Extending Tapis Workflow Management Framework with Elastic Google Cloud Distributed System using CloudyCluster by Omnibond

1st Eric Lam
University of Hawaii, Honolulu, HI, USA
lameric@hawaii.edu

2nd Sean Cleveland
University of Hawaii - System, Honolulu, HI, USA
seanbc@hawaii.edu

3rd Cole Mcknight
Omnibond Systems, USA
cole@omnibond.com

4th Boyd Wilson
Omnibond Systems, USA
boyd@omnibond.com

5th Richard Cardone
Texas Advanced Computing Center, Austin, TX, USA
rcardone@tacc.utexas.edu

6th Maytal Dahan
Texas Advanced Computing Center, Austin, TX, USA
maytal@tacc.utexas.edu

7th Joe Stubbs
Texas Advanced Computing Center, Austin, TX, USA
jstubbs@tacc.utexas.edu

8th Gwen A. Jacobs
University of Hawaii - System, Honolulu, HI, USA
gwenh@hawaii.edu

***Abstract*—The goal of a robust cyberinfrastructure (CI) ecosystem is to catalyse discovery and innovation. Tapis does this through offering a sustainable production-quality set of API services to support modern science and engineering research, which increasingly span geographically distributed data centers, instruments, experimental facilities, and a network of national and regional CI. Leveraging frameworks, such as Tapis, enables researchers to accomplish computational and data-intensive research in a secure, scalable, and reproducible way and allows them to focus on their research instead of the technology needed to accomplish it.**

This project aims to enable the integration of the Google Cloud Platform (GCP) and CloudyCluster resources into Tapis-supported science gateways to provide on-demand scaling needed by computational workflows. The new functionality uses Tapis event-driven Abaco Actors and CloudyCluster to create an elastic distributed cloud computing system on demand. This integration allows researchers and science gateways to augment cloud resources on top of existing local and national computing resources.

***Index Terms*—Tapis, CloudyCluster, Google Cloud Platform, Science Gateway**

I. Introduction

Science is a collaborative effort, and thus demand for development environments for researchers to collaborate with little friction has increased significantly. Many of these science gateways leverage advanced cyberinfrastructures that accelerate and scale scientific analysis. Tapis is a science gateway framework that fuels collaboration by providing a way for researchers to manage and share both code and computational resources from multiple sources.

Presented at Gateways 2022, San Diego, USA, October 18–20, 2022. https://zenodo.org/communities/gateways2022/

Previously, researchers were only able to leverage already available resources. With this new functionality enabled through Tapis and CloudyCluster, researchers will be able to create large distributed compute systems on Google Cloud on-demand and register the system with Tapis to submit jobs. Figure 1 depicts the typical Tapis workflow and the CloudyCluster addition.

1) User submits a job to Tapis. The job is a request to the user-registered app which is an image classifier, with an input image file.
2) Tapis deploys the user registered app container.
3) Tapis runs the job on a user-registered system.
4) If requested, Tapis will create and register a Google Cloud distributed system with CloudyCluster that users can run the job on.

The goal of this work was to build a containerized proof-of-concept that enables science gateways to dynamically provision a cloud HPC and launch distributed workflows using the Tapis Framework to stage, submit, monitor and archive computational jobs and corresponding inputs and outputs. This paper presents the approach taken in developing this new capability.

II. Background

A. Tapis

Tapis is an open source, NSF funded RESTful Application Programming Interface (API) platform for distributed computation [1]. It provides production-grade capabilities to

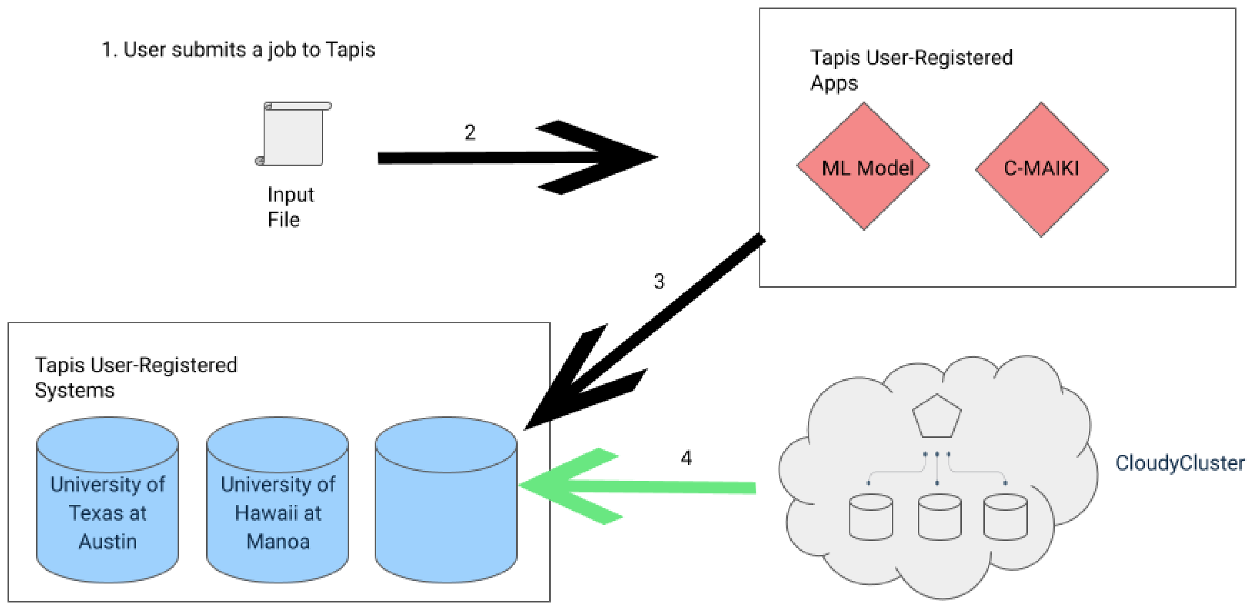
1. User submits a job to Tapis
Input File
2
Tapis User-Registered Apps
ML Model
C-MAIKI
3
Tapis User-Registered Systems
University of Texas at Austin
University of Hawaii at Manoa
4
CloudyCluster

Fig. 1. Current workflow (black) and implemented addition (green).

1) Securely execute workflows that span geographically distributed providers
2) Store and retrieve streaming/sensor data for real-time and batch job processing, with support for temporal and spatial indexes and queries
3) Leverage containerized codes to enable portability.
4) Create reproducible computation with history and provenance tracking built into the API
5) Manage access to data and results through a fine-grained permissions model, so that digital assets can be securely shared with colleagues or the community at large.

The Tapis framework can be leveraged as a hosted solution or distributed between various institutions. A central instance is currently hosted by the Texas Advanced Computing Center (TACC) at the University of Texas at Austin. Other institutions, such as the University of Hawaii (UH), have a hybrid deployment with subsets of Tapis API services deployed locally while leveraging others hosted at TACC. This project leverages Tapis as the science gateway API framework for launching and managing end-to-end computational workflows across distributed systems.

B. *CloudyCluster*

Omnibond designed CloudyCluster to decrease the time and simplify the effort required to create an HPC environment on demand. CloudyCluster quickly provisions compute instances with a wide selection of open HPC applications, dynamically configured for access to storage. All instances within a compute group are based on the same image and the same instance type. Users can create and manage CloudyCluster environments using a self-service, mobile-friendly web interface. A CloudyCluster environment includes:

1) Parallel storage with OrangeFS [2]
2) Schedulers such as SLURM [3]
3) CCQ, a metascheduler that provides job-initiated elastic HPC
4) WebDAV for native file access
5) Compute instance images containing popular Open Source software packages

This project leverages the CloudyCluster platform with the Google Compute Platform (GCP) for provisioning a SLURM based HPC cluster that can elastically provision compute nodes based on job submission parameters.

C. *Automaton*

The Automaton Project is a management tool designed to automate the steps required to dynamically provision a large scale cluster environment in the cloud, execute a set of jobs or a custom workflow, and de-provision the cluster environment in a single operation. [4] The Automaton provisioning process is driven by a single configuration file that defines all of the parameters required to create a fully functional HPC Environment in the Cloud. It is designed to be modular and pluggable to support a variety of HPC Schedulers, environment types, workflows, and cloud providers. This project leverages the Automaton code as part of the provisioning workflow of

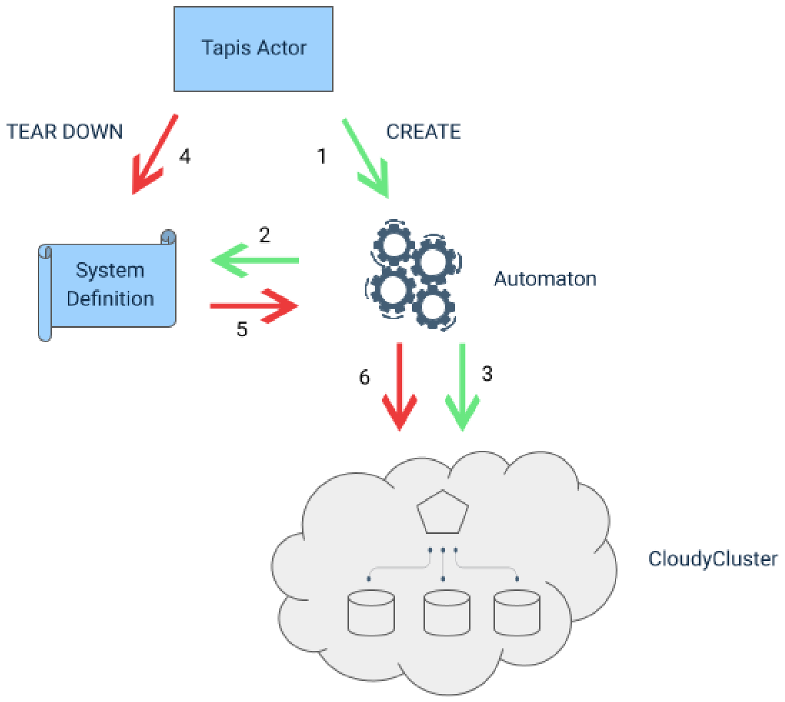

Fig. 2. Steps for cluster creation (green) and deletion (red).

a computational Tapis system in the CloudyCluster Google Compute Platform (GCP) environment to provide an HPC environment for job execution.

III. Approach and Implementation

The approach taken in developing this new interaction between Tapis and CloudyCluster was: 1) Integrate deployment with Tapis Applications API to provision on-demand SLURM clusters on GCP at user-requested scale and register the cluster with Tapis. 2) Develop scripts that allow Tapis to submit computational workloads to the CloudyCluster meta-scheduler, CCQ. 3) Use the new integration for deployment of computational workloads via these new on-demand provisioning scripts with HPC/HTC sample workflows.

A. Integrating GCP CloudyCluster deployment with Tapis

Automaton was used as the main tool for deploying the cluster. To allow this project to use Automaton with Tapis, the Automaton code had to be containerized as there are a number of dependencies and libraries required in order to use Automaton to deploy a cluster. Containerization also allows the Automaton code to be registered as a Tapis Actor object that can then be issued messages to deploy or decommission the clusters.

Automaton creates the cluster according to specifications in a configuration file, which includes information on the initial system to create as well as login credentials for an initial user or service account. A challenge identified in integration was connecting a newly deployed login node instance to Tapis to accept job submission and determining how to persist information of login DNS and environment after initial cluster deployment, as they are required in order to decommission the cluster later. To address this challenge, a python wrapper script was developed in order to issue commands to Automaton and process the output. The wrapper script uses the processed output and registers a new execution system definition with Tapis, storing the DNS and environment name as Tapis system object attributes that are persisted and accessible later.

B. Allowing Tapis to use the CCQ meta-scheduler

Since Tapis currently does not natively support CloudyCluster's meta-scheduler, CCQ, for submitting and monitoring computation jobs, some ”shim” codes were created to map SLURM commands to CCQ. The cluster deployment workflow uploads 3 bash script files (sbatch.sh, squeue.sh and sacct.sh) and then creates aliases to those files for the corresponding SLURM commands. These bash scripts provide a mapping between CCQs corresponding commands (ccq, ccqstat) and transforms some of the outputs to return values Tapis expects. With these aliased scripts in place, Tapis is then able to successfully submit and monitor job statuses just as it would for a typical SLURM scheduler based cluster.

C. Using Tapis to deploy a GCP cluster and run computation jobs

We implemented the above work as a Tapis Actor, which is a constantly running cloud function which takes a JSON input message. This allows for different functionality based on the messaged passed. Using the Tapis Actor, gateways or researchers can issue messages to deploy clusters, submit Tapis jobs to the clusters, and decommission clusters.

Figure 2 explains the two processes for cluster creation and deletion with the Tapis Actor.

1) If CREATE message is given, Automaton calls Google Cloud APIs to create the CloudyCluster instance.
2) The system is registered with Tapis, with the DNS of the login instance and the environment name saved as extra parameters in the system definition.
3) Default cluster is created, consisting of a single g1-small CCQ enabled Slurm Scheduler, a g1-small Login instance, a 100GB OrangeFS Filesystem, and a g1-small NAT instance
4) If TEAR DOWN message is given, the DNS and environment name is looked up from the system definition
5) Automaton calls Google Cloud APIs with the DNS and environment name as parameters to delete the cluster.
6) Cluster is removed and unregistered as a Tapis system.

By leveraging this Tapis Actor, a science gateway can create on-demand cloud cluster resources in GCP that gateway users can leverage as execution systems for their computational workflows. Since the cluster is deployed with CloudyCluster, the resulting cluster is elastic, so nodes are provisioned on demand according to the resource requests from the job.

.

IV. Future Work

Future work would be to add additional environment variables as parameters for the Tapis Actor registration, which could be used to allow for customization of resources in the default cluster. This way, a user or gateway administrator could define those when registering the Actor. Another focus would be to implement this solution for into an existing Tapis science gateway to allow for scaling and provide additional computing capabilities when existing local resources may be over-subscribed or unavailable.

V. Conclusion

The goal of this work was to enable science gateways to dynamically provision Cloud HPC cluster compute resources and launch distributed workflows using the Tapis Framework to manage the end-to-end workflow of a computational job, including tracking the history and provenance. Through the use of Tapis, Automaton and CloudyCluster this proof-of-concept has been achieved and enables the ability to deploy an entire cluster and elastically provision compute node resources when a job is submitted with Tapis to run a distributed workflow. This new capability can provide additional options for science gateways and researchers in access to cloud resources in dynamic and cost efficient manner.

VI. Software Availability

The source code for this project is available on Github at https://github.com/UH-CI/automaton_tapis

VII. Acknowledgements

This work was supported by the National Science Foundation grants - Tapis Framework:[1931439 and 1931575], Collaborative Research: SS2-SSI: The Agave Platform: An Open Science-As-A-Service Cloud Platform for Reproducible Science NSF OAC #145041. #1931439 and #1931575, and RII Track 1: ‘Ike Wai Securing Hawai‘i’s Water Future NSF OIA #1557349. We also acknowledge Google for the cloud credits provided that enabled validation and testing of this work.

References

[1] J. Stubbs, R. Cardone, M. Packard, A. Jamthe, S. Padhy, S. Terry, J. Looney, J. Meiring, S. Black, M. Dahan, S. Cleveland, and G. Jacobs, “Tapis: An api platform for reproducible, distributed computational research,” 2020, submitted.

[2] S. Y. W. B. L. I. E. C. Quarles, “Scalable distributed directory implementation on orange file system,” in *SNap 2011*, 2011.

[3] A. B. Yoo, M. A. Jette, and M. Grondona, “Slurm: Simple linux utility for resource management,” in *Workshop on job scheduling strategies for parallel processing*. Springer, 2003, pp. 44–60.

[4] A. H. Brandon Posey, Christopher Gropp and A. Apon, “Automated cluster provisioning and workflow management for parallel scientific applications in the cloud,” https://tigerprints.clemson.edu/computing_pubs/38, 2017.