SLATE and the Mobility of Capability

Robert Gardner*, University of Chicago; Joseph Breen, University of Utah; Lincoln Bryant, University of Chicago; and Shawn McKee, University of Michigan
**Corresponding author address:* Computation Institute, University of Chicago, Chicago, IL, 60637, USA; email: rwg@uchicago.edu

Abstract: *SLATE (Services Layer at the Edge) is a new project that, when complete, will implement "cyberinfrastructure as code" by augmenting the canonical Science DMZ pattern with a generic, programmable, secure and trusted underlayment platform. This platform will host advanced container-centric services needed for higher-level capabilities such as data transfer nodes, software and data caches, workflow services and science gateway components. SLATE will use best-of-breed data center virtualization components, and where available, software defined networking, to enable distributed automation of deployment and service lifecycle management tasks by domain experts. As such it will simplify creation of scalable platforms that connect research teams, institutions and resources to accelerate science while reducing operational costs and development cycle times. Since SLATE will be designed to require only commodity components for its functional layers, its potential for building distributed systems should extend across all data center types and scales, thus enabling creation of ubiquitous, science-driven cyberinfrastructure. By providing automation and programmatic interfaces to distributed HPC back-ends and other cyberinfrastructure resources, SLATE will amplify the reach of science gateways and therefore the domain communities they support.*

1. Introduction

Much of science today is propelled by multi-institutional research collaborations that require platforms connecting experiment facilities, computational resources, and data distributed among laboratories, research computing centers, and in some cases commercial cloud providers. The scale of the data and complexity of the science drive this diversity. In this context, research computing teams strive to empower their universities with emergent technologies which bring new and more powerful computational and data capabilities that foster multi-campus and multi-domain collaborations to accelerate research. Recently, many institutions invested in their campus network infrastructure with these goals in mind. Yet even for the most advanced, well-staffed, and well-equipped campus research computing centers the task is daunting. Acquiring and maintaining the full scope of cyber-engineering expertise necessary to meet the complex and expanding demands of data and computationally driven science is too costly and does not scale to the full spectrum of science disciplines. The diversity of computation, data and research modalities all but ensures that scientists spend more time on computation and data management related tasks than on their domain science while research computing staff spend more time integrating domain specific software stacks with limited applicability and sustainability beyond the immediate communities served. Capabilities elsewhere are not available locally, and vice versa. How should campus and HPC resource providers evolve their cyberinfrastructure to more easily incorporate data infrastructure building blocks developed in other contexts?

2. Approach

We will address this challenge in complexity and scaling by introducing a **S**ervices **L**ayer **A**t **T**he data center "**E**dge" (SLATE) which will enable distributed automation, centralized delivery and operation of data, software, gateway and workflow infrastructure. Much as Google re-imagined the data center [1] and initiated a wave of data center virtualization development, we view advanced

Proceedings of Gateways 2017, University of Michigan, Ann Arbor, MI; October 23-25, 2017. https://gateways2017.figshare.com/.

"cyberinfrastructure as code" as an appropriate metaphor for transforming the way advanced science platforms are built and operated. SLATE will augment the Science DMZ pattern [2] by adding a secure and trusted "underlayment" platform to host advanced data and software services needed for higher-level, connective functions between data centers. These connective services could include, for example, a domain specific content delivery network endpoint, a collaboration data cache, a job (workflow) scheduler, a service component of a distributed science gateway, an http-based software cache, or a resource discovery service (for higher level, meta scheduling systems).

A major focus of SLATE will be development of community accepted Science DMZ patterns capable of supporting advanced and emerging technologies. In this project, we are focusing on production services for data mobility, management, and access as driven by data-intensive science domains. For operators of distributed data management services, content delivery networks, and science gateways SLATE will closely resemble the NIST definition [3] of a PaaS (Platform-as-a-Service) though the underlying infrastructure is not limited to a cloud context. We are leveraging experience and lessons learned in the deployment and operation of data systems linking over 60 data centers for the LHC computing grid. Figure 1 gives a schematic of the SLATE concept in the local HPC context.

The SLATE concept should accommodate large, well-equipped HPC centers as well as research computing facilities at institutions with fewer resources as well as commercial cloud providers. Modern HPC centers which support data intensive science typically have a Science DMZ which hosts dedicated data transfer nodes, perfSONAR [4, 5] measurement hosts, and security and enforcement policies needed for high performance, wide area applications. A dedicated SLATE edge platform will augment the existing Science DMZ by offering a platform capable of hosting advanced, centrally managed research computing edge services including, where the local infrastructure permits it, the deployment of virtual circuits and other software defined networking constructs. For a small institution with limited resources, the SLATE concept may provide a complete Science DMZ infrastructure to more quickly integrate these institutions into centrally

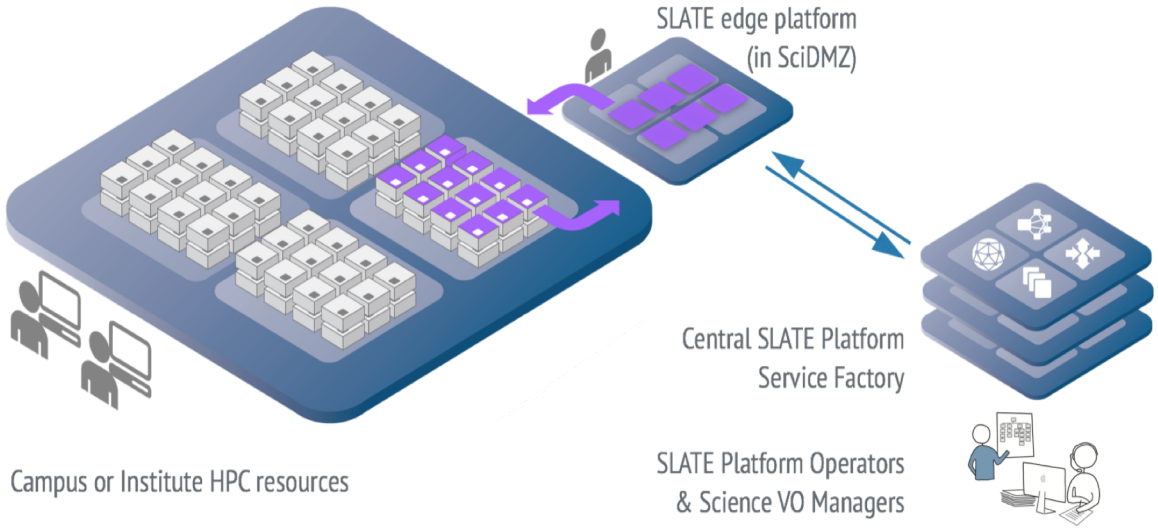


Fig 1: A SLATE edge platform within a campus Science DMZ hosts trusted services operated by a central team which might be operating a network of such services across several campuses. Science "app" developers interact with the SLATE platform service factory to define and launch elements of a science gateway, data cache, or local workflow service.

managed research platforms. The SLATE concept will allow local resource administrators the ability to simply install the infrastructure while offering central research groups the services needed for management of software and science tools used on the platform. Thus, a local researcher in a small institution could focus on the science and connecting any requisite science instrumentation, while the local IT staff would not have the burden of trying to understand the science requirements, application software dependencies, data and workflow particulars, etc. A good science use-case comes from a medium sized collaboration to detect dark matter.

3. A Platform for Dark Matter Searches

Observations of the cosmic microwave background fluctuation, large-scale galaxy surveys, and studies of large-scale structure formation indicate that a large fraction of the matter in the universe is not visible. An exotic but as yet undiscovered elementary particle could explain these observations. Several experiments have been built in last two decades to prove the existence of such elusive particles but their detection has proved challenging as we do not have yet a clear picture of what they are and if they really exist.

The XENON1T experiment, a two-phase xenon Time Projection Chamber has been built in the Laboratori Nazionali del Gran Sasso (LNGS) in Italy to study fundamental questions about the existence and make-up of dark matter. Commissioning began during the first few months of 2016, with the first large scale science run beginning in December 2016. Thanks to its one ton fiducial mass and ultra-low background, the XENON1T experiment will soon begin probing properties of dark matter in yet unexplored regions.

A data processing and analysis hub is hosted by the UChicago Research Computing Center (RCC) to complement processing and analysis facilities in Europe (Stockholm is the European analysis hub). A distributed data management service for the collaboration has been built so that experiment and simulation data sets at various stages of processing can be distributed and shared easily throughout the 22 member institutes. The XENON1T Rucio service [6] is deployed in a network of five storage endpoints (GridFTP-based data transfer nodes) in Europe and the U.S providing a highly scalable global namespace, a reliable and fast transfer service, a subscription ("placement rules") service, and a deletion service for data lifecycle management. The raw experimental data are uploaded at LNGS and automatically replicated to specific HPC centers for processing, and data products from those systems are registered into the system for distribution to the analysis hubs. The Rucio client tools provide a uniform data access model for end-user physicists.

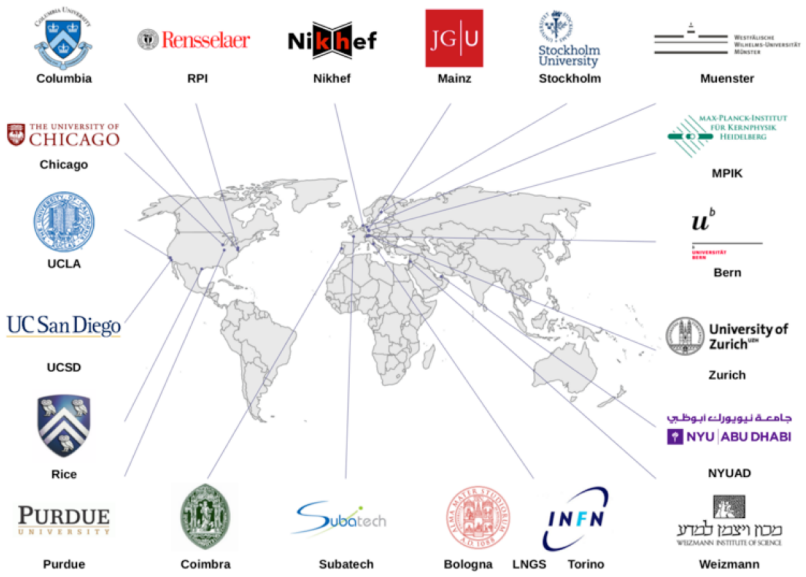


Fig. 2: The XENON collaboration which requires a data management and processing platform that stretches across the resources of its member institutes.

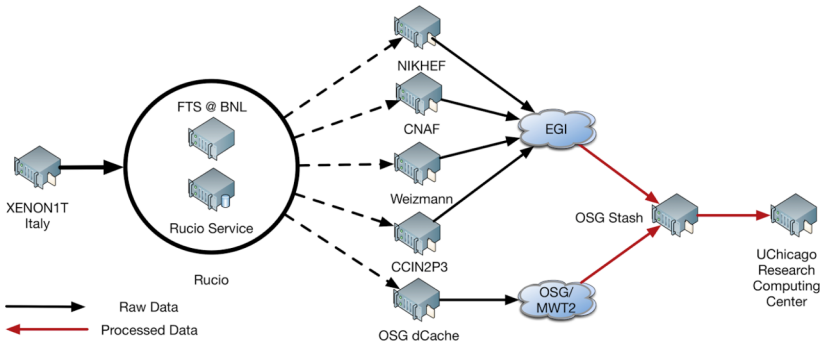


Fig 3: The multi-institution data management platform for the XENON collaboration. A central server for managed file transfer is hosted at Brookhaven National Laboratory, and central file and dataset catalogs and service agents are hosted by the University of Chicago. At each of the storage endpoints is a GridFTP service. Each of these services is managed individually so that the platform itself requires efforts from 10 individual administrators. With a SLATE hosted platform, the services, configuration, monitoring and optimization could be managed by a single operator, requiring only basic server management at the endpoints.

A major obstacle has been preparation of the storage endpoints in the system which required expertise and effort at each site. This preparation is where a SLATE platform would have shined and reduced a several months-long deployment and validation process into a week or two: pre-configured, containerized GridFTP servers customized with checksum calculation, and system registration plugin modules could have been deployed by the central data manager rather than by local systems administrators. In the future, SLATE will streamline updates to the system including Rucio client software and deployment of monitoring sensors for system-wide analytics, providing capabilities where needed.

4. Planned Components and Capabilities

SLATE will leverage advances made by previous testbeds [7-10] and similar platforms [11, 12] and will integrate best-of-breed data center virtualization and service orchestration technologies from the open source community. With the explosion of IaaS (Infrastructure as a Service) [13] offered through data center virtualization software, container orchestration engines, etc., the technical paths and best practices are still being discovered and invented at a rapid pace. We expect over the lifetime of this project to see advances in infrastructure delivery and advances at all layers in the service stack. Our challenge then is to remain flexible enough to shift direction when the advantages are clear.

The system architecture diagram of Figure 4 gives a schematic picture of the SLATE system in our vision. The SLATE Platform Factory will provide views for science data operators as well as for operators of the underlying infrastructure at a local institution. Health checking and alarming systems will be implemented at all critical points in the platform. The platform will bring distributed services from multiple data centers logically into one system, and like nodes in a virtual data center they can be scheduled, monitored, terminated, etc. The SLATE platform software we develop would be open source, and independent groups would be able to use it to build new forms of advanced cyberinfrastructure for their communities. As an example, we will use SLATE for widespread deployment of a caching service like the XRootD [14] system. The SLATE team will maintain a local integration testbed and validation service for site configurations. Containerized services will be curated by the central group and be validated for functionality, performance and security. Data services at the site can manage data of a semi-persistent type or purely ephemeral (caching) type. Other obvious edge services would be a Globus Connect [15] service or an HTTP web caching service.

The SLATE provisioning service, while firstly targeting the (bare metal) SLATE edge nodes, will have the ability to handle multiple infrastructure types, including various public and private cloud providers. In this context, we will not re-invent the wheel but will leverage the plethora of provisioning tools for cloud infrastructure,

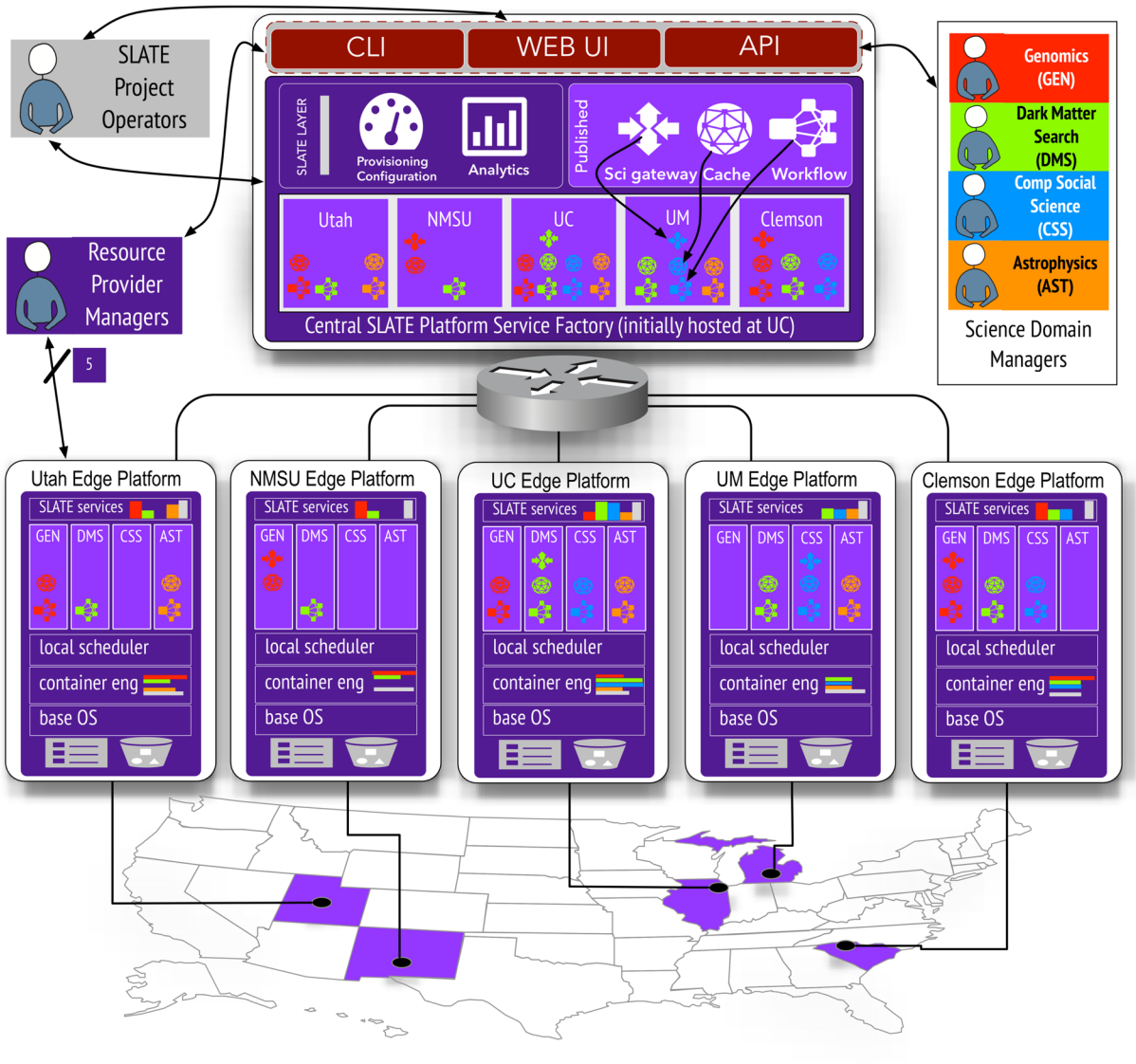


Fig 4: Overview of the planned SLATE edge management system which allows science domain managers to implement building block services such as data management systems or caching networks. A SLATE Platform Service Factory provides global provisioning, service orchestration, monitoring and analytics functions for deployed applications. Shown are three different kinds of users: the SLATE Project Operators, Resource Provider Managers and Science Domain Managers. Science Domain Managers use SLATE to define their applications and science workflow. Site Resource Managers use SLATE to register, provision and enable access to their sites as well as set up policies and quota for their supported science virtual organizations. The SLATE Project Operators use the central services to monitor and maintain all SLATE services at all sites.

including, potentially, Terraform [16], Apache libCloud [17], etc. when appropriate. The container engine will likely be based on Docker [18] though we have experimented with LXD [19] and found it to be potentially better suited to the task. The container scheduler service options are many, though Kubernetes [20, 21] and Mesos [22] are leading candidates.

By integrating best-of-breed tools in wide-scale use in the cloud-native community, we minimize software development while providing sustainable flexibility. In summary, the components of SLATE

platform are expected to consist of the following:

- Controller/console for globally distributing "applications", the main interface for the SLATE controller platform or directly to an edge platform node
- Discovery service (local and multi-data center) and a scheduling service (e.g. Mesos or Kubernetes)
- Automated core provisioning service (e.g. Ansible [23], Puppet [24])
- System monitoring and log aggregation (e.g. Check_mk [25], Elastic Stack [26], OSSIM [27])

1.1 SLATE Platform Node

A SLATE Platform Node (SPN) will be an appliance-like device that sits within the Science DMZ to provide a variety of local services to an institution (Figure 5). The operating system for SPNs will likely be based on CoreOS [28], with additions and modifications as-needed. The base OS will only run the container engine, SLATE-specific orchestration and authentication services, and necessary software to communicate to the baseboard management controller. Updates to the base OS image will be published and GPG-signed on a regular basis.

Container services. The first container type will be the "system" container, maintained by the SLATE team, which will provide a number of back-end services for the platform. As a matter of principle, all system services that are feasible to be containerized will be, so as to keep the base operating system as lightweight as possible. Some example system containers include:

- **API**: Provides an interface for both command-line and web-based tools to schedule, start, and stop containers on the service.
- **Monitoring**: Consumes data from services, including performance statistics, logs, errors, etc. and publishes to the central operations team and the local site administrator.
- **Admin UI**: A web-based interface for system operators to manage the local SPN and allow local site administrators to view performance metrics and logs, kill containers, reboot the appliance, etc.

The second, more interesting container type supported by the SPN will be the "application container". Application containers will be curated by a team of operators.

Remote Administration: The SPN base OS will have software installed to communicate with the SPN baseboard management controller via IPMI or vendor-specific tools (e.g., Dell OpenManage). The central operations team will be able to monitor for failed disks, fans, or other hardware, source replacements and send them to the site, and report this to the local site administrator for servicing.

Image Updater: As the platform grows, or as vulnerabilities, bugs, or other deficiencies appear, it will be necessary to patch the base operating system from time to time. The SPN will have a daemon for polling for system updates from the central operations platform, and flag itself for a reboot once the new OS image has been downloaded and verified. OS images will be downloaded from a secure (HTTPS) server and be GPG signed.

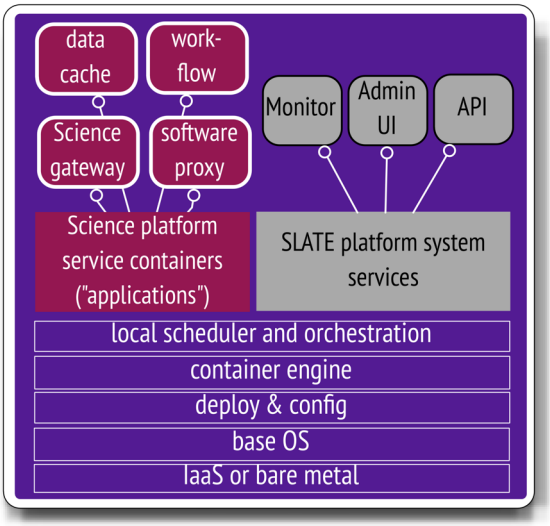


Fig 5: The planned SLATE service architecture on an edge platform node.

Disk Management: SPNs in our first testbed will come with a reasonably large disk array (48TB in the "A" model, 32TB in the "B" model) with a variety of science users, so data management is a necessary feature of the platform. Science domain users of the SLATE platform will need to declare the disk utilization and I/O requirements of their application ahead of time, or run their application through the SLATE integration and test framework

to determine these values. Once determined, the application can be deployed onto a SLATE node at an institution. SLATE nodes will split their storage between *ephemeral storage* - used opportunistically by domain users with a fair-share policy and fast expiry time, and *persistent storage* - space requested specifically by the science domain user or gateway developer.

perfSONAR: All sites will be equipped with two perfSONAR nodes, each capable of bandwidth and latency measurements using two of the on-board NICs.

secure access to only system managers and SLATE personnel. For the data plane, the platform host or the switch will require one or more data plane connections, as negotiated with campus administrators. The data plane connection(s) will, ideally, exist on a campus programmable Science DMZ infrastructure. For small campuses, this connection may be a simple dedicated connection that does not reside behind the campus firewall. For large sites, the SLATE switch allows one or more uplinks for the data plane and many access ports for scaling additional nodes.

Table 1: Example SLATE controller command line interface commands and function.

Command	Function
`$ slate deploy --site B [service ID]`	Deploy service ID to site B
`$ slate start / stop --site B [service ID]`	Selectively control services
`$ slate diag —-global [service ID]`	Retrieve system-wide diagnostic reports for a service
`$ slate update --global [service ID]`	Update a configuration globally
`$ slate validate --group VO --cloud US,IT --all`	Validate the all services on behalf of a virtual organization over US and Italy community clouds
`$ slate replicate —-src A —-dest B,C [service ID]`	Copy a running service and data from site A to sites B and C
`$ slate migrate —-src A —-dest B [service ID]`	Move a running service and data from site A to site B

Network: The networking component of the SLATE node will work with existing campus networks and campus programmable Science DMZ infrastructure to provide a layer 2 and/or layer 3 substrate for migrating large data sets between resources. Two versions of the SLATE node will be available, a “large site” node and a “small site” node. The large SLATE node will provide an OpenFlow capable switch capable of 10Gb/s or greater switching speeds. The small SLATE node will provide a single host with Open Virtual Switch that will allow connectivity at 1Gb/s and 10Gb/s. For the implementation, each campus will be able to decide which style node it can support. The campus will provide one management connection for the platform host and one management connection for each of the perfSONAR nodes. Management connections will be specially routed via SDN to

The networking component of the SLATE node will provide OpenFlow [29] capabilities through the provided switch or through Open Virtual Switch (OVS). To effectively utilize the OpenFlow capabilities to interconnect to partner institutions, the campus will need to provision a path to Internet2’s AL2S across its regional. Many campuses have gone through this exercise previously in order to support NSF funded GENI racks [7, 10]. Implementations for GENI racks have utilized dedicated circuits, dedicated VLANs, or campus provided OpenFlow configurations. The SLATE nodes would utilize similar configurations when possible.

1.2 Planned Central Services

SLATE will provide a web-based interface that will be used as a sort of concierge service for

matching resources to science domains. The portal will have different views and options based on the type of user. For example, local site administrators and managers will be able to see the performance of their SLATE appliance on-site, and utilization by science domains. Science domain users will be able to see throughput and disk utilization on a per site basis. Users who aren't logged in will be able to see overall utilization of the SLATE platform and other front-facing niceties.

1.3 SLATE Application Containers

All project software running on the SLATE platform will need to run inside of an application container. This allows great flexibility, as experts in a particular domain can provide an operating system image. All application containers will have to declare a set of resource requirements. These requirements may include disk utilization, minimum throughput (MB/s), firewall ports, memory and CPU. The SLATE platform will be able to compare these resource declarations to the available resources, and provide a candidate list of resource targets for the application. The platform may go further and suggest changes that could be made to the application for greater proliferation.

1.4 SLATE Integration and Test Framework

Before deployment on production SPNs, any applications that wish to run through the SLATE platform will need to be run through an integration and test framework on the SLATE development environment. The integration framework will provide feedback to the application developer, with details about whether or not the application has run successfully on the platform, and any relevant performance metrics.

5. SLATE and Science Gateways

We view SLATE as a potential amplifier for science gateway processing back-ends. The platform will enable distributed automation and centralized delivery and operational controls for data, software, and workflow services. Thus, a distributed "DevOps" model of programming and development will be possible in a way that shortens the development and testing timeline. The SLATE team will partner with the Science Gateways Community Institute (SGCI) to ensure that existing expertise, lessons and preferred operational modalities of science gateway back-ends are leveraged. The SLATE project will collaborate closely with the SGCI to assure that its platform will be added in the Scientific Software Collaborative (SSC).

In summary, we see SLATE offering a means to more easily build science gateway back-ends, therefore streamlining the development process, while the SGCI provides a community for outreach and adoption of SLATE among science communities and university HPC resource providers.

6. Conclusions and Outlook

At the time of this article the SLATE project has just begun. Currently the focus is on establishing a three-site testbed on which to evaluate local container orchestration frameworks, the federation of such service frameworks across multiple sites, and the management of both stateless and stateful services. The project welcomes contributions and participation from groups and individuals focused on building distributed research platforms and gateway systems. Interested readers may follow developments at http://slateci.io/.

7. Acknowledgments

SLATE is funded by NSF CIF21 DIBBs grant award number 1724821. Collaborating institutions are the University of Chicago, the University of Michigan, the University of Utah, New Mexico State University and Clemson University.

8. References

1. Barroso, L.A. and U. Hölzle, *The Datacenter as a Computer: An Introduction to the Design of Warehouse-Scale Machines.* Synthesis Lectures on Computer Architecture, 2009. **4**(1): p. 1-108.
2. Eli Dart, L.R., Brian Tierney, Mary Hester, and Jason Zurawski, *The Science DMZ: a network design pattern for data-intensive science*, in *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*. 2013, ACM: Denver, Colorado. p. 1-10.
3. Fang Liu, J.T., Jian Mao, Robert Bohn, John Messina, Lee Badger, Dawn Leaf *NIST Cloud*

Computing Reference Architecture, . National Institute of Standards and Technology Special Publication, 2011.
4. Hanemann, A., Boote, J. W., Boyd, E. L., Durand, J., Kudarimoti, L., Lapacz, R., Swany, D. M., Zurawski, J., and Trocha, S., *PerfSONAR: A Service Oriented Architecture for Multi-Domain Network Monitoring*, in *Third International Conference on Service Oriented Computing*. 2005, Springer Verlag: Amsterdam, The Netherlands. p. 241-254.
5. *PerfSonar-PS Publications*. 2005-2009; Available from: http://www.perfsonar.net/publications.html.
6. Garonne, V. *Rucio – The next generation of large scale distributed system for ATLAS Data Management*. in *CHEP2013*. 2013.
7. *GENI (Global Environment for Network Innovations), a virtual laboratory for networking and distributed systems research and education*. Available from: https://www.geni.net/.
8. *GENI Network Stitching Architecture*. 2016; Available from: http://groups.geni.net/geni/wiki/GeniNetworkStitching.
9. Ilia Baldine, Y.X., Anirban Mandal, Paul Ruth, Chris Heerman and Jeff Chase, *ExoGENI: A Multi-domain Infrastructure-as-a-Service Testbed*, in *Testbeds and Research Infrastructure. Development of Networks and Communities: 8th International ICST Conference, TridentCom 2012, Thessanoliki, Greece, June 11-13, 2012, Revised Selected Papers*, T. Korakis, M. Zink, and M. Ott, Editors. 2012, Springer Berlin Heidelberg: Berlin, Heidelberg. p. 97-113.
10. Mark Berman, J.C., Lawrence Landweber, Akihiro Nakao, Max Ott, Dipankar Raychaudhuri, Robert Ricci, and Ivan Seskar, *GENI: A federated testbed for innovative network experiments.* Computer Networks, 2014. **61**: p. 5-23.
11. *Cyverse life sciences platform*. Available from: http://www.cyverse.org/about.
12. *The Pacific Research Platform*. Available from: http://prp.ucsd.edu/.
13. Mell, P. and T. Grance, *The NIST Definition of Cloud Computing*. 2011, NIST Special Publication 800-145, National Institute of Standards and Technology.
14. *The XRootD Project*. Available from: http://xrootd.org/.
15. Globus. *Guide for Globus Resource Providers*. 2016; Available from: https://docs.globus.org/resource-provider-guide/.
16. HahsiCorp. *Terraform: Infrastructure as Code*. 2016; Available from: https://www.terraform.io/.
17. *Apache libCloud: One Interface to Rule them All*. 2016; Available from: https://libcloud.apache.org/about.html.
18. *Docker - An open platform for distributed applications for developers and sysadmins*. 2015; Available from: https://www.docker.com/.
19. *The Linux Container Hypervisor*. 2016; Available from: http://www.ubuntu.com/cloud/lxd.
20. Google. *Kubernetes: Production-Grade Container Orchestration*. 2016; Available from: http://kubernetes.io/.
21. Brendan Burns, B.G., David Oppenheimer, Eric Brewer, and John Wilkes, *Borg, Omega, and Kubernetes.* Queue, 2016. **14**(1): p. 70-93.
22. Benjamin Hindman, et al., *Mesos: a platform for fine-grained resource sharing in the data center*, in *Proceedings of the 8th USENIX conference on Networked systems design and implementation*. 2011, USENIX Association: Boston, MA. p. 295-308.
23. Hall, D., *Ansible Configuration Management*. 2013: Packt Publishing. 92.
24. Loope, J., *Managing Infrastructure with Puppet*. 2011: O'Reilly Media, Inc. 46.
25. *Check_mk: The open source IT monitoring solution in the tradition of Nagios*. Available from: http://mathias-kettner.com/check_mk.html.
26. *Elastic Stack*. 2016; Available from: https://www.elastic.co/guide/index.html.
27. *AlienVault OSSIM: The World's Most Widely Used Open Source SIEM*. 2016; Available from: https://www.alienvault.com/products/ossim.
28. *CoreOS: Self-Driving Container Infrastructure*. Available from: https://coreos.com/.
29. N. McKeown, et al., *OpenFlow: Enabling Innovation in Campus Networks.* ACM SIGCOMM Computer Communication Review, 2008. **38**(2): p. 69-74.