Import/Export configuration

1. **Copy Jobs directory**

`JENKINS_HOME` has a fairly obvious directory structure that looks like the following:

```
JENKINS_HOME
 +- config.xml     (jenkins root configuration)
 +- *.xml          (other site-wide configuration files)
 +- userContent    (files in this directory will be served under your http://server/userContent/)
 +- fingerprints   (stores fingerprint records)
 +- nodes          (slave configurations)
 +- plugins        (stores plugins)
 +- secrets        (secretes needed when migrating credentials to other servers)
 +- workspace (working directory for the version control system)
     +- [JOBNAME] (sub directory for each job)
 +- jobs
     +- [JOBNAME]      (sub directory for each job)
         +- config.xml     (job configuration file)
         +- latest         (symbolic link to the last successful build)
         +- builds
             +- [BUILD_ID]     (for each build)
                 +- build.xml      (build result summary)
                 +- log            (log file)
                 +- changelog.xml  (change log)
```

All the settings, build logs, artifact archives are stored under the JENKINS_HOME directory. Simply archive this directory to make a back up. Similarly, restoring the data is just replacing the contents of the JENKINS_HOME directory from a back up.

Back ups can be taken without stopping the server, but when you restore, please do stop the server.

Moving/copying/renaming jobs

1. Move a job from one installation of Jenkins to another by simply copying the corresponding job directory.
2. Make a copy of an existing job by making a clone of a job directory by a different name.
3. Rename an existing job by renaming a directory. Note that the if you change a job name you will need to change any other job that tries to call the renamed job.

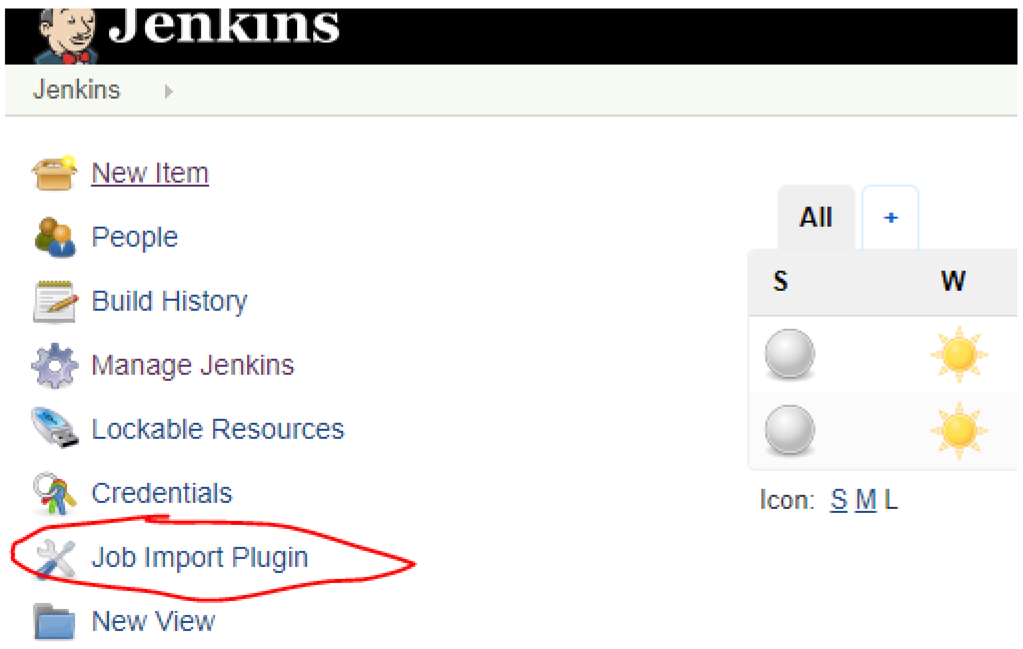
```
cd $JENKINS_HOME/jobs
tar -czf jobs.tgz jobs

copy the files to another server using SCP command

scp -r jobs.tgz user@server:$JENKINS_HOME
extract the job folder and restart instance
```

2. Using Job Import plugin

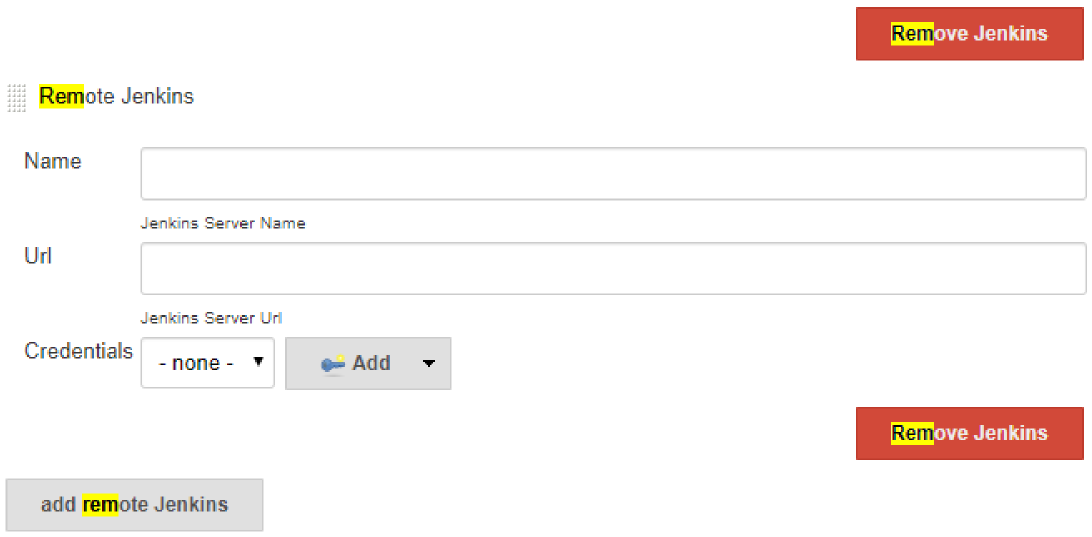
We require Job Import plugin to be installed to import job from one machine to another.

Install plugin ManageJenkins>Mange plugins > Job import

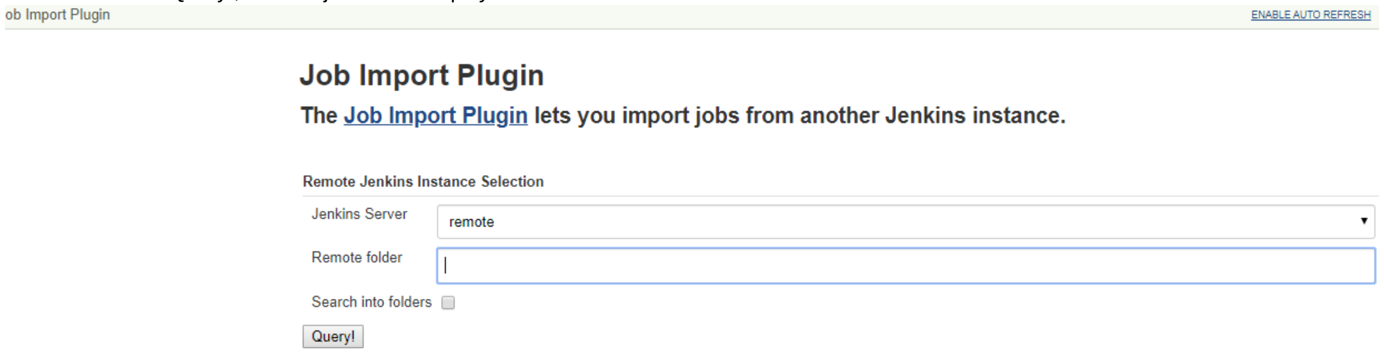
Once Jenkins is installed, Job Import plugin will be displayed in the menu.



Goto Managejenkins>configure system >add remote server details



Once we click on Query!, Jenkins job will be displayed from the remote Jenkins.



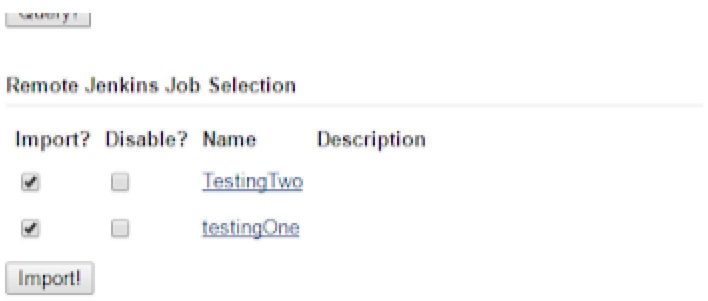
Job Import Plugin

The Job Import Plugin lets you import jobs from another Jenkins instance.

Remote Jenkins Instance Selection

Jenkins Server	remote
Remote folder	
Search into folders	☐

Jobs will be imported once user clicks on Import.



Remote Jenkins Job Selection

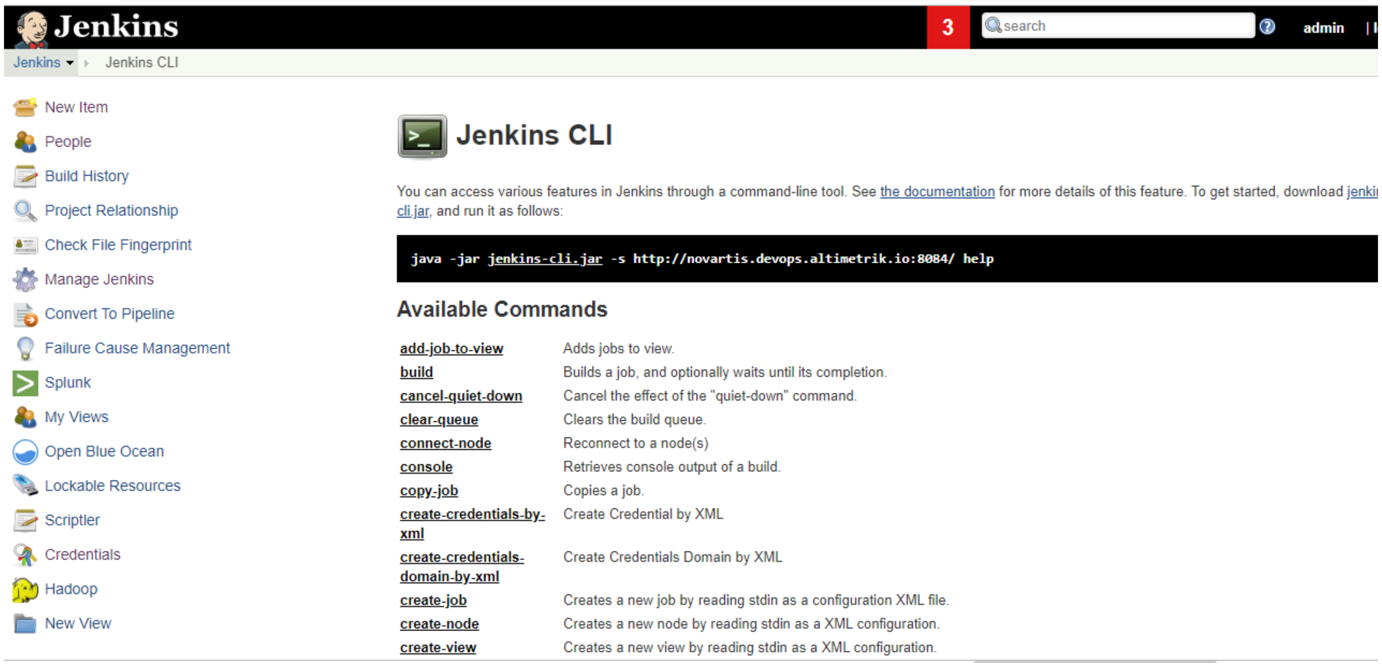
Import?	Disable?	Name	Description
✓	☐	TestingTwo	
✓	☐	testingOne	

3. Using Jenkins CLI

steps Import and export jobs in jenkins using Cli

Open Jenkins and Go to the job which you want to export.

Manage jenkins > jenkins Cli



Jenkins CLI

You can access various features in Jenkins through a command-line tool. See the documentation for more details of this feature. To get started, download jenki cli.jar, and run it as follows:

```
java -jar jenkins-cli.jar -s http://novartis.devops.altimetrik.io:8084/ help
```

Available Commands

add-job-to-view	Adds jobs to view.
build	Builds a job, and optionally waits until its completion.
cancel-quiet-down	Cancel the effect of the "quiet-down" command.
clear-queue	Clears the build queue.
connect-node	Reconnect to a node(s)
console	Retrieves console output of a build.
copy-job	Copies a job.
create-credentials-by-xml	Create Credential by XML
create-credentials-domain-by-xml	Create Credentials Domain by XML
create-job	Creates a new job by reading stdin as a configuration XML file.
create-node	Creates a new node by reading stdin as a XML configuration.
create-view	Creates a new view by reading stdin as a XML configuration.

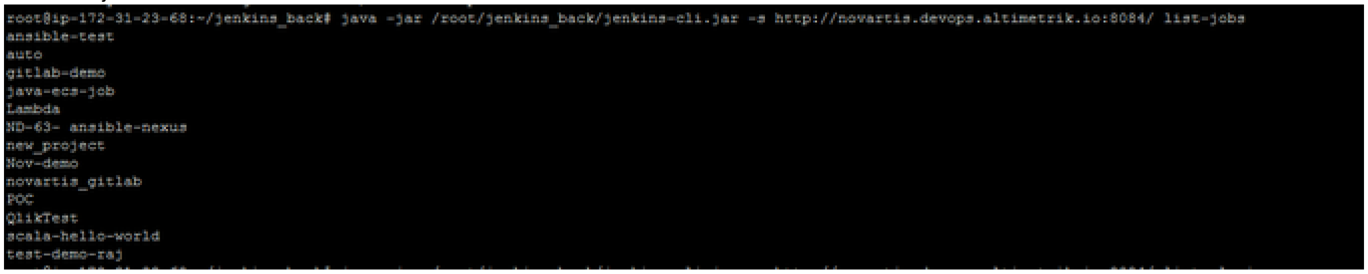
Now download CLI Jar as shown below and save.

Once you have jar ready use below command to export job in XML format.

List-jobs

```
java -jar /root/jenkins_back/jenkins-cli.jar -s
http://novartis.devops.altimetrik.io:8084/ list-jobs
```

Xml file for jobs look like:

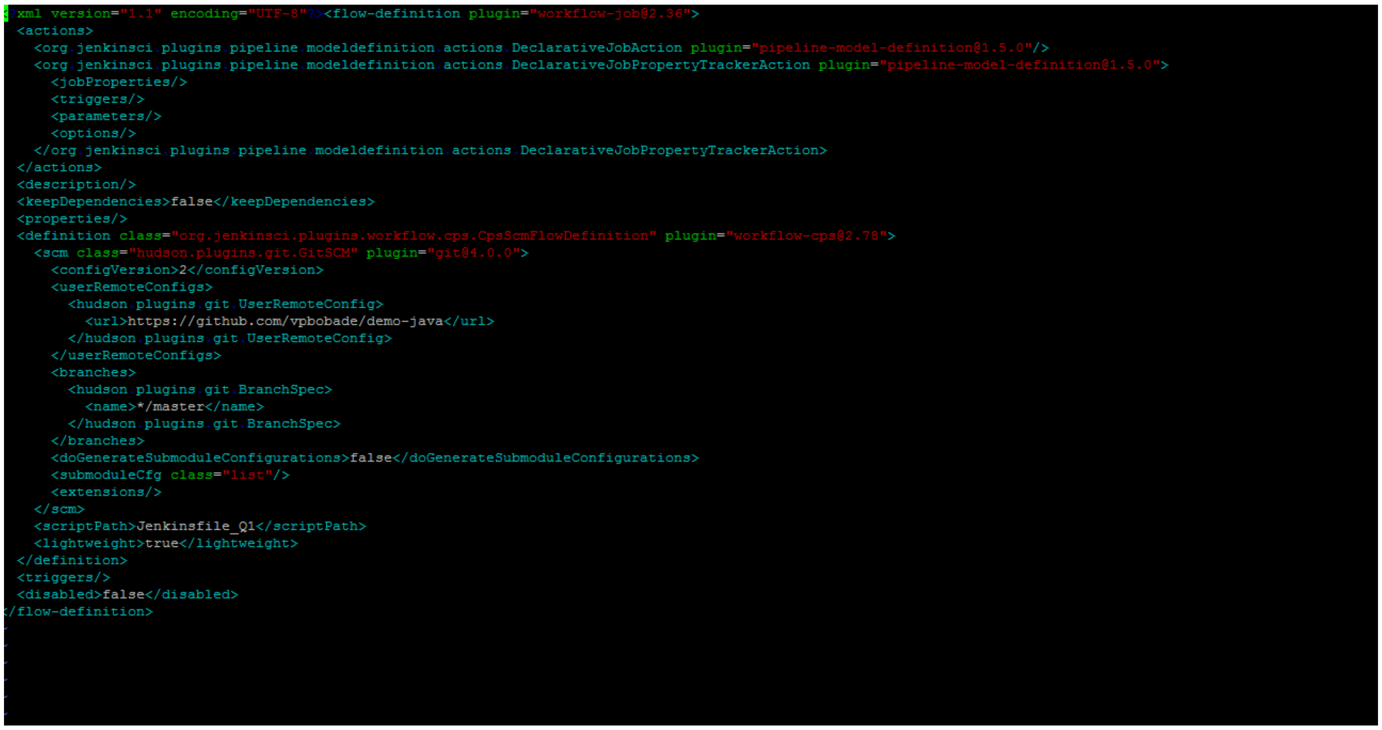


```
root@ip-172-31-23-68:~/jenkins_back# java -jar /root/jenkins_back/jenkins-cli.jar -s http://novartis.devops.altimetrik.io:8084/ list-jobs
ansible-test
auto
gitlab-demo
java-ecs-job
Lambda
ND-63- ansible-nexus
new_project
Nov-demo
novartis_gitlab
POC
QlikTest
scala-hello-world
test-demo-raj
```

Get-job

```
java -jar /root/jenkins_back/jenkins-cli.jar -s
http://novartis.devops.altimetrik.io:8084/ get-job auto >jenkins.xml
```

Get-job xml files looks like for particular job

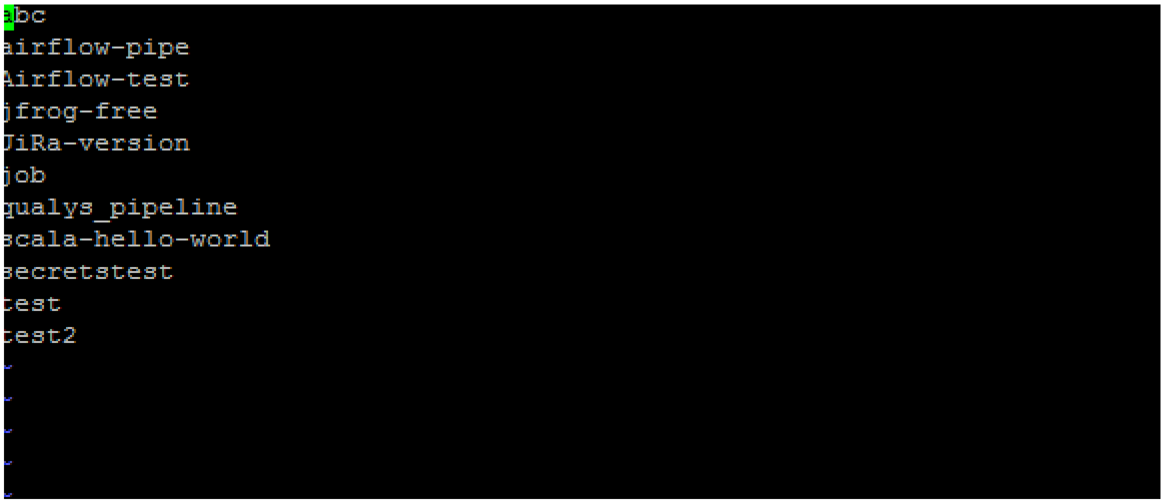


```
<?xml version="1.1" encoding="UTF-8"?><flow-definition plugin="workflow-job@2.36">
  <actions>
    <org.jenkinsci.plugins.pipeline.modeldefinition.actions.DeclarativeJobAction plugin="pipeline-model-definition@1.5.0"/>
    <org.jenkinsci.plugins.pipeline.modeldefinition.actions.DeclarativeJobPropertyTrackerAction plugin="pipeline-model-definition@1.5.0">
      <jobProperties/>
      <triggers/>
      <parameters/>
      <options/>
    </org.jenkinsci.plugins.pipeline.modeldefinition.actions.DeclarativeJobPropertyTrackerAction>
  </actions>
  <description/>
  <keepDependencies>false</keepDependencies>
  <properties/>
  <definition class="org.jenkinsci.plugins.workflow.cps.CpsScmFlowDefinition" plugin="workflow-cps@2.78">
    <scm class="hudson.plugins.git.GitSCM" plugin="git@4.0.0">
      <configVersion>2</configVersion>
      <userRemoteConfigs>
        <hudson.plugins.git.UserRemoteConfig>
          <url>https://github.com/vpbobade/demo-java</url>
        </hudson.plugins.git.UserRemoteConfig>
      </userRemoteConfigs>
      <branches>
        <hudson.plugins.git.BranchSpec>
          <name>*/master</name>
        </hudson.plugins.git.BranchSpec>
      </branches>
      <doGenerateSubmoduleConfigurations>false</doGenerateSubmoduleConfigurations>
      <submoduleCfg class="list"/>
      <extensions/>
    </scm>
    <scriptPath>Jenkinsfile_Q1</scriptPath>
    <lightweight>true</lightweight>
  </definition>
  <triggers/>
  <disabled>false</disabled>
</flow-definition>
~
~
~
~
~
~
```

To import

```
java -jar jenkins-cli.jar -s newjenkinsserver create-job test<
jenkins.xml
```

XML file for jobs look like:



```
abc
airflow-pipe
Airflow-test
jfrog-free
JiRa-version
job
qualys_pipeline
scala-hello-world
secretstest
test
test2
~
~
~
~
~
```

Shell script for import and export configuration

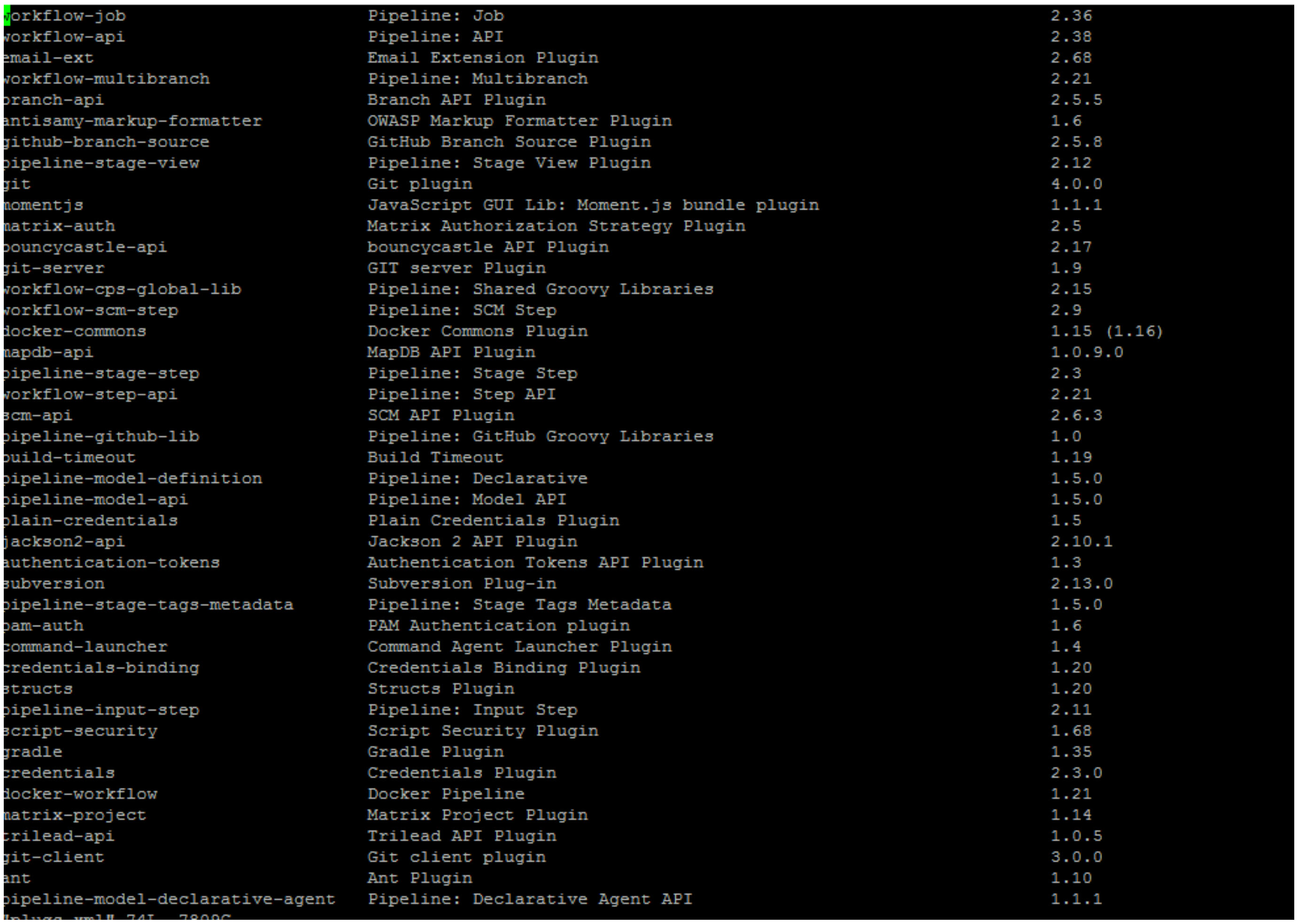
```
#!/bin/bash

for i in $(java -jar /root/test/jenkins-cli.jar -s
http://54.88.7.101:8080/  list-jobs);
  do
    java -jar /root/test/jenkins-cli.jar -s http://54.88.7.101:8080/
get-job ${i} > ${i}.xml
  done
for f in *.xml
  do
    java -jar /root/test/jenkins-cli.jar -s http://52.200.249.165:8080/
-auth admin:admin create-job ${f%.xml} < ${f}
  done
```

To Import Plugins on Remote machine:

To List plugins CMD:

```
java -jar /root/jenkins_back/jenkins-cli.jar -s
http://52.200.249.165:8080/ -auth admin:admin list-plugins
```

Xml for plugins look like:


```
workflow-job                       Pipeline: Job                                                    2.36
workflow-api                       Pipeline: API                                                    2.38
email-ext                          Email Extension Plugin                                           2.68
workflow-multibranch               Pipeline: Multibranch                                            2.21
branch-api                         Branch API Plugin                                                2.5.5
antisamy-markup-formatter          OWASP Markup Formatter Plugin                                    1.6
github-branch-source               GitHub Branch Source Plugin                                      2.5.8
pipeline-stage-view                Pipeline: Stage View Plugin                                      2.12
git                                Git plugin                                                       4.0.0
momentjs                           JavaScript GUI Lib: Moment.js bundle plugin                      1.1.1
matrix-auth                        Matrix Authorization Strategy Plugin                             2.5
bouncycastle-api                   bouncycastle API Plugin                                          2.17
git-server                         GIT server Plugin                                                1.9
workflow-cps-global-lib            Pipeline: Shared Groovy Libraries                                2.15
workflow-scm-step                  Pipeline: SCM Step                                               2.9
docker-commons                     Docker Commons Plugin                                            1.15 (1.16)
mapdb-api                          MapDB API Plugin                                                 1.0.9.0
pipeline-stage-step                Pipeline: Stage Step                                             2.3
workflow-step-api                  Pipeline: Step API                                               2.21
scm-api                            SCM API Plugin                                                   2.6.3
pipeline-github-lib                Pipeline: GitHub Groovy Libraries                                1.0
build-timeout                      Build Timeout                                                    1.19
pipeline-model-definition          Pipeline: Declarative                                            1.5.0
pipeline-model-api                 Pipeline: Model API                                              1.5.0
plain-credentials                  Plain Credentials Plugin                                         1.5
jackson2-api                       Jackson 2 API Plugin                                             2.10.1
authentication-tokens              Authentication Tokens API Plugin                                 1.3
subversion                         Subversion Plug-in                                               2.13.0
pipeline-stage-tags-metadata       Pipeline: Stage Tags Metadata                                    1.5.0
pam-auth                           PAM Authentication plugin                                        1.6
command-launcher                   Command Agent Launcher Plugin                                    1.4
credentials-binding                Credentials Binding Plugin                                       1.20
structs                            Structs Plugin                                                   1.20
pipeline-input-step                Pipeline: Input Step                                             2.11
script-security                    Script Security Plugin                                           1.68
gradle                             Gradle Plugin                                                    1.35
credentials                        Credentials Plugin                                               2.3.0
docker-workflow                    Docker Pipeline                                                  1.21
matrix-project                     Matrix Project Plugin                                            1.14
trilead-api                        Trilead API Plugin                                               1.0.5
git-client                         Git client plugin                                                3.0.0
ant                                Ant Plugin                                                       1.10
pipeline-model-declarative-agent   Pipeline: Declarative Agent API                                  1.1.1
"plugs.xml" 74L, 7809C
```

To install plugins:

```
java -jar /root/jenkins_back/jenkins-cli.jar -s
http://52.200.249.165:8080/ -auth admin:admin install-plugin
${Plugin_Name}
```

Automation shell script to import plugins on remote machine:

```
#!/bin/bash

for i in $(java -jar /root/test/jenkins-cli.jar -s
http://54.88.7.101:8080/  list-plugins);
  do
    java -jar /root/test/jenkins-cli.jar -s http://54.88.7.101:8080/
get-job ${i} > ${i}.xml
  done
for f in *.xml
  do
    java -jar /root/test/jenkins-cli.jar -s http://52.200.249.165:8080/
-auth admin:admin install-plugin ${f%.xml} < ${f}
  done
```