超算平台下的应用容器化实践简介

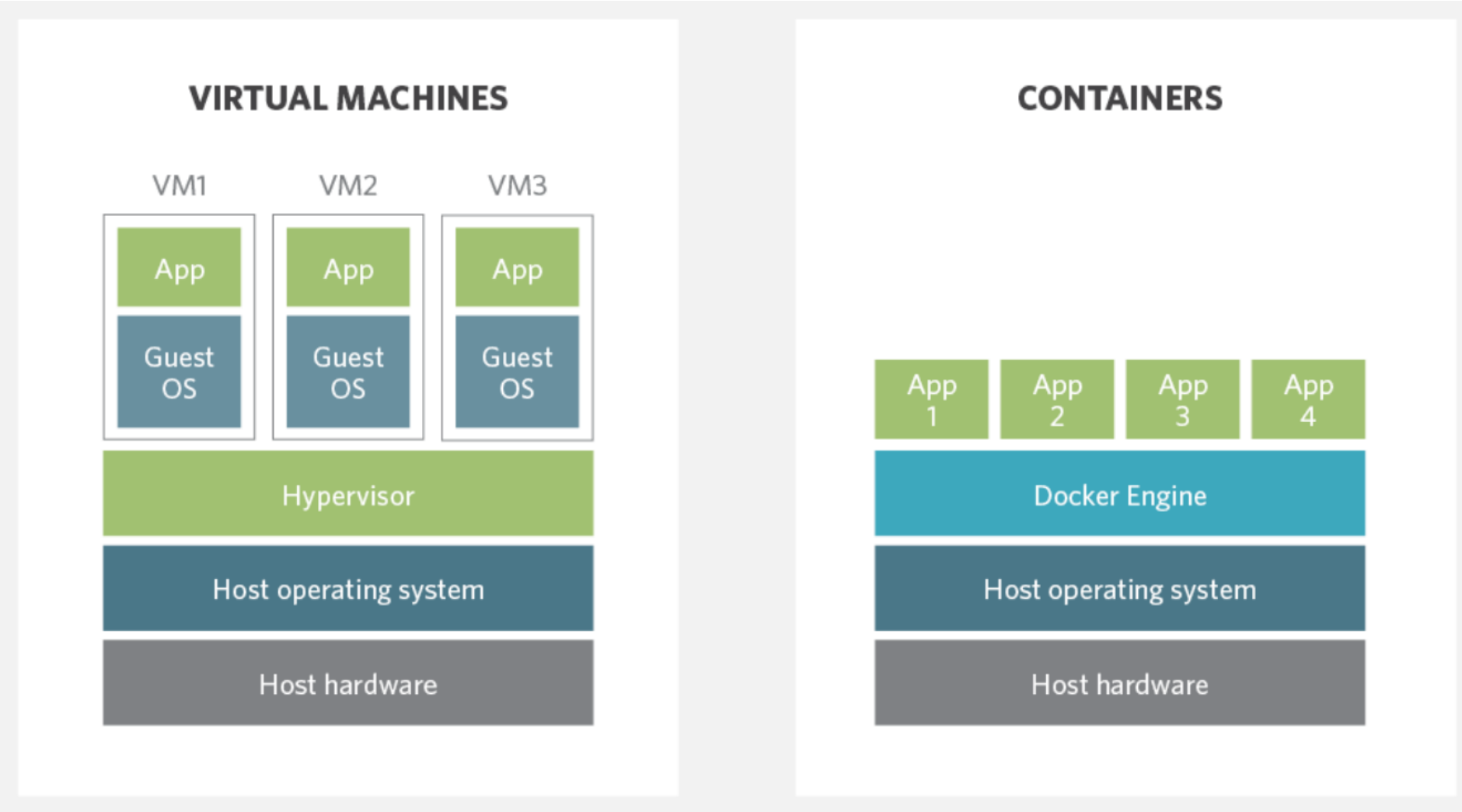
张文帅 (wszhang@ustc.edu.cn)

中国科学技术大学超级计算中心

2019 年 03 月 15 日

目录

容器与虚拟机



以 `Docker` 容器为例

容器的优势

1. 解决应用对特定系统版本的依赖问题，使得多种系统版本下的程序可以在同一个平台无缝连续的运行。
2. 封装复杂依赖关系的应用，使得开发人员直接生成具有生产能力的镜像，减轻随工程量不断增加带来的持续上升运维压力（不过也增加了镜像系统的运维需求）。
3. 更有效的隔离资源，严格限定作业运行时可使用的资源空间，降低作业间的互相影响。
4. 通过作业迁移手段，增强作业运行崩溃时的断点续算能力，减少计算浪费，并增加节点的利用率。

容器分类

- 系统级容器

OpenVZ

具有强大的自我定制的 kernel，旧版本可在 CentOS6 系统基础上安装，最新版本更改为发行整套系统，不再支持基于 CentOS7 安装。尤其，Checkpoint/Restore 底层做了较大的变更，放弃自有的成熟的在 Kernel 内部工作的机制，改用基于用户层的 CRIU 技术，虽然 CRIU 社区与技术前景更好，但当前并不太成熟。

LXC/LXD

当前由 Ubuntu 主推，为 Docker 0.9 版本之前使用的容器底层技术，支持热迁移，使用标准 Linux 内核，不需要定制的补丁，能获得较好的上游社区支持。

- 应用级容器

Docker

IT 工业应用最多的容器方案，特色是标准化了使用 Dockerfile 创建容器的过程方法，使得应用容器更加方便易行，但是对于 HPC 应用，附加功能有些过于庞大，对 MPI 应用并不友好，同时很重要的安全隔离做的不好，在 HPC 这种裸跑用户应用的生产环境，很容易造成安全问题。

Shifter

由 NERSC 开发，较早且比较成熟的 HPC 下的容器方案，具有一个 Image Gateway，用户可直接从 dockerhub 选择容器或自己定制化，执行容器化的应用。安全隔离方面有所改进，但是仍有需要 root 权限的地方。其被 Torque、Slurm 等多种资源调度平台支持，只是系统的配置与用户使用仍有一些不方便。

Singularity

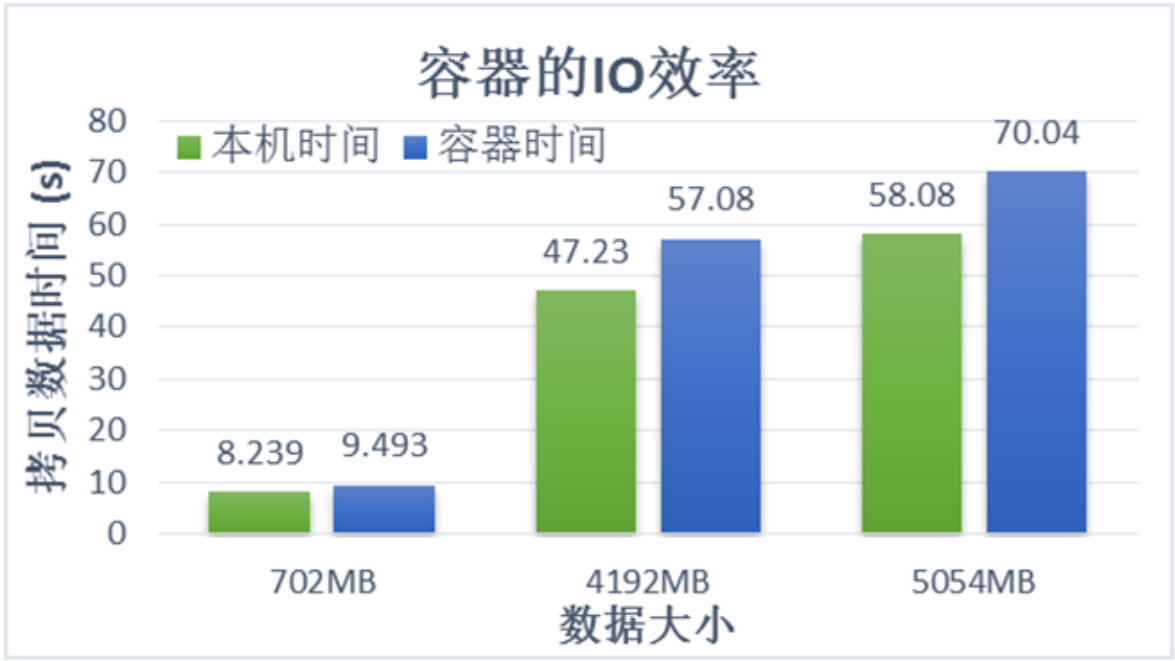
定位与 Shifter 相似，但是在安全隔离上面做的最好，其仅限定在用户空间内启动并运行容器，所以应用的运行完全被隔离不会影响到系统的安全。其提供了从 Docker 镜像直接 pull 并转换为自有镜像格式的方法，也维护有一个自有的镜像库。用户的使用方式也易学易用。

更多容器比较

Attribute	self-compile	virtual machine hypervisor	Shifter chroot	Singularity priv. ns.	Docker userns*	rkt userns*	NsJail userns	Charliecloud userns
Workflow (G1)								
User-defined kernel and settings	·	✓	·	·	·	·	·	·
Use package managers, e.g. `apt-get`, `yum`	·	✓	✓	✓	✓	✓	✓	✓
No conflicts with host software	·	✓	✓	✓	✓	✓	✓	✓
Industry-standard image build	·	·	✓	·	✓	✓	·	✓
Reproducible image build	·	·	✓	✓	✓	✓	·	✓
Resources (G2)								
No privileged or trusted daemons	✓	✓	·	✓	·	✓	✓	✓
No additional network infrastructure	✓	·	✓	✓	·	✓	✓	✓
Network filesystems see no UDSS metadata	·	✓	✓	✓	✓	✓	✓	✓
Direct device access	✓	·	✓	✓	✓	✓	✓	✓
Direct filesystem access	✓	·	✓	✓	✓	✓	✓	✓
Direct high-speed network access	✓	·	✓	✓	✓	✓	✓	✓
Simplicity (G3)								
Implementation language	n/a	varies	C, Python, C++, sh	C, sh, Python	Go	Go	C	C, sh
Lines of code	n/a	varies	19,000	11,000	133,000	52,000	4,000	800
No resource manager-specific code	✓	✓	·	✓	✓	✓	✓	✓
No communication framework-specific code	✓	✓	✓	·	✓	·	✓	✓
No root operations on center resources	✓	·	·	·	·	✓	✓	✓
No guest supervisor process	✓	·	·	·	·	✓	·	✓
No cache, configuration, or other state	✓	·	·	·	·	·	✓	✓

[Charliecloud: Unprivileged containers for user-defined software stacks in HPC, Priedhorsky, Reid Randles, Timothy C., 2016]

OpenVZ 性能测试结果



Gaussian 容器作业的迁移测试

■ **Checkpoint 与 Restore 时间**

这里测试了不同进程数与算例数的 Gaussian 容器的 Checkpoint 和 Restore 时间，测试时的内存占用、临时计算数据大小以及测试结果见下表。检查点建立与重载时间与当时的存储占用大小直接正比相关。

Gaussian 容器的 checkpoint 和 restore 时间				
CPU 数 x 同时运行算例数	内存(GB)	临时文件(GB)	Checkpoint	Restore
24x1	32	7.4	5 分 11 秒	3 分 32 秒
12x2	24	2.8	3 分 53 秒	2 分 39 秒
8x3	15	3.6	2 分 05 秒	1 分 31 秒
6x4	28	6	4 分 58 秒	3 分 17 秒

■ **热迁移时间与内存占用**

本次 Gaussian 程序热迁移的硬盘文件有 31GB，内存文件、计算的临时数据文件以及迁移时间见下表。此次测试数据中，尽管同时运行的算例数不同，但是总的线程数相同，且容器全部数据的存储占用相差不多，故而热迁移时间相差较小。

Gaussian 容器热迁移时间			
CPU 核数 x 运行算例数	内存(GB)	临时文件(GB)	热迁移时间
24x1	35GB	3.3GB	114 分 14 秒
8x3	31GB	7.9GB	109 分 34 秒
6x4	30GB	6.2GB	103 分 56 秒

Singularity 镜像获取 I

pull from Singularity Image:

```
$ singularity pull shub://vsoch/hello-world
```

pull form Docker Image

```
$ singularity pull docker://godlovedc/lolcow
```

build from Singularity shub or Docker docker://

```
$ singularity build hello-world.simg shub://vsoch/hello-world
$ singularity build lolcow.simg docker://godlovedc/lolcow
```

build from recipe file (definition file)

```
# singularity build example_tf_gpu.simg Singularity.latest
# singularity build --sandbox /path/to/image-root-directory/ Singularity.latest
```

这里需要 Root 权限，但可以在自己机器上编译镜像（squashfs 格式）或镜像目录（sandbox）后上传。在 sandbox 模式下，生成容器就是在指定目录下生成一个系统 root 根目录（里面包含/bin /lib /etc 等），类似 chroot 工作模式，可对系统直接修改定制。

recipe file (definition file) I

```
 1 bootstrap:docker
 2 From:nvidia/cuda:8.0-cudnn6-devel-ubuntu16.04
 3
 4 %environment
 5
 6     LD_LIBRARY_PATH=/host-libs:/usr/local/cuda/extras/CUPTI/lib64:/usr/local/cuda/lib64
 7     export LD_LIBRARY_PATH
 8     PATH=/usr/local/cuda/bin:/usr/local/bin:/usr/bin:/bin:/usr/sbin:/sbin
 9     export PATH
10
11 %post
12
13     export DEBIAN_FRONTEND=noninteractive && \
14         apt-get update && apt-get upgrade -y --allow-unauthenticated && \
15         apt-get install -y --allow-unauthenticated \
16             build-essential \
17             cmake \
18             cuda-drivers \
19             curl \
20             git \
21             libfreetype6-dev \
22             libpng12-dev \
23             libssl-dev \
24             libxpm-dev \
25             libzmq3-dev \
26             module-init-tools \
27             pkg-config \
28             python \
29             python-dev \
30             python-tk \
```

recipe file (definition file) II

```
31              python3 \
32              python3-dev \
33              python3-tk \
34              rsync \
35              software-properties-common \
36              unzip \
37              zip \
38              zlib1g-dev \
39              openjdk-8-jdk \
40              openjdk-8-jre-headless \
41              vim \
42              wget
43
44      apt-get clean
45      rm -rf /var/lib/apt/lists/*
46
47      # bazel is required for some TensorFlow projects
48      echo "deb [arch=amd64] http://storage.googleapis.com/bazel-apt stable jdk1.8" >/etc/apt/
            sources.list.d/bazel.list
49      curl https://bazel.build/bazel-release.pub.gpg | apt-key add -
50
51      export DEBIAN_FRONTEND=noninteractive && \
52          apt-get update && \
53          apt-get install -y --allow-unauthenticated \
54              bazel
55
56      curl -O https://bootstrap.pypa.io/get-pip.py && \
57          python get-pip.py && \
58          rm get-pip.py
59
```

recipe file (definition file) III

```
60      pip --no-cache-dir install \
61              h5py \
62              ipykernel \
63              jupyter \
64              matplotlib \
65              numpy \
66              pandas \
67              Pillow \
68              scipy \
69              sklearn
70
71      python -m ipykernel.kernelspec
72
73      echo "/usr/local/cuda/lib64/" >/etc/ld.so.conf.d/cuda.conf
74      echo "/usr/local/cuda/extras/CUPTI/lib64/" >>/etc/ld.so.conf.d/cuda.conf
75
76      # Install TensorFlow GPU version
77      pip uninstall tensorflow-gpu || true
78      pip install --upgrade tensorflow-gpu==1.4
79
80      # keras
81      pip install --upgrade keras
82
83      ##############################
84      # now do the same for python3
85
86      curl -O https://bootstrap.pypa.io/get-pip.py && \
87          python3 get-pip.py && \
88          rm get-pip.py
89
```

recipe file (definition file) IV

```
90      pip3 --no-cache-dir install \
91              h5py \
92              ipykernel \
93              jupyter \
94              matplotlib \
95              numpy \
96              pandas \
97              Pillow \
98              scipy \
99              sklearn
100
101     python3 -m ipykernel.kernelspec
102
103     # Install TensorFlow GPU version
104     pip3 uninstall tensorflow-gpu || true
105     pip3 install --upgrade tensorflow-gpu==1.4
106
107     # keras
108     pip3 install --upgrade keras
109
110     ############################
111
112     # make sure we have a way to bind host provided libraries
113     # see https://github.com/singularityware/singularity/issues/611
114     mkdir -p /host-libs /etc/OpenCL/vendors
115     echo "/host-libs/" >/etc/ld.so.conf.d/000-host-libs.conf
116
117     # required directories
118     mkdir -p /cvmfs
119
```

recipe file (definition file) V

```
120     # root
121     cd /opt && \
122         wget -nv https://root.cern.ch/download/root_v6.10.02.Linux-ubuntu16-x86_64-gcc5.4.tar.
                gz && \
123         tar xzf root_v6.10.02.Linux-ubuntu16-x86_64-gcc5.4.tar.gz && \
124         rm -f root_v6.10.02.Linux-ubuntu16-x86_64-gcc5.4.tar.gz
125
126     # xrootd
127     cd /opt && \
128         wget http://xrootd.org/download/v4.7.1/xrootd-4.7.1.tar.gz && \
129         tar xzf xrootd-4.7.1.tar.gz && \
130         cd xrootd-4.7.1 && \
131         mkdir build && \
132         cd  build && \
133         cmake /opt/xrootd-4.7.1 -DCMAKE_INSTALL_PREFIX=/opt/xrootd -DENABLE_PERL=FALSE && \
134         make && \
135         make install && \
136         cd /opt && \
137         rm -rf xrootd-4.7.1.tar.gz xrootd-4.7.1
138
139     # stashcp
140     cd /opt && \
141         git clone https://github.com/opensciencegrid/StashCache.git
142
143     # build info
144     echo "Timestamp:" `date --utc` | tee /image-build-info.txt
```

Singularity exec I

加载 Singularity 容器环境

```
1 [wszhang@tcadmin wszhang]$ module load singularity/3.0.2
2
3 [wszhang@tcadmin wszhang]$ singularity pull shub://vsoch/hello-world
4 Progress |==================================| 100.0%
5 Done. Container is at: /home2/wszhang/vsoch-hello-world-master-latest.simg
6
7 [wszhang@tcadmin wszhang]$ singularity exec
      /home2/wszhang/vsoch-hello-world-master-latest.simg grep LTS /etc/os-release
8 VERSION="14.04.5 LTS, Trusty Tahr"
9 PRETTY_NAME="Ubuntu 14.04.5 LTS"
```

加载 GPU: --nv, 加载系统目录: --bind /opt

```
$ singularity exec --nv --bind /opt /.../vsoch-hello-world-master-latest.simg
```

容器 Shell 界面

```
$ singularity shell /.../vsoch-hello-world-master-latest.simg
```

容器化具有外部依赖库的程序 I

如何容器化具有商业授权组件的应用？是否应该在容器中包含 Intel MKL 之类的体积巨大的库?

为了简便构造并公开分发此类应用容器，可能不方便在容器中包含这些工具链，此时，可以选择在容器内引用外部依赖库（只是与容器技术愿景有所背离）。假设，我们有程序库处于宿主系统的/opt 目录，我们需要：

- 编辑文件/etc/ld.so.conf.d/host-libs.conf，添加库路径：

  ```
  $ singularity shell -w -B /opt /home2/wszhang/singularity/sandbox/ vim /etc/ld.so.conf.d
  /host-libs.conf
  ```

  （其中 -w 参数表示将保存对容器的修改）

- 更新容器中的共享库 Cache: /etc/ld.so.cache$ singularity shell -w -B /opt /home2/wszhang/singularity/sandbox/ ldconfig

LSF 下提交 Singularity 作业 I

```
[wszhang@tcadmin wszhang]$ bsub -n 4 -q testv3 -oo %J.log mpijob singularity exec
      /home2/wszhang/vsoch-hello-world-master-latest.simg hostname
Job <1268775> is submitted to queue <testv3>.

[wszhang@tcadmin wszhang]$ cat 1268775.log
Sender: LSF System <lsfadmin@node284>
Subject: Job 1268775: <mpijob singularity exec /home2/wszhang/vsoch-hello-world-master-latest.
      simg hostname> in cluster <tc4600> Done

Job <mpijob singularity exec /home2/wszhang/vsoch-hello-world-master-latest.simg hostname> was
      submitted from host <tcadmin> by user <wszhang> in cluster <tc4600>.
Job was executed on host(s) <4*node284>, in queue <testv3>, as user <wszhang> in cluster <
      tc4600>.
</home/nic/wszhang> was used as the home directory.
</home2/wszhang> was used as the working directory.
Started at Results reported on
Your job looked like:

------------------------------------------------------------
# LSBATCH: User input
mpijob singularity exec /home2/wszhang/vsoch-hello-world-master-latest.simg hostname
------------------------------------------------------------

Successfully completed.

Resource usage summary:

    CPU time :                                   0.49 sec.
```

LSF 下提交 Singularity 作业 II

```
25     Max Memory :                             -
26     Average Memory :                         -
27     Total Requested Memory :                 -
28     Delta Memory :                           -
29     Max Swap :                               -
30     Max Processes :                          -
31     Max Threads :                            -
32     Run time :                               1 sec.
33     Turnaround time :                        2 sec.
34
35 The output (if any) follows:
36
37 node284
38 node284
39 node284
40 node284
```

交互式提交并测试 Singularity 环境 I

```
[wszhang@tcadmin wszhang]$ bsub -n 28 -q k80 -Is singularity shell --nv --bind /opt
      /home2/wszhang/singularity/vsoch-hello-world-master-latest.simg
Job <1358694> is submitted to queue <k80>.
<<Waiting for dispatch ...>>
<<Starting on k803>>
Singularity vsoch-hello-world-master-latest.simg:~/eclipse_project/newgit/abacus-NewGit/ABACUS
      .1.0.0/source> nvidia-smi
Thu Mar 14 20:32:55 2019
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 396.26                 Driver Version: 396.26                    |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|===============================+======================+======================|
|   0  Tesla K80           Off  | 00000000:05:00.0 Off |                    0 |
| N/A   27C    P8    27W / 149W |     11MiB / 11441MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+
|   1  Tesla K80           Off  | 00000000:06:00.0 Off |                    0 |
| N/A   32C    P8    29W / 149W |     11MiB / 11441MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+
|   2  Tesla K80           Off  | 00000000:85:00.0 Off |                    0 |
| N/A   30C    P8    27W / 149W |     11MiB / 11441MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+
|   3  Tesla K80           Off  | 00000000:86:00.0 Off |                    0 |
| N/A   28C    P8    29W / 149W |     11MiB / 11441MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+
+-----------------------------------------------------------------------------+
| Processes:                                                       GPU Memory |
```

交互式提交并测试 Singularity 环境 II

```
27 |  GPU       PID   Type   Process name                             Usage      |
28 |=============================================================================|
29 |  No running processes found                                                 |
30 +-----------------------------------------------------------------------------+
31 Singularity vsoch-hello-world-master-latest.simg:~/eclipse_project/newgit/abacus-NewGit/ABACUS
       .1.0.0/source> which
       nvidia-smi
32 /usr/bin/nvidia-smi
33 Singularity vsoch-hello-world-master-latest.simg:~/eclipse_project/newgit/abacus-NewGit/ABACUS
       .1.0.0/source> exit
```

联系信息

中国科学技术大学超算中心

办公室科大东区新图书馆一楼东侧 126 室

电话:0551-63602248

信箱:sccadmin@ustc.edu.cn

主页:http://scc.ustc.edu.cn