Los Alamos
NATIONAL LABORATORY

Mixed Compute Environments with OpenCHAMI

HPC-DO

Sean Gibson
Richard Kim
Samuel Quan
Travis Cotton (Mentor)
Thomas MacKell (Mentor)

ROSY: d3711a6a

Growing Demand for Mixed Workloads

1. Moving beyond traditional HPC workflows
   a. Kubernetes, Run:ai, similar
   b. Batch-scheduling vs cloud-based WLMs
2. Mixing contexts:
   a. HPC: finite resources, infinite workload demand (training)
   b. Cloud: infinite resources, finite workload demand (inference)
3. We would like to run both types of WLM on the same cluster

Los Alamos NATIONAL LABORATORY

Challenges for Mixed Workloads

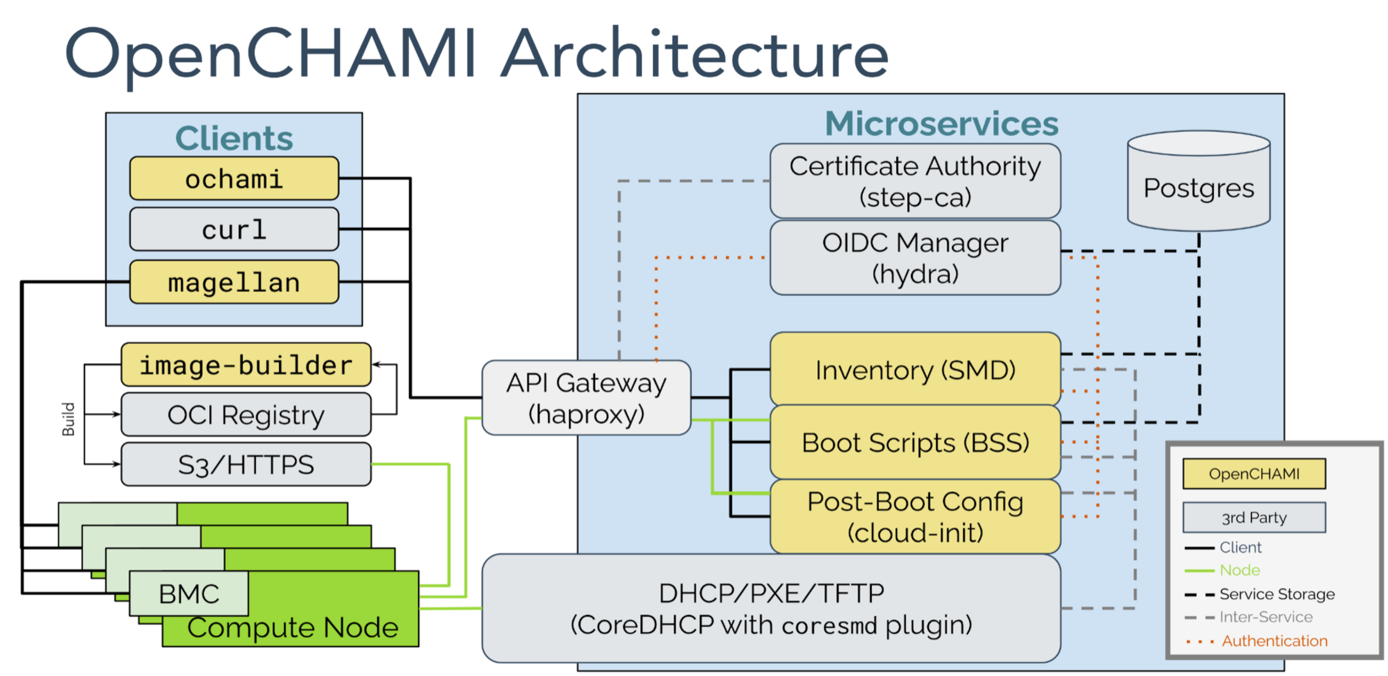
1. Static configuration of resources may lead to idling nodes
2. Downtime, fluctuating resource demands
    a. Idling nodes
    b. Take nodes down to swap to other workload domain
3. Must be able to quickly swap and scale as demand changes
4. Different WLM will require different setups and compute images

Deploying Slurm & Kubernetes with OpenCHAMI

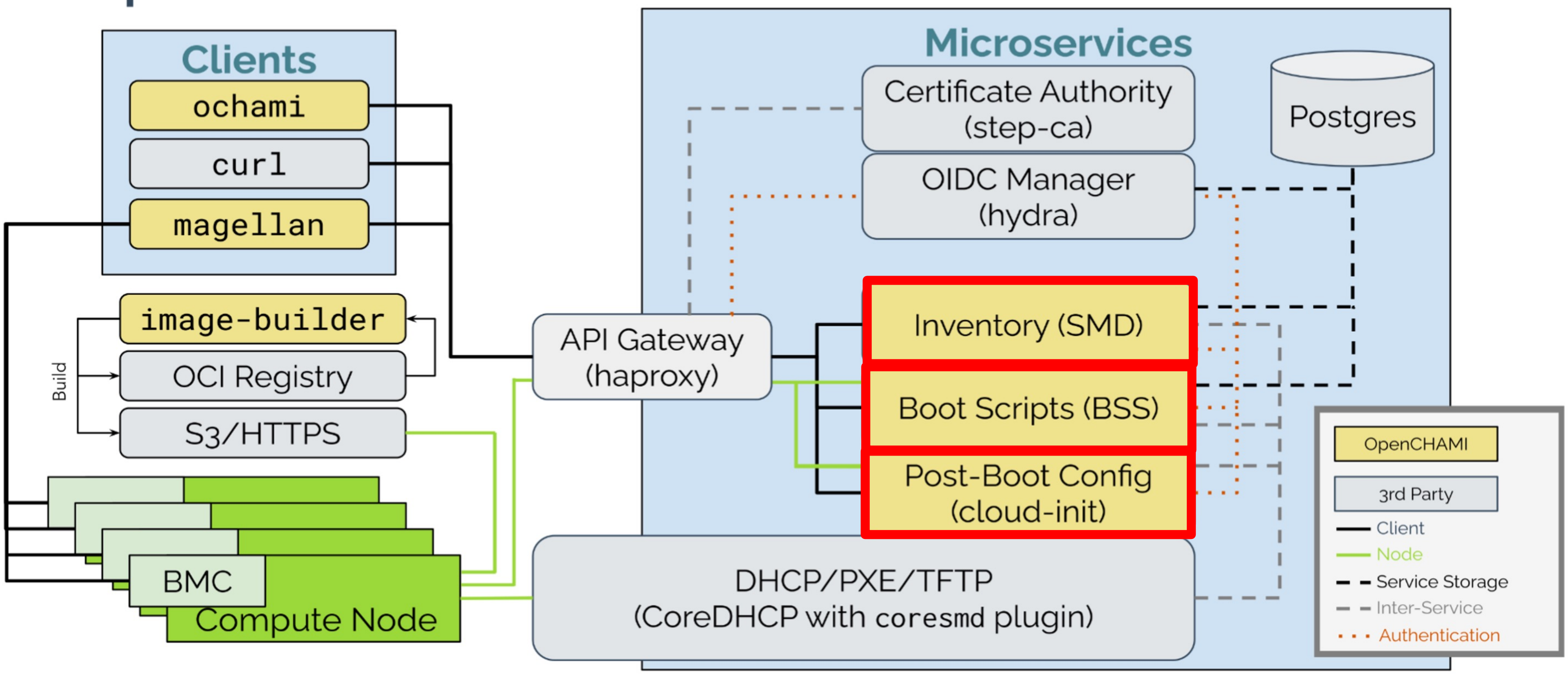
1. We deploy Slurm and K8s as our test workload managers
   a. Configure each image in Podman with the required resources
      i. K8s: kubectl, kubeadm, kubelet, kube-proxy
      ii. Slurm: slurmd, munge, chronyd
      iii. Both: networking setup
2. Setup our head node in a production environment with OpenCHAMI
   a. K8s: calico, storage classes, persistent volumes & claims
   b. Slurm: slurmctld, munge
3. Use OpenCHAMI to boot a node with the image

OpenCHAMI

1. OpenCHAMI is a cloud-like software that helps manage HPC environments.



OpenCHAMI Architecture



Los Alamos
NATIONAL LABORATORY

Configuring Environments w/ OpenCHAMI

1. Custom Images in BSS
   a. Save custom image for Slurm/K8s installs
      i. kernel
      ii. initrd
      iii. rootfs
   b. Store in BSS

Los Alamos NATIONAL LABORATORY

Configuring Environments w/ OpenCHAMI

2. Groups in Cloud-init
    a. File payload (runcmd)
        i. Kubernetes: joining control plane
        ii. Slurm: munge setup
        iii. Both: starting services

```
runcmd:
    - sudo nmcli connection reload
    - sudo nmcli connection up ens259f0
    - sudo systemctl enable --now docker
    # filesys
    - systemctl stop containerd
    - umount /var/lib/containerd
    - mount -t tmpfs -o size=20480M tmpfs /var/lib/containerd
    - systemctl restart containerd

    # NFS Server
    - dnf install nfs-utils -y
    - mkdir -p /nfs/imports/myshare
    - sudo mount -v \
      -t nfs 172.16.0.254 /nfs/exports/myshare \
      /nfs/imports/myshare/

    # misc
    - sysctl -w net.ipv4.ip_forward=1
    - systemctl enable --now kubelet
    - systemctl disable --now firewalld
    - kubeadm join 10.15.3.41:6443 --token ufodre.0rdcrsy9ro2scbah
```

Los Alamos NATIONAL LABORATORY

Configuring Environments w/ OpenCHAMI

```
runcmd:
    - sudo nmcli connection reload
    - sudo nmcli connection up ens259f0
    - sudo systemctl enable --now docker
    # filesys
    - systemctl stop containerd
    - umount /var/lib/containerd
    - mount -t tmpfs -o size=20480M tmpfs /var/lib/containerd
    - systemctl restart containerd

    # NFS Server
    - dnf install nfs-utils -y
    - mkdir -p /nfs/imports/myshare
    - sudo mount -v \
      -t nfs 172.16.0.254:/nfs/exports/myshare \
      /nfs/imports/myshare/

    # misc
    - sysctl -w net.ipv4.ip_forward=1
    - systemctl enable --now kubelet
    - systemctl disable --now firewalld
    - kubeadm join 10.15.3.41:6443 --token ufodre.0rdcrsy9ro2scbah
```

Los Alamos NATIONAL LABORATORY

Swapping between Workloads

1. With WLMs setup, now we need a way to quickly swap and scale compute resources between them
2. Goals
   a. Quickly swaps nodes
   b. Support heterogeneous workloads to run on the same cluster
      i. Slurm and Kubernetes

Los Alamos NATIONAL LABORATORY

SPREAD

1. Command Line Tool: SPREAD
    a. Quick swaps
    b. Heterogeneous workloads
    c. Supports nodesets
    d. Manages post-boot scripts

Figure 1 SPREAD© logo

SPREAD: Managing Custom Images

1. Capabilities
    a. addImage <initramfs> <rootfs> <kernel> opt: <ci-group(s)>
        i. Stores image into minIO/S3/local
        ii. creates cloud-init config
    b. deleteImage
    c. listImage(s)

```
(senv) [root@cb-head ochami]# spread listImages
```

Nodes by Image

kube 5.14.0-570.23.1.el9_6.x86_64

Short Name	X-Name	Mac Address
cb09	x1000c0s8b0n0	ec:e7:a7:05:96:f0
cb08	x1000c0s7b0n0	ec:e7:a7:05:95:b0
cb07	x1000c0s6b0n0	ec:e7:a7:05:a2:78
cb06	x1000c0s5b0n0	ec:e7:a7:05:98:d0
cb05	x1000c0s4b0n0	ec:e7:a7:05:a0:0c
cb04	x1000c0s3b0n0	ec:e7:a7:05:9e:d0
cb03	x1000c0s2b0n0	ec:e7:a7:05:9f:04
cb02	x1000c0s1b0n0	ec:e7:a7:05:93:40
cb01	x1000c0s0b0n0	ec:e7:a7:05:a1:5c

slurm 5.14.0-570.23.1.el9_6.x86_64

Short Name	X-Name	Mac Address

slurm test2

test 5.14.0-570.25.1.el9_6.x86_64

Los Alamos NATIONAL LABORATORY

SPREAD: Switching Images

1. Capabilities
   a. change <node> <image_name>
      i. Changes BSS params & image for node
      ii. remove from old ci-group
      iii. add to new ci-group
      iv. runs optional config commands
      v. Download and load new kernel on node
      vi. Create symlink to catch reboots

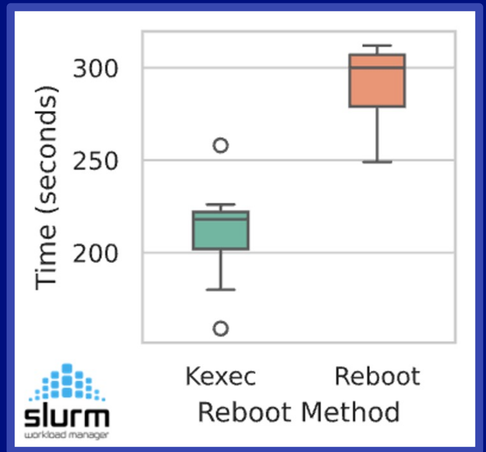
```
[root@cb-head ~]# spread change cb05 slurm
node:  cb05
Multiple versions for image: slurm
0 :  5.14.0-570.23.1.el9_6.x86_64
1 :  test2
Select your option [0-1]: 0
cb05 swapped to slurm
```

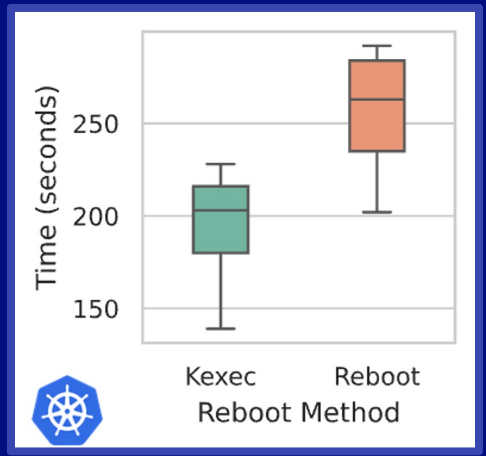
Los Alamos
NATIONAL LABORATORY

SPREAD: Speed Test

1. Compare kexec vs traditional reboot node ready times
    a. Measure time from reboot command to availability on the workload manager
    b. Use scontrol/kubectl logs to get the time first available
2. We find kexec provides significant speedup
    a. Slurm: 27.8% average speedup
    b. K8s: 24.1% average speedup

Kexec vs Reboot Ready Times for Slurm & K8s





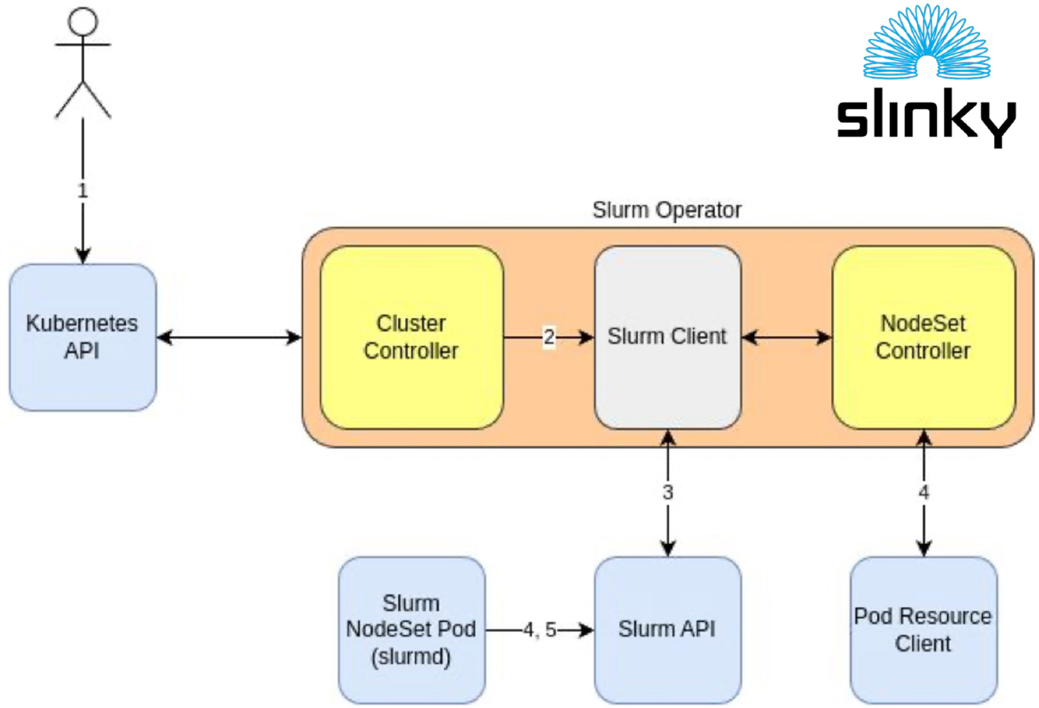
Los Alamos NATIONAL LABORATORY

Slurm Slinky

Slinky (Deployed + Tested)

a. Run Slurm in K8s

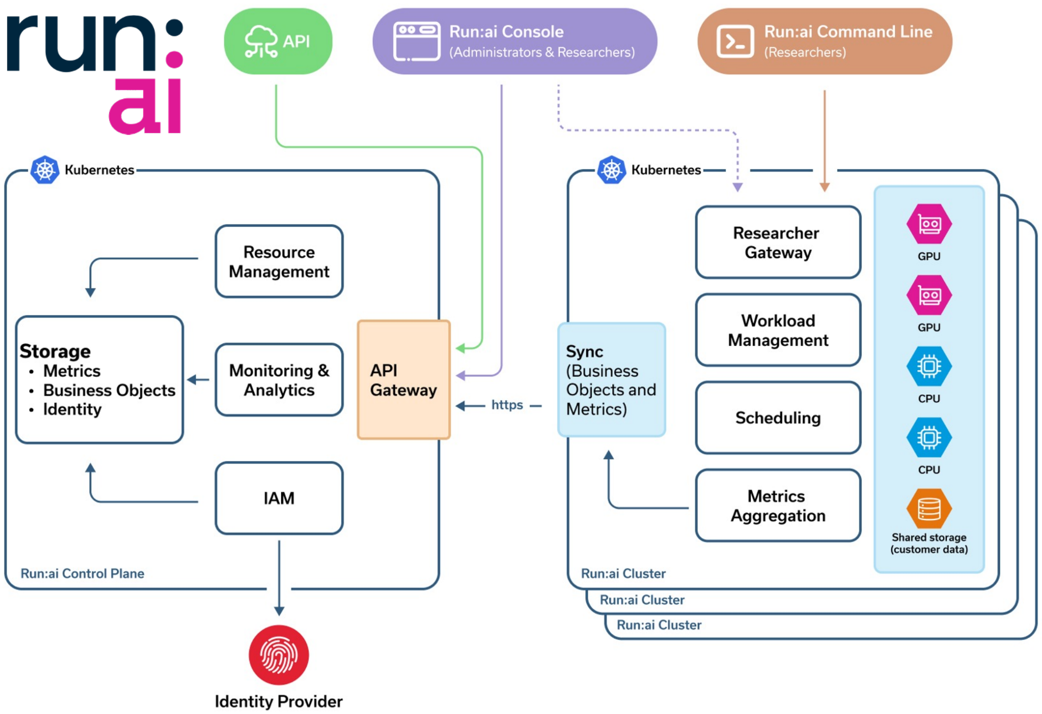
b. Auto-scale Slurm clusters running as K8s pods



Run:ai

Run:ai (Deployed)

a. GPU orchestration tool
b. no GPUs
    i. fake-gpu-operator
    ii. Github issue



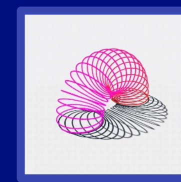
Conclusion

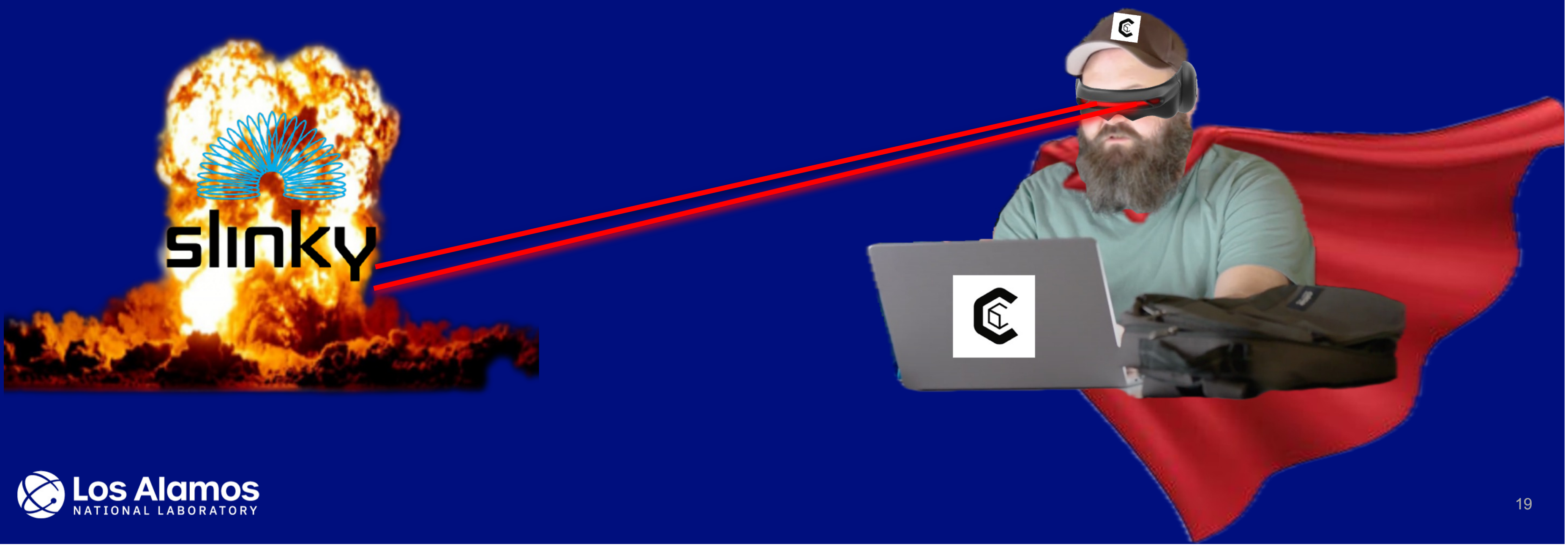
1. Deployed Kubernetes and Slurm images with OpenCHAMI
2. Used OpenCHAMI to swap nodes between compute environments
3. Built command line tool SPREAD to manage swapping
4. Deployed and tested containerized resources on our cluster
    a. Slurm Slinky
    b. Run:ai

Future Directions

1. Test SPREAD on larger clusters
2. Optimize ready times (dracut)
3. Integrate SPREAD into OpenCHAMI
4. Daemon-ize SPREAD to hold internal states and automatically scale workloads
5. Partition intra-node resources across workloads

Questions?





Los Alamos
NATIONAL LABORATORY

QUESTIONS

What does SPREAD stand for?

Shell-based **P**rovisioning for **R**esource **E**nvironment **A**llocation and **D**istribution