Continuous Integration Sandbox using Ansible

Chethan Aravind T S
Student, M.Tech, Network & Internet, Dept. of ECE
Sri Jayachamarajendra college of Engineering
Mysuru, Karnataka, INDIA.

Dr. M. N. Jayaram
Associate Professor, Dept. of ECE
Sri Jayachamarajendra college of Engineering
Mysuru, Karnataka, INDIA.

***Abstract*-The two key technologies in computing, hardware and software, exist side by side. Improvements in one drive improvements in the other. Both are full of major advances and technological dead ends, and both are replete with colorful characters and dynamic start-up companies.But there are key differences between the hardware and the software industries. Hardware design and manufacture is a comparatively costly exercise, with a consequently high cost of entry. Nowadays only large, or largish, companies can do hardware. But many of the major software advances have been the results of individual effort. Anybody can start a software company, and many of the largest and most successful of them have come from nowhere, the result of one or a few individual's genius and determination. There are many different types of software. There are application softwares, such as financial programs, word processors and spreadsheets, that let us do the sort of work we buy computers for. There are system softwares, such as operating systems and utilities that sit behind the scenes and make computers work. There are application development tools, such as programming languages and query tools that help us develop applications.**

Key words: Hardware, software, computers, continuous integration, Ansible

I INTRODUCTION

In the age of the Internet, where software is more mission-critical than ever, it's no longer enough for our development projects to succeed some of the time. We need to deliver excellence, consistently and we must do it faster than ever. Successful Software Development proceeds from the fact that there is no one way to develop software systems and introduces a model for a mature software development process that accommodates flexibility, the Systems Engineering Environment (SEE).

This model comprises two fundamental, interlocked elements: the policies and procedures that define how software development is performed and the technologies available to get the job done. Using the SEE framework, learn how to: understand and "sell" the business case for software improvement; establish and nourish an ongoing, productive dialogue between developers and customers; manage the multiple constituencies, personalities, issues, and egos that complicate software development; create plans that reflect the need for change—and take into account real-world risks; write clearer, more useful contracts and statements of work.

II LITERATURE SURVEY

High performance computing systems are becoming increasingly complex, both in node architecture and in the multiple layers of software stack required to compile and run applications. As a consequence, the likelihood is increasing for application performance regressions to occur as a result of routine upgrades of system software components which interact in complex ways. The purpose of this study is to evaluate the effectiveness of continuous integration tools for application performance monitoring on HPC systems. In addition, this paper also describes a prototype system for application performance monitoring based on Jenkins, a Java-based continuous integration tool. The monitoring system described leverages several features in Jenkins to track application performance results over time. Preliminary results and lessons learned from monitoring applications on Cray systems at the Oak Ridge Leadership Computing Facility are presented. [1]

The Cisco AMP solution stands apart from traditional antimalware technologies by providing augmented protection, improved visibility, and enhanced control. Cisco AMP uses an extensive infrastructure of sandboxes to analyze hundreds of thousands of files each day. Benefits from this infrastructure are in two specific ways.

First, search our database for previously inspected files. Because Cisco AMP processes hundreds of thousands of files every day, often times the diagnosis is already available. If the file has already been analyzed, access a full report of that information immediately and respond accordingly.

Second, upload files that are unknown for individual analysis and get results in minutes. After it is submitted, the file is put into a remote, protected sandbox environment. Analysis is fast, and the results are then cataloged for future queries. [2]

The creation of a secure, virtualized system sandbox environment at the level of the respective applications is how we can ensure the safety of our applications and infrastructure. The proposed sandbox model allows us to generate a secure environment for various untrusted applications and resolve potential security incidents, such as zero day vulnerabilities. The resulting work is a functional sandbox within the MS Windows operating system, which protects the system against potentially hazardous applications. The sandbox has a minimal impact on the semantics and the time of the executed program and provides an efficient sandbox configuration interface. [3]

III IMPLEMENTATION

The first step in this project is to have a GIT repository to commit our code. This action can be then used as one of the triggering actuators for Jenkins. Upon every commit Jenkins will fetch the code and makes a WAR file Out of it. WAR file or the Web Archive is the version of code suitable for a web deployment. For security purpose, A copy of the created WAR file is pushed to the SFTP server. Jenkins will now trigger the Ansible Job. This is the main step in this project. The Ansible job is nothing but running the Ansible playbooks which will install Hadoop in the local system and then another ansible playbook which will Provision the AWS EC2 Instance. Log in to the AWS Instance and then Install Tomcat Server on the Instance

GitHub - is a code hosting platform for version control and collaboration. It lets you and others work together on projects from anywhere.

GitHub assigns a repository which will be used to store the code from developers. GitHub makes it easy to add data whenever we want. Git can also be accessed from a API level interface also called a web-hook which is featured in this project.

IV SPECIFICATIONS AND REQUIREMENTS

Component	Requirement
Processor	i5 Processor
RAM	8GB (Windows)
Hard Disk capacity	500GB (Windows)

Table 1 Hardware Specifications of Workstations

Component	Requirement
Operating System	UBUNTU/ RHEL 6/ Centos
RAM	6 GB
Hard Disk Capacity	50 GB

Table 2 Hardware Specification of Hadoop VM

Component	Requirement
Operating System	UBUNTU/ RHEL 6/ Centos
RAM	4 GB
Hard Disk Capacity	30 GB

Table 3 Hardware Specifications of Jenkins and AWS VM

Jenkins - is a self-contained, open source automation server which can be used to automate all sorts of tasks such as building, testing, and deploying software. Jenkins can be installed through native system packages, Docker, or even run standalone by any machine with the Java Runtime Environment installed as shown in this project.

Ansible - is an IT automation tool. It can configure systems, deploy software, and orchestrate more advanced IT tasks such as continuous deployments or zero downtime rolling updates.

Ansible's main goals are simplicity and ease-of-use. It also has a strong focus on security and reliability, featuring a minimum of moving parts, usage of OpenSSH for transport, and a language that is designed around auditability by humans–even those not familiar with the program.

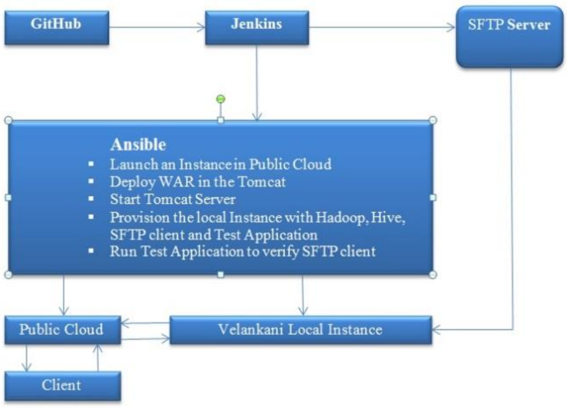
GitHub
Jenkins
SFTP Server
Ansible
Launch an Instance in Public Cloud
Deploy WAR in the Tomcat
Start Tomcat Server
Provision the local Instance with Hadoop, Hive, SFTP client and Test Application
Run Test Application to verify SFTP client
Public Cloud
Velankani Local Instance
Client

Figure 1: Block Diagram

V WORKFLOW

A flowchart indicates sequences and decision points as well as starting and stopping points. On resetting the power, the procedural flow starts with initialization of modules involved.

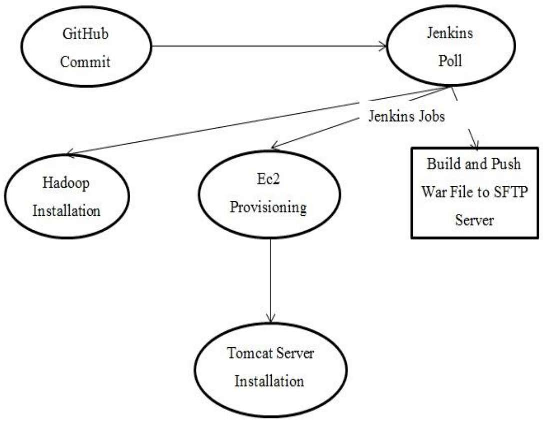

Figure 2: Flowchart

VI CONCLUSION AND FUTURE WORK

A. *Conclusion*

Exploring new possibilities has been the way of our lives ever since we had consciousness without which we wouldn't have been where we are. Technological race among the competitive companies has given birth to revolutionary tools that can make the process of software production a simple task which in turn make the lives of everyone even more easily.

Continuous Integration is an excellent software production/testing practice. CI also eliminates the wall between the developer and the testing team. Integrating the processes together along with customer involvement improves the quality and production time of softwares as well as gives room for changes at runtime.

Implementation of CI inside a Sandbox not only gives us a safe place to run and test our code but also saves a lot of infrastructure. Let us take a look at the hardware we have used in this project

Component	Requirement
Operating System	UBUNTU/ RHEL 6/ Centos
RAM and ROM of local System (on Virtual Machine)	10GB RAM 80 GB ROM 1 virtual CPU per VM
RAM and ROM of AWS Instance	1 GB RAM 5 GB ROM 1 virtual CPU

Table 4 Hardware usage

As we can see we have used only required amount of resource to create an environment that will allow us to test our code. Maybe the server we have created in the sandbox is not capable of handling multiple requests at once but is capable enough to process the requests of the tester which is exactly what we need.

B. Future Work

Ansible is a relatively new comer to the configuration management world. Though it is tough to use it for advanced configuration today, it promises a great future in the field of configuration management. A growing community will only make the tool even better every day.

Continuous integration is not the end of line as far as Software Development Cycle goes. There are more complex yet efficient methods like RAD (Rapid Application Development) and other variants of the Continuous models like Continuous Deployment and Continuous Delivery which can be implemented in the future.

REFERENCES

[1] Use of Continuous Integration Tools for Application Performance Monitoring, Ver´onica G. Vergara Larrea, Wayne Joubert and Chris Fuson Oak Ridge Leadership Computing Facility Oak Ridge National Laboratory Oak Ridge, TN, USA

[2] Cisco Advanced Malware Protection Sandboxing Capabilities, Cisco White Paper.

[3] Application Security through Sandbox Virtualization, ActaPolytechnicaHungarica, Vol. 12, No. 1, 2015

[4] Pre-Production Testing - Best Practices & Guidance, White Paper

[5] Sandboxing Applications, VassilisPrevelakis,Department of Computer and Information Science, University of Pennsylvania and DiomidisSpinellis, Department of Technology and Management, Athens University of Economics and Business.

[6] Continuous Integration, Delivery and Deployment: A Systematic Review on Approaches, Tools, Challenges and Practices, MojtabaShahin, Muhammad Ali Babar, Liming Zhu.

[7] A. Phillips, M. Sens, A. de Jonge, and M. van Holsteijn, The IT Manager’s Guide to Continuous Delivery: Delivering business value in hours, not months: XebiaLabs, 2015.

[8] J. Humble, and D. Farley, Continuous Delivery: Reliable Software Releases Through Build, Test, and Deployment

[9] Ansible documentation available at: http://www.ansible.com/docs

[10] Amazon Web Services Documentation: https://aws.amazon.com/documentation