



S5.B.01 Phase 4

Déploiement de services

Maxence Lagourgue

17 décembre 2025

Table des matières

I	Outils	2
II	Création du cluster avec Rancher	3
II.A	Installation de rancher	3
II.A.1	Accès distant au cluster (Optionnel)	3
II.A.2	Installation d'helm	3
II.A.3	Installation de kubectl	3
II.A.4	Pod K3S	4
II.A.5	Pour arrêter/pauser Rancher	4
II.B	Réinitialisation du cluster	5

I Outils

Dans cette partie, les outils utilisés seront :

- Rancher pour la gestion des clusters
- RKE2 pour la mise en œuvre Kubernetes des nœuds de travail
- k3s pour le cluster Rancher
- kubectl pour la gestion des ressources
- Helm pour la gestion des applications

Plus tard, si nous avons le temps, nous utiliserons Ansible pour automiser la chaîne de production Rancher.

II Création du cluster avec Rancher

Pour utiliser Rancher, plusieurs méthodes d'installation s'offrent à nous. L'une avec docker, l'autre en tant que noeud Kubernetes. Les autres installations reposent sur l'utilisation d'un Cloud Provider ainsi que Terraform donc inutile dans notre cas.

Exemple de tutorial : [Tutorial Rancher 2025](#)

Faire le gitlab en tant qu'application kubernetes/rancher.

II.A Installation de rancher

```
curl -sfL https://get.k3s.io | INSTALL_K3S_VERSION="v1.24.10+k3s1" sh -s - server  
--cluster-init
```

II.A.1 Accès distant au cluster (Optionnel)

<IP_OF_LINUX_MACHINE> est l'IP de la machine distante sur laquelle se trouve le cluster.

```
scp root@<IP_OF_LINUX_MACHINE>:/etc/rancher/k3s/k3s.yaml ~/.kube/config  
nano ~/.kube/config
```

II.A.2 Installation d'helm

```
sudo apt-get install curl gpg apt-transport-https --yes  
  
curl -fsSL https://packages.buildkite.com/helm-linux/helm-debian/gpgkey | gpg --dearmor |  
sudo tee /usr/share/keyrings/helm.gpg > /dev/null  
  
echo "deb [signed-by=/usr/share/keyrings/helm.gpg]  
https://packages.buildkite.com/helm-linux/helm-debian/any/ any main" | sudo tee  
/etc/apt/sources.list.d/helm-stable-debian.list  
  
sudo apt-get update  
sudo apt-get install helm
```

II.A.3 Installation de kubectl

```
sudo apt-get install -y apt-transport-https ca-certificates curl gnupg  
  
curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.34/deb/Release.key | sudo gpg --dearmor -o  
/etc/apt/keyrings/kubernetes-apt-keyring.gpg
```

```

sudo chmod 644 /etc/apt/keyrings/kubernetes-apt-keyring.gpg

echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg]
https://pkgs.k8s.io/core:/stable:/v1.34/deb/ /' | sudo tee
/etc/apt/sources.list.d/kubernetes.list
sudo chmod 644 /etc/apt/sources.list.d/kubernetes.list

sudo apt-get update
sudo apt-get install -y kubectl

```

II.A.4 Pod K3S

<Hostname> correspond au nom de domaine utilisé pour contacter le pod rancher.

```

export KUBECONFIG=/etc/rancher/k3s/k3s.yaml

kubectl create namespace cattle-system
kubectl config set-context --current --namespace=cattle-system

kubectl apply -f
https://github.com/cert-manager/cert-manager/releases/download/v1.19.2/cert-manager.crds.yaml

helm repo add rancher-latest https://releases.rancher.com/server-charts/latest

helm repo add jetstack https://charts.jetstack.io

helm repo update

helm install cert-manager jetstack/cert-manager \
--namespace cert-manager \
--create-namespace

helm install rancher rancher-latest/rancher \
--namespace cattle-system \
--set hostname=10.0.1.3.sslip.io \
--set replicas=1 \
--set bootstrapPassword=$iutinfo

```

Il faut maintenant attendre car l'installation nécessite quelques minutes. On peut vérifier avec `kubectl get pods -n cattle-system`. Une fois fait, on se connecte à la page et on récupère le mot de passe :

```

kubectl get secret --namespace cattle-system bootstrap-secret -o
go-template='{{.data.bootstrapPassword|base64decode}}{{"\n"}}'

```

On définit un nouveau mot de passe qui est `qJHiA@wwaagi46U`.

II.A.5 Pour arrêter/pauser Rancher

```

kubectl scale --replicas=0 deployment/rancher -n cattle-system

```

Cela permet d'arrêter temporairement le pod Rancher.

II.B Réinitialisation du cluster

Pour revenir à l'état 0 du cluster, il est possible de :

```
rm -rf /var/lib/rancher/k3s/server/db/etcd  
/usr/local/bin/k3s-killall.sh  
systemctl restart k3s.service
```