Singularity containers on HPC

W. Hayek, B. Bethwaite, B. Roberts

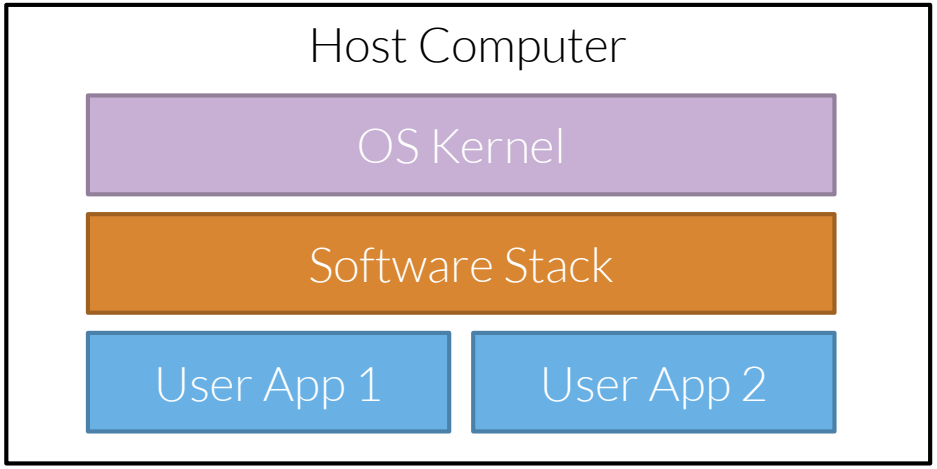
New Zealand eScience Infrastructure

Overview

1. Singularity containers
2. Containerising a real-world application – Map Cache Builder
3. Should everyone use containers?
4. Summary

Singularity containers

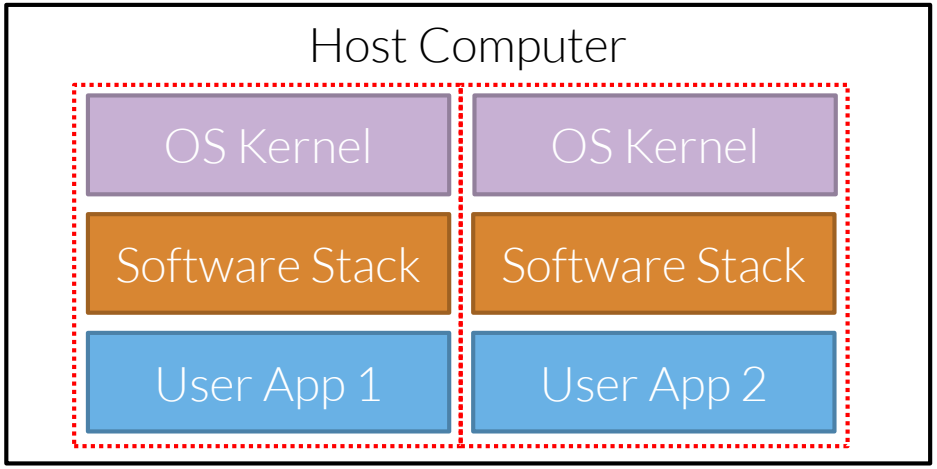
Singularity containers



No virtualisation ("bare metal server")

- Run user applications directly on host
- Share all resources (RAM, storage, peripherals)
- Share operating system kernel and software stack

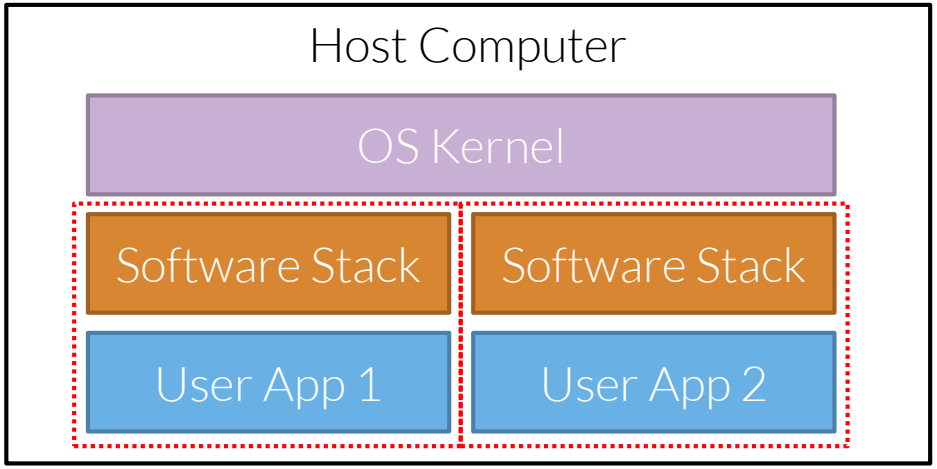
Singularity containers



Virtual machine ("full virtualisation")

- Separate OS kernel, resources, software stack
- Very flexible but very resource-hungry
- Can use Windows, Linux, MacOS, … inside VM

Singularity containers



Container ("OS-level virtualisation")

- Share kernel, but separate resources and software
- Less hungry for resources than full virtualisation
- Can use different Linux flavour in each container

Singularity containers

Main benefits

- Easy portability between systems
- Consistent operation – all dependencies included
- Version control of container images
- Facilitate resource management and isolation
- Automation using tools such as Puppet and Kubernetes

Use cases

- Cloud portability, scalability, modularity
- Microservice software architecture
- Reproducible science applications

Singularity containers

HPC and science constraints

- Shared environment – users cannot have root
- Low containerisation overhead is a must
- GPU, MPI, and scheduler support needed
- Integration with high-performance file systems
- Reproducibility – version management and immutable containers

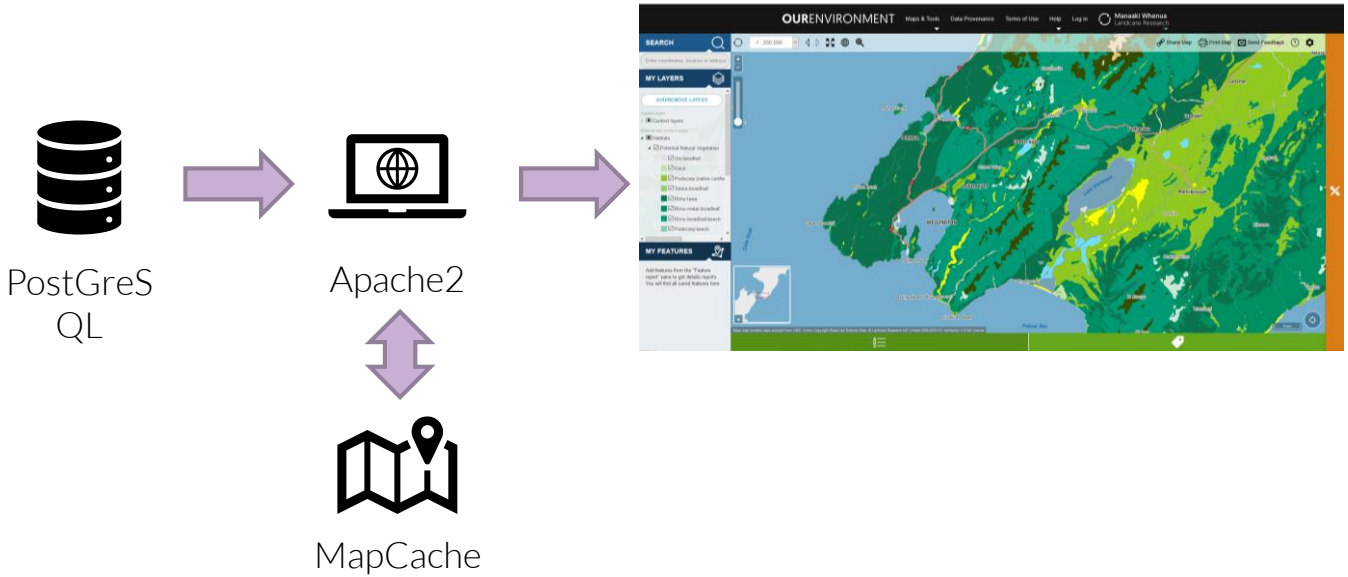
Singularity containers

Singularity

- Designed for HPC and science environments
- Supports MPI and works well with schedulers
- “Untrusted users running untrusted containers”
- Integrates well with HPC file systems
- Does not require root at runtime for user-space applications
- Can use Docker containers
- Containers are immutable by default to support reproducibility
- Currently under rapid development

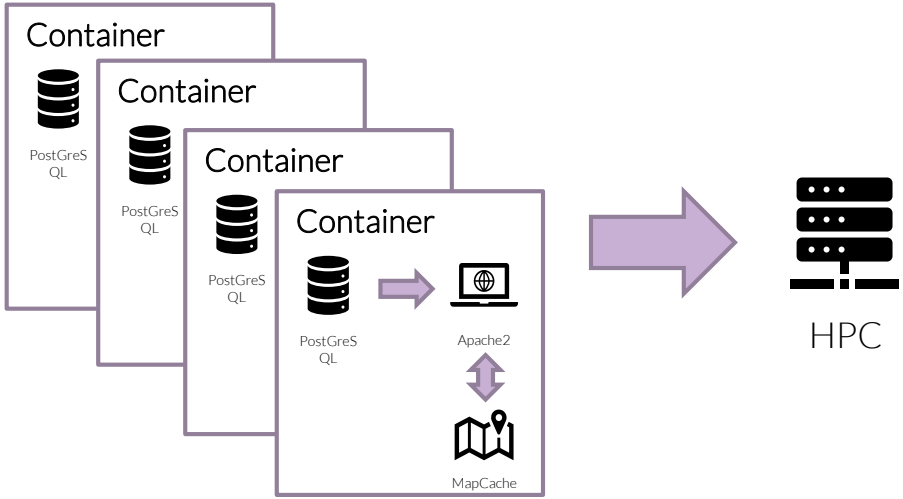
Containerising a real-world application

Containerising a real-world application



- Turn geospatial data into map layers for MWLR's map services

Containerising a real-world application



- Bundle up servers and worker app in image
- Deploy many container instances on the HPC
- Significant speed-ups – runtime reduced from weeks to hours

Containerising a real-world application

- NeSI consultancy project with MWLR
- Required integration with automation tools – built via Puppet/Docker
- Use network isolation – no access to PostGreSQL/Apache2 from the outside!
- Run PostGreSQL and Apache2 in user space (no root/separate user!) – ideal for portability, but needs a bit of hacking ☺
- “Fakeroot” feature available but some restrictions exist (limited support on GPFS/Spectrum Scale)

Should everyone use containers?

Should everyone use containers?

1. Potentially extra complexity in designing a container and useful software architecture
2. Extra complexity with setting up Slurm scripts, MPI integration etc.
3. Building containers can require a decent understanding of Linux administration when things go wrong
4. Security – containers could come from shady sources and run, e.g., a cryptominer
5. Fossilisation – software developers could be encouraged to defer code maintenance

Summary

Summary

- Singularity containers are a mature solution
- Can bundle up complex workflows and make them portable
- Still add some complexity - probably not practical for every use case

Join the Singularity workshop this afternoon!

Save the Date:

Science Coding Conference 2020
9 – 11 September 2020
Auckland, NZ

Call for Submissions open soon! Watch http://sciencecodingconference.nz for details

NeSI @ eResearch NZ - Talks & Workshops:

Wednesday 12 Feb

1:30 - 1:50 pm - **Megan Guidry** - Training: It's better together

1:30 - 5:30 pm - **Chris Scott** - First steps in machine learning with NeSI

1:50 - 2:10 pm - **Callum Walley** - Engineering HPC: What's going on?

2:10 - 2:30 pm - **Marko Laban** - Cloud-native technologies in eResearch: Benefits & challenges

2:50 - 3:00 pm - **Jun Huh** - Learning how to learn

3:30 - 4:30 pm - **Megan Guidry** - Building and supporting a NZ digital literacy training community

3:30 - 4:30 pm - **Blair Bethwaite** - Research Cloud NZ

Thursday 13 Feb

11:00 - 11:20 am - **Wolfgang Hayek** - Singularity containers on HPC

11:00 am - 12:20 pm - **Brian Flaherty** - Building a national/regional data transfer platform: Globus BoF

1:30 - 1:50 pm - **Nick Jones** - Advancing New Zealand's computational research capabilities and skills

1:30 - 1:50 pm - **Jun Huh** - User journey-driven product management

1:30 - 5:30 pm - **Blair Bethwaite** - Containers in HPC tutorial

1:50 - 2:10 pm - **Brian Flaherty** - Where Data Lives: NeSI, taonga and growing repository services

Thursday 13 Feb (cont.)

1:50 - 2:10 pm - **Jeff Zais** - Worldwide trends in computer architectures for data science

2:10 - 2:30 pm - **Dinindu Senanayake** - HPC for life sciences: Handling the challenges posed by a domain that relies on big data

3:30 - 5:30 pm - **Jana Makar** - Growing the eResearch workforce in an inclusive way

Friday 14 Feb

11:20 - 11:40 am - **Alexander Pletzer** - Enhancing eResearch productivity with NeSI's consultancy service

1:30 - 3:40 pm - **Nooriyah Lohani** - Research Software Engineering (RSE) community update and next steps in New Zealand