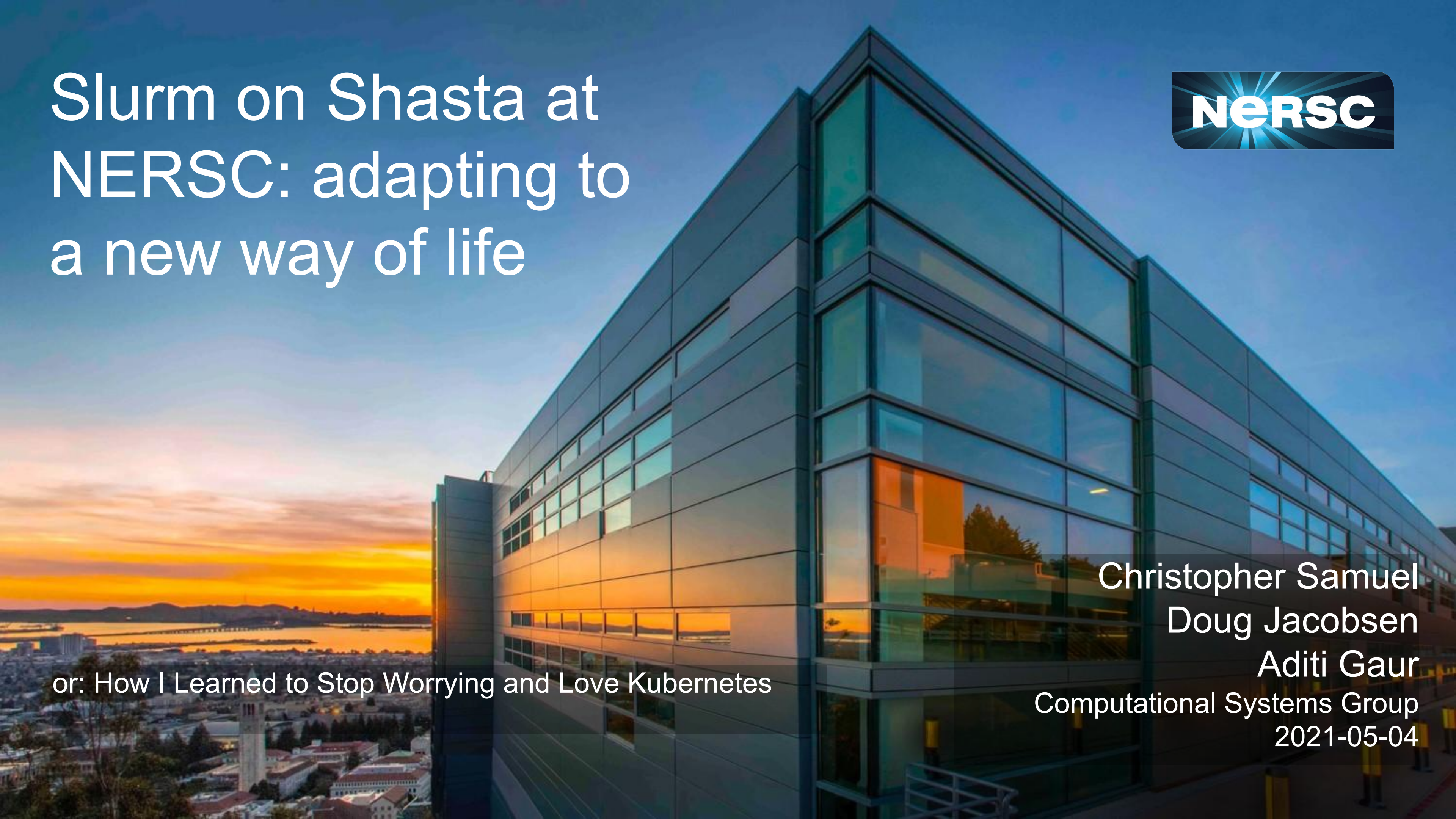


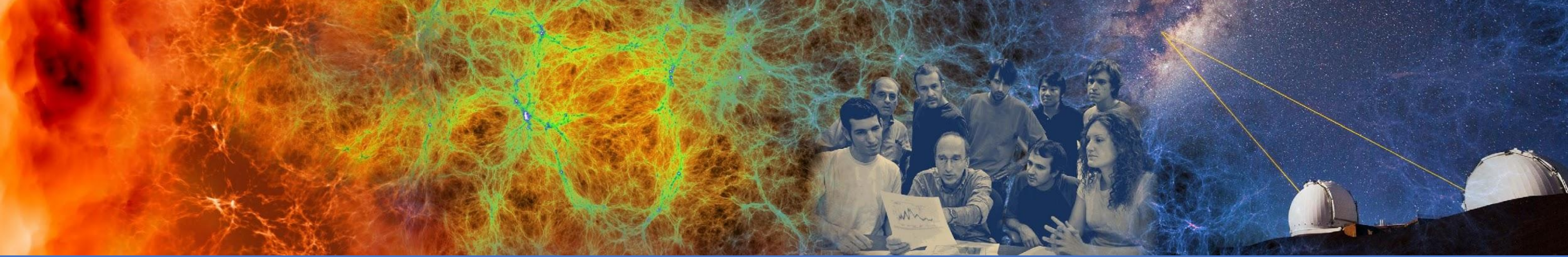
Slurm on Shasta at NERSC: adapting to a new way of life

or: How I Learned to Stop Worrying and Love Kubernetes

Christopher Samuel
Doug Jacobsen
Aditi Gaur
Computational Systems Group
2021-05-04

Acknowledgement of Country (from Australia)

I live and work on the land of the Ohlone first nations people and so I pay my respects to their Elders past, present and emerging.

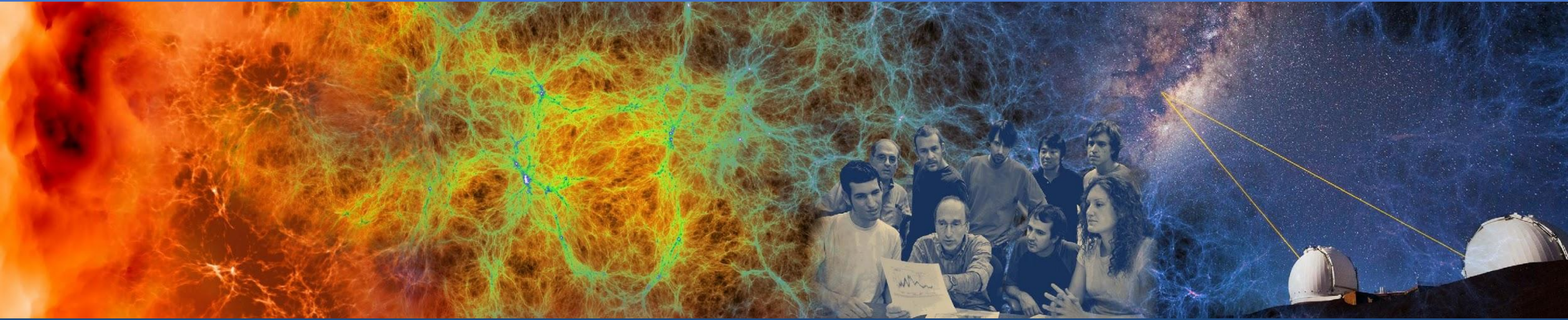


Where are we coming from?

Slurm on XC

- Everything in git
  - Slurm patches and build info in git
  - Configuration in the same git, deployed via ansible
  - All custom NERSC RPMs stored in their own git repo
- Challenges
  - Massively diverse workload, 1 to 9000 node jobs
  - Very high slurmctld load (lots of concurrent srun's & curious users)
  - Need to balance capability jobs with near-realtime workloads
- XC foibles
  - slurmctld nodes underpowered, driven a lot of optimisation work
  - No local disk, GPFS copes admirably with our Slurm I/O load

Shasta & N9: Implications & opportunities



Perlmutter - continuous operations!

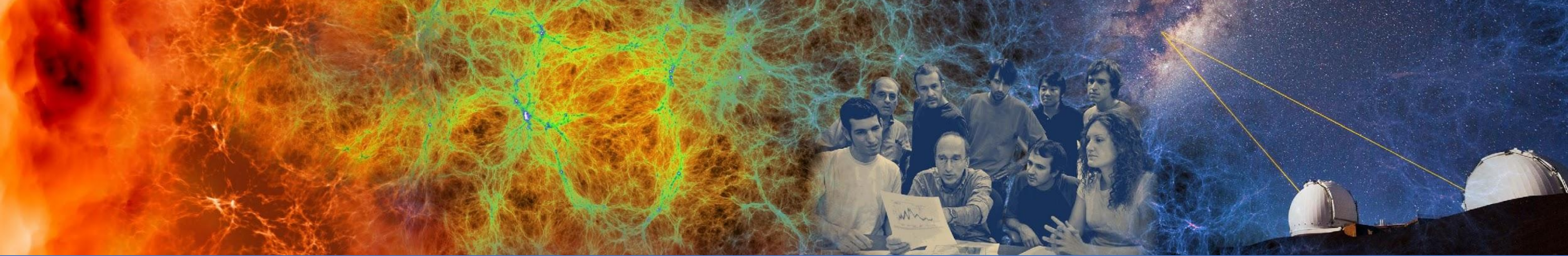
- A big goal of Perlmutter is “continuous operations”
  - No monthly maintenances with whole system down
  - Rolling upgrades wherever possible
  - Minimal disruption to user workloads
- Kubernetes offers great promise for this
  - Ability to move pods to other workers for HW or FW work
  - Deal with node failure by restarting pods on others
  - Failure is inevitable, cope with it
- How do we deal with Slurm in this environment?

Slurm, Shasta, Stoicism

- Isolation of components in pods
  - If something explodes the shrapnel should not hurt others
  - Rely on kubernetes for restarting failed slurmctld
- Leverage 3rd party operators and deployments
  - Avoid reinventing the wheel
  - Take advantage of others experience
  - “Do it better, faster”
- scrontab to replace use of crontab for users (NERSC NRE)
  - Fault tolerance, no issue with “favourite” login nodes going down
  - Requires Slurm 20.11.x (not in Shasta yet)

Why roll our own?

- NERSC is constantly updating and patching Slurm for our needs
    - So must have own containers with our own RPMs
- NERSC needs extra capabilities for pods - for example:
    - Lots of lua infrastructure for our job submit policy engine
    - redis container to locally cache project & user balances
    - postfix container for emails on job start/completion (TBD)
- Split out PVC creation for state directory from the slurmctld pod
    - Avoids helm deleting the state directory if the slurmctld chart uninstalled
- Split out database handling to separate, more fault tolerant, service



Where are we on Perlmutter now?

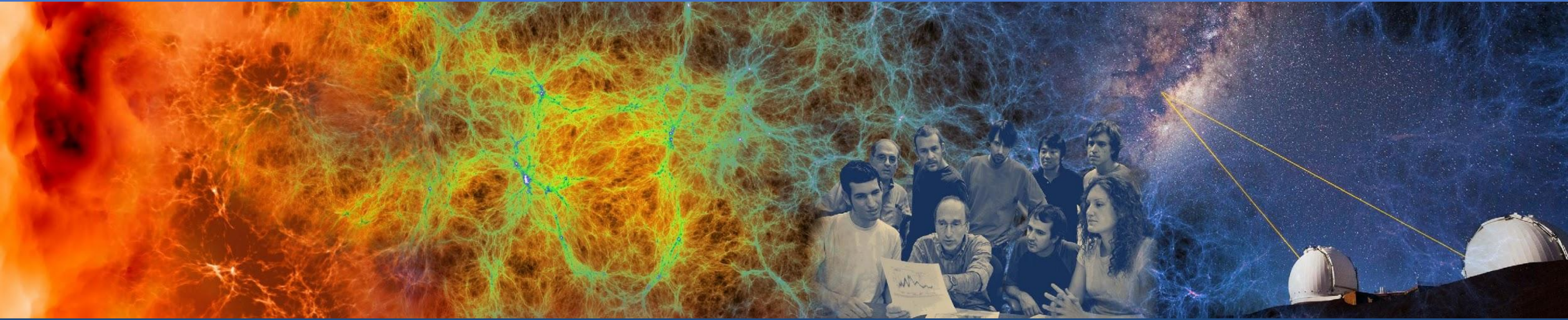
Current status

- MySQL Galera cluster for fault tolerance - “Boring but essential”
  - Deployed Percona XtraDB Cluster & Operator
  - Configures number of MySQL replicas with their PVCs
  - Adds load balancers, backups, etc.
- Build our own container images
- 3 independent Helm charts (for now)
  - slurm-pvc, slurmctld, slurmdbd
  - Slurm daemons have munge & sssd sidecars
  - slurmctld also has redis and nginx sidecars (so far)
  - sssd starts first, has lifecycle poststart check so next container only starts once LDAP lookups work - slurm daemons start last

Current status

- Slurm configuration
  - No liveness/readiness checks for Slurm daemons
    - we went to great pains to ensure systemd didn’t kill slurmctld, we don’t want kubernetes to do this and risk corrupting slurm state
  - Slurmctld configuration deployed as a configmap from git
    - lots of templating!
  - Slurmdbd configuration deployed as a secret
    - passwords stored in Hashicorp vault on NERSC manager VM
  - Configless mode to give single point of configuration
    - Simplifies compute node configuration
  - slurmd’s on login nodes for cron jobs, so have cached config
- Refactoring Slurm configs to use more templating than on XC

Ongoing work and pain points

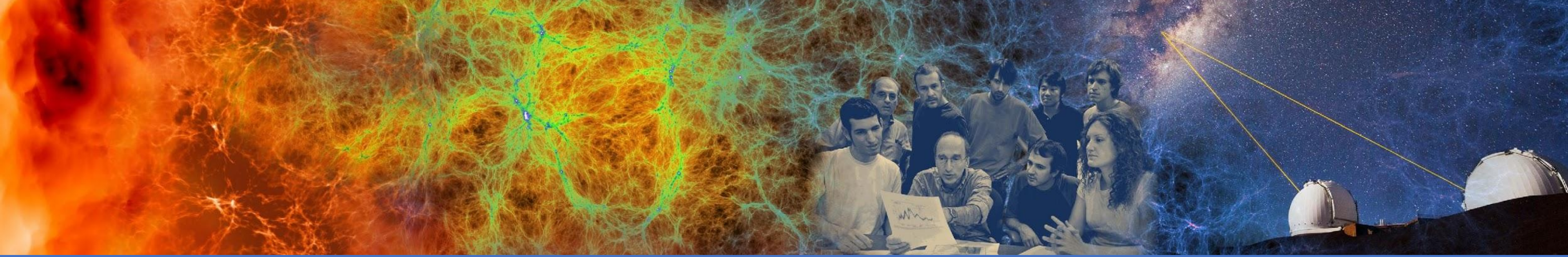


Ongoing work

- Configless mode is great, but...
  - Only covers certain config files, not scripts like prolog/epilogs etc
  - Have an nginx container as part of slurmctld pod that mounts a configmap which contains just these scripts
  - compute nodes to grab them with `wget -N` - only fetch if changed
- Need to send email - must add postfix sidecar to relay to NERSC MTA
- `scontrol reboot` needs capmc integration work
  - Currently "`node_reinit`" (reboot) via capmc not supported
  - Need sidecar that will power them off, wait, then power back on
- slurmrestd - REST API for slurm - run behind authenticating proxy
  - Waiting on info from HPE regarding integration for Shasta 1.4
- Capture core files from slurmctld - I mean it never happens, but...

Pain points

- No ability to add custom DNS SRV record support in Shasta 1.4
    - Needed for ideal support for Slurm configless mode
    - Have a workaround thanks to info from SchedMD
- Weird macvlan/multus issue
    - Uninstalling the slurmctld/slurmdbd helm chart results in inability to deploy again (presume same true for an upgrade)
    - IP address appears still in use via unreleased network namespace
    - Workaround from David Gloe @ HPE - have to scale the pod to 0 replicas before uninstalling/upgrading the helm chart.
- Grappling with slurm logs - vital to debugging Slurm & user issues
    - Storing in container works but with no rotation they get huge
    - Kibana unwieldy for large logs, kubectl doesn't show enough



🔌 QUICK PLUG 🔌

Slurm on HPE BoF: Friday 14th May 1400Z

- 6x10min slots for what you do with Slurm
- 1 hour for group discussion
- Email a brief talk idea to csamuel@lbl.gov
- Run by Aditi, Doug, myself (NERSC), Andrew (Pawsey)

NERSC BERKELEY LAB U.S. DEPARTMENT OF ENERGY Office of Science

Thank you! Any questions?