MicroK8s - Guida Rapida e Documentazione Comandi

📋 Indice

🚀 Installazione e Setup

Installazione Base

bash

```bash
# Installa snapd (se non presente)
sudo apt update
sudo apt install -y snapd

# Installa MicroK8s
sudo snap install microk8s --classic

# Verifica installazione
microk8s version
```

Configurazione Utente

bash

```bash
# Aggiungi utente al gruppo microk8s
sudo usermod -a -G microk8s $USER
sudo chown -f -R $USER ~/.kube

# Ricarica gruppi (logout/login oppure)
newgrp microk8s
```

Avvio Iniziale

bash

```
# Avvia MicroK8s
microk8s start

# Verifica stato
microk8s status --wait-ready

# Controlla cluster
microk8s kubectl get nodes
```

⚙️ Configurazione kubectl

Metodo 1: Alias (Raccomandato)

bash

```
# Crea alias permanente
echo 'alias kubectl="microk8s kubectl"' >> ~/.bashrc
source ~/.bashrc

# Test
kubectl get nodes
```

Metodo 2: Configurazione Separata

bash

```
# Crea directory
mkdir -p ~/.kube

# Genera configurazione
microk8s config > ~/.kube/config

# Imposta permessi
chmod 600 ~/.kube/config

# Installa kubectl (opzionale)
sudo snap install kubectl --classic
```

Verifica Configurazione

bash

```
# Controlla configurazione
kubectl config view

# Controlla context
kubectl config get-contexts

# Test cluster
kubectl get nodes
kubectl get pods --all-namespaces
```

🔧 Gestione Cluster

Comandi di Base

bash

```
# Stato cluster
microk8s status

# Avvia cluster
microk8s start

# Ferma cluster
microk8s stop

# Riavvia cluster
microk8s stop && microk8s start

# Verifica salute cluster
microk8s status --wait-ready
```

Informazioni Cluster

bash

```
# Versione MicroK8s
microk8s version

# Informazioni dettagliate cluster
microk8s kubectl cluster-info

# Configurazione cluster
microk8s config

# Nodi del cluster
kubectl get nodes -o wide
```

Reset Cluster

bash

```
# ⚠️ ATTENZIONE: Cancella tutto!
microk8s reset

# Riavvia dopo reset
microk8s start

# Riabilita addons necessari
microk8s enable dns hostpath-storage
```

👥 Gestione Nodi

Cluster Single-Node

bash

```
# Verifica nodo locale
kubectl get nodes

# Dettagli nodo
kubectl describe node $(hostname)

# Risorse nodo
kubectl top node
```

Cluster Multi-Node

Master Node

bash

```
# Genera token per join
microk8s add-node

# Output esempio:
# microk8s join 192.168.0.202:25000/TOKEN/HASH

# Lista nodi nel cluster
kubectl get nodes

# Rimuovi nodo dal cluster
microk8s remove-node <node-name>
```

Worker Node

bash

```
# Join al cluster (esegui comando dal master)
microk8s join 192.168.0.202:25000/TOKEN/HASH

# Verifica join
microk8s status

# Lascia il cluster
microk8s leave
```

Gestione Nomi Nodi

bash

```
# Cambia hostname sistema
sudo hostnamectl set-hostname k8s-master

# Aggiorna /etc/hosts
sudo nano /etc/hosts
# Aggiungi: 127.0.1.1 k8s-master

# Riavvia MicroK8s per applicare
microk8s stop && microk8s start

# Verifica nuovo nome
kubectl get nodes
```

🔄 Riavvio Cluster

Riavvio Soft (Raccomandato)

bash

```
# Riavvio standard
microk8s stop
sleep 5
microk8s start

# Verifica stato
microk8s status --wait-ready
kubectl get nodes
```

Riavvio con Verifica Completa

bash

```
# Stop con verifica
microk8s stop
microk8s status  # Deve mostrare "stopped"

# Start con attesa
microk8s start
microk8s status --wait-ready

# Verifica cluster
kubectl get nodes
kubectl get pods --all-namespaces
```

Riavvio Hard Sistema

bash

```
# Riavvio completo Raspberry Pi
sudo reboot

# Dopo riboot, verifica MicroK8s
microk8s status

# Avvia se necessario
microk8s start
```

Riavvio Servizi Sistema

bash

```bash
# Riavvia daemon MicroK8s
sudo systemctl restart snap.microk8s.daemon

# Riavvia servizi di rete
sudo systemctl restart systemd-networkd
sudo systemctl restart systemd-resolved
```

One-Liner Riavvio Completo

bash

```bash
# Comando unico per riavvio e verifica
microk8s stop && sleep 5 && microk8s start && microk8s status --wait-ready && kubectl
```

🔌 Addons

Addons Essenziali

bash

```bash
# DNS (necessario)
microk8s enable dns

# Storage locale
microk8s enable hostpath-storage

# Ingress controller
microk8s enable ingress

# Dashboard web
microk8s enable dashboard
```

Lista Addons

bash

```bash
# Addons disponibili
microk8s status

# Addons abilitati
microk8s status | grep enabled

# Addons disabilitati
microk8s status | grep disabled
```

Gestione Addons

bash

```
# Abilita addon
microk8s enable <addon-name>

# Disabilita addon
microk8s disable <addon-name>

# Abilita multipli
microk8s enable dns hostpath-storage ingress
```

Addons Avanzati

bash

```
# Prometheus (monitoraggio)
microk8s enable prometheus

# Grafana (dashboard metriche)
microk8s enable grafana

# Cert-manager (certificati SSL)
microk8s enable cert-manager

# MetalLB (load balancer)
microk8s enable metallb
```

🐛 Debugging e Troubleshooting

Verifica Stato

bash

```
# Stato generale
microk8s status

# Stato dettagliato
microk8s status --wait-ready

# Ispezione cluster
microk8s inspect
```

Log e Debug

bash

```
# Log MicroK8s
sudo journalctl -u snap.microk8s.daemon -f

# Log specifico pod
kubectl logs <pod-name> -n <namespace>

# Log con follow
kubectl logs -f <pod-name> -n <namespace>

# Descrizione risorse
kubectl describe pod <pod-name>
kubectl describe node <node-name>
```

Risoluzione Problemi

bash

```
# Nodi NotReady
kubectl get nodes
kubectl describe node <node-name>

# Pod in errore
kubectl get pods --all-namespaces
kubectl delete pod <pod-name> -n <namespace>

# Problemi di rete
microk8s disable dns
microk8s enable dns

# Reset configurazione
microk8s reset
microk8s start
```

Servizi e Networking

bash

```bash
# Esponi deployment (sintassi corretta)
kubectl expose deployment <name> --port=80 --target-port=80 --type=NodePort

# Esempi comuni:
kubectl expose deployment nginx --port=80 --target-port=80 --type=NodePort
kubectl expose deployment app --port=8080 --target-port=3000 --type=NodePort

# Verifica servizi
kubectl get services
kubectl describe service <service-name>

# Test connettività
curl http://localhost:<nodeport>
curl http://<node-ip>:<nodeport>
```

⚡ Comandi Essenziali

Gestione Cluster

Comando	Descrizione
`microk8s start`	Avvia MicroK8s
`microk8s stop`	Ferma MicroK8s
`microk8s status`	Stato cluster
`microk8s reset`	Reset completo
`microk8s version`	Versione MicroK8s

Gestione Nodi

Comando	Descrizione
`microk8s add-node`	Genera token join
`microk8s join <token>`	Join al cluster
`microk8s leave`	Lascia cluster
`microk8s remove-node <node>`	Rimuovi nodo

kubectl Essenziali

Comando	Descrizione
`kubectl get nodes`	Lista nodi
`kubectl get pods --all-namespaces`	Lista tutti i pod
`kubectl describe node <name>`	Dettagli nodo
`kubectl top node`	Uso risorse nodi

Addons

Comando	Descrizione
`microk8s status`	Lista addons
`microk8s enable <addon>`	Abilita addon
`microk8s disable <addon>`	Disabilita addon
`microk8s enable dns hostpath-storage`	Abilita multipli

Debug

Comando	Descrizione
`microk8s inspect`	Diagnostica cluster
`kubectl logs <pod>`	Log pod
`kubectl describe <resource>`	Dettagli risorsa
`microk8s status --wait-ready`	Attendi ready

🎯 Workflow Tipici

Setup Nuovo Cluster

bash

```
# 1. Installazione
sudo snap install microk8s --classic
sudo usermod -a -G microk8s $USER
newgrp microk8s

# 2. Avvio
microk8s start
microk8s status --wait-ready

# 3. Configurazione kubectl
echo 'alias kubectl="microk8s kubectl"' >> ~/.bashrc
source ~/.bashrc

# 4. Addons base
microk8s enable dns hostpath-storage ingress

# 5. Verifica
kubectl get nodes
kubectl get pods --all-namespaces
```

Aggiunta Worker Node

bash

```
# Su Master
microk8s add-node

# Su Worker
microk8s join <token-from-master>

# Verifica da Master
kubectl get nodes
```

Troubleshooting Standard

bash

```bash
# 1. Verifica stato
microk8s status
kubectl get nodes

# 2. Controlla pod
kubectl get pods --all-namespaces

# 3. Se problemi, riavvia
microk8s stop && microk8s start

# 4. Verifica dopo riavvio
microk8s status --wait-ready
kubectl get nodes
```

Deploy Applicazione Test

bash

```bash
# 1. Deploy nginx
kubectl create deployment nginx --image=nginx:alpine

# 2. Scala
kubectl scale deployment nginx --replicas=3

# 3. Esponi (specificando port e target-port)
kubectl expose deployment nginx --port=80 --target-port=80 --type=NodePort

# 4. Verifica
kubectl get pods,services
```

🚨 Comandi di Emergenza

Cluster Non Risponde

bash

```
# 1. Force stop
sudo systemctl stop snap.microk8s.daemon

# 2. Kill processi
sudo pkill -f microk8s

# 3. Riavvia
microk8s start

# 4. Se ancora problemi, reset
microk8s reset && microk8s start
```

Nodo Corrotto

bash

```
# Rimuovi nodo dal cluster
microk8s remove-node <node-name>

# Sul nodo corrotto
microk8s reset
microk8s start

# Riaggiungi al cluster
microk8s add-node  # su master
microk8s join <token>  # su worker
```

Ripristino Completo

bash

```
# ⚠️ ATTENZIONE: Cancella tutto!
microk8s reset
sudo snap remove microk8s
sudo snap install microk8s --classic
# Riconfigura tutto da capo
```

📝 Note Importanti

Sicurezza

- MicroK8s usa configurazioni sicure di default
- I pod girano in namespace isolati

- Firewall integrato per servizi esposti

Performance

- Ottimizzato per dispositivi edge
- Uso ridotto di risorse rispetto a Kubernetes standard
- Ideale per Raspberry Pi e IoT

Limiti

- Non supporta tutti i plugin di Kubernetes standard
- Alcuni addon potrebbero non essere disponibili
- Focus su semplicità vs flessibilità completa

Best Practices

- Usa sempre addons ufficiali
- Monitora uso risorse con `kubectl top`
- Backup configurazioni importanti
- Testa sempre in ambiente di sviluppo

Questa documentazione è specifica per MicroK8s su Raspberry Pi con Ubuntu Server. Aggiorna regolarmente con `sudo snap refresh microk8s`.