GPUs on GCE instances: Alpha Release

Overview
- SLA
- Pricing
- Restrictions

Create an instance with one or more GPUs
- Create an instance and install CUDA 7.0
- Create an instance and install CUDA 8.0
  - Installing CUDA 8.0 on Linux images
  - Installing CUDA 8.0 on Windows images
- Run a CUDA sample
- Handling host maintenance events

Support

Feedback

Overview

As an Alpha release, Google Compute Engine provides machine types with NVIDIA Tesla K80 GPUs in passthrough mode. You can use these machine types to create virtual machine instances with full and direct access to the GPU devices, which can accelerate specific workloads and applications.

SLA

This is an Alpha release of GPUs on Google Compute Engine. This feature is not covered by any service level agreement (SLA) or deprecation policy and may be subject to backward-incompatible changes.

Pricing

Instances with GPU machine types are billed on a per-minute basis with a minimum of 10 minutes per instance. The price for GPU machine types is the base cost for the predefined machine type plus an additional cost for each GPU die that is attached. For example, in the U.S., an n1-standard-4-k80x1 instance in the US costs $0.20 per hour for the base machine type and $0.70 per hour for the single attached K80 GPU die for a total cost of $0.90 per hour. Sustained use discounts apply only to the base machine type and not to the additional price for the GPU die.

- US price per GPU die:
  - On demand: **$0.70 per hour**
  - One year commitment: **$0.367 per hour**
- Asia and Europe price per GPU die:
  - On demand: **$0.77 per hour**
  - One year commitment: **$0.404 per hour**

If the available GPU machine types do not meet your needs for this early access release, send a request to gpu-whitelist-request@google.com and specify the memory and vCPU configuration that you require.

Restrictions

GPU instances have the following restrictions:

- Projects that have access to this Alpha release can create GPU instances in the following zones:
  - us-central1-a
  - us-east1-d
  - europe-west1-b
  - asia-east1-a
- GPU instances cannot migrate on host maintenance. When you create GPU instances, you must disable instance migration, which means that your instances must terminate due to host maintenance at some point.
- The NVIDIA driver on Linux is a kernel module, so you must reinstall the NVIDIA driver any time you change your operating system kernel. For example, if you update your operating system kernel after you install the NVIDIA driver, the driver will not initialize properly on restart.
- Each project can use only a limited number of GPUs. If your project meets its GPU quota, you will not be able to create additional GPU instances.

Create an instance with one or more GPUs

You can create a GPU instance with a GPU machine type, a public image, and any driver that you like, but this guide covers basic configurations with public images. If you have not created an instance on this project before, initialize GCE by opening the console here: https://console.cloud.google.com/compute/

Some public images work best with the driver in the CUDA 7.0 installer. Other images work better with the driver included in the CUDA 8.0 installer. For the best experience, use the image and CUDA installer that work best together.

- CUDA 7.0

    - CentOS 7 and RHEL 7
- CUDA 8.0
    - CentOS 7 and RHEL 7
    - openSUSE 13.2
    - SLES 12
    - Ubuntu 16.04
    - Windows Server 2012 R2

Create an instance and install CUDA 7.0

You can create the instance using any of the normal processes for creating and starting an instance, but this example uses the gcloud tool. The the CUDA 7.0 .run installer works on the CentOS 7 and RHEL 7 public images.

1. Download and install the gcloud SDK on your local workstation or open the Cloud Shell to get an environment where you can run *gcloud* commands.
2. Use the `instances create` command to create a new GPU instance. Specify the zone, select a GPU machine type, and set the instance to terminate on migration. Optionally, you can include the `--local-ssd interface=SCSI` flag to attach one or more local SSDs to the instance. If you do add local SSDs, understand what to expect from local SSD data persistence.

   ```
   gcloud compute instances create gpu-instance-1 \
   --zone us-central1-a --machine-type n1-standard-4-k80x1 \
   --maintenance-policy TERMINATE --project [WHITELISTED_PROJECT_ID] \
   --image-family centos-7 --image-project centos-cloud
   ```

   Alternatively, you can also use the RHEL 7 image family:

   ```
       --image-family rhel-7 --image-project rhel-cloud
   ```

3. After the instance starts, connect to the instance.

   ```
   gcloud compute ssh gpu-instance-1 \
   --zone us-central1-a --project [WHITELISTED_PROJECT_ID]
   ```

4. On your instance, update all of your packages and the kernel. Then, restart the instance to apply those updates. This ensures that you have the latest kernel version before you run the CUDA installer.

   ```
   sudo yum update -y && sudo reboot
   ```

5. Reconnect to the instance.

   ```
   gcloud compute ssh gpu-instance-1 \
   --zone us-central1-a --project [WHITELISTED_PROJECT_ID]
   ```

6. After you reconnect to your instance, install the required packages for the Nvidia driver.
   ```
   sudo yum install gcc perl gcc-c++ kernel-devel -y
   ```

7. Download the CUDA 7.0 installer.
   ```
   curl -O
   http://developer.download.nvidia.com/compute/cuda/7_0/Prod/local_installers/cuda_7.0.28_linux.run
   ```

8. Run the CUDA installer and follow the install instructions. This step also installs the Nvidia driver. For this example, the default options work fine. Install the CUDA samples so you can test the configuration later.
   ```
   sudo sh cuda_7.0.28_linux.run
   ```

9. Restart the instance to enable the driver.
   ```
   sudo reboot
   ```

10. Reconnect to the instance.
    ```
    gcloud compute ssh gpu-instance-1 \
    --zone us-central1-a --project [WHITELISTED_PROJECT_ID]
    ```

11. After you reconnect to the instance, run the Nvidia System Management Interface to verify that the driver works properly. The following command lists information about each GPU that is connected to your instance:
    ```
    nvidia-smi
    ```

You can now run CUDA applications as well as other applications that require a GPU.

Create an instance and install CUDA 8.0

You can create the instance using any of the normal processes for creating and starting an instance, but this example uses the gcloud tool. The the CUDA 8.0 installer works on the following public images:

- CentOS 7, RHEL 7, or Ubuntu 16.04 with the *.run* installer.
- openSUSE 13.2 or SLES 12 with the *.rpm* installer.
- Windows Server 2012 R2 with the *.exe* installer.

Installing CUDA 8.0 on Linux images

1. Download and install the gcloud SDK on your local workstation or open the Cloud Shell to get an environment where you can run *gcloud* commands.
2. Use the *instances create* command to create a new GPU instance. Specify the zone, select a GPU machine type, and set the instance to terminate on migration. Optionally, you can include the `--local-ssd interface=SCSI` flag to attach one or more local SSDs to

the instance. If you do add local SSDs, understand what to expect from Local SSD data persistence.

```
gcloud compute instances create gpu-instance-1 \
--zone us-central1-a --machine-type n1-standard-4-k80x1 \
--maintenance-policy TERMINATE --project [WHITELISTED_PROJECT_ID] \
--image-family centos-7 --image-project centos-cloud
```

Alternatively, you can also use one of the following image families or images:

```
--image-family rhel-7 --image-project rhel-cloud
--image-family ubuntu-1604-lts --image-project ubuntu-os-cloud
--image opensuse-13-2-v20160222 --image-project opensuse-cloud
--image sles-12-sp1-v20160301 --image-project suse-cloud
```

3. After the instance starts, connect to the instance.

```
gcloud compute ssh gpu-instance-1 \
--zone us-central1-a --project [WHITELISTED_PROJECT_ID]
```

4. On your instance, update your packages and the kernel. Then, restart the instance to apply those updates. This ensures that you have the latest kernel version before you install the Nvidia driver.
   - CentOS 7 and RHEL 7:

     ```
     sudo yum update -y \
     && sudo yum install wget gcc perl gcc-c++ kernel-devel -y \
     && sudo reboot
     ```

   - Ubuntu 16.04:

     ```
     sudo apt-get update && sudo apt-get install gcc perl g++ \
     linux-source linux-headers-$(uname -r) \
     linux-image-extra-$(uname -r) linux-image-extra-virtual -y \
     && sudo reboot
     ```

   - SLES 12 or openSUSE 13.2:

     ```
     sudo zypper update && sudo reboot
     ```

5. Reconnect to the instance.

```
gcloud compute ssh gpu-instance-1 \
--zone us-central1-a --project [WHITELISTED_PROJECT_ID]
```

6. After you reconnect to your instance, download the CUDA 8.0 installer for your distribution.
   - CentOS 7, RHEL 7, and Ubuntu 16.04 *.run* installer:

     ```
     wget
     ```

```
https://developer.nvidia.com/compute/cuda/8.0/prod/local_installers/cuda_8.0.44_linux-run
```

- SLES 12 *.rpm* installer:
  ```
  curl -O
  http://developer.download.nvidia.com/compute/cuda/repos/sles12/x86_64/cuda-repo-sles12-8.0.44-1.x86_64.rpm
  ```

- openSUSE 13.2 *.rpm* installer:
  ```
  curl -O
  http://developer.download.nvidia.com/compute/cuda/repos/opensuse132/x86_64/cuda-repo-opensuse132-8.0.44-1.x86_64.rpm
  ```

7. Follow the install instructions for your specific distribution and installer type.
   - CentOS 7, RHEL 7, and Ubuntu 16.04 *.run* installer:
     ```
     sudo sh cuda_8.0.44_linux-run
     ```
   - SLES 12 *.rpm* installer:
     ```
     sudo rpm -i cuda-repo-sles12-8.0.44-1.x86_64.rpm \
     && sudo zypper refresh && sudo zypper install cuda -n
     ```
   - openSUSE 13.2 *.rpm* installer:
     ```
     sudo rpm -i cuda-repo-opensuse132-8.0.44-1.x86_64.rpm \
     && sudo zypper refresh && sudo zypper install cuda -n
     ```
8. After you complete the install process, restart the instance to enable the driver.
   ```
   sudo reboot
   ```
9. Reconnect to the instance.
   ```
   gcloud compute ssh gpu-instance-1 \
   --zone us-central1-a --project [WHITELISTED_PROJECT_ID]
   ```
10. After you reconnect to the instance, run the Nvidia System Management Interface to verify that the driver works properly. The following command lists information about each GPU that is connected to your instance:
    ```
    sudo nvidia-smi
    ```

You can now run CUDA applications as well as other applications that require a GPU.

Installing CUDA 8.0 on Windows images

1. Download and install the gcloud SDK on your local workstation or open the Cloud Shell to get an environment where you can run *gcloud* commands.

2. Use the *instances create* command to create a new GPU instance. Specify the zone, select a GPU machine type, and set the instance to terminate on migration. Optionally, you can include the `--local-ssd interface=SCSI` flag to attach one or more local SSDs to the instance. If you do add local SSDs, understand what to expect from Local SSD data persistence.

   ```
   gcloud compute instances create gpu-instance-1 \
   --zone us-central1-a --machine-type n1-standard-4-k80x1 \
   --maintenance-policy TERMINATE --project [WHITELISTED_PROJECT_ID] \
   --image-family windows-2012-r2 --image-project windows-cloud
   ```

3. After the instance starts, generate a new password on the Windows instance.
4. Connect to the Windows instance using RDP.
5. On the Windows instance, download the CUDA 8 local installer for Windows Server 2012 R2. You might need to configure Internet Explorer to accept your download request.
6. Run the CUDA 8 installer and follow the on-screen instructions to complete the installation.
7. Restart the Windows instance to complete the driver installation.

Run a CUDA sample

If you installed CUDA on your instance either as part of the driver install or separately, you can run CUDA on the instance. Run a CUDA sample to verify that the GPU works correctly.

For CUDA 8.0 on Windows Server 2012 R2, use the instructions in the CUDA Toolkit Documentation. For CUDA 7.0 or 8.0 on Linux, use the following instructions:

1. Connect to your instance:

   ```
   gcloud compute ssh gpu-instance-1 \
   --zone us-central1-a --project [WHITELISTED_PROJECT_ID]
   ```

2. Install the make tool:
   - CentOS/RHEL versions 6 or 7:

     ```
     sudo yum install make -y
     ```

   - Ubuntu 16.04:

     ```
     sudo apt-get install make -y
     ```

   - openSUSE 13.2 or SUSE 12:

     ```
     sudo zypper install make
     ```

3. Change directories to one of the CUDA samples. For CUDA 8.0 installed from the .rpm package, copy the samples to your home directory first.
   - CUDA 7.0 .run installer:
     ```
     cd ~/NVIDIA_CUDA-7.0_Samples/6_Advanced/mergeSort
     ```
   - CUDA 8.0 .run installer:
     ```
     cd ~/NVIDIA_CUDA-8.0_Samples/6_Advanced/mergeSort
     ```
   - CUDA 8.0 .rpm package installer:
     ```
     cp -R /usr/local/cuda/samples ~/samples \
     && cd ~/samples/6_Advanced/mergeSort
     ```
4. Run make:
   ```
   make
   ```
5. Run the mergeSort sample:
   ```
   sudo ./mergeSort
   ```

Handling host maintenance events

Your GPU instances must terminate when the host machine undergoes maintenance. You cannot set your instances to automatically migrate for host maintenance events. Instead, monitor the */computeMetadata/v1/instance/maintenance-event* metadata value to receive advanced notice that your instance will terminate. If the request to the metadata server returns *NONE,* the instance is not scheduled to terminate. For example:

```
curl
http://metadata.google.internal/computeMetadata/v1/instance/maintenance-event -H "Metadata-Flavor: Google"
```

```
NONE
```

If the metadata server returns a timestamp, the timestamp indicates when your instance will be forcefully terminated. Compute Engine gives GPU instances a **one hour** termination notice, while normal instances receive only a 60 second notice.
For more details and examples for how to monitor maintenance events, see the Storing and Retrieving Instance Metadata documentation.

Support

If you need technical support during the Alpha release, contact the GPU developer support group at gpu-dev-support@google.com.

Feedback

If you have feedback about GPU instances on GCP or feedback for this document, send it to gpu-dev-support@google.com.