POWER8 Continuous Integration Technology Review

Leon Leighton, Thomas Olson, Derek Wong
Project 35

Abstract

CONTENTS

I. Cluster Management

A. Options

1) OpenStack: OpenStack is a cluster management solution originally created by Rackspace and NASA, and now managed by the OpenStack Foundation. It contains a number of projects that provide services such as object and block storage, identity management, networking and compute resource management, bare metal provisioning, and DNS services. This modular approach allows users to select which parts they need to accomplish their goals without having to install components they will not be using.

2) CloudStack: CloudStack is an Apache Software Foundation cluster management project which aims to provide many of the same services as OpenStack. It follows a more monolithic approach where most components are distributed as part of a single binary.

3) Ganeti: Ganeti is a cluster management solution from Google that focuses on managing virtual machines. It can be used to create new virtual machines, manage disks, and manage failover of virtual machine instance.

B. Goals for use

The primary goal for using cluster management software is to manage the creation and destruction of temporary virtual machines and containers that will be used for building and testing projects.

C. Criteria for Evaluation

1) Community Support: Our chosen solution should have a community that provides support in the form of documentation, bug fixes, and assistance.
2) Linux Support: Linux must be supported as both host and guest.
3) Container Support: Optional, but we would like to have support for building and testing in containers.
4) Jenkins Plugin Support: Jenkins is a likely piece of our solution, therefore having Jenkins plugin that interacts with our cluster management solution is highly desired.

D. Table

E. Discussion

OpenStack, CloudStack, and Ganeti can all run on Linux and can host Linux virtual machines. Ganeti does little more than this, however, and will not be considered further. Both OpenStack and CloudStack can interact with Jenkins through a plugin. OpenStack appears to have a larger community and a more mature container solution than CloudStack.

F. Selection of best option

OpenStack is our chosen solution for cluster management. It meets all of our criteria for evaluation.

II. Continuous Integration Software

A. Options

1) Jenkins:
2) Buildbot:
3) Strider:

B. Goals for use

Automate building and testing Open Source projects.

C. Criteria for Evaluation

1)
2)

D. Table

E. Discussion

F. Selection of best option

III. Configuration Management

A. Options

1) Ansible:

2) Chef:
3) Puppet:

B. Goals for use

C. Criteria for Evaluation

1)
2)

D. Table

E. Discussion

F. Selection of best option

IV. Login/Authentication

A. Options

1) Google Login Plugin
2) Active Directory Plugin
3) Jenkins Login

B. Goals for use

The goal for having an authentication is to provide our users a safe and secure place to build their project and to use our continuous integration service. It will help prevent hackers from obtaining private information that could be detrimental to our users. This is a security measure that is absolutely necessary for our project.

C. Criteria for Evaluation

The main criteria to look for in this technology is security. Security is what keeps users confidential information such as source code from being leaked.

D. Table

E. Discussion

The google login plugin is a really straight forward method for authentication. Users can simply use their google accounts to login. This method requires an OAuth 2.0 credentials from the Google Developers Console and there are clear instructions on the internet to accomplish this task [1]. The second option is to use the active directory plugin. This plugin is used with Jenkins to authenticate the username and password through active directory [2]. The third option is to use Jenkins login. Users can simply create a Jenkins account and have access to its functionalities. They will also have the ability to give authorizations to different accounts for their projects.

F. Selection of best option

The best option is to use Jenkins login. We intend to build our continuous integration service project on a POWER8 architecture using a Jenkins backend; the most reasonable approach is to use Jenkins own login system.

V. Frontend/Web frameworks

A. Options

1) Tomcat
2) Glassfish
3) Jetty

B. Goals for use

The goal for having a web framework is so that our users have a platform to use our continuous integration service. We will provide a simple and easy to use user interface (UI) so that our users would not have a complicated time using our services.

C. Criteria for Evaluation

The criteria in this technology are load time, usability, security, and stability. Having an easy-to-use user interface can give our users an easy time to use our service and it will help reduce confusion. The page load time should load at a reasonable speed so that our users would not have to wait a long time using our service. Security is important for our web framework because we dont want hackers to gain access to our services. In order to provide our users a good experience, we need a secure environment. Having good stability is important as well. Our service need to maintain at a functional state at all time so that it doesnt hinder our users from working.

D. Table

E. Discussion

Tomcat is very popular and it is known as a lightweight application that offers all the basic features required running a server. It is an open source project so it is cost free. It is highly flexible because it allows users to tweak their code as they see fit, and it has many built-in features [3]. The second option is Glassfish and it is also an open source project, which means that it is free of cost. It has high performance and it was considered the fastest open source application server, according to SPECjAppServer2004 benchmark results [4]. Jetty provides a Web server and javax.servlet container. They have a heavy focus on multi-connection HTTP and features such as SPYDY, which can significantly reduce page load latencies [5].

F. Selection of best option

Overall, all three of the options are great, but the best option to use is Tomcat. Glassfish has a lot of features and is the biggest out of the three options, but we are only using it for our frontend, so most features are unnecessary. It has a big memory footprint, which means that it would consume more resources. Jetty is a very light application and it consumes the least amount of resources but it is too small for our project. Tomcat is ideal because it sits in between these two options and it offers a good amount of features to use.

VI. Tracing State of builds/tests

A. Options

1) Build Monitor Plugin
2) Lava Lamp Notifier
3) Radiator View Plugin

B. Goals for use

The goal to having the functionality to trace the build and test state is so that our users can see the status of their projects. This will reveal the success or failure of projects that were build by our service and it will show where the failures have occurred.

C. Criteria for Evaluation

The criteria to look for in this technology are usability, feature to display pass or fail builds, feature to display progress of builds, and feature to display where the error has occurred for build fails. Having an easy-to-use user interface (UI) allows our users to operate our service with minimal issues. The ability to display pass or fail build is the main criteria for this technology so this is a must have. The feature to display progress when building a project is helpful for our users to see how much longer it will take until the build is complete. The feature to see where an error has occurred in the build process is another important criteria; it will help developers save time from manually tracking down the problem.

D. Table

E. Discussion

The build monitor plugin supports many features and incorporates other plugins as well. The features in this plugin include:

- Displays the status and progress of selected jobs, the view is updated automatically every couple of seconds using AJAX. No ‘Enable Auto Refresh’ needed.
- Displays the names of people who might be responsible for ‘breaking the build’.
- Supports theClaim plugin, so that you can see who’s fixing a broken build.
- SupportsView Job Filters, so that you can easily create Build Monitors for ‘slow builds’, ‘only failing’, etc.
- SupportsBuild Failure Analyzer, so that you know not onlywho, but alsowhatbroke the build.

- SupportsCloudBees Folders Plugin, so that you can have project- and team-specific nested Build Monitors.
- The number of columns and size of the font used is easily customizable, making it trivial to accommodate screens of different sizes.
- UI configuration is stored in a cookie, making it possible to display different number of columns and using different font size on each of the screens at your office.
- Can work in a colour-blind-friendly mode
- [6]

The lava lamp notifier is a really simple design created to indicate if a build passed or fail. The only necessary component to use this type of method to track our build status is a USB LED light. The light will simply light up green for pass and red for fail [7]. This is not ideal for our type of project because our users would like to know more information than just a pass or fail notification. The third option is the radiator view plugin. This plugin is somewhat similar to the build monitor plugin but with less features. It will display all the project that are building and indicate pass or fails on screen [8].

F. Selection of best option

The best option to use is the build monitor plugin. This technology has the most features out of all the other options I listed, and all the features are very useful for our users.

REFERENCES

[1] recampbell Campbell. (2015, Nov) Google login plugin. [Online]. Available: https://wiki.jenkins-ci.org/display/JENKINS/Google+Login+Plugin
[2] K. Kawaguchi. (2016, Oct) Active directory plugin. [Online]. Available: https://wiki.jenkins-ci.org/display/JENKINS/Active+Directory+plugin
[3] A. Mangat. (2010, Sept) Top 7 features in tomcat 7: The new and the improved.
[4] Oracle. (2010) The oracle glassfish server advantage for small businesses. [Online]. Available: http://www.oracle.com/us/products/middleware/application-server/glassfish-server-adv-sbiz-wp-080573.pdf
[5] webtide. (2016) Why choose jetty. [Online]. Available: https://webtide.com/why-choose-jetty/
[6] J. Molak. (2016) Build monitor plugin. [Online]. Available: https://wiki.jenkins-ci.org/display/JENKINS/Build+Monitor+Plugin
[7] E. Randall. (2010) Lava lamp notifier. [Online]. Available: https://wiki.jenkins-ci.org/display/JENKINS/Lava+Lamp+Notifier
[8] M. Howard. (2011) Radiator view plugin. [Online]. Available: http://wiki.hudson-ci.org/display/HUDSON/Radiator+View+Plugin