GDL Deployment Guide

9/24/2020

Table of Contents

Contents

Search the table of all articles in this guide, listed in alphabetical order, to find the article you need.

Article	Description
Article	Description

"> Deployment Guide Genesys Data Layer Lab Version."> Access Docker Container Accessing Docker Container"> Access Log Files Debugging log files for troubleshooting."> Configure Producer and Consumer Config Configuring client side ports to access Kafka layers."> Deploy Kafka Clusters Deployment of GDL clusters in Docker environment."> Docker Ready Environment Docker Setup Instructions"> Environment Variables Environment variables in GDL"> Installing Genesys Data Layer Installation steps from Installation Package (IP) files."> Life Cycle of Docker Container Life cycle of Docker Containers"> Overview Overview about Genesys Data Layer"> Prerequisites Prerequisities for GDL"> Prepare for Operating System Detailed Instructions to set up CentOS/7 in Windows 10 Environment"> Preparing to Install Genesys Data Layer Hardware requirements for Installing Genesys Data Layer."> Restart Kafka Clusters How to restart Kafka clusters?"> Verify GDL Docker Container Verification steps for GDL Docker.

Genesys Data Layer Lab Version.

Deployment Guide Overview

Welcome to the The Genesys Data Layer (GDL) deployment guide. This document introduces you to configuration, installation, setup, and start procedures involved in the setup of Genesys Data Layer. It lists the procedures involved in setting up the operating system (CentOS/7 through Windows 10), setting up the Docker ready environments, and deploying the GDL component through Docker containers.

This document is valid only for the 9.0.0 release of GDL. The deployment steps described in this guide is only for a lab environment and not suitable for production deployments.

Overview

Contents

Overview about Genesys Data Layer

GDL Overview

Genesys Data Layer (GDL) is built on the Apache Kafka distributed data pipeline and streaming platform.Genesys products write data to GDL and Genesys InfoMart (GIM) reads from it to populate the GIM database, which is used for historical reporting.

Docker Compose Files

GDL consists of three components: Kafka, Zookeeper and Management station. Based on the number of components, the different cluster types are:

- Zookeeper - 1, Kafka - 1, and Management station.
- Zookeeper - 1, Kafka - 3, and Management station.
- Zookeeper - 3, Kafka - 3, and Management station.

These cluster types can be defined through docker-compose files available in the IP (Installation Package) files. The different docker-compose files are:

- docker-compose-single.yml - Creates 1 Zookeeper, 1 Kafka, and the Management Station.
- docker-compose-half.yml - Creates a cluster with 1 Zookeeper, 3 Kafka, and the Management Station.
- docker-compose-full.yml - Creates a cluster with 3 Zookeepers, 3 Kafka, and the Management Station.

The definitions and structure of Docker Compose files can be referred here.

All the mandatory configuration parameters required for GDL lab versions are pre-configured in the `docker-compose` files.

Management Station API

The Management station API allows the Genesys Data Layer users to view and modify the GDL cluster properties and configuration through RESTful API. It allows to view the number or brokers, run test cases, create topics and volumes, etc.

The main API of GDL Management Station API are:

1. Cluster API (Operations pertaining to metadata in Kafka Cluster) to verify the number of brokers and access points of Kafka.
2. Command API (Execute deployment test) to execute the deployment test cases of Kafka layers.
3. Topic API (Operations pertaining to Kafka topics) to create, retrieve, update, and delete Kafka topics.

Volume and Log Files

Named volumes and log files are used by each Zookeeper and Kafka to persist the application log under `/var/lib/zookeeper`, and `/var/lib/kafka` respectively.

The server log for debugging purposes are under `/mnt/log/kafka`, `/mnt/log/zookeeper` and `/mnt/log/tenant-api` for the 3 components (Kafka, Zookeeper, and tenant-api) respectively.

The naming convention in the `docker-compose` file reflects the instance that uses the volume. For example, in `docker-compose-single.yml`, the volume names used in the compose files are

1. **zookeeper:**
2. **kafka:**
3. **tenant-api**

Environment Variables

Contents

Environment variables in GDL

Mandatory Environment Variables

Although all the mandatory configuration parameters are set in the docker-compose files, the following mandatory environment variables can be used to change the configuration of Kafka and Zookeeper in .yml files.

IP addresses of the Zookeeper pod(s)

It should be a comma delimited string used to generate the value of zookeeper.connect. The format of the array is:

```
{IP Address1}:{port1}[;{IP Address2}:{port2};{IP Address3}:{port3}]
```

Examples:

```
KAFKA_ZOOKEEPER_CONNECT=10.64.4.253:2181,10.64.4.10:2181,10.64.4.78:2181
```

For single host deployment:

```
KAFKA_ZOOKEEPER_CONNECT=localhost:2181, localhost:2183,localhost:2185
```

IP addresses of the Kafka pods

The format of the array is given below (if the listeners are more, it should be delimited by a comma):

```
KAFKA_ADVERTISED_LISTENERS=PLAINTEXT://localhost:9092
```

Note, the hostname cannot be used other than "localhost" in one case.

Broker ID

```
KAFKA_BROKER_ID = 1
```

Zookeeper Environment Variable

```
ZOOKEEPER_CLIENT_PORT=2181
```

If it is a multi-Zookeeper cluster, the following Environment variables are also mandatory.

- ZOOKEEPER_SERVERS
- ZOOKEEPER_SERVER_ID

Prerequisites

Contents

Prerequisities for GDL

Prerequisites for Deploying GDL

Before configuring Genesys Data Layer, you must:

- Make sure you have Centos/7 environment. It can be a physical Centos/7 server or you can configure the Vagrant software for building and managing virtual machine environment with Centos/7.
- Install Docker in Centos/7.
- Unzip the IP (Installation Package) file, **`IP_GDL_900_ENU_DockerLinux.zip`** and place in a folder. (E.g. **`C:\dockerlinux\`**).
- Add the above path in vagrant properties file to use as a Shared folder between host machine and guest machine.

Preparing to Install Genesys Data Layer

Contents

Hardware requirements for Installing Genesys Data Layer.

Preparing to Install GDL

This page describes example steps to prepare a system for the installation of Genesys Data Layer, including the installation of Docker on a CentOS/7 server. After you complete the instructions on this page, you will be ready to deploy GDL Docker images.

> **Important**
>
> This page provides an example scenario, Docker version on CentOS Linux release 7 (CentOS 7) .This page does not describe all deployment scenarios, and is applicable to only the indicated software (Operating System, Docker) release. For other releases, the required steps may vary.

Before you begin

Ensure that you have a suitably-prepared environment, including the following minimum hardware and system requirements

GDL lab version require significant resources, so note the following minimums:

- 64-bit compatible CPU architecture
- Minimum of 10 GB of RAM. More is strongly recommended and is required in many cases. Lab version deployments commonly have 16 to 64 GB RAM.
- Minimum of 40 GB of available disk space.
- Windows 10 Environment

Installing Genesys Data Layer

Contents

Installation steps from Installation Package (IP) files.

Before you begin

Before you complete the steps on this page, prepare the environment as described in Preparing to Install Genesys Data Layer

In addition:

Acquire the Genesys Data Layer Installation package

> ### Warning
> The Genesys Data Layer Installation Package comprises several files, some of which are too large to be downloaded using Genesys Download Center. Contact Customer Care for assistance.

Ensure that you have the latest GDL Packages (IP); talk to your Genesys representative for information about where to download the IP.

IP/File	tar files
Docker container (Docker Linux platform) **IP_GDL_900_ENU_DockerLinux.zip**. Contains **IP_GDL_900_ENU_DockerLinux.tar** file and 3 .yml files such as • docker-compose-cluster-full.yml • docker-compose-cluster-half.yml • docker-compose-single.yml	**IP_GDL_900_ENU_DockerLinux.tar**— contains the GDL Docker image, which contains the zookeepers and kafka clusters. This image is used for lab version deployments.

Deploying the containers

Use the steps in this section to deploy the Docker containers. Note that Genesys does not ship Docker as a part of Genesys Data Layer. You must install Docker in your environment before you can load the Genesys Data Layer

containers; see Preparing to Install Genesys Data Layer. Install Docker according to the instructions on the Docker Ready Environment. A Docker deployment provides a complete self-contained environment, so that you do not need to manually configure ports or address compatibility issues. Genesys Data Layer docker images (lab version) is available through IP files.

Procedure: Unzip files from IP

Purpose: Use the steps in this procedure to prepare the Docker containers.

Steps

Complete the following steps on the host machine, except where noted otherwise:

1. Create a folder named dockerlinux in C: drive.
2. Copy the docker container image file from the IPs (IP_GDL_900_ENU_DockerLinux.zip) and unzip the contents of file at C:\dockerlinux folder.
3. Prepare for Operating System
4. Docker Ready Environment
5. Deploy Kafka Clusters

Prepare for Operating System

Contents

Detailed Instructions to set up CentOS/7 in Windows 10 Environment

Recommended OS

- Linux CentOS 7 is recommended and tested version.
- Windows 10 is not recommended due to Hyper-V issues existing in Windows 10 environment.
- Mac OS is possible, but not tested with these procedures.

Install Virtual Box

Download the VirtualBox setup file for Windows hosts and execute the setup file with admin privileges.

Link:https://www.virtualbox.org/wiki/Downloads

Install Vagrant

Download Vagrant 2.1.2 version (recommended tested version) to build and manage virtual machine environment. The general downloads for all versions can be obtained here, and the archived versions are available here.

Run the `vagrant 2.1.2 x86_64.msi` setup file, and install in `C:\` drive.

Configure Vagrant

1. Create a folder of any name (say GDL) in `C:\HashiCorp\Vagrant\`
2. Launch Command Prompt as Administrator (Run as Administrator)
3. Change the directory of Command Prompt as `C:\HashiCorp\Vagrant\GDL`
4. Type the command **`vagrant init centos/7`** and click **Enter**.

```
C:\>cd Hashicorp

C:\HashiCorp>cd Vagrant

C:\HashiCorp\Vagrant>cd GDL

C:\HashiCorp\Vagrant\GDL>vagrant init centos/7
```

Vagrant creates a configuration file named Vagrantfile in GDL folder.

CentOS 7 Specifications

1. Open the VagrantFile (under C:/Hashicorp/Vagrant/GDL) using a text editor(run text editor as Administrator) and change the following properties of the file.
    1. `config.vm.box="centos/7"` (in line 15)

```
13    # Every Vagrant development environment
14    # boxes at https://vagrantcloud.com/sea
15    config.vm.box = "centos/7"
16
```

    2. `config.vm.network "private_network", ip: "192.168.33.10"` (in line 35)

```
33    # Create a private network, which allows host-only access to the
34    # using a specific IP.
35    config.vm.network "private_network", ip: "192.168.33.10"
36
```

    3. `config.vm.provider "virtualbox" do|vb|`(in line 52)

```
50    # Example for VirtualBox:
51    #
52     config.vm.provider "virtualbox" do |vb|
53    #   # Display the VirtualBox GUI when booting the machine
```

    4. `vb.memory= "4096"` (in line 57)
    5. `end` (in line 58)

```
56    #   # Customize the amount of memory o
57     vb.memory = "4096"
58     end
```

    6. `config.vm.synced_folder "C:/dockerlinux/", "/Shared", nfs: true`(in line 70)

```
69    # SHELL
70    config.vm.synced_folder "C:/dockerlinux", "/Shared", nfs: true
71  end
```

Ensure the path specified in #6 (line 70) contains the unzipped content of IP file **`IP_GDL_900_ENU_DockerLinux.zip`**. It contains four files as given below.

1. docker-compose-cluster-full.yml
2. docker-compose-single.yml
3. docker-compose-cluster-half.yml
4. IP_GDL_900_ENU_DockerLinux.tar

In Docker environment, these files are accessed from the host machine(Windows 10) through guest machine (CentOS/7).

Logging on to CentOS using Vagrant

1. Type the command **`vagrant up`** to boot up the newly created CentOS/7 Virtual Machine VM.
    1. If you encounter error while booting up CentOS/7 due to hyper-V, follow the steps given here to remove the hyper-V issue.
    2. If you encounter error regarding shared folders (mounting issues), type the command **`vagrant plugin install vagrant-vbguest`** to use the folders of host machine (Windows 10) in guest machine (CentOS/7).
2. Type the command **`vagrant reload`** to reflect the changes made in vagrant box.
3. Type the command **`vagrant ssh`** to login the CentOS/7 VM as Vagrant user.

The VM setup with CentOs/7 is ready for the installation of Docker software.

Docker Ready Environment

Contents

Docker Setup Instructions

Install Docker for CentOS

1. Follow the instructions for installing Docker in CentOS as given here.
2. Login as super user in CentOS using `sudo su` command.
3. Type the following command to install required packages.
    1. ```
       sudo yum install -y yum-utils \
       device-mapper-persistent-data \
       lvm2
       ```
4. Type the following command to download the Docker setup file.
    1. ```
       sudo yum-config-manager \
       --add-repo \
       https://download.docker.com/linux/centos/docker-ce.repo
       ```
5. Type the following command to install the Docker software.
    1. `sudo yum install docker-ce`
6. Type the following commands to enable Docker softare.
    1. `sudo su` (Enable as a super user)
    2. `systemctl enable docker`
7. Type the following command to start the Docker software after successful installation.
    1. `sudo systemctl start docker`

Install Docker Compose

Docker Compose enables a quick way of installing pre-configured Kakfa layers, and follow the commands to install Docker compose.

1. Install Docker Compose by typing the following command
    1. ```
       sudo curl -L https://github.com/docker/compose/releases/download/1.22.0/docker-compose-$(uname -s)-$(uname -m) -o /usr/local/bin/docker-compose
       ```
2. Execute these commands in root directory. (Access root directory using command **`cd /`**)
       ```

1. `sudo chmod +x /usr/local/bin/docker-compose`
2. Execute the following command under `/usr/bin` of root folder `sudo ln -s /usr/local/bin/docker-compose docker-compose`

This Docker environment is ready for GDL deployment at the end of this step.

Deploy Kafka Clusters

Contents

Deployment of GDL clusters in Docker environment.

Execute Docker Compose Files

Change the directory of command prompt to `Shared` folder in CentOS/7. The **Shared** folder can access the files from host machine where the IP packages were unzipped.

1. Display the list of files in Shared folder by using **`ls`** command. The folder displays four files as follows.
    1. docker-compose-cluster-full.yml
    2. docker-compose-single.yml
    3. docker-compose-cluster-half.yml
    4. IP_GDL_900_ENU_DockerLinux.tar
2. Execute the following command to run the Docker image.
    1. `docker image load -i IP_GDL_900_ENU_DockerLinux.tar`
3. Execute the following command to bring up the GDL clusters.
    1. `sudo docker-compose -f(docker compose file name) up`
4. To enable full cluster GDL (3 Zookeepers and 3 Kafka clusters and 1 Management API), type the following command.
    1. `sudo docker-compose -f docker-compose-cluster-full.yml up`

Wait till the log message appears as **started application in nn.nnn seconds (JVM Running for nn.nnn)**

Verify GDL Docker Container

Contents

Verification steps for GDL Docker.

Verify Docker Deployment

The successfully running GDL Docker containers can be verified by using the following command in another instance of Command Prompt (login as vagrant ssh in root folder)

```
docker ps
```

It lists the successfully running Docker containers such as Zookeeper, Kakfa, and tenant-api Docker containers.

Verify using Management API

The successfully running GDL Docker containers can also be verified by accessing Management station API.

1. Click here to access the Management Station API with default username as **admin** and password as **admin**. (The IP address of the Swagger API UI can be modified in Vagrant properties file inconfig.vm.network "private network" property)
2. Click **Cluster API** (Operations pertaining to metadata in Kafka Cluster) to verify the number of brokers and access points of Kafka. It returns the port numbers of Kafka access points where the messages are coupled to Kafka layers (GDL).
    1. Click **Try it out**.
    2. Click **Execute**.
    3. The Response Body JSON file returns the number of available brokers and its port numbers.

Execute Deployment Test

Click **Command API** (Execute deployment test) to execute the deployment test cases of Kafka layers. It is an essential test case where the pseudo messages are published and consumed within the Kafka ports.

1. Click **Try it out**.
2. Select **start**.
3. Click **Execute**. (Ensure all your Kafka layers are up before executing the test case)

4. The Response Body JSON file returns the message as **command started**.

The test cases take approximately seven to ten minutes to verify the deployment of Kafka layers. After ten minutes, follow the given steps to access test results.

1. Click **Try it out**.
2. Select **result**.
3. Click **Execute**.

The Response Body JSON file returns the result of executed test cases.The success message indicates the flawless data transfer between producer and consumer.

Access Log Files

Contents

Debugging log files for troubleshooting.

Access Server Log Files

Log files of Kafka layers can be accessed by following commands.

1. Change the directory of command prompt to `/var/lib/docker/volumes`
2. List all the available volumes by using command **`ls`**
3. Change the directory to specific kakfa layer, say **`cd docker_kafka11-log`**
4. List all the available folders by using command **`ls`**
5. Change the directory to `cd` **`_data/`**
6. List all the available log files by using command **`ls`**

`server.log` file can be used to check the log files of kafka layer.

Access Data Log Files

Volume files of Kafka layers and zookeeper layers (as defined in the respective docker-compose.yml files) can be accessed by following commands.

1. Change the directory of command prompt to `/var/lib/docker/volumes`
2. List all the available volumes by using command **`ls`**
3. Change the directory to specific kakfa layer, say **`cd docker_kafka11`**
4. List all the available folders by using command **`ls`**
5. Change the directory to `cd` **`_data/`**
6. List all the available volume files by using command **`ls`**

The same procedure can be repeated for other volumes of zookeeper.

Restart Kafka Clusters

Contents

How to restart Kafka clusters?

Stop Kafka Clusters

To stop the clusters gracefully, type the following commands in command prompt.

1. Press **Ctrl+C.**(The clusters shutdowns in a sequential order to complete the process)
2. `sudo halt` (To exit as a super user)
3. `vagrant halt` (To halt the virtual machine)
4. `exit`

Restart Kafka Clusters

To restart the full cluster (3 Zookeepers + 3 Kafka layers + 1 Management API), follow the below steps.

1. Launch Command Prompt as administrator.
2. Change the directory to Vagrant folder. (E.g `C:\HashiCorp\Vagrant\GDL`)
3. `vagrant up`
4. `vagrant ssh`
5. `sudo su`
6. `cd /Shared`
7. `docker image/load -i IP_GDL_900_ENU_DockerLinux.tar`
8. `sudo docker-compose -f docker-compose-cluster-full.yml up`

Kafka clusters starts running at the end of this step.

Configure Producer and Consumer Config

Contents

Configuring client side ports to access Kafka layers.

Access GDL

Client VM must be able to Telnet to all the Kafka nodes within the same network. Further reference of connecting Kafka nodes can be referred at https://kafka.apache.org/documentation/

Refer Producer Config and Consumer Config section for configuring the Kafka layers.

Mention the Kafka ports in Config property as given below.

`bootstrap.servers-host1:port1,host2:port2,...`

where `host1:port1` will be `localhost:9092` and so on.

Access Docker Container

Accessing Docker Container

To access within Docker container, use the following commands.

1. Change the directory of command prompt in CentOS/7 to `/Shared`
2. Change the directory of command prompt to `/var/lib/docker/volumes`
3. `docker exec -it (docker image name) /bin/bash`
    1. For example: `docker exec -it shared_kafka-3_1 /bin/bash`
4. `ps -efw |grep -i kafka`
5. `top`

It can be used to check the CPU memory and RAM usage of Docker containers at any instance of time.

Life Cycle of Docker Container

Contents

Life cycle of Docker Containers

Life Cycle and Auto Restart of Docker Containers

The Dockerfile is built in such a way when a Kafka or a Zookeeper process is dead, the Docker containers cease to serve the Kafka bus. The Docker orchestration infrastructure or managed Docker environment restarts the Docker container to continue the data flow among different layers.