and preferred mechanism for persisting data between Docker containers. Additionally, Volumes provide advantages over bind mounts in a few ways:

- Easier to backup
- Management via Docker CLI
- Run on Linux or Windows systems
- Share between multiple containers
- Drivers that let you store volumes on remote hosts or cloud platforms

With Oracle GoldenGate 19c Microservices, Docker volumes are a great choice because the deployment home can be placed within a volume and persisted over container rebuilds and upgrades.

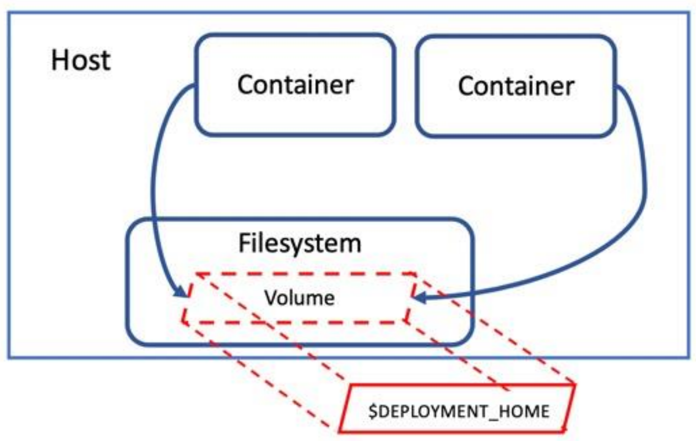


Figure 2: Deployment Homes in Docker Volumes

Building

Docker Image

The basis for the Docker container is an image that must be built with a Dockerfile and a few corresponding scripts. The software that is needed to build this container is:

- `Oracle GoldenGate 19c Microservices`
- `Oracle Database 19c Client – base lite`
- `Oracle Database 18c Client – base lite`

The images are also built on top of the Oracle provided Docker images (oraclelinux:7-slim). With all these items in place, you can write a Dockerfile that will build the Docker image and pre-stage all components needed.

Network

Docker provides a bridge network by default. This default network has limitations when looking at inter-communication between containers. To work around this, simply create a new network that the containers will connect to and use for network activity.

Command to create a network is:

```
docker network create \
--driver=bridge \
--subnet=172.20.0.0/16 \
--ip-range=172.20.10.0/16 \
<network_name>
```

Volumes

Docker provides a quick command that allows for the creation of volumes within the host filesystem. A best practice and a good habit to get into is to name the volume similar to what will be *in* the volume and matching a directory *within* the container. This allows you to identify the contents of the volume within the container and the local host filesystem.

For Oracle GoldenGate 19c Microservices, the deployment home is located outside of the Oracle Goldengate home. This makes the usage of volumes simple to use and a great option in flexibility between container builds.

Command to create a volume is:

```
Docker create volume <volume_name>
```

Start Oracle GoldenGate 19c Microservices Container

With the Oracle GoldenGate 19c Microservices image ready, network established, and a volume ready; the Docker container can be started.

To run the container:

```
docker run -dit
--memory=1024M \
--privileged \
-v gg_deployments:/opt/app/oracle/gg_deployments:rw \
--hostname=gg19c \
-p 59011:5901/tcp \
-p 1522:1521/tcp \
-p 443:443/tcp \
-p 2220:22/tcp  \
--network ggtest \
--name ogg \
rheodata:ogg19.1.0.0.4
```

Configure Oracle GoldenGate 19c Microservices

With the container running, there are multiple ways of configuring Oracle GoldenGate 19c Microservices. Ideally, with the automation that Docker builds provide, the ServiceManager and first deployment would be established. In reality and for customization purposes, it is easier to access the container and build the deployments that are needed.

Preferred Method

With Docker containers, if it is built with xterm rpm and a vnc server, the container can be accessed via a X11 terminal. From the X11 terminal, the Oracle GoldenGate Configuration Assistant (OGGCA) can be ran and deployments configured.

Silent Method

If the container is built without the xterm rpm and a vnc server, the container can be configured by using the silent install method of the Oracle GoldenGate Configuration Assistant. The downside of this approach is that the OGGCA does not come with a response file that is needed to run a silent install.

Interact

Docker is a great way of taking an application like Oracle GoldenGate 19c Microservices and make it easy to install. Once it is installed and deployments are configured, there a multiple way of interacting with Oracle GoldenGate.

Web Browser

The web browser is the easiest way of interacting with Oracle GoldenGate 19c Microservices. By accessing Oracle GoldenGate via HTTP or HTTPS, the application can be logged into and configured with minimal effort.

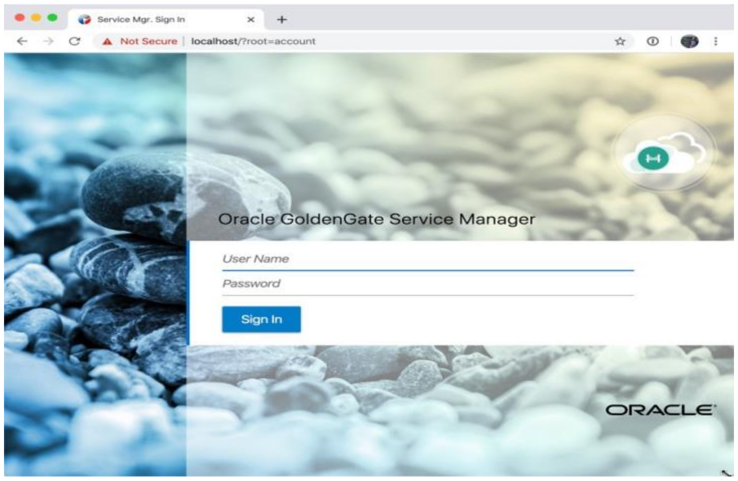


Figure 3: Oracle GoldenGate Browser Access on Docker

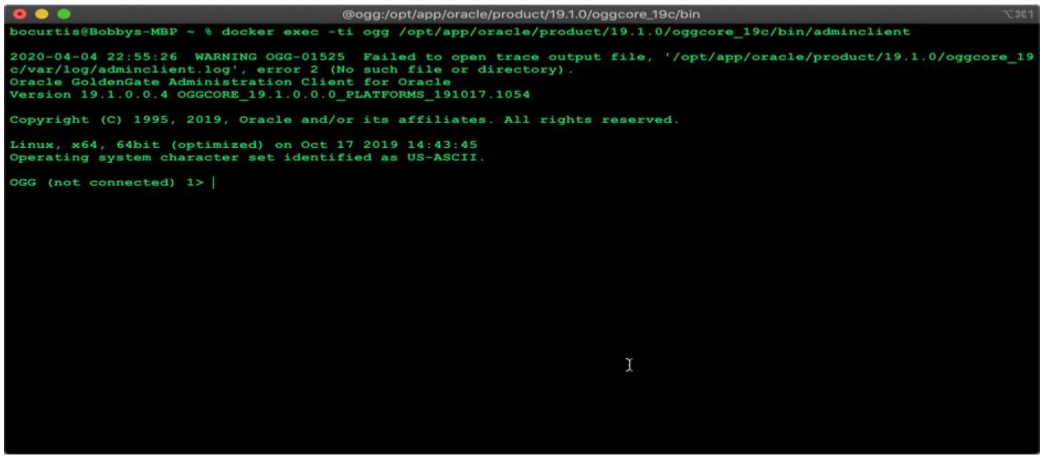
Command Line

Oracle GoldenGate 19c Microservices has a brand new command line tool called Administration Client (AdminClient). In order to access AdminClient from outside the Docker container, simply use the Docker exec command.

Open AdminClient via Docker:

```
Docker exec -ti \
<container_name> \
$OGG_HOME/bin/adminclient
```

Note: To access $OGG_HOME/bin from outside the container, you will need to use the absolute path.



```
@ogg:/opt/app/oracle/product/19.1.0/oggcore_19c/bin
bocurtis@Bobbys-MBP ~ % docker exec -ti ogg /opt/app/oracle/product/19.1.0/oggcore_19c/bin/adminclient

2020-04-04 22:55:26  WARNING OGG-01525  Failed to open trace output file, '/opt/app/oracle/product/19.1.0/oggcore_19
c/var/log/adminclient.log', error 2 (No such file or directory).
Oracle GoldenGate Administration Client for Oracle
Version 19.1.0.0.4 OGGCORE_19.1.0.0.0_PLATFORMS_191017.1054

Copyright (C) 1995, 2019, Oracle and/or its affiliates. All rights reserved.

Linux, x64, 64bit (optimized) on Oct 17 2019 14:43:45
Operating system character set identified as US-ASCII.

OGG (not connected) 1> |
```

Figure 4: Oracle GoldenGate AdminClient from within Docker

Summary

Setting up and running Oracle GoldenGate 19c Microservices in Docker containers is a great way to jump-start many data integration initiatives. By leveraging Docker to build a flexible and scalable deployment model, Oracle GoldenGate 19c Microservices can be provisioned quickly and provide production ready environments within minutes that are easy to use. This paper covered a number of aspects of moving Oracle GoldenGate 19c Microservices into container-based architecture that centralize management without effecting data streams. Do not consider this paper to be the final word on how to configure Oracle GoldenGate 19c Microservices within Docker; things are always changing but this is a great starting place.

For more information contact RheoData at solutions@rheodata.com.