OneDevOps | Build-On

A Solution for Scalable on Demand CI Infrastructure

Setting up a CI Ecosystem – The Complications…

Infrastructure needs differ with Technologies / applications

Multiple plug-ins needed for different Technologies / applications

Multiple Configurations are needed with each set-up

Time-consuming Manual Hardware scaling

The Voice of the industry…

<5% Average Hardware Utilization when using Jenkins *(Source: eBay)*

>30 Days Average Time taken to set-up Jenkins Master-Slave ecosystem

Setting up a CI Ecosystem – The Impact

Build Engineer

Limited Scalability

Cumbersome Maintenance

Error Prone Provisioning

Developer

Lower productivity

Snowflake build configurations

Longer Waiting times

Continuous Integration - Tools alone not enough!

Container Technology – Gaining momentum

200%

Growth in usage of Dockers since 2015

41%

People who are already using Containers in 2016

18%

Container usage that has reached Production

Source: Rightscale cloud report 2016

Benefits of Using Containers for CI

- Consume lesser Hardware resources
- Can be provisioned on-demand and can be freed after execution
- Self-contained capsules that can be customized for specific applications
- Provide unlimited scalability

Create containers On-demand with

OneDevOps | Build-On

Get **Unlimited scalability** for Continuous Integration
Increase utilization and agility of **Build infrastructure**

Build-On: Key Features

Optimized Build Infrastructure

- Self-Contained container provisioning
- Minimal manual intervention
- VMs and Containers freed after the execution, thus optimizing infrastructure

Code-Based Management of Environments & Pipeline

- Enables Developers to descriptively define pipeline & Jobs as simple YAML files
- Codified approach for standing up Jenkins and the associated Plugins ecosystem

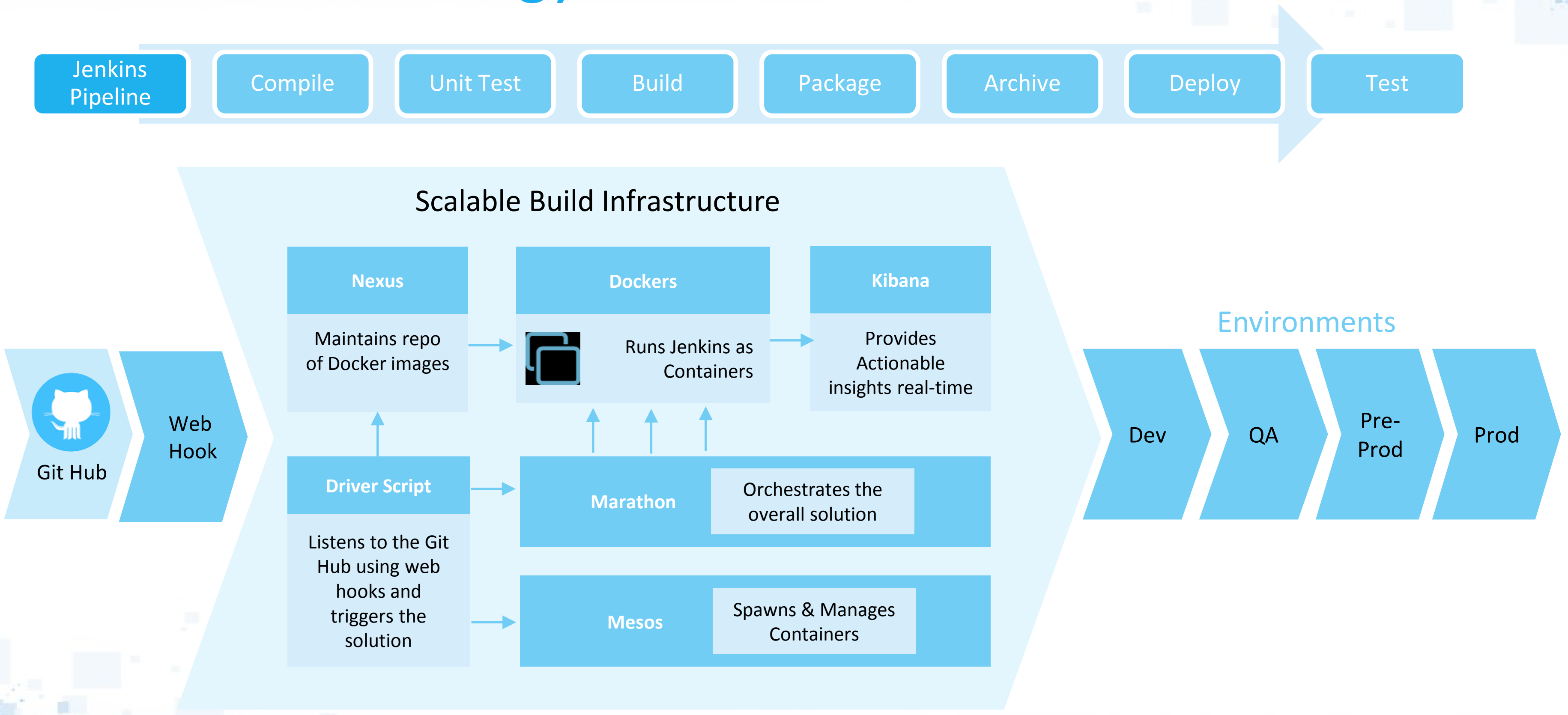
Scalable, Self-Controlled Build Environments

- Built based on the best names in Industry – Docker, Apache Mesos, Marathon and Jenkins
- Scale up and Scale Down vertically and horizontally based on need

Flexibility in Supporting Multiple Technologies

- Extensible solution to support other CI tools
- Works on standard on premise (VMWare, Open Stack) or cloud infrastructure (AWS, Rack Space, etc)
- Pluggable with most CMP solutions for cost analytics and governance

Build-On: Technology Stack


Jenkins Pipeline
Compile
Unit Test
Build
Package
Archive
Deploy
Test
Scalable Build Infrastructure
Nexus
Maintains repo of Docker images
Dockers
Runs Jenkins as Containers
Kibana
Provides Actionable insights real-time
Driver Script
Listens to the Git Hub using web hooks and triggers the solution
Marathon
Orchestrates the overall solution
Mesos
Spawns & Manages Containers
Git Hub
Web Hook
Environments
Dev
QA
Pre-Prod
Prod

Simplify creation of Scalable CI Containers

1

Configure CI Master

Leverage existing Jenkins Master and Jobs

2

Place Build Requests

Jobs or requests are triggered based on identified events

3

Dynamic Slave are spawned

Builds are executed on containers running as slave nodes

4

Pipeline Execution

Slaves executes pipeline stages

5

Jobs executed automatically

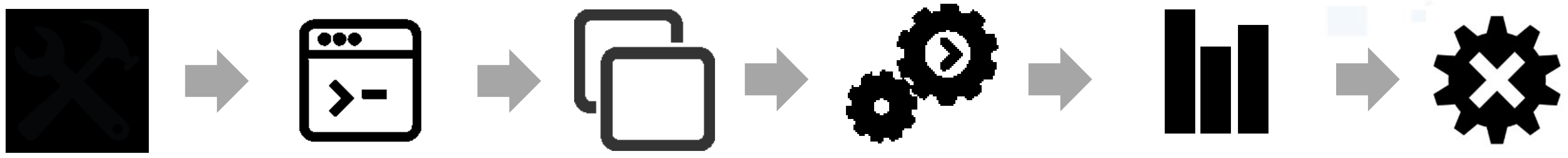
Post build console logs are archived and presented as dashboards

6

Teardown

After build completion, slave containers are destroyed

Build Pipeline as Code



1

Define Pipeline as Code

Descriptively define jobs in YAML files

2

Check-in Code

Focus on your actual software development

3

Auto run CI

Check in automatically detected and CI is triggered

4

Pipeline Execution

Headless CI executes pipeline stages

5

Dashboards

Post build console logs are archived and presented as dashboards

6

Teardown

After build completion, headless CI containers are destroyed

Benefits of Build-On

Scalable CI Set-up for Build Engineer

- Unlimited On-Demand Scalability
- Low Maintenance
- Easy Configuration and updates

Pipeline-as-Code for Developer

- Headless CI that scales on demand
- Codified pipelines increases productivity
- Improves feedback and quality by avoiding queues

Next Gen developer friendly CI solution that scales on demand, enabling focus on delivering quality code, faster.

Resources

Basics
Challenges of Build Engineer

Basics
Challenges of Developer

Demo Video
Build as a service

Demo Video
Pipeline as code

Visit us online at www.cognizant.com or follow us on Twitter: @Cognizant

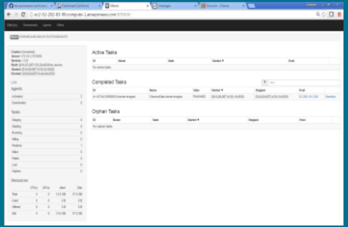
KEEP CHALLENGING™

Cognizant®

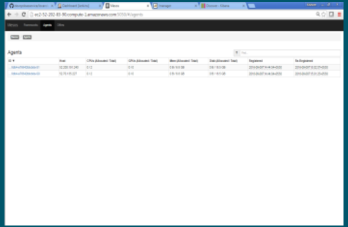
Appendix – Screen Shots

Deliver Continuous Integration in a Box

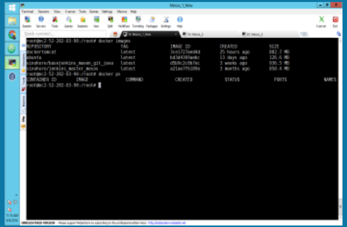
Mesos UI –
No Jobs are running



Shows the list of
Mesos slave.

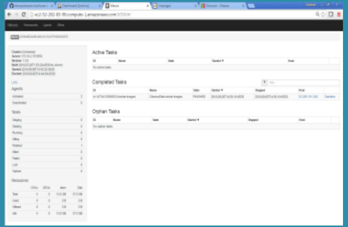


List of docker images and
running containers

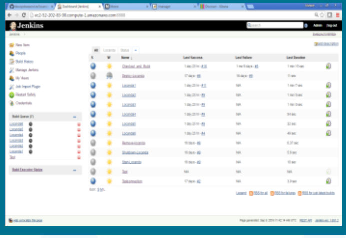


```
root@ec2-52-202-83-90:/root# docker images
REPOSITORY                            TAG       IMAGE ID        CREATED         SIZE
dockertomcat                          latest    3ce1727aed4d    25 hours ago    882.7 MB
ubuntu                                latest    bd3d4369aebc    13 days ago     126.6 MB
sinshere/basejenkins_maven_git_java   latest    d5b9c2c0b7ec    3 weeks ago     936.5 MB
sinshere/jenkins_master_mesos         latest    a21ae7f6389e    3 months ago    858.4 MB
root@ec2-52-202-83-90:/root# docker ps
CONTAINER ID    IMAGE    COMMAND    CREATED    STATUS    PORTS    NAMES
root@ec2-52-202-83-90:/root#
```

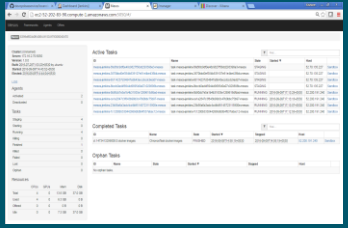
Dockerized Jenkins has
predefined plugins and
configurations



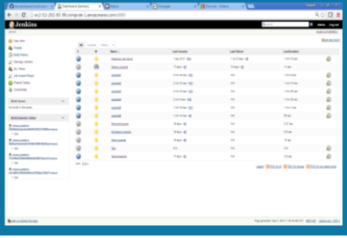
Jenkins started at 8888 port
Triggering Builds
note Zero executors



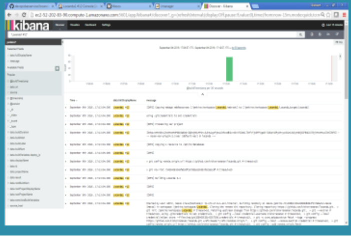
On-demand slaves are spinning
up. Mesos run the slave Jenkins
as containers



Jobs has been run and the
slaves are getting terminated.
Utilization returns to normal

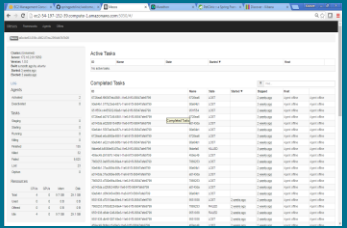


Logs of the job executed in
Kibana

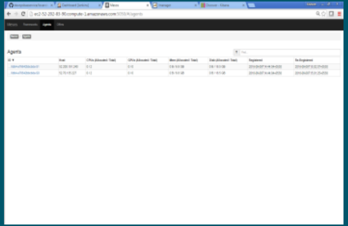


Generate Build Pipeline as Code

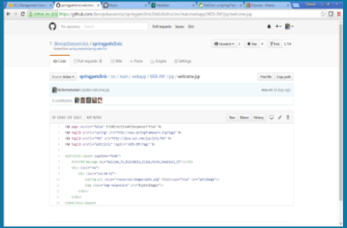
Mesos UI – Mesos Master showing all resources in the Idle state, ready for utilization



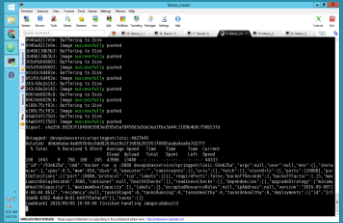
Shows the list of Mesos slave.



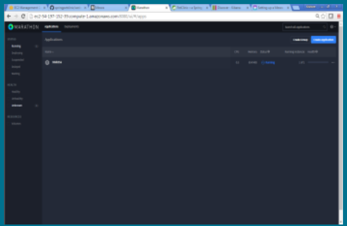
Original code in GitHub Repo Check H2 tag, it is same as the app deployment title



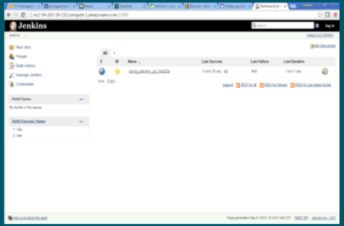
Git-Hub Triggers Web-hook and completes task



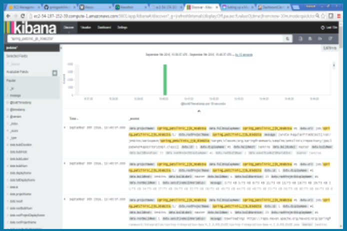
Marathon showing where the headless CI is running



Jenkins started in the port



Logs in Kibana searched by the Job name



Jobs get terminated in marathon