Déploiement live d'un cluster K3s sur Linux

Présentation rapide du contexte

Ce déploiement vise à offrir une introduction concrète à Kubernetes via l'installation et l'utilisation d'un cluster léger K3s sous Linux. K3s est une version allégée de Kubernetes, adaptée à des environnements de test ou à des ressources limitées, simplifiant ainsi les premiers pas dans l'écosystème Kubernetes.

```
oot@debian:~# curl -sfL https://get.k3s.io | sh -
[INFO]  Finding release for channel stable
[INFO]  Using v1.32.5+k3s1 as release
[INFO]  Downloading hash https://github.com/k3s-io/k3s/releases/download/v1.32.5+k3s1/sha256sum-amd64.txt
[INFO]  Downloading binary https://github.com/k3s-io/k3s/releases/download/v1.32.5+k3s1/k3s
[INFO]  Verifying binary download
[INFO]  Installing k3s to /usr/local/bin/k3s
[INFO]  Skipping installation of SELinux RPM
[INFO]  Creating /usr/local/bin/kubectl symlink to k3s
[INFO]  Creating /usr/local/bin/crictl symlink to k3s
[INFO]  Creating /usr/local/bin/ctr symlink to k3s
[INFO]  Creating killall script /usr/local/bin/k3s-killall.sh
[INFO]  Creating uninstall script /usr/local/bin/k3s-uninstall.sh
[INFO]  env: Creating environment file /etc/systemd/system/k3s.service.env
[INFO]  systemd: Creating service file /etc/systemd/system/k3s.service
[INFO]  systemd: Enabling k3s unit
reated symlink /etc/systemd/system/multi-user.target.wants/k3s.service → /etc/systemd/system/k3s.service.
[INFO]  Host iptables-save/iptables-restore tools not found
[INFO]  Host ip6tables-save/ip6tables-restore tools not found
[INFO]  systemd: Starting k3s
```

Test

```
root@debian:/home/enzo# cp /etc/rancher/k3s/k3s.yaml ~/.kube/config
root@debian:/home/enzo# chown $(id -u):$(id -g) ~/.kube/config
root@debian:/home/enzo# export KUBECONFIG=~/.kube/config
root@debian:/home/enzo# kubectl get nodes
NAME     STATUS   ROLES                  AGE    VERSION
debian   Ready    control-plane,master   3m3s   v1.32.5+k3s1
root@debian:/home/enzo#
```

On peut voir que ça fonctionne et qu'un node remonte

Intégration de Helm

L'installation rapide de Helm a facilité le déploiement de charts, ensembles structurés de ressources Kubernetes.

```
curl -fsSL -o get_helm.sh
https://raw.githubusercontent.com/helm/helm/main/scripts/get-helm-3
chmod +x get_helm.sh
./get_helm.sh
```

`helm version`
Pour tester que l'installation c'est faite correctement

Kubernetes Dashboard via Helm

À l'aide de Helm, nous avons déployé le chart officiel du Kubernetes Dashboard, interface graphique officielle permettant une gestion intuitive du cluster.

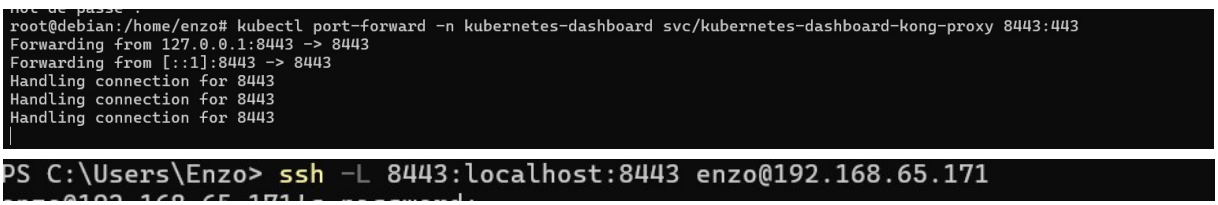
```
kubectl apply -f - <<EOF
apiVersion: v1
kind: ServiceAccount
metadata:
  name: admin-user
  namespace: kubernetes-dashboard
EOF

kubectl apply -f - <<EOF
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: admin-user
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: cluster-admin
subjects:
- kind: ServiceAccount
  name: admin-user
  namespace: kubernetes-dashboard
EOF
```

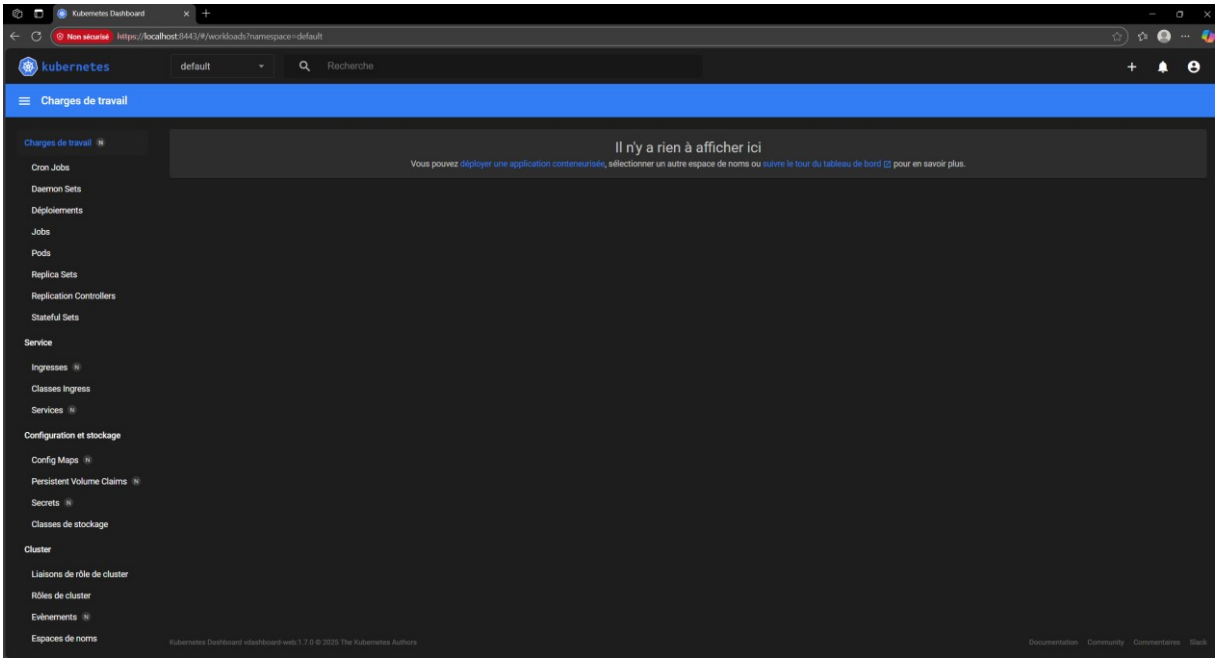
Ensuite j'ai utilisé cette commande pour récupérer le token de connexion

```
kubectl -n kubernetes-dashboard create token admin-user
```

Accès au Dashboard avec Port Forwarding

Pour accéder au Kubernetes Dashboard depuis notre poste local, nous avons réalisé un port forwarding à l'aide des commandes suivantes :

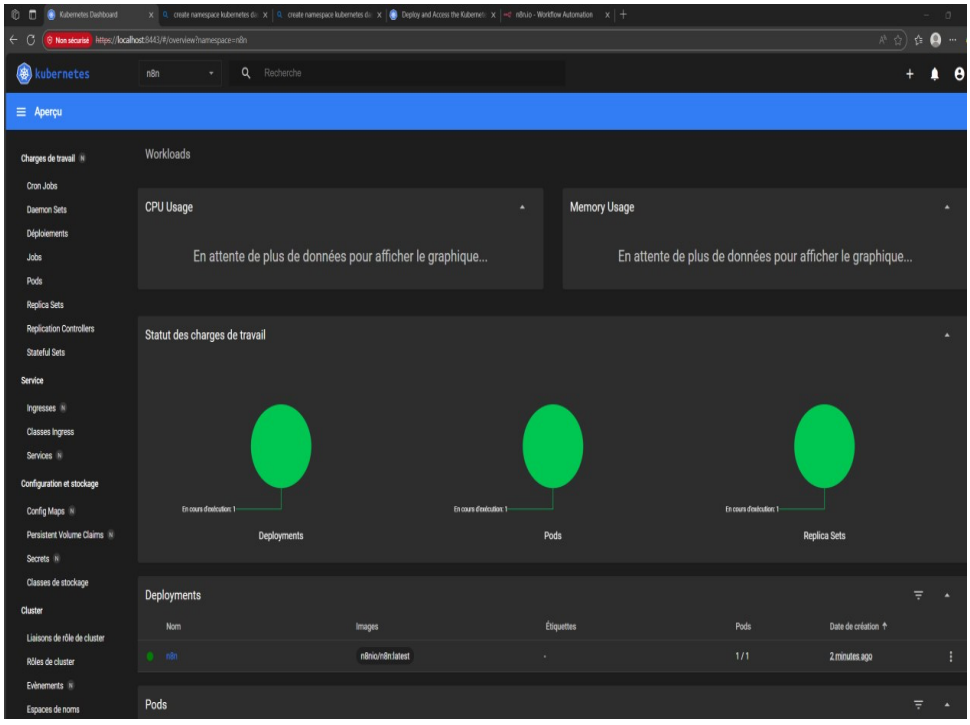




Ensuite j'ai créé le namespace N8N depuis le Dashboard en cliquant sur le +

J'ai cliqué à nouveau sur le + pour y déposer mon YML afin de déployer un node n8N

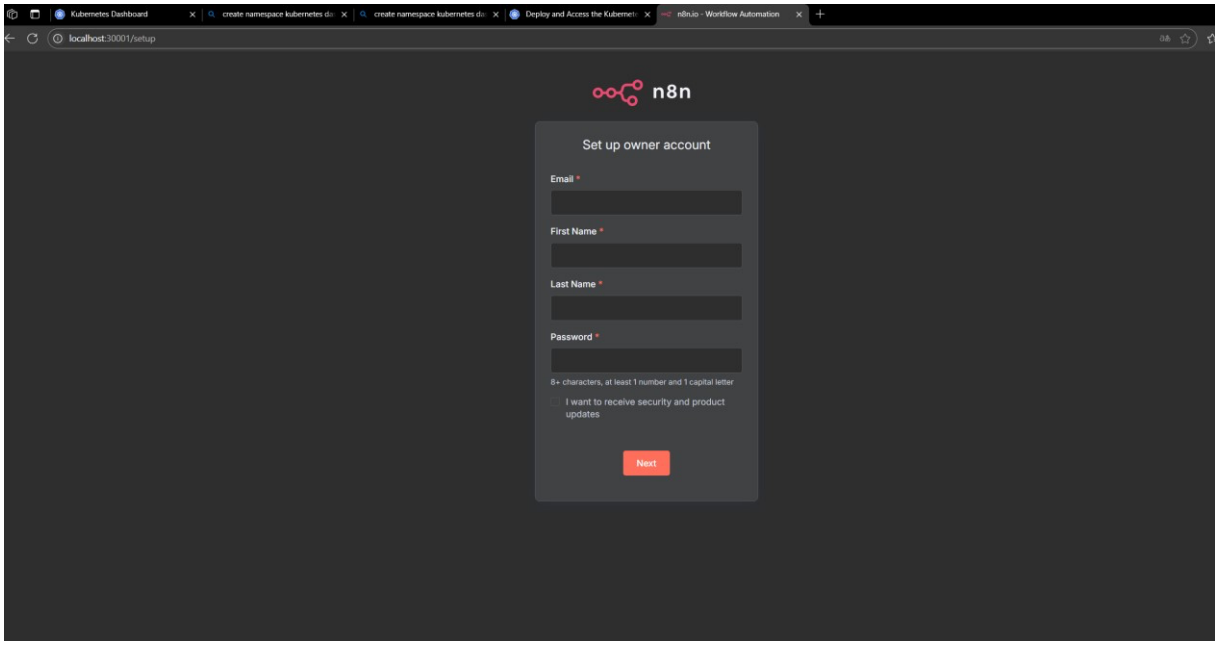
```yaml
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: n8n-pvc
  namespace: n8n
spec:
  accessModes:
    - ReadWriteOnce
  resources:
    requests:
      storage: 1Gi
  storageClassName: local-path
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: n8n
  namespace: n8n
spec:
  replicas: 1
  selector:
    matchLabels:
      app: n8n
  template:
    metadata:
      labels:
        app: n8n
    spec:
      containers:
      - name: n8n
        image: n8nio/n8n:latest
        ports:
        - containerPort: 5678
        volumeMounts:
        - name: n8n-storage
          mountPath: /home/node/.n8n
        env:
        - name: DB_TYPE
          value: "sqlite"
        - name: N8N_BASIC_AUTH_ACTIVE
          value: "true"
        - name: N8N_BASIC_AUTH_USER
          value: "admin"
        - name: N8N_BASIC_AUTH_PASSWORD
          value: "yourpassword"
      volumes:
      - name: n8n-storage
        persistentVolumeClaim:
          claimName: n8n-pvc
---
apiVersion: v1
kind: Service
metadata:
  name: n8n
  namespace: n8n
spec:
  type: NodePort
  ports:
  - port: 5678
    targetPort: 5678
    nodePort: 30001
  selector:
    app: n8n
```



Ci-dessus un aperçu du déploiement réussi

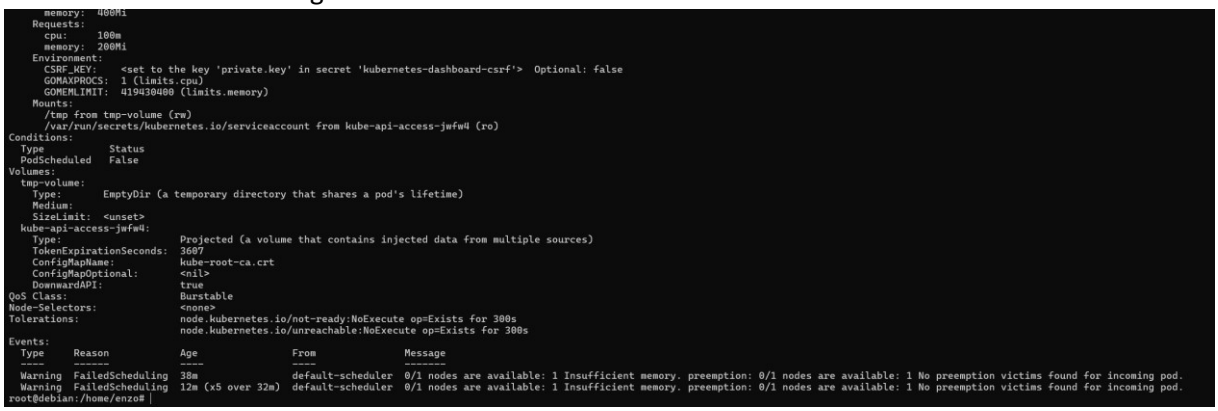
On peut voir que ça fonctionne après avoir redirigé le port 3001 de ma VM sur mon pc

```
PS C:\Users\Enzo> ssh -L 30001:localhost:30001 enzo@192.168.65.171
```



Problèmes Rencontrés

A la connexion sur le Dashboard j'avais une erreur qui étais du a un manque de ram comme j'ai pu le voir ci-dessous dans les logs



```
    memory:  400Mi
  Requests:
    cpu:     100m
    memory:  200Mi
  Environment:
    CSRF_KEY:     <set to the key 'private.key' in secret 'kubernetes-dashboard-csrf'>  Optional: false
    GOMAXPROCS:  1 (limits.cpu)
    GOMEMLIMIT:  4194304000 (limits.memory)
  Mounts:
    /tmp from tmp-volume (rw)
    /var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-jwfw4 (ro)
Conditions:
  Type           Status
  PodScheduled   False
Volumes:
  tmp-volume:
    Type:       EmptyDir (a temporary directory that shares a pod's lifetime)
    Medium:
    SizeLimit:  <unset>
  kube-api-access-jwfw4:
    Type:                     Projected (a volume that contains injected data from multiple sources)
    TokenExpirationSeconds:   3607
    ConfigMapName:            kube-root-ca.crt
    ConfigMapOptional:        <nil>
    DownwardAPI:              true
QoS Class:                    Burstable
Node-Selectors:               <none>
Tolerations:                  node.kubernetes.io/not-ready:NoExecute op=Exists for 300s
                              node.kubernetes.io/unreachable:NoExecute op=Exists for 300s
Events:
  Type     Reason            Age                From               Message
  ----     ------            ----               ----               -------
  Warning  FailedScheduling  38m                default-scheduler  0/1 nodes are available: 1 Insufficient memory. preemption: 0/1 nodes are available: 1 No preemption victims found for incoming pod.
  Warning  FailedScheduling  12m (x5 over 32m)  default-scheduler  0/1 nodes are available: 1 Insufficient memory. preemption: 0/1 nodes are available: 1 No preemption victims found for incoming pod.
root@debian:/home/enzo#
```