JOINT INSTITUTE FOR NUCLEAR RESEARCH
Laboratory of Information Technology

FINAL REPORT ON THE SUMMER STUDENT PROGRAM

SALSA development and deployment on HybriLIT and Kubernetes cluster

Supervisor:
Oksana Ivanovna Streltsova

Student:
Branislav Beke, Slovakia
Stredna priemyselna skola elektrotechnicka, Kosice

Participation period:
July 11 – September 05

Dubna, 2019

Table of Contents

Abstract

This report describes work done at projects which are used or to be used at HybriLIT cluster and newly created Kubernetes cluster located at LIT in JINR.

The main project was to advance development and general design of SALSA (Scalable Adaptive Large Structures Analysis) which is to be used to process large data from various sources. This framework aims to be highly flexible, fault-tolerant and user-friendly.

The side project was aimed at helping administrators with Let’s encrypt wild card certificate deployment.

Acknowledgment

I would like to thank my supervisor Oksana Ivanovna Streltsova for inviting me to take part in this program at Joint Institute of Nuclear Research in Dubna.

I would like to thank Martin Vala from University of Pavol Jozef Safarik in Kosice for help, ideas and motivation for this project.

I would also like to thank Yuri Aleksandrovitch Butenko for his indispensable help with the project as well as during our stay in Dubna.

I would like to thank Elena Karpova for organization of our practice.

Last but not least I would like to thank everyone else who helped us, as well as everyone who expressed support towards our project.

SALSA development

SALSA is a project developed for processing large volumes of data. Its design differs from others in several ways. Most notably we aim to differ between different kinds of load (e.g. GPU, CPU or I/O) so we can keep the cluster running as efficiently as possible.

Codebase refactoring

My first task was to refactor the whole SALSA codebase. While working on this task I have learned a lot about C++ standards, most notably C++11 (which at the time of writing was most recent standard available on compilers bundled with CentOS 7). I have also learned a lot about OOP (Object Oriented Programming) concepts and idioms which help with code development and maintenance.

Part of refactoring was to deploy consistent naming scheme. After deciding on several code styles we created our own. This style emphasizes readability of the code as well as its clarity. I have also tweaked settings of our code formatter, so our codebase remains up to the standard.

Load balancing design

Another part of my work here was to design implementation with obmon project[1], which is used as daemon for monitoring system services. This program aggregates and provides us with information necessary to make our decisions. By not implementing this functionality directly we avoid feature creep as well as increase the modularity of the whole system.

Every node will decide for itself if it can process more jobs based off of metadata received when the job is submitted to the network. These metadata, however, will not be perfect from the beginning, thus there is a potential for this project to include some level of AI which would decide how individual tasks load the servers.

1 https://gitlab.openbrain.sk/obl-dev/obmon

Our load balancing design is unique in a way that it can load up servers with different kinds of load. For example, when we have a node which has unused GPU but fully loaded CPU for extended period of time (or, by binary analysis, we know that task will not require GPU), we can almost certainly run another task that will be mostly GPU intensive.

Framework rebuild

During my last week here we decided to ultimately split SALSA into two parts: cluster management and processing itself. This will simplify codebase and enable us to create a shared library that can be integrated into a wider spectrum of projects. New design decisions will also lead to more stable API/ABI which will ease development and deployment by a whole lot.

LEAST development

LEAST (Let's Encrypt Automated Support scripts) is a utility aimed to help administrators with Let's encrypt wild card certificate maintenance.

Let's encrypt is a certificate authority issuing basic certificates free-of-charge[2]. Their client tool of choice is certbot[3], which can be fully automated for basic certificates. However, when issuing wild card certificates we have to prove domain ownership via creating a verification blob on the target server and by adding verification entry into DNS for the domain. Because most of domains used by our projects do not use supported DNS authorization plugins, we have to apply DNS entries automatically.

LEAST aims to simplify this process by requiring administrators to only create configuration files for domains and provide support script (if it's not available already). While certbot itself can handle plugins, it cannot handle situations when for example you are issuing a single certificate for a single server with multiple domain names with DNS at different providers.

For situations as these we take advantage of certbot's manual script hook. This hook gets called for each domain entry separately, thus allowing more granular control of authorization process. This allows us to deploy more complicated certificates quite easily without the need for administrator, which would have to manually update certificates.

LEAST also solves issue with site security and availability. Since the certificate renewal process is fully automated, we can run default service that checks for expired certificates twice a day. With this we avoid expired certificates, which can cause problems for example with `HTTP Strict Transport Security` header.

2 https://letsencrypt.org/
3 https://letsencrypt.org/docs/client-options/