Corporate Tools for GPU Access and Software Development

Alex Gartner, Senior Software Systems Engineer

Christine Harvey, Lead High Performance & Analytic Computing Engineer

The view, opinions, and/or findings contained in this report are those of The MITRE Corporation and should not be construed as an official Government position, policy, or decision, unless designated by other documentation.

This technical data deliverable was developed using contract funds under Basic Contract No. W56KGU-18-D-0004.

HPC at MITRE Overview

HPC at MITRE

MITRE has two High Performance Computing (HPC) systems managed by the Enterprise Technical Computing (ETC) center

Senate

- Located at the McLean Campus
- Purchased from PSSC Labs in FY16
- 896 Compute Cores
- 10 GPUs (K80 and M40)
- 180+ TB of raw Storage Space
- 56 Gbps Infiniband Network

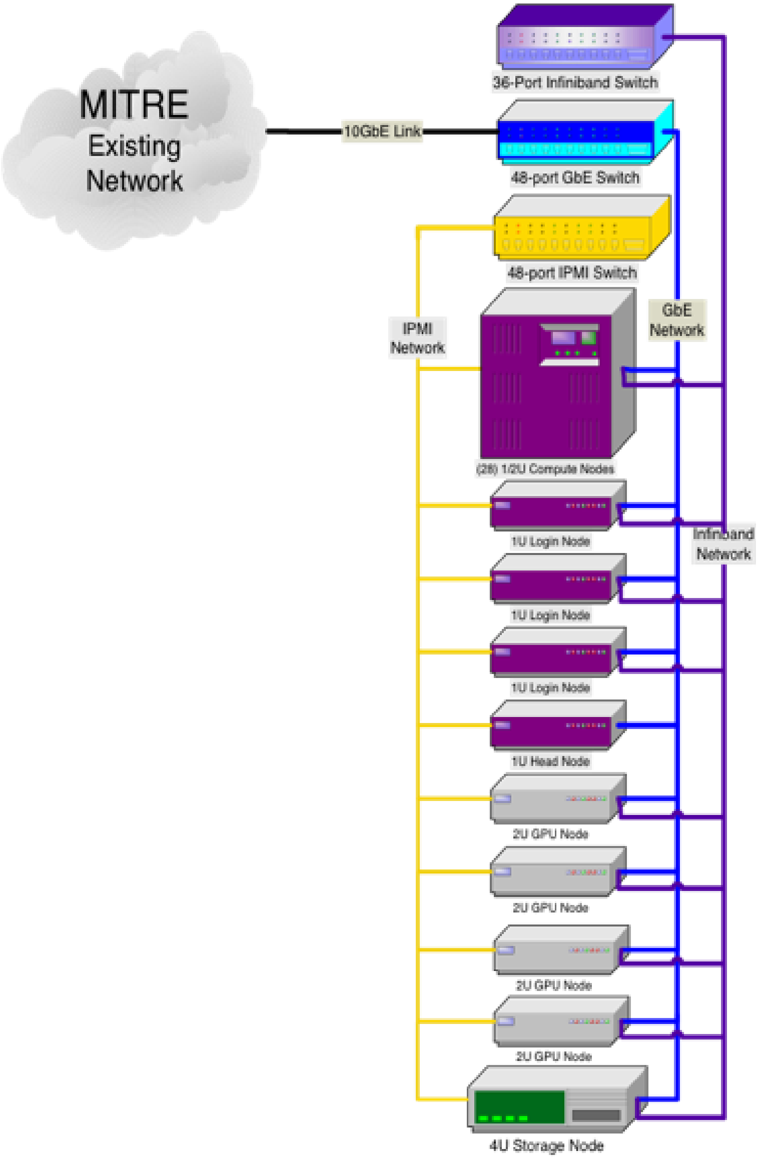
Sammet

- Located at Bedford Campus
- Purchased from Microway in FY17
- 240 Compute Cores
- 24 GPUs (P100s and V100s)
- 500 TB Storage
- 100Gbps Intel Omnipath Network
- 1 FPGA-ready Node

Access to HPC systems is provided at no cost to projects.

Senate Architecture

- **Head Node (Senate)**
- **3 Login / Development Nodes (Senator01-03)**
  - 2x Intel Xeon E5-2630v4 Processors (10 Cores Each)
  - 128 GB High Performance DDR4 2133 MHz ECC
- **28 Compute Nodes (node001-028)**
  - 2x Intel Xeon E5-2660v4 Processors (14 Cores Each)
  - 192 GB High Performance DDR4 2133 MHz ECC Registered System Memory
  - 1 TB SATAIII 7200 PRM Enterprise Hard Drives
- **4 GPU Compute Nodes (gnode001-004)**
  - 2x K80 Nodes with 2x NVIDIA K80 Cards Each
  - 2x M40 Nodes with an NVIDIA M40 Tesla Card Each
- **1 Storage Node**
  - 144 TB Raw Storage Space

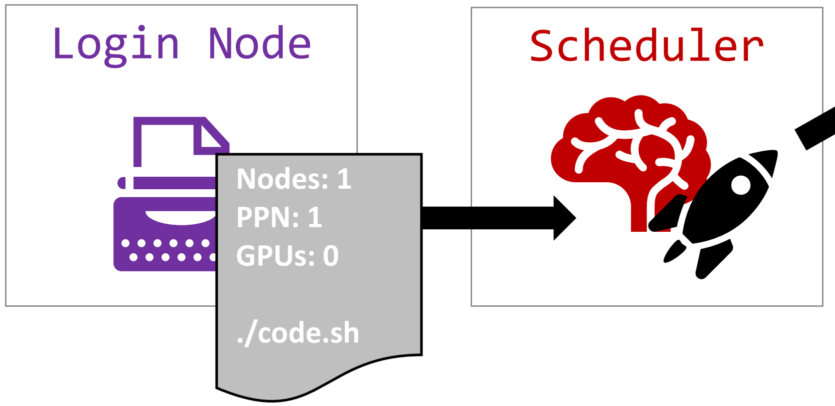


Sammet Architecture

- **Head Node (Sammet)**
- **2 Login / Development Nodes (Sammet01-02)**
  - 2x Intel Xeon E5-2630v4 Processors (10 Cores Each)
  - 256GB DDR4 2400 MHz ECC/Registered Memory
- **10 GPU Nodes (gpu1-10)**
  - 2x Intel Xeon E5-2630v4 Processors (10 Cores Each)
  - 256GB DDR4 2400 MHz ECC/Registered Memory
  - 1 TB Seagate Enterprise Capacity SATA 512E
  - 2x P100 16GB GPUs
- **1 (Project Owned) GPU Node (gpu11)**
- **1 FPGA-ready Node (fpga1)**
  - Space for four full-height, full-length FPGA cards
- **1 Storage Node**
  - 500 TB Seagate Enterprise Capacity 3.5" V6 SATA 6Gbps

Resource Managers and Job Scheduling

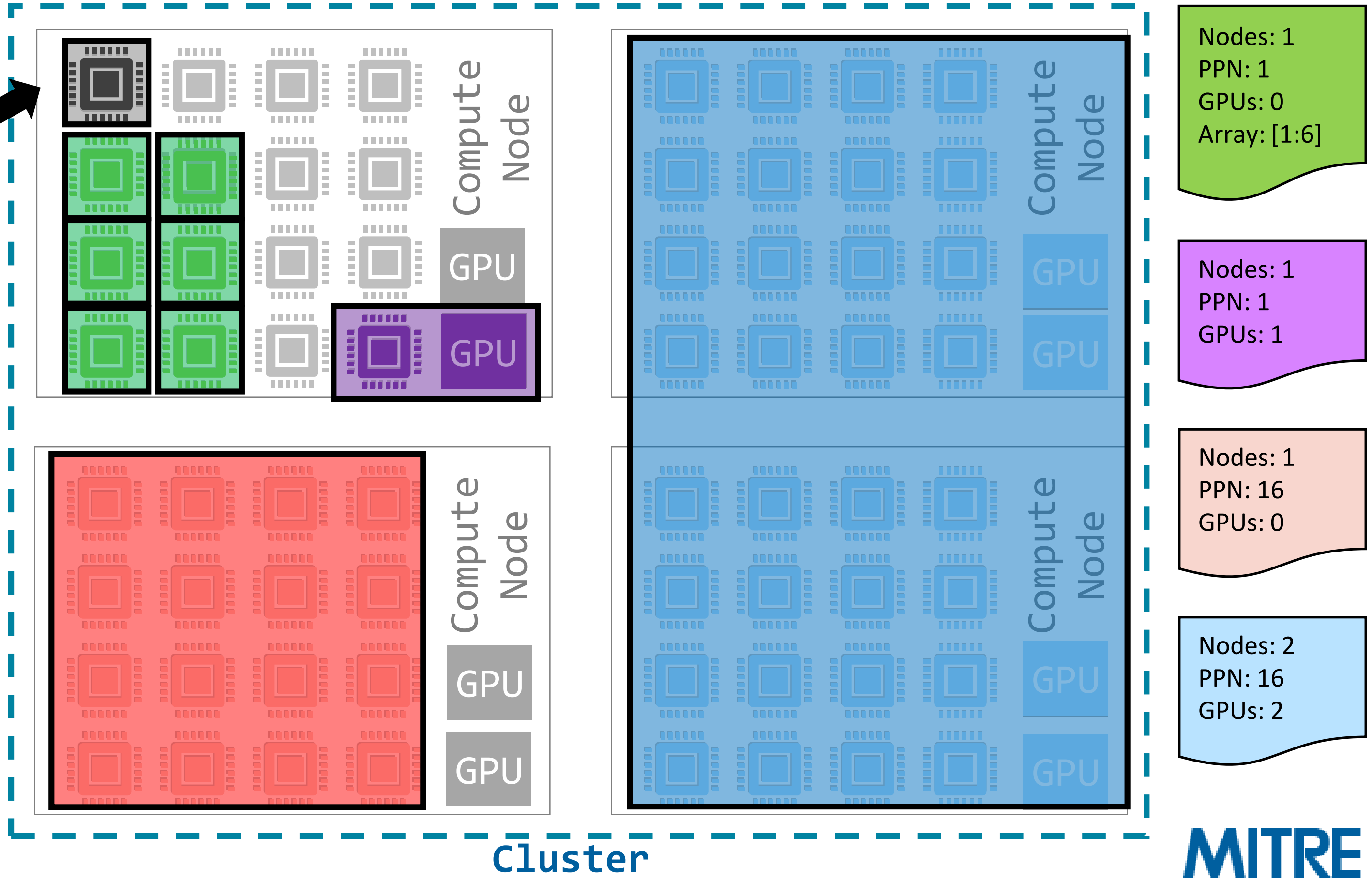
- **The scheduler allocates resources based on the availability of nodes, processors, GPUs, and ongoing/incoming requests**



Nodes: 1
PPN: 1
GPUs: 0

./code.sh

A job is a request for compute resources needed to perform computational work



Nodes: 1
PPN: 1
GPUs: 0
Array: [1:6]

Nodes: 1
PPN: 1
GPUs: 1

Nodes: 1
PPN: 16
GPUs: 0

Nodes: 2
PPN: 16
GPUs: 2

User Experience

1. **Attend Training**

2. **Build environments (Anaconda, CUDA, …)**

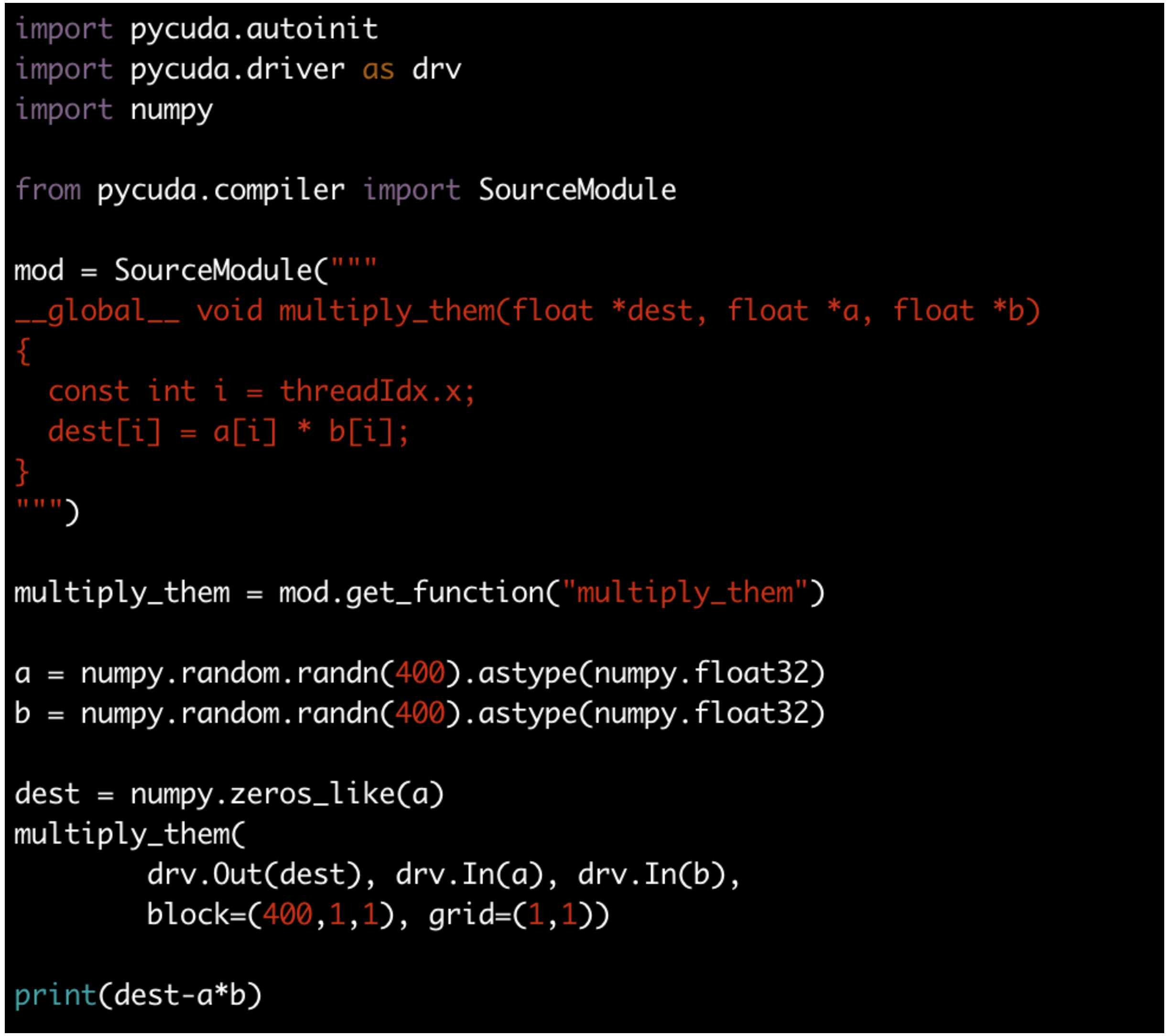
3. **Develop analysis/modeling code**
   - Perform debugging using *interactive* reservations on the cluster

4. **Develop scripts for job submission to the cluster**

5. **Submit jobs to the scheduler**

A Simple Example: hello_world_pycuda.py

hello_world_pycuda.sh

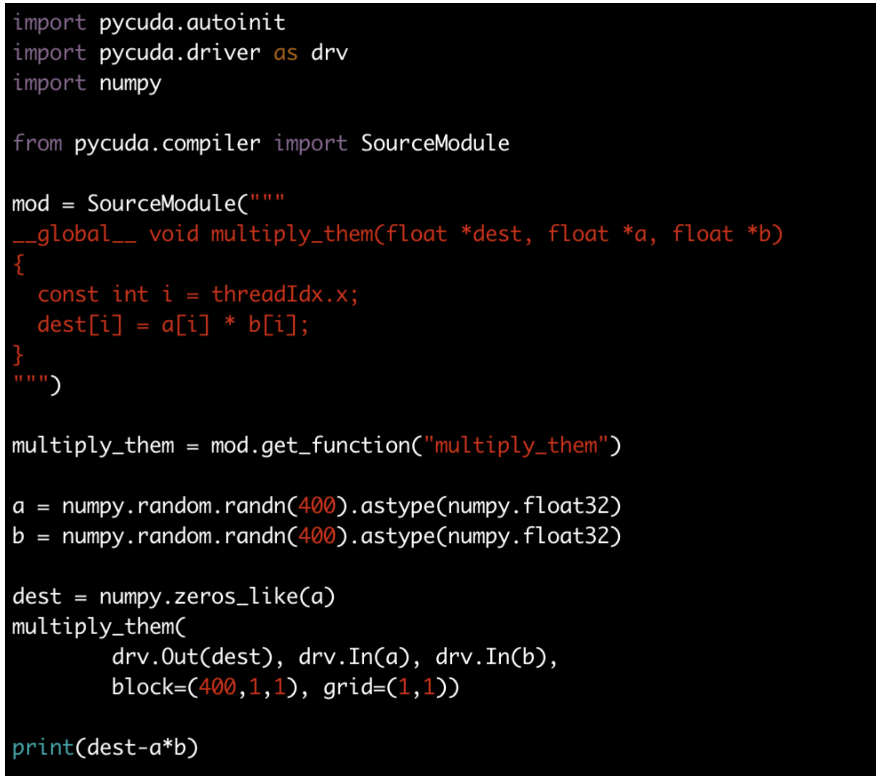


```
import pycuda.autoinit
import pycuda.driver as drv
import numpy

from pycuda.compiler import SourceModule

mod = SourceModule("""
__global__ void multiply_them(float *dest, float *a, float *b)
{
  const int i = threadIdx.x;
  dest[i] = a[i] * b[i];
}
""")

multiply_them = mod.get_function("multiply_them")

a = numpy.random.randn(400).astype(numpy.float32)
b = numpy.random.randn(400).astype(numpy.float32)

dest = numpy.zeros_like(a)
multiply_them(
        drv.Out(dest), drv.In(a), drv.In(b),
        block=(400,1,1), grid=(1,1))

print(dest-a*b)
```

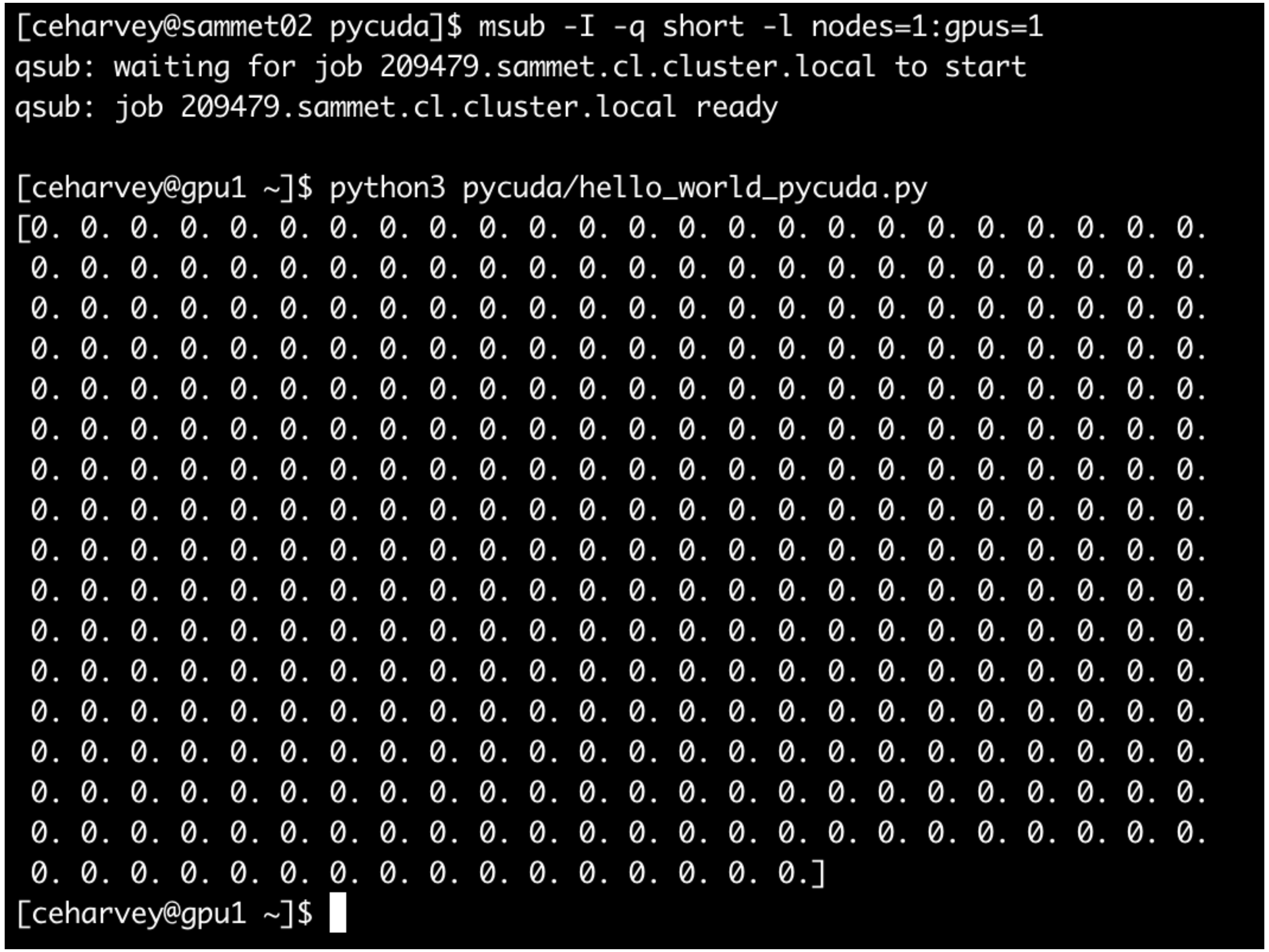
Example from https://documen.tician.de/pycuda/

A Simple Example: `hello_world_pycuda.py`

hello_world_pycuda.sh



```
import pycuda.autoinit
import pycuda.driver as drv
import numpy

from pycuda.compiler import SourceModule

mod = SourceModule("""
__global__ void multiply_them(float *dest, float *a, float *b)
{
  const int i = threadIdx.x;
  dest[i] = a[i] * b[i];
}
""")

multiply_them = mod.get_function("multiply_them")

a = numpy.random.randn(400).astype(numpy.float32)
b = numpy.random.randn(400).astype(numpy.float32)

dest = numpy.zeros_like(a)
multiply_them(
        drv.Out(dest), drv.In(a), drv.In(b),
        block=(400,1,1), grid=(1,1))

print(dest-a*b)
```

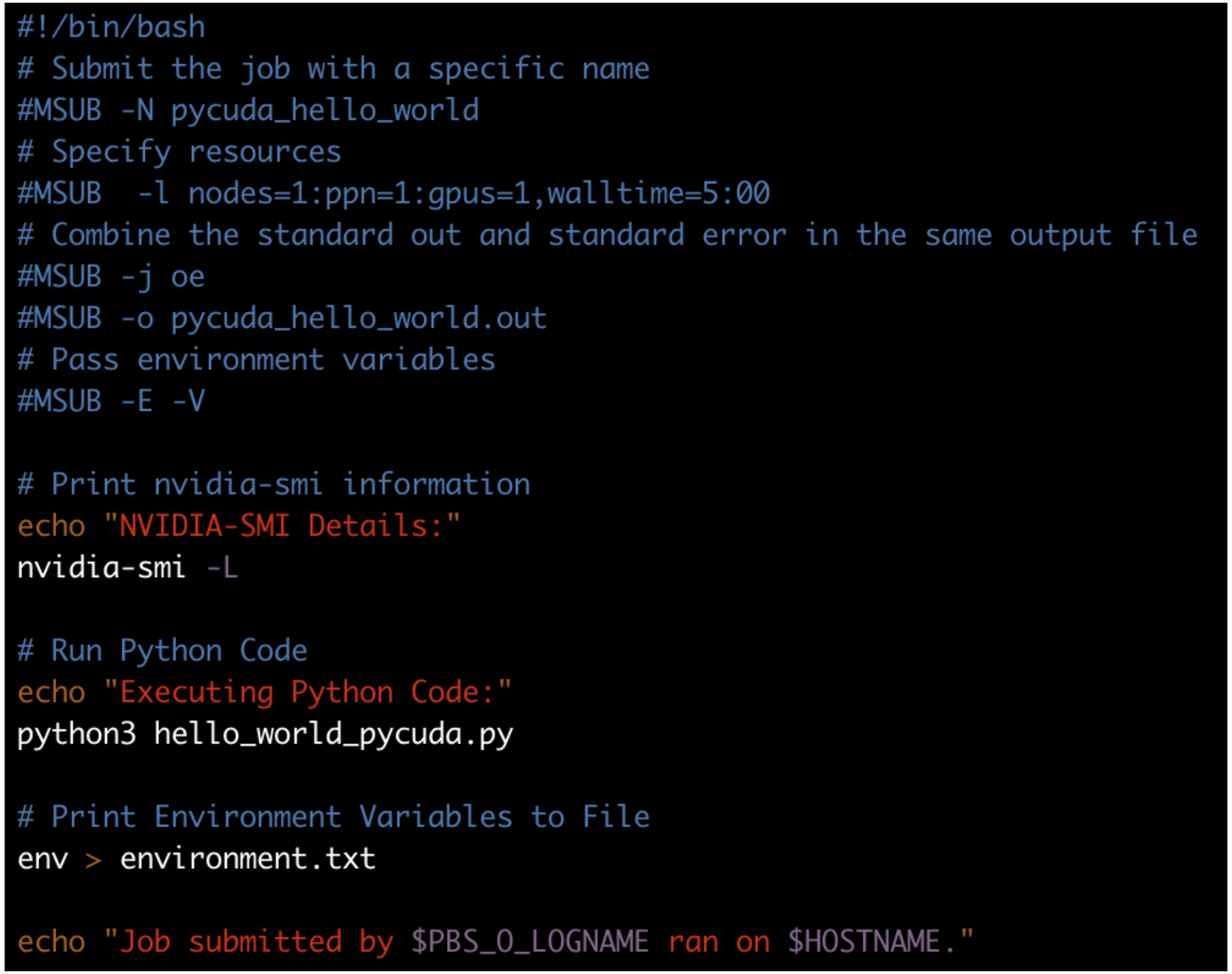


```
[ceharvey@sammet02 pycuda]$ msub -I -q short -l nodes=1:gpus=1
qsub: waiting for job 209479.sammet.cl.cluster.local to start
qsub: job 209479.sammet.cl.cluster.local ready

[ceharvey@gpu1 ~]$ python3 pycuda/hello_world_pycuda.py
[0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
[ceharvey@gpu1 ~]$
```

Example from https://documen.tician.de/pycuda/

A Simple Example: run_pycuda.sh

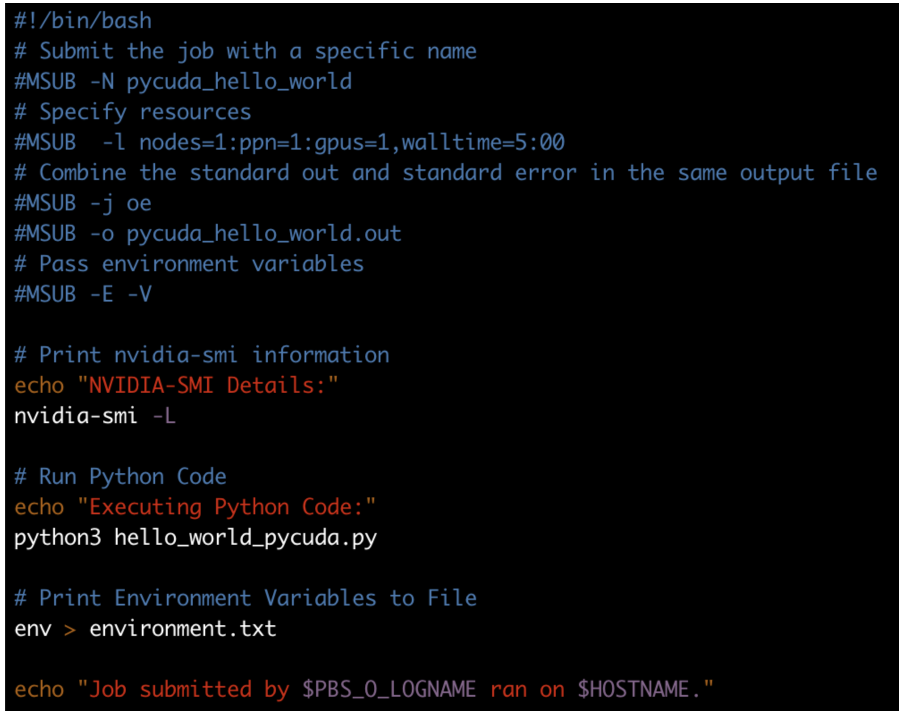
run_pycuda.sh



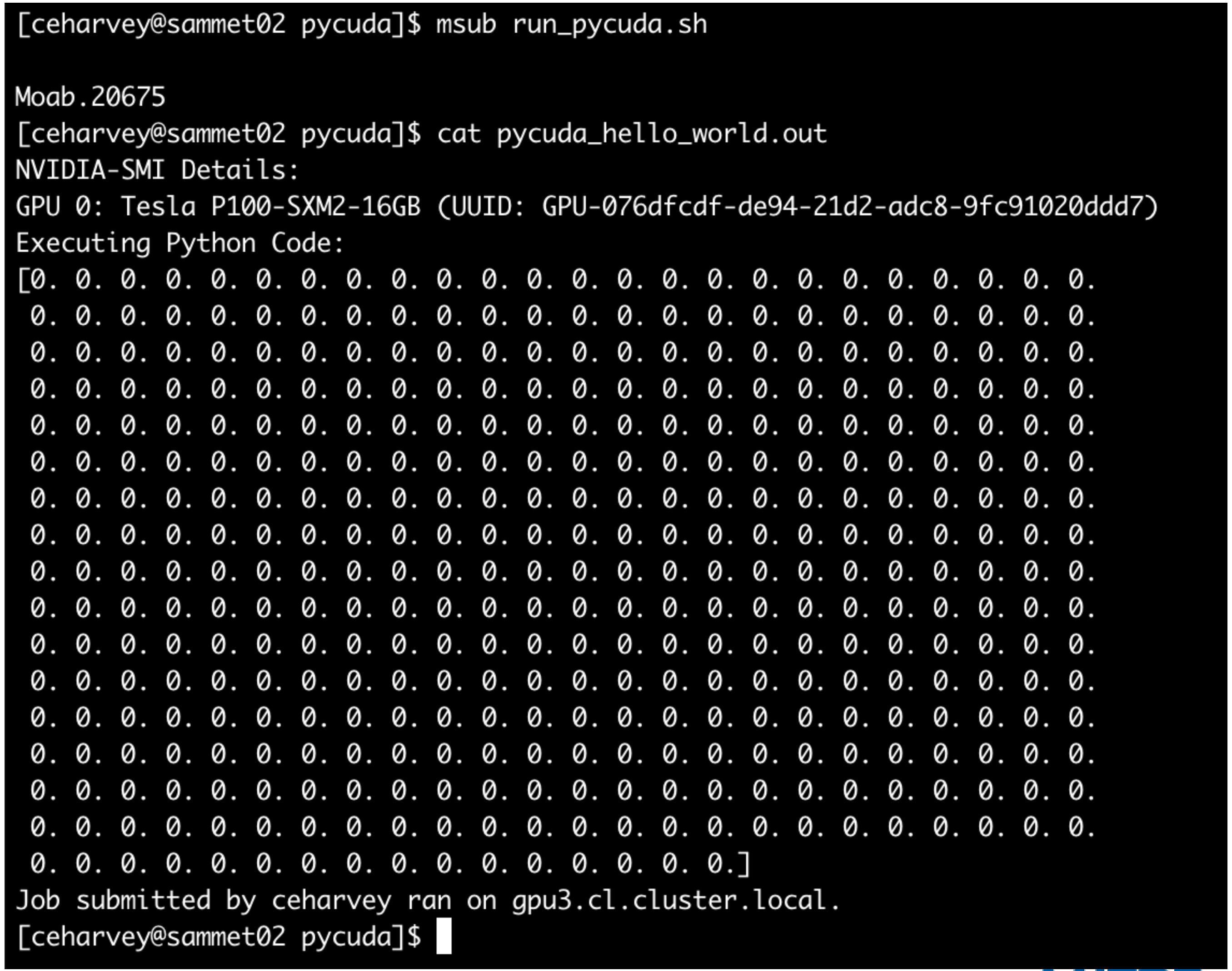
```
#!/bin/bash
# Submit the job with a specific name
#MSUB -N pycuda_hello_world
# Specify resources
#MSUB  -l nodes=1:ppn=1:gpus=1,walltime=5:00
# Combine the standard out and standard error in the same output file
#MSUB -j oe
#MSUB -o pycuda_hello_world.out
# Pass environment variables
#MSUB -E -V

# Print nvidia-smi information
echo "NVIDIA-SMI Details:"
nvidia-smi -L

# Run Python Code
echo "Executing Python Code:"
python3 hello_world_pycuda.py

# Print Environment Variables to File
env > environment.txt

echo "Job submitted by $PBS_O_LOGNAME ran on $HOSTNAME."
```

A Simple Example: `run_pycuda.sh`

run_pycuda.sh



```bash
#!/bin/bash
# Submit the job with a specific name
#MSUB -N pycuda_hello_world
# Specify resources
#MSUB  -l nodes=1:ppn=1:gpus=1,walltime=5:00
# Combine the standard out and standard error in the same output file
#MSUB -j oe
#MSUB -o pycuda_hello_world.out
# Pass environment variables
#MSUB -E -V

# Print nvidia-smi information
echo "NVIDIA-SMI Details:"
nvidia-smi -L

# Run Python Code
echo "Executing Python Code:"
python3 hello_world_pycuda.py

# Print Environment Variables to File
env > environment.txt

echo "Job submitted by $PBS_O_LOGNAME ran on $HOSTNAME."
```



```
[ceharvey@sammet02 pycuda]$ msub run_pycuda.sh

Moab.20675
[ceharvey@sammet02 pycuda]$ cat pycuda_hello_world.out
NVIDIA-SMI Details:
GPU 0: Tesla P100-SXM2-16GB (UUID: GPU-076dfcdf-de94-21d2-adc8-9fc91020ddd7)
Executing Python Code:
[0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]
Job submitted by ceharvey ran on gpu3.cl.cluster.local.
[ceharvey@sammet02 pycuda]$
```

Project and User Overview

- **113 Projects**
  - Text / PDF document analysis
  - Video and Image processing
  - Large-scale distributed simulations and “embarrassingly parallel” problems
  - Machine and Deep Learning on GPUs

- **289 Users**
  - Users are from 56+ Different Departments
    - Human Language Technology
    - Data Analytics
    - Transportation Data Analytics
    - Cognitive Science & Artificial Intelligence
    - Communications SIGINT, & PNT
    - Data Engineering & Biometrics
    - Model-based Analytics

HPC Services

The ETC HPC team provides the following:

- Account Creation
- Training for HPC Systems and Schedulers
- Project Consultation Meetings
- Software installs, management, and upkeep
- Ticket assistance: project questions, scheduler problems, parallelization help, general debugging assistance

All HPC services are provided at no cost to projects with the exception of extensive environment configuration or assistance with developing distributed code.

Problem

- **DL/ML users need a simple platform to tinker**
- **Existing platforms not flexible enough**

container-manager toolset

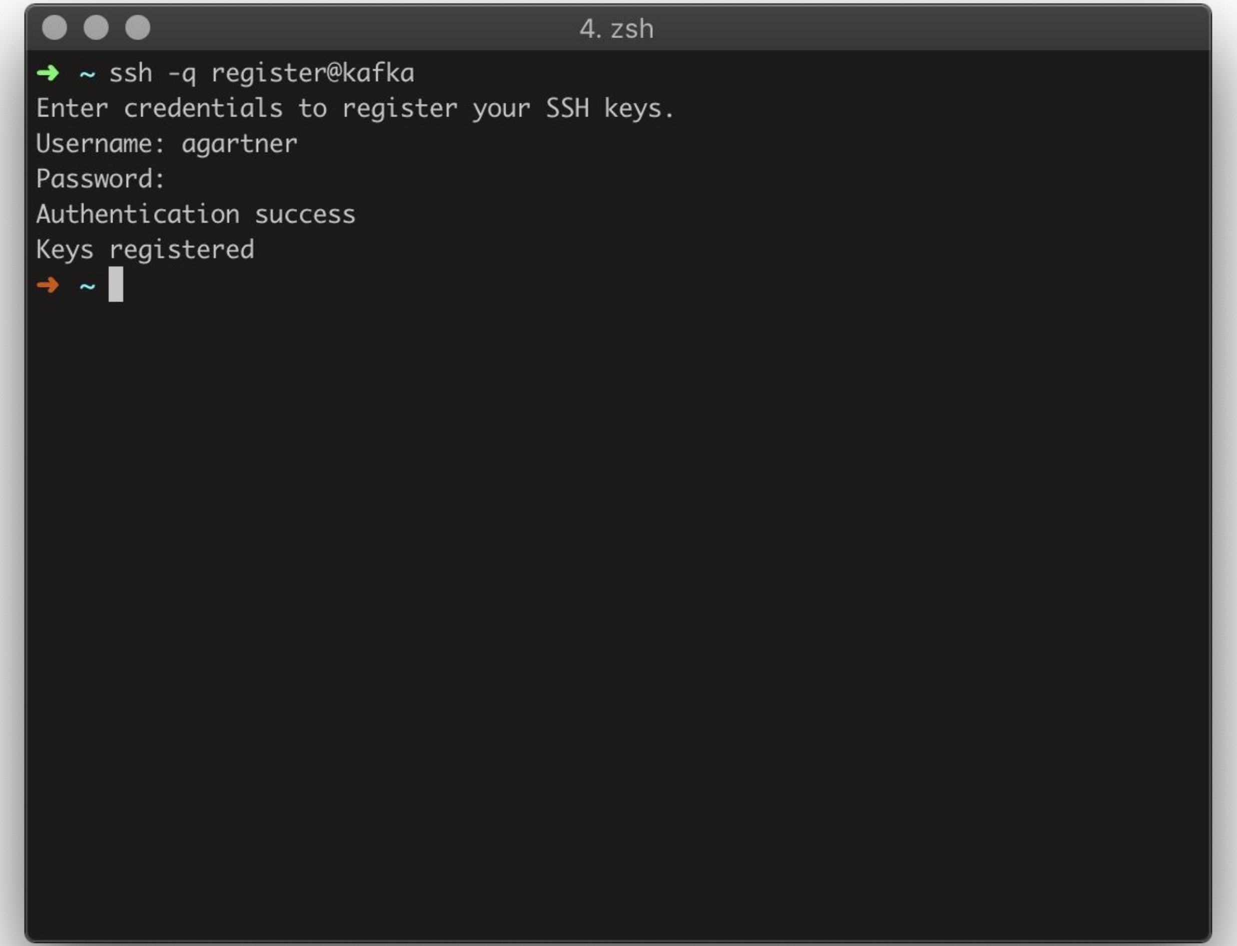
Solution

- Use lxd to provide system containers
- Develop SSH infrastructure to manage and access containers
- Develop GPU manager/scheduler

Single Host Design

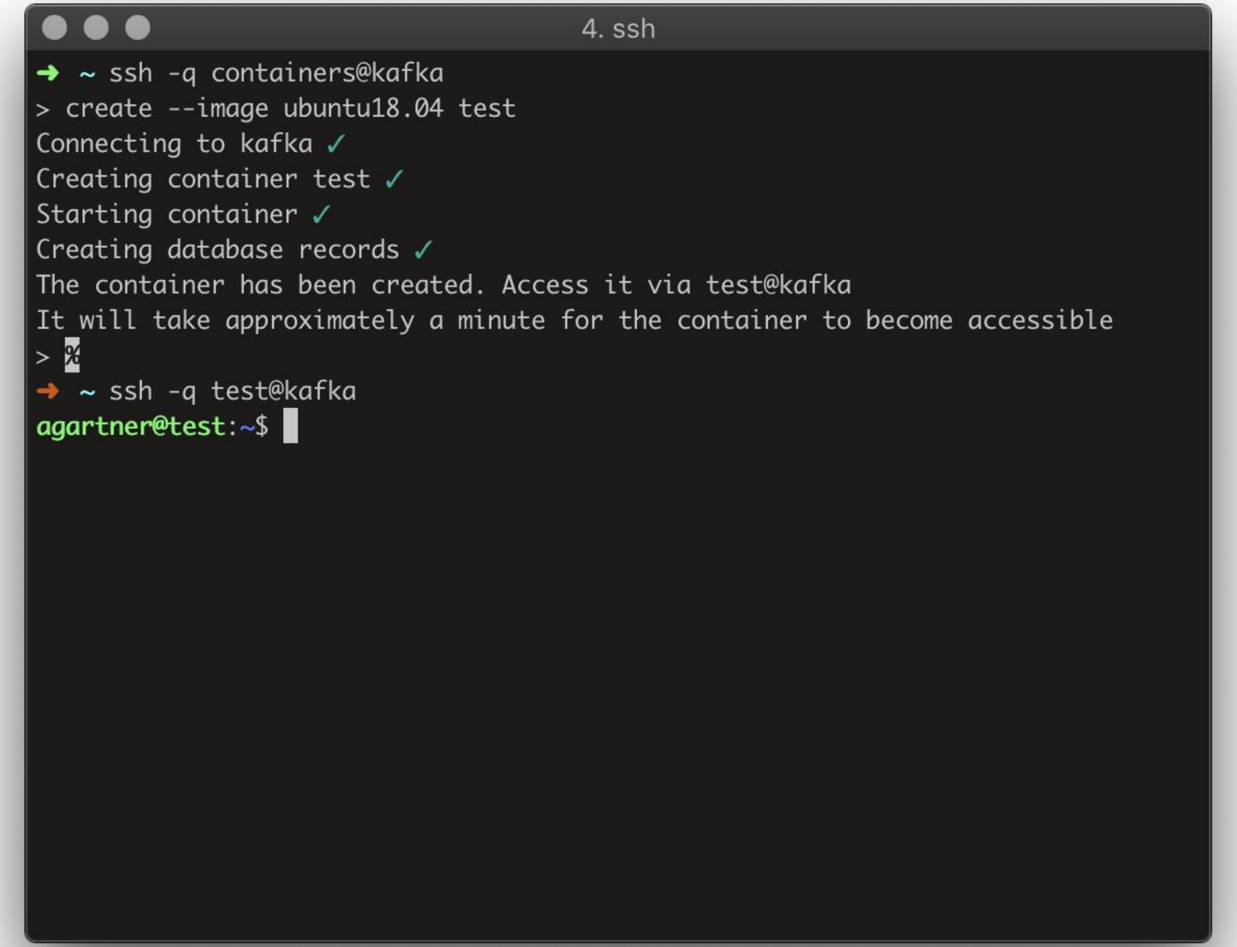
```
                                +------------------------------------+
  User Laptops                  |                    Containers      |
+-------------+                 |                  +-------------+   |
|    alex     | -----> +-----------------+ ------> |    alex     |   |
+-------------+        |                 |         +-------------+   |
                       |Connection Broker|                           |
+-------------+        |                 |         +-------------+   |
|     bob     | -----> +-----------------+ ------> |     bob     |   |
+-------------+                 |                  +-------------+   |
                                |                                    |
                                +------------------------------------+
```

Registration



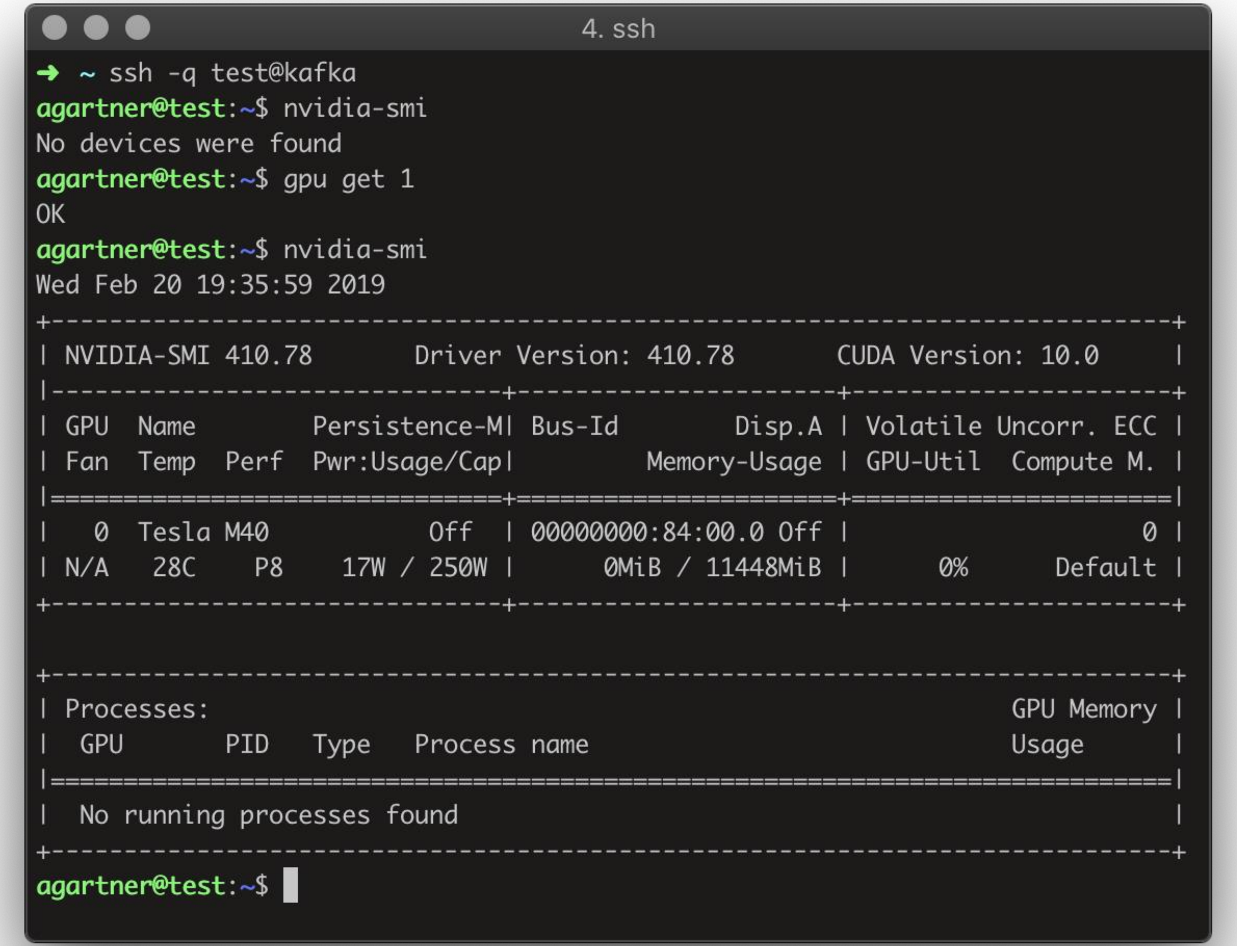
```
➜ ~ ssh -q register@kafka
Enter credentials to register your SSH keys.
Username: agartner
Password:
Authentication success
Keys registered
➜ ~ 
```

Creation and Access



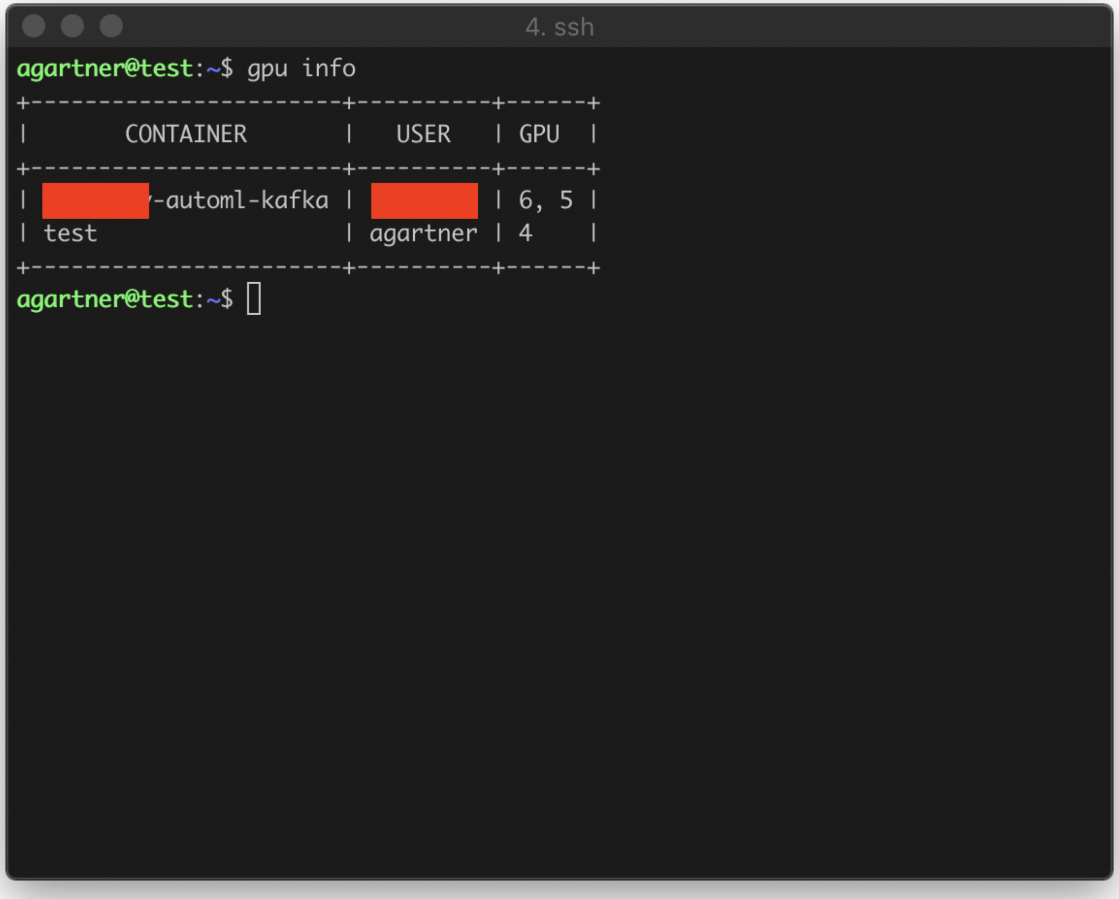
```
➜  ~ ssh -q containers@kafka
> create --image ubuntu18.04 test
Connecting to kafka ✓
Creating container test ✓
Starting container ✓
Creating database records ✓
The container has been created. Access it via test@kafka
It will take approximately a minute for the container to become accessible
> %
➜  ~ ssh -q test@kafka
agartner@test:~$
```

GPU Management

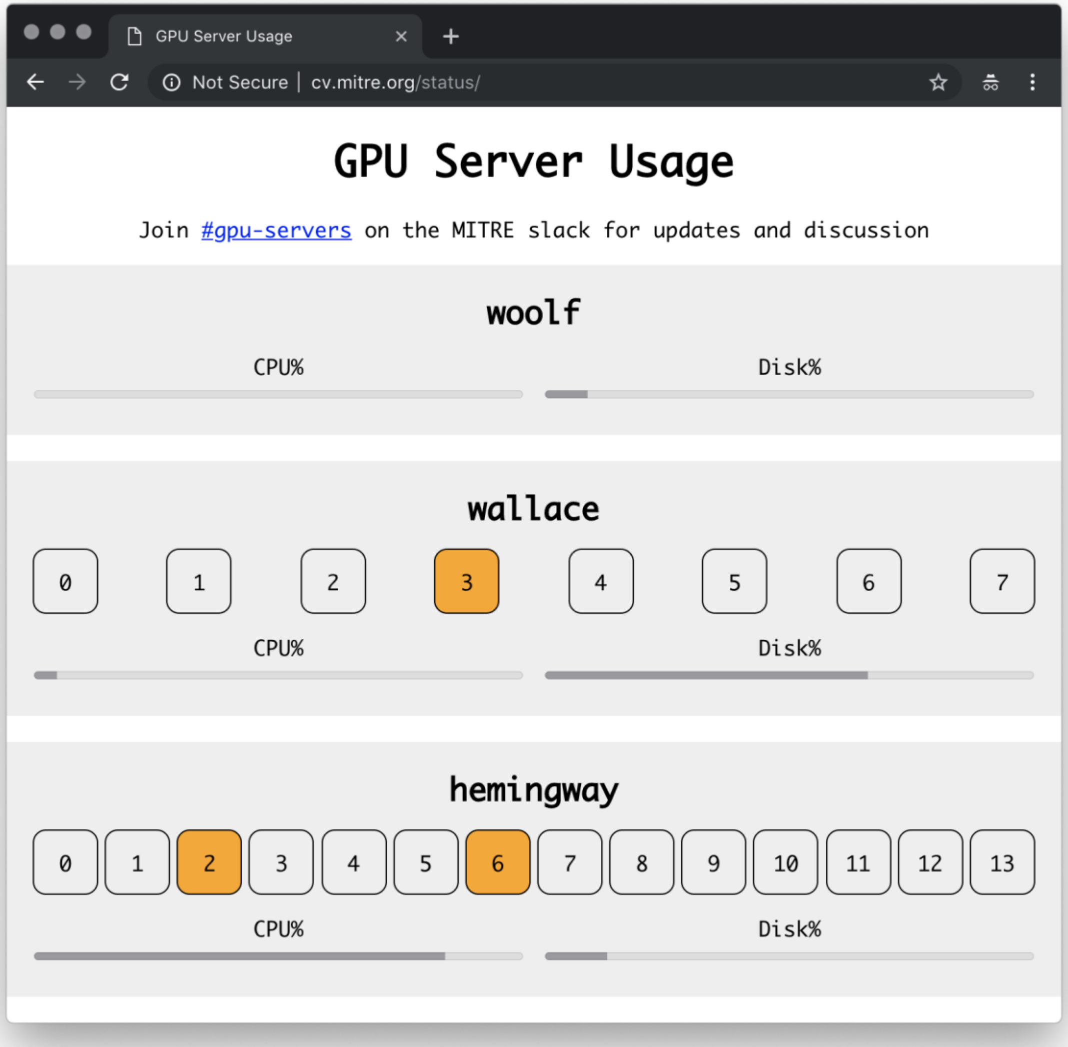


```
➜  ~ ssh -q test@kafka
agartner@test:~$ nvidia-smi
No devices were found
agartner@test:~$ gpu get 1
OK
agartner@test:~$ nvidia-smi
Wed Feb 20 19:35:59 2019
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 410.78       Driver Version: 410.78       CUDA Version: 10.0     |
|-------------------------------+----------------------+----------------------+
| 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 M40           Off  | 00000000:84:00.0 Off |                    0 |
| N/A   28C    P8    17W / 250W |      0MiB / 11448MiB |      0%      Default |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                       GPU Memory |
|  GPU       PID   Type   Process name                             Usage      |
|=============================================================================|
|  No running processes found                                                 |
+-----------------------------------------------------------------------------+
agartner@test:~$ 
```

Status and Monitoring



```
agartner@test:~$ gpu info
+----------------------+---------+------+
|       CONTAINER      |  USER   | GPU  |
+----------------------+---------+------+
| [redacted]-automl-kafka | [redacted] | 6, 5 |
| test                 | agartner | 4    |
+----------------------+---------+------+
agartner@test:~$ 
```



GPU Server Usage

Join #gpu-servers on the MITRE slack for updates and discussion

woolf

CPU% Disk%

wallace

0 1 2 3 4 5 6 7

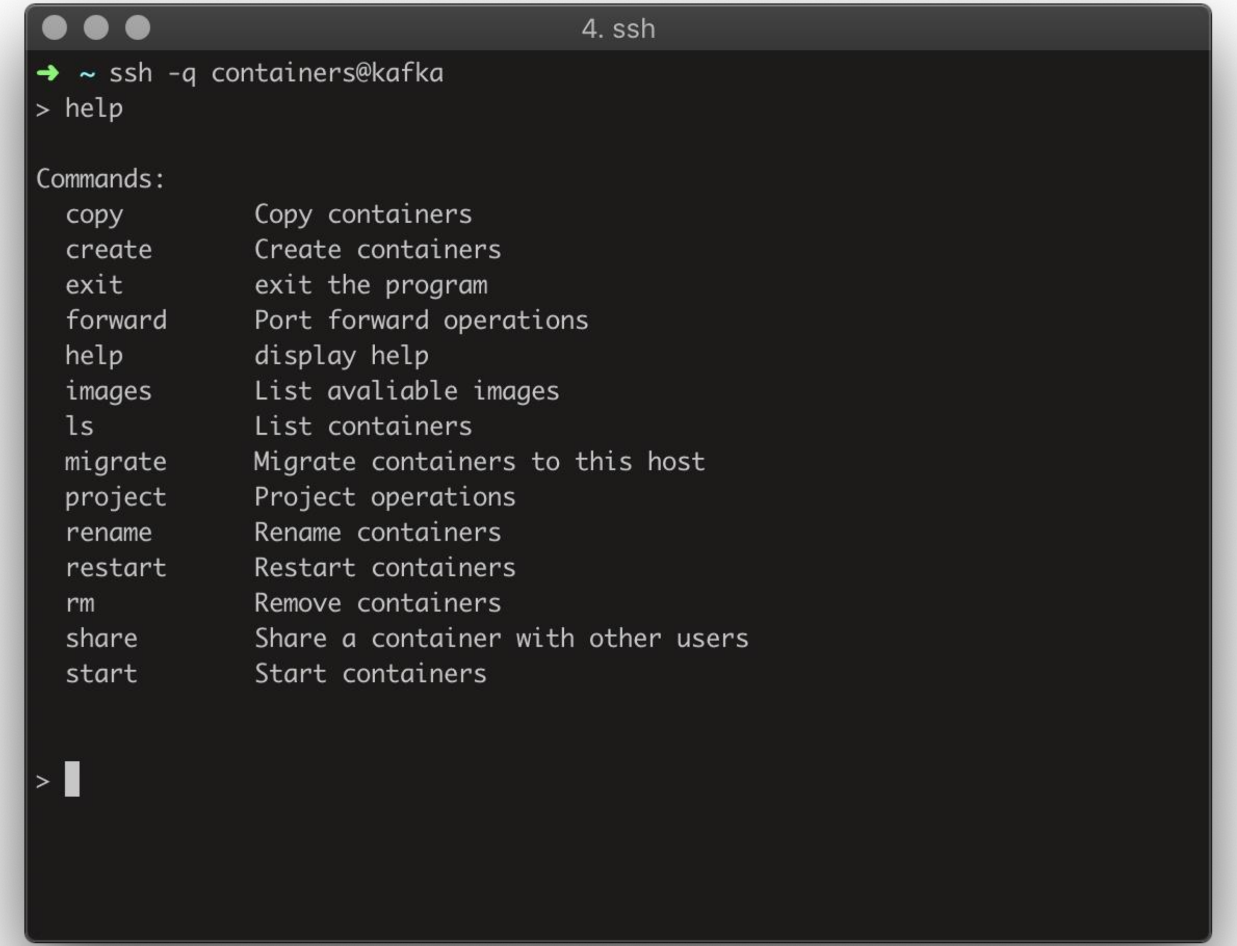
CPU% Disk%

hemingway

0 1 2 3 4 5 6 7 8 9 10 11 12 13

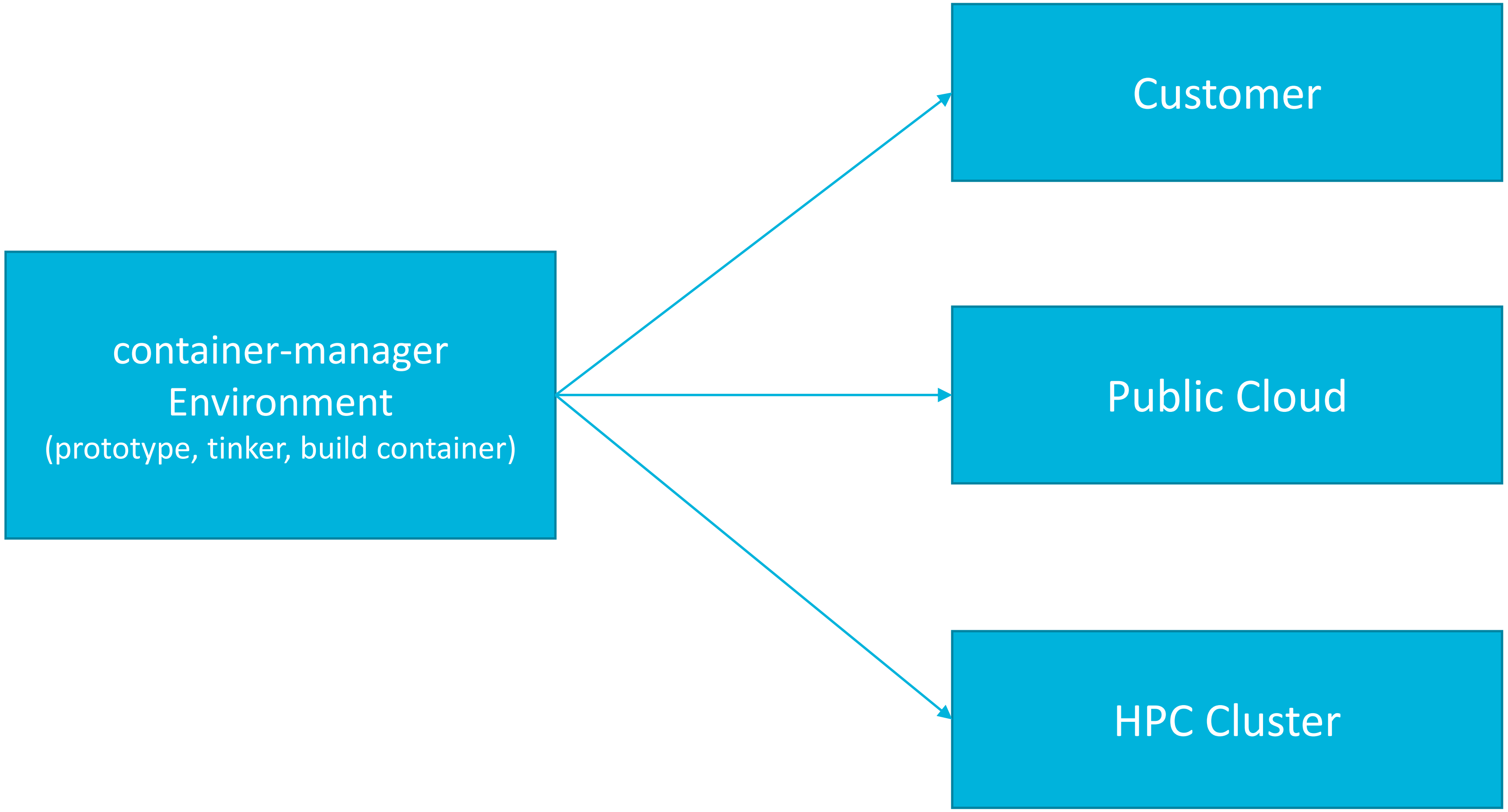
CPU% Disk%

Other Operations



```
➜ ~ ssh -q containers@kafka
> help

Commands:
  copy          Copy containers
  create        Create containers
  exit          exit the program
  forward       Port forward operations
  help          display help
  images        List avaliable images
  ls            List containers
  migrate       Migrate containers to this host
  project       Project operations
  rename        Rename containers
  restart       Restart containers
  rm            Remove containers
  share         Share a container with other users
  start         Start containers


>
```

User Pipeline



Code

- https://container-manager.gitlab.io/docs/
  - Code is on the same GitLab project
  - Opensource very soon (few LICENSE issues)

Future Work and Applications

- **Continue to streamline quick-access to GPUs for employees**
- **Innovation grant proposal to incorporate container-manager on the HPC systems**

MITRE

MITRE’s mission-driven teams are dedicated to solving problems for a safer world. Through our federally funded R&D centers and public-private partnerships, we work across government to tackle challenges to the safety, stability, and well-being of our nation.

Learn more www.mitre.org