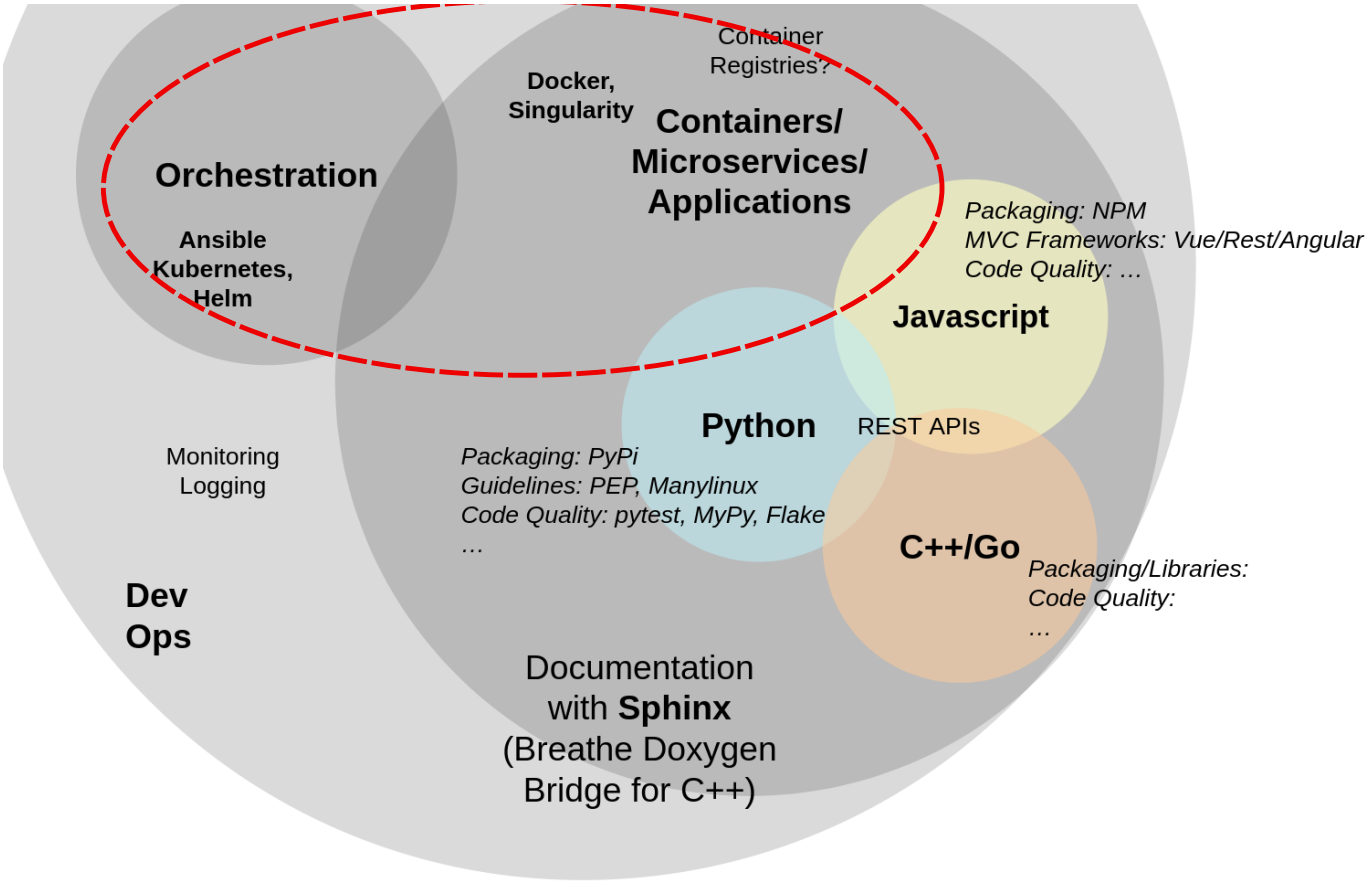
Workflow Orchestration for Material Science

Dr. Naweiluo Zhou, Dr. Giorgio Scorzelli, Dr. Valerio Pascucci, Dr. Michella Taufer
Paula Olaya Garcia, Dr. Jakob Luettgau
Contact: naweiluo.zhou@utk.edu

Overview

Workflow Orchestration on Cloud and HPC

Conclusion Remarks and Next steps


Container
Registries?
Docker,
Singularity
Containers/
Microservices/
Applications
Orchestration
Ansible
Kubernetes,
Helm
Packaging: NPM
MVC Frameworks: Vue/Rest/Angular
Code Quality: …
Javascript
Python
REST APIs
Monitoring
Logging
Packaging: PyPi
Guidelines: PEP, Manylinux
Code Quality: pytest, MyPy, Flake
…
C++/Go
Packaging/Libraries:
Code Quality:
…
Dev
Ops
Documentation
with Sphinx
(Breathe Doxygen
Bridge for C++)

Outline

Overview of Workflow Orchestration on Cloud and HPC

- Workflow containerisation
  - enable reproducibity
  - provide environment
  - portable
- Provision and orchestration of resources: compute, storage, network, etc.
- High Performance Storage

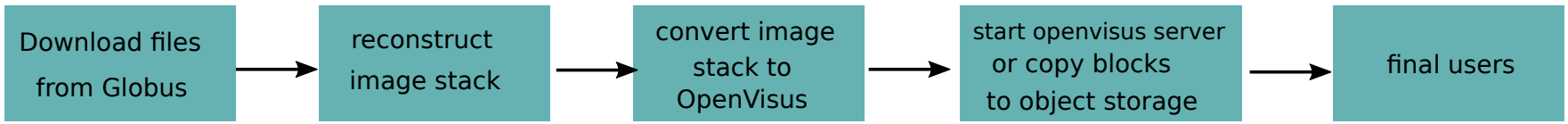


Figure: simplified workflow for material science

Resource Provision and Orchestration

Cluster deployment automation

- Provide easy access clusters.
- Ready to use

Smart scheduling

- Automatically identify the workload
- Decide the type of node to schedule: e.g. computation intensive, storage demanding, etc.?

Cluster Deployment Automation —Case study

Deployment Approches

- user-friendly interface: Jupyter Notebook
- Ansible: a software orchestration tool
- Python scripts
- bash scripts

Frameworks

- Kubernetes (k8s)— a widely-used container orchestrator
- Container engines, i.e Docker and/or Singularity

Automatic Cluster Deployment on Demand

—Chameleon Use case

Chameleon cluster: a large-scale, reconfigurable experimental platform hosted by the University of Chicago and TACC: bare-metal and VM nodes with root privileges.

Automation tools [1]

- Ansible Playbooks (yaml scripts)
- bash scripts

Cluster architecture

- one Ansible host
- A k8s cluster
  - one k8s login node (also the Ansible host)
  - one master node
  - one or multiple worker nodes

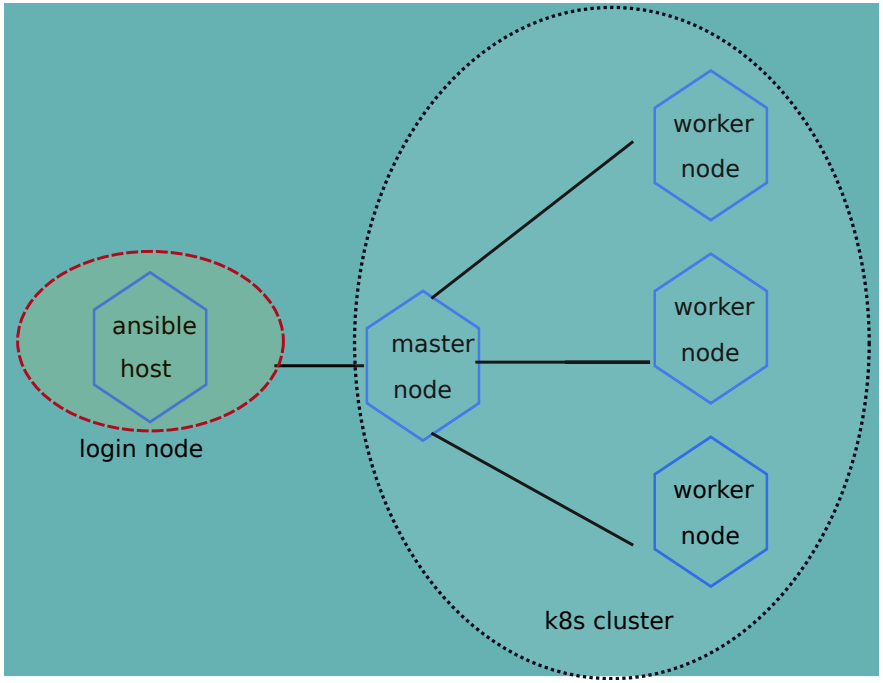
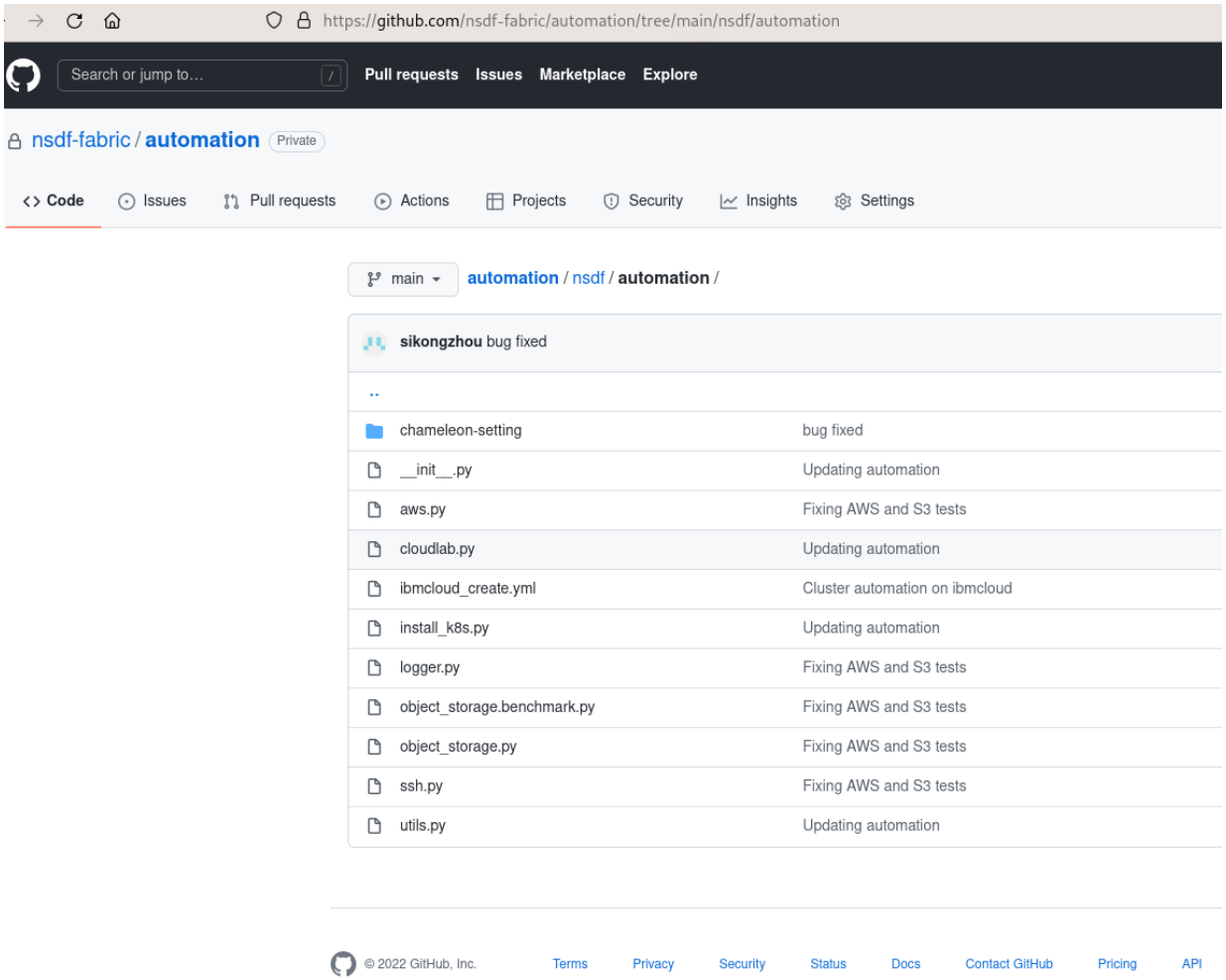


Figure: Our cluster arch on Chameleon

1. https://github.com/nsdf-fabric/automation/tree/main/nsdf/automation

Automatic Cluster Deployment on Demand



https://github.com/nsdf-fabric/automation/tree/main/nsdf/automation

High Performance Storage

Material science workflow requires:

1. Large data size
2. Fast data access

Storage Benchmarking

- Explore the state-of-the-art High-performance Storage Tech:
  - file storage
  - block storage
  - object storage
- Identify the suitable frameworks
- Identify the optimization technlogies.
  - Many threads/processes
  - Non-blocking
  - Hybrid
- benchmaking to analyse the storage throughput

Storage Benchmarking —A case study on Chameleon

```
=================================
                          Local-Path   vs              Longhorn    :            Change
IOPS (Read/Write)
        Random:          78,029 / 67,871   vs     20,538 / 13,723    :   -73.68% / -79.78%
    Sequential:          49,593 / 86,024   vs     25,332 / 23,434    :   -48.92% / -72.76%
  CPU Idleness:                      96%   vs                89%    :               -7%

Bandwidth in KiB/sec (Read/Write)
        Random:        344,901 / 402,383   vs    364,769 / 270,326    :    5.76% / -32.82%
    Sequential:        442,456 / 381,376   vs    419,740 / 280,534    :   -5.13% / -26.44%
  CPU Idleness:                      95%   vs                91%    :               -4%

Latency in ns (Read/Write)
        Random:        123,293 / 31,932   vs    411,993 / 336,327    :  234.16% / 953.26%
    Sequential:         36,896 / 32,565   vs    317,705 / 332,241    :  761.08% / 920.24%
  CPU Idleness:                      94%   vs                90%    :               -4%
```

Figure: Comparison of two types of storage: Local storage VS Longhorn on k8s evaluated with Kbench

- IOPS: IO operations per second. Higher is better
- Bandwidth: Throughput. Higher is better.
- Latency: The total time each request spent in the IO path. Lower is better.

Outline

Workflow Orchestration on Cloud and HPC

Conclusion Remarks and Next steps

Concludsion and Next Steps

Summary

1. Workflow containerisation
2. Resource provision and orchestration
3. High performance storage
4. two case studies (preliminary results):
   - cluster deployment automation/orchestration on Chameleon
   - storage benchmarking

Next steps

- identify the bottleneck of the workflow
- workflow containerisation
- workflow automation

Feedbacks...

1. Processing time of your given amount of data
2. Which part of the workflow procedure is the bottleneck

Acknowledegement: This work is funded by NSF OAC #2138811