Bridging Local Cyberinfrastructure and XSEDE with CyberGIS-Jupyter

[Extended Abstract]

Anand Padmanabhan
Department of Geography and Geographic Information Science
University of Illinois of Urbana-Champaign
apadmana@illinois.edu

Dandong Yin
Department of Geography and Geographic Information Science
University of Illinois of Urbana-Champaign
dyin4@illinois.edu

Fangzheng Lyu
Department of Geography and Geographic Information Science
University of Illinois of Urbana-Champaign
flu8@illinois.edu

Shaowen Wang
Department of Geography and Geographic Information Science
University of Illinois of Urbana-Champaign
shaowen@illinois.edu

INTRODUCTION

The fabric of national and international cyberinfrastructure ecosystems for scientific discovery and innovation can be viewed as distributed computing environments composed of powerful supercomputers, various cloud computing resources, and numerous local cyberinfrastructure (including both cloud and HPC) resources. However, extensive computational work conducted by academic researchers are often siloed in one of these environments. Science Gateways [5, 6], by simplifying access to advanced cyberinfrastructure resources, have made significant progress on connecting these silos by enabling researchers in many fields to access advanced cyberinfrastructure through web browsers. In this context, this research bridges between national and local cyberinfrastructure resources through: (a) horizontal scaling of CyberGIS-Jupyter between the cloud resources provided by JetStream on the Extreme Science and Engineering Discovery Environment (XSEDE) and a VMWare-based cloud environment on Virtual ROGER, a local cyberinfrastructure resource hosted by the CyberGIS Center for Advanced Digital and Spatial Studies at the University of Illinois at Urbana-Champaign campus; and (b) enabling the submission of computationally intensive models to the batch systems of both Virtual ROGER and Comet (an XSEDE resource).

PEARC '19, July 28-August 1, 2019, Chicago, IL, USA

https://doi.org/10.1145/3332186.3333257

ACM ISBN 978-1-4503-7227-5/19/07.

Specifically, we have developed a mechanism to provide access to a scalable JupyterHub platform together with cutting-edge cyberGIS software and hardware [4], called CyberGIS-Jupyter [7], which allows seamless access to high-performance computing (HPC) resources while shielding the complexity of managing cyberinfrastructure access from users. The user-friendly environment provided by CyberGIS-Jupyter along with computational scalability achieved through this research provides a powerful environment for conducting collaborative and reproducible research at scale with seamless access to advanced cyberinfrastructure at both local and national levels. The rest of this paper describes the architecture of our solution and articulates the corresponding implementation.

ARCHITECTURE AND IMPLEMENTATION

The architecture consists of four major layers: (a) user layer; (b) application layer; (c) cloud resources; and (d) HPC resources. The components of each of these layers and the interactions between them can be depicted in Figure 1. There are two levels at which users interact with cyberinfrastructure resources: (1) logging into JupyterHub and accessing their single user Jupyter Notebook server as an interactive session which runs transparently on local and national resources, and (2) submitting computationally intensive jobs that leverage HPC resources locally and on XSEDE.

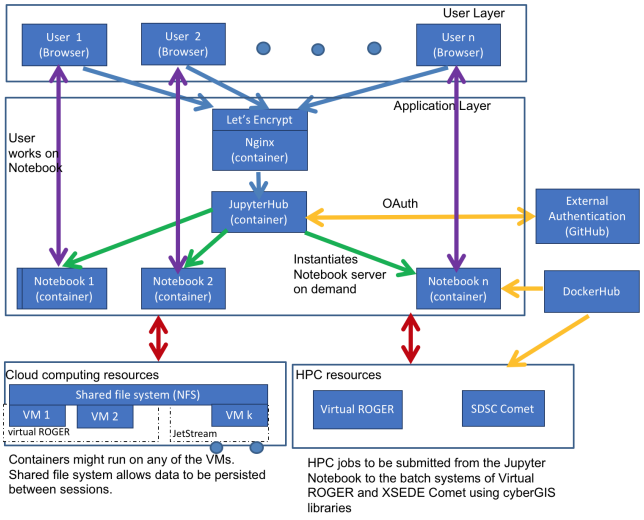

Figure 1: Architecture of Scalable CyberGIS-Jupyter Deployment

Considering the first level interaction, our key innovation here is the horizontal scaling between a local cyberinfrastructure resource (Virtual ROGER) and XSEDE (Jetstream), which was achieved using Docker Swarm [3]. From a user's perspective, the workflow is specified as follows: they first login to the JupyterHub environment through browser, which redirects to GitHub for authentication; if successful, a Jupyter environment is spawned in which they can develop notebooks and write codes. However, the backend implementation to enable this seamless interaction is complex. First, we install docker on all the virtual machines (VMs) to enable Docker Swarm. One or more of the VMs will serves as the swarm manager and the rest of the VMs will join as workers. Then we start a docker service to instantiate JupyterHub. The container image used for this JupyterHub installation comes from our own repository on Docker Hub and contains customization to automatically create user directories and configurations for authentication, network, and storage systems needed to start up a working JupyterHub instance. When a user initiates a login attempt with JupyterHub he/she is redirected to GitHub for authentication using OAuth, on successful completion of which authorization is checked. The load balancing and the certificate for securing https connections are provided by nginx and letsencrypt respectively, which are also run as docker containers. The authorization is based on a white list maintained in our confirmation. Once a user is authorized, a single Jupyter server instance is started up for the user on one of the nodes of the swarm (either on Virtual ROGER or JetStream in our case) and the user begins interaction with the notebook. The container image of the Jupyter Notebook server is automatically downloaded from our Docker Hub repository. Additionally, we need data to be persisted and shared on all the VMs that are part of the Docker Swarm, so that users can see the same version of their data and notebooks between sessions regardless of which cyberinfrastructure resources their notebooks run on. To enable this, we installed and enabled a Network File System (NFS) that is accessible from all the VMs. Before conducting the scaling experiments, we were anticipating a number of problems with networking and firewall issues that would limit our horizontal scaling tests. However, in practice, we had to open up only a single port between the VMs that are part of the Docker Swarm nodes. This port is used for broadcasting communication between the nodes.

The second level of interaction that allows computing jobs to be run on batch systems is enabled by a cyberGIS library that is installed in the Jupyter environment. One of the models we have experimented with is the SUMMA (Structure for Unifying Multiple Modeling Alternative) [2], a hydrological modeling tool built on a common set of conservation equations and a common numerical solver, which together constitute the structural modeling core for enabling a controlled and systematic analysis of alternative modeling options. Using the interface provided by the cyberGIS library, the SUMMA model along with necessary parameters are made available on the worker nodes running computation, and the model itself is run as a Singularity container. Our tests running the model on both Virtual ROGER and Comet through the batch systems were successful.

HORIZONTAL SCALING RESULTS

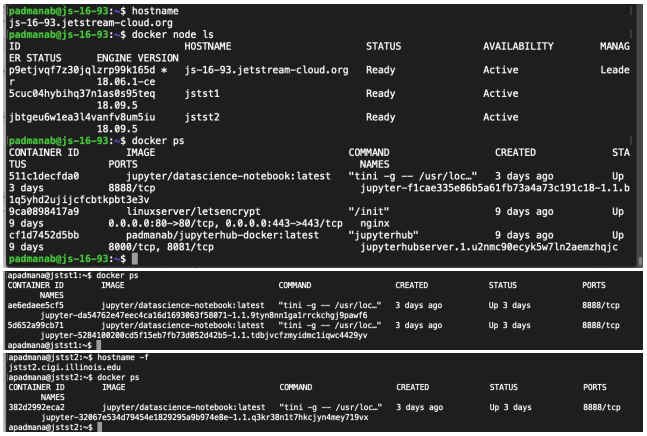

Figure 2: Screenshots showing a small test Docker Swarm environment running across Virtual ROGER and JetStream. It also shows the execution of JupyterHub, nginx, letsencrypt, and associated notebooks of various nodes of Docker Swarm

Figure 2 illustrates a small test installation with 3 virtual machines, one on JetStream and two on Virtual ROGER. Putting them together using Docker Swarm, we were able to seamlessly run Jupyter Notebooks and scale horizontally. We started with two VMs, as more users started notebook servers, the existing resources on these two VMs failed to spawn containers for new users due to the lack of resources. To alleviate this, we started a third server and added it to the Docker Swarm. After the operation, new users' Jupyter servers were immediately deployed. This represents a powerful capability we plan to make available to CyberGIS-Jupyter users and will help with community training events. For example, we will deploy this solution for a national summer school on

cyberGIS and geospatial data science [1], so we can easily serve around 40-50 active users using CyberGIS-Jupyter for learning.

CONCLUDING DISCUSSION

We successfully demonstrated the capability to bridge a local high-performance computing environment with national cyberinfrastructure in two cases (a) horizontal scaling of JupyterHub between Virtual ROGER and JetStream, and (b) successful HPC job submission to batch systems on Virtual ROGER and XSEDE Comet. This represents a significant step forward for supporting cyberGIS users with their research and education activities. As a next step, we plan to investigate ways to automatically scale the resource usage based on system needs without any intervention from users. Solutions leveraging Kubernetes are available but need to be investigated about their applicability to our approach.

ACKNOWLEDGMENTS

This research is supported in part by the National Science Foundation (NSF) under grant numbers 1443080, 1664119, and 1743184. This work used the Extreme Science and Engineering Discovery Environment (XSEDE), which is supported by National Science Foundation under grant number ACI-1548562. We greatly appreciate the XSEDE allocation support through two awards: SES170023 and EAR190007 and the support provided by the XSEDE Extended Collaborative Support Service (ECSS) program. Our computational work also used Virtual ROGER, which is a cyberGIS supercomputer supported by the CyberGIS Center for Advanced Digital and Spatial Studies and the School of Earth, Society and Environment at the University of Illinois. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of NSF.

REFERENCES

[1] AAG-UCGIS Summer School 2019. (2019, March 20). Retrieved May 21, 2019, from http://cybergis.illinois.edu/event/aag-ucgis-summer-school-2019/

[2] Clark, M. P., Nijssen, B., Lundquist, J. D., Kavetski, D., Rupp, D. E., Woods, R. A., … Others. (2015). The structure for unifying multiple modeling alternatives (SUMMA), Version 1.0: Technical description. NCAR Tech. Note NCAR/TN-5141STR. Retrieved from http://opensky.ucar.edu/islandora/object/technotes%3A526/datastream/PDF/download/citation.pdf

[3] Naik, N. (2016). Building a virtual system of systems using docker swarm in multiple clouds. In 2016 IEEE International Symposium on Systems Engineering (ISSE) (pp. 1–3).

[4] Wang, S. (2010). A CyberGIS Framework for the Synthesis of Cyberinfrastructure, GIS, and Spatial Analysis. Annals of the Association of American Geographers. Association of American Geographers, 100(3), 535–557.

[5] Wang, S., Liu, Y., Wilkins-Diehr, N., & Martin, S. (2009). SimpleGrid toolkit: Enabling geosciences gateways to cyberinfrastructure. Computers & Geosciences, 35(12), 2283–2294.

[6] Wilkins-Diehr, N., Gannon, D., Klimeck, G., Oster, S., & Pamidighantam, S. (2008). TeraGrid Science Gateways and Their Impact on Science. Computer, 41(11), 32–41.

[7] Yin, D., Liu, Y., Hu, H., Terstriep, J., Hong, X., Padmanabhan, A., & Wang, S. (2018). CyberGIS-Jupyter for reproducible and scalable geospatial analytics. Concurrency and Computation: Practice & Experience, 308, e5040.