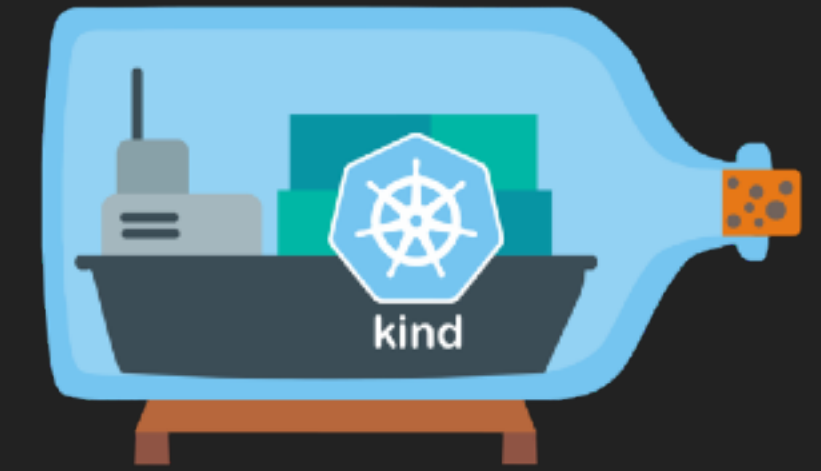


BUT HOW (TO GET STARTED)

- ▶ Make your first steps with kind or k3d on your local machine
- ▶ K3s on a single node is good enough for your home-lab and a multi-node k3s is probably enough for 99% of your (personal) use-cases!



```
cedi@mae:~  
cedi@mae ~  
cedi@mae ~$ kind create cluster --name test  
Creating cluster "test" ...  
✓ Ensuring node image (kindest/node:v1.25.2)   
✓ Preparing nodes   
✓ Writing configuration   
✓ Starting control-plane   
✓ Installing CNI   
✓ Installing StorageClass   
Set kubectrl context to "kind-test"  
You can now use your cluster with:  
  
kubectrl cluster-info --context kind-test  
  
Thanks for using kind! 😊  
cedi@mae ~$ kubectrl cluster-info --context kind-test  
Kubernetes cluster info: https://kubernetes.io/docs/tasks/tools/kubectl-cluster-info/#  
cedi@mae ~$
```

```
cedi@mae:~  
cedi@mae ~$ k3d cluster create test  
INFO[0000] Prep: Network  
INFO[0000] Created network 'k3d-test'  
INFO[0000] Created image volume k3d-test-images  
INFO[0000] Starting new tools node...  
INFO[0000] Starting Node 'k3d-test-tools'  
INFO[0001] Creating node 'k3d-test-server-0'  
INFO[0001] Creating LoadBalancer 'k3d-test-serverlb'  
INFO[0001] Using the k3d-tools node to gather environment information  
INFO[0001] Starting new tools node...  
INFO[0001] Starting Node 'k3d-test-tools'  
INFO[0002] Starting cluster 'test'  
INFO[0002] Starting servers...  
INFO[0002] Starting Node 'k3d-test-server-0'  
INFO[0006] All agents already running.  
INFO[0006] Starting helpers...  
INFO[0006] Starting Node 'k3d-test-serverlb'  
INFO[0012] Injecting records for hostAliases (incl. host.k3d.internal) and for 3 network members into CoreDNS configmap...  
  
INFO[0014] Cluster 'test' created successfully!  
INFO[0014] You can now use it like this:  
kubectrl cluster-info  
  
cedi@mae ~$ export KUBECONFIG=$(k3d kubeconfig write test)  
cedi@mae ~$
```

BUT HOW (TO GET STARTED)

IF YOU WANT TO RUN YOUR HOMELAB: K3S



- ▶ There is an amazing K3s Ansible Role
- ▶ For long term K3s usage: Use a systemd service unit to keep k3s running

```
..de/ansible/k3s-ansible +
cedi@mae ~/src/cedi/av0de/ansible/k3s-ansible [master] ?1 ~1 zsh
bat roles/k3s/master/templates/k3s.service

File: roles/k3s/master/templates/k3s.service
1 [Unit]
2 Description=Lightweight Kubernetes
3 Documentation=https://k3s.io
4 After=network-online.target
5
6 [Service]
7 Type=notify
8 ExecStartPre=/sbin/modprobe br_netfilter
9 ExecStartPre=/sbin/modprobe overlay
10 ExecStart=/usr/local/bin/k3s server --data-dir /var/lib/rancher/k3s --flannel-backend=none --disable-network-policy --disable-traefik
11 KillMode=process
12 Delegate=yes
13 # Having non-zero Limit*s causes performance problems due to accounting overhead
14 # in the kernel. We recommend using cgroups to do container-local accounting.
15 LimitNOFILE=1048576
16 LimitNPROC=infinity
17 LimitCORE=infinity
18 TasksMax=infinity
19 TimeoutStartSec=0
20 Restart=always
21 RestartSec=5s
22
23 [Install]
24 WantedBy=multi-user.target
```