SC21
St. Louis, MO | science & beyond.

Chameleon at NCAR

Running a bare-metal cloud on found hardware
Or: Adventures with ARM nodes

Michael Sherman
Computer Science, University of Chicago
MCS, Argonne National Laboratory
shermanm@uchicago.edu

Chameleon in a Nutshell

- Chameleons like to change: a testbed that adapts itself to your experimental needs
  - **Deep Reconfigurability** (bare metal) and isolation + KVM cloud (cost/isolation trade-off)
  - **Capabilities**: power on/off/reboot, custom kernel, serial console, etc.
  - **Democratized Hardware:** NSF funded, no-cost access unlike Commercial Clouds
- Balance: **large-scale** vs **diverse** hardware
  - Large-scale: large homogenous partition (~15,000 cores), ~6PB of storage over 2 sites (**UC**, **TACC**) connected via 100G network
  - Diverse: X86, Intel, AMD, ARM (on CHI@NCAR), FPGAs, GPUs, Corsa SDN switches, etc.
- Cloud++: Chameleon Infrastructure (CHI) via mainstream cloud tech
  - Powered by **OpenStack** with bare-metal reconfiguration (Ironic) + “special sauce”
  - Resource reservations (Blazar) contributed as official OpenStack component
  - **CHI-in-a-Box** sites at Northwestern, and in-progress: NCAR, IIT, EVL
- Workflows for Researchers: Reproducibility, Repeatability, and Sharing
  - Users can package their imperative and non-transactional experiments using **Jupyter,** share it via **Trovi**, publish via **Zenodo** integration, and let anyone reproduce them with **Daypass**.

Open Testbed – By The Numbers

400+
Papers published

45
Countries

160+
Institutions

Over
6,000
Users

750+
Unique projects

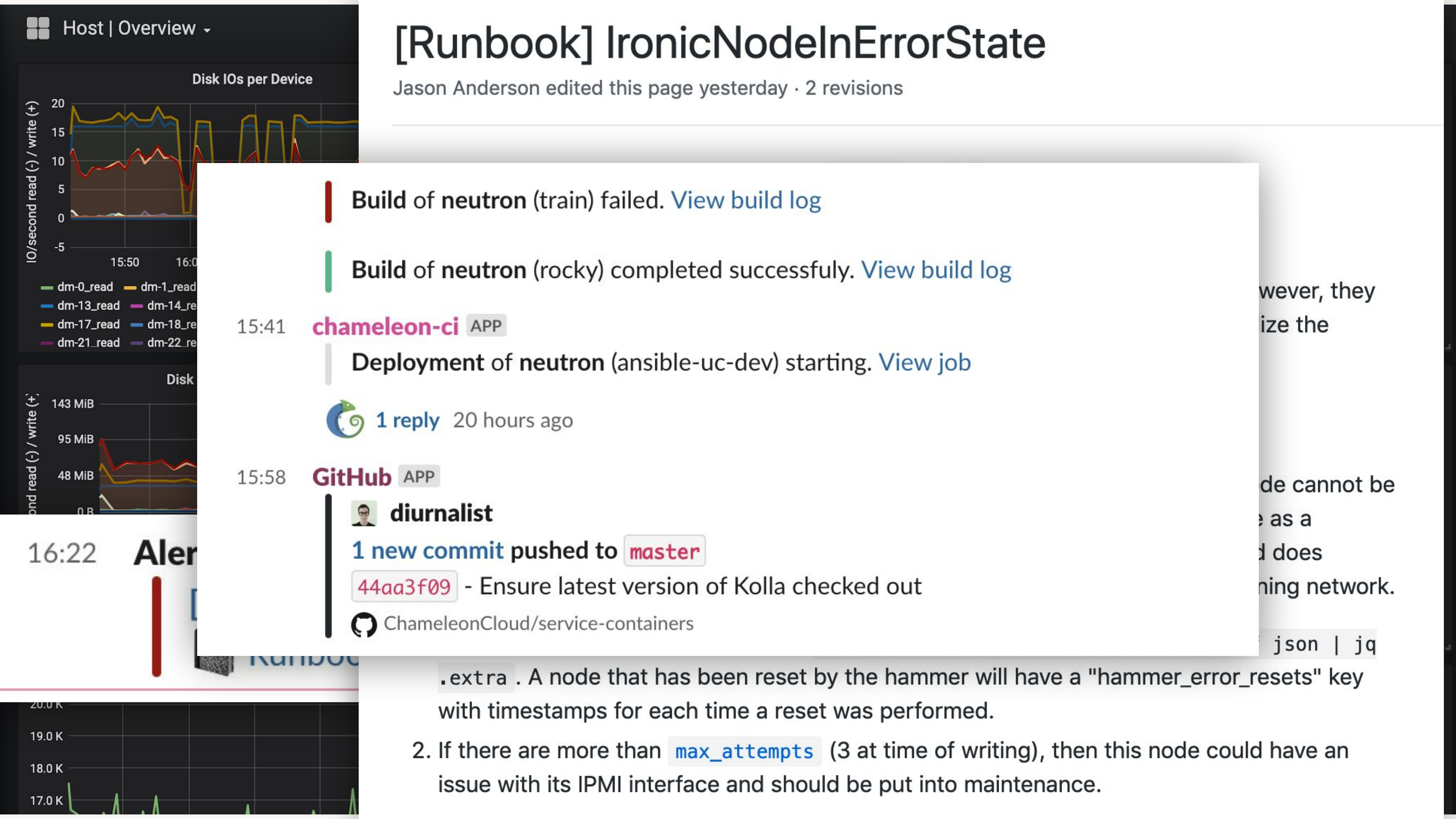
6+
Years Old

Challenges in Operating a Bare-Metal Cloud

- Different from HPC
  - User freedom: Root access, serial console, full disk images
  - No batch scheduling, user has interactive access. Time to deploy is crucial.
  - Repeatability and isolation: Deploy requires new disk image and networking
  - Dynamic, multi-tenant networking environment
- Cost vs Reliability
  - Very Heterogeneous hardware
  - Higher hardware and operations costs per node
  - Availability, repeatability more important than perfect utilization
- More things to go wrong
  - Hardware is often unreliable, especially chassis managers for provisioning
  - Users are interactive, no scheduler, triggers frequent deploys
  - Failures can be the user’s fault (specifying a bad disk image), or the system’s (bad cable, temporary service outage).
  - Sometimes both (User sets a kernel flag that should work, but firmware crashes)

Lowering Operations Cost with Chameleon and OpenStack

- Working with mainstream open source project
  - Familiar interfaces, transferable skills: 858 deployments, 441 organizations, 63 countries
  - Working with large community (~8,400 total contributors, ~6,000 reviewing code)
  - Access to existing documentation and support systems
- Consistent deployment process, scaling horizontally as hub-and-spoke
  - Releases created at UChicago, deployed to Dev, Staging, then Production Site
  - All other sites pull and deploy releases
  - Supported disk images and experiments follow the same pattern
- Usability / Reliability
  - We eat our own dog food, catch errors and surprises early
  - Work to make the deployment process easy, consistent, and frequent
  - Favor multiple small releases over giant risky ones
- A suite of operational tools
  - Monitoring and Alerting: smoke tests, live monitoring with coverage, centralized logging
  - Remediation: runbooks and hammers (automated repair)
  - Create a process around maintenance (automated scripts ensure uniformity)



[Runbook] IronicNodeInErrorState

Jason Anderson edited this page yesterday · 2 revisions

Build of **neutron** (train) failed. View build log

Build of **neutron** (rocky) completed successfuly. View build log

15:41 **chameleon-ci** APP

Deployment of **neutron** (ansible-uc-dev) starting. View job

1 reply 20 hours ago

15:58 **GitHub** APP

diurnalist

1 new commit pushed to `master`

`44aa3f09` - Ensure latest version of Kolla checked out

ChameleonCloud/service-containers

16:22 Aler

Runboo

wever, they

ize the

de cannot be

e as a

d does

ning network.

json | jq

`.extra`. A node that has been reset by the hammer will have a "hammer_error_resets" key with timestamps for each time a reset was performed.

2. If there are more than `max_attempts` (3 at time of writing), then this node could have an issue with its IPMI interface and should be put into maintenance.

CHI-in-a-Box: Chameleon Near You!

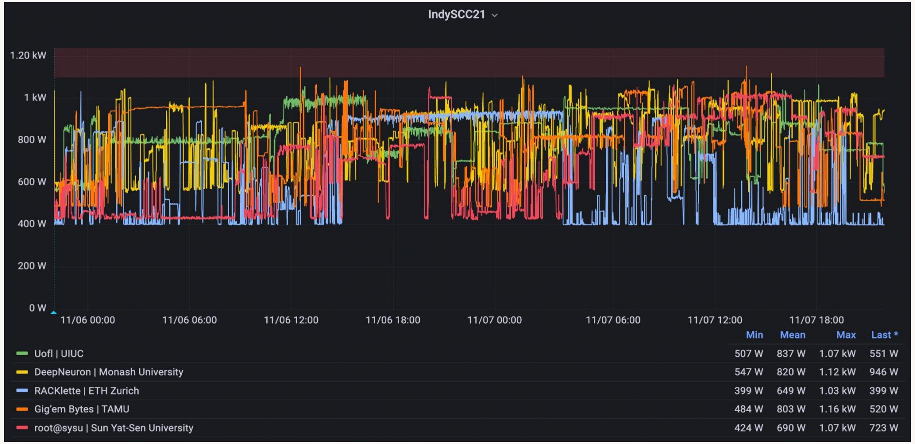
- CHI-in-a-box: packaging a commodity-based testbed
  - First released in summer 2018, continuously improving
  - Packaging systems as well as operations model
- CHI-in-a-box scenarios
  - Chameleon extension: join the Chameleon testbed (currently serving only selected users), and includes both user and operations support
  - Part-time Chameleon extension: like Chameleon extension but with the option to take the testbed offline for certain time periods (support is limited); includes a stable point of presence and BYOD for dynamically adding and removing resources
- Take advantage of some centralized services:
  - Federated authentication with single-sign-on
  - Usage enforcement and accounting
  - Jupyterhub with Chameleon API and authentication baked in
  - Hardware discovery and resource browser
- https://github.com/chameleonCloud/chi-in-a-box

Bringing up the NCAR site for IndySCC 2021

- Needed dedicated, interesting hardware for 5 teams, but supply chains are a mess
  - Repurposed “found” ARM nodes at NCAR, originally for cluster evaluation
- Use chi-in-a-box to deploy control plane running on single VM
  - 4 cores, 8gb memory (after some tuning), public and private network interfaces, 100gb disk space to hold user images.
- Had to port some capabilities from upstream, now supported for all sites
  - Enabled ipxe, and compiled binary for arm64 (not in repos)
  - Built ironic pre-boot agent for ARM64
  - Built new supported images with UEFI boot support
  - Disabled multi-tenant networking, dedicated switch wasn’t available yet
  - Grab power metrics from PDUs when out of band controller lacks support
- Lessons learned
  - System was able to function despite very minimal support from the ARM hardware, documentation, or vendor
  - BMCs are flaky. Had to disable some “auto-healing” mechanisms in OpenStack
  - A lot of layers to run a complex system, but familiar components underneath

Outcome and Parting Thoughts

- After SC21, (and some sprucing up), CHI@NCAR will be available for general use by our community
- If you wish your spare hardware was more useful, please get in touch!
- I'm happy to take any questions and comments.



	Min	Mean	Max	Last *
🟩 Uofl | UIUC	507 W	837 W	1.07 kW	551 W
🟨 DeepNeuron | Monash University	547 W	820 W	1.12 kW	946 W
🟦 RACKlette | ETH Zurich	399 W	649 W	1.03 kW	399 W
🟧 Gig'em Bytes | TAMU	484 W	803 W	1.16 kW	520 W
🟥 root@sysu | Sun Yat-Sen University	424 W	690 W	1.07 kW	723 W

"Lessons Learned from the Chameleon Testbed". In Proceedings of the 2020 USENIX Annual Technical Conference (USENIX ATC '20), https://bit.ly/3omq5dL

IndySCC power metrics available at https://chi.hpc.ucar.edu:3000
Become a new user at https://chameleoncloud.org
Getting Started docs at: https://chameleoncloud.readthedocs.io/