Containers: Installing NVIDIA docker on Bright 8.0

Preliminary support for NVIDIA docker has been implemented for 8.0, deeper integration with Bright Cluster Manager is still a work in progress.

It is recommended to have a separate software image for the nodes containing GPU's, for example **/cm/images/gpu-image** so we can install the package in there once and provision all the GPU nodes at once.

Step 1: Make sure docker is installed first

Docker may have been previously installed in a different category, it is possible to execute the setup again for the gpu category

Example run of **cm-docker-setup -e -i**:

```
root@mbrt-ubuntu-trunk:~# cm-docker-setup -e -i
Run docker on head node (default: "no"):
no,
yes
> no
Node categories (default: "default", set to "none" if you don't want to deploy against any
categories):
default,
gpu,
kube,
kube-gpu,
none
> gpu
Additional docker registries (default: none):
>
Storage backend (default: devicemapper):
default,
devicemapper
>
Use block device (default: no):
no,
yes
>
Loopback file size in GB for containers data (default: 100)
>
Loopback file size in GB for metadata (default: 2):
>
```

Setting up docker engine ...
Docker Engine has been setup successfully.
The compute nodes where docker will run has finished imageupdate.

Step 2: Install the cm-nvidia-docker and cuda drivers inside the GPU image

Example for Ubuntu:

chroot /cm/images/gpu-image

apt install cm-nvidia-docker cuda-driver
systemctl enable nvidia-docker

Example for CentOS/RHEL:

yum --installroot=/cm/images/gpu-image install cm-nvidia-docker cuda-driver

chroot /cm/images/gpu-image
systemctl enable nvidia-docker

Full output example from an Ubuntu system:

```
root@mbrt-ubuntu-trunk:~# chroot /cm/images/gpu-image
root@mbrt-ubuntu-trunk:/# apt install cm-nvidia-docker cuda-driver
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  cm-nvidia-docker
0 upgraded, 1 newly installed, 0 to remove and 177 not upgraded.
Need to get 2,261 kB of archives.
After this operation, 14.0 MB of additional disk space will be used.
Get:1 mirror://updates.brightcomputing.com/deb/cm/trunk/ubuntu/mirrors
.txt xenial/main amd64 cm-nvidia-docker amd64 1.0.1-100015-cm-7f66df12
37 [2,261 kB]
Fetched 2,261 kB in 0s (2,544 kB/s)
E: Can not write log (Is /dev/pts mounted?) - posix_openpt (2: No such
 file or directory)
Selecting previously unselected package cm-nvidia-docker.
```

```
(Reading database ... 146472 files and directories currently installed
.)
Preparing to unpack .cm-nvidia-docker_1.0.1-100015-cm-7f66df1237_amd64
.deb ...
Unpacking cm-nvidia-docker (1.0.1-100015-cm-7f66df1237) ...
Setting up cm-nvidia-docker (1.0.1-100015-cm-7f66df1237) ...
Configuring user
Setting up permissions
setcap cap_fowner+pe /cm/local/apps/nvidia-docker/1.0.1/bin/nvidia-doc
ker-plugin
Running ldconfig
Processing triggers for libc-bin (2.23-0ubuntu7) ...
root@mbrt-ubuntu-trunk:/# systemctl enable nvidia-docker
Created symlink /etc/systemd/system/multi-user.target.wants/nvidia-doc
ker.service, pointing to /lib/systemd/system/nvidia-docker.service.
```

Step 3: Reboot GPU nodes

Recommended way of provisioning is issuing a reboot, as an imageupdate is not sufficient.

```
root@mbrt-ubuntu-trunk:~# pdsh -g category=gpu reboot
```

Final step: Test if all components are working correctly

The nvidia-docker service should be running, module files should be loaded and running the "Hello world" which is running **nvidia-smi** inside a container to output the info about the available GPU(s).

```
root@node003:~# systemctl status nvidia-docker
? nvidia-docker.service - NVIDIA Docker plugin
   Loaded: loaded (/lib/systemd/system/nvidia-docker.service; enabled;
 vendor preset: enabled)
   Active: active (running) since Wed 2017-08-23 17:16:56 CEST; 37s ag
o
```

```
     Docs: https://github.com/NVIDIA/nvidia-docker/wiki
  Process: 1741 ExecStartPost=/bin/sh -c /bin/echo unix:///var/lib/nvi
dia-docker/nvidia-docker.sock > /etc/docker/plugins/nvidia-docker.spec
 (code=exited, status=0/SUCCESS)
  Process: 1733 ExecStartPost=/bin/sh -c /bin/mkdir -p  /etc/docker/pl
ugins (code=exited, status=0/SUCCESS)
 Main PID: 1730 (nvidia-docker-p)
    Tasks: 5
   Memory: 8.2M
      CPU: 179ms
   CGroup: /system.slice/nvidia-docker.service
           ??1730 /cm/local/apps/nvidia-docker/current/bin/nvidia-dock
er-plugin -s /var/lib/nvidia-docker -d /usr/local/nvidia-docker

Aug 23 17:16:56 node003 systemd[1]: Starting NVIDIA Docker plugin...
Aug 23 17:16:56 node003 nvidia-docker-plugin[1730]: /cm/local/apps/nvi
dia-docker/current/bin/nvidia-docker-plugin | 2017/08/23 17:16:56 Load
ing NVIDIA unified memory
Aug 23 17:16:56 node003 systemd[1]: Started NVIDIA Docker plugin.
Aug 23 17:16:56 node003 nvidia-docker-plugin[1730]: /cm/local/apps/nvi
dia-docker/current/bin/nvidia-docker-plugin | 2017/08/23 17:16:56 Load
ing NVIDIA management library
Aug 23 17:16:56 node003 nvidia-docker-plugin[1730]: /cm/local/apps/nvi
dia-docker/current/bin/nvidia-docker-plugin | 2017/08/23 17:16:56 Disc
overing GPU devices
Aug 23 17:16:57 node003 nvidia-docker-plugin[1730]: /cm/local/apps/nvi
dia-docker/current/bin/nvidia-docker-plugin | 2017/08/23 17:16:57 Prov
isioning volumes at /usr/local/nvidia-docker
Aug 23 17:16:57 node003 nvidia-docker-plugin[1730]: /cm/local/apps/nvi
dia-docker/current/bin/nvidia-docker-plugin | 2017/08/23 17:16:57 Serv
ing plugin API at /var/lib/nvidia-docker
Aug 23 17:16:57 node003 nvidia-docker-plugin[1730]: /cm/local/apps/nvi
dia-docker/current/bin/nvidia-docker-plugin | 2017/08/23 17:16:57 Serv
ing remote API at localhost:3476
root@node003:~# module load docker/engine/1.12.6
root@node003:~# docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED
          STATUS              PORTS               NAMES
root@node003:~# module load nvidia-docker/1.0.1
root@node003:~# nvidia-docker ps
CONTAINER ID        IMAGE               COMMAND             CREATED
          STATUS              PORTS               NAMES
root@node003:~# nvidia-docker run --rm nvidia/cuda nvidia-smi
Wed Aug 23 15:19:00 2017
+-----------------------------------------------------------------------
--------+
| NVIDIA-SMI 375.66                 Driver Version: 375.66
        |
|-------------------------------+----------------------+--------------
```

```
--------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Unco
rr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Com
pute M. |
|===============================+======================+==============
========|
|   0  Tesla K40c          Off  | 0000:00:08.0     Off |
      0 |
| 23%   32C    P8    22W / 235W |      2MiB / 11439MiB |      0%
Default |
+-------------------------------+----------------------+--------------
--------+


+-----------------------------------------------------------------------
--------+
| Processes:                                                       GPU
 Memory |
|  GPU       PID  Type  Process name                               Usa
ge      |
|=======================================================================
========|
|  No running processes found
        |
+-----------------------------------------------------------------------
--------+
root@node003:~#
```

NVIDIA docker will need to pull an image the first time, the image it will use to run above example.

The service will also copy and mount needed NVIDIA libraries inside the container, this may also take a short while in case it's the first run.

Unique solution ID: #1382
Author: Ray Burgemeestre
Last update: 2017-08-29 11:40