

Cài đặt Jenkins trong kubernetes

Jenkins được cài với Helm chart.

```
curl https://raw.githubusercontent.com/helm/helm/main/scripts/get-helm-3 | bash
```

Kiểm tra:

```
helm version
```

Chuẩn bị trước khi cài Jenkins vào kubernetes

1. Tạo namespaces cho jenkins

```
kubectl create namespace jenkins
```

Xem secret đang tồn tại:

```
Kubectl get secret -n jenkins
```

2. Tạo secret để tạo kubeconfig cho kubectl trong pod jenkins thông qua secret: (secret đọc file, cung cấp cho tạo aws/credentials trong pod jenkins trên kubernetes, cho phép sử dụng aws thông qua thông tin đăng nhập lưu ở aws/credentials cho jenkins. Trong lệnh dùng cần dùng "\$HOME" thay vì "~" bởi vì "~" là shell nên không được hiểu trong câu lệnh kubectl, còn "\$HOME" là biến môi trường nên sẽ luôn đúng cho các trường hợp).

```
kubectl create secret generic aws-credentials --from-file=credentials=$HOME/.aws/credentials -n jenkins
```

3. Chuẩn bị image cho jenkins bằng docker, lưu trên docker hub.

Login docker trước khi đẩy image lên docker hub

docker login (mở cửa url trong cửa sổ login, tiếp theo dán mã được cấp từ cửa sổ login vào trang login trên trình duyệt)

```
docker build -t vothinhuydt/jenkins:txu .
```

```
docker push vothinhuydt/jenkins:txu
```

4. Tạo thư mục để mount vào jenkins lưu trữ dữ liệu lâu dài khi pod jenkins bị xóa tạo lại...

```
sudo mkdir -p /mnt/data/jenkins
```

```
sudo chown -R 1000:1000 /mnt/data/jenkins
```

Reboot máy để có hiệu lực: **reboot**

```
ls -ldn /mnt/data/jenkins
```

```
ls -ld /mnt/data/jenkins
```

5. Cài đặt Jenkins với helm

```
helm install jenkins ./jenkins-chart -n jenkins
```

* Gỡ cài đặt Jenkins

```
helm uninstall jenkins -n jenkins
```

6. Sau khi cài đặt jenkins bằng helm, chạy jenkins thiết lập ban đầu cho jenkins

a. jenkins hiện tại được cài đặt và chạy trong kubernetes và được ingress vào theo url:

jenkins.txuapp.com. Do đó từ node control-plane, cài đặt cloudflare và tunnel url: jenkins.txuapp.com đến NodePort của Ingress Controller ở port(**xxxxx**:8080). Các bước từ tạo tunnel, route dns, và run tunnel như sau:

- Tạo tunnel cho jenkins: (sẽ tạo ra một **id** và file **id.json** của tunnel, dùng thông tin này để tạo file chạy tunnel route dns)

```
cloudflared tunnel create jenkins-tunnel
```

Tạo một route dns cho gắn với tunnel vừa tạo

cloudflared tunnel route dns jenkins-tunnel jenkins.txuapp.com

Tạo file “cloudflare-jenkins-tunnel.yml” config thông tin để chạy tunnel, nội dung:

tunnel: 4a06d22d-240f-4e63-b772-58502933d65f

credentials-file: ~/.cloudflared/4a06d22d-240f-4e63-b772-58502933d65f.json

ingress:

- hostname: jenkins.txuapp.com

- service: http://localhost:31179

- service: http_status:404

Chạy tunnel với file config:

cloudflared tunnel --config cloudflare-jenkins-tunnel.yml run

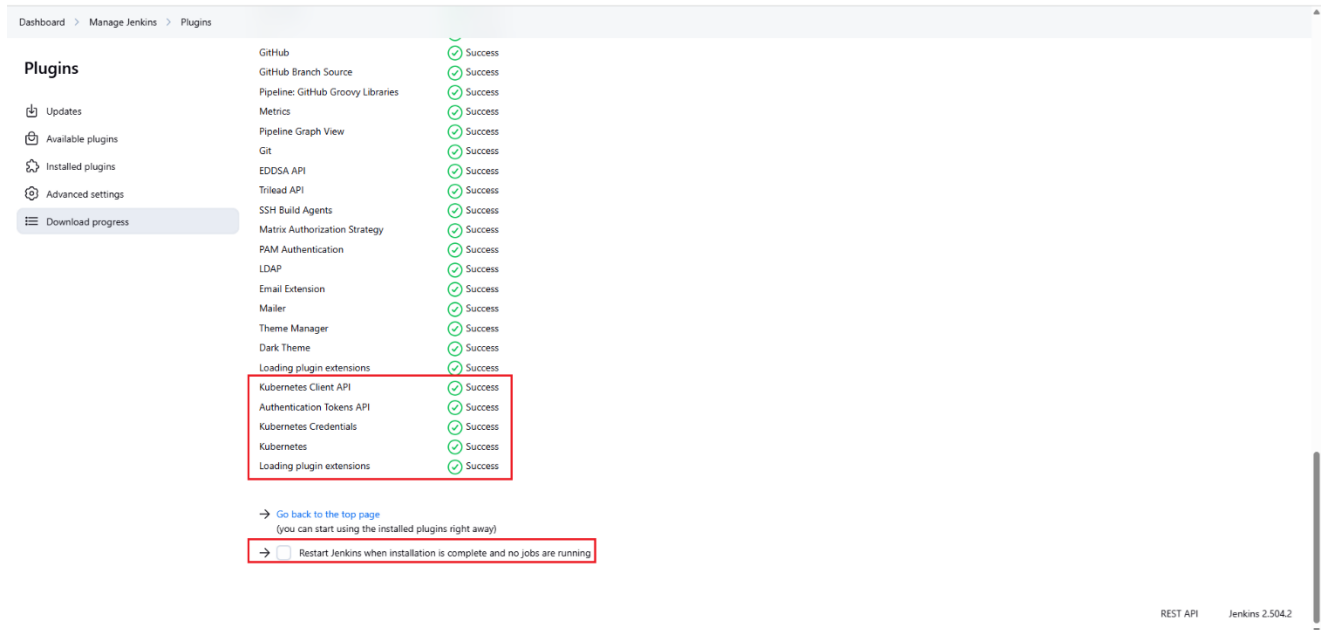
b. Lấy mật khẩu để đăng nhập và thiết lập ban đầu cho jenkins:

kubectl exec -it -n jenkins <jenkins-pod-name> -- cat /var/jenkins_home/secrets/initialAdminPassword

c. Tiếp theo cài đặt plugin kubernetes cho jenkins trên UI của jenkins (việc cài đặt plugin kubernetes cho phép jenkins chạy các jenkins agent và giao tiếp với jenkins controller thông qua dns nội bộ của jenkins “http://jenkins.jenkins.svc.cluster.local:8080”):

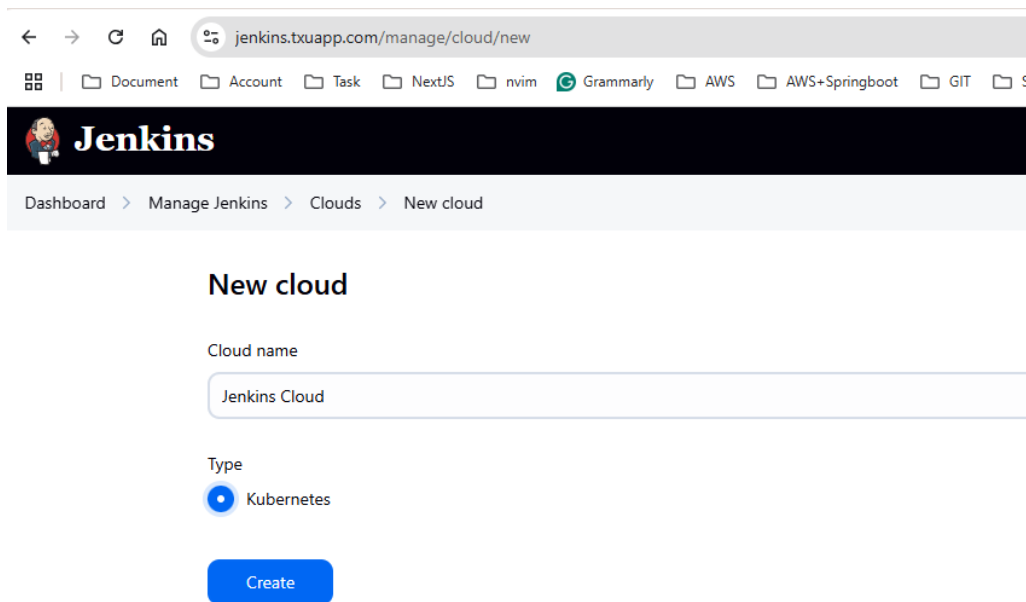
The screenshot shows the Jenkins web interface at jenkins.txuapp.com. The 'Plugins' tab is selected in the top navigation bar. On the left sidebar, the 'Available plugins' section is active. The main content area displays a list of plugins. The 'Kubernetes' plugin (version 4353.vb1_47977da_9417) is highlighted with a red box and has a blue checkmark in the 'Install' column, indicating it is already installed. Other plugins listed include 'Kubernetes Client API', 'Kubernetes Credentials', 'Kubernetes CLI', 'Kubernetes Credentials Provider', 'Kubernetes Pipeline', and 'GitLab Credentials - Kubernetes Integration'. The 'Released' column shows the time since each plugin was last updated.

Install	Name	Released
<input checked="" type="checkbox"/>	Kubernetes 4353.vb1_47977da_9417 Cloud Providers Cluster Management kubernetes Agent Management This plugin integrates Jenkins with Kubernetes	12 days ago
<input type="checkbox"/>	Kubernetes Client API 6.10.0-251.v5565f100500 kubernetes Library plugins (for use by other plugins) Kubernetes Client API plugin for use by other Jenkins plugins.	3 mo 10 days ago
<input type="checkbox"/>	Kubernetes Credentials 192.v4d5b_1c429d17 kubernetes credentials Common classes for Kubernetes credentials	3 mo 10 days ago
<input type="checkbox"/>	Kubernetes CLI 1.364.vadef0cb8b823 kubernetes Configure kubectl for Kubernetes	2 mo 22 days ago
<input type="checkbox"/>	Kubernetes Credentials Provider 1.281.v331e3f5a_05a_9 kubernetes credentials Provides a read only credentials store backed by Kubernetes.	16 days ago
<input type="checkbox"/>	Kubernetes Pipeline : DevOps Steps 1.6 pipeline kubernetes	6 yr 4 mo ago
<input type="checkbox"/>	GitLab Credentials - Kubernetes Integration 424.vd4c848a_61813 kubernetes gitlab Integrates gitlabToken credential type from the gitlab-branch-source-plugin with the k8s credential provider.	3 mo 19 days ago



d. Cấu hình jenkins cloud kiểu kubernetes để cho phép các jenkins agent có thể giao tiếp jenkins controller. Do các jenkins agent chạy trong cluster cùng với jenkins controller nên cần cấu hình “URL Jenkins: <http://jenkins.jenkins.svc.cluster.local:8080>”. Lưu ý: kết thúc url có đặt dấu “.” Để báo cho dns trong kubernetes biết đây là url đầy đủ mà không thêm phần searches, gây sai url và dns không phân giải chính xác. Mặt khác nếu không dùng url đầy đủ như trên thì cũng xảy ra tình trạng searches ở các container khác nhau sẽ không có những hậu tố “ví dụ: cluster.local” theo yêu cầu. Ngoài ra vì các jenkins agent và controller đều giao tiếp nội bộ nên cách hợp lý nhất là dùng dns nội bộ thay vì dùng dns public “jenkins.txuapp.com” vì khi đó yêu cầu phải mở thêm port cho jenkins ra bên ngoài, các agent phải truy cập từ bên ngoài thay vì truy cập ngay bên trong, điều đó không hợp lý và cần nhiều cấu hình hơn.

Để cấu hình jenkins cloud kiểu kubernetes thì trước hết cần tạo một cloud kiểu kubernetes:



Chú ý: hai mục “Kubernetes Namespaces” và “Jenkins URL”

Dashboard > Manage Jenkins > Clouds > New cloud

☐ Disable https certificate check ?

Kubernetes Namespace
jenkins

Agent Docker Registry ?

☐ Inject restricted PSS security context in agent container definition ?

Credentials
- none -

+ Add

Connected to Kubernetes v1.29.15

Test Connection

☐ WebSocket ?
☐ Direct Connection ?

Jenkins URL ?
http://jenkins.jenkins.svc.cluster.local:8080

Save

e. Tạo và chạy một job kiểu “multi branch”

Dashboard > New Item

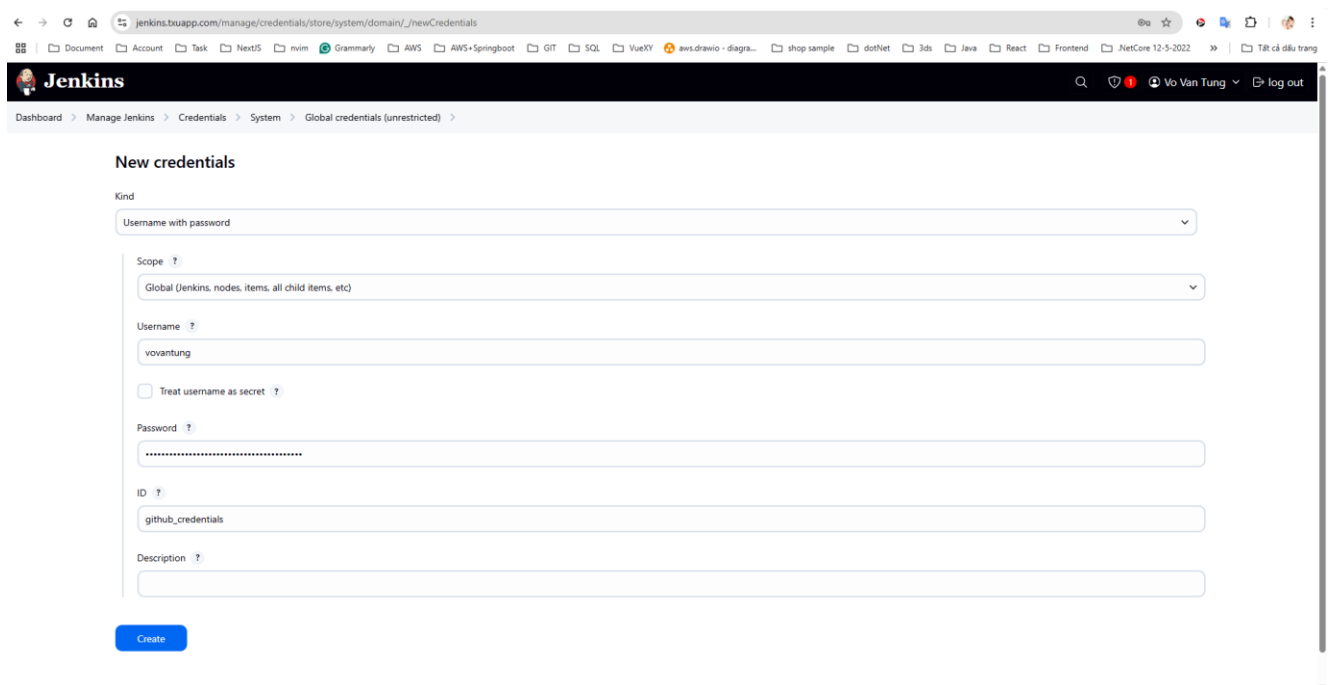
Enter an item name
pl1

Select an item type

- ☐ Freestyle project
Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.
- ☐ Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.
- ☐ Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.
- ☐ Folder
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.
- ☒ Multibranch Pipeline
Creates a set of Pipeline projects according to detected branches in one SCM repository.
- ☐ Organization Folder
Creates a set of multibranch project subfolders by scanning for repositories.

OK

Để cấu hình job kiểu “multibranch pipeline”, cần tạo **credentials** để đăng nhập github, thông tin cần có là username, passwork (token generate từ github)



The screenshot shows the Jenkins 'New credentials' configuration page. The 'Kind' is set to 'Username with password'. The 'Scope' is 'Global (Jenkins, nodes, items, all child items, etc)'. The 'Username' is 'vovantung'. There is an unchecked checkbox for 'Treat username as secret'. The 'Password' field is masked with asterisks. The 'ID' is 'github_credentials'. The 'Description' field is empty. A blue 'Create' button is at the bottom.

Kind: Username with password

Scope: Global (Jenkins, nodes, items, all child items, etc)

Username: vovantung

☐ Treat username as secret

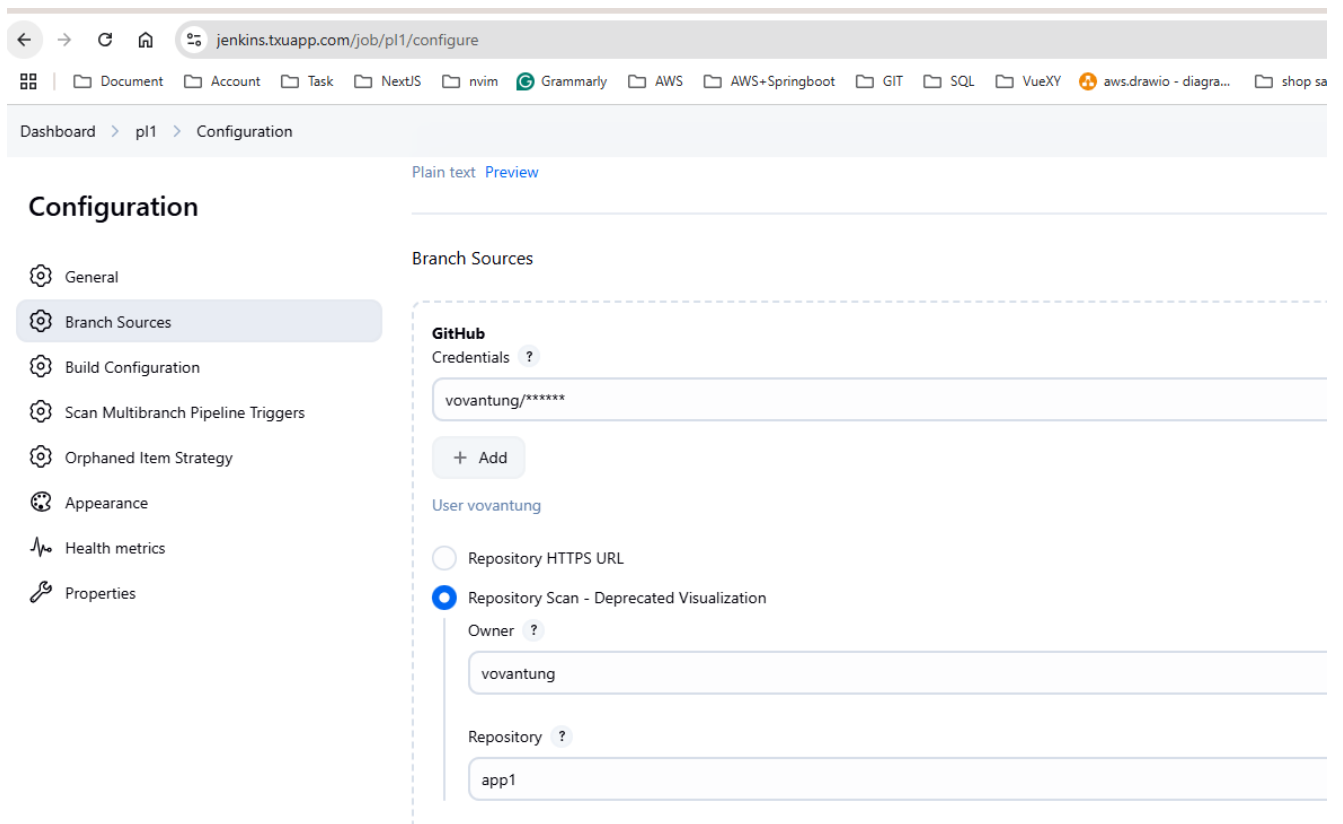
Password: [masked]

ID: github_credentials

Description: [empty]

Create

Cấu hình job



The screenshot shows the Jenkins job configuration page for 'pl1'. The left sidebar has a menu with 'General', 'Branch Sources', 'Build Configuration', 'Scan Multibranch Pipeline Triggers', 'Orphaned Item Strategy', 'Appearance', 'Health metrics', and 'Properties'. The 'Branch Sources' section is active. It shows a 'GitHub' branch source with 'Credentials' set to 'vovantung/*****'. There is a '+ Add' button. Below, it shows 'User vovantung' and two radio buttons: 'Repository HTTPS URL' (unselected) and 'Repository Scan - Deprecatcd Visualization' (selected). The 'Owner' is 'vovantung' and the 'Repository' is 'app1'.

jenkins.tuapp.com/job/pl1/configure

Dashboard > pl1 > Configuration

Plain text Preview

Configuration

- General
- Branch Sources
- Build Configuration
- Scan Multibranch Pipeline Triggers
- Orphaned Item Strategy
- Appearance
- Health metrics
- Properties

Branch Sources

GitHub

Credentials: vovantung/*****

+ Add

User vovantung

☐ Repository HTTPS URL

☒ Repository Scan - Deprecatcd Visualization

Owner: vovantung

Repository: app1

Mục “Behaviours” add thêm **Filter by name (with wildcards)**, nhập Include: “main” nếu muốn chỉ trigger job khi push code trên nhánh “main”

Filter by name (with wildcards)

Include ?

main

Exclude ?

Add ▾

Mục “**Build Configuration**” chọn “**Script Path**” là đường dẫn Jenkinsfile trong thư mục gốc của code project

Build Configuration

Mode

by Jenkinsfile

Script Path ?

jenkins/Jenkinsfile

Mục “Scan Multibranch Pipeline Triggers”:

Scan Multibranch Pipeline Triggers

☒ Periodically if not otherwise run ?

Interval ?

1 minute

Khi đã tạo job kiểu “multibranch pipeline” như trên, và khi có sự kiện push code lên branch được chỉ định của repo (ví dụ push code lên branch “**main**” của repo “<https://github.com/vovantung/app1.git>”). Job được trigger sẽ chạy pipeline được định nghĩa trong Jenkinsfile. Khi cloud kubernetes được cài đặt trong jenkins thì các jenkins agent sẽ thực hiện các stage và giao tiếp với jenkins controller qua “Jenkins URL” mà ta đã chỉ định trong phần cấu hình cloud của jenkins (phần trên đã chỉ định giao tiếp qua dns nội bộ “<http://jenkins.jenkins.svc.cluster.local.:8080>”). Tuy nhiên một số jenkins agent ngoài giao tiếp với jenkins

controller qua dns nội bộ đã được chỉ định (full url) thì chúng cũng cần thực hiện với giao tiếp với bên ngoài internet (chẳng hạn kaniko cần giao tiếp với docker.io để pull image “[eclipse-temurin:17-jdk-alpine](#)” cho việc build image từ dockerfile). Do đó với container kaniko cần đặt “ndots:1” để tránh việc dns nội bộ thực hiện gắn thêm phần searches (chẳng hạn “cluster.local”) gây ra không thể phân giải dns này.

Trong mô hình jenkins agent thực hiện nhiệm vụ được giao từ jenkins controller và các jenkins agent giao tiếp trở lại với jenkins controller thông qua dns nội bộ “[http://jenkins.jenkins.svc.cluster.local:8080](#)” thì nhiệm vụ chủ yếu được thực hiện bởi các jenkins agent (là các container riêng biệt và có thể chia sẻ dữ liệu chung).

```
18 - name: kubect1
19   image: yothinhuydt/jenkins-agent-kubect1:txu
20   command: ['cat']
21   tty: true
22   volumeMounts:
23     - name: workspace-volume
24       mountPath: /workspace
25 - name: maven
26   image: maven:3.9-eclipse-temurin-17
27   command: ['cat']
28   tty: true
29   volumeMounts:
30     - name: workspace-volume
31       mountPath: /workspace
32 - name: kaniko
33   image: gcr.io/kaniko-project/executor:debug
34   command: ['cat']
35   tty: true
36   volumeMounts:
37     - name: workspace-volume
38       mountPath: /workspace
39   env:
40     - name: AWS_ACCESS_KEY_ID
41       valueFrom:
42         secretKeyRef:
43           name: aws-creds-kaniko
44           key: AWS_ACCESS_KEY_ID
45     - name: AWS_SECRET_ACCESS_KEY
46       valueFrom:
47         secretKeyRef:
48           name: aws-creds-kaniko
49           key: AWS_SECRET_ACCESS_KEY
50     - name: AWS_REGION
51       valueFrom:
52         secretKeyRef:
53           name: aws-creds-kaniko
54           key: AWS_REGION
55   volumes:
56     - name: workspace-volume
57       emptyDir: {}
58   ---
59 }
60 }
```

Khi đó jenkins controller chỉ đóng vai trò trung tâm giao nhiệm vụ (nằm trong các stage, chẳng hạn clone code, build code với maven, build code thành image với kaniko và push lên các repo như ecr, docker hub;...) cho các jenkins agent, các jenkins agent thực hiện và trả lời lại jenkins controller.

Chi tiết cụ thể các agent được định nghĩa và thực hiện nhiệm vụ cụ thể (xem thêm trong Jenkinsfile)

Ví dụ, trong jenkins agent (kaniko) thực hiện nhiệm vụ nhận dữ liệu chung được chia sẻ từ jenkins agent (maven), nhận dockerfile và file app2.jar đã build bởi jenkins agent (maven) ở stage trước đó, và thực hiện trước pull image “eclipse-temurin:17-jdk-alpine” từ internet, sau đó build thành image chứa ứng dụng app1.jar và push image mới build lên ecr. Kaniko được thiết kế có khả năng build image mà không cần docker meacon, và push image lên ecr hoặc docker hub... Quá trình push image lên ecr chẳng hạn cần phải có quyền push image, do đó cần gắn các biến môi trường (thông qua secret trong kubernetes) chứa thông tin xác thực với aws thực hiện quyền push image lên ecr.

```
kubectl create secret generic aws-creds-kaniko --from-literal=AWS_ACCESS_KEY_ID=aws-access-key-id --from-literal=AWS_SECRET_ACCESS_KEY=aws-secret-key --from-literal=AWS_REGION=ap-southeast-1 -n jenkins
```

Việc thực hiện gắn secret cho kaniko được thực hiện bằng cách tạo một secret dạng secret key và gắn thông tin đó vào container kaniko ở bước định nghĩa kamoko trong jenkinsfile.

```
13     serviceAccountName: jenkins-sa
14     containers:
15     - name: jnlp
16       image: jenkins/inbound-agent:latest
17       tty: true
18     - name: kubectl
19       image: vothinhuydt/jenkins-agent-kubectl:txu
20       command: ['cat']
21       tty: true
22       volumeMounts:
23       - name: workspace-volume
24         mountPath: /workspace
25     - name: maven
26       image: maven:3.9-eclipse-temurin-17
27       command: ['cat']
28       tty: true
29       volumeMounts:
30       - name: workspace-volume
31         mountPath: /workspace
32     - name: kaniko
33       image: gcr.io/kaniko-project/executor:debug
34       command: ['cat']
35       tty: true
36       volumeMounts:
37       - name: workspace-volume
38         mountPath: /workspace
39       env:
40       - name: AWS_ACCESS_KEY_ID
41         valueFrom:
42           secretKeyRef:
43             name: aws-creds-kaniko
44             key: AWS_ACCESS_KEY_ID
45       - name: AWS_SECRET_ACCESS_KEY
46         valueFrom:
47           secretKeyRef:
48             name: aws-creds-kaniko
49             key: AWS_SECRET_ACCESS_KEY
50       - name: AWS_REGION
51         valueFrom:
52           secretKeyRef:
53             name: aws-creds-kaniko
54             key: AWS_REGION
55     volumes:
```

Tương tự kaniko, thì jenkins agent (kubectl) cũng cần thực hiện việc kết nối với **kubernetes api server** để có thể dùng command “**kubectl**”. Để thực hiện được việc này, cần thực hiện tạo một “**ServiceAccount**” tên “” có quyền “**rbac.authorization.k8s.io/v1**” tức có quyền lấy thông tin xác thực của kubernetes.

“apiVersion: v1

kind: ServiceAccount

metadata:

name: jenkins-sa

namespace: {{ .Release.Namespace }}

apiVersion: rbac.authorization.k8s.io/v1

kind: ClusterRoleBinding

metadata:

name: jenkins-sa-cluster-admin

roleRef:

apiGroup: rbac.authorization.k8s.io

kind: ClusterRole

name: cluster-admin

subjects:

- kind: ServiceAccount

name: jenkins-sa

namespace: {{ .Release.Namespace }}

Khi gán “serviceAccountName: jenkins-sa” cho một pod thì các container của pod sẽ chứa các quyền xác thực với k8s, khi đó chỉ cần cài kubectl cli, kubectl cli sẽ thực hiện lấy quyền xác thực được cấp và cấu hình cho kubeconfig (nằm trong `/var/jenkins_home/.kube/config`). Có nghĩa là, các container được gán quyền “`rbac.authorization.k8s.io/v1`” và cài đặt kubectl cli sẽ thực hiện các lệnh “kubectl”.

```
1 pipeline {
2   agent {
3     kubernetes {
4       yaml """
5       apiVersion: v1
6       kind: Pod
7       spec:|
8         dnsPolicy: ClusterFirst
9         dnsConfig:
10          options:
11            - name: ndots
12              value: "1"
13         serviceAccountName: jenkins-sa
14         containers:
15           - name: jnlp
16             image: jenkins/inbound-agent:latest
17             tty: true
18           - name: kubectl
19             image: vothinhuydt/jenkins-agent-kubectl:txu
20             command: ['cat']
21             tty: true
22             volumeMounts:
23               - name: workspace-volume
24                 mountPath: /workspace
25           - name: maven
26             image: maven:3.9-eclipse-temurin-17
27             command: ['cat']
28             tty: true
29             volumeMounts:
30               - name: workspace-volume
31                 mountPath: /workspace
```