

Jenkins

Agenda:

1. Installing Jenkins, setting up, configuration
2. Security LDAP - authorization
3. How to schedule a job
4. integration:
 - Integration with GIT/GITHUB
 - Integration with Ant
 - Integration with Maven
 - Integration with Ansible
 - Integration with Nexus
5. Troubleshooting (location of log files of startup, failed plugin and etc)
6. how to schedule backups
7. restore from backups
8. Configure Slaves
9. Monitoring the workload
10. Managing plugins

What is Jenkins:

Leading open source continuous Integration server

Highly configurable system by itself

400+ community developed plugins

Multi platform

Easy to use

Free & Open source

<https://jenkins.io/index.html>

Features:

Jenkins provides continuous Integration services for software deployment.

Based on tomcat

Originally started as Hudson then renamed as Jenkins

leading open source continuous integration server.

Flexibility

Jenkins is a highly configurable system by itself.

400+ community developed plugins

Less Effort to Integrate existing existing test buckets.

Multi platform

Easy to use

Free & Open Source

Documentation: <https://jenkins.io/index.html>

Failing is Ok, but we need to fail fast

Catching a bug in test costs less than catching in production

Code quality review

Proper documentation

This can be done using custom scripts, but as the project increases it is difficult to maintain.

There was split in Hudson when oracle took over SUN,

90% Hudson and Jenkins are same

Build can be started by

Commit a version in version control

Scheduling via cron like mechanism

Tests can be executed every week/hour etc

If A fails do B, if A passes do C – jobs can be dependent like a pipeline

What is a build:

Build has many components – one of them is version control

No matters how many times a code changes, build has to happen, code has to be compiled.

Build also has to be tested

Communicate the results to all stake holders

- Product owner
- System admin
- QA
- Developer

History – can be maintained

Continuous delivery vs Continuous deployment

Continuous Deployment: deploying the product to customers continuously without approval, this is automatic

Continuous delivery: approval is needed, ready to deploy. This is not automatic

Test cases have to be automated for Continuous Integration

Download Jenkins:

Download LTS – long term support , not weekly build which is not safe

Download native packages not war packages

Jenkins rpm comes with webserver

Download Stable Jenkins rpm from <http://pkg.jenkins-ci.org/redhat-stable/>

1. Install it using rpm -ivh

```
root@ctx3p12 MISC]# rpm -ivh jenkins-2.9-1.1.noarch.rpm
```

```
warning: jenkins-2.9-1.1.noarch.rpm: Header V4 DSA/SHA1 Signature, key ID d50582e6: NOKEY
```

```
Preparing...
```

```
##### [100%]
```

```
Updating / installing...
```

```
1:jenkins-2.9-1.1
```

```
##### [100%]
```

2. Install Java using yum
3. Start the Jenkins service

```
[root@ctx3p12 MISC]# sudo systemctl start jenkins.service
```

```
[root@ctx3p12 MISC]# cat /var/lib/jenkins/secrets/initialAdminPassword
```

a45c1a99da7c49518ce07b7db91950e4

[root@ctx3p12 MISC]#

firewall-cmd --permanent --zone=public --add-port=8080/tcp

to open port 8080 on Jenkins server

Jenkins by default listens on port 8080

So open <http://FQDN:8080>

To change port

/etc/sysconfig/jenkins

http port to 8089

jenkins home

/var/lib/jenkins/config.xml

How to install

sudo yum -y install java

java is a prerequisite for Jenkins.

yum whatprovides service

sudo wget -O /etc/yum.repos.d/jenkins.repo <http://pkg.jenkins-ci.org/redhat/jenkins.repo>

- Downloads the repo file from Jenkins site

sudo rpm --import <https://jenkins-ci.org/redhat/jenkins-ci.org.key>

- Imports the key from Jenkins site for authentication

sudo yum install jenkins

- Downloads and installs the Jenkins packages from Jenkins repository

sudo systemctl status jenkins.service

- Start the service and check the status – it should be Active

Download Jenkins

Install it

Configure plugins

Install all of them

Install GIT related them

Installing all may slow down Jenkins

Its not recommended to enable auto refresh

URL/restart will restart Jenkins

Manage Jenkins Configure global security click on enable security

Authorization any one can do any thing

Security realm Jenkins own user database

LDAP is for corporate security to sign up using intranet id

Manage Jenkins project based authority enable everything for admin
enable all permissions

Setup security

Click on Enable security

Difference between save and apply

Apply will remain in the same, save will go back to previous page

Manage Jenkins -> configure system

No. of executors – no. of core you have on CPU

GIT hub server - create user id in git hub server

Add git hub server

Go to client

Git clone <https://github.com/RSCTFVT/>

Commit back from SVN

Add a file here

Git add *

Git commit -m "some commit"

Jenkins was developed by Kohsuke Kawaguchi.

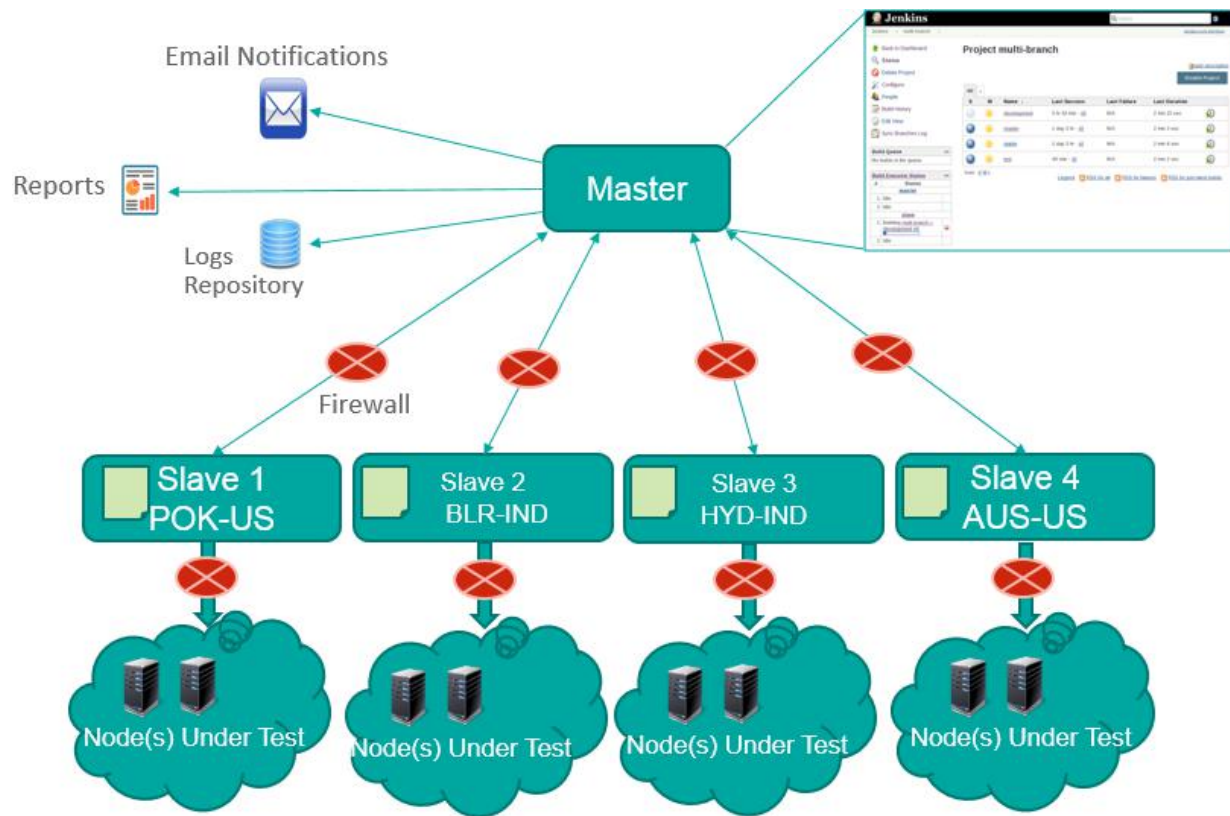
Backup of Jenkins:

1. Take backup of home directory using tar -cvf command
tar -cvf jenkins-backup-21dec.tar Jenkins
gzip jenkins-backup-21dec.tar
then copy it to other machine
Incase of crash, extract this file in newly created Jenkins server
2. Take backup of single job using xml
<http://ctx3p11.in.company.com:8080/job/RSCTBAT/config.xml>
open a job with config.xml extension and take backup in a file.

After installation:

```
[root@jupiter-vm751 ~]# cat /var/lib/jenkins/secrets/initialAdminPassword  
1b5561f487c14ea3b9c6c386eca33cfc
```

Configuring Slaves:



Jenkins Basic Terminology

- **Job**

- They all refer to runnable tasks that are controlled / monitored by Jenkins.
- Created by organizing **Steps** in particular sequence.



- **Build** : Result of one run of a Job.
- <https://wiki.jenkins-ci.org/display/JENKINS/Terminology>

GIT

Git is becoming popular in Devops world

Git is to keep track of changes

Ansible/puppet/chef/vagrant – files we store on GITHUB

Why GIT :

1. Build & Release - devops eng works with developers
2. For storing automation scripts
3. Open source - so many open source projects are hosted on GIT (Linux, chef, puppet etc)

System admins take backup by copying.

Cp file file_1

Cp file_1 file_2

Its difficult to track

Or we can name with date

That is also difficult to track

If there are many scripts – how to handle ?

There should be a centralized location to keep track of all changes – that is called version control system

Don't keep track

Save numbered zip file

Fomal version control

Diff , windiff – can see the difference

Even that is difficult

Developed by Linux torwalds – keep track of source code

Used for just for source code

Do not use for file storage

Its fast

Don't need access to server

Good at merging simultaneous changes

Everyone is using it

Bitbucket is alternative

GIT HUB – to store our scripts

Singup to github

Just give username and password

Login to it

Create new repository

Other CI tools:

Hudson

Bamboo

Microsoft tfs – team foundation server

Mercurial

Configuring slaves on the Jenkins master:

1. Create jenkins user on the slave

```
useradd -d "/home/jenkins" -c "jenkins user" -s "/bin/bash" -m  
jenkins
```

and set the password to slave as root

```
passwd Jenkins
```

2. Configure password less authentication on the server and slave for the Jenkins user

Login as Jenkins to master and execute

```
ssh-copy-id -i ~/.ssh/id_rsa.pub jenkins@rscthydnet1
```

3. Go to Jenkins manage nodes New node,
4. Select as Permanent agent

5. Select options as below

The screenshot shows the Jenkins 'New Node' configuration page. The left sidebar contains links: 'Back to Dashboard', 'Manage Jenkins', 'New Node' (highlighted with a red box), and 'Configure'. Below these are sections for 'Build Queue' (showing 'No builds in the queue.') and 'Build Executor Status' (showing 'master (offline)' and 'rscthydn1.in.ibm.com' with 10 idle executors). The main configuration area includes fields for 'Name' (slave1), 'Description', '# of executors' (10), 'Remote root directory' (/home/jenkins), 'Labels' (slave1), 'Usage' (Use this node as much as possible), 'Launch method' (Launch slave agents via SSH), 'Host' (slave1), 'Credentials' (jenkins/*), and 'Availability' (Keep this agent online as much as possible). The 'Node Properties' section has checkboxes for 'Date pattern for the BUILD_TIMESTAMP (build timestamp) variable', 'Environment variables', and 'Tool Locations'. A 'Save' button is at the bottom.

Once slave is added, make it online by

Clicking on

Manage Jenkins Manage Node Click on Slave Launch agent

This will make the slave online

Integration with GIT:

Install git on slave

Register your self on github and create a sample project

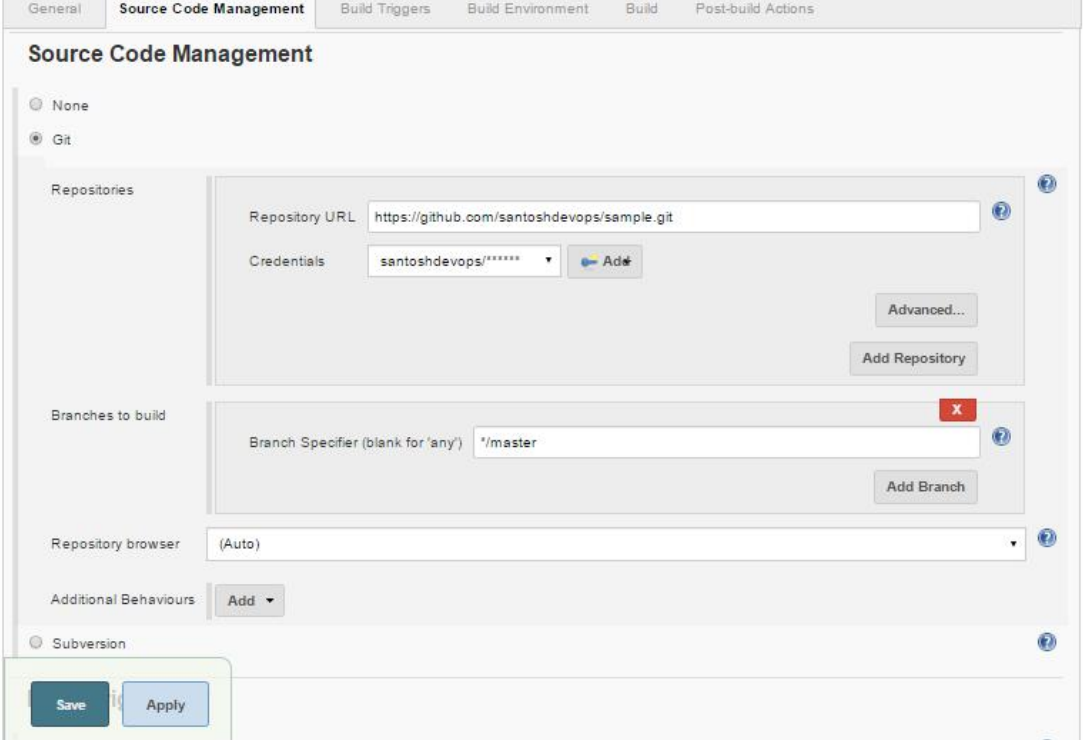
Add git plugin to Jenkins

Create a new job

Select Source Code Management as git

Provide the Repository URL as the git url (copy from clone or download option on git hub)

Provide the credentials (Same as github)



The screenshot shows the Jenkins configuration page for Source Code Management. The 'Source Code Management' tab is selected. Under the 'Git' radio button, the 'Repositories' section contains a 'Repository URL' field with the value 'https://github.com/santoshdevops/sample.git' and a 'Credentials' dropdown menu with 'santoshdevops/*****' selected. There are 'Advanced...' and 'Add Repository' buttons. The 'Branches to build' section has a 'Branch Specifier (blank for \'any\')' field with the value '*/master' and an 'Add Branch' button. The 'Repository browser' dropdown is set to '(Auto)'. The 'Additional Behaviours' section has an 'Add' button. At the bottom, there are 'Save' and 'Apply' buttons.

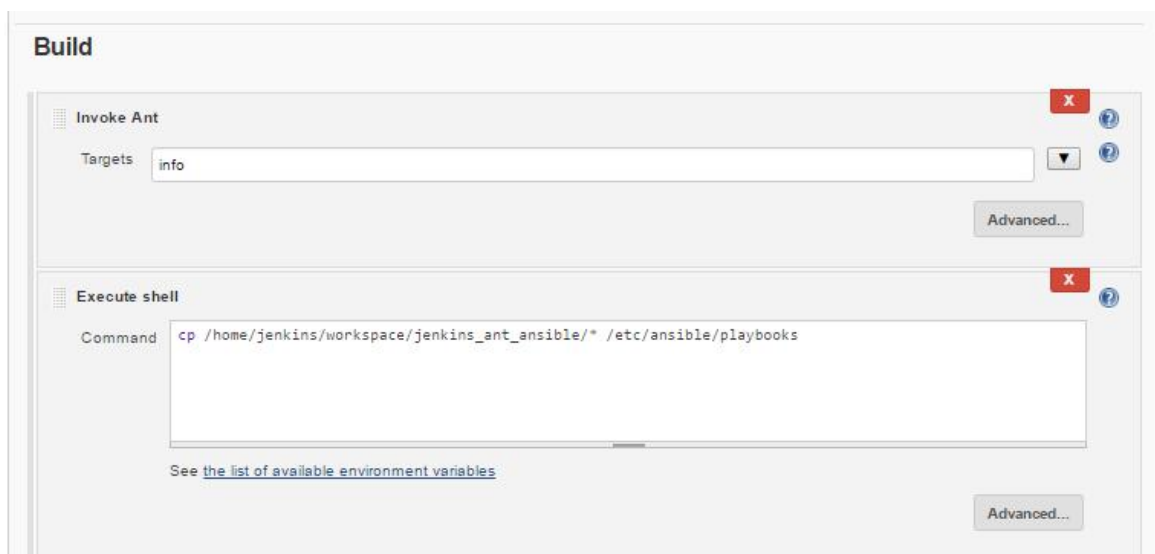
Select build triggers as Poll SCM for every minute - * * * * *

To run this project on a particular slave

Click the checkbox

Restrict where this project can be run and select label of the slave.

Ant:



The image shows a Jenkins 'Build' configuration page. It contains two steps:

- Invoke Ant**: The 'Targets' field is set to 'info'. There is an 'Advanced...' button to the right.
- Execute shell**: The 'Command' field contains the text `cp /home/jenkins/workspace/jenkins_ant_angular/* /etc/ansible/playbooks`. Below the field is a link that says 'See the list of available environment variables'. There is an 'Advanced...' button to the right.

Ansible

Invoke Ansible Playbook

Playbook path: /etc/ansible/playbooks/apache.yml

Inventory:

- ☒ Do not specify Inventory
- ☐ File or host list
- ☐ Inline content

Host subset:

Credentials: jenkins/***** Add

☐ sudo

Advanced...

Add build step

Integration with Nexus build repository:

Nexus is a build repository stores the builds in a hierarchical directory structure.

Nexus can

- Gives a centralized repository for managing all popular component formats.

- Gain insight into component security, license, and quality issues.

- Improve productivity by efficiently distributing components to developers around the corner, or around the world.

- Modernize software development with intelligent staging and release functionality.

- Sleep comfortably with world-class support and training.

How to install Nexus:

```
cd /usr/local/
```

```
wget http://sonatype.org/downloads/nexus-latest-bundle.tar.gz
```

```
gunzip nexus-latest-bundle.tar.gz
```

```
tar -xvf nexus-latest-bundle.tar
```

```
mv nexus-2.14.3-02 sonatype-work /usr/local
```

```
ln -s nexus-2.14.3-02 nexus
```

```
cd nexus
```

```
export RUN_AS_USER=root
```

./nexus start

Make sure nexus is running by executing “ ps -ef | grep -i nexus
”

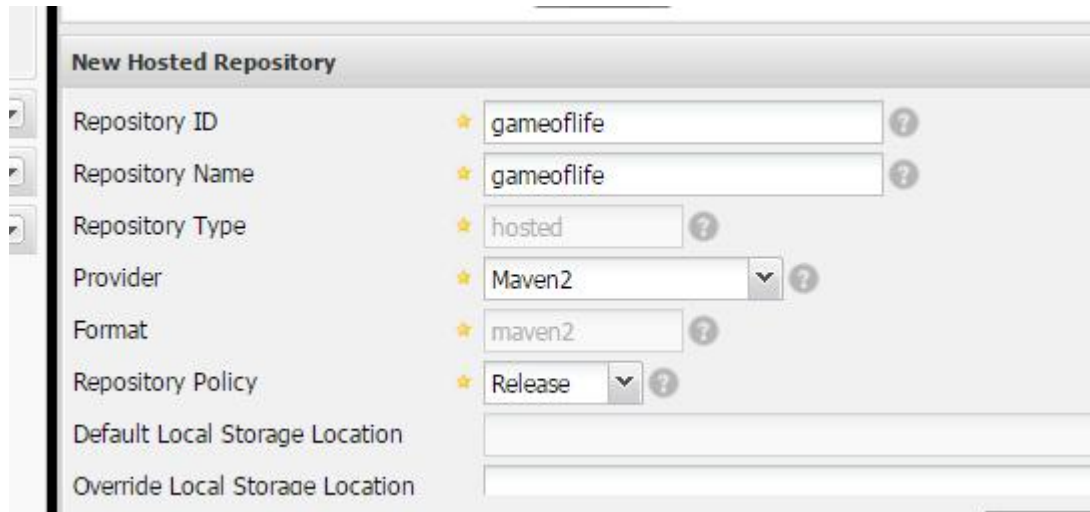
admin/admin123 – default password for nexus login.

Open nexus GUI:

URL: <http://rscthydnet2:8081/nexus/#welcome>

Create a repository in Nexus to host builds

Go to repository → add hosted repository



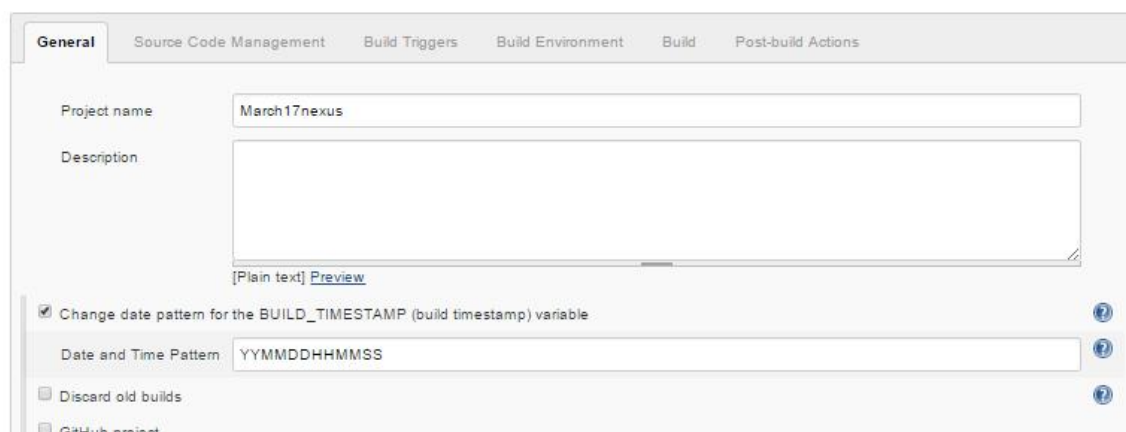
The screenshot shows the 'New Hosted Repository' configuration page in Nexus. The form includes the following fields:

- Repository ID: gameoflife
- Repository Name: gameoflife
- Repository Type: hosted
- Provider: Maven2
- Format: maven2
- Repository Policy: Release
- Default Local Storage Location: (empty)
- Override Local Storage Location: (empty)

In the Jenkins add the plugin **Nexus Artifact Uploader**

Also install the zen timestamp plugin to use \$BUILD_TIMESTAMP variable in the job

Configure the job as below



The screenshot shows the Jenkins job configuration page with the following settings:

- Project name: March17nexus
- Description: (empty)
- Change date pattern for the BUILD_TIMESTAMP (build timestamp) variable: ☒
- Date and Time Pattern: YYMMDDHHMMSS
- Discard old builds: ☐
- GitHub orient: ☐

Nexus artifact uploader

Nexus Details

Nexus Version: NEXUS2

Protocol: HTTP

Nexus URL: cdx2p01.in.ibm.com:8081/nexus

Credentials: admin***** [Add](#)

Groupid: DEV

Version: \$BUILD_ID

Repository: gameoflife

Artifacts

Artifact	
Artifactid	\$BUILD_TIMESTAMP
Type	war
Classifier	
File	gameoflife-web/target/gameoflife.war

[Add](#)

[Add build step](#)

After job is executed nexus creates repos as below

Releases hosted ANALYZE maven2 Release In Service

Snapshots hosted maven2 Snapshot In Service

gameoflife

[Browse Index](#) **[Browse Storage](#)** [Configuration](#) [Routing](#) [Summary](#) [Artifact Upload](#)

[Refresh](#) Path Lookup:

- gameoflife
 - DEV
 - 170376090341
 - 4
 - 170376090341-4.war
 - 170376090341-4.war.md5
 - 170376090341-4.war.sha1
 - archetype-catalog.xml

And it can be accessed using a link

Index of /repositories/gameoflife/DEV/170376090341/4

Name	Last Modified	Size	Description
Parent Directory			
170376090341-4.war	Fri Mar 17 09:21:16 IST 2017	3192487	
170376090341-4.war.md5	Fri Mar 17 09:21:17 IST 2017	32	
170376090341-4.war.sha1	Fri Mar 17 09:21:17 IST 2017	40	

Jenkins logs:

Jenkins logs will be stored under :

/var/log/jenkins/jenkins.log

Some commonly used plugins:

Extended choice parameter

Build with parameters

Rebuild now

Html report generator

Junit report generator

Thinkbackup

Monitoring

Crontab syntax:

* * * * *

minutes 0-59

hours 0-12

day of the month 1-31

month of the year 1-12

day of the week sun,mon,tue,wed,thu,fri,sat

command/script

crontab - utility to schedule the jobs

10AM - good morning

9AM, 5PM - cleanup the logs

1st of month --> msg

10 * * * * echo Hi --> every 10 minutes

* 10 * * * date >> /file --> every day 10AM

* 17 * * * /scr --> every day 17th hour - 5PM

* 10,17 * * * /script

* 10,11,12,17 * * /script

* * 1 * * /scr --> every month on 1st

To have upstream job and downstream job:

Use a plugin called Jobfan in

While creating job, we can have a condition like, if Job A is built successfully then only start the job B.

If build is successful, then go for test.