Evaluation of the components of a distributed workload managers on new platforms
DOE OMNI Internship Programs - Summer 2022

Project: GlideinWMS
Location: Fermilab – Batavia – Illinois
Intern: Rami Aboushadi – MS - Jackson State University- Jackson – Mississippi

Introductions: GlideinWMS

The purpose of the GlideinWMS is to provide a simple way to access the Grid resources. GlideinWMS is a Glidein Based WMS (Workload Management System) that works on top of HTCondor. Glideins are like placeholders, a mechanism by which one or more remote resources temporarily join a local HTCondor pool. The HTCondor system is used for scheduling and job control.

Figure 1: GlideinWMS Structures
Source: https://glideinwms.fnal.gov/doc.prd/frontend/index.html

- Users submit jobs to the User Pool HTCondor Shed process.
- The GlideinWMS Frontend polls the user pool to make sure that there are enough glideins (workers) to satisfy user jobs. It submits requests to the glidein factory to submit glideins.
- The Glidein Factory and WMS Pool receives requests from the frontend(s) and submits a HTCondor starts wrapper (glidein) to entry points (grid sites).
- The grid sites receive the (glidein) jobs and start a HTCondor.
- Then glidein shows up as a resource in the User Pool.
- The user jobs are matched with these resources.
The GlideinWMS Factory and Frontend and HEPCloud's Decision Engine are coded in Python. To run on well supported platforms and to take advantage of new features, over time it is necessary to support new platforms, new Operating Systems and new Python versions. Python 3.6 is the supported Python on EL7 and was EOL in December 2021. GlideinWMS and HEPCloud decided to move to Python 3.9 and to EL8, which provides better support for Python versions newer than 3.6. Also Worker Nodes, the computers where the scientific software is run, evolve over time and it is useful to know the effectiveness of new architectures in running scientific applications. The goal of this project is to test different part of the systems, adapt GlideinWMS and write recommendations for the use of new platforms.

Methods:

To evaluate the GlideinWMS and HEPCloud components on REHL8 I followed a series of procedures:
- Setup of AlmaLinux 8 VMs with Python 3.9.
- Running the continuous integration tests on RHEL8 with Python 3.9. And troubleshooting and fixing some failing unit tests.
- Installation, testing and adaptation of GlideinWMS Frontend and Factory on RHEL8 with Python 3.9.
- Installation, testing and adaptation of Decision Engine on RHEL8 with Python 3.9.
- Benchmarking of new Hyper-threaded worker nodes with different loads from a real Dune workflow.

Figure 2: Example of some files that failed unit test before and after debugging and troubleshooting.

Conclusions:

I learned and I've got more experienced during the project with the help of my mentor and the project team. I was able to troubleshoot the unit tests and fix some problems with deprecated Python constructs and with a different handling of multi-processing. We decided to skip the test of epoll and remove it from the possible alternatives to handle multi-processing.

I was able to install all the systems on Alma Linux 8 (RHEL8 compatible) with Python 3.9: GlideinWMs Frontend, VO Pool, and Factory, and HEPCloud's Decision Engine.

The DUNE workflow was able to scale on the Worker node using all cores.
Evaluation and experiences:

It is very valuable experiences in term of learning and developing my skills. Helps from projects team and service staff including housing was outstanding and very accurate.

The mentor, supervisor and developer provided the following:

- Communications and support with service desk and housing to meet all essential needs for the project and work atmosphere.
- Welcome tour to Fermilab building.
- Meeting with HepClouds - GlideinWMS interns and supervisors
- Prepared shells for bash command and python to all OS.
- Provided most of the commands needed for the projects.
- Zoom meeting to learn about GitHub account and git commands.
- Create, upload, add, deleting files to GitHub account.
- Zoom meeting to navigate Linux systems, using shell commands, install and testing software.
- Zoom meeting for using python shells and commands.
- Creating VM on Stack.
- Round table meeting with all interns, supervisors, Mentors and developers for reports and troubleshooting