Rebuilding Bridges

The tools used to deploy and maintain Bridges-2

G. Ryan Sablosky
sablosky@psc.edu
Pittsburgh Supercomputing Center
Pittsburgh, Pennsylvania, USA

ABSTRACT

Bridges-2 is an NSF-funded heterogeneous supercomputing cluster at the Pittsburgh Supercomputing Center. The successor to the Bridges system (2014-2021), Bridges-2 builds on the flexibility demonstrated by its predecessor to support a wide variety of scientific workflows. This paper, building on a 2017 overview of the infrastructure supporting the original Bridges, is intended as a mid-cycle overview of the infrastructure developed to support the Bridges-2 project. It covers the lessons learned from the predecessor system, the initial design and development of the support infrastructure, modifications and improvements made over the last two years of production operations, and how those improvements have been shared with other systems and projects across the Pittsburgh Supercomputing Center.

CCS CONCEPTS

• **Social and professional topics → System management**; **Network operations**; **File systems management**.

KEYWORDS

System Administration, High Performance Computing, Configuration Management, Version Control

ACM Reference Format:
G. Ryan Sablosky. 2023. Rebuilding Bridges: The tools used to deploy and maintain Bridges-2. In *Practice and Experience in Advanced Research Computing (PEARC '23), July 23–27, 2023, Portland, OR, USA.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3569951.3597589

1 INTRODUCTION

Much like its predecessor, Bridges[16], Bridges-2[4] is a heterogeneous platform designed to support computational and data-intensive workflows across a wide variety of scientific fields. To handle general compute, Bridges-2 contains 488 Regular and 16 Large Memory nodes, each with two 64-core AMD EPYC 7742 CPUs, and 256 or 512 GB of RAM respectively. For larger datasets, there are four Extreme Memory nodes, with four 24-core Intel Xeon Platinum 8260M CPUs and 4TB of RAM. For workflows requiring GPUs, Bridges-2 contains 24 GPU nodes, each containing two 20-core Intel Xeon Gold 6248 CPUs and eight NVIDIA Volta V100 GPUs[16]. Additionally, resources from the Bridges-AI[6] extension to the original Bridges system were migrated to Bridges-2, adding nine HPE Apollo 6000 Gen10 servers with eight NVIDIA V100 GPUs apiece, and an NVIDIA DGX-2 system with a further 16 NVIDIA V100 GPUs. Storage for Bridges-2 is provided by two Lustre filesystems, a 200TB fast NVMe-based filesystem called Jet and a 15PB hard disk-based filesystem called Ocean, both provided by an HPE Clusterstor E1000 system [16]. The infrastructure powering Bridges-2 also supports Neocortex[5], a unique compute resource consisting of two Cerberas CS-2 AI accelerator units and a dedicated HPE Superdome Flex support cluster. During the deployment of Bridges-2, the team wanted to address some of the shortcomings of the infrastructure that had grown to support the original Bridges.

2 PREVIOUS BRIDGES INFRASTRUCTURE

2.1 Provisioning

Provisioning duties on Bridges were handled by OpenStack[18, 33]. This had the appealing property of being able to handle the provisioning of virtual machines and hardware nodes using the same framework. Unfortunately, as a large and complicated software system, over the lifetime of Bridges OpenStack and its maintenance burden proved to be more trouble than it was worth. Requests from researchers for virtual machines ended up being below the expectation of "potentially hundreds"[33], and the relatively minimal churn of hardware nodes requiring re-provisioning meant that much of the power and flexibility of OpenStack went underutilized. For Bridges-2, we decided to switch to a more simple, standalone hardware provisioning solution, letting an oVirt-based[20] cluster handle VM responsibilities.

2.2 Configuration management

Configuration management on Bridges was built on Puppet[22, 33]. After evaluating alternative configuration management systems, such as Ansible[2], Salt[30], and Chef[7], we did not find a compelling reason to move away from a solution we had institutional experience with. While Puppet's performance and feature set were satisfactory, the organic growth of the configuration codebase we had developed over time made diagnosing problems, adding new machines, and incorporating new features a hassle. In addition, modern development best practices were not adhered to. Version control was not enforced, making it nearly impossible to figure out when and why a particular change was made and by who. There was a mixing of logic and data throughout the codebase – for example, a module meant to generically configure graphics cards across the center had hard-coded exceptions for specific nodes in Bridges

PEARC '23, July 23–27, 2023, Portland, OR

ACM ISBN 978-1-4503-9985-2/23/07…$15.00
https://doi.org/10.1145/3569951.3597589

based on their hostname. Occasionally this made changing a configuration option feel more like an archaeological dig or a game of Jenga than editing a line in a text file. Finally, the Bridges codebase had not been updated for new revisions of Puppet and CentOS. To continue using the existing codebase, changes would have to be made throughout to update from the end-of-life Puppet 4 to the supported Puppet 6 and to find substitutes for abandoned or deprecated options in CentOS/RHEL 8. Faced with a decision between updating a messy codebase with a patchy history or starting from scratch, we opted for a clean slate.

2.3 Monitoring

Bridges relied on Naemon, an open-source fork of Nagios, for the bulk of its monitoring[14, 33]. System metrics were collected, via a separate system, into an InfluxDB[13] instance. Display and graphing duties were handled by Grafana[12]. Alerts were sent via email. This system was adequate, but for Bridges-2 a more unified and streamlined solution was desired, incorporating modern notification methods such as smartphone push alerts.

3 NEW INFRASTRUCTURE

3.1 Provisioning

Due to PSC's involvement in the OpenHPC[17] project, we decided to limit our evaluation of provisioning systems to the two available as part of OpenHPC at the time: xCAT[36] and Warewulf 3[35]. Between the two, Warewulf was a better fit for the way the operations team worked – being built from common open-source tools and familiar system administration concepts, it was easier to wrap our head around each component and understand what was going on behind the scenes. However, as stated above, Bridges-2 is not a homogeneous system. Warewulf works great out-of-the-box for a system built on mostly-identical nodes, but there are subtle variations in initial configurations in the various components of Bridges-2 that Warewulf's approach did not make trivially easy to manage. While Warewulf certainly supports different OS versions, filesystem layouts, and networking configurations across nodes, managing and updating the overlapping requirements for various groups of nodes within Bridges-2 with Warewulf's database-driven command-line interface proved to be more complex and less robust than desired.

To manage this complexity and add additional functionality missing in Warewulf, we developed a wrapper around it called Lycanthrope, so named because it's literally just a more fancy and pretentious way to say "werewulf". Lycanthrope stores the default configuration for the cluster, as well as exceptions and modifications to the various configuration parameters, in a YAML file. This allows us to track the basic configuration of our cluster in version control, modify every parameter fed into Warewulf for any system in our cluster, and easily re-provision dissimilar nodes simultaneously with one command. Lycanthrope integrates with several different types of baseboard management controller using the Redfish REST API standard to manage chassis power and boot device order. Later development added the ability to pull networking and system configuration information from our Netbox datacenter infrastructure management platform, discussed below in section 4.1.

3.2 Configuration management

We started our ground-up rewrite of our configuration management codebase by identifying both the useful parts of the old configuration, as well as the pain points. While the existing codebase contained a great deal of valuable information about how things were configured, there were several recurring complaints, mostly centered around organization and discoverability, as well as the lack of enforcement of version control. To deal with this, our rewrite set out to adopt and enforce Puppet best practices. The organizational issues were tackled by adopting the role-profile pattern, as suggested by the Puppet documentation[23]. In this paradigm, nodes are assigned to one and only one role. Each role is a list of profiles. Each profile configures a set of Puppet modules and resources in a particular way. We built our profiles and custom modules with an eye towards their potential use with other systems and clusters at PSC. Custom Puppet modules and generic, parameterized common profiles contain the logic of how to configure something – the proper way to format a configuration file, which services to set up for an application, and so on. Cluster-specific profiles contain specifics about what we want to configure the systems to do, such as what servers to authenticate against or which mail server to use. This allows us to set things up the same way across all our systems while respecting the specific requirements of each system and each cluster.

The other major new practice we adopted was managing our Puppet codebase through r10k[26]. R10k works with the Git[10] version control system to make deploying our codebase to the Puppet catalog servers more manageable. It automates the process of fetching Puppet modules from both our internal Git repositories and the public Puppet Forge[24] service, as well as the creation of self-contained configuration environments. R10k looks at a special Git repository, called a control repo, that contains the configuration information for the cluster. Each branch of the control repo is turned into a Puppet environment, with a specific version of the configuration data and modules. This allows us to test new changes and additions to our configurations without affecting the systems in production. It also allows us to keep Puppet code in sync across more than one Puppet server, enabling us to have a more robust active-active pair of Puppet servers compared to the single server used by the original Bridges.

In addition to these major changes to how Puppet gets used at PSC, we took the opportunity to adopt a few smaller changes in our day-to-day use of Puppet.

- The eYAML[9] public-key encryption system enables us to securely store sensitive information and secrets in version control alongside the rest of our Puppet configuration
- Basic unit testing using the RSpec[28] framework and linting using Rubocop[29] helps us ensure changes don't break existing code and that a uniform code style is maintained
- Linting and unit tests are run on every commit pushed to our internal Gitlab[11] server by Gitlab-CI
- Technical documentation for Puppet modules and classes is automatically extracted from code using Puppet Strings[25]

3.3 Monitoring

By migrating our monitoring system to Prometheus[21] and Alertmanager[1], we gained the ability to collect metrics about every aspect of Bridges-2, and have a single notification flow for a variety of system events. Prometheus's architecture and straightforward, text-based metric format makes it fairly easy to collect data from nearly any source of data on the system, regardless of whether or not it supports Prometheus natively. We continued to use Grafana to graph data and generate user-facing dashboards, as well as InfluxDB for long-term storage of metrics. With Puppet profiles, we place every node into one or more target groups (such as "compute", "login", "admin") and target specific tests to a specific group.

4 EVOLUTION SINCE INITIAL DEPLOYMENT & FUTURE PLANS

4.1 Integration of a single-source-of-truth for hardware information

After the initial setup of Bridges-2, PSC as a whole started to adopt the Netbox[15] datacenter inventory management and IP address management system to track hardware and IP allocations across the center. Netbox contains information about the past and current hardware and networking configurations of every node in Bridges-2, along with links to external data sources for that node such as Puppet reports. Networking information is pulled by our Puppet servers and provisioners to streamline re-provisioning and redeployment of nodes taken out for repair. Future goals for Netbox at PSC include integration with our monitoring system, so that every alert generated by a system is associated and logged for that system, along with automatic addition of information on virtual machines added to our Bridges-2 oVirt[20] cluster.

4.2 Regression test suite for systems

Along with the unit tests in our configuration codebase, we've started to adopt the ReFrame[27] HPC regression testing framework to test the functionality of the cluster itself. Currently limited to spot-checks for filesystem functionality, user login success, and job submission after maintenance & emergency downtimes, we're working on expanding its use to the test suites of the various software packages provided to researchers using Bridges-2.

4.3 Improvements to monitoring

One of the longer-term goals for adopting a new monitoring framework was easier integration into notification systems and other tools. Naemon was configured to only send alerts to email addresses. Alertmanager is much more flexible, sending alerts to email, Slack[31], or the Opsgenie[19] alerting service depending on the severity of the issue and the group tasked with responding. This has enabled a much more rapid and comprehensive response to outages and service degradation. Future avenues we're exploring for expanding our monitoring program include a public, automated status page for quickly and easily informing users and other interested parties of any issues affecting their use of the cluster.

4.4 Deployment outside Bridges-2

The infrastructure template created for Bridges-2 is now in use throughout the center. Two clusters developed in cooperation with Carnegie Mellon University, Vera[34] and TRACE[32], have been brought up following this template. Two pre-existing clusters, the Human BioMolecular Atlas Program[8] and Brain Image Library[3] projects, are in the process of being migrated to a shared, center-wide infrastructure based on the Bridges-2 template. PSC's own internal systems, used to support everything from our DNS systems to staff frontends, are also moving to take advantage of the center-wide monitoring and configuration management infrastructure.

5 CONCLUSION

The infrastructure developed for Bridges-2 evolved from the suite of technologies used to provision and maintain the original Bridges cluster. Bridges laid down an important foundation for future cluster deployment at PSC, and Bridges-2 by and large built on that foundation. Over the past two years, the combination of Warewulf and Lycanthrope for provisioning, Git-driven Puppet for configuration management, Prometheus and Alertmanager for monitoring, and Netbox for information tracking has demonstrated both the stability required to keep the system running smoothly, as well as the flexibility to adapt to new situations outside its original intended application.

ACKNOWLEDGMENTS

Thanks to Edward Hanna, Martin Gesheff, Brian Johanson, and Derek Simmel for providing feedback on this document. The Bridges system, including Bridges-AI, is supported by NSF award number 1445606. The Bridges-2 system is supported by NSF award number 1928147. The Neocortex system is supported by NSF award number 2005597. The Brain Image Library system is supported by Federal Award Identification Number R24MH114793. The HuBMAP system is supported by NIH Award OT2OD033759.

REFERENCES

[1] Prometheus Authors 2023. *Alertmanager Documentation.* Prometheus Authors. Retrieved April 20, 2023 from https://prometheus.io/docs/alerting/latest/alertmanager/
[2] Red Hat, Inc. 2023. *Ansible website.* Red Hat, Inc. Retrieved April 20, 2023 from https://www.ansible.com/
[3] Kathy Benninger, Greg Hood, Derek Simmel, Luke Tuite, Arthur Wetzel, Alexander Ropelewski, Simon Watkins, Alan Watson, and Marcel Bruchez. 2020. Cyberinfrastructure of a Multi-Petabyte Microscopy Resource for Neuroscience Research. In *Practice and Experience in Advanced Research Computing.* ACM, New York, NY, USA, 1–7. https://doi.org/10.1145/3311790.3396653
[4] Shawn T. Brown, Paola Buitrago, Edward Hanna, Sergiu Sanielevici, Robin Scibek, and Nicholas A. Nystrom. 2021. Bridges-2: A Platform for Rapidly-Evolving and Data Intensive Research. In *Practice and Experience in Advanced Research Computing.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3437359.3465593
[5] Paola A. Buitrago and Nicholas A. Nystrom. 2021. Neocortex and Bridges-2: A High Performance AI+HPC Ecosystem for Science, Discovery, and Societal Good. In *Communications in Computer and Information Science.* Springer International Publishing, Cham, 205–219. https://doi.org/10.1007/978-3-030-68035-0_15
[6] Paola A. Buitrago, Nicholas A. Nystrom, Rajarsi Gupta, and Joel Saltz. 2020. Delivering Scalable Deep Learning to Research with Bridges-AI. In *Communications in Computer and Information Science.* Springer International Publishing, Cham, 200–214. https://doi.org/10.1007/978-3-030-41005-6_14
[7] Progress Software Co. 2023. *Chef website.* Progress Software Co. Retrieved April 20, 2023 from https://www.chef.io/
[8] HuBMAP Consortium. 2019. The human body at cellular resolution: the NIH Human Biomolecular Atlas Program. *Nature* 574, 7777 (Oct. 2019), 187–192.

https://doi.org/10.1038/s41586-019-1629-x
[9] Perforce Software, Inc. 2023. *eYAML source code*. Perforce Software, Inc. Retrieved April 20, 2023 from https://github.com/voxpupuli/hiera-eyaml/
[10] Git Project 2023. *Git website*. Git Project. Retrieved April 20, 2023 from https://git-scm.com/
[11] GitLab B.V. 2023. *GitLab website*. GitLab B.V. Retrieved April 20, 2023 from https://about.gitlab.com/
[12] Grafana Labs 2023. *Grafana website*. Grafana Labs. Retrieved April 20, 2023 from https://grafana.com/
[13] InfluxData Inc. 2023. *InfluxDB website*. InfluxData Inc. Retrieved April 20, 2023 from https://www.influxdata.com/
[14] Naemon project 2023. *Naemon website*. Naemon project. Retrieved April 20, 2023 from https://www.naemon.io/
[15] Netbox Project 2023. *Netbox website*. Netbox Project. Retrieved April 20, 2023 from https://netbox.dev/
[16] Nicholas A. Nystrom, Michael J. Levine, Ralph Z. Roskies, and J. Ray Scott. 2015. Bridges. In *Proceedings of the 2015 XSEDE Conference on Scientific Advancements Enabled by Enhanced Cyberinfrastructure - XSEDE '15*. ACM Press, New York, NY, USA, Article 30, 8 pages. https://doi.org/10.1145/2792745.2792775
[17] OpenHPC Project 2023. *OpenHPC website*. OpenHPC Project. Retrieved April 20, 2023 from https://openhpc.community/
[18] OpenInfra Foundation 2023. *OpenStack website*. OpenInfra Foundation. Retrieved April 20, 2023 from https://www.openstack.org/
[19] Atlassian 2023. *Opsgenie website*. Atlassian. Retrieved April 20, 2023 from https://www.atlassian.com/software/opsgenie
[20] oVirt project 2023. *oVirt website*. oVirt project. Retrieved April 20, 2023 from https://www.ovirt.org/
[21] Prometheus Authors 2023. *Prometheus website*. Prometheus Authors. Retrieved April 20, 2023 from https://prometheus.io/
[22] Perforce Software, Inc. 2023. *Puppet website*. Perforce Software, Inc. Retrieved April 20, 2023 from https://www.puppet.com/
[23] Perforce Software, Inc. 2023. *Puppet documentation: Designing system configs (roles and profiles)*. Perforce Software, Inc. Retrieved June 9, 2023 from https://www.puppet.com/docs/puppet/7/designing_system_configs_roles_and_profiles.html
[24] Perforce Software, Inc. 2023. *Puppet Forge website*. Perforce Software, Inc. Retrieved April 20, 2023 from https://forge.puppet.com/
[25] Perforce Software, Inc. 2023. *Puppet Strings source*. Perforce Software, Inc. Retrieved April 20, 2023 from https://github.com/puppetlabs/puppet-strings
[26] Perforce Software, Inc. 2023. *r10k source code*. Perforce Software, Inc. Retrieved April 20, 2023 from https://github.com/puppetlabs/r10k
[27] Swiss National Supercomputing Centre 2023. *ReFrame documentation*. Swiss National Supercomputing Centre. Retrieved April 20, 2023 from https://reframe-hpc.readthedocs.io/en/stable/
[28] RSpec Project 2023. *RSpec website*. RSpec Project. Retrieved April 20, 2023 from https://rspec.info/
[29] Rubocop Project 2023. *Rubocop website*. Rubocop Project. Retrieved April 20, 2023 from https://rubocop.org/
[30] VMWare, Inc. 2023. *Salt website*. VMWare, Inc. Retrieved April 20, 2023 from https://saltproject.io/
[31] Slack Technologies, LLC. 2023. *Slack website*. Slack Technologies, LLC. Retrieved April 20, 2023 from https://slack.com
[32] College of Engineering, Carnegie Mellon University 2023. *Tartan Research Advanced Computing Environment (TRACE) website*. College of Engineering, Carnegie Mellon University. Retrieved April 20, 2023 from https://www.cmu.edu/engineering/trace/
[33] Richard Underwood. 2017. Building Bridges. In *Proceedings of the Practice and Experience in Advanced Research Computing 2017 on Sustainability, Success and Impact*. ACM, New York, NY, USA, Article 5, 5 pages. https://doi.org/10.1145/3093338.3093339
[34] McWilliams Center for Cosmology, Carnegie Mellon University 2023. *Vera documentation*. McWilliams Center for Cosmology, Carnegie Mellon University. Retrieved April 20, 2023 from https://vera-doc.psc.edu/
[35] Warewulf Project 2023. *Warewulf 3 website*. Warewulf Project. Retrieved April 20, 2023 from https://warewulf.lbl.gov/
[36] xCAT Community 2023. *xCAT website*. xCAT Community. Retrieved April 20, 2023 from https://xcat.org/