Sunil Kumar

Sunil.k@neudesic.com

Abstract

Loren ipsum

Mule flex gateway

[Document subtitle]

Contents

[Introduction 2](#_Toc143244748)

[Flex gateway demonstration using Docker in connected mode. 3](#_Toc143244749)

[Prerequisites 3](#_Toc143244750)

[Install, register, and run Flex Gateway 3](#_Toc143244751)

[Containerizing Mule runtime 4](#_Toc143244752)

[Installing runtime manager agent. 7](#_Toc143244753)

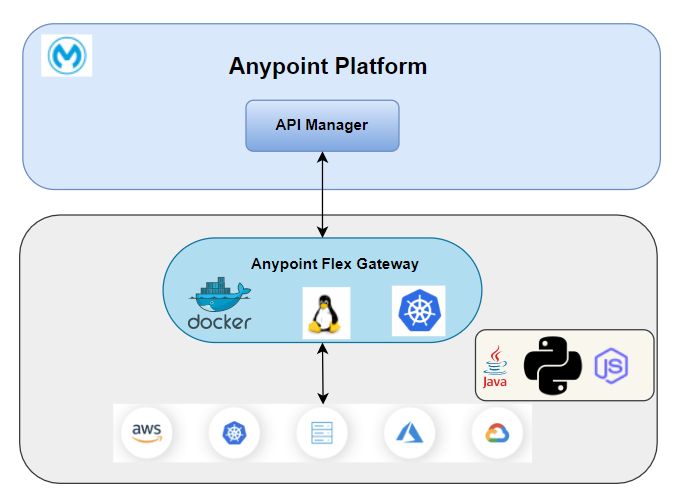
[Deploy Mule API and manage it via Flex gateway. 8](#_Toc143244754)

[Deploy Non-Mule API and manage it via Flex gateway. 9](#_Toc143244755)

[Flex gateway demonstration using Kubernetes in connected mode 10](#_Toc143244756)

# Introduction

Anypoint Flex Gateway is MuleSoft’s new API gateway offering to manage and secure APIs running anywhere (cloud, on premise) in any tech stack (java, .net, python, etc.)

With the new Anypoint flex gateway, Organizations will be able to manage and govern all the APIs under one roof.

Anypoint Flex Gateway can be run in two ways. We will be using the connected mode for this tutorial.

* Connected Mode – In this mode API management is done using Anypoint control plane.
* Local Mode – API Management is done locally using stored configuration files.

# Flex gateway demonstration using Docker in connected mode.

In this tutorial, we will –

1. Install Mule flex gateway image in a docker container.
2. Register the flex gateway to Anypoint platform.
3. Manage and apply policies to a ule application

## Prerequisites

* Anypoint Platform account with valid credentials
* Valid Flex Gateway permissions on Anypoint account
* Docker Desktop

## Install, register, and run Flex Gateway

Objective of this section is to install and run a mule flex gateway on a docker container. This gateway will act as a front-end interface for our APIs. It provides various capabilities to manage, secure, monitor, and control the flow of API traffic between clients and backend services.

1. Log in to Anypoint Platform and navigate to **Runtime Manager**. Click on the **Flex Gateways** tab on the left and select **Add Gateway**. This will present different options to select where to set up the Flex Gateway.
2. Select **Container** and then **Docker**. This will dynamically change the webpage and show some commands to setup the gateway. These commands should be executed.

A screenshot of a computer

Description automatically generated

1. Execute the below docker command to pull the flex gateway image in docker.

*docker pull mulesoft/flex-gateway*

1. Verify the image was correctly installed by executing following command.

*docker images*

1. Register Flex Gateway in docker to Anypoint Platform by running the below command after replacing,

**<gateway-registration-folder>** with the folder name where registration detail will be saved.

**<organization-id>** with the value of organization id.

**<access-token>** with the value of access token.

**<gateway-name>** with the name of flex gateway.

If docker is running in Windows -

*docker run --entrypoint flexctl -v "<gateway-registration-folder>":/registration mulesoft/flex-gateway register --organization=<organization-id> --token=<access-token> --output-directory=/registration --connected=true <gateway-name>*

If docker is running in Linux

*docker run --entrypoint flexctl -u $UID -v "<gateway-registration-folder>":/registration mulesoft/flex-gateway register --organization=<organization-id> --token=<access-token> --output-directory=/registration --connected=true <gateway-name>*

Note - Copy the command from Runtime Manager to register your gateway. It already contains the token and organization ID you need to authenticate and connect the gateway with your Anypoint Platform account.

Once above command executes, it creates a new flex gateway in the Anypoint Platform.

Start the gateway by running the below command after replacing,

**<gateway-registration-folder>** with the folder name where registration detail was saved using previous command.

*docker run --rm -v "<gateway-registration-folder>":/usr/local/share/mulesoft/flex-gateway/conf.d -p 8081:8081 mulesoft/flex-gateway*

A screenshot of a computer

Description automatically generated

## Containerizing Mule runtime

The objective of this section is to containerize Mule Runtime environment, along with dependencies, and configurations, into a Docker container. Later we will use this runtime to deploy our Mule API’s.

1. Download [Mule standalone runtime](https://www.mulesoft.com/lp/dl/anypoint-mule-studio). It will be downloaded as a zip file.
2. Create a file in the same directory where mule runtime got downloaded, and name it as **Dockerfile**. This file will contain instructions for building a docker container image. Copy and paste the following content in **Dockerfile** and replace,

**<mule-runtime-zip-filename>** with the name of zip file that got downloaded.

*FROM openjdk:8-jdk*

*ENV MULE\_RUNTIME\_ZIP=<mule-runtime-zip-filename>*

*ENV MULE\_HOME=/opt/mule*

*ADD $MULE\_RUNTIME\_ZIP /opt*

*RUN set -x && cd /opt && unzip $MULE\_RUNTIME\_ZIP && mv mule-enterprise-standalone-\* mule && $MULE\_HOME/bin/mule -installLicense $MULE\_HOME/conf/$LICENSE\_FILE*

*WORKDIR $MULE\_HOME*

*VOLUME $MULE\_HOME/apps*

*VOLUME $MULE\_HOME/conf*

*VOLUME $MULE\_HOME/domains*

*VOLUME $MULE\_HOME/logs*

*#Check if Mule License installed*

*RUN ls -ltr $MULE\_HOME/conf/*

*CMD echo "------ License installed ! --------"*

*# HTTP Service Port*

*# Expose the necessary port ranges as required by the Mule Apps*

*EXPOSE 8082-8091*

*EXPOSE 9000*

*EXPOSE 9082*

*# Configure external access:*

*# HTTPS Port for Anypoint Platform communication*

*EXPOSE 443*

*# Mule remote debugger*

*EXPOSE 5000*

*# Mule JMX port (must match Mule config file)*

*EXPOSE 1098*

*# Mule MMC agent port*

*EXPOSE 7777*

*# AMC agent port*

*EXPOSE 9997*

*# Mule Cluster ports*

*EXPOSE 5701*

*EXPOSE 54327*

*# HTTP Service Port*

*EXPOSE 8081*

*# HTTPS Service Port*

*EXPOSE 8091*

*# Start Mule runtime*

*CMD echo "------ Start Mule runtime --------"*

*ENTRYPOINT ["./bin/mule"]*

1. Open a terminal, navigate to the directory containing **Dockerfile** and Mule Runtime zip file, and run the following command to build the docker container image after replacing **<image-tag-name>** with the appropriate values.

*docker build -t <image-tag-name> -f Dockerfile .*

1. Once the image is built, run the container by executing below command after replacing,

**<container-name>** and **<image-tag-name>** with the appropriate values

*docker run -d --name <container-name> -p 9000:9001 -i -t <image-tag-name>*

Note: -p 9000:9001: This option maps a port from the host to a port in the container. In this case, it maps port 9000 on the host to port 9001 in the container. The syntax is -p host-port: container-port.

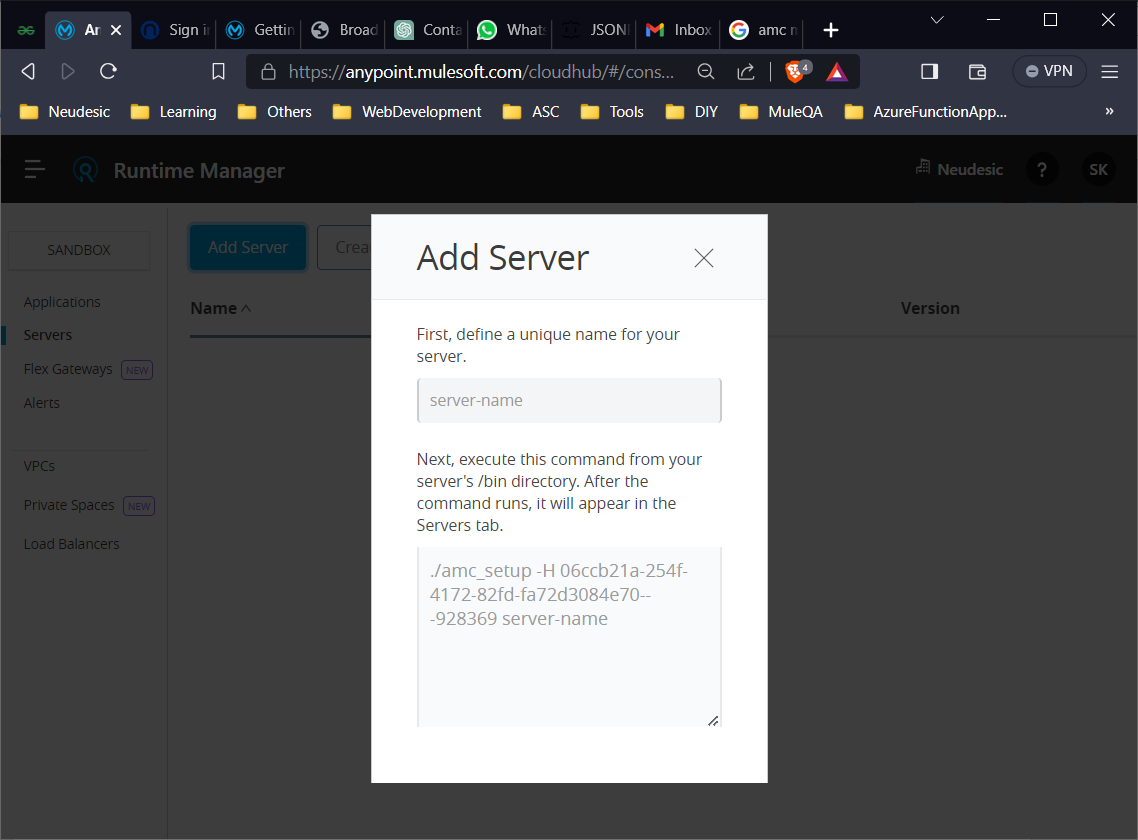
A computer screen shot of a program

Description automatically generated

## Installing runtime manager agent.

The objective of this section is to link our containerized mule runtime from above section with our Anypoint runtime manager by installing a runtime manager agent. Anypoint Runtime Manager agent registers Mule runtime engine (Mule) with Runtime Manager. After registration, a new server is created in Anypoint runtime manager which eventually will be used to deploy our mule API’s on containerized runtime.

1. Log in to Anypoint Platform and navigate to **Runtime Manager**. Click on the **Servers** tab on the left and select **Add Server**. This will open a screen similar to the one below. Enter the server’s name and copy the generated command.



1. Open terminal and execute the below command after replacing **<container-name>** with appropriate value. This will open an interactive container terminal.

*docker exec -it <container-name> /bin/sh*

1. Execute the following commands one after other. This will install runtime manager agent and registers Mule runtime engine (Mule) with Runtime Manager

*cd bin*

*<command-copied-from-above-step>*

1. Restart the Mule runtime container. This will start the server created in runtime manager.

*docker restart <container-name>*

1. Verify the server created in runtime manager.

A screenshot of a computer

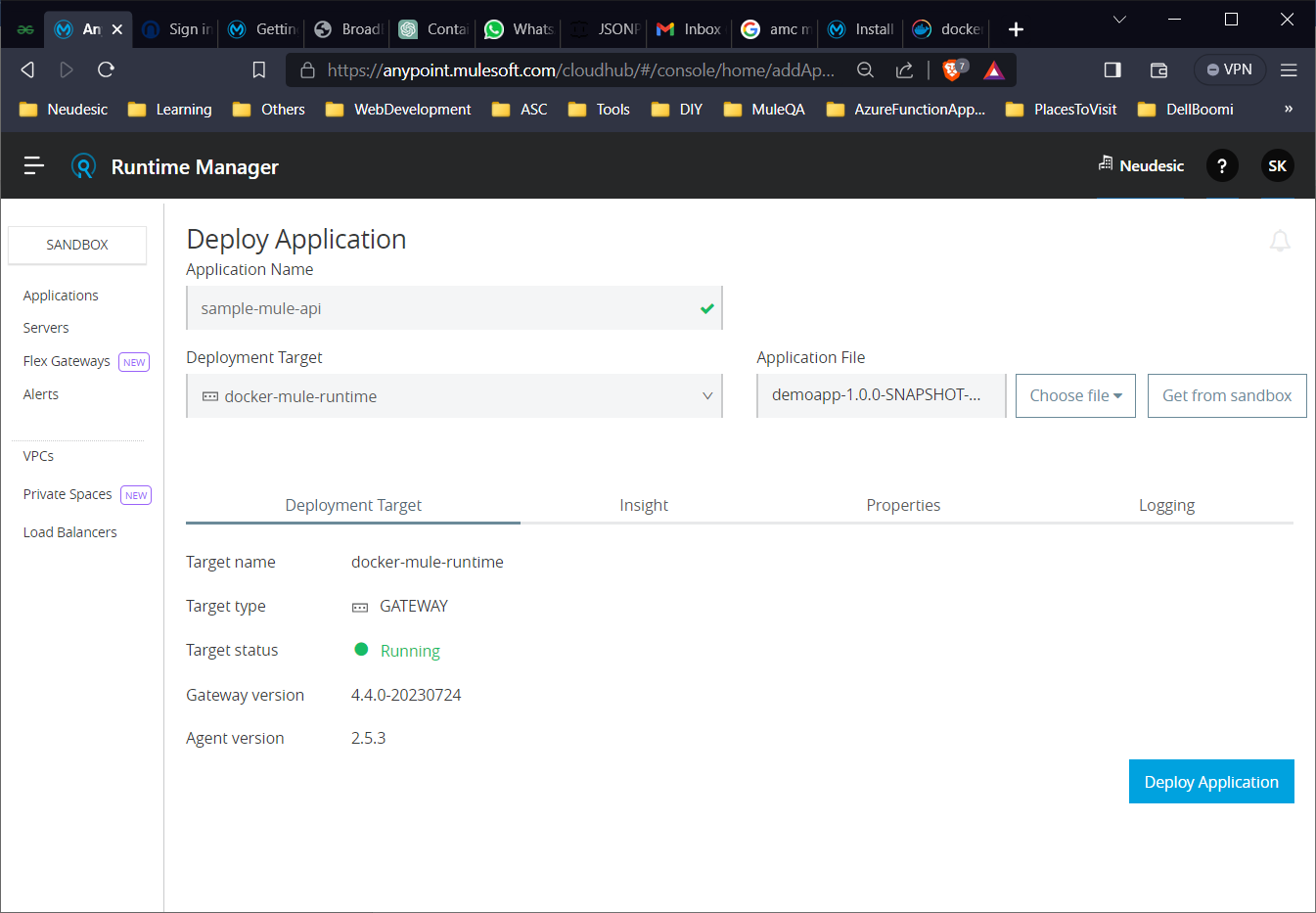
Description automatically generated

## Deploy Mule API and manage it via Flex gateway.

1. Log in to Anypoint Platform and navigate to **Runtime Manager**. Click on the **Applications** tab on the left and select **Deploy application**. This will open a screen similar to the one below. Enter the appropriate values, make sure the deployment target should be the server created a step before and then click on Deploy Application.

Note: Here we are deploying the jar below which is listening at port **9001**, on path as **/api/employees**. So, to make the http call we will follow [**http://localhost:9000/api/employees**](http://localhost:9000/api/employees). Port 9000 because container is mapping 9000 to 9001 already.





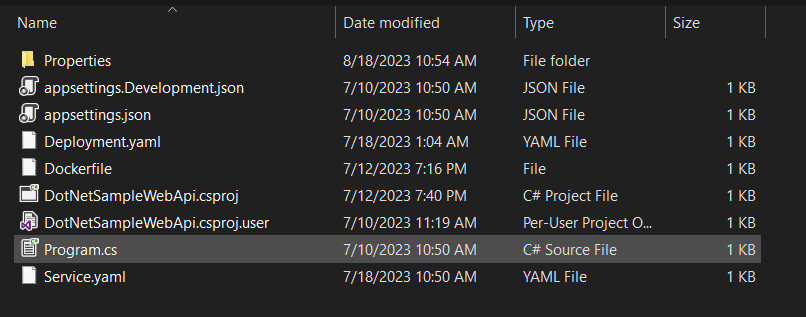
1. Once the application is deployed. Try to browse [**http://localhost:9000/api/employees**](http://localhost:9000/api/employees)**.**
2. In Anypoint Platform and navigate to **API Manager** 🡪 **Add API** 🡪 **Add new API**. On the Add API page, select **Flex gateway** in runtime and name of gateway from the gateway list and click on next.
3. Select **Create new API**, enter asset name, select asset types as **HTTP API** and click on next.
4. Under downstream configuration, select protocol as **HTTP**, port as **8081** and base path as **/** and click on next.
5. Under upstream configuration, enter **http://<local-ip>:9000**, after replacing <local-ip> with computer local Ip and click on next. On the next page, review and settings and click on **Save & Deploy**.
6. Once deployed, try to access **http://<local-ip>:9000/api/employees**. If this works, proceed with next step.
7. On the same page, click on **Policies** on the left pane and click on **Add policy**. Select **Basic Authentication – Simple** and click on next. Provide the username and password and click on apply.
8. Once policy is applied try to browse **http://<local-ip>:9000/api/employees**

## Deploy Non-Mule API and manage it via Flex gateway.

Objective of this section is to deploy a non-mule API to docker and manage it via Flex gateway.



Extract the above project and navigate till .csproj directory.



From this directory run the following commands in powershell.

*$ImageName = ‘dotnet-api’*

*$ContainerName = ‘dotnet-api-container’*

*dotnet restore*

*dotnet build*

*docker build -t $ImageName -f Dockerfile .*

*docker run -d --name $ContainerName -p 80:80 -i -t $ImageName*

Once above commands executes successfully, a container will spin up in docker having dotnet API, which can be accessed at <http://localhost/weatherforecast>

# Flex gateway demonstration using Kubernetes in connected mode

*$SubscriptionId = ‘269a893d-d53b-4f63-a49a-439b6eb31805’*

*$ResourceGroupName = ‘FlexGateway’*

*$AzureContainerRegistryName = ‘azconreg0001’*

*$AzureKubernetesServiceName = ‘azkubser0001’*

*az login*

*az account set --subscription $SubscriptionId*

*az group create --name $ResourceGroupName --location southindia*

*az acr create --name $AzureContainerRegistryName --resource-group $ResourceGroupName –location southindia --sku Basic*

*az aks create --resource-group $ResourceGroupName --name $AzureKubernetesServiceName --node-vm-size Standard\_B4ms --location southindia --attach-acr azconreg0001 --tier free --generate-ssh-keys*