

# Google Compute Engine Driver

Version: 1.0

Date: 20160307

Source: <https://github.com/TuxInvader/BrocadeCloud>

## Table of Contents

Google Compute Engine Driver.....	1
Auto-Scaling Primer.....	2
The Google Driver.....	2
Pre-requisites.....	2
Deployment Steps.....	3
Upload the driver.....	3
Create Cloud Credentials.....	3
Create your pool.....	4
Authentication.....	5
Authentication via Metadata Server.....	5
Authentication via OAUTH2.....	7
Google Auto-Scaling FAQ.....	11
1. How do I create my image?.....	11
2. How can I test the autoscaler?.....	13

## Auto-Scaling Primer

Auto-Scaling is a feature in Brocade vTM which allows the Traffic Manager to automatically scale your pool of servers on demand. The Auto-Scaler process constantly monitors the availability and responsiveness of your nodes, and ensures the pool is always “right-sized” for the current demands.

When the Auto-Scaler decides that the pool is not sized correctly for the current load, it will take an action to rectify the situation. If the Auto-Scaler determines that you have more nodes running than are needed, then it will take action to remove one of the nodes. When it determines that the pool has too few nodes to serve the current load, then it will take action to create a node.

The Auto-Scaler makes use of drivers to help it integrate with 3<sup>rd</sup> party cloud APIs. The Auto-Scaler makes the decisions, but then uses the driver to make the required changes in the cloud. Typically a driver provides three core functions:

- status - reports the current nodes deployed and their state
- createnode - used by the Auto-Scaler to deploy a new node into the cloud
- destroynode - used by the Auto-Scaler to remove a node from the cloud

Some cloud drivers provide additional functionality, which can be used by the system administrator or by other vTM sub-systems. However the Auto-Scaler itself only uses the three described above.

For more information on Auto-Scaling, please refer to the Brocade vTM User Guide.

## The Google Driver

The Google driver implements the three required Auto-Scaling functions: status, createnode, and destroynode, for integration with Google Compute Engine.

Besides the Auto-Scaling functions, the driver can also be used to:

- authclient – function to generate authentication details for using OAUTH2.

For a full list of actions and their arguments, execute the driver with no parameters.

## Pre-requisites

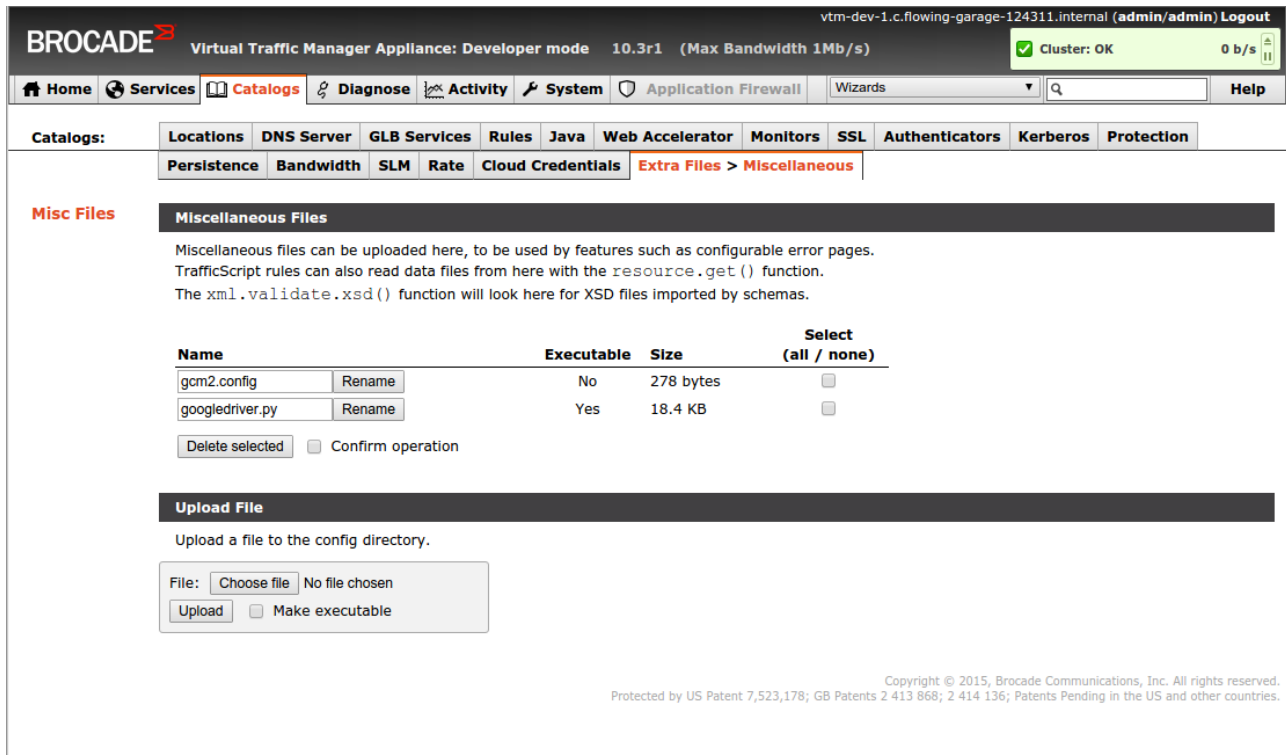
The python functions used by the Auto-Scaler require only standard python libraries, and so it will work out-of-the-box on a standard Brocade vTM appliance or base Linux OS install. We import only: sys, os, re, json, requests and time.

The function “authclient” requires OAuth2client in order to generate the token authorization URL and retrieve your access and refresh tokens from Google. However you only need to run that process once, and it doesn't have to be run on the vTM.

# Deployment Steps

## 1. Upload the driver

The first step is to upload the driver to the Extra Files catalogue of your vTM. Navigate to Catalogs → Extra Files → Miscellaneous and upload the googledriver.py file, remembering to mark it as executable.



The screenshot shows the Brocade vTM web interface. The top navigation bar includes 'Home', 'Services', 'Catalogs', 'Diagnose', 'Activity', 'System', 'Application Firewall', 'Wizards', and 'Help'. The 'Catalogs' section is active, showing a list of categories: 'Locations', 'DNS Server', 'GLB Services', 'Rules', 'Java', 'Web Accelerator', 'Monitors', 'SSL', 'Authenticators', 'Kerberos', and 'Protection'. Under 'Extra Files > Miscellaneous', there is a 'Misc Files' section. It contains a table of files:

Name	Executable	Size	Select (all / none)
gcm2.config	No	278 bytes	<input type="checkbox"/>
googledriver.py	Yes	18.4 KB	<input type="checkbox"/>

Below the table, there are buttons for 'Delete selected' and 'Confirm operation'. An 'Upload File' section is also visible, with a 'File:' field containing 'Choose file' and 'No file chosen', and an 'Upload' button with a 'Make executable' checkbox.

If you have chosen to use OAUTH2 authentication (See Authentication), then you will need to upload your OAUTH JSON file here too. If the vTM is running in GCE, then this step is optional.

## 2. Create Cloud Credentials

Navigate to Catalogs → Cloud Credentials and create a new configuration. Select the googledriver.py from the “Cloud API” drop down.

Configure the credentials for your environment:

- Credential 1 – Set this to either “local” or the name of your OATH JSON file
- Credential 2 – Set the to the name of your project.
- Credential 3 – Set this to the Region in which you are deployed.

The screenshot shows the Brocade VTM web interface. At the top, it displays 'BROCADE Virtual Traffic Manager Appliance: Developer mode 10.3r1 (Max Bandwidth 1Mb/s)'. The navigation menu includes Home, Services, Catalogs, Diagnose, Activity, System, Application Firewall, and Wizards. The 'Catalogs' section is active, showing tabs for Locations, DNS Server, GLB Services, Rules, Java, Web Accelerator, Monitors, SSL, Authenticators, Kerberos, and Protection. Under 'Catalogs', there are sub-tabs for Persistence, Bandwidth, SLM, Rate, Cloud Credentials, and Extra Files. The 'Cloud Credentials Catalog' is selected, showing a message: 'The Cloud Credentials Catalog contains sets of user credentials used by Autoscaling to create and destroy nodes. No Cloud Credentials have been created.' Below this is a 'Create new Cloud Credentials' form with the following fields:
 

- Name: Google
- Cloud API: googledriver.py
- Credential 1: local
- Credential 2: [Redacted]
- Credential 3: [Redacted]

 A 'Create Cloud Credentials' button is at the bottom of the form.

### 3. Create your pool

Navigate to Services → Pools and create a new pool. Check the box to indicate that this will be an auto-scaled pool. You will be taken to the Auto-Scaling configuration.

The screenshot shows the 'Autoscaling' configuration page for a pool named 'autopool (not used, 0 nodes)'. The page title is 'Autoscaling' and it includes an 'Unfold All / Fold All' button. The main content area is titled 'Autoscaling settings' and contains the following configuration options:
 

- Are the nodes of this pool subject to autoscaling? If yes, nodes will be automatically added and removed from the pool by the chosen autoscaling mechanism.
  - autoscale!enabled:  Yes  No
- Whether or not autoscaling is being handled by an external system. Set this value to Yes if all aspects of autoscaling are handled by an external system, such as RightScale. If set to No, the traffic manager will determine when to scale the pool and will communicate with the cloud provider to create and destroy nodes as necessary.
  - autoscale!external:  Yes  No
- Select the Cloud API and account you want to create nodes with.
  - Cloud Credentials: Google (googledriver.py)
  - Manage Cloud Credentials
- The identifier that tells your custom cloud API provider what type of nodes to create.
  - Image ID: lamp-stack
- The identifier that tells your custom cloud API provider what size/type of machine to create.
  - Machine Type: n1-standard-1
- Prefix to add to the name of a new node, if this custom cloud API provider supports this.
  - Name Prefix: lamp

Autoscale!enabled should be “Yes”, and Autoscale!external should be “No”.

You must also provide an “Image ID” to be used as the image for deploying machines, a Machine Type, and also a prefix for the nodes.

You will also need to change the autoscale!ipstouse setting to “Private IP Addresses”.

Review the other settings for the pool, and when you're happy; click update.

The Auto-Scaler will now deploy nodes in accordance with your settings. After a minute or two, you should see message in the Event Log indicating that a new machine has been deployed.

The Event Log shows the contents of the error logs on each traffic manager machine.

Timescale: Last N events 15 Event Filter: All Events Show 'Configuration modified' messages: Yes No

Stop updating Display Download

07/Mar/2016:14:06:37 +0000	INFO	Pool autopool: Pool now has working nodes	vtm-dev-1
07/Mar/2016:14:06:37 +0000	INFO	Pool autopool: Autoscaler has updated pool autopool: it now has 1 node	vtm-dev-1
07/Mar/2016:14:06:37 +0000	INFO	Pool autopool: An autoscaled pool's state has changed from 'growing' to 'refractory'; have 0 nodes min=1 max=4 conf=100	vtm-dev-1
07/Mar/2016:14:06:37 +0000	INFO	Pool autopool: Creation of new node with id '12343026058871893798' is now complete; status: 'active'; public address: 104.155.84.210; private address: 10.132.0.4	vtm-dev-1
07/Mar/2016:14:06:37 +0000	INFO	Status change for node '12343026058871893798' in autoscaled pool: status: 'pending' -> 'active'	vtm-dev-1
07/Mar/2016:14:06:33 +0000	INFO	Autoscaling added IP address 10.132.0.4	vtm-dev-1
07/Mar/2016:14:06:08 +0000	INFO	Pool autopool: Creation of new node instigated	vtm-dev-1
07/Mar/2016:14:06:08 +0000	INFO	Pool autopool: An autoscaled pool's state has changed from 'below minimum' to 'growing'; have 0 nodes min=1 max=4 conf=100	vtm-dev-1
07/Mar/2016:14:06:08 +0000	WARN	Pool autopool: Minimum size undercut - growing	vtm-dev-1

## Authentication

The Auto-Scaling driver needs to authenticate its requests to the compute API using an `access_token`. There are two ways for the driver to get hold of an `access_token`.

The simplest method is to request the token from the Metadata server; only available if the driver is being run inside the Compute Engine Cloud.

The alternative is to create a set of OAUTH2 credentials, and then authorize the autoscaling driver to use them. The OAUTH2 approach takes some extra time to set up, but it means the driver can potentially run outside of the Compute Engine.

## Authentication via Metadata Server

When your vTM is running inside the Compute Engine it can make use of the “default” service account in order to perform scaling actions. However you must deploy the vTM with the correct read/write access to the compute engine API. If you are deploying the vTM using the Compute Engine UI, then you should either:

- Project Access Tick box “Allow API Access to all Google Cloud Services in the same project”
- Or enable “Read/Write” for Compute under “Access & Security”

The screenshot shows the Google Cloud Platform console for 'Compute Engine' under 'VM instances'. The main content area is titled 'VM instances' and includes a warning about TCP ports for Developer vTM (1M). The 'Project access' section is highlighted in yellow, showing a checkbox for 'Allow API access to all Google Cloud services in the same project'. Below this, there are tabs for 'Management', 'Disks', 'Networking', and 'Access & security'. The 'SSH Keys' section has a text input field for 'Username' and an 'Add item' button. The 'API access' section is also highlighted in yellow, showing a dropdown menu for 'Compute' set to 'Read Write'. The right sidebar contains 'About Developer vTM (1M)', 'Technical details' (Type: v1, Version: 10, Last updated: 3/4), 'Documentation', and 'Terms of Service'.

If you are deploying your vTM using the gcloud cli tool, then you need to ensure that you pass in the read/write compute engine scope (see --scopes). Something like this should work fine:

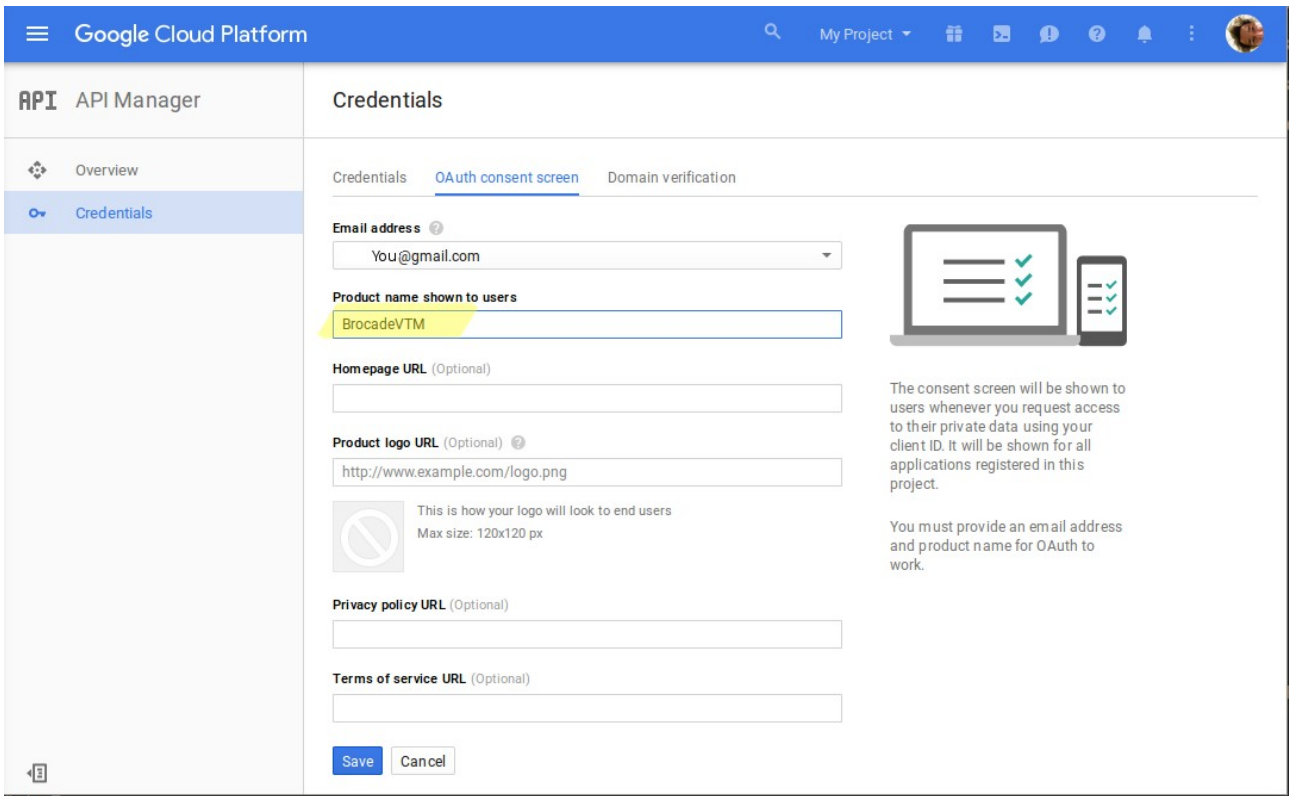
```
gcloud compute instances create $name \
  --image https://www.googleapis.com/compute/v1/projects/brocade-public-1063/global/images/vtm-103-stm-dev-64 \
  --machine-type n1-standard-1 \
  --metadata-from-file startup-script=startup-script.sh \
  --scopes default=https://www.googleapis.com/auth/compute \
  --tags tcp-9090-server,http-server,https-server,ssh-server
```

That's it, the local metadata server will provide access\_tokens to the driver with enough permission to create and destroy compute instances.

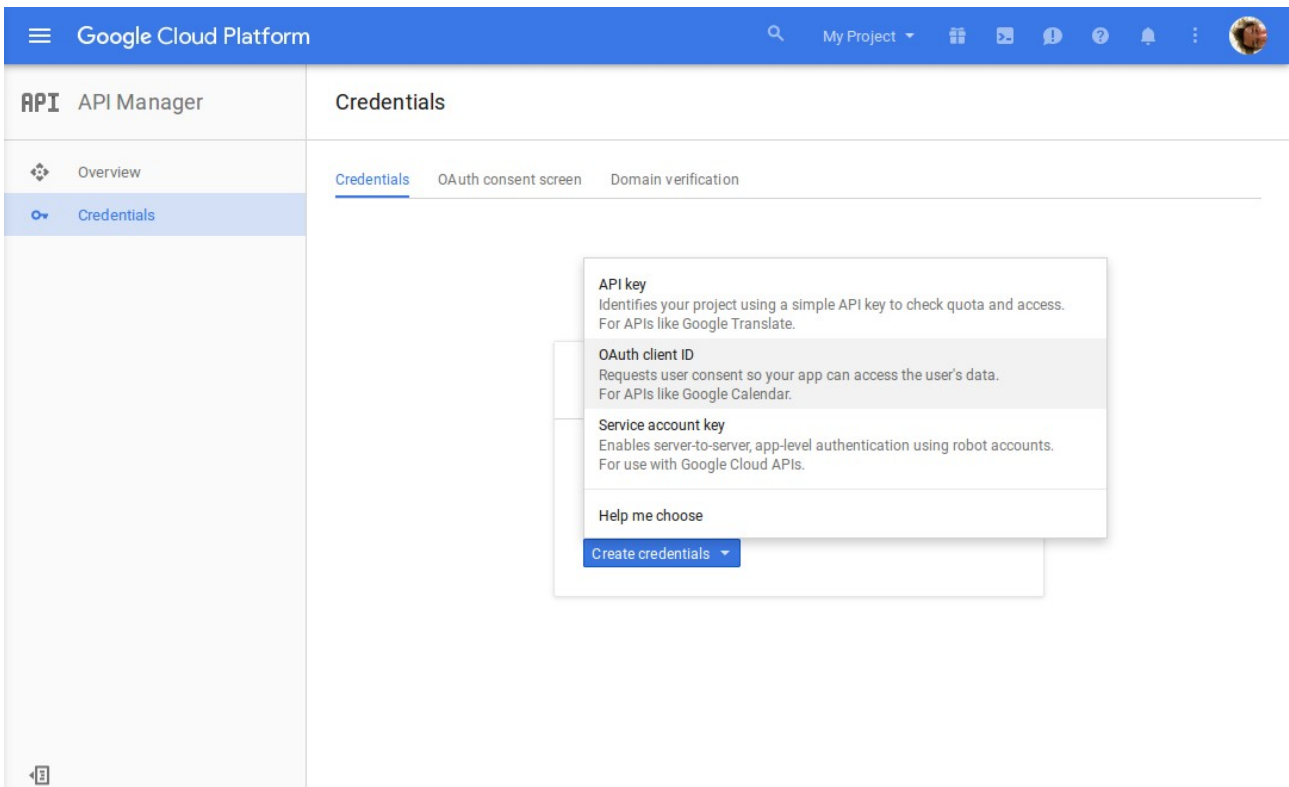
## Authentication via OAUTH2

If your vTM is running outside of the Compute Engine Cloud, or if you don't want Google to automatically issue tokens using the metadata server, then you should enable OAUTH2.

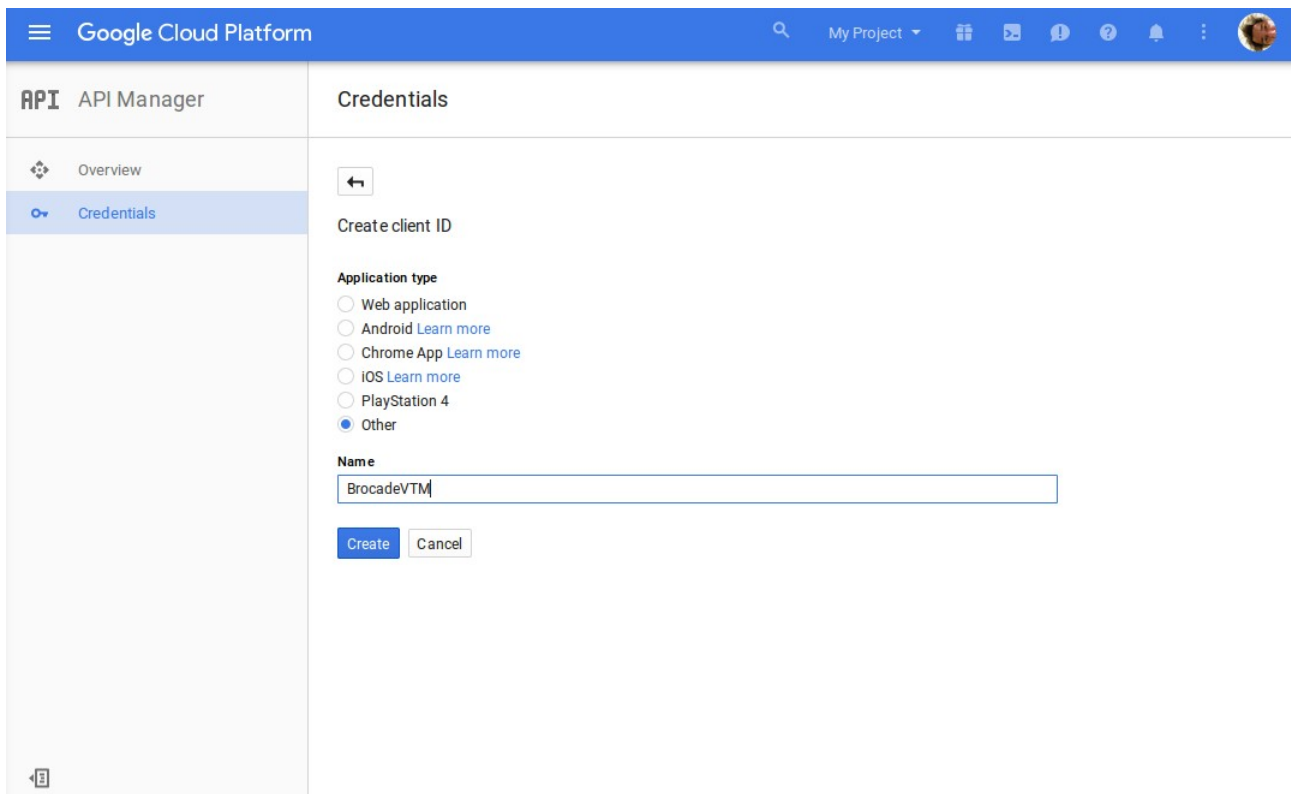
1. Navigate to the API Manager section of the GCE UI
2. Under Credentials → OAUTH Consent Screen → Enter Product Name “BrocadeVTM”



3. Under Credentials → Credentials → Create a new “OAUTH Client ID”



4. Set Application Type to “Other”, and Name to “BrocadeVTM”



5. Click “Create”

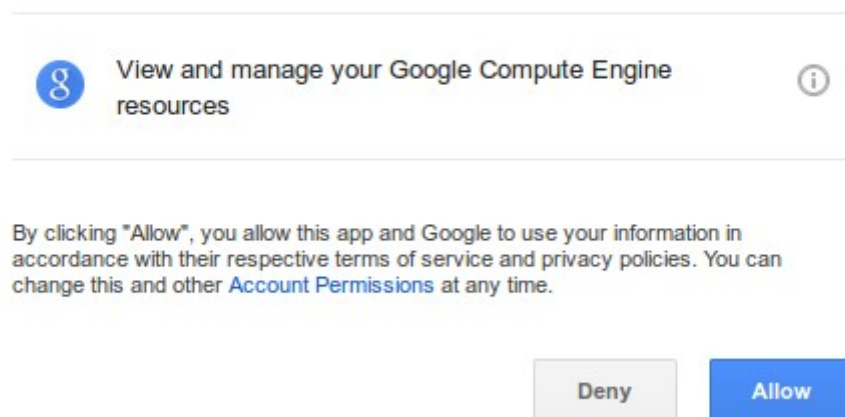
6. You will now be provided with a “client\_id” and a “client\_secret”. Keep them safe!

7. Execute the googledriver script

```
googledriver.py authclient --clientid=<client_id> --secret=<client_secret>
```

8. You will be given a URL to open in a browser and authorize the client to access your compute resources. Please follow the instructions

↳ BrocadeVTM would like to:



9. Once authorized you will be given a key. Provide the key to the googledriver prompt.

10. You now have an Access Token and more importantly a Refresh Token.

11. Create a JSON Configuration file for use by the vTM. It should look like this:

```
{  
  "token_uri": "https://www.googleapis.com/oauth2/v4/token",  
  "refresh_token": "<Your Refresh Token>",  
}
```



```
"client_id": "<Your Client ID>",
"client_secret": "<Your Client Secret>"
}
```

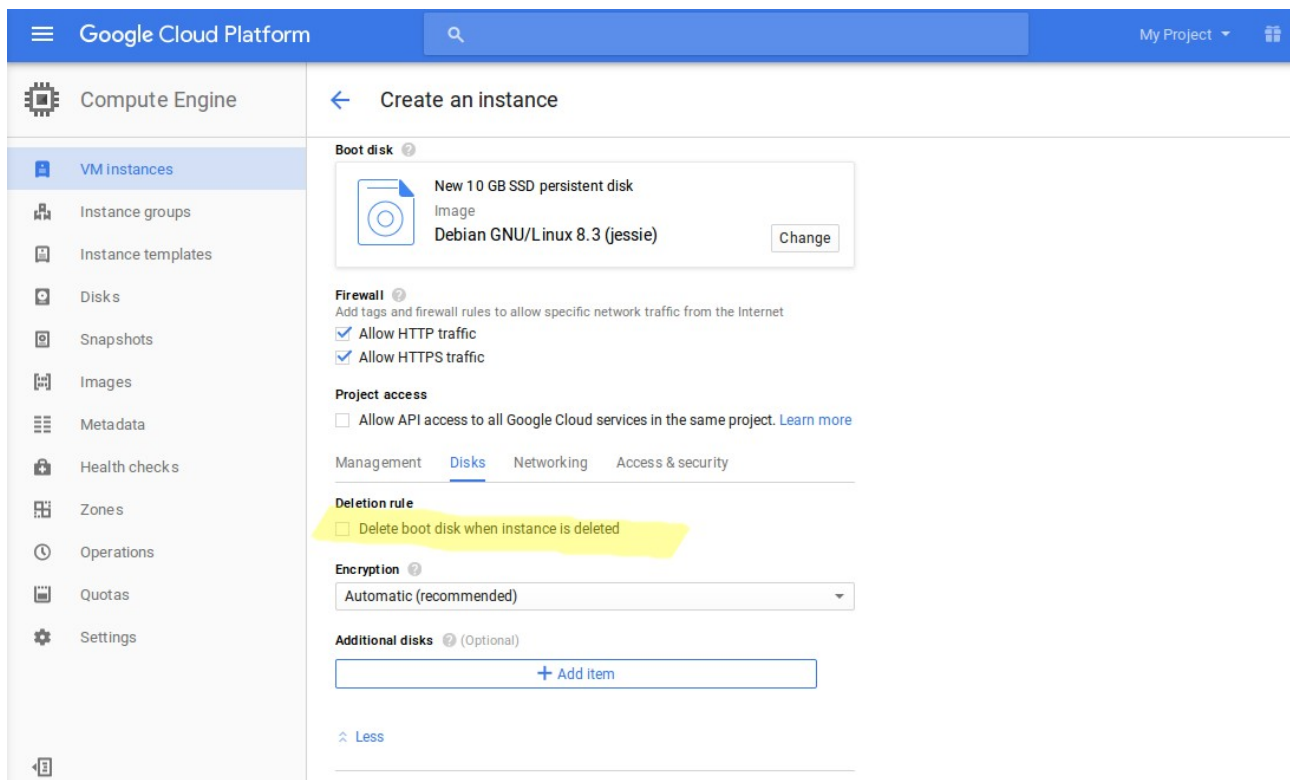
12. upload the json configuration file to the Catalogs → “Extra Files” on the vTM and provide the file name in Credential 1 of the “Cloud Credentials” configuration.

## Google Auto-Scaling FAQ

### 1. How do I create my image?

The image which you use in your pool should be preconfigured so that it starts up and is ready to serve traffic immediately. The specifics will depend largely on the type of application you are deploying, but lets take a simple LAMP stack as an example:

1. Deploy a Linux server in your GCE project using a default image. Ensure that you expand the advanced options. Under “disks” deselect the “delete boot disk when instance is deleted” option.

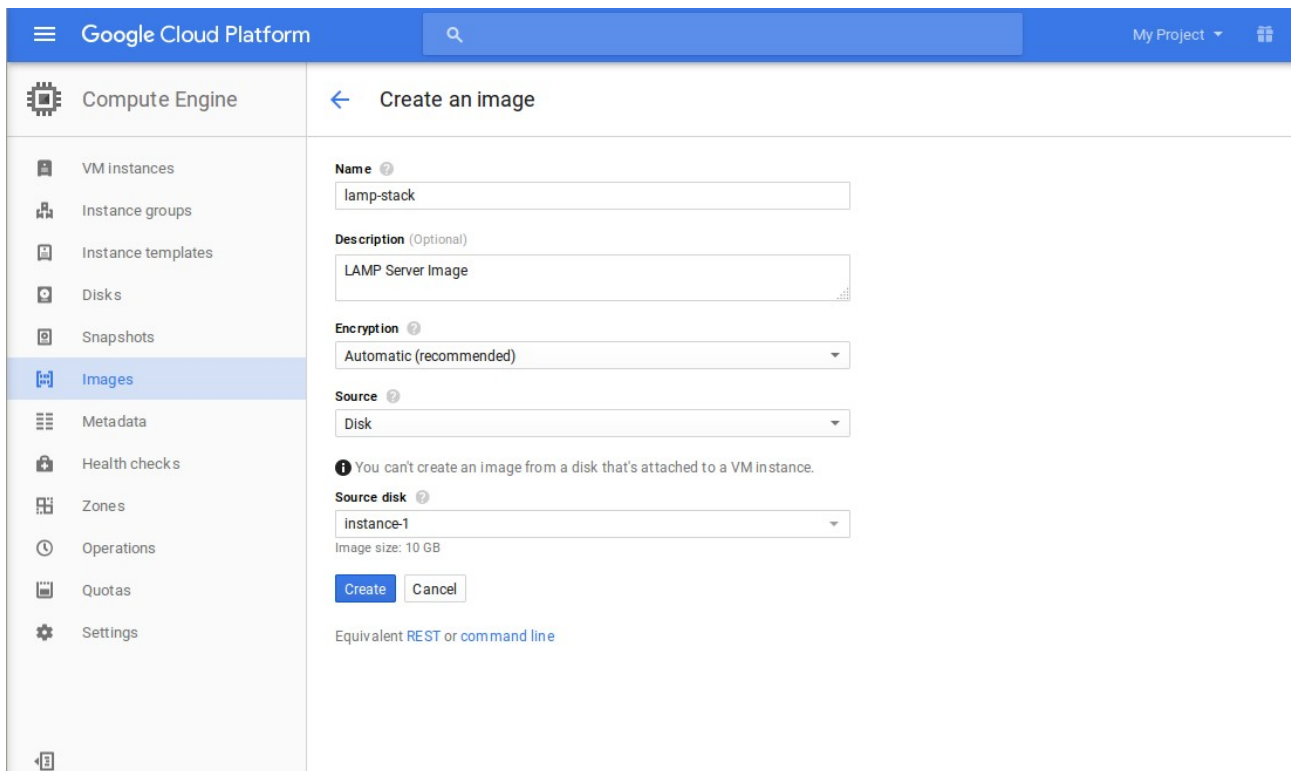


2. Deploy the instance.

3. SSH onto the instance and install your favourite database, webserver, and programming language. Ensure that they are set to start at boot.

4. Delete the instance. If you're asked, do not delete the boot disk!

5. Navigate to Images and select “Create-Image”. You should find the boot disk in the Source Disk drop down. Create the image.



6. You can now use that image in your pools auto-scaling configuration.

## 2. How can I test the autoscaler?

Get the Status report:

```
/opt/zeus/zxtm/conf/extra/googledriver.py status --cred2=<project> --cred3=<zone>
```

Create a node:

```
/opt/zeus/zxtm/conf/extra/googledriver.py createnode --cred2=<project> --cred3=<zone>  
--name=test1 --imageid=<image-name> --sizeid=<machine-type>
```

You could replace the “cred2/3” with a `--cloudcreds=<name>` if you have already created a cloud credentials configuration in the vTM.