DECEMBER 16TH, 2025

MPI APPLIANCES FOR HPC ON CHAMELEON

ROHAN BABBAR
2025 Summer of Reproducibility / Google Summer of Code Contributor

KEN RAFFENETTI
Research Software Engineer
MCS Division
Argonne National Laboratory

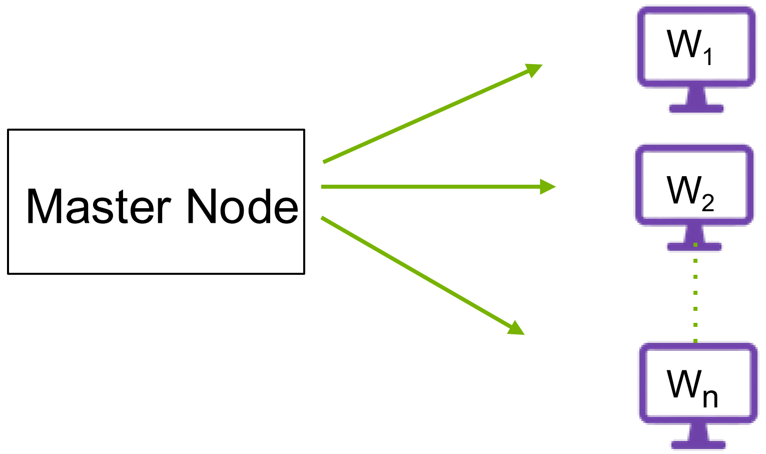
Overview

- **Introduction**
- Objective
- Implementation Strategy
- Deliverables
- Conclusion

Introduction

The Message Passing Interface (MPI) is the backbone of high-performance computing (HPC), enabling efficient scaling across thousands of processing cores. However, reproducing MPI-based experiments remains challenging due to dependencies on specific library versions, network configurations, and multi-node setups.

To address this, we aim to provide standardized MPI environments on the Chameleon testbed. Users can utilize the appliances we maintain to deploy configurable MPI cluster on any number of nodes. The appliance setup scripts ensure that all nodes have the same MPI libraries, software, and network settings, making experiments easier reproduce (and scale).

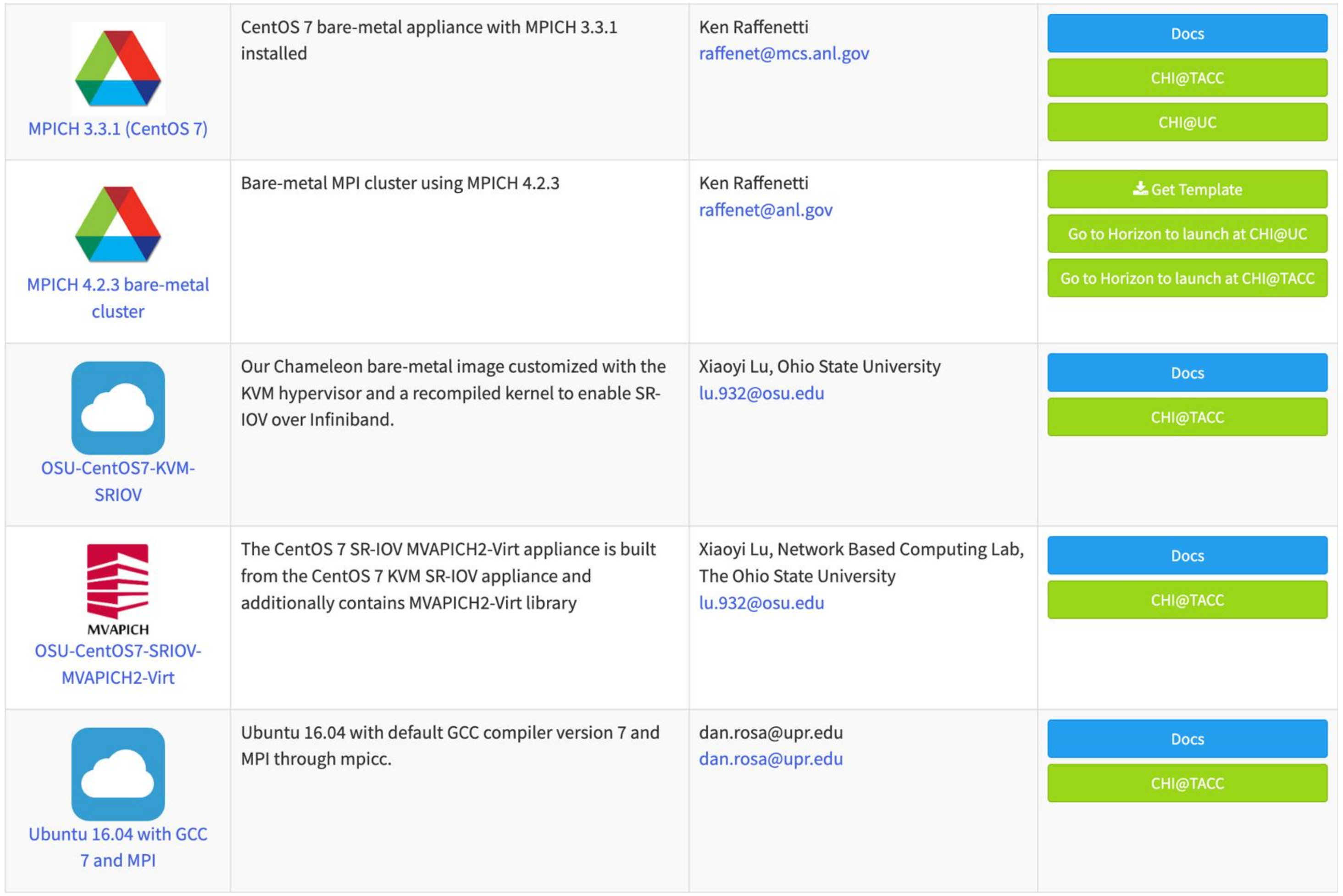


What is MPI?

- MPI: Message Passing Interface
  - The MPI Forum organized in 1992 with broad participation by:
    - Vendors
    - Portability library developers
    - Users
  - Incorporates the best ideas in a “standard” way
    - Each API function takes fixed arguments
    - Each API function has fixed semantics
      - Standardizes what the MPI implementation provides and what the application can and cannot expect
      - Each system can implement it differently as long as the semantics match
- MPI is not…
  - a language
  - a compiler (mpicc/mpicxx/mpifort are just convenient helpers)
  - a specific implementation or product
- Commonly used *implementations* of MPI. Users often not even aware of differences!
  - MPICH
  - Open MPI
  - MVAPICH, Intel MPI, Cray MPICH, …

MPI on Chameleon

- Prior to this project: some number of appliances existed to use MPI on Chameleon
  - Somewhat fragmented
  - Limited to older versions of OSes and MPI
  - Complex appliances not functional
  - Users unlikely to find the exact software needed by their project



MPICH 3.3.1 (CentOS 7)	CentOS 7 bare-metal appliance with MPICH 3.3.1 installed	Ken Raffenetti raffenet@mcs.anl.gov	Docs / CHI@TACC / CHI@UC
MPICH 4.2.3 bare-metal cluster	Bare-metal MPI cluster using MPICH 4.2.3	Ken Raffenetti raffenet@anl.gov	Get Template / Go to Horizon to launch at CHI@UC / Go to Horizon to launch at CHI@TACC
OSU-CentOS7-KVM-SRIOV	Our Chameleon bare-metal image customized with the KVM hypervisor and a recompiled kernel to enable SR-IOV over Infiniband.	Xiaoyi Lu, Ohio State University lu.932@osu.edu	Docs / CHI@TACC
MVAPICH OSU-CentOS7-SRIOV-MVAPICH2-Virt	The CentOS 7 SR-IOV MVAPICH2-Virt appliance is built from the CentOS 7 KVM SR-IOV appliance and additionally contains MVAPICH2-Virt library	Xiaoyi Lu, Network Based Computing Lab, The Ohio State University lu.932@osu.edu	Docs / CHI@TACC
Ubuntu 16.04 with GCC 7 and MPI	Ubuntu 16.04 with default GCC compiler version 7 and MPI through mpicc.	dan.rosa@upr.edu dan.rosa@upr.edu	Docs / CHI@TACC

Examples of existing MPI appliances

Supercomputing Conference Reproducibility Initiative

- Why is Reproducibility Important?
  - Easier for other researchers to compare, adopt, and extend published work
- SC Conference Technical Papers
  - Required to submit an Artifact Description (AD) Appendix. This includes (1) the computational artifacts (software, datasets, environment configuration, etc.) and (2) instructions and documentation describing the artifacts and how to use them.
  - Optional Artifact Evaluation (AE) Appendix is step-by-step instructions to reproduce the results of the paper. Each AE Appendix is reviewed by the reproducibility committee using an **allotment of resources on Chameleon Cloud**.
- A clear need for easily configurable HPC appliances on Chameleon ClouD

2025 Summer of Reproducibility

- Collaboration between the Open Source Research Experience (OSRE) organized by the UC Santa Cruz and the REPETO project
  - Funding for students to contribute to open source projects
  - Make computational research efforts reproducible
  - Open Source Program Office serves as official mentor organization for Google Summer of Code
- MPI Appliance for HPC Research on Chameleon (Rohan Babbar)
  - Open Source Program Office serves as official mentor organization for Google Summer of Code
  - Data Analyst at Stryker Corporation.
  - Likes to contribute to open-source (primarily in Python). Maintain some of the packages like PyLops-MPI, PreliZ, ArviZ-plots.
  - Google Summer of Code contributor in 2023 and 2024 with NUMFOCUS.
  - Interested in:
    - ❑ Scientific Computing
    - ❑ MPI Programming
    - ❑ Python Packaging

REPETO
Building a Network for Practical Reproducibility in Experimental Computer Science

Open Source Program Office
UC Santa Cruz
University of California, Santa Cruz

Overview

▷ Introduction

▷ **Objective**

▷ Implementation Strategy

▷ Deliverables

▷ Conclusion

Objectives

The aim of this project is to create an MPI appliance that is configurable and easily deployable. The key objectives of this project are:

- Pre-built MPI Images: Create ready-to-use images with MPI and all dependencies installed.
- Automated Cluster Configuration: Develop Ansible playbooks to configure cluster, including host setup, SSH key distribution, and MPI configuration across nodes.
- Cluster Orchestration: Develop orchestration template to provision resources and invoke Ansible playbooks for automated cluster setup.
- MPI Cluster Requirements (Simple):
  - Login node (aka main or master node) to accept SSH connections
  - NFS share for storing application binaries and data
  - MPI installation for building and running programs
  - Hostfile for mpiexec

Overview

- Introduction
- Objective
- **Implementation Strategy**
- Deliverables
- Conclusion

OpenStack Image Creation

- Build on Ubuntu 22.04 as base environment.

- Spack + Lmod integration:
    - Spack handles installation of software packages.
    - Lmod (Lua Modules) provides a user-friendly way to load/unload software environments dynamically.
    - Together, they allow users to easily switch between MPI versions, libraries, and other dependencies like GPU toolkits.

- MPICH and Open MPI pre-installed. Load/unload depending on need.

- Three image variants for various HPC workloads: CPU-only, NVIDIA GPU (CUDA 12.8), and AMD GPU (ROCm 6.4.2).

Why Spack?

- Spack is a package manager designed for HPC (High-Performance Computing) environments.
- Spack allows multiple versions or variants of the same package to coexist. For example, both MPICH 4.1 and 3.4 can be installed. Users load whichever version they need.
- Moreso than distro or general-purpose package managers, Spack is MPI-aware. Spack supports usage of several interchangeable MPI implementations, both open and closed source.
- Package variants allow users to specify compilers, flags, or features such as GPU support. Example: Installing MPICH with GCC and CUDA support for NVIDIA nodes, and an LLVM build for CPU-only nodes.
- Modules (Lmod) integration enables easy and flexible environment setup. Many HPC centers employ a Spack+Lmod setup, including Argonne LCRC and ALCF clusters like Aurora
- Spack offers *many* packages — development tools (spack install git), container runtimes (spack install apptainer podman), and HPC apps like GROMACS (spack install gromacs +mpi ^mpich)

Cluster Configuration using Ansible

Ansible is an open-source tool for Infrastructure as Code, enabling automating tasks, configuration management, and software provisioning. YAML playbooks, agentless architecture, idempotent in nature, wide ecosystem.

Cluster Configuration using Ansible

Assigned specific roles to different nodes in the cluster and combined them into a single playbook to configure the entire cluster automatically.

- Host Setup → Configure /etc/hosts entries for all nodes.
- Mount NFS shares on each node.
- Generate SSH keypair on master node → Add master's public key to workers' authorized_keys.
- Scan worker node keys → update known_hosts on master.
- (Optional) Manage software:
  - Install new compilers with Spack
  - Add new Spack packages
  - Update modules to include them
- Also creates a hostfile at /etc/mpi/hostfile for use with mpiexec

Orchestration

With the OpenStack MPI+Spack image deployed and Ansible scripts ready for cluster configuration, we now orchestrate the cluster deployment:

- Spin up instances from the MPI+Spack image to ensure all nodes share the same environment.
- Generate dynamic inventory of master and worker nodes for Ansible.
- Execute Ansible playbook to configure hosts, NFS, SSH, and software stack.
- Ready-to-use cluster – deployable with minimal manual effort.

Python-CHI(Jupyter) + Ansible

Python library designed to facilitate interaction with the Chameleon testbed. Often used within environments like Jupyter notebooks.

- Create leases, launch instances, and set up shared storage using *python-chi* commands.
- Automatically generate *inventory.ini* for Ansible based on launched instances.
- Run Ansible playbook programmatically using *ansible_runner*.
- Outcome: fully configured, ready-to-use HPC cluster; SSH into master to run examples.

Heat Orchestration Template

Heat Orchestration Template(HOT) is a YAML based configuration file. Its purpose is to define/create a stack to automate the deployment and configuration of OpenStack cloud resources.

Heat Orchestration Template

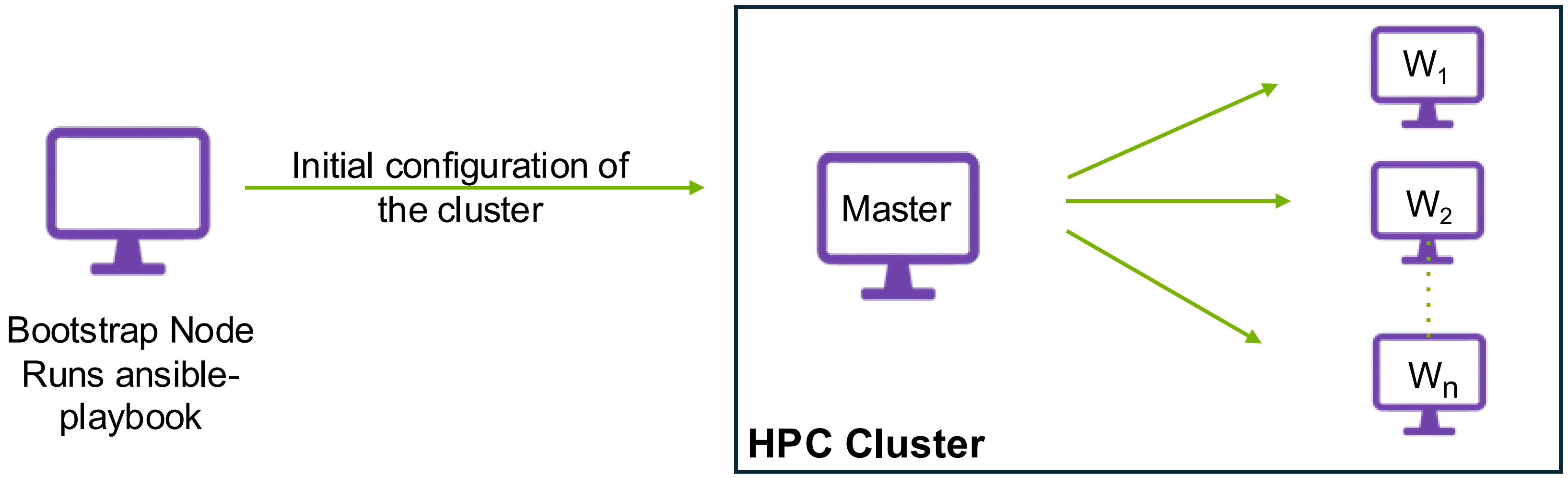
Heat Orchestration Template(HOT) is a YAML based configuration file. Its purpose is to define/create a stack to automate the deployment and configuration of OpenStack cloud resources.

Issues In Heat Deployment

- OS::Nova::Keypair (new version): In the latest OpenStack version, the stack fails to launch if the public_key parameter is not provided for the keypair, as auto-generation is no longer supported.

- OS::Heat::SoftwareConfig: Deployment scripts fail, hang, or time out, preventing proper configuration of nodes and causing unreliable deployments.

Heat Orchestration Template


Initial configuration of the cluster
Bootstrap Node Runs ansible-playbook
Master
W_1
W_2
W_n
HPC Cluster

- Launch Instances: Provision nodes via the HOT template in OpenStack.
- Bootstrap Node: Set up a bootstrap node and install Git and Ansible. Run ansible-playbook from the bootstrap node to configure master and worker nodes, including SSH, host communication, and MPI setup.
- Outcome: fully configured, ready-to-use HPC cluster; SSH into master to run examples.

Cluster Launched According to Spec

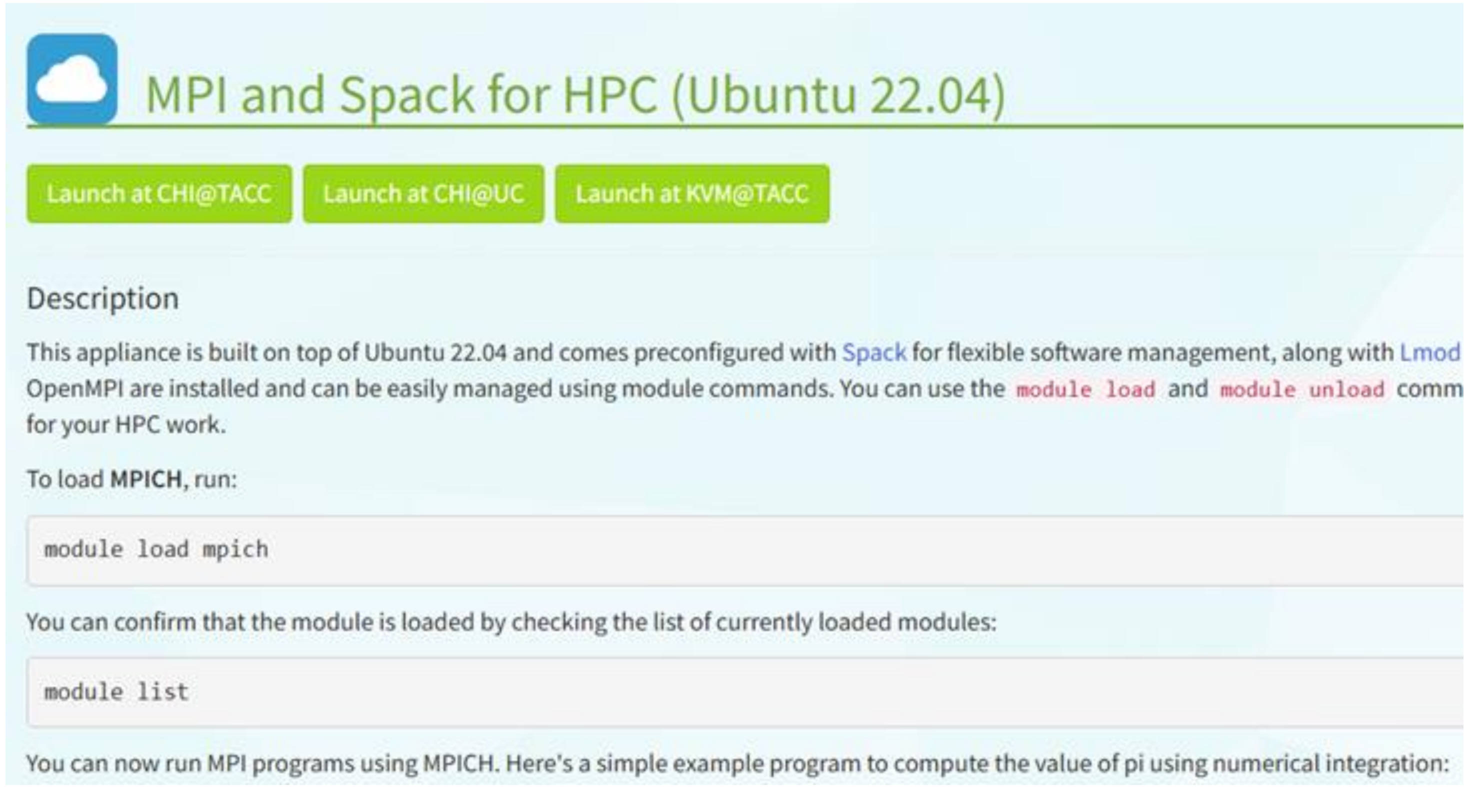
- All nodes start from the same pre-built images, ensuring identical OS, MPI libraries, and other dependencies.
- Spack guarantees exact software versions and configuration, avoiding differences across nodes or over time.
- Ansible reliably configures master–worker roles and the full cluster environment without manual errors.
- Heat templates / Python-CHI + Ansible allow fully scripted, repeatable cluster provisioning.
- Anyone can recreate the same fully configured MPI cluster anytime, enabling reliable experiments.

Overview

- Introduction
- Objective
- Implementation Strategy
- **Deliverables**
- Conclusion

MPI and Spack for HPC (Ubuntu 22.04)

https://chameleoncloud.org/appliances/127/



MPI and Spack for HPC (Ubuntu 22.04)

Description

This appliance is built on top of Ubuntu 22.04 and comes preconfigured with Spack for flexible software management, along with Lmod
OpenMPI are installed and can be easily managed using module commands. You can use the `module load` and `module unload` comm
for your HPC work.

To load **MPICH**, run:

```
module load mpich
```

You can confirm that the module is loaded by checking the list of currently loaded modules:

```
module list
```

You can now run MPI programs using MPICH. Here's a simple example program to compute the value of pi using numerical integration:

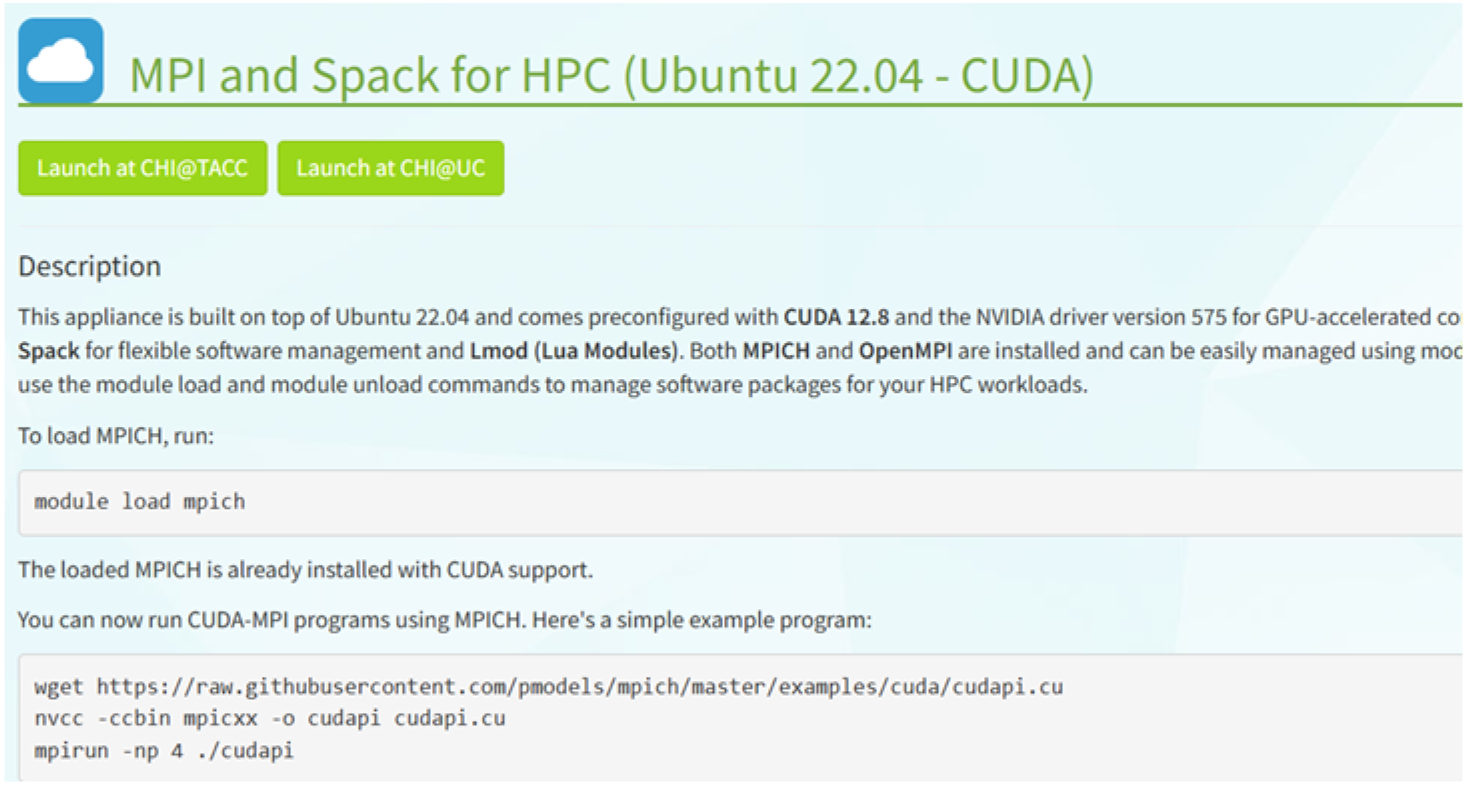
Base OS: Ubuntu 22.04

Preconfigured: Spack + Lmod for reproducible software environments

MPI Support: MPICH and OpenMPI installed via Spack

MPI and Spack for HPC (Ubuntu 22.04) - CUDA

https://chameleoncloud.org/appliances/130/



MPI and Spack for HPC (Ubuntu 22.04 - CUDA)

Description

This appliance is built on top of Ubuntu 22.04 and comes preconfigured with **CUDA 12.8** and the NVIDIA driver version 575 for GPU-accelerated co
Spack for flexible software management and **Lmod (Lua Modules)**. Both **MPICH** and **OpenMPI** are installed and can be easily managed using moc
use the module load and module unload commands to manage software packages for your HPC workloads.

To load MPICH, run:

```
module load mpich
```

The loaded MPICH is already installed with CUDA support.

You can now run CUDA-MPI programs using MPICH. Here's a simple example program:

```
wget https://raw.githubusercontent.com/pmodels/mpich/master/examples/cuda/cudapi.cu
nvcc -ccbin mpicxx -o cudapi cudapi.cu
mpirun -np 4 ./cudapi
```

Base OS: Ubuntu 22.04

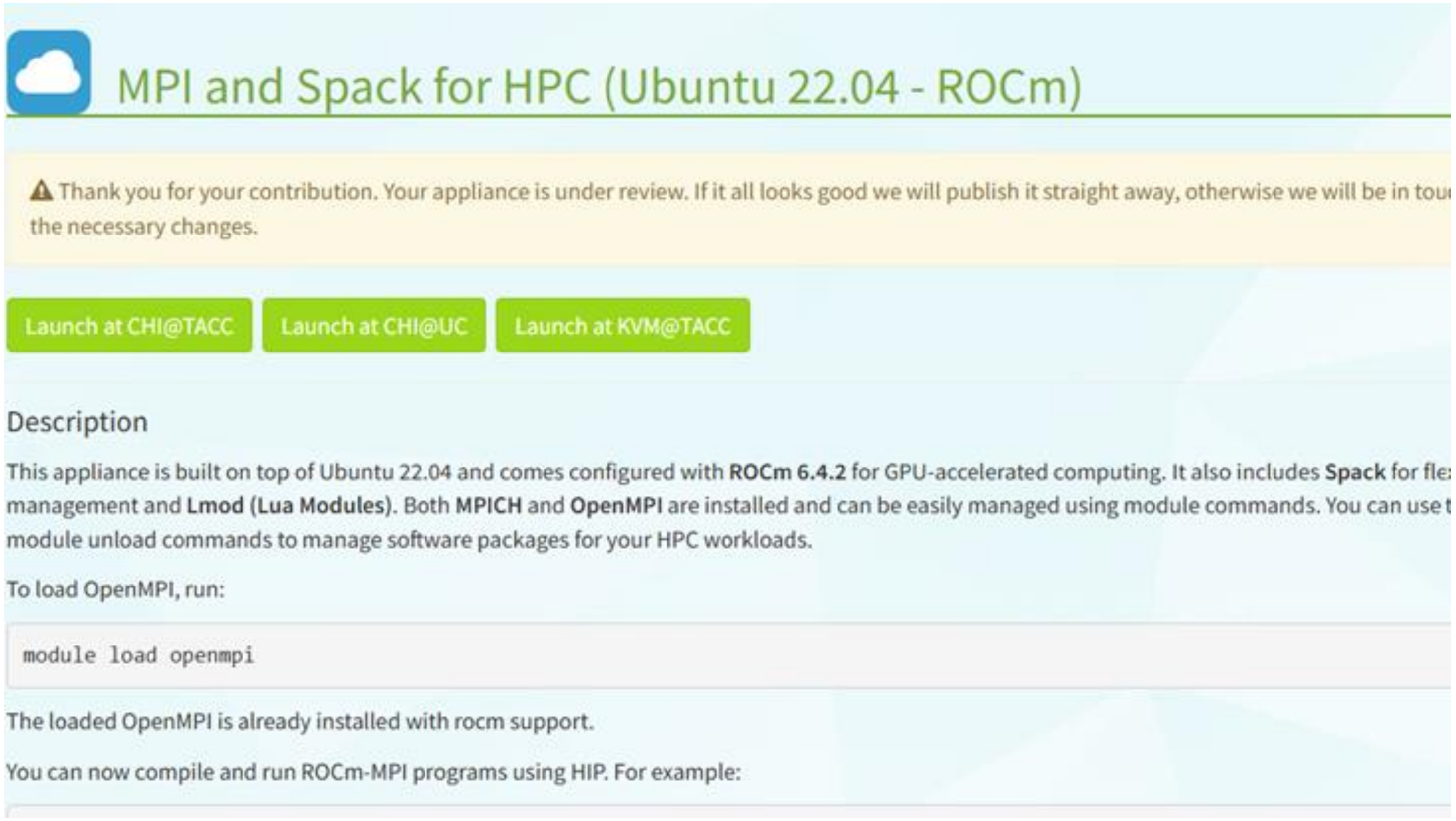
Preconfigured: Spack + Lmod for reproducible software environments

CUDA-12.8 Support

MPI Support: MPICH+cuda and OpenMPI+cuda installed via Spack

MPI and Spack for HPC (Ubuntu 22.04) - ROCm

https://chameleoncloud.org/appliances/131/


MPI and Spack for HPC (Ubuntu 22.04 - ROCm)

Thank you for your contribution. Your appliance is under review. If it all looks good we will publish it straight away, otherwise we will be in tou
the necessary changes.

Description

This appliance is built on top of Ubuntu 22.04 and comes configured with **ROCm 6.4.2** for GPU-accelerated computing. It also includes **Spack** for fle
management and **Lmod (Lua Modules)**. Both **MPICH** and **OpenMPI** are installed and can be easily managed using module commands. You can use t
module unload commands to manage software packages for your HPC workloads.

To load OpenMPI, run:

```
module load openmpi
```

The loaded OpenMPI is already installed with rocm support.

You can now compile and run ROCm-MPI programs using HIP. For example:

Base OS: Ubuntu 22.04

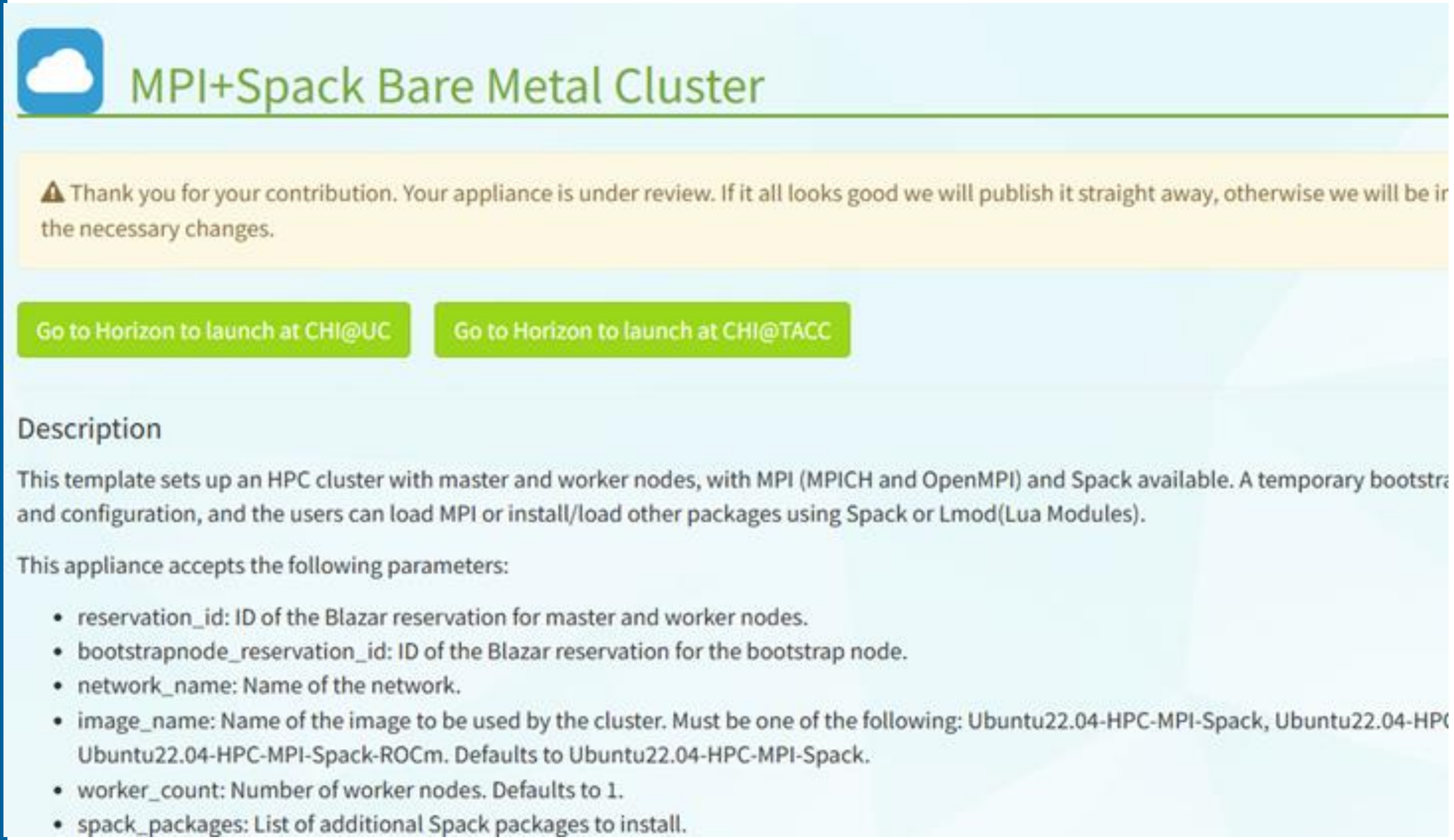
Preconfigured: Spack + Lmod for reproducible software environments

ROCm-6.4.2 base Support

MPI Support: MPICH+rocm and OpenMPI+rocm installed via Spack

Complex Appliance

https://chameleoncloud.org/appliances/132/



MPI+Spack Bare Metal Cluster

Thank you for your contribution. Your appliance is under review. If it all looks good we will publish it straight away, otherwise we will be ir
the necessary changes.

Go to Horizon to launch at CHI@UC
Go to Horizon to launch at CHI@TACC

Description

This template sets up an HPC cluster with master and worker nodes, with MPI (MPICH and OpenMPI) and Spack available. A temporary bootstra
and configuration, and the users can load MPI or install/load other packages using Spack or Lmod(Lua Modules).

This appliance accepts the following parameters:

- reservation_id: ID of the Blazar reservation for master and worker nodes.
- bootstrapnode_reservation_id: ID of the Blazar reservation for the bootstrap node.
- network_name: Name of the network.
- image_name: Name of the image to be used by the cluster. Must be one of the following: Ubuntu22.04-HPC-MPI-Spack, Ubuntu22.04-HP(
Ubuntu22.04-HPC-MPI-Spack-ROCm. Defaults to Ubuntu22.04-HPC-MPI-Spack.
- worker_count: Number of worker nodes. Defaults to 1.
- spack_packages: List of additional Spack packages to install.

Heat Orchestration Template for creating a bare metal cluster.

Trovi Artifact

https://chameleoncloud.org/experiment/share/7424a8dc-0688-4383-9d67-1e40ff37de17

Chameleon Trovi is a sharing portal that allows you to share digital research and education artifacts

- The artifact contains Ansible code and tasks for configuring the cluster.
- Includes example Jupyter notebooks showing how to use Ansible and Python-CHI to create a fully working cluster.
- The artifact provides an example of creating an OpenStack stack using a Heat Orchestration Template.

GitHub Repository

https://github.com/pmodels/cc-hpc-appliances

Canonical source for Trovi artifacts

- Report issues
- Propose changes via Pull Request.
- Potential improvements: extend to work with different base images besides Ubuntu 22.04, add regular CI testing, run performance and/or scaling experiments, integrate a scheduler like Slurm or PBS, create and manage additional cluster user accounts.

Overview

- Introduction
- Objective
- Implementation Strategy
- Deliverables
- **Conclusion**

Conclusion

In conclusion, this work demonstrates an approach to building and configuring MPI clusters on the Chameleon testbed. By using standardized images, Ansible automation, and Orchestration Templates, we ensure that every node is consistently set up, reducing manual effort and errors. The artifact, published on Trovi, makes the entire process transparent, reusable, and easy to implement, enabling users/researchers to reliably recreate and extend the cluster environment for their own experiments.

Thank You