Deploying IBM Spectrum LSF auto-scale clusters on Google Cloud

Li Flame

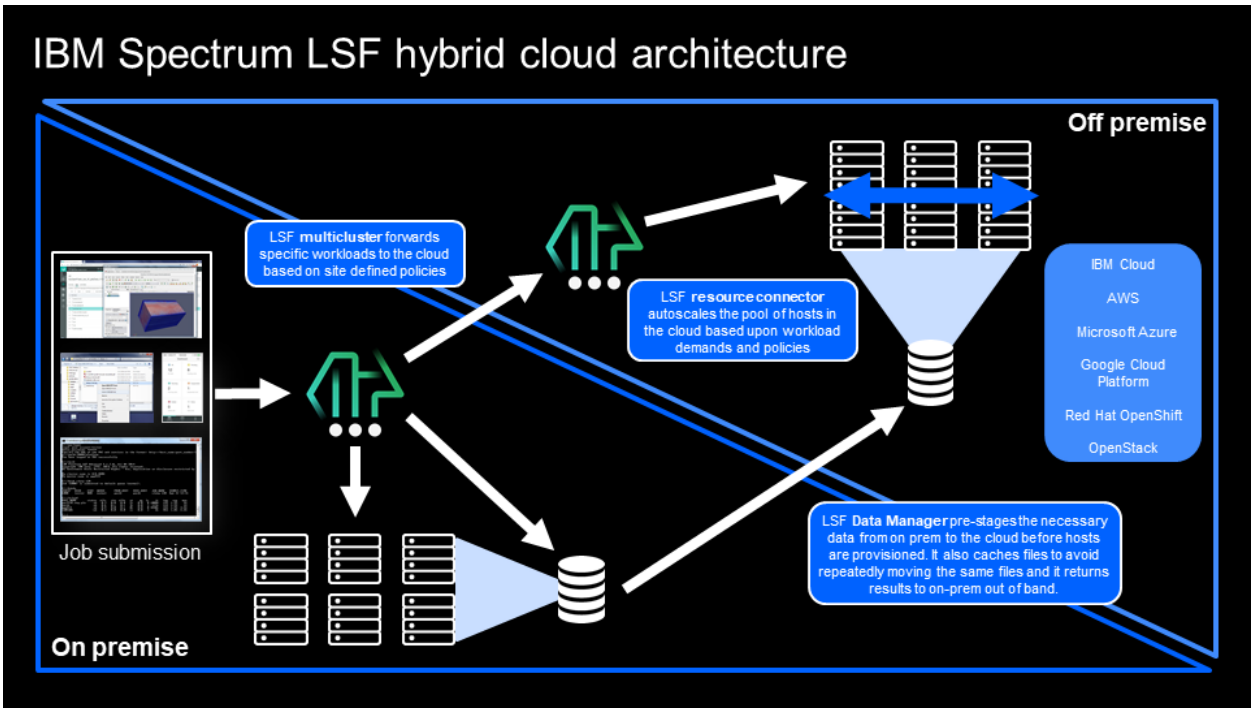
IBM
May 28, 2021

Table of Contents

About IBM Spectrum LSF

IBM Spectrum LSF (“LSF”) is a powerful workload management platform for demanding, distributed HPC environments. LSF provides a comprehensive set of intelligent, policy-driven scheduling features that enables full utilization of your compute infrastructure resources and ensures optimal application performance.

IBM Spectrum LSF features dynamic hybrid cloud capabilities, enabling deployments on-premises, in the cloud, and in the hybrid cloud. Responding to workload, IBM Spectrum LSF dynamically provisions external cloud resources from various cloud providers including Google Cloud Platform. Resources that are no longer needed are returned in time and the administrator can define which containers are permitted to run in the environment. IBM Spectrum LSF hybrid HPC cloud cluster size also dynamically responds to demands. The cluster grows and shrinks accordingly, so you pay only for what you need. The generalized interface supports containerized workloads of Docker, Shifter, Singularity and Enroot technologies. The consistency of deployments on-premises and in the cloud is secured by the help of containerized applications as IBM Spectrum LSF facilitates resource binding, interactive and parallel job support, and automatic container re-running during failure of containerized workloads.



You can now deploy IBM Spectrum LSF on Google Cloud as an auto-scaling cluster based on job requirements. The latest versions of LSF support the various specification and selection

properties of Google Cloud VMs, including scheduling computing jobs on VMs with GPUs, creating VMs in specific zones and regions, and selecting minimum CPU platforms. LSF also supports other Google Cloud features including instance templates, bulk APIs, preemptible VMs, and local SSD disks. Note that the bulk API is used by default by LSF.

LSF resource connector overview

The resource connector for IBM Spectrum LSF (previously referred to as *host factory*) enables LSF clusters to borrow resources from Google Cloud. LSF clusters can borrow hosts from Google Cloud to satisfy pending workload. The borrowed resources join the LSF cluster as hosts. When the resources become idle, the LSF resource connector returns them to Google Cloud.

The resource connector generates requests for extra hosts from Google Cloud and dispatches jobs to dynamic hosts that join the LSF cluster. When the Google Cloud reclaims the hosts, the resource connector requeues the jobs that are running on the LSF hosts, shuts down LSF daemons, and releases the hosts to Google Cloud.
The following is a use case of how the LSF resource connector runs jobs on Google Cloud VM instances.

```
[user@lsfmanage log]$ bhosts
HOST_NAME          STATUS       JL/U    MAX  NJOBS    RUN  SSUSP  USUSP    RSV
lsfmanage.us-centr ok              -      2      0      0      0      0      0
[user@lsfmanage log]$ bsub -R googlehost sleep 100
Job <652> is submitted to default queue <normal>.
[user@lsfmanage log]$ bjobs
JOBID   USER    STAT  QUEUE      FROM_HOST   EXEC_HOST   JOB_NAME   SUBMIT_TIME
652     root    RUN   normal     lsfmanage.u compute-oet sleep 100  Apr  9 06:58
[user@lsfmanage log]$ bhosts
HOST_NAME          STATUS       JL/U    MAX  NJOBS    RUN  SSUSP  USUSP    RSV
compute-oetct0001. closed          -      1      1      1      0      0      0
lsfmanage.us-centr ok              -      2      0      0      0      0      0
[root@lsfmanage log]#
```

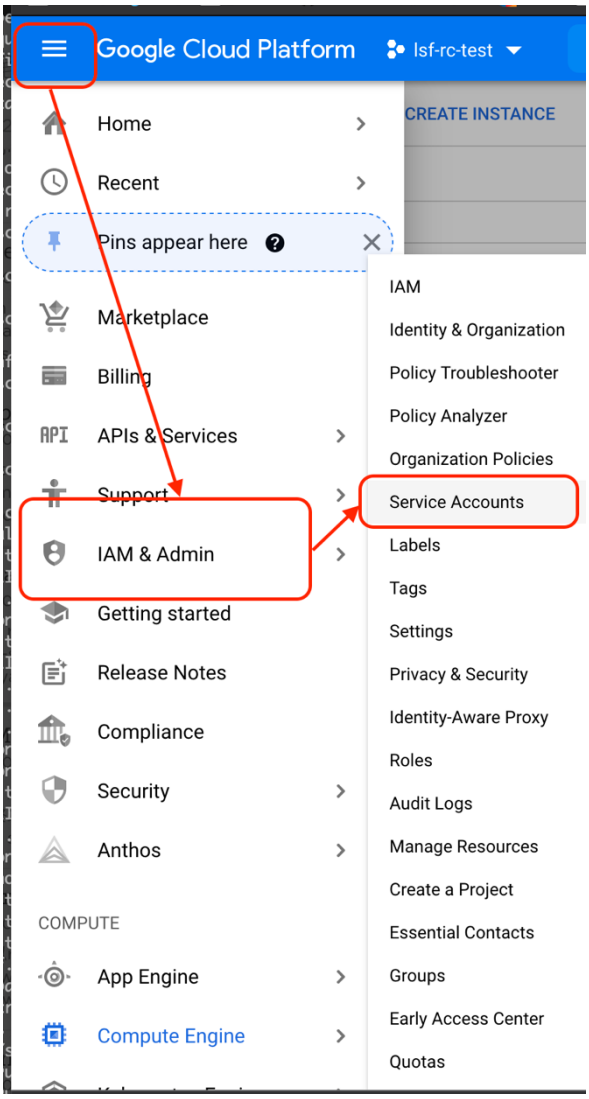
Creating LSF clusters on Google Cloud and configure for LSF resource connector

1. Create a Google Cloud project
    a. Log in to the Google Cloud console and create a new project or select an existing project.

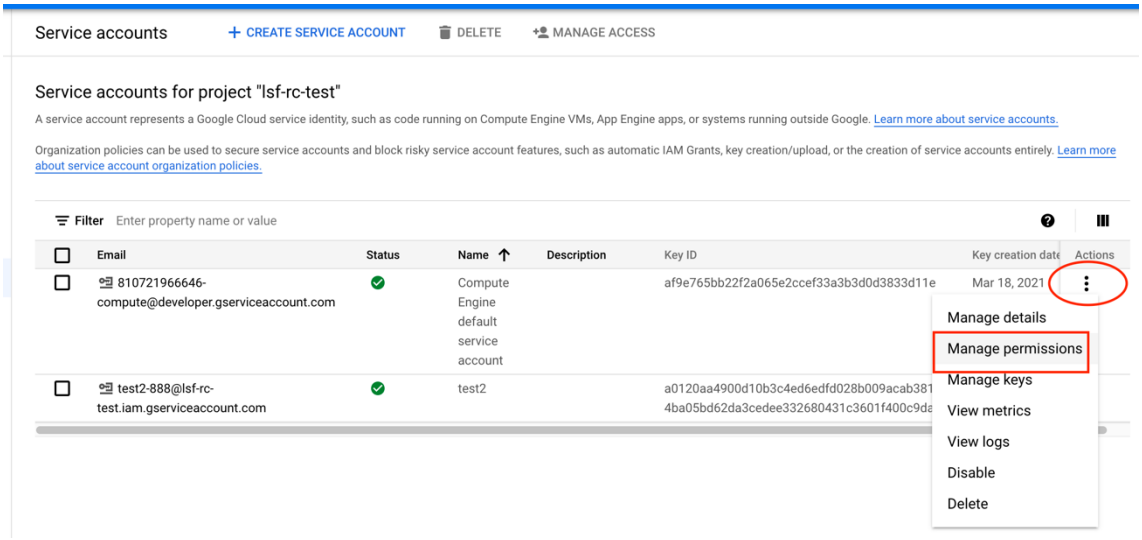
       Your future work is based on this project.

    b. After creating or selecting a project, configure the service account for the project.

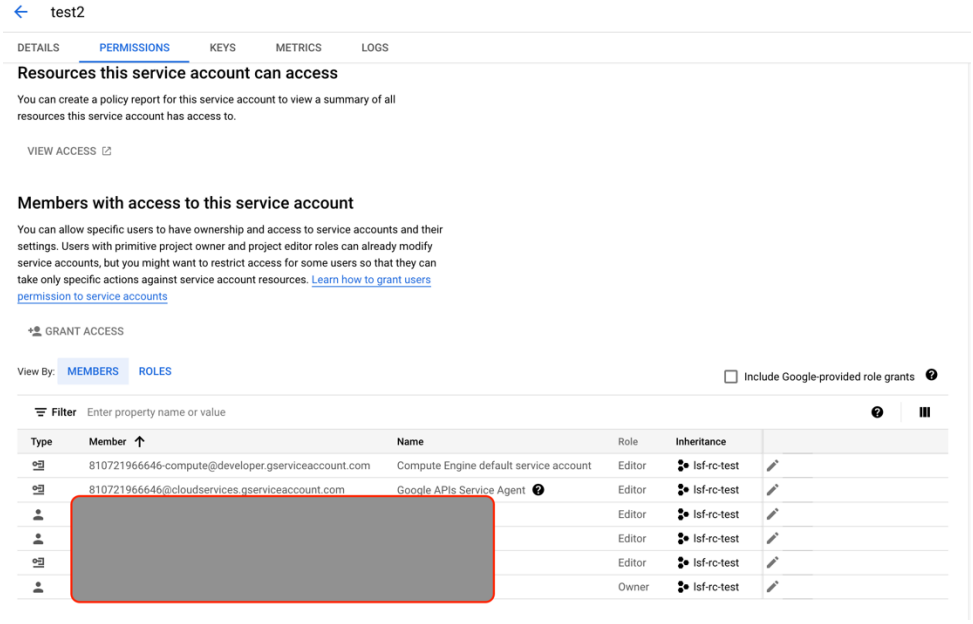
       Select **IAM & Admin > Service Accounts**.



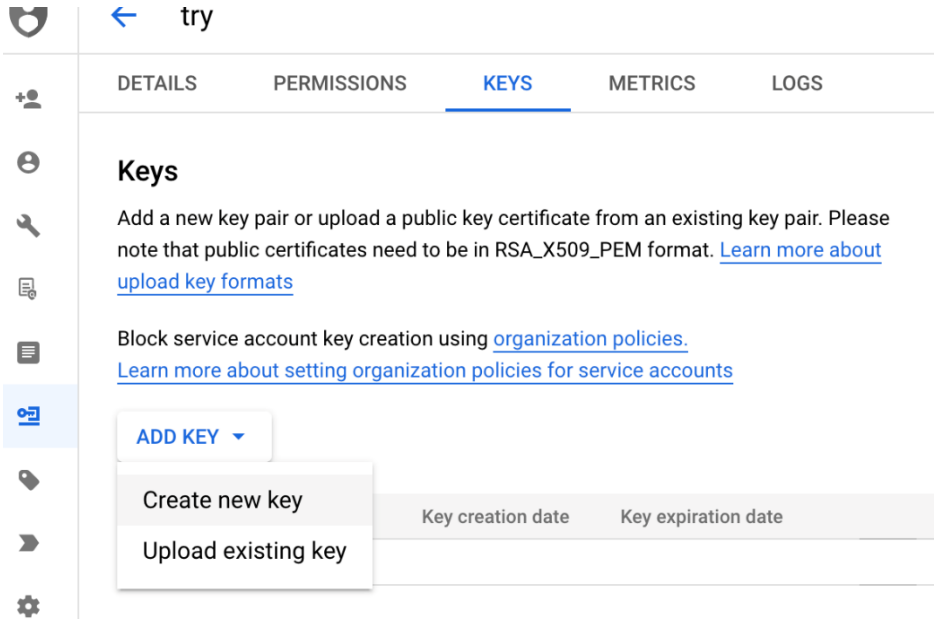
c. Create a new service account in this project, then select **Manage permissions** in the action menu to set the service account's permissions.



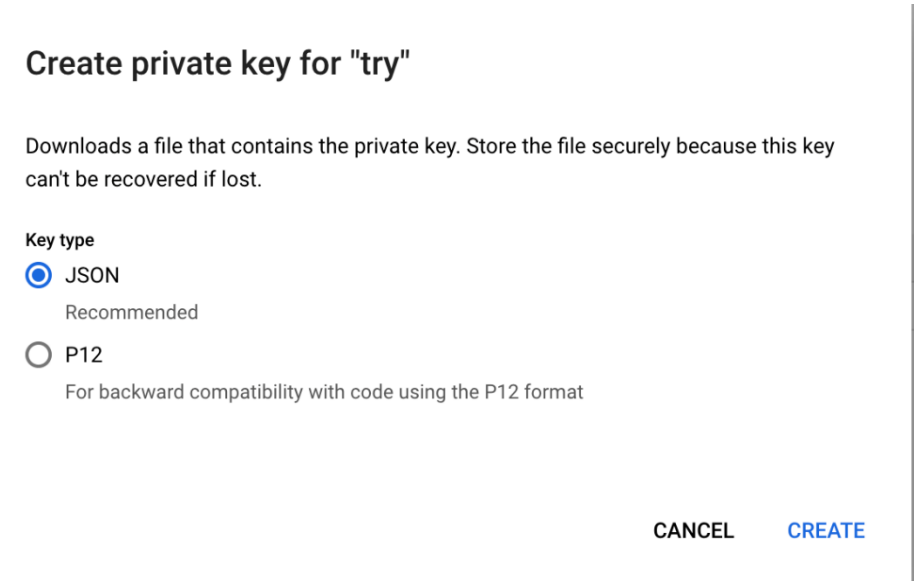
d. Make sure that you set the proper permissions, and that the role of the current service account is **Editor**.



e. Download a credential key for later use.
Click **KEYS**, then click **ADD KEY** and select **Create new key**.

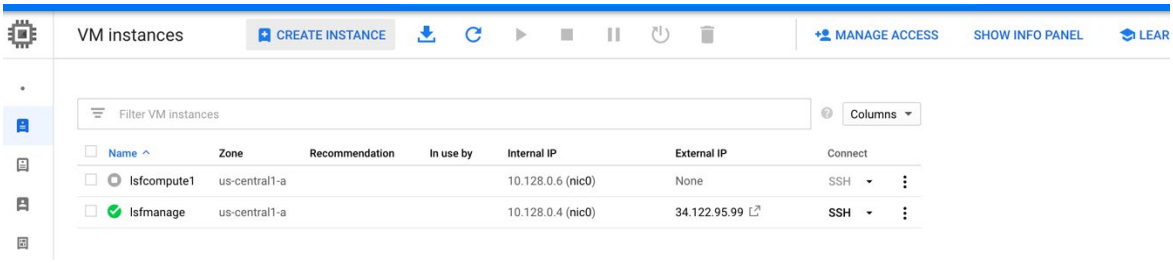


In the next pane, select **JSON**, then click **CREATE** to download a JSON file to your computer. Keep this file, which you will use when creating Google Cloud VM instances for LSF jobs.

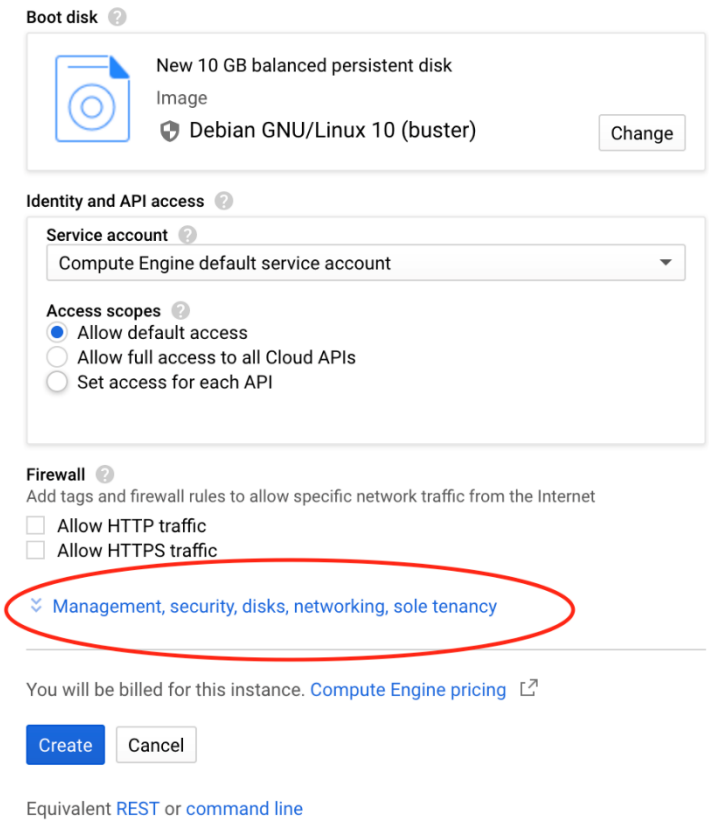


2. Create a Google Cloud VM instance as an LSF management node.
    a. Click the **Compute Engine** menu item, then select **VM instances**.

    b. Click **CREATE INSTANCE**.



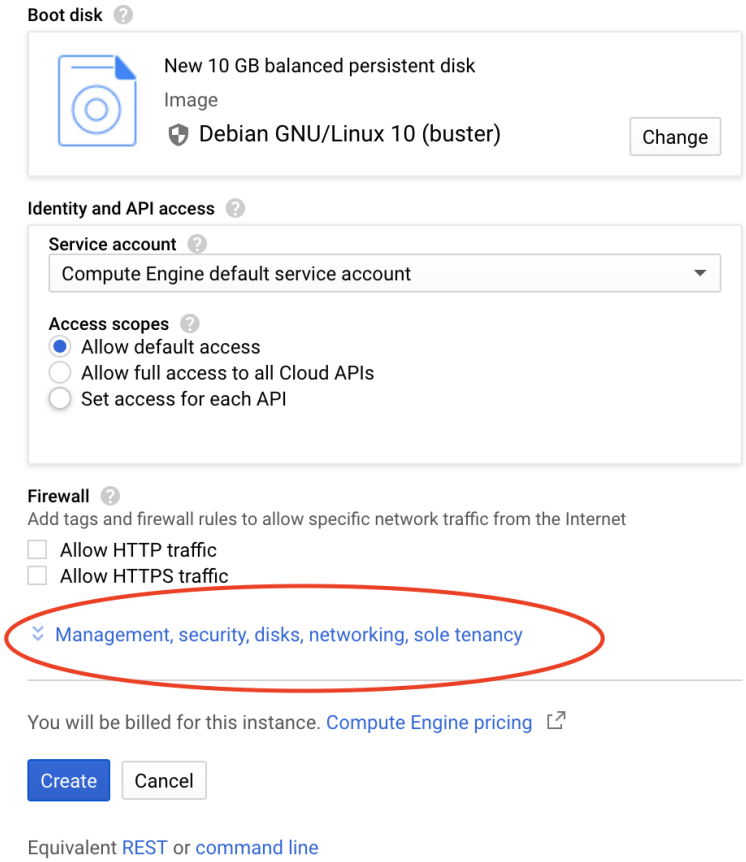
c. Select **New VM instance** and specify the required information.
   - Specify the name and labels according to your requirements.
   - Select a region and zone for location of the VM. Different regions and zones may have different available resources and only certain zones have certain resources such as GPUs. This may impact how you select your machine properties by selecting different zone and regions.
   - For **Boot disk**, you can select the OS type and boot disk size.
   - If required, click **Management, security, disks, networking, sole tenancy** to expand the configuration panel, where you can perform tasks such as setting SSH security or attach extra data disks.



   - In the **Security** section, if you add the ssh-key that you generated from a local host, you can use ssh to log in to the created VM from a

local directory using the external IP address of that VM. This makes it convenient to install and configure LSF on the VM instance.

d. After setting all the required properties for the instance, click **Create** to create the VM, and the VM will automatically start up after it is created.



3. Install and configure basic parameters for the LSF management node.

a. Get the external IP address from the VM list.
b. Transfer the LSF installation package and entitlement file into the new VM instance.
c. Log in to the new VM instance.
d. Install LSF 10.1.
   For more information on installing LSF, see IBM Spectrum LSF documentation.
   If you created the VM with RHEL8, you must install libnsl first.
e. Configure the following parameters in the following files in the new LSF management node:
   - lsf.conf

```
LSF_ROOT_USER=Y
LSB_RC_EXTERNAL_HOST_FLAG=googlehost
LSB_RC_EXTERNAL_HOST_IDLE_TIME=2
LSB_GPU_NEW_SYNTAX=extend
```

```
LSF_GPU_AUTOCONFIG=Y
```

- lsbatch/*clustername*/configdir/lsb.modules
  In the PluginModule section, define the following:
  ```
  Schmod_demand                ()                  ()
  ```

- lsf.shared:
  ```
  Begin Resource
  RESOURCENAME    TYPE     INTERVAL  INCREASING DESCRIPTION
  Googlehost      Boolean  ()        ()         (googlehost
  flag)
  rc_account      String   ()        ()         (RC required)
  instanceID      String   ()        ()         (RC required)
  templateID      String   ()        ()         (RC required)
  clusterName     String   ()        ()         (RC required)
  providerName    String   ()        ()         (RC required)
  End Resource
  ```

- lsbatch/*clustername*/configdir/lsb.queues:
  Configure a queue for running jobs in Google Cloud VMs, then configure the following parameter in the queue:
  ```
  RC_HOSTS=googlehost
  ```

- Enable user_data.sh by running the following commands:
  ```
  cp
  $LSF_TOP/10.1/resource_connector/google/scripts/example_user_d
  ata.sh
  $LSF_TOP/10.1/resource_connector/google/scripts/user_data.sh
  ```

- Edit the user_data.sh file and change the following parameter values:

  ```
  LSF_TOP=lsf_top_directory
  LSF_CONF_FILE=$LSF_TOP/conf/lsf.conf
  . $LSF_TOP/conf/profile.lsf
  Env >> $logfile
  ```

  Make sure user_data.sh will start LSF daemons in the end. If the LSF daemons fail to start, check the contents of the defined logfile.

- Enable Google in the $LSF_TOP/conf/resource_connector/hostProviders.json file by adding the Google provider:

```
{
    "providers":[
        {
            "name": "google",
            "type": "googleProv",
            "path": "resource_connector/google/provider.json"
        }
```

```
  ]
}
```

- Create the $LSF_TOP/conf/resource_connector/google/provider.json file:

```
{
    "host_type": "google_host",
    "interfaces":
    [{
        "name": "getAvailableTemplates",
        "action":
"resource_connector/google/scripts/getAvailableTemplates.sh"
    },
    {
        "name": "getReturnRequests",
        "action":
"resource_connector/google/scripts/getReturnRequests.sh"
    },
    {
        "name": "requestMachines",
        "action":
"resource_connector/google/scripts/requestMachines.sh"
    },
    {
        "name": "requestReturnMachines",
        "action":
"resource_connector/google/scripts/requestReturnMachines.sh"
    },
    {
        "name": "getRequestStatus",
        "action":
"resource_connector/google/scripts/getRequestStatus.sh"
    }]
}
```

- Copy the $LSF_TOP/conf/resource_connector/google/conf directory to the management node VM if not already on the management node VM:

```
cp -rpf $LSF_TOP/10.1/resource_connector/google/conf
$LSF_TOP/conf/resource_connector/google/
```

- Configure the project and credential files.
  Transfer the credential file in json format into the LSF management node VM, configure the project name and the credential file location in the $LSF_TOP/conf/resource_connector/google/conf/googleprov_config.json file:

```
{
  "LogLevel": "INFO",
  "GCLOUD_PROJECT_ID": "lsf-rc-test",
```

```
  "GCLOUD_CREDENTIAL_FILE": "/home/yanli/lsf-rc-test-
af9e765bb22f.json"
}
```

- Start up the LIM, RES, and sbatchd daemons on the LSF management node by running the lsfstartup command:
  lsfstartup

4. Create a Google Cloud VM for the compute node and install LSF on it.
   a. Create another VM to install the LSF compute node and use this VM as the source disk to generate VMs for LSF jobs in the future.
   b. Create a VM in same subnet with LSF management node.
   c. Log in to the instance that you created.
   d. Copy the LSF packages and the entitlement file to the instance.
      For example, copy the following files:
      - lsf10.1_linux2.6-glibc2.3-x86_64.tar.Z
      - lsf10.1_lsfinstall.tar.Z
      - lsf_std_entitlement.dat
   e. If your VM does not have ed installed, use the `yum install` command to install it.
      ```
      yum install ed
      ```
   f. Edit the server.config file in the LSF installer and enable the following parameters:
      - LSF_TOP: Specify the same location as the LSF management node.
      - LSF_ADMINS: Specify the same users as the LSF management node.
      - LSF_TARDIR: Specify the location of the LSF packages.
      - LSF_ENTITLEMENT_FILE: Specify the location of the entitlement file.
      - LSF_SERVER_HOSTS: Specify the name of the LSF management node.
      - LSF_LOCAL_RESOURCES: Specify "[resource googlehost]" to define googlehost as a local resource.
      - LSF_LIM_PORT: Specify the same LIM port as the LSF management node.

      For example:
      ```
      LSF_TOP="/opt/lsf"
      LSF_ADMINS="lsfadmin"
      LSF_TARDIR="/opt/install/"
      LSF_LICENSE="/opt/install/lsf_std_entitlement.dat"
      LSF_SERVER_HOSTS="management.myserver.com"
      LSF_LOCAL_RESOURCES="[resource googlehost]"
      LSF_LIM_PORT="7869"
      LSF_GET_CONF=lim
      ```

5. Run the LSF installation script.
   ```
   ./lsfinstall -s -f server.config
   ```

After installation, make sure the <*LSF_TOP*>/conf/lsf.conf configuration file contains the "googlehost" resource:

```
cat /opt/lsf/conf/lsf.conf
LSF_GET_CONF=lim
LSF_CONFDIR=/opt/lsf/conf
LSF_LIM_PORT=7869
LSF_SERVER_HOSTS="management.myserver.com"
LSF_VERSION=10.1
LSF_LOCAL_RESOURCES="[resource googlehost]"
LSF_TOP=/opt/lsf/
...
```

6. If needed, configure your DNS/NIS server.

   The default DNS server for Google Cloud works with LSF.

7. Manually start the LSF daemons on the instance

   Manually start the LSF daemons on the instance with the lsfstartup command and make sure that the instance can join the LSF cluster as a dynamic host. If the instance does not join the cluster, check the following items, then restart the LSF daemons:

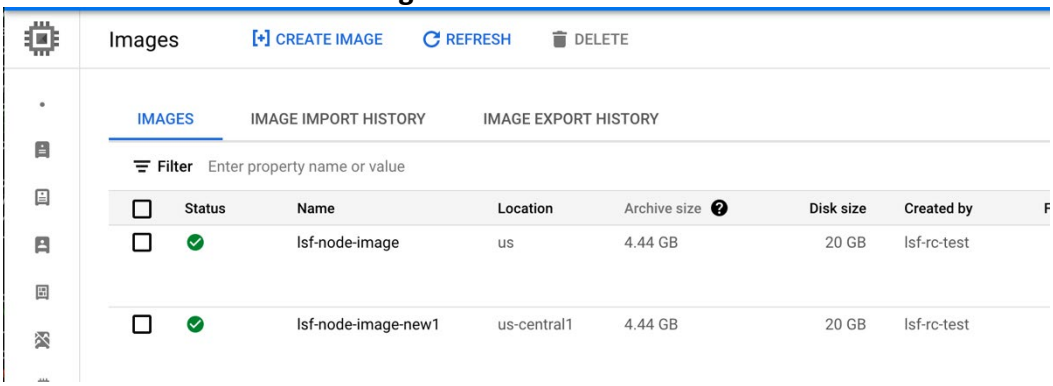
   - The instance firewall or firewall rules of the VPC network.
   - The management host and the instance can ping each other using the private IP address.

   If the instance joins the cluster, you can shut down the compute node VM.

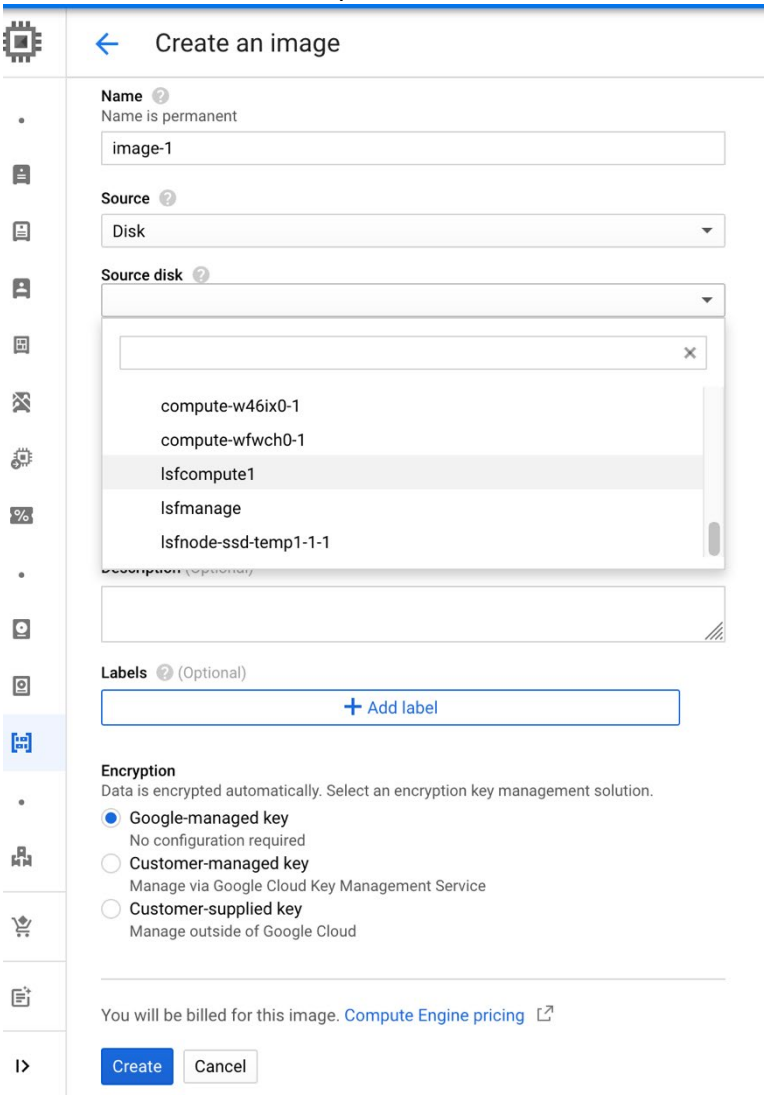
8. Request a VM with the Google Cloud storage image.

   In Google Cloud, you can create a storage image from the installed LSF compute node, then the LSF resource connector calls the Google API to create a VM on which to execute LSF jobs.
   To use the VM in this manner, first create a storage image on the Google Cloud Console.

   a. Select **Compute Engine** from the menu, then select **STORAGE > Images**.
   b. Click **Create an image**.



c. Specify a name for the storage image, then select **Source > Disk** and select the name of the LSF compute node VM from the **Source disk** list.



d. Click **Create** and wait for Google Cloud to create the storage image.
e. Log in to the LSF management node VM
f. Configure the LSF_TOP/conf/resource_connector/google/conf/googleprov_templates.json file to define a template for the created storage image.
For example,

```
{
        "templateId": "gcloud-VM-1",
        "maxNumber": 100,
        "attributes": {
            "type": ["String", "X86_64"],
            "ncores": ["Numeric", "1"],
```

```
                "ncpus": ["Numeric", "1"],
                "nthreads": ["Numeric", "2"],
                "ngpus": ["Numeric", "1"],
                "ngpus_physical": ["Numeric", "1"],
                "gpuextend":["String",
"ngpus=1;nnumas=1;gbrand=Tesla;gmodel=K80;gmem=10240;nvlink=yes"
],
                "define_ncpus_threads": ["Boolean", "1"],
                "mem": ["Numeric", "3840"],
                "zone": ["String", "us_east1-d"],
                "googlehost": ["Boolean", "1"]
            },
            "imageId": "lsf-gcloud-dynamic-vm",
            "region": "us-east1",
            "zone": "us-east1-d",
            "vmType": "n1-standard-1",
            "gpuType": "nvidia-tesla-k80",
            "gpuNumber":"1",
            "instanceTags" : "lsf-vpn-instance=gcloud-VM-1",
            "userData": "zone=us-east1-d"
        }
```

- `templateID` defines the template name that is used within LSF (gcloud-VM-1).
- `attributes` defines the LSF resources that are used for LSF scheduling to select a template that most fits the pending job's requitements if there are multiple templates and storage names.
- Parameter values that are defined outside of `attributes` are used while calling the Google Cloud API to trigger VM. These parameters must follow the Google Cloud API.

After defining this template, you can run the following LSF command to submit a job:

```
bsub -R "googlehost" -gpu "num=1" myjob
```

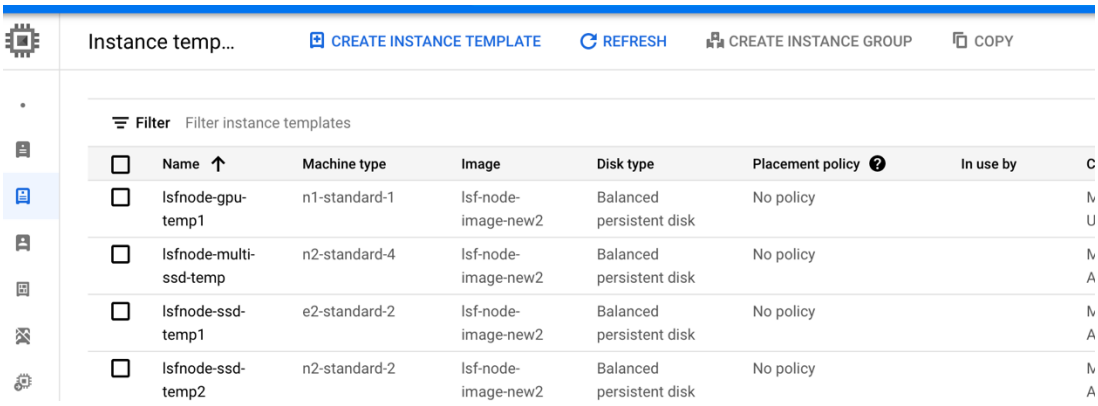
LSF checks to see if there is an available VM with the "googlehost" resource defined that is ready for the job. If there is no VM available, LSF navigates the templates that are defined in the googleprov_templates.json file and checks the resources that are defined in attributes. If LSF finds a template with attributes that fit the pending job's requirements, LSF calls the Google API to create a VM according the requirements for the matching template.

9. Request a VM with the Google Cloud instance.

   Google Cloud has a concept of instance templates, and LSF can use this kind of instance template to directly create a VM. First, create an instance template based on the LSF compute node that you successfully installed.

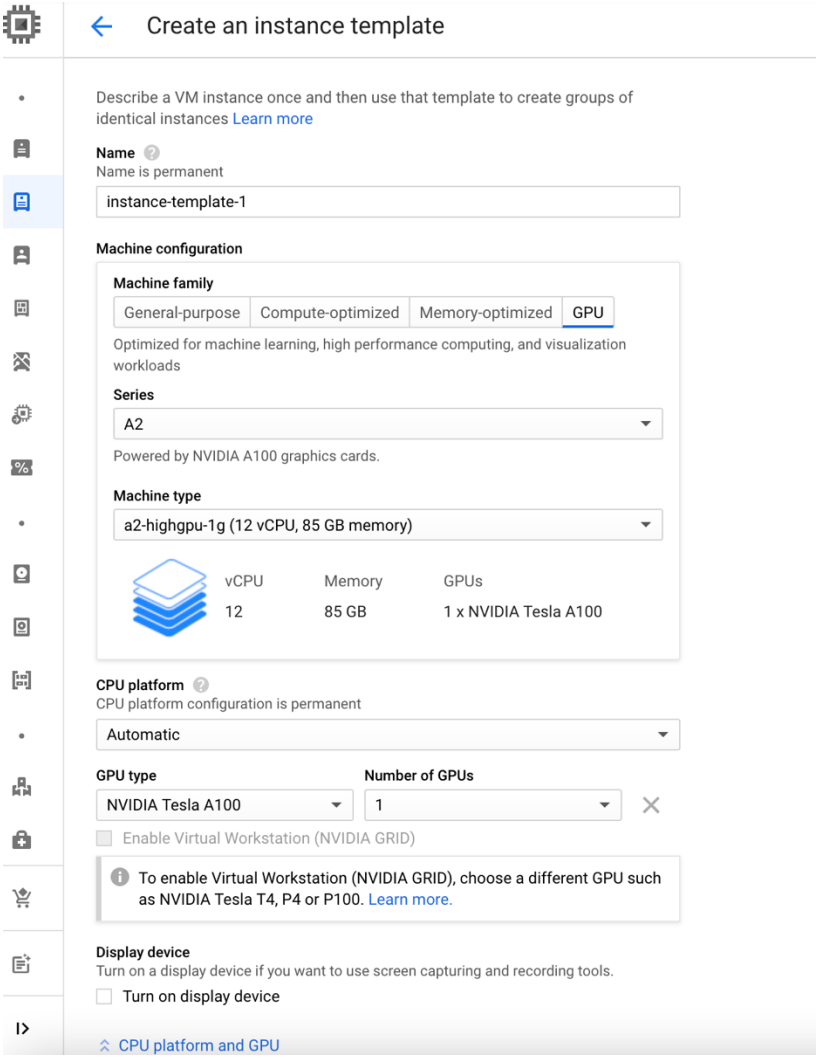
   a. Select **Compute Engine** from the menu, then select **VIRTUAL MACHINES > Instance templates**.

Click “CREATE INSTANCE TEMPLATE”.

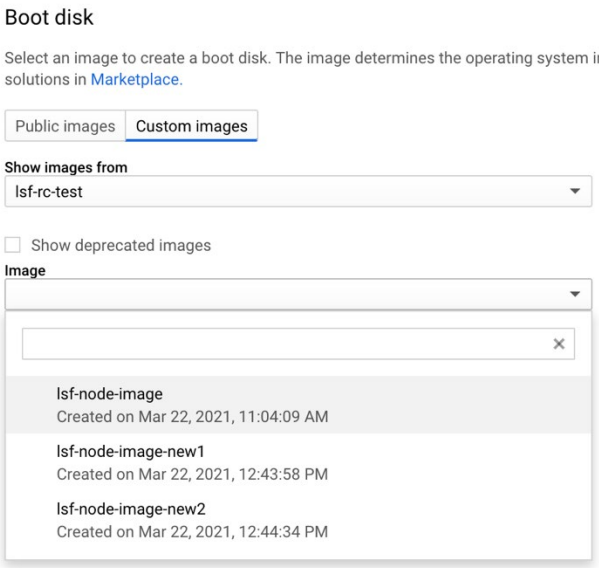


   b. Specify a name for the instance template and select the CPU memory and GPU type/number/size for the template by clicking **GPU**, then selecting the GPU type and number for the template.

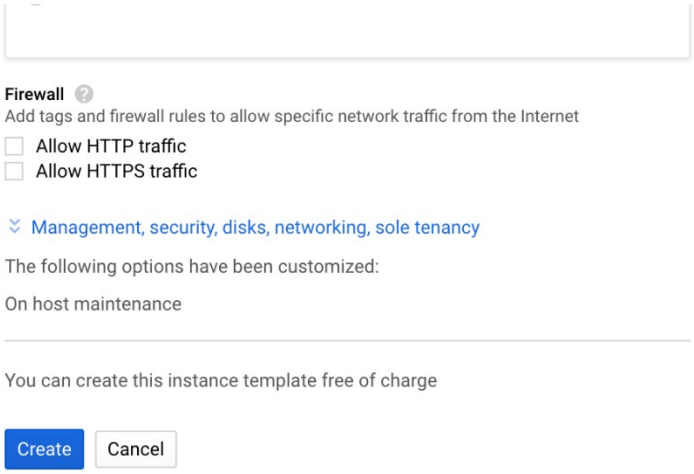
      TIP: For different regions and zones, the CPU/GPU resources and quotas are also different. Refer to Google Cloud documentation to check which regions and zones have the special CPU/GPU resources that you need.



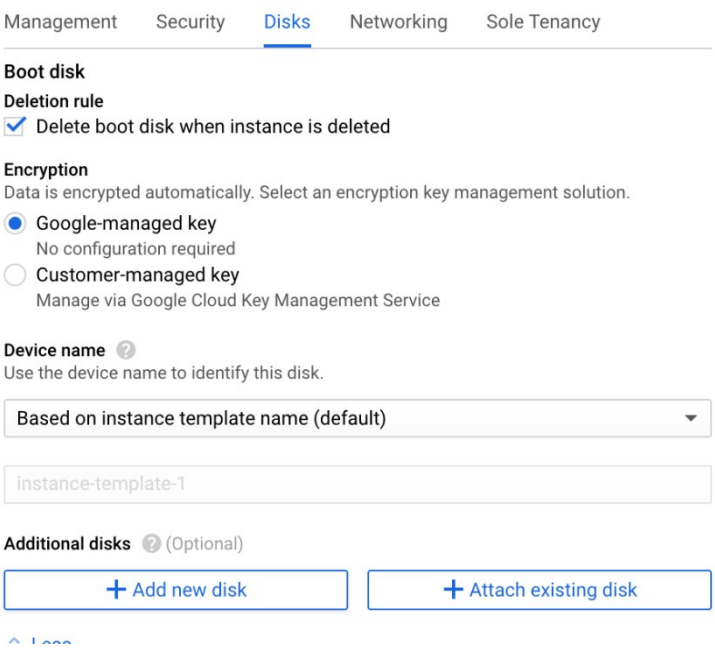
c. Click **Boot disk > Change** to select the image and OS.
d. In the **Boot disk** page, select **Custom images**, then select the storage image that you created from the LSF compute node VM.



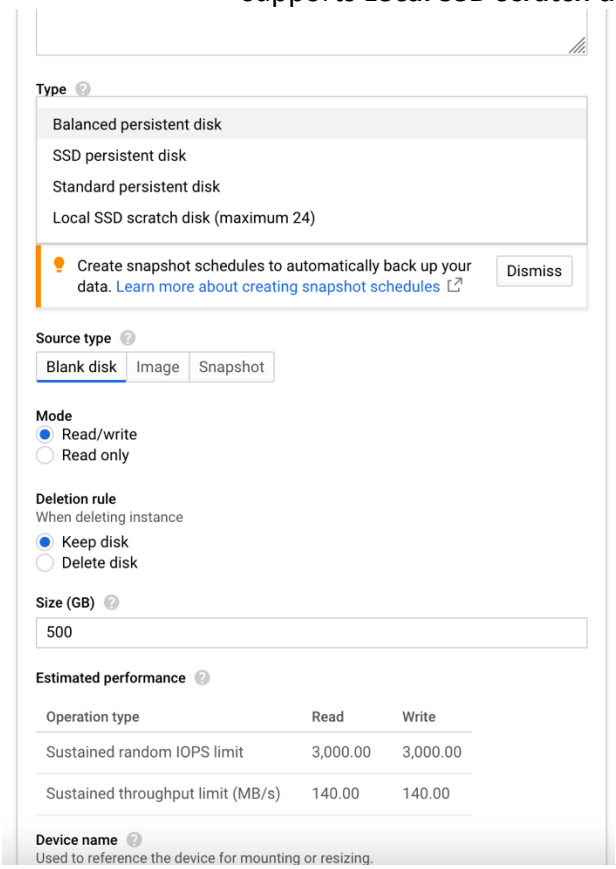
e. Click **Select** to go back to the previous page, then click **Management,security,disks,networking,…** to expand the page and select **Disks**.



f. In the **Disks** page, you can add extra disks beside the boot disk. Click **Add new disk** to add extra disks.

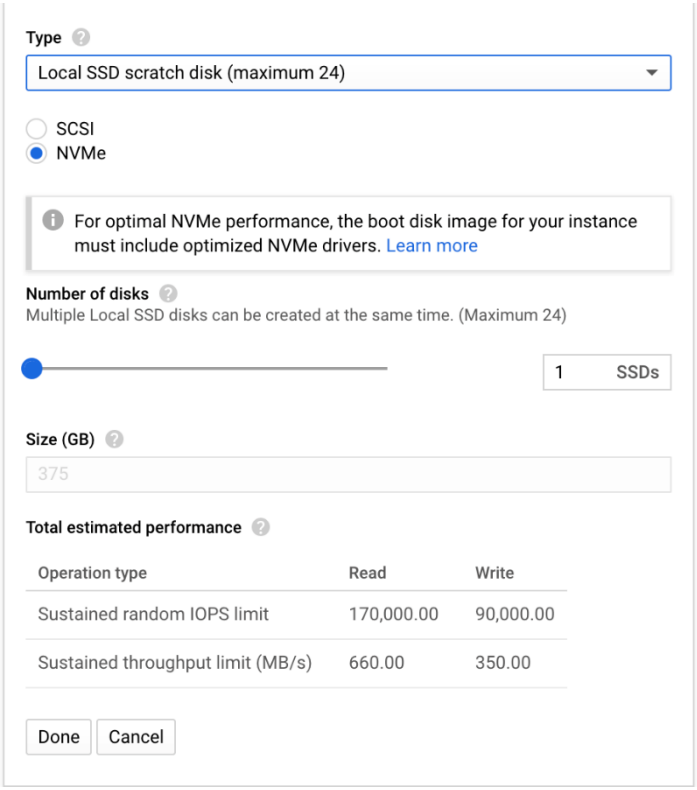


g. When adding disks, you can select which type of disks to add. LSF also supports **Local SSD scratch disk**. Select this from the list of Type.



h. After selecting **Local SSD scratch disk**, there are two 2 sub-types (SCSI and NVMe). LSF supports both of these, and in the LSF_TOP/10.1/resource_connector/google/scripts/user_data.sh script file,

LSF demonstrates a way to automatically mount both kinds of local SSD disks to /tmp.



i. After specifying all the features that you need, click **Create** to create the instance template.

In the LSF_T/OP/conf/resource_connector/google/conf/googleprov_templates.json file, you can use the instance template directly for the Google Cloud API to create the VM.

Use `launchTemplateId` to specify which instance template is used to create the VM.

```
{
      "templateId": "Template-VM-5",
      "launchTemplateId":"lsfnode-gpu-temp1",
      "maxNumber": 2,
      "attributes": {
          "type": ["String", "X86_64"],
          "ncores": ["Numeric", "2"],
          "ncpus": ["Numeric", "2"],
          "nthreads": ["Numeric", "1"],
```

```
                "mem": ["Numeric", "8000"],
                "ngpus":["Numeric","2"],
                "ngpus_physical":["Numeric","2"],

"gpuextend":["String","ngpus=2;nnumas=1;gbrand=Tesla;gmodel=K80;
gmem=10240;nvlink=yes"],
                "googlehost": ["Boolean", "1"]
            },
            "zone": "us-east1-b",
            "instanceTags": "gputemp1"

        }
```

To enable launch instance templates in googleprov_templates.json, specify the `launchTemplateId` and `zone` attributes. All other attributes become optional after you enable launch instance templates. Define all other attributes in the specified instance template, because all settings that are defined in the googleprov_templates.json file override the settings in the instance template.

Within the launch instance template, the attributes are grouped into the following categories:

Disk attributes:
- imageId

Network attributes:
- vpc
- subnetId
- privateNetworkOnlyFlag

GPU attributes:
- gpuType (see https://cloud.google.com/compute/docs/gpus)
- gpuNumber

Other attributes:
- vmType
- minCpuPlatform
- instanceTags (Labels must be defined in googleprov_template.json. Labels that are defined in the template are overridden by default)

If you define an attribute in the googleprov_templates.json, all attributes in the same group override the attribute values in the launch templates, even attributes that you did not define. For example, if you define `gpuType` in googleprov_templates.json to override the value in the launch template, the default value of `gpuNumber` also overrides any gpuNumber value in the launch

template even if you did not specify a value for gpuNumber. To avoid this issue, do not define any attributes in googleprov_templates.prov other than `launchTemplateId` and `zone` if you are using launch instance templates.