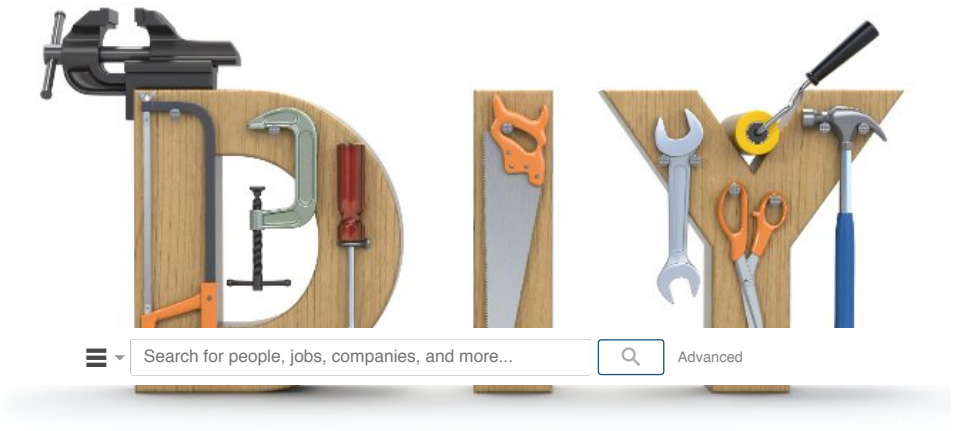


Do-It-Yourself Continuous Integration on Cloud Toolkit (Part1)

Published on August 26, 2016

Ravi Gumpu | Follow
Pega System Architect | BDD | Test Automation

20 1 2

Continuous Integration (CI) and Continuous Delivery (CD) are essential parts to any modern development and deployment process. Gone are the days of monolithic releases with massive changes, today it's all about releasing fast and often. Most teams have come to rely on some sort of automated CI and CD system. In this article we will discuss about how we can take CI and CD process to Cloud (Google Cloud) using Jenkins as the tool. We will also discuss on how to integrate the local development environment to GitHub and Jenkins (on Google Cloud).

Continuous Integration

Wikipedia defines Continuous Integration as

... the practice, in software engineering, of merging all developer working copies with a shared mainline several times a day.

Continuous Integration is very frequently accompanied by Continuous Delivery/Deployment (CD) and very often when people talk about CI they refer to both.

CI relies on two main principles:

Changes are merged to master as often reasonably possible. The tasks are explicitly split up to create minimal working sets.

- Each change is fully tested. Automated testing is the heart of CI. In a team environment, and even on a personal project, it's nearly impossible to insure that

latest changes don't break existing code without tests. Every time a change set is merged to master, CI runs entire suite to guarantee nothing was impacted negatively.

Do-It-Yourself – Continuous Integration

Create a repository in GitHub

1. From your local Development Environment Push it to GitHub
2. Verify on GitHub whether all the artefacts are available in your chosen repository or not

Integration Phase 2 – Set up Jenkins on Google Cloud

Google Compute Engine - Google Compute Engine delivers virtual machines running in Google's innovative data centers and worldwide fiber network. Compute Engine's tooling and workflow support enable scaling from single instances to global, load-balanced cloud computing.

- Compute Engine's VMs boot quickly, come with persistent disk storage, and deliver consistent performance. Our virtual servers are available in many configurations including predefined sizes or the option to create Custom Machine Types optimized for your specific needs. Flexible pricing and automatic sustained use discounts make Compute Engine the leader in price/performance.

Step1 – Open up Google Cloud Console

Step2 – Go to VM instances

Step3 – Create a VM instance

Step4 – Start the VM instance

Step5 – Go to the SSH terminal of the above created VM instance

Step6 – Install Basic Packages (Git) sudo apt-get install git

Step7 – Setup Jenkins on VM

Install

```
wget -q -O - http://pkg.jenkins-ci.org/debian/jenkins-ci.org.key | sudo
apt-key add -
sudo sh -c 'echo deb http://pkg.jenkins-ci.org/debian binary/ >
/etc/apt/sources.list.d/jenkins.list'
sudo apt-get update
sudo apt-get install jenkins
```

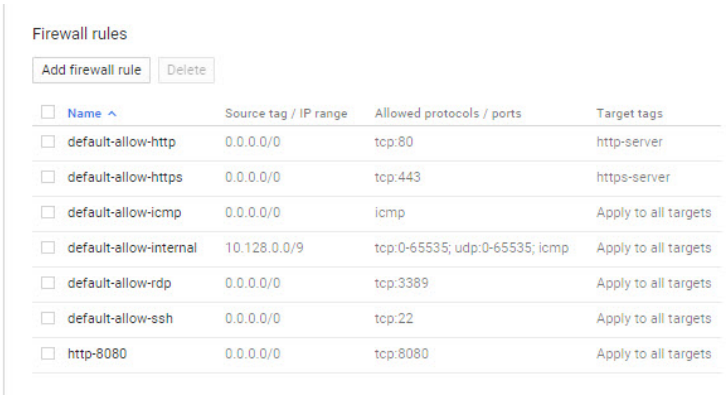
- Check if 8080 is being used on your VM, if not there's no need to change the port. Otherwise change Port as follows: run vim /etc/default/jenkins and edit the file as follows (note: on some systems this file may be readonly and you may need to use sudo for the edit command):

```
# port for HTTP connector (default 8080; disable with -1)
```


- To access Jenkins you need to add firewall rule to open port in gce-networks

1. Select Network

2. Add firewall rule to allow Jenkins port



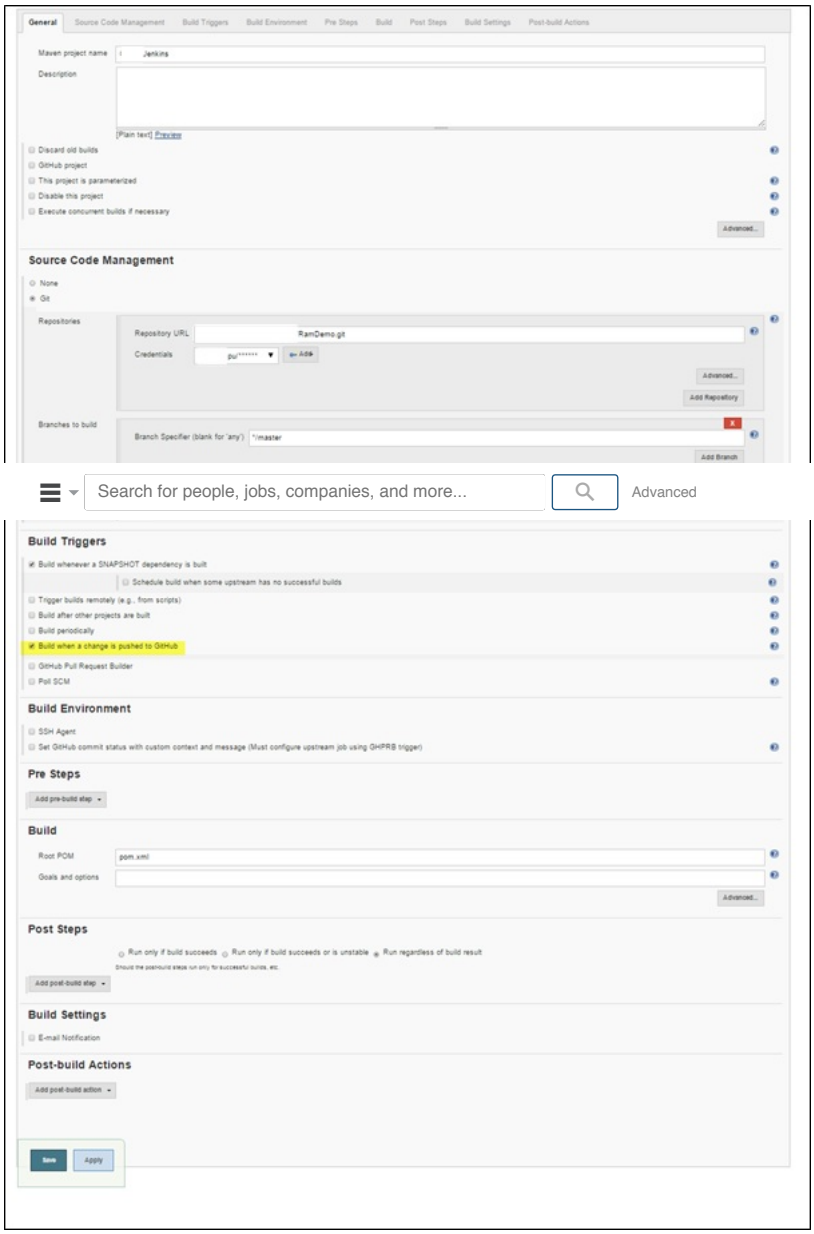
3. Start Jenkins: run sudo /etc/init.d/jenkins start

4. Open up the VM on which Git and Jenkins are installed by adding the above port number to the IP of the VM

Ex - http://10X.19X.18X.40:8080/

5. Follow the standard instructions of setting up the Jenkins

6. Set up a Maven Project in Jenkins and configure with the GitHub details.



Integration Phase 3 – Integrate GitHub and Jenkins

"Jenkins Hook Url" is the URL of your Jenkins server's webhook endpoint. For example: http://ci.jenkins-ci.org/github-webhook/.

Testing Phase – End to End integration testing

Step1 – Add new Tests/App Code in your local dev

Step2 – Push it to GitHub

Step3 – Go to the Jenkins Console and observe that the Jenkins is able to pick up the changes which are pushed to GitHub and start building

Step4 – Validate the changes pushed to GitHub in Jenkins Console

Tagged in: jenkins, continuous integration, cloud computing

Report this

Ravi Gumpu
Pega System Architect | BDD | Test Automation
7 posts

Follow

1 comment

Recommended

Leave your thoughts here…

Sunny Poswal ··· 39m
SAFe® Agilist, DevOps Architect & Agile Coach
Ravi, Very helpful post for the folks who want to learn CI by doing it. Keep-it-up!!
Like Reply | 1

Don't miss more posts by Ravi Gumpu

Bringing Serene (ity) to Web- services Automation
Ravi Gumpu on LinkedIn

User Story Readiness Index – Measuring the bandwidth of “INVEST”…
Ravi Gumpu on LinkedIn

BDD Anti Patterns
Ravi Gumpu on LinkedIn

Looking for more of the latest headlines on LinkedIn?

Discover more stories