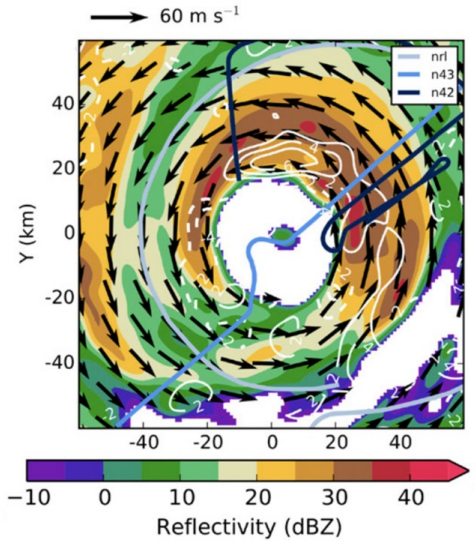
Increasing the portability and reproducibility of a scientific application using containers and Spack

Mentors: Jian Sun, Brian Vanderwende, John Dennis

Joe Ammatelli
University of Washington

July 28, 2022

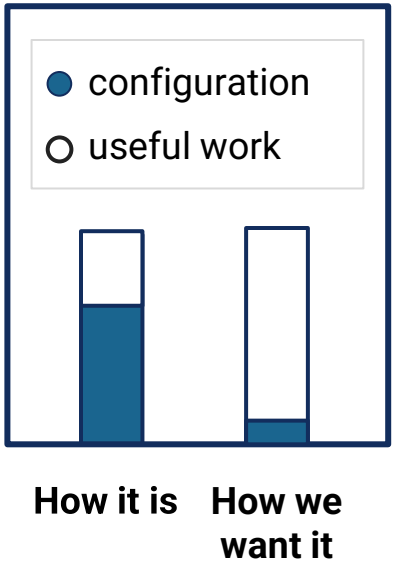
Samurai - an application with build challenges



SAMURAI

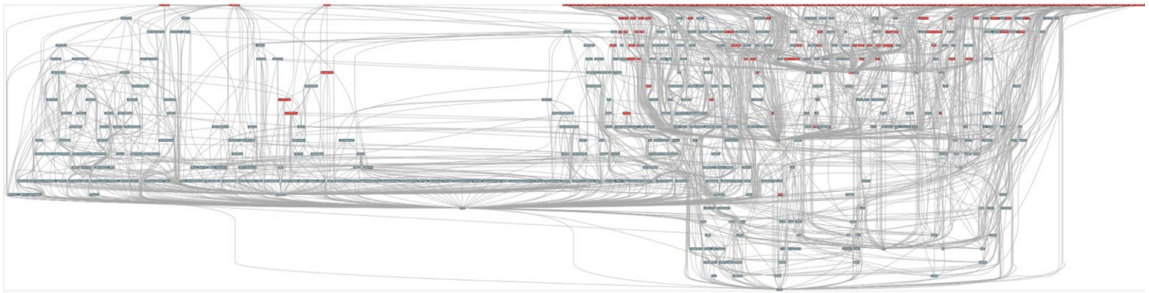
Spline Analysis at Mesoscale Utilizing Radar and Aircraft Instrumentation

1. **Sensitive to package dependency versions and build tool versions**
1. **Build scripts make assumptions about location and configuration of installed packages**
1. **Relies on fragile dependency with non-standard build structure**
1. **Build fails on some systems**
1. **Difficult to debug build errors**



Problem Statement

Build complexity: some software stacks have 10s or 100s of dependencies

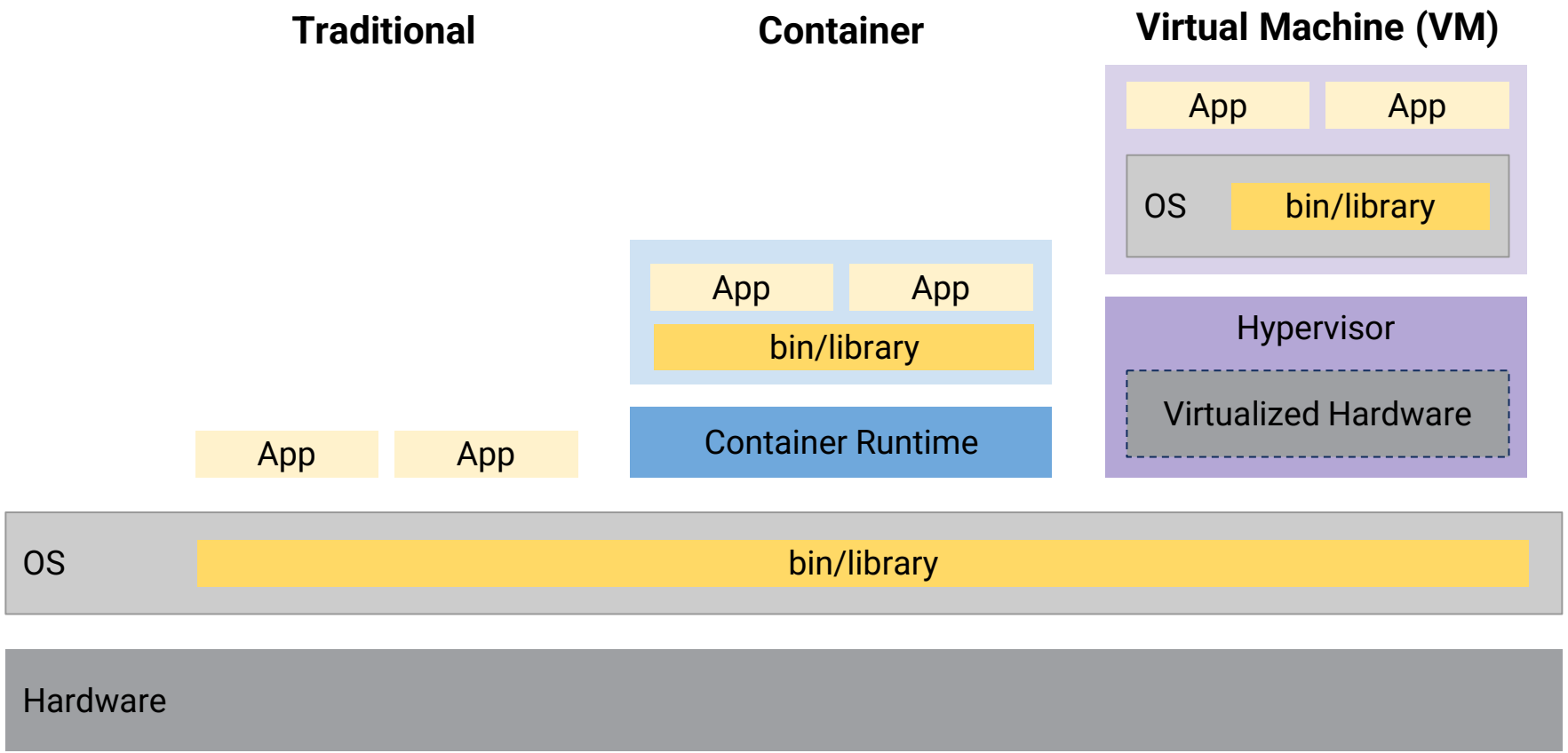


Todd Gamblin, Gregory Becker, Massimiliano Culpo, Michael Kuhn, and Harmen Stoppels. Managing HPC Software Complexity with Spack. ISC-HPC 2022. Hamburg, Germany. May 29, 2022.

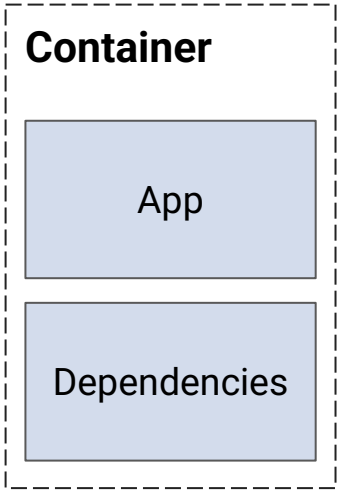
Portability: deploying applications on different platforms requires system configuration/rebuilding

(a) impractical to build large applications manually
(b) can be difficult to port builds to new systems

Containers - achieve lightweight portability



Solution - combine containers with Spack



Docker → **develop and distribute a container**

Spack (package manager) → **populate the container**

Singularity → **deploy the container on HPC**

Prototype locally

Store in the cloud

Deploy anywhere

Objectives

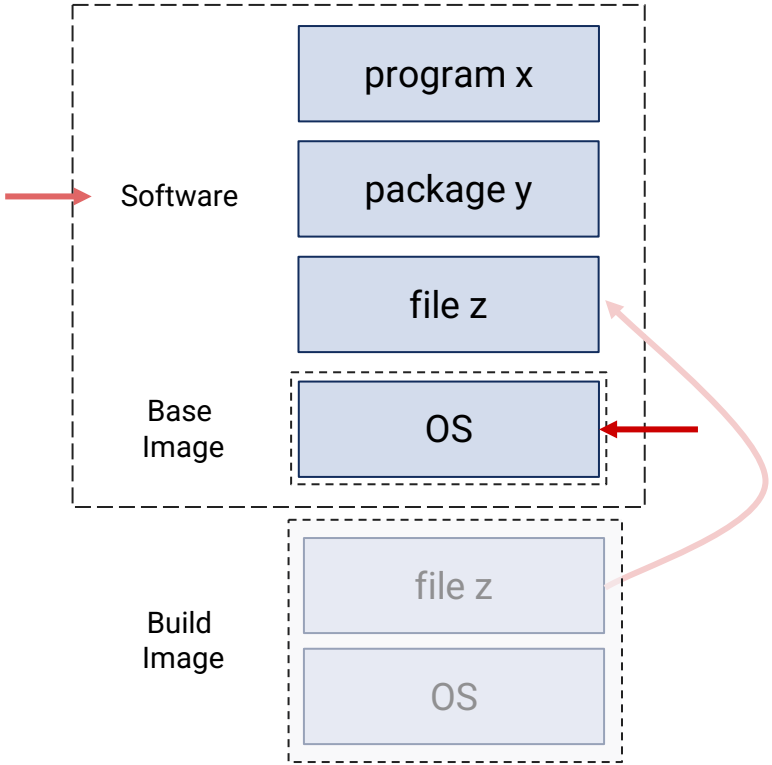
1. **Containerize Samurai**: a fragile application with build challenges
    a. **Use Spack to setup** the container **software environment**
    a. **Optimize containers for** multiple HPC architectures (**CPU and GPU**)
    a. **Verify correct program output** and **evaluate performance** on Cheyenne and Casper; **compare to bare-metal** Samurai
1. **Develop a lightweight container** that is easy to distribute
1. Extra Credit: Demonstrate how containerized scientific applications can be easily deployed on different platforms (NCAR clusters, DOE clusters, University clusters, AWS)
1. Extra credit: create a container/suite of containers for all multiple apps

What goes in a container?

Opaque View of Container

Virtualized OS
(isolated file system, specific OS lib/bin, sandboxed processes)

Container Internals



Dockerfile

```
FROM <base_image>
RUN <command>
COPY <local_file>
COPY <file_from_image>
ENTRYPOINT <shell>
```

Augmenting a container with Spack

Spack lets you install software how you want *and* manages package install complexity

version

customized dependencies

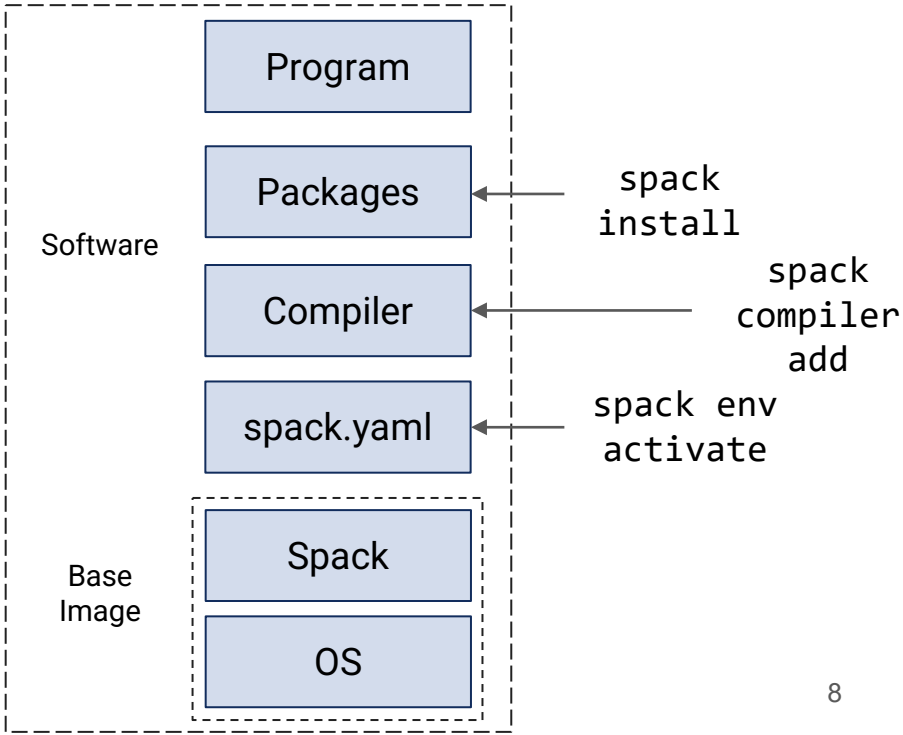
```
$ spack install netcdf-c@4.8.1%nvhpc^hdf5@1.12%gcc
```

package

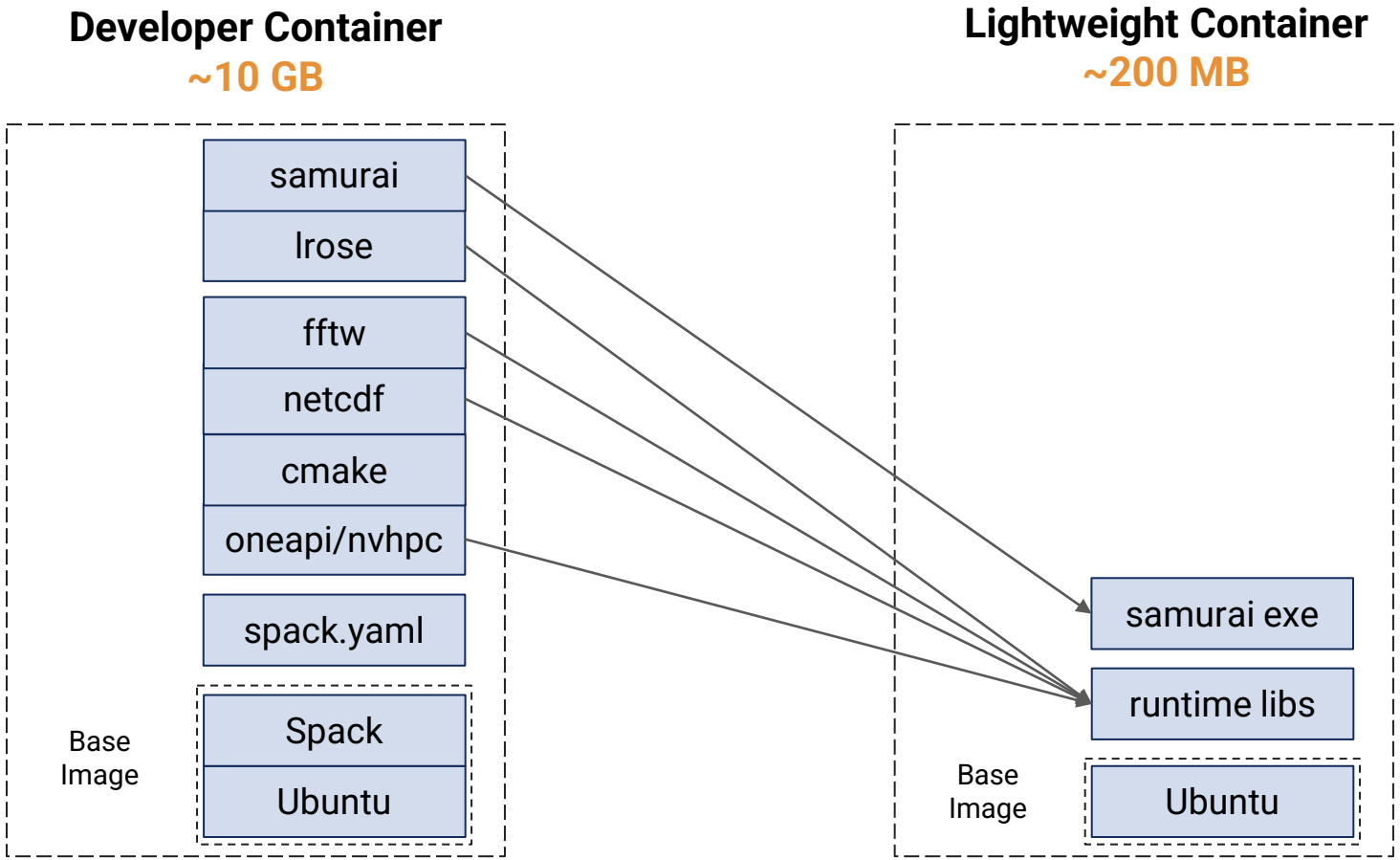
compiler

Container Internals

Spack Commands



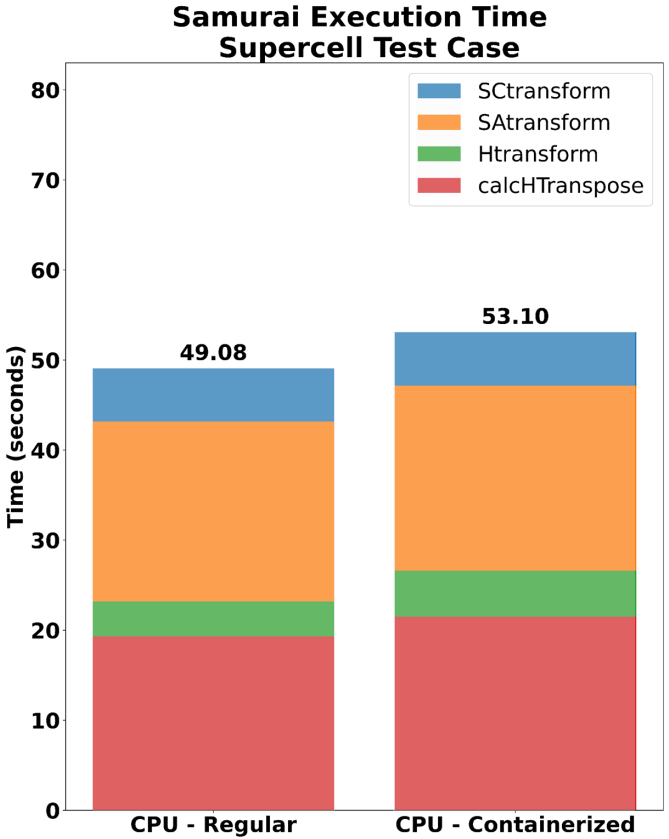
Containerizing Samurai

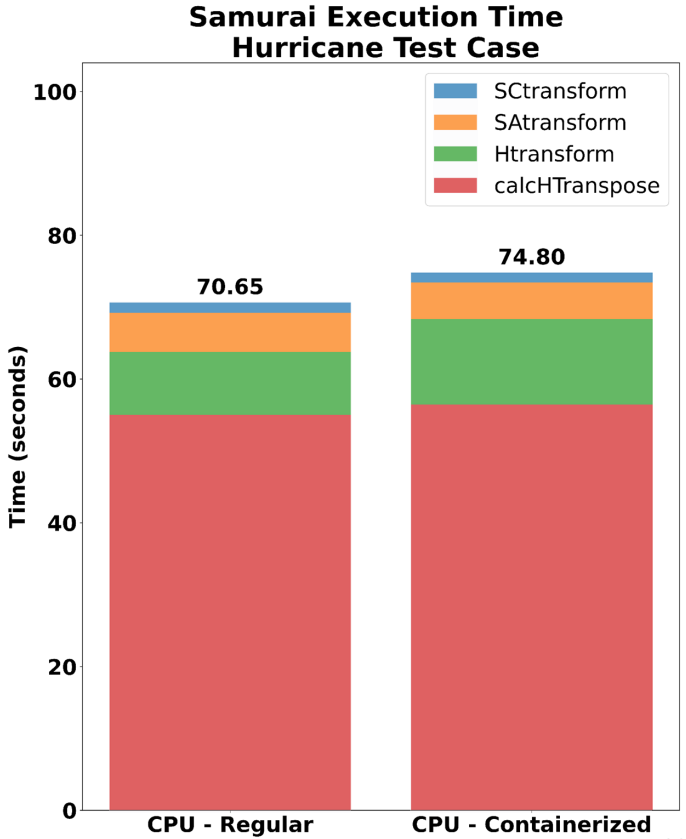


Portability and Performance - Cheyenne

System Configuration

- 1 full node
- 36 cpu threads
- 128 GB memory

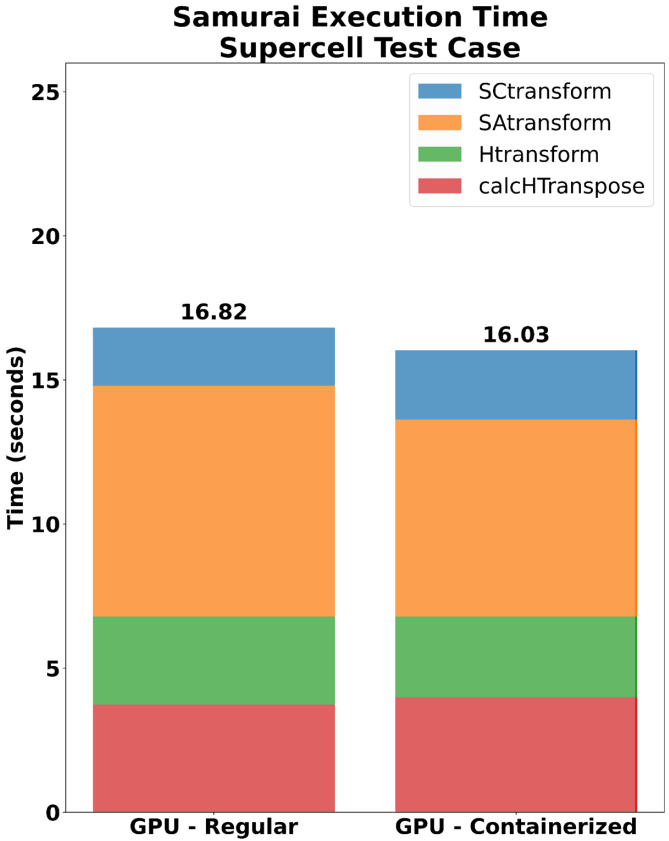


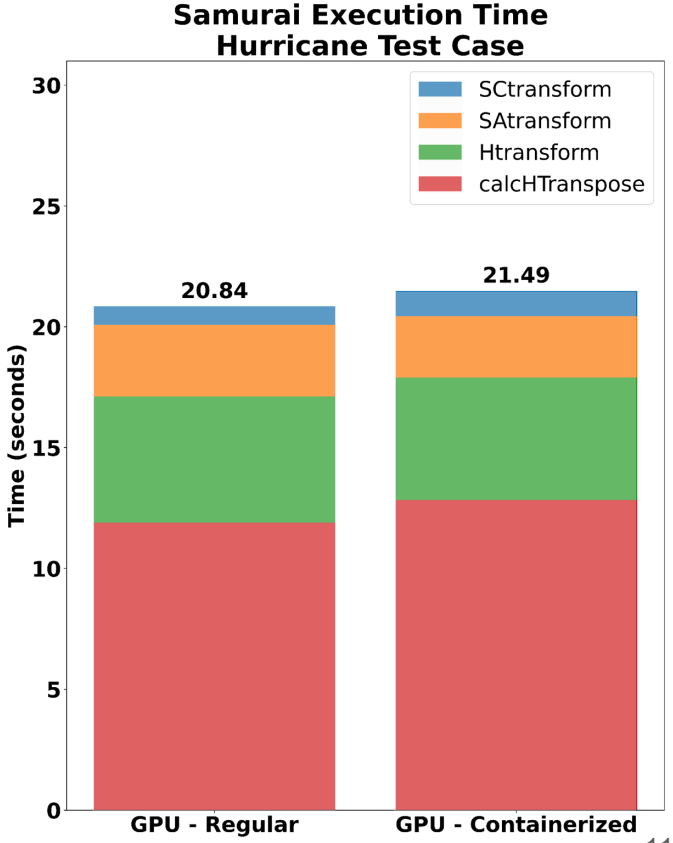


Portability and Performance - Casper

System Configuration

- 1 full node
- 1 cpu thread
- 1 V100 GPU
- 300 GB memory



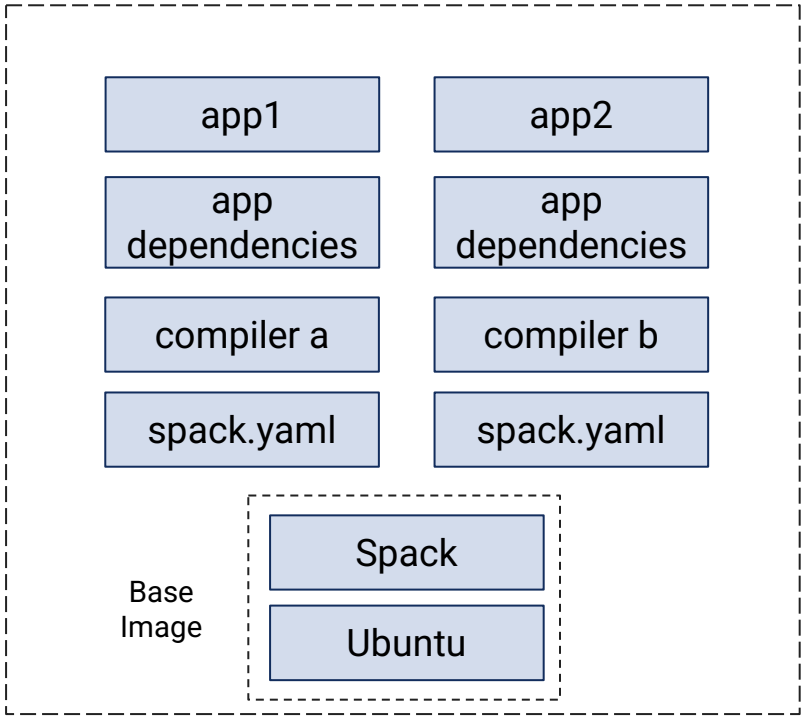


Outcomes

1. Demonstrated **containerized Samurai** provides **competitive performance** on both **Casper and Cheyenne**
1. **Customized containers to run on both CPUs and GPUs**
1. Generated **lightweight container** that is **easy to distribute**

Future Work

Containerized App Suite



Contributions to Extreme-scale Scientific Software Stack (E4S)

Many thanks to

Mentors: Jian Sun, Brian Vanderwende, and John Dennis

SIParCS organizers: Virginia Do, Francesgladys Pulido, AJ Lauer, and Jerry Cyccone