

|  |  |
| --- | --- |
|  | Technical Document |
| Building K8S Managed Security and Integration Containers on the Google Cloud Platform (GCP)  GCP-K8S-Build |  |

# Description

The following guide discusses the deploying of the API Gateway and Live API Creator Security and Integration Solutions within the Google Cloud Platform (GPC) Kubernetes environment.

# Features

Use Kubernetes on GCP to Host Security and Integration Solutions:

* Easily Manage Creation and Deletion of Security and Integration Containers
* Automatically Configure the Security and Integration Solutions on the GCP Platform
* Utilize the GCP Platform to PoC and Demonstrate the Various Software Applications of the Security and Integration Solution

# Solutions

* API Gateway
* Live API Creator

Table of Contents

[Description 1](#_Toc7139876)

[Features 1](#_Toc7139877)

[Solutions 1](#_Toc7139878)

[Configuring the API Gateway and Live API Creator (LAC) within the Google Cloud Platform (GCP) - Kubernetes 3](#_Toc7139879)

[Create a Free GCP Account 3](#_Toc7139880)

[Create a New GCP Project 3](#_Toc7139881)

[Create a Kubernetes Cluster within GCP 5](#_Toc7139882)

[Create a GCP Firewall Rule 7](#_Toc7139883)

[Configure the Kubernetes Manifest Files 10](#_Toc7139884)

[Enter the Kubernetes Cluster’s Cloud Shell 10](#_Toc7139885)

[Download the Manifest Files from GitHub 11](#_Toc7139886)

[Edit the Deployment Script and LAC API Gateway Policy File 13](#_Toc7139887)

[Initialize the Security and Integration Containers 14](#_Toc7139888)

[Testing and Verifying the Newly Created Kubernetes Containers 15](#_Toc7139889)

[Validating the API Gateway Container 15](#_Toc7139890)

[Validating the LAC Container 17](#_Toc7139891)

[Validating the API Gateway LAC Integration Policy: 19](#_Toc7139892)

[Appendix A 21](#_Toc7139893)

[Updating API Gateway Config Manifest Files with a New API Gateway License 21](#_Toc7139894)

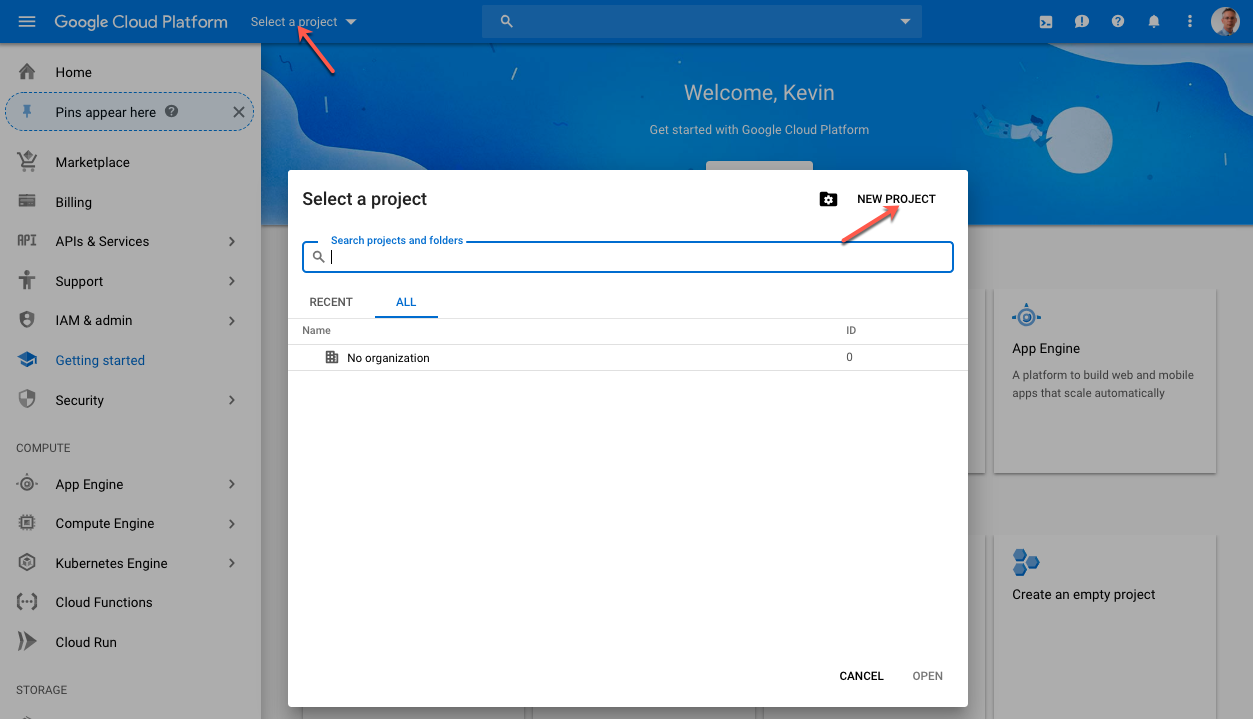
Configuring the API Gateway and Live API Creator (LAC) Security and Integration Solutions within the Google Cloud Platform (GCP) - Kubernetes

Create a Free GCP Account

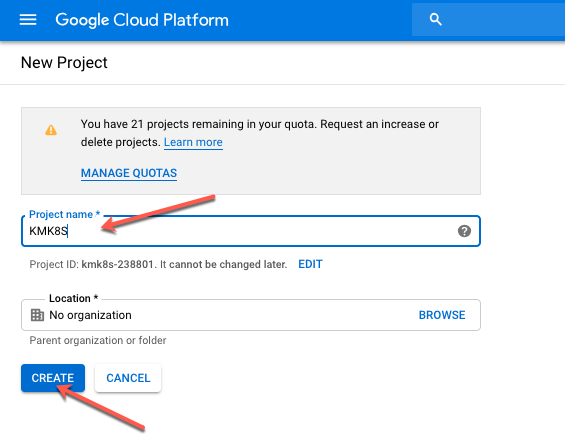
1. Open the following link in a Browser:
   1. <https://cloud.google.com/>
2. Click the Blue “Get started for free” button in the upper right-hand corner and follow the instructions
3. *Signing up for a free Google Cloud Platform account grants 12 months of free access up to $300.00. Because Kubernetes Containers are ephemeral in nature, it is recommended to delete the Kubernetes Cluster created during the use of this document after testing/demonstrating is complete to avoid any unwarranted charges. Simply follow the steps in this document to re-create the Cluster and start-up the containers when future testing/demonstration is needed.*

Create a New GCP Project

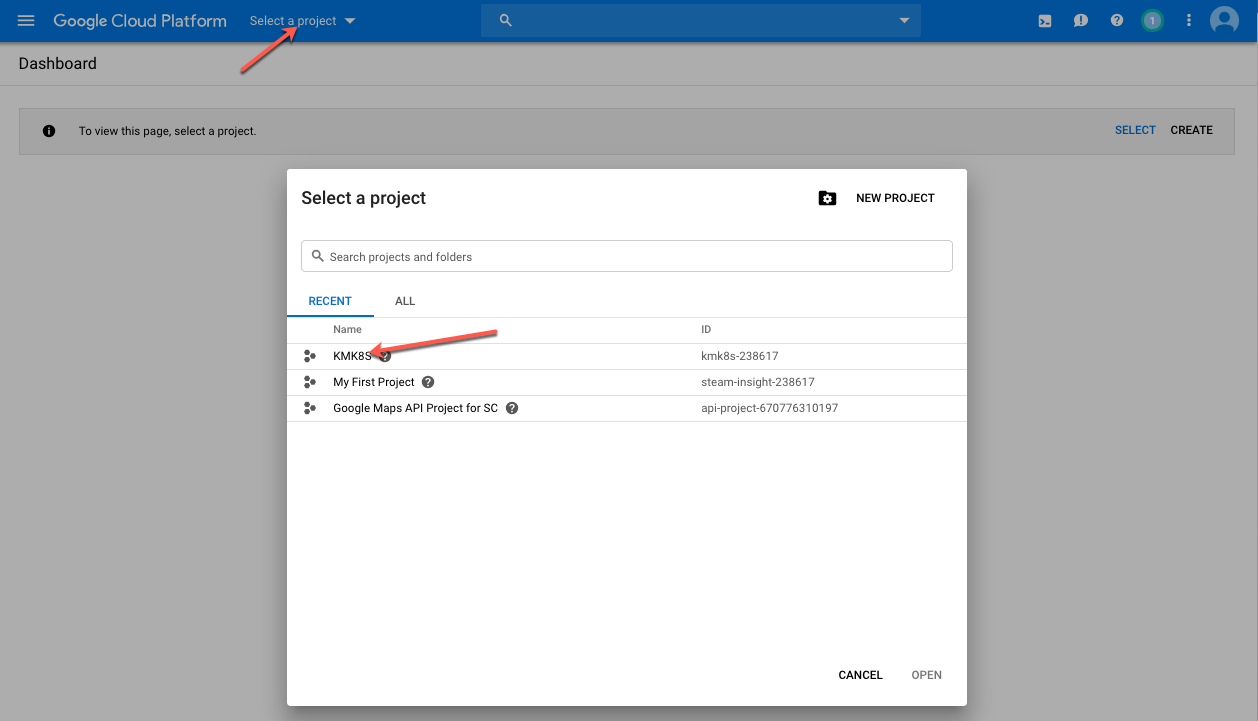
1. Login to the GCP Console
2. Click on the “Select a project” drop down to begin creation of a new project
3. Click “NEW PROJECT”
4. GCP Create New Project: Select a project



1. Give the project a name. In the below example the name is “KMK8S”
2. Set the Location to “No organization”
3. Click “CREATE”
4. GCP Create New Project: Name the New Project



1. Wait for the new Project to be created and select it from the “Select a Project” drop down if it is not already selected.
2. GCP Create New Project: Select the Newly Created Project

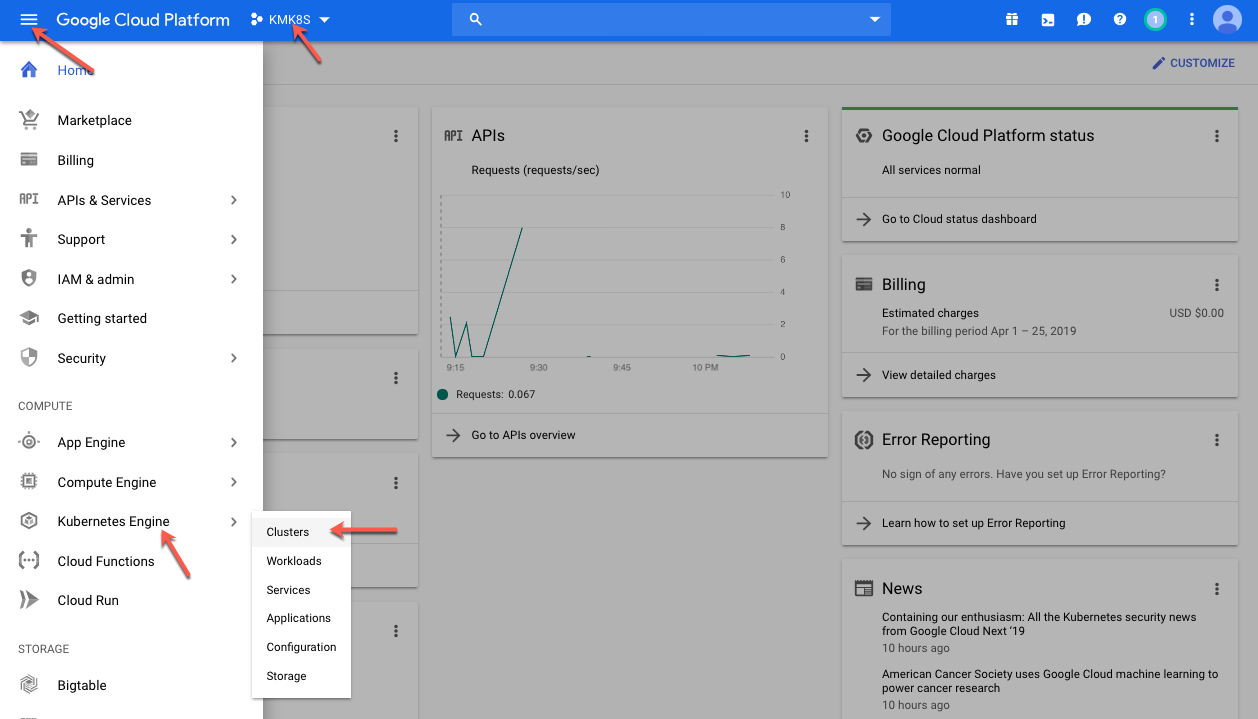


# Create a Kubernetes Cluster within GCP

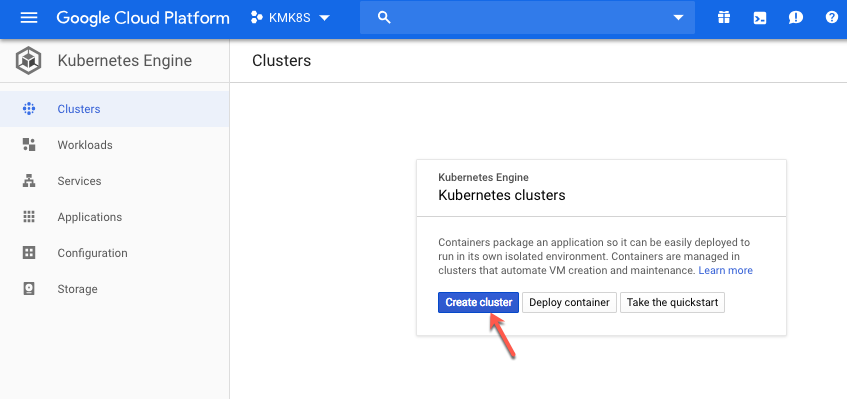
Creating a new Kubernetes Cluster allows GCP to host the Security and Integration Containers and have them managed by the Google Kubernetes Engine (GKE). Each of the Security and Integration Services will have their own Pods, i.e., the API Gateway and LAC Services will be hosted within their own Containerized Kubernetes Pod.

For the purposes of this guide, the Kubernetes cluster will be created with the following compute resources: One Node, 2 vCPUs and 7.5GB of RAM.

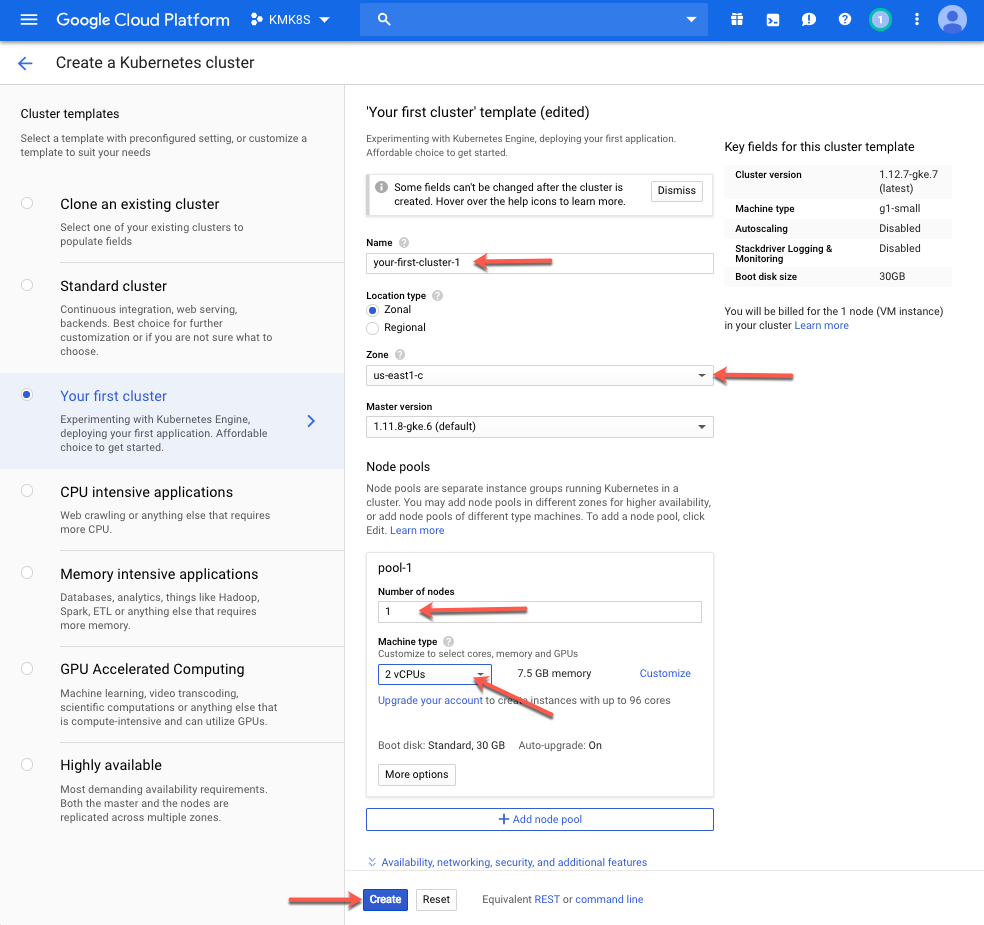
1. From the GCP Console make sure the newly created project is selected and Click the Hamburger icon in the upper left-hand corner beside the “Google Cloud Platform” title.
2. Scroll down to Kubernetes Engine 🡪 Clusters
3. K8S Cluster Creation: Kubernetes Engine 🡪 Clusters



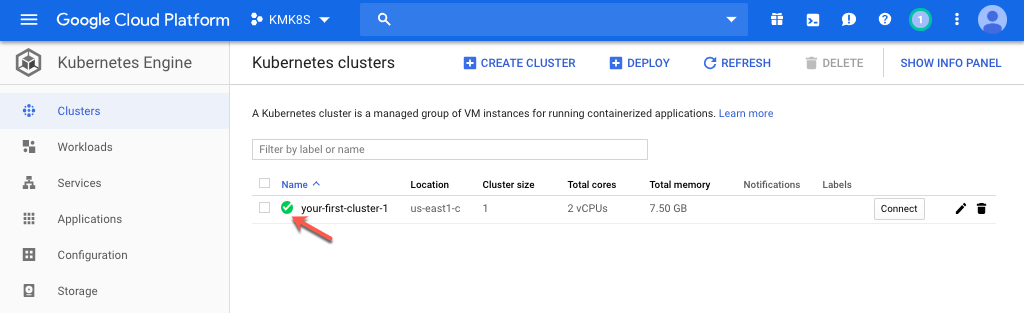
1. Click the “Create cluster” button
2. K8S Cluster Creation: Create Cluster



1. Select “Your first Cluster” on the left-hand side
2. Give the Cluster a Name or keep the Default Name
3. Select the Nearest Zone from the Zone Drop Down List. In this Example “us-east1-c” was selected
4. Change the Number of nodes to 1
5. Select 2vCPUs with 7.5 GB of memory from the Machine type Drop Down List
6. Leave all other options unchanged and click “Create”
7. K8S Cluster Creation: Cluster Configuration



1. Wait for the Cluster to complete its creation and turn green.
2. K8S Cluster Creation: Cluster Create Completion



Create a GCP Firewall Rule

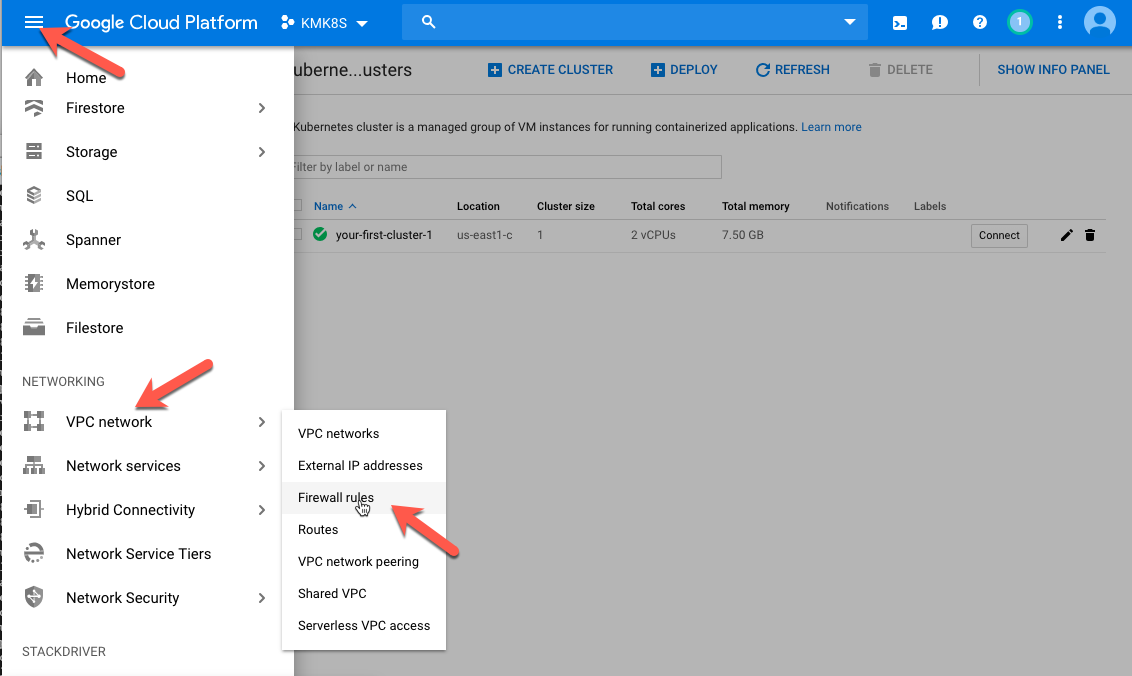
By default, incoming traffic is disabled for all requests to the GCP. In order to access the Kubernetes Containers, a Firewall Rule must be added to the VPC Network to enable access to the internal services such as the API Gateway Policy Manager or the LAC Administration Console.

For the purposes of this guide a Firewall Rule will be created to enable access to all incoming traffic on the following tcp ports: 32080, 31843 and 31080. These ports are the NodePorts exposed by the Manifest files for the Containers and will allow access to the API Gateway’s internal ports of 8080, 8443 and to the LAC Administration console port of 8080.

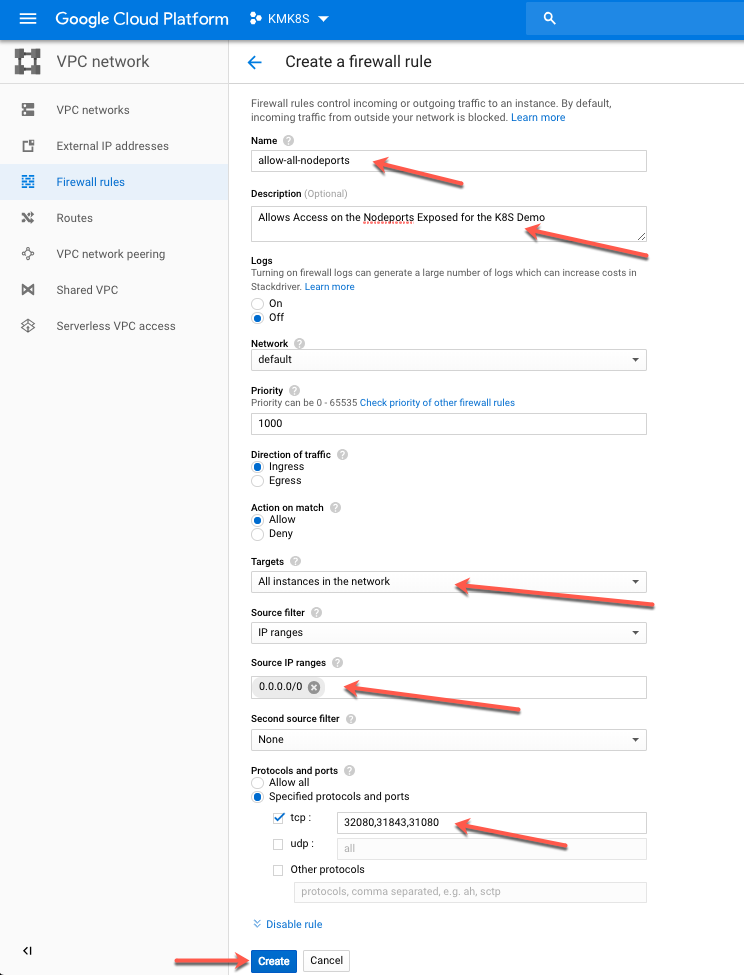
1. Kubernetes NodePort Map

|  |  |  |
| --- | --- | --- |
| Kubernetes Container | Exposed NodePort | Mapped Internal Port |
| API Gateway | 32080 | 8080 |
| API Gateway | 31843 | 8443 |
| Live API Creator | 31080 | 8080 |

1. From the GCP Console make sure the newly created project is selected and Click the Hamburger icon in the upper left-hand corner beside the “Google Cloud Platform” title.
2. Scroll down to VPC network and select VPC network 🡪 Firewall rules
3. Create Firewall Rule: Firewall Rules



1. Select “CREATE FIREWALL RULE” from the top and add the following details to the rule:
   1. Name: allow-all-nodeports
   2. Description: Allows Access on the Nodeports Exposed for the K8S Demo
   3. Targets: All instance in the network
   4. Source IP ranges: 0.0.0.0/0 or use your NAT’d IP Address discovered with whatsmyip
   5. Specified protocols and ports: Select the tcp box and type: 32080,31843,31080
2. Click “Create”
3. Create Firewall Rule: Configure Firewall Rule

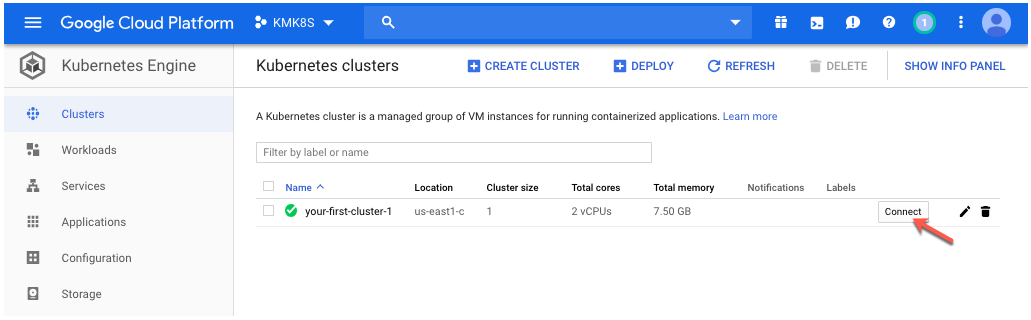


Configure the Kubernetes Manifest Files

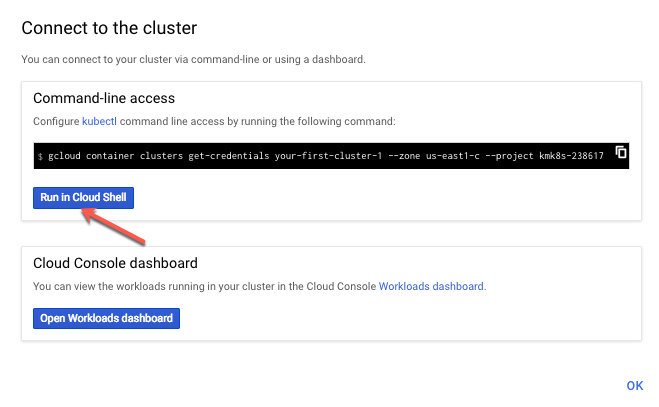
All of the necessary Manifest (YAML) files as well as shell scripts which initialize the containers from the Manifest files are hosted in GitHub. The following steps guide the reader through the process of downloading the files from GitHub and configuring them to work within the newly created Kubernetes Cluster.

Enter the Kubernetes Cluster’s Cloud Shell

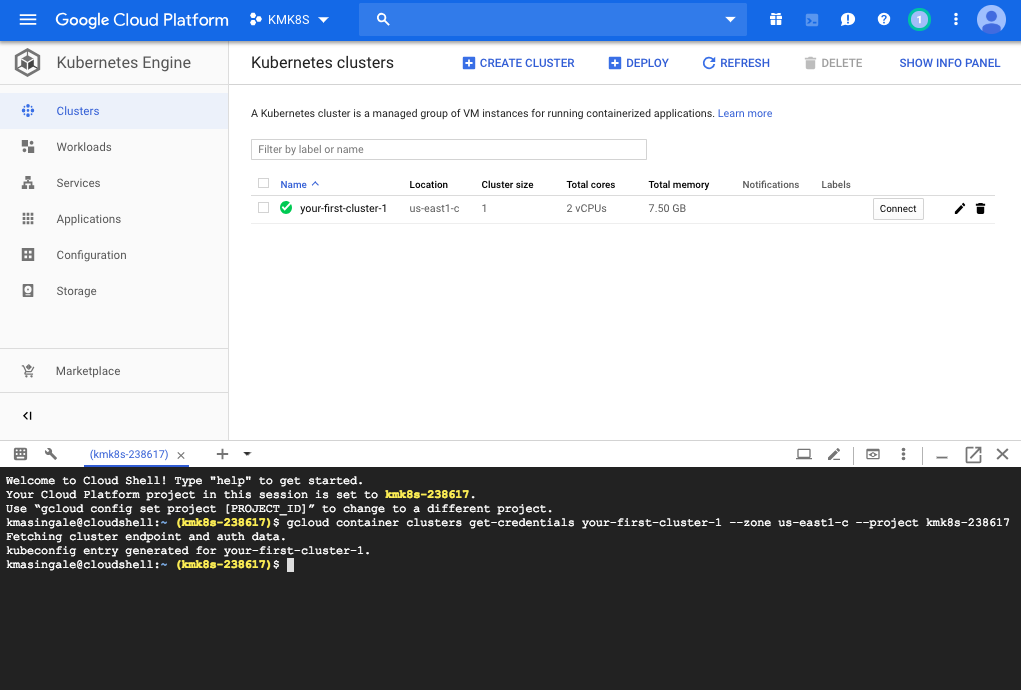
1. From the GCP Console make sure the newly created project is selected and Click the Hamburger icon in the upper left-hand corner beside the “Google Cloud Platform” title.
2. Scroll down and click on “Kubernetes Engine”
3. Connect to the Kubernetes cluster by clicking “Connect”
4. Connect to K8S Cluster: Connect



1. Click the “Run in Cloud Shell” Button
2. Connect to K8S Cluster: Connect To the Cluster

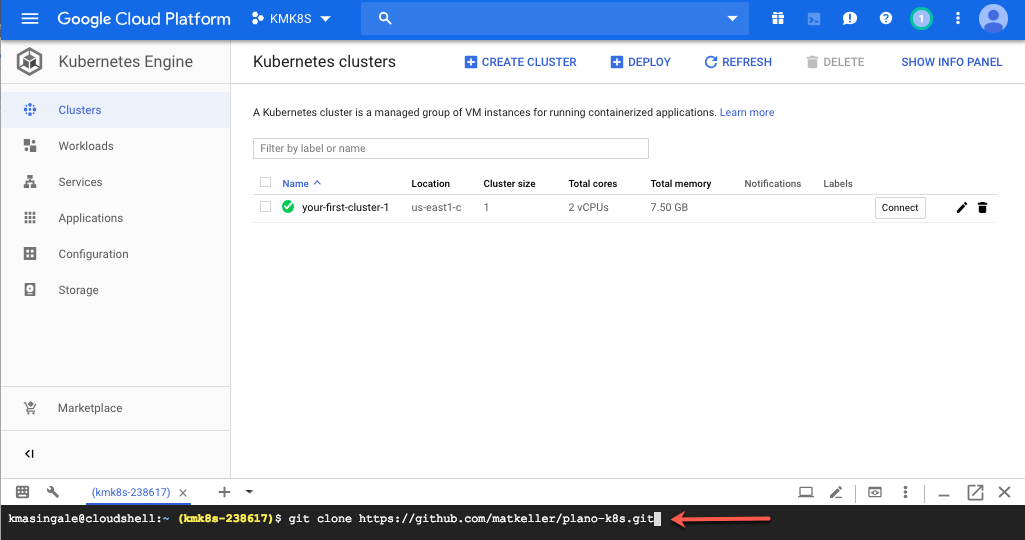


1. Press “return” on the keyboard to enter the Cloud Shell once the Shell is displayed at the bottom of the Browser
2. Connect to K8S Cluster: Enter Cloud Shell

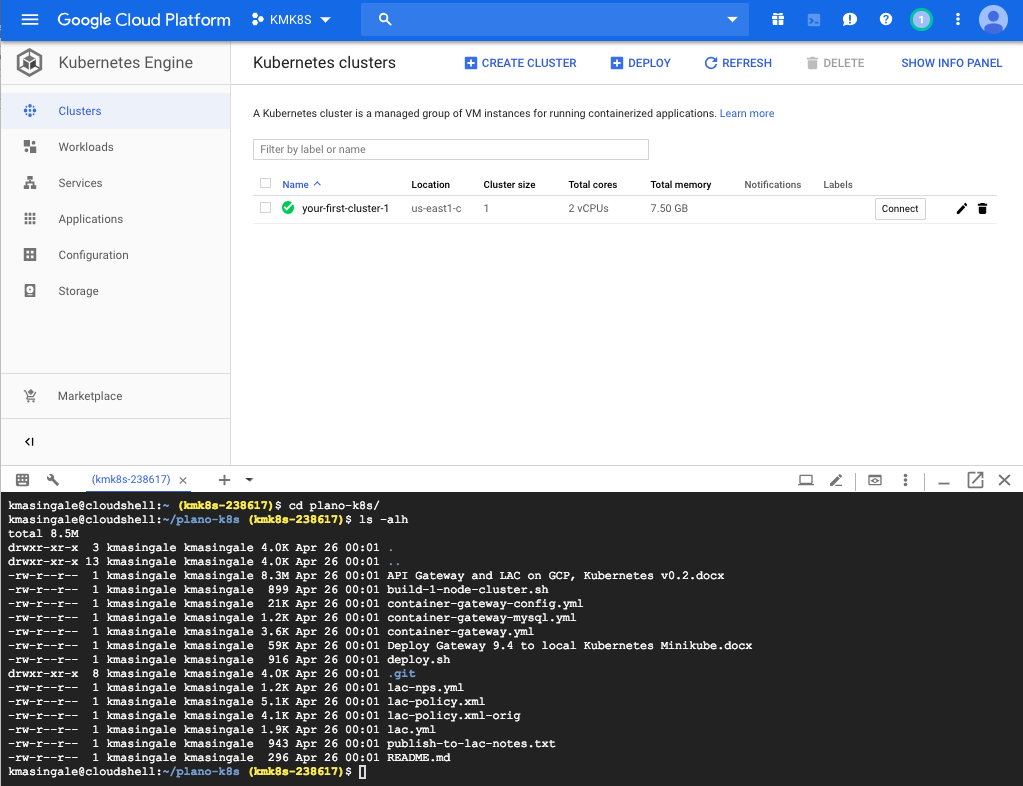


Download the Manifest Files from GitHub

1. Run the following command from within the Cloud Shell:
   1. git clone <https://github.com/matkeller/plano-k8s.git>
2. Manifest Files: Download



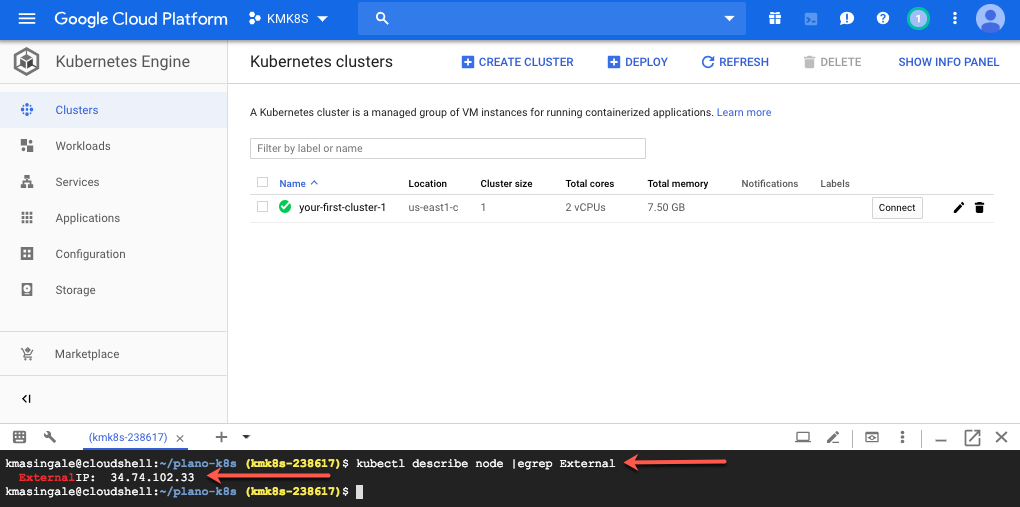
1. Verify the files have downloaded by running the following commands within the Cloud Shell:
   1. cd plano-k8s
   2. ls -alh
2. Manifest Files: Verification of Download



Edit the Deployment Script and LAC API Gateway Policy File

deploy.sh

1. Obtain the Kubernetes Cluster Node’s External IP Address by running the following command from the Cloud Shell:
   1. kubectl describe node |egrep External
2. Cluster Node: External IP Address



1. Make sure the current directory is the plano-k8s directory and run the following commands within the Cloud Shell to use the vi editor to open the deploy.sh script and replace the 192.186.99.100 placeholder address with the External IP Address noted above in step 1:
   1. vi deploy.sh
   2. :%s/192.168.99.100/34.74.102.33/
2. Press esc on the keyboard and type “:wq!” without the surrounding quotes to exit the vi editor and save the deploy.sh script.
3. Run the following command within the Cloud Shell to change the permissions on the deploy.sh and allow execution:
   1. chmod +x deploy.sh

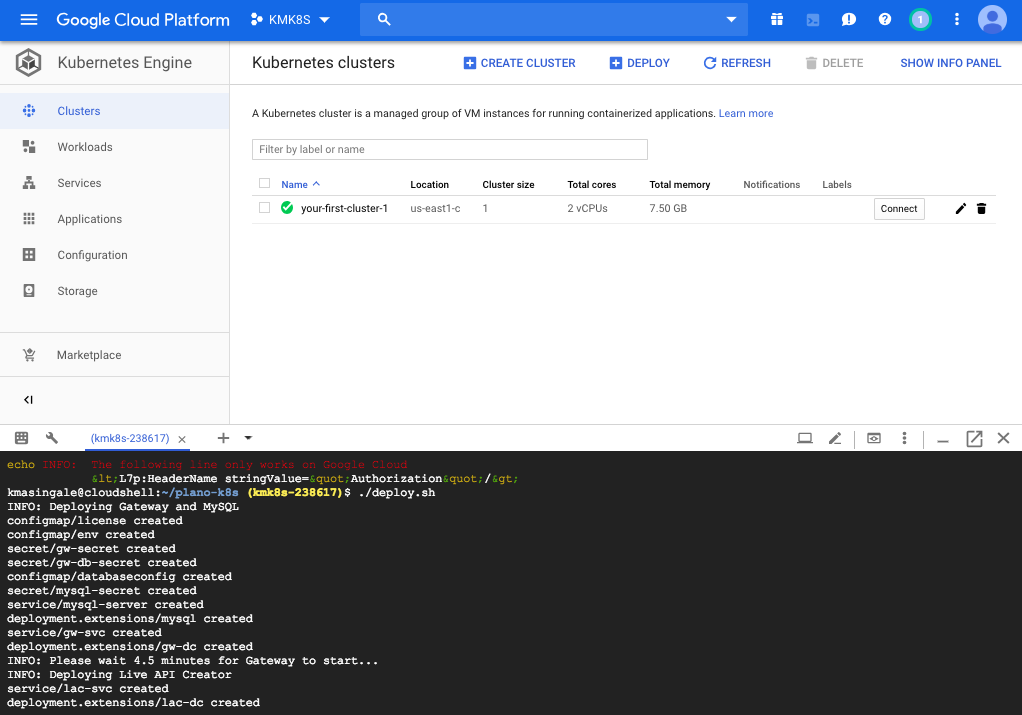
lac-policy.xml

1. Run the following commands within the Cloud Shell to use the vi editor to open the lac-policy.xml script and replace the 10.128.0.7 placeholder address with the External IP Address noted above.
   1. vi lac-policy.xml
   2. :%s/10.128.0.7/34.74.102.33/
2. Press esc on the keyboard and type “:wq!” without the surrounding quotes to exit the vi editor and save the lac-policy.xml file.

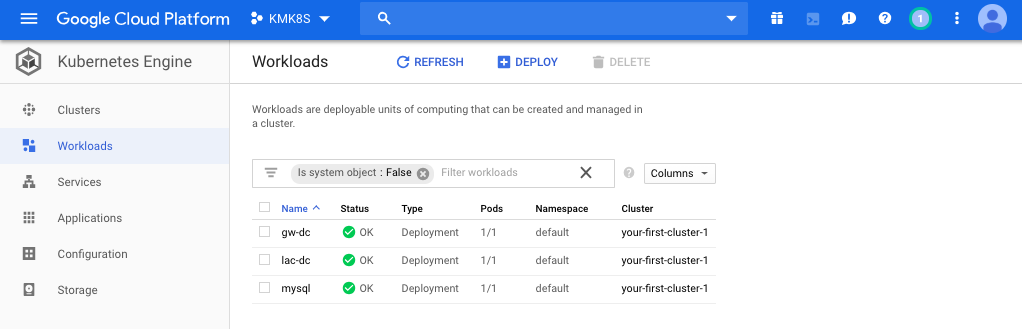
Initialize the Security and Integration Containers

Initializing the Containers using the downloaded Manifest files and Deployment script will create multiple containers within the GCP Kubernetes Cluster that will be externally accessible and easily taken down and rebuilt upon demonstration or testing.

1. From within the same Cloud Shell as above, make sure the current directory is the plano-k8s directory and run the following command:
   1. ./deploy.sh
2. Initialize Containers: Run Deployment Script



1. If all goes well, 3 Workloads will be created with a status of “OK” and a Green Icon:
2. Initialize Containers: Successful Workloads



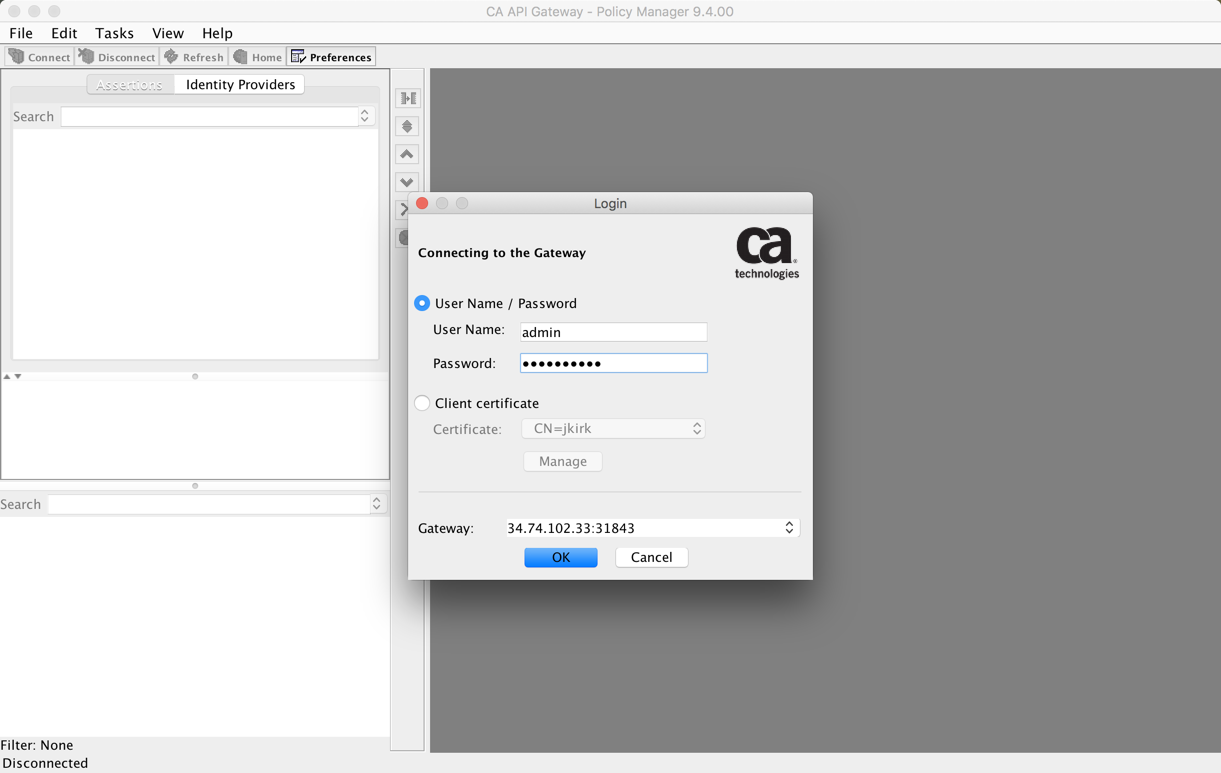
Testing and Verifying the Newly Created Kubernetes Containers

Once the Containers are up and running they should be verified by testing the different components. This section will validate the installation of the Gateway and LAC Containers by connecting to Policy Manager and the LAC Administration Console as well as test the lac-policy installed by the deployment script.

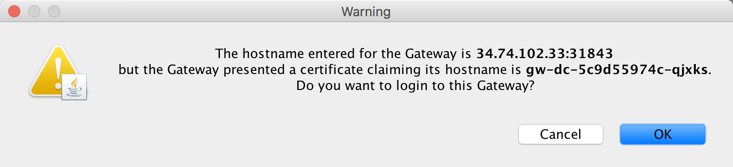
Validating the API Gateway Container

Launching Policy Manager

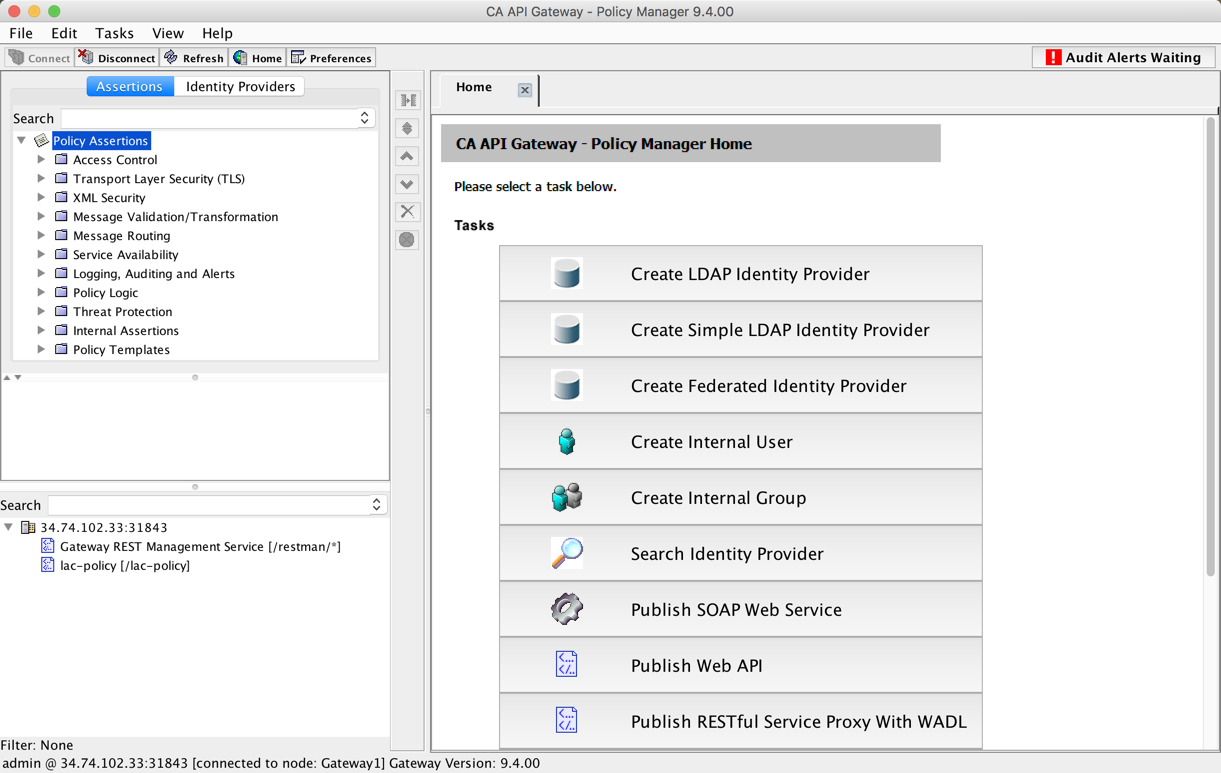
1. Launch Policy Manager
2. Login using the following information:
   1. User Name: admin
   2. Password: CAdemo123!
   3. Gateway: 34.74.102.33:31843 – ***Note: Replace the IP address with the External IP Address gathered earlier***
3. Policy Manager: Launching



1. Ignore the certificate warning and click “OK”:
2. Policy Manager: Ignore Certificate Warning



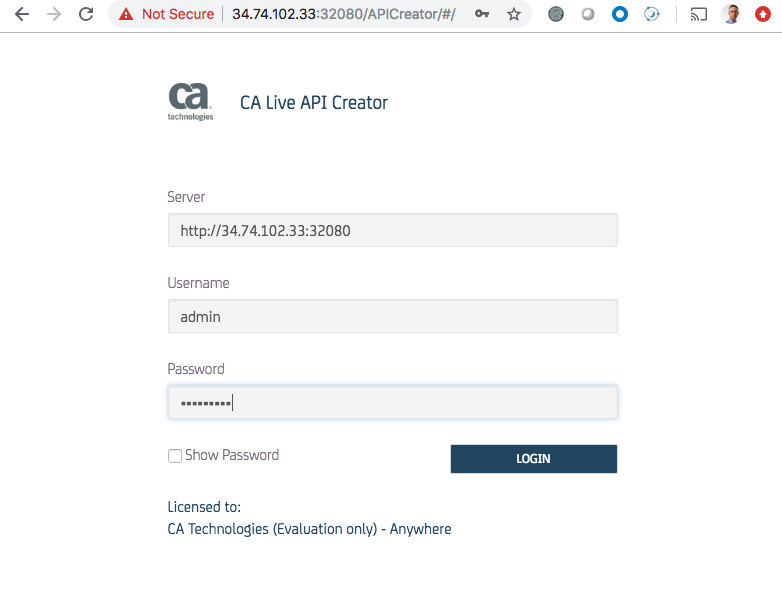
1. The following UI will be displayed if a connection to the API Gateway Container was successful:
2. Policy Manager: Landing



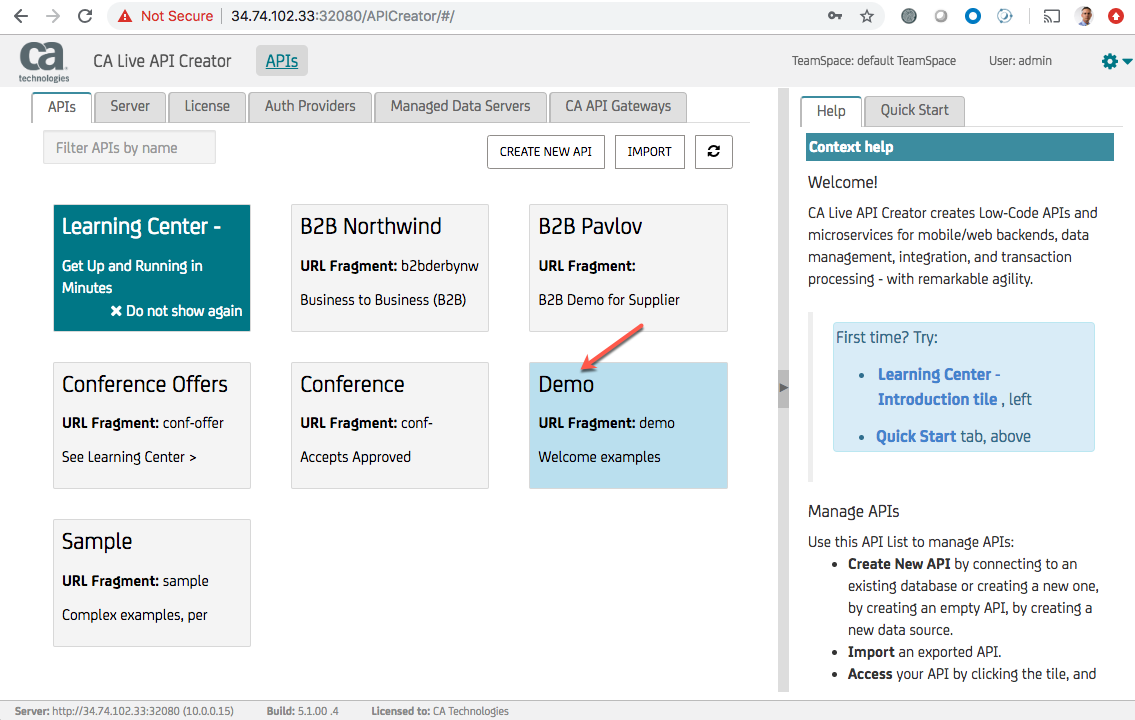
Validating the LAC Container

Launching LAC Administration Console

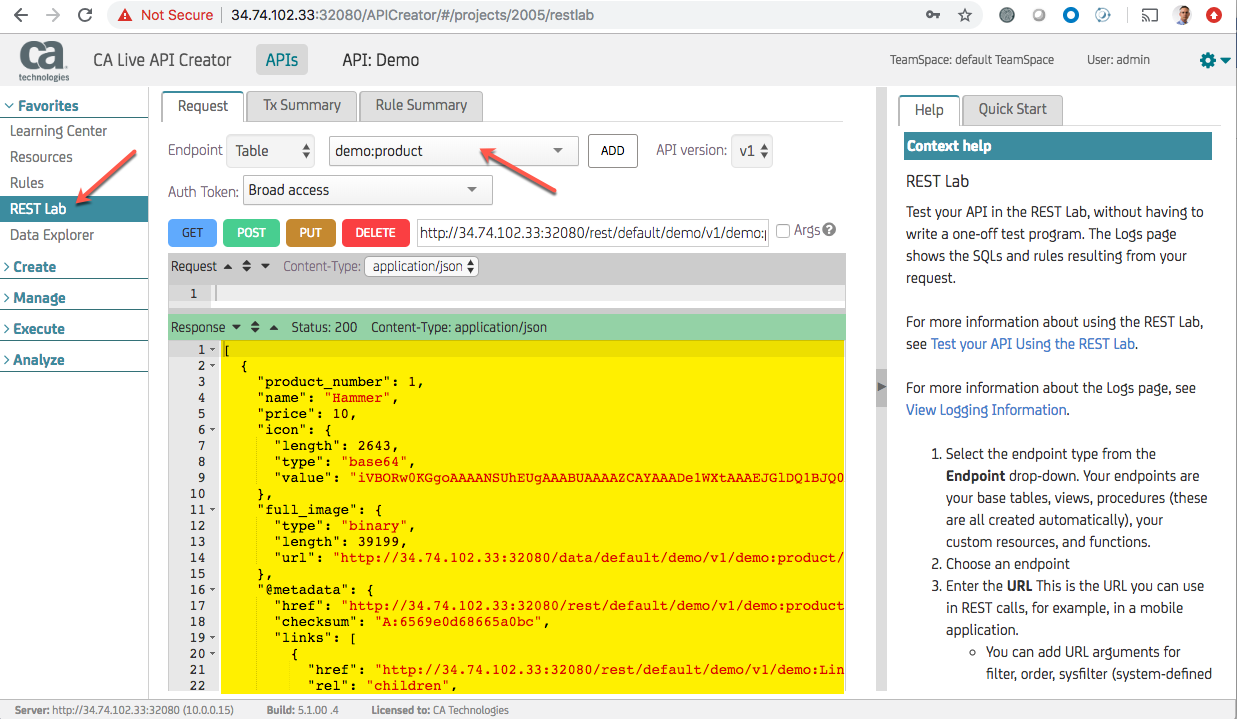
1. Launch the LAC Administration Console from a Web Browser using the following address:
   1. <http://34.74.102.33:32080/APICreator/#/> – ***Note: Replace the IP address with the External IP Address gathered earlier***
2. Login using the following information:
   1. User Name: admin
   2. Password: Password1
3. LAC Admin Console: Login Page



1. Select the Demo API
2. LAC Admin Console: Landing Page



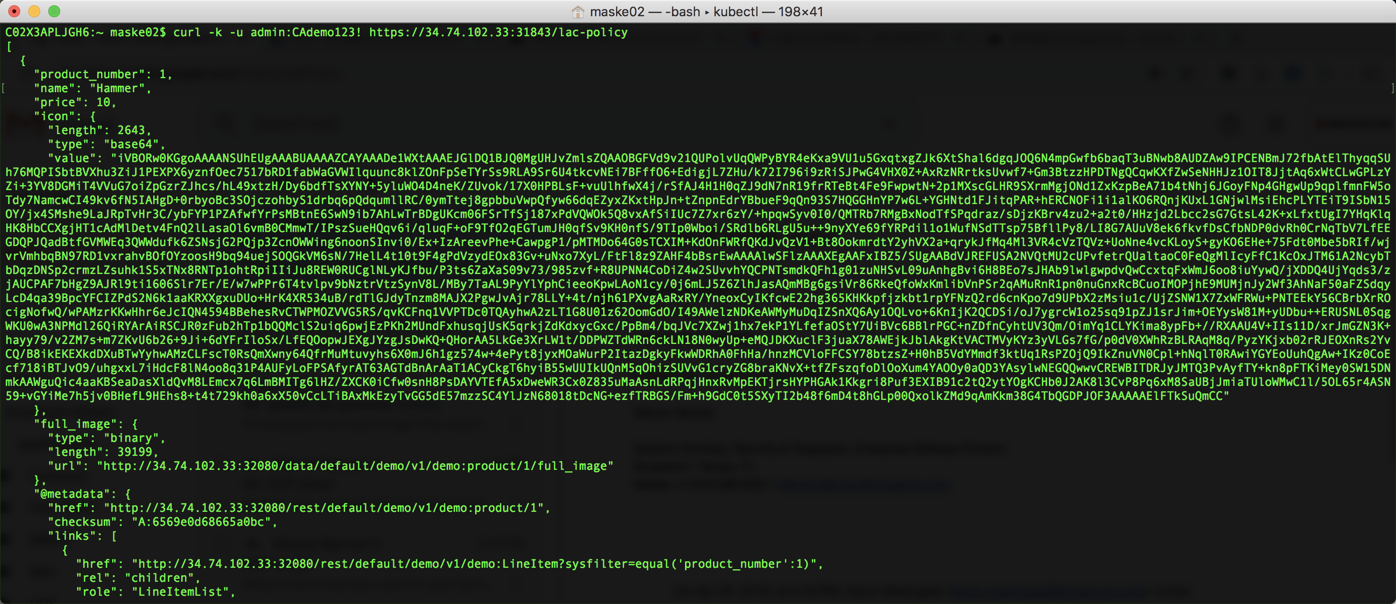
1. Select “Rest Lab” on the Left-Hand Side
2. Select “demo:product” from the Endpoint drop-down list
3. Click “Get” – If all is well with the LAC Container, the following output will be displayed:
4. LAC Admin Console: Rest Lab Get Response



Validating the API Gateway LAC Integration Policy:

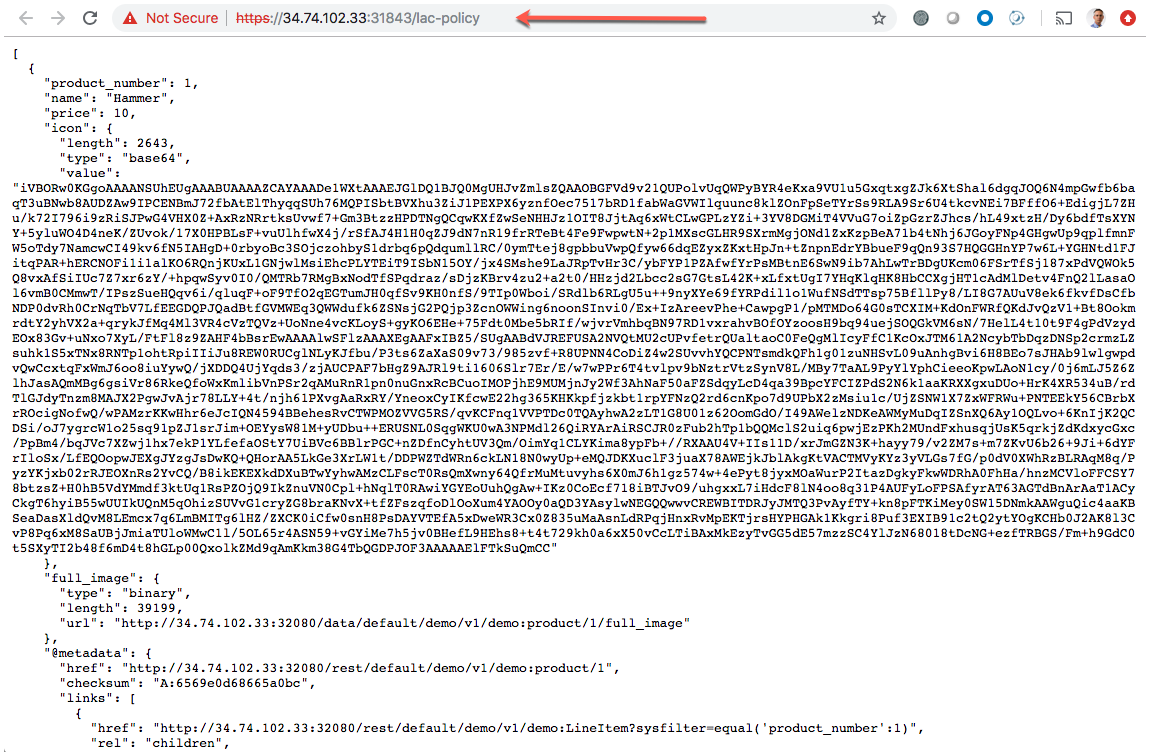
Validating via Curl

1. Run the following curl command from either the Cloud Shell or a Terminal Window to test the API Gateway lac-policy:
   1. curl -k -u admin:CAdemo123! <https://34.74.102.33:31843/lac-policy> – ***Note: Replace the IP address with the External IP Address gathered earlier***
   2. The output returned should be the same as the output highlighted in yellow above in the LAC Rest Lab
2. Validate LAC Policy: Curl Results



Validating via Browser

1. Using a Browser, connect to the following address and credentials:
   1. <https://34.74.102.33:31843/lac-policy> – ***Note: Replace the IP address with the External IP Address gathered earlier***
   2. User Name: admin
   3. Password: CAdemo123!
   4. The output returned should be the same as the output highlighted in yellow above in the LAC Rest Lab and the output from the Curl Command
2. Validate LAC Policy: Browser Results

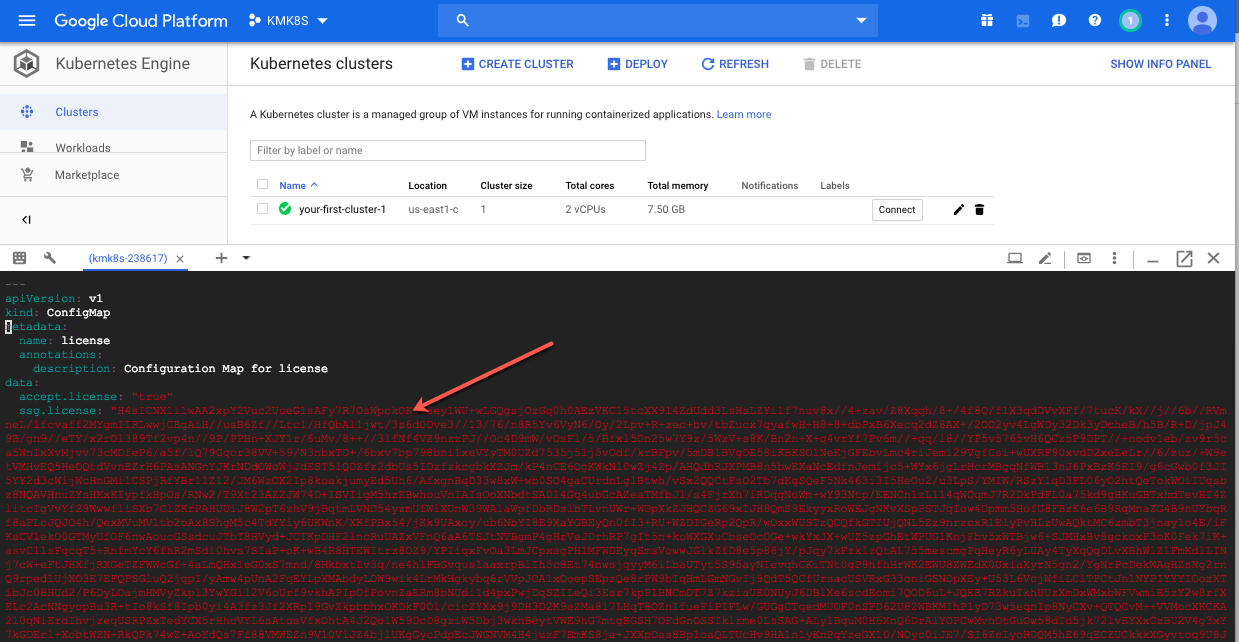


Appendix A

Updating API Gateway Config Manifest Files with a New API Gateway License

The current GitHub project includes a container-gateway-config.yml file that incorporates a license that will expire in 12/06/2019. If this document is utilized beyond 12/06/2019, the Gateway Kubernetes Container/Workload will not build successfully. The container-gateway-config.yml file must be updated with the following steps in order to utilize this guide.

1. This section assumes that a new API Gateway License file has been requested from Support, downloaded, unzipped and renamed to “license.xml”. Also note that the license file should be in Raw XML format.
2. Run the following commands within a terminal window from the directory where the license.xml file resides to get a gzipped and base64 version of the license file:
   1. gzip license.xml
   2. openssl base64 -e -A -in license.xml.gz -out license.xml.gz.base64
3. Open the license.xml.gz.base64 file using a text editor carefully copy the contents.
4. Using vi within the Cloud Shell carefully paste the previously copied contents into the container-gateway-config.yml file replacing the value after the “ssg.license” key. Make sure to encapsulate the pasted contents in a pair of double-quotes the way the existing line was encapsulated.
5. container-gateway-config.yml: ssg.license Line





Broadcom, the pulse logo, Connecting everything, CA Technologies, and the CA technologies logo are among the trademarks of Broadcom and/or its affiliates in the United States, certain other countries, and/or the EU.

Copyright © 2019 by Broadcom. All Rights Reserved.

The term “Broadcom” refers to Broadcom Inc. and/or its subsidiaries. For more information, please visit [www.broadcom.com](http://www.broadcom.com).

Broadcom reserves the right to make changes without further notice to any products or data herein to improve reliability, function, or design. Information furnished by Broadcom is believed to be accurate and reliable. However, Broadcom does not assume any liability arising out of the application or use of this information, nor the application or use of any product or circuit described herein, neither does it convey any license under its patent rights nor the rights of others.