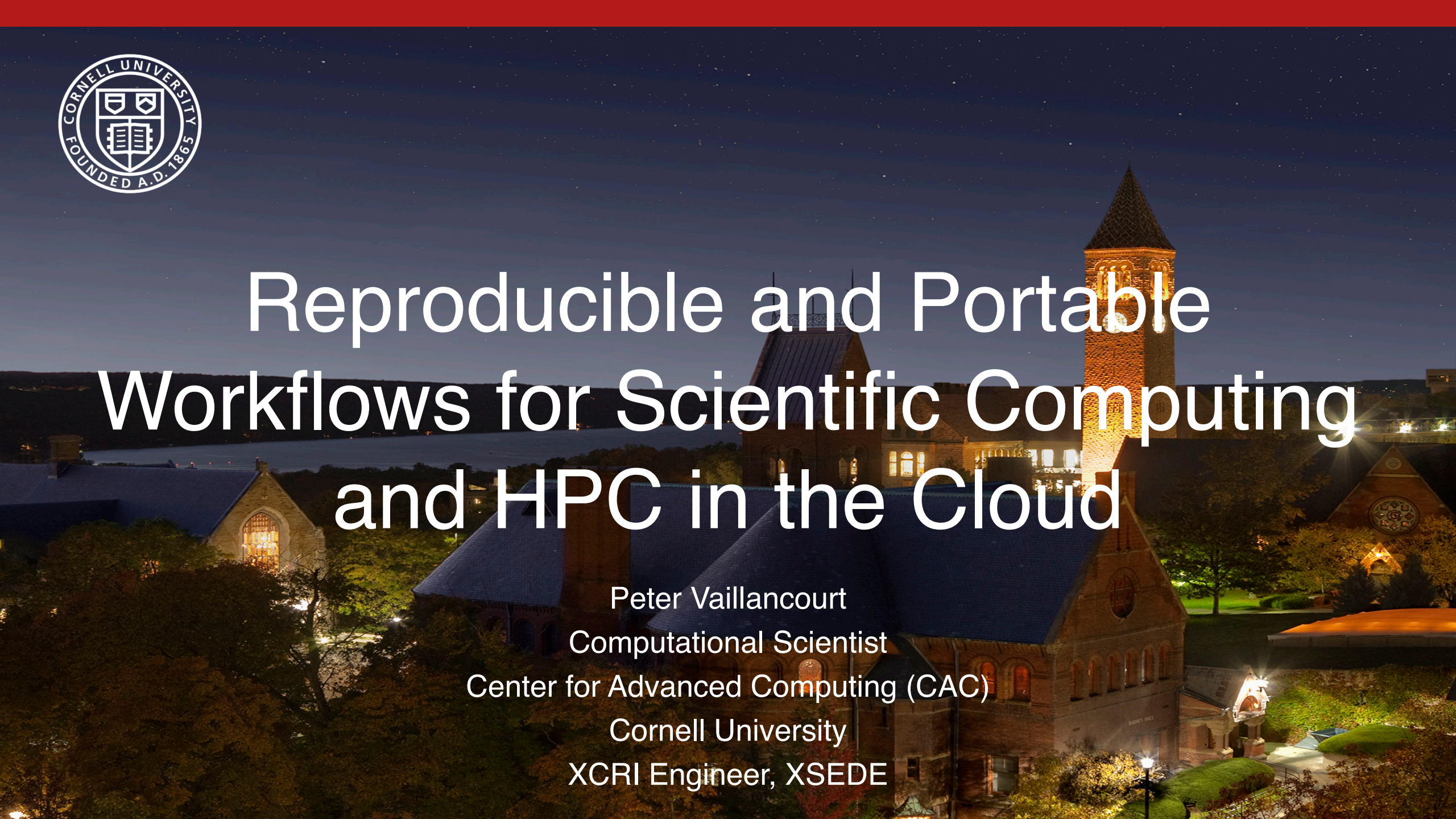


Reproducible and Portable Workflows for Scientific Computing and HPC in the Cloud

Peter Vaillancourt
Computational Scientist
Center for Advanced Computing (CAC)
Cornell University
XCRI Engineer, XSEDE

Aristotle Science Team

Collaborators

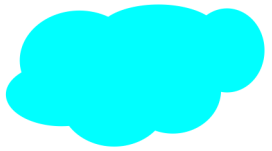
- Bennett Wineholt, CAC
- Brandon Barker, CAC
- Adam Brazier, CAC
- Rich Knepper, CAC
- Rich Wolski, UCSB

Students

- Plato Deliyannis, CAC REU student
- Jackie Zheng, CAC REU student
- Akshay Suresh, graduate student in Cornell Astronomy

Outline

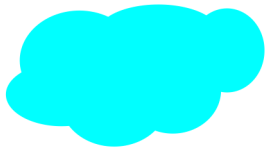
- Motivation
- Aristotle Cloud Federation
  - Water Resource Management
  - Weather Modeling
  - Radio Transient Detection
- Implementation
  - Containerization & Automation
  - Clouds
- Outcomes
  - Reproducible Containers
  - Multi-Cloud Deployment Automation
- Future work



Motivation

Why Deploy in the Cloud?

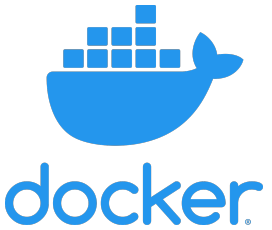
- Fast turn-around
- Portability
- Developer Productivity



Motivation

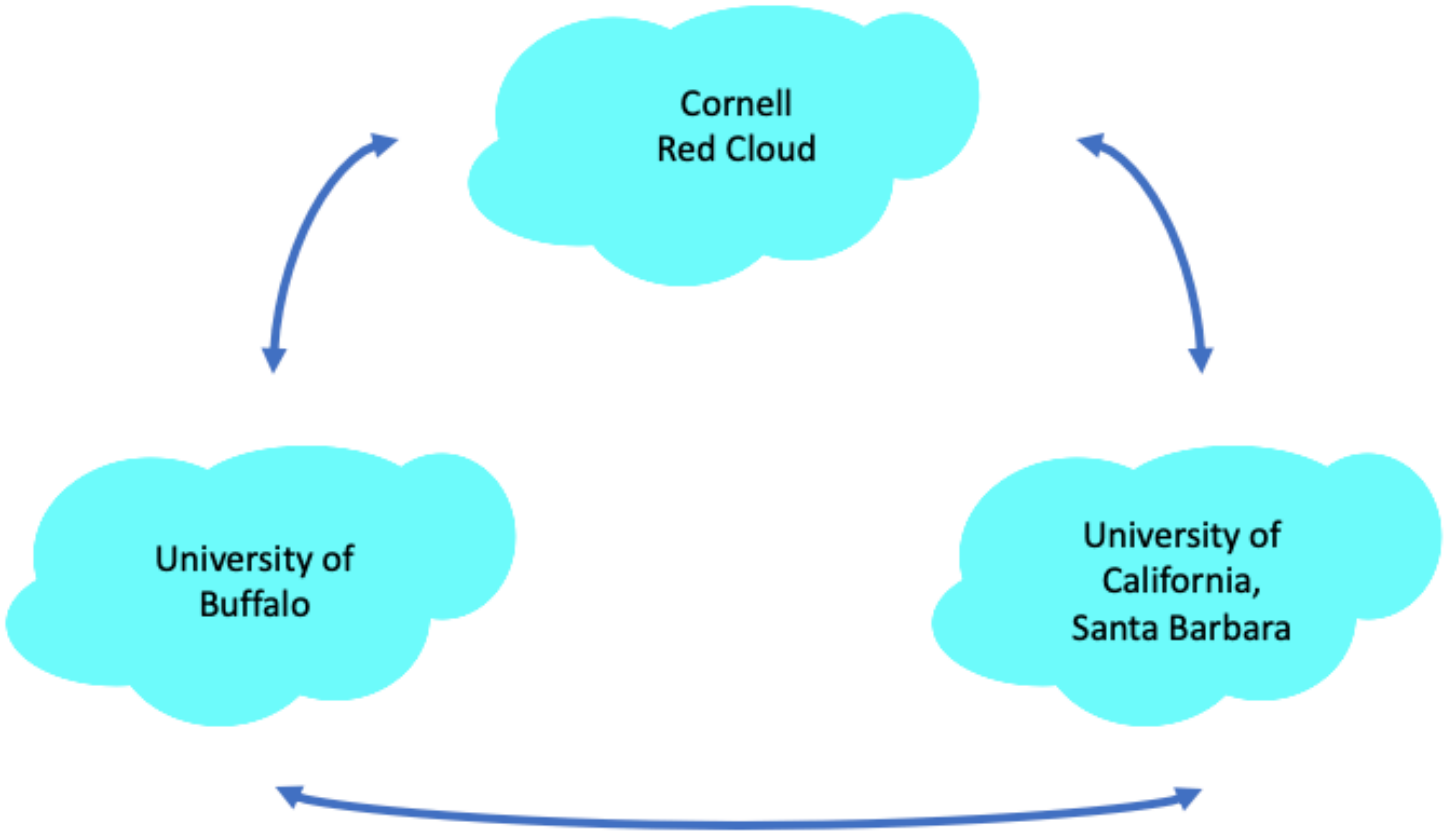
Why Use Containers?

- Reproducibility
- Portability
- Ease of Use



Overview

ARISTOTLE
CLOUD FEDERATION

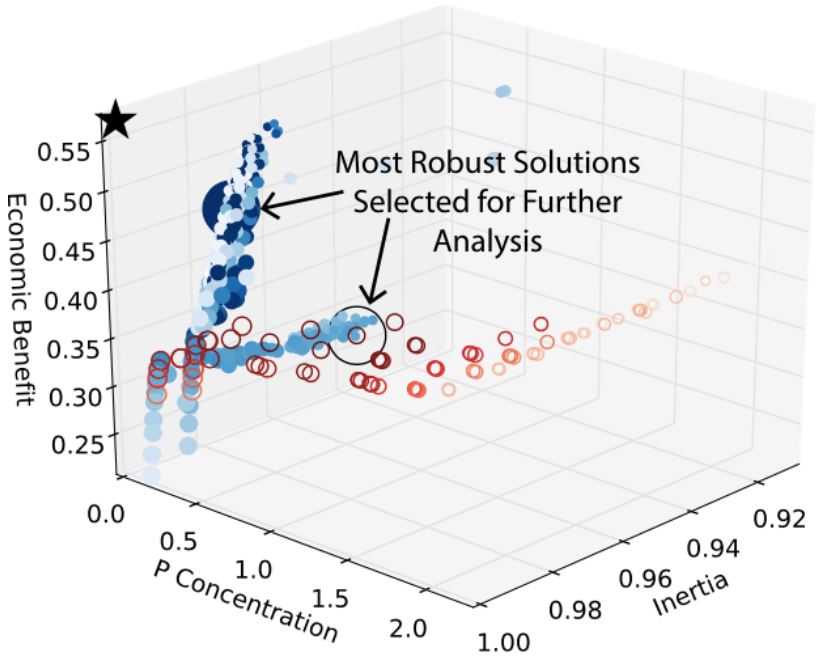


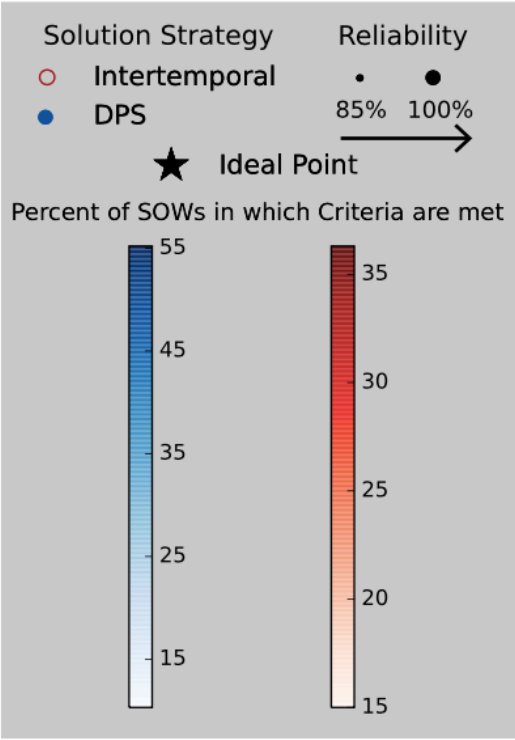
Aristotle Cloud Federation

Water Resource Management

- Pat Reed, Julianne Quinn, et. al., Environmental Science, Cornell

- Lake_Problem_DPS
  - federatedcloud/Lake_Problem_DPS
  - MORDM Framework
  - Multi-node
  - Some MPI communication

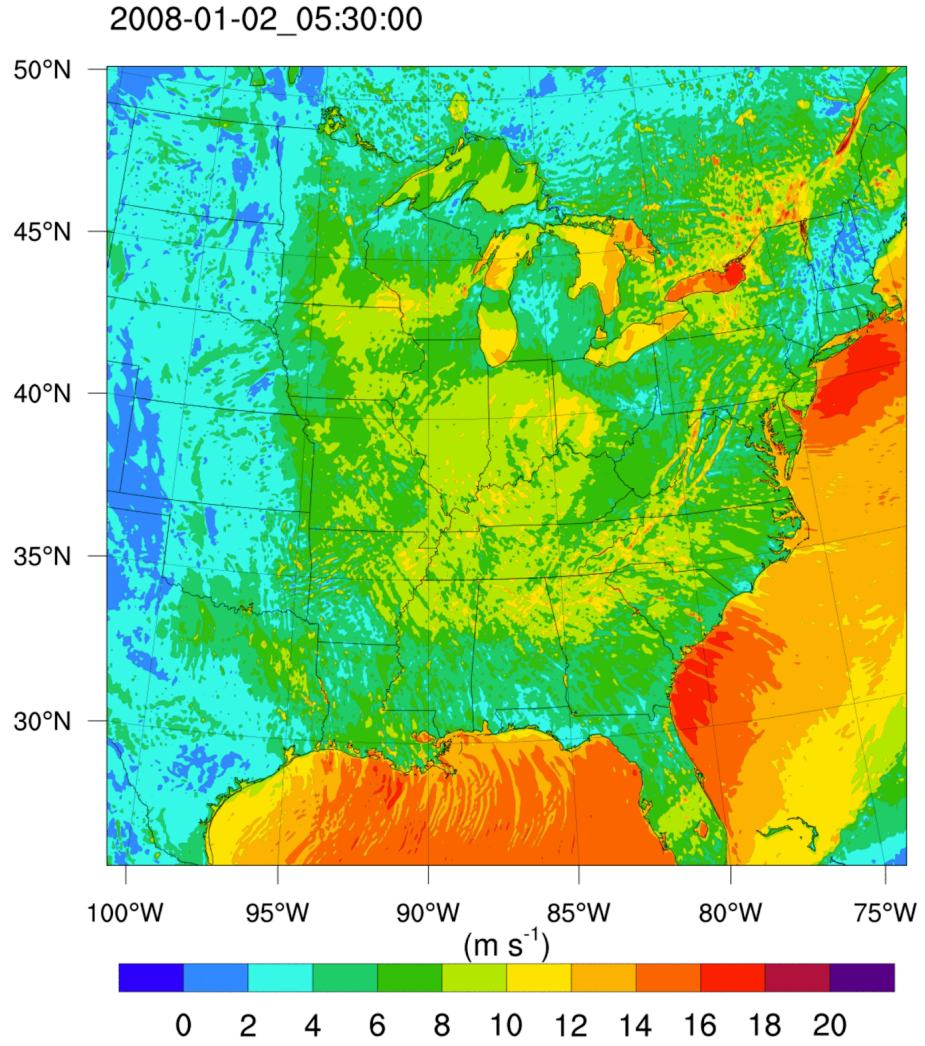




Quinn et al. 2017 10.1016/j.envsoft.2017.02.017

Aristotle Cloud Federation

Weather Research and Forecasting (WRF)

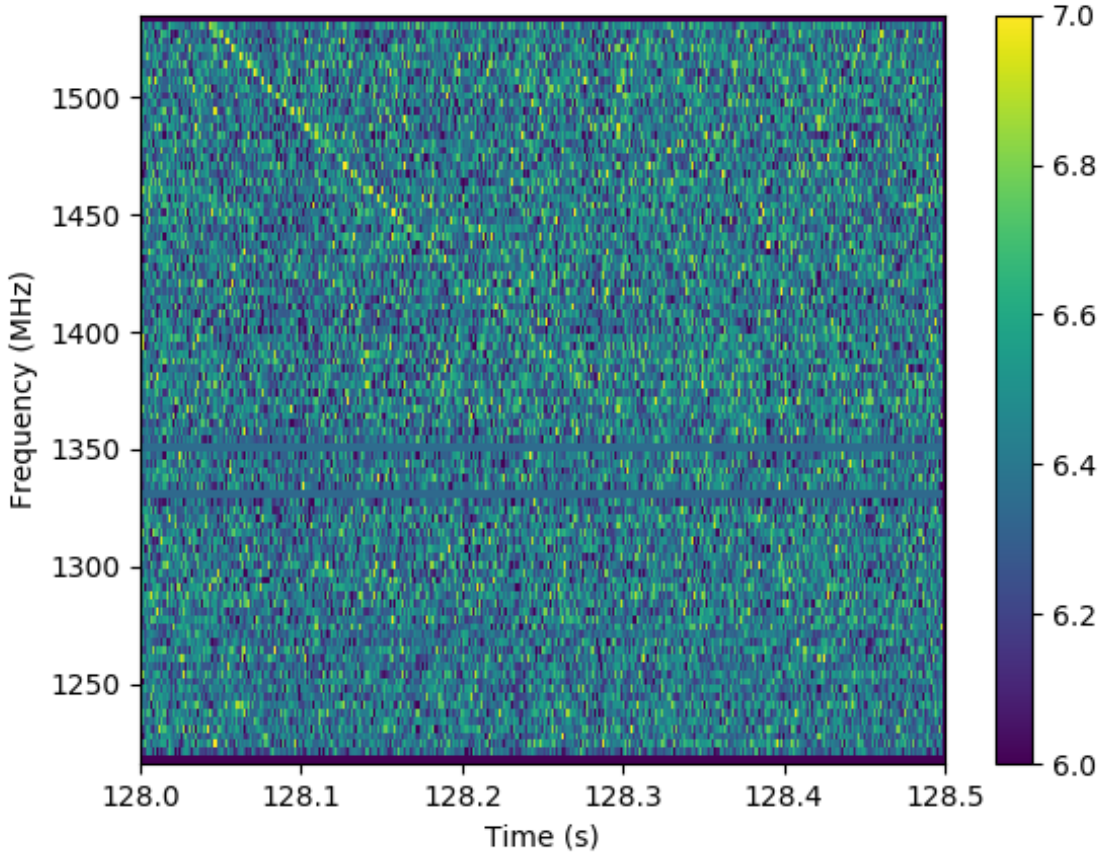


- Sara Pryor and Tristan Shephard et. al., Atmospheric Sciences, Cornell
- NCAR WRF
  - Multi-node
  - High MPI communication
  - Many workflow complexities

Aristotle Cloud Federation

Radio Transient Detection

- Jim Cordes and Shami Chatterjee et. al., Astronomy, Cornell
- FRB_pipeline
  - federatedcloud/FRB_pipeline
  - No MPI
  - High data throughput
- FOF algorithm implementation



Lebofsky et al. 2019 10.1088/1538-3873/ab3e82

Implementation

Containerization & Automation

Nix

docker

HashiCorp
Terraform

+

ANSIBLE

federatedcloud/ansible-terraform

Implementation

Containerization & Automation

- Docker container
- Data stored accessibly
- Compute VMs deployed with Terraform
- SSH keys configured with Ansible
- Containers deployed to VMs with Ansible
- Output staged on remote storage
- Cleanup with Terraform

Nix

docker

HashiCorp Terraform + ANSIBLE

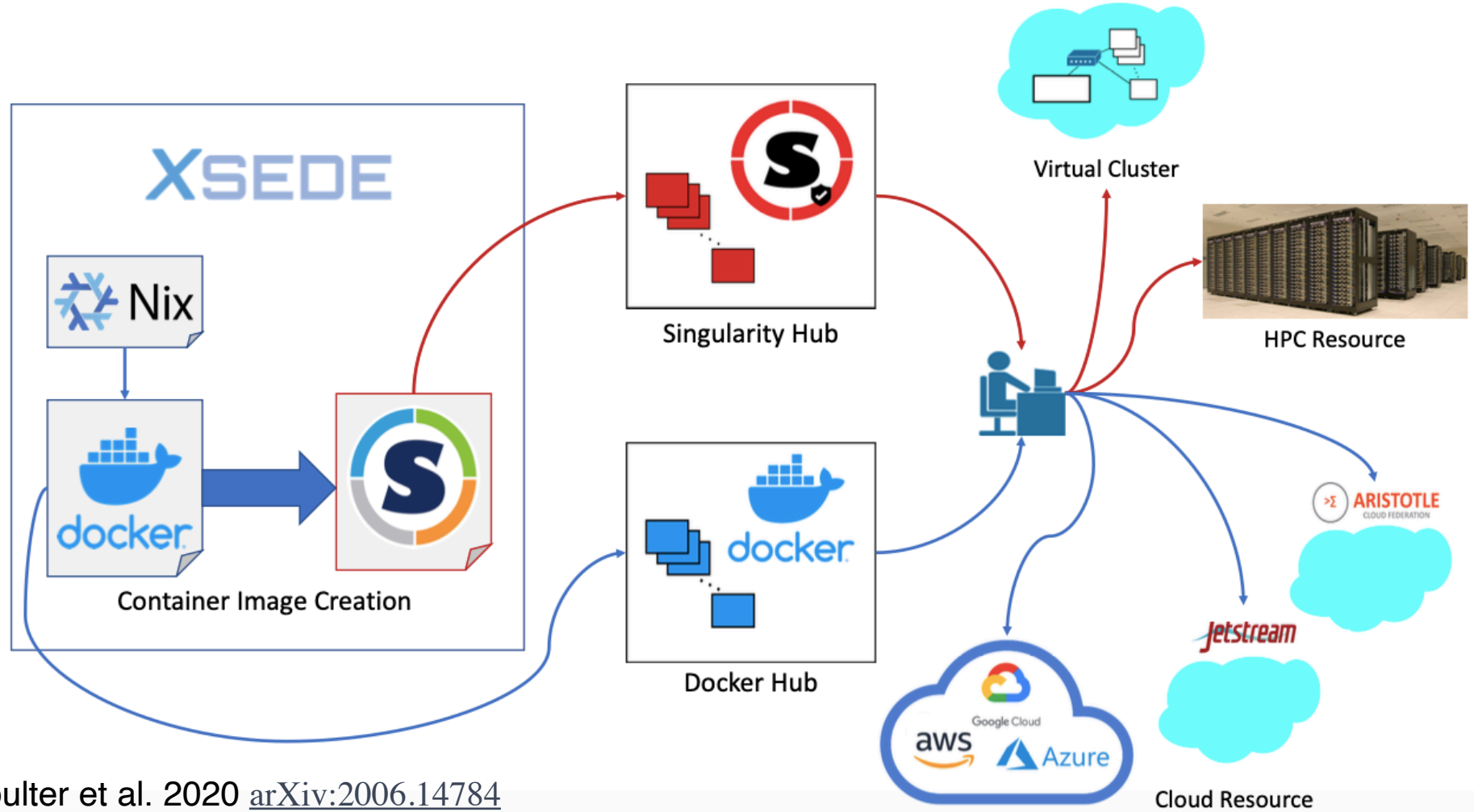
Implementation

Clouds

- Private cloud: Aristotle
  - OpenStack
  - Standard networks and VMS
- Public cloud: AWS
  - Also tested on GCP
  - Sensible defaults chosen
  - Scripts are customizable

Practical Outcomes

Reproducible Containers

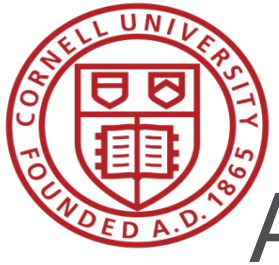


Vaillancourt and Coulter et al. 2020 arXiv:2006.14784

Practical Outcomes

Multi-Cloud Deployment Automation

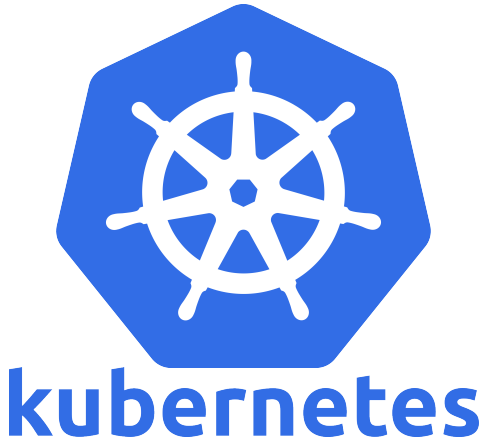
- Deploy HPC applications in the cloud
- Portable and customizable on multiple clouds, adds reproducibility
- Supports iterative development
- Eases infrastructure implementation
- Software tool choice matters – Terraform and Ansible are rapid
- Conevient to run from Ansible



Future Work

Automation of Kubernetes Cluster Deployments

+



Thank you for attending!

Questions?

pzv2@cornell.edu

Peter Vaillancourt
Computational Scientist
Center for Advanced Computing (CAC)
Cornell University
XCRI Engineer, XSEDE