HKG18-419: OpenHPC on Ansible
- Easier deployment with Ansible -

Renato Gloin and Takeharu Kato, HPC

Outline

- What OpenHPC is
- The Problem - OpenHPC has no Configuration Management System
- What Ansible is
- OpenHPC + Ansible = OpenHPC deployment easier
- Future work

OpenHPC

- OpenHPC is a collaborative, community effort that initiated from a desire to aggregate a number of common ingredients required to deploy and manage High Performance Computing (HPC) Linux clusters including provisioning tools, resource management, I/O clients, development tools, and a variety of scientific libraries.
  - http://www.openhpc.community/
- OpenHPC packages are build on the Open Build System
  - https://build.openhpc.community/
- Packages hosted on GitHub
  - https://github.com/openhpc/ohpc

OpenHPC Deployments Today

- A recipe with all rules described in the official documents
- LaTex snippets containing shell code

```
% begin_ohpc_run
\begin{lstlisting}[language=bash]
[sms](*\#*) (*\install*) lmod-defaults-gnu7-openmpi3-ohpc
\end{lstlisting}
% end_ohpc_run
```

- Are converted and merged into a (OS-dependent) bash script

```
# ----------------------------------------
# Install Performance Tools (Section 4.4)
# ----------------------------------------
yum -y install ohpc-gnu7-perf-tools
yum -y install lmod-defaults-gnu7-openmpi3-ohpc
```

- Plus a *input.local* file, with some cluster-specific configuration

OpenHPC Recipe - Problems

- The *input.local* file exports shell variables, and don't have enough information
  - Not all parts of the script are correctly conditionalized based on those options
  - Most people that use those scripts, heavily modify them
  - Others do in a completely different way
- The *recipe.sh* is **not** idempotent
  - It modifies the contents of root config files using Perl and regex
  - It echoes lines at the end of config files
  - It sets up networking and services that can't be set twice (moving files around)
- Extensibility is impossible without editing the files
  - It neither has ***everything*** (conditionalized), nor has external hooks to add local rules
  - We need to *at least* edit the *input.local* configuration, which needs to be kept and updated

OpenHPC Recipe - Attempted Solutions

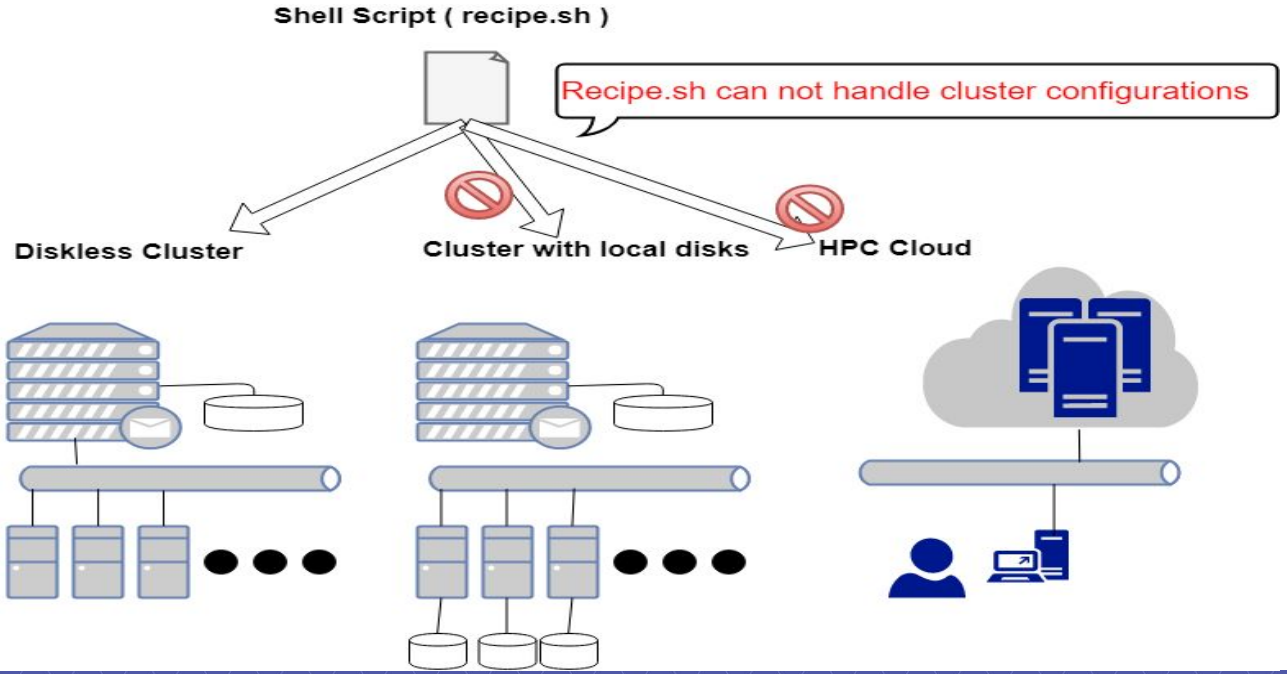
- We tried to change the scripts to conditionalize everything
  - Split into different scripts, one for each chapter
  - Heavily modified the *input.local* file
  - Automation (or people, manually), can run only selected scripts
- Problems
  - We created yet another layer (*bash_local*), which sets up *input.local*
  - Still using environment variables
  - We haven't tested every possible scenario, no idea if it would work elsewhere… :(
- Why reinvent the wheel?
  - After long discussion with the communities, folks were quite happy to properly automate it
  - Ansible wasn't the only solution people were using, but was by far the most popular
  - So we're having a go with Ansible…
  - With the high hopes of moving the upstream deployments and other community builds to it

```
Name
00_base_install.sh
01_master_pkgs.sh
02_master_network.sh
03_master_provision.sh
04_optional.sh
05_master_files.sh
06_master_bootstrap.sh
07_devtools.sh
08_system_start.sh
09_test_suite.sh
bash_local
generate-warewulf-packages.sh
input.local
recipe.sh
```

Biggest Obstacle in OpenHPC Deployment

- Lack of ``Configuration Management System'' (CMS)

  ➔A kind of CMS is needed to build various HPC clusters easy.



OpenHPC + Ansible = Easier Deployment

- What Ansible is

  A simple automation language which can describe the structure and configuration of IT infrastructure with Ansible playbook(YAML).
- The reasons why Ansible is the best solution for the OpenHPC deployment
  - The most promising CMS tool (Ansible development is led by Red Hat).
  - Ansible playbooks are idempotent.
  - Ansible can manage nodes, tasks according to the the structure of the cluster.

Ansible will be a main stream in deployment for servers

Improvements from recipe.sh

- Improvements from recipe.sh:
  - We can configure where packages are installed, according to node type.
  - We can replay the installation process at many times.
  - We can adopt same recipe for various HPC systems(Intel, AArch64, bere metal cluster, Cloud, Container)
- Time required for OpenHPC installation:

We took **a week** to install OpenHPC by hand before.

We can now install OpenHPC in **3 hours**.

Current Status

- We've ported recipe.sh to an Ansible playbook.
  Our playbook does not depend on HPC-specific tools (Warewulf, xCat)

The playbook could be applied to other LEG tasks with minor changes.

- What we are doing now:
  - Testing...We are testing on X64 machines, and we'll test it on AArch64 soon.
  - Rewrite for common use...Dividing the playbook into Ansible roles for each package. It is useful to use the playbook as ingredients for installation task.
  - Upstreaming ... We'll collaborate with OpenHPC community to make OpenHPC installation easy.

Our playbook to be published soon and be upstreamed.

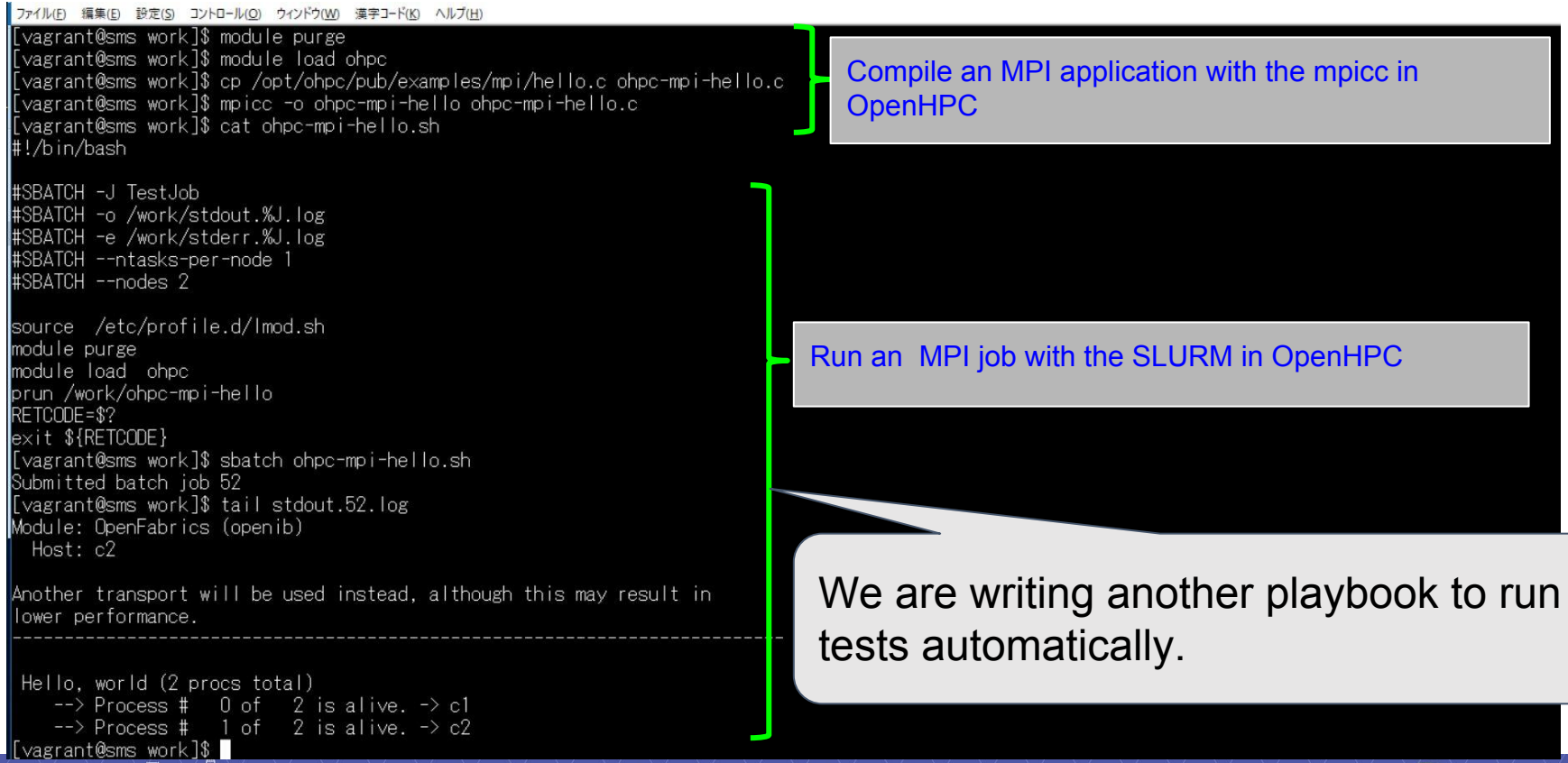
Screen shot (Install phase)

[Inventory file]

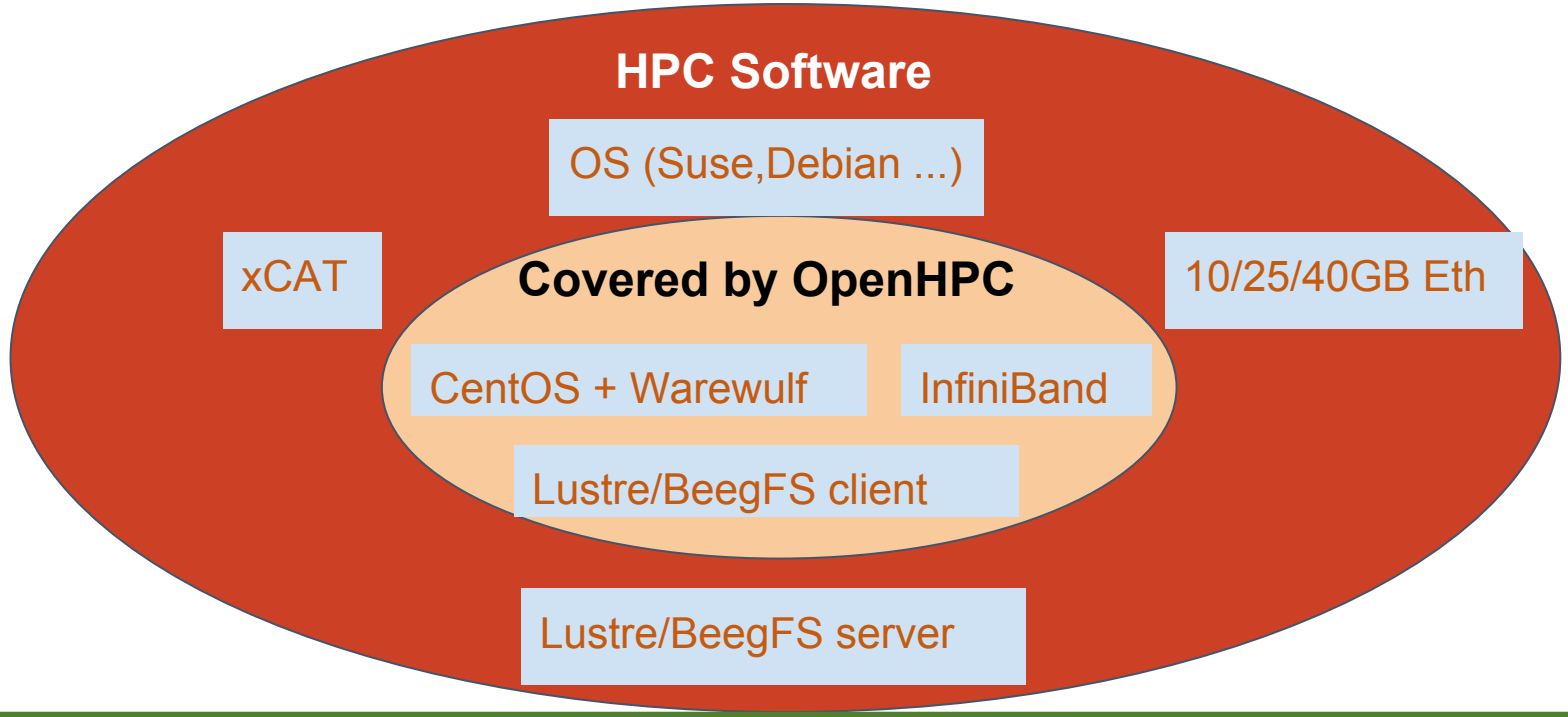
```
[sms]
192.168.33.11
[cnodes]
192.168.33.21
192.168.33.22
[ionodes]
192.168.33.11
[devnodes]
192.168.33.21
192.168.33.22
```

It can handle the types of nodes properly.
(e.g., Install slurm-server on the master node only)

Screen shot(Test phase)



Promote Use of Ansible in HPC



Ansible will helps to support various packages, devices, OS in HPC.

Future Work

- Ansible in OpenHPC … Improving maintainability
  - Considering the way to generate a playbook from the document (Are LaTeX snippets preferable?)
  - Taking advantage of the power of the Ansible community?
- Ansible in HPC … Making Ansible more popular in HPC
  - Sharing Ansible playbooks for HPC systems through ansible galaxy.
  - Introducing Ansible to OpenHPC community for upstreaming.
- Ansible in general … Easy to write site-specific configurations

  Some ideas:
  - Creating a tool to generate host-specific configuration files.
  - Using Kconfig (make menuconfig) to choose packages to be installed.
  - A kind of tool for describing network topology to generate network configurations (like OpenStack dashboard).

OpenHPC Ansible - Integration with Provision

- Before we can install OpenHPC on a cluster, we need to provision the master
  - This is done via a Jenkins job, using MrProvisioner's API to flash kernel/initrd/preseed
  - Once the node is up (for now, polling on machine), we fire Ansible up
  - SSH to the node, clone the repo, run the playbook
- OpenHPC provisions the compute nodes (via Warewulf/xCAT)
  - All machines can see MrP, so we need to avoid DHCP/TFTP clash
  - Essentially via MAC address binding, but this complicates the Jenkins job
  - We need the list of addresses beforehand
- Having one VLAN per cluster is less flexible
  - We restrict ourselves to fixed number of compute nodes per cluster
  - Changing involves networking configuration (not dynamic in MrP yet)
  - Forces master to provide NAT/DNS access for the compute nodes

Thank You

#HKG18

HKG18 keynotes and videos on: connect.linaro.org

For further information: www.linaro.org

OpenHPC Ansible - Introduction

- What is ansible?

  A simple automation language which can describe the structure and configuration of IT infrastructure with Ansible playbook(YAML).

- Characteristics of Ansible:
  - Idempotent … Ansible playbook describes the state of node instead of operations or procedures
  - Lightweight … Ansible does not need an installation agent on each node.
  - Configuration Management … It can handle groups of nodes(SMS, CN, etc)
  - Easy to extend … Well defined plugin interfaces for plugin modules.
  - Orchestration … It can also be applied to clouds, a container orchestration.

OpenHPC Ansible - Improvements from recipe.sh

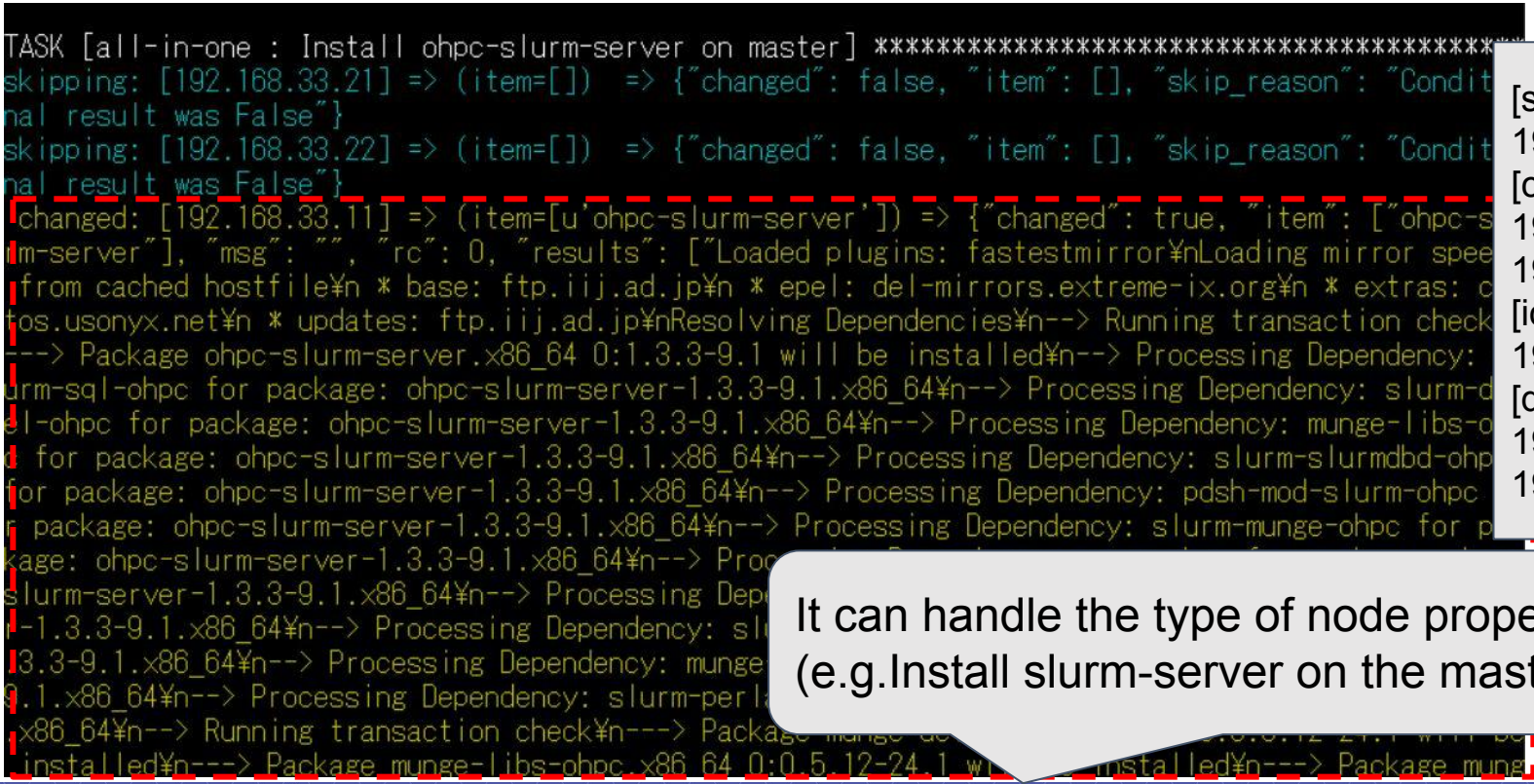
- Improvements from recipe.sh:
  - Installing OpenHPC according to the role of nodes with Ansible inventory:
    - SMS...Management tools, Slurm server
    - Computing node...Libraries, Slurm client, Lustre client
    - Development node … Compiler, Debugger
  - Sharing the install methods among different architectures(Intel, AArch64)
    - Ansible can handle conditional installation and configuration
      (e.g. Install psm packages(intel interconnect) into Intel machines only)
  - Supporting various HPC systems
    - Diskless environment (e.g. Warewulf)
    - Small cluster which has node local storages.
    - Cloud/Container (e.g OpenStack, AWS, docker) in the future

OpenHPC Ansible - Current Status

- We've ported recipe.sh to an ansible playbook.
  - It can install OpenHPC(both Warewulf and Non-Warewulf clusters on CentOS7)
  - It can handle the structure of the cluster(both diskless and non-diskless environment).
    - It can set computing node network up(e.g DHCP, hosts, and so on).
    - It can install and configure OpenHPC packages according to each node type (SMS, computing node, development node).
    - It can be used for both AArch64 and Intel machines.
- What we are doing now:
  - Testing our playbook on virtual machines(x86-64)
    - Of Course we'll test the playbook on AArch64 VM soon...
  - Writing the playbook for testing
  - Dividing the playbook into Ansible role for each package for the flexible installation and to share installation methods for HPC packages through Ansible Galaxy(*).

(*) A kind of community pages for Ansible(https://galaxy.ansible.com/)

OpenHPC Ansible - Screen shot(Install phase)



[Inventory file]

```
[sms]
192.168.33.11
[cnodes]
192.168.33.21
192.168.33.22
[ionodes]
192.168.33.11
[devnodes]
192.168.33.21
192.168.33.22
```

It can handle the type of node properly.
(e.g.Install slurm-server on the master node only)

OpenHPC Ansible - Screen shot(Test phase)

```
ファイル(F)  編集(E)  設定(S)  コントロール(O)  ウィンドウ(W)  漢字コード(K)  ヘルプ(H)
[vagrant@sms work]$ module purge
[vagrant@sms work]$ module load ohpc
[vagrant@sms work]$ cp /opt/ohpc/pub/examples/mpi/hello.c ohpc-mpi-hello.c
[vagrant@sms work]$ mpicc -o ohpc-mpi-hello ohpc-mpi-hello.c
[vagrant@sms work]$ cat ohpc-mpi-hello.sh
#!/bin/bash

#SBATCH -J TestJob
#SBATCH -o /work/stdout.%J.log
#SBATCH -e /work/stderr.%J.log
#SBATCH --ntasks-per-node 1
#SBATCH --nodes 2

source  /etc/profile.d/lmod.sh
module purge
module load  ohpc
prun /work/ohpc-mpi-hello
RETCODE=$?
exit ${RETCODE}
[vagrant@sms work]$ sbatch ohpc-mpi-hello.sh
Submitted batch job 52
[vagrant@sms work]$ tail stdout.52.log
Module: OpenFabrics (openib)
  Host: c2

Another transport will be used instead, although this may result in
lower performance.
--------------------------------------------------------------------------


 Hello, world (2 procs total)
    --> Process #   0 of   2 is alive. -> c1
    --> Process #   1 of   2 is alive. -> c2
[vagrant@sms work]$
```

Compile MPI application with the mpicc in OpenHPC

Run a MPI job with the SLURM in OpenHPC

We are writing another playbook to run test automatically.

OpenHPC Ansible - Flexible installation with Ansible

- Improvement of the installation flexibility:

  We are dividing configuration variables in recipe.sh into three parts(in progress):

  - Cluster wide configurations(node name of SMS, FS Server)
  - Computing node specific configurations(MAC/IP addr, BMC)
  - Package specific configurations(CHROOT for Warewulf)

[The basic structure of Ansible playbook]

```
ohpcOrchestration/
  +---- group_vars/
        +-- all.yml                cluster wide configurations
        +-- group1,group2 … node group(e.g. computing nodes) specific configurations
  +---- host_vars/
        +--- host1,host2     … host specific configurations
  +---- roles/                     package specific tasks, templates, config files, config variables
         +--- package-name/
               +--- tasks/main.yml … YAML file to describe installation method of package-name
               +--- vars/main.yml … package-name specific configuration variables
               +--- templates/*.j2 … template files for configuration files
```

OpenHPC Ansible - Scenarios

- We’re going to test on real AArch64 machines
  - CentOS + Warewulf
  - InfiniBand
  - Lustre/BeegFS client
- Need to improve
  - xCAT support
  - Lustre/BeegFS server support
  - 10/25/40GB Eth support
  - Increasing supported Operating System
    (OpenSuse, Debian, and so on. Plugin development may be needed for this)
  - Container based distribution(Ansible can create container images to distribute)

Ansible can be used to install OpenHPC into various HPC systems including bare metal cluster, HPC clouds, Containers(e.g. Docker) in the future.

OpenHPC Ansible - Unsolved Problems

- Improving maintenance ability
  - Considering the way to generate a playbook from the document
    (Are LaTeX snippets preferable?)
  - Taking advantage of the power of the ansible community?
- Easy to write site specific configurations

  Following things might be useful (Idea only for now):
  - Creating tool to generate host specific configuration files from a CSV file
  - Using Kconfig(make menuconfig) to choose packages to be installed.
  - Some kind of tools for describing network topology easily(like OpenStack dashboard) to generate network configurations.
- Making Ansible more popular in HPC world
  - Sharing Ansible playbooks for HPC systems through ansible galaxy.
  - Introducing Ansible to OpenHPC community

OpenHPC Ansible - Integration with Provision

- Before we can install OpenHPC on a cluster, we need to provision the master
  - This is done via a Jenkins job, using MrProvisioner's API to flash kernel/initrd/preseed
  - Once the node is up (for now, polling on machine), we fire Ansible up
  - SSH to the node, clone the repo, run the playbook
- OpenHPC provisions the compute nodes (via Warewulf/xCAT)
  - All machines can see MrP, so we need to avoid DHCP/TFTP clash
  - Essentially via MAC address binding, but this complicates the Jenkins job
  - We need the list of addresses beforehand
- Having one VLAN per cluster is less flexible
  - We restrict ourselves to fixed number of compute nodes per cluster
  - Changing involves networking configuration (not dynamic in MrP yet)
  - Forces master to provide NAT/DNS access for the compute nodes