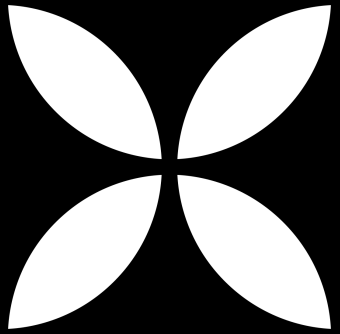


Karpenter

le ~~futur~~ présent de la gestion des noeuds Kubernetes

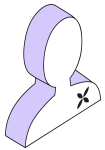
Mathieu Corbin

Senior Staff SRE @Qonto

https://www.mcorbin.fr/

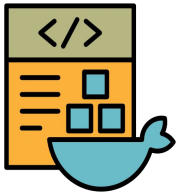
@_mcorbin

500k clients, 8 pays



1600+ Qontoers

> 150 déploiements en prod par jour

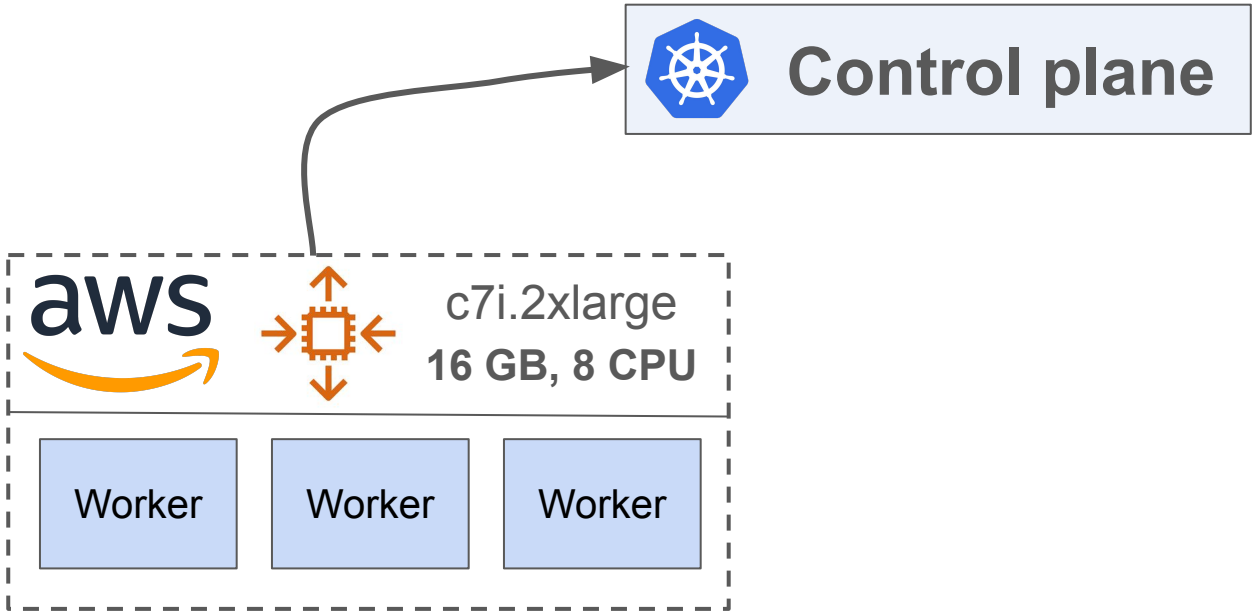


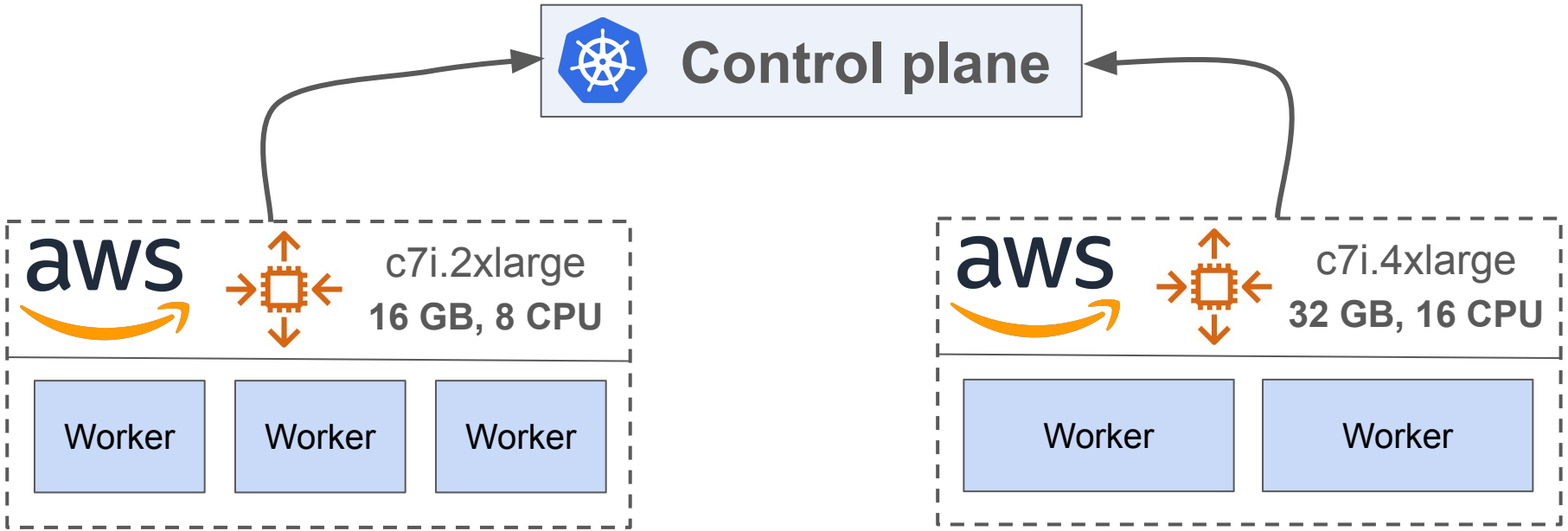
15 clusters Kubernetes

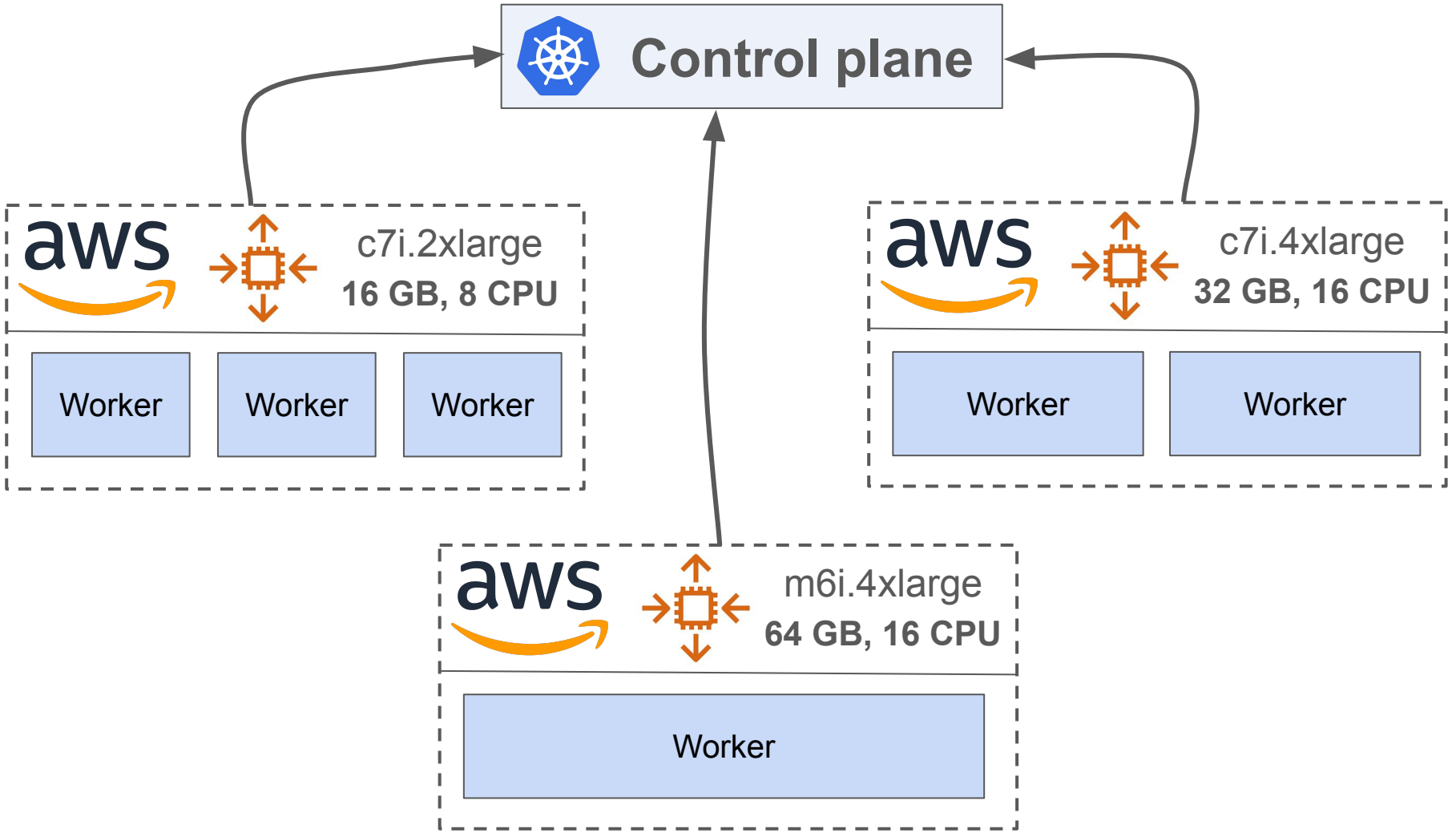
Gérer les noeuds d’un cluster Kubernetes est pénible !

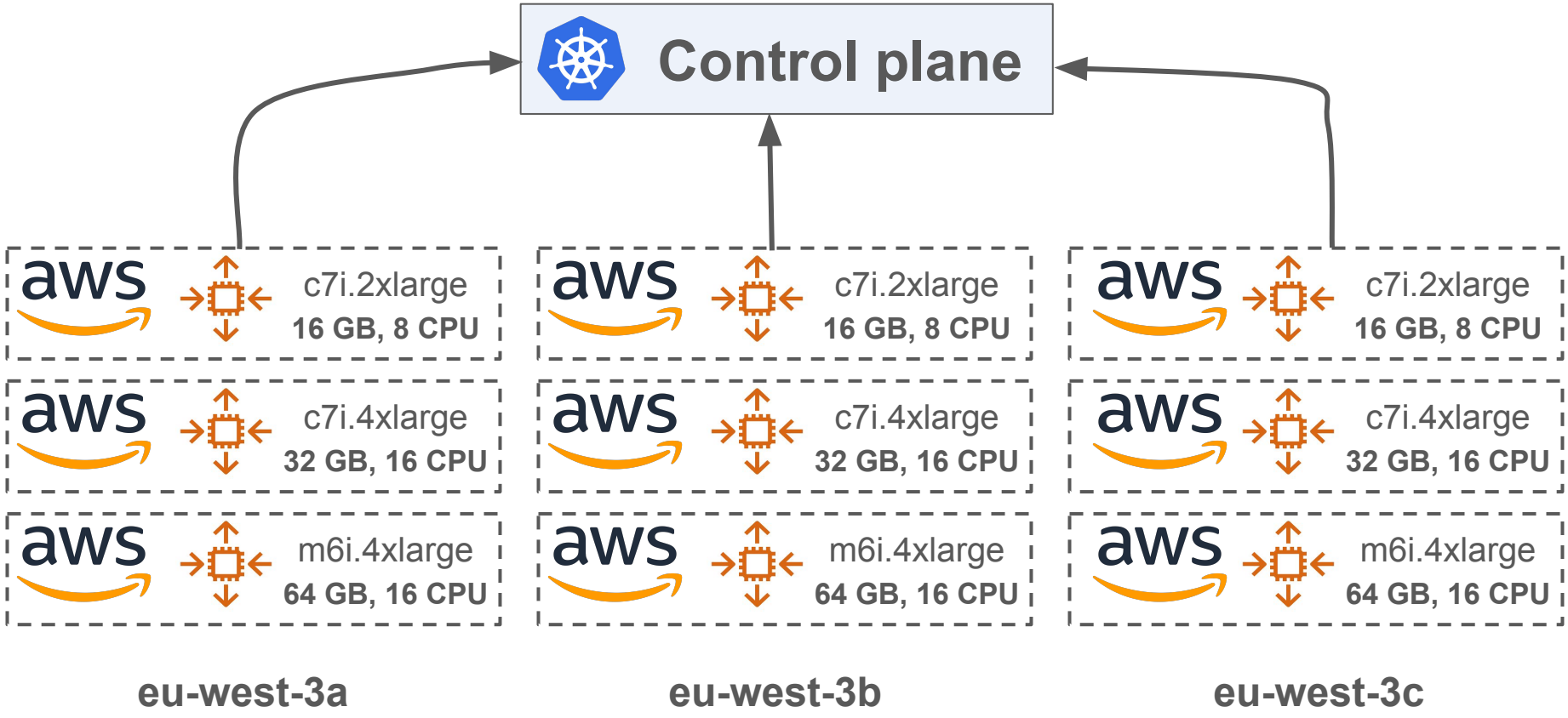
(même sur le cloud)

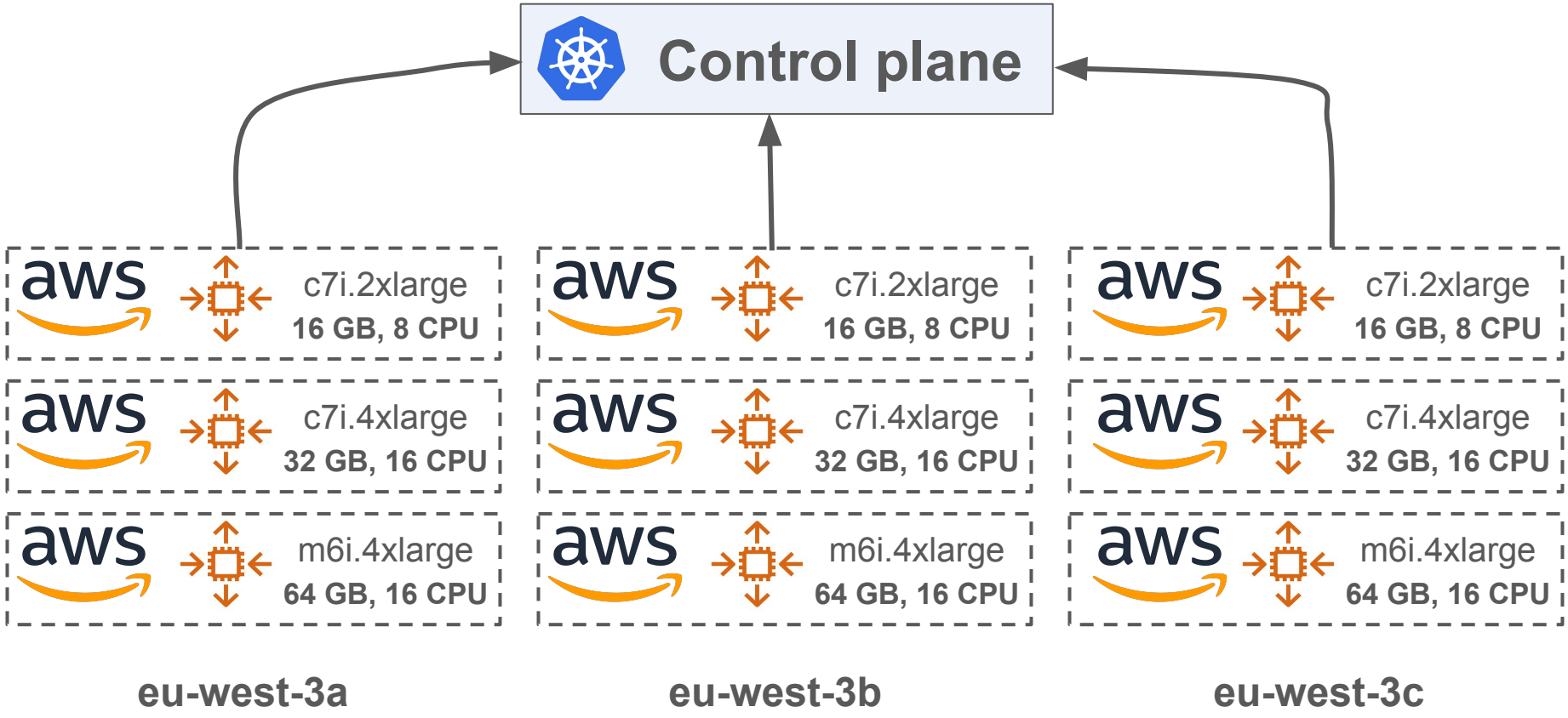

Control plane


Control plane
aws
c7i.2xlarge
16 GB, 8 CPU
Worker
Worker
Worker


Control plane
aws
c7i.2xlarge
16 GB, 8 CPU
Worker
Worker
Worker
aws
c7i.4xlarge
32 GB, 16 CPU
Worker
Worker

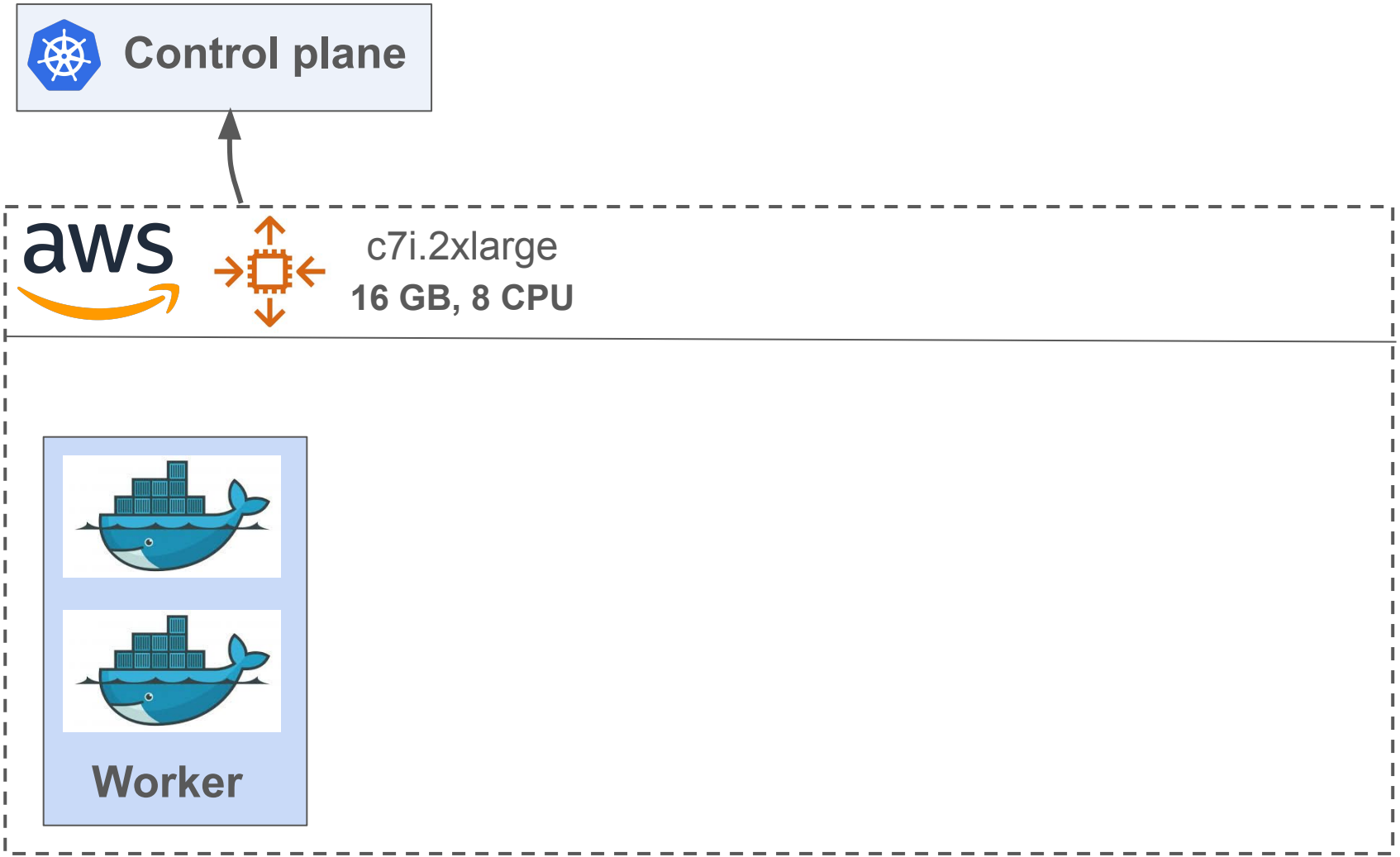

Control plane
aws
c7i.2xlarge
16 GB, 8 CPU
Worker
Worker
Worker
aws
c7i.4xlarge
32 GB, 16 CPU
Worker
Worker
aws
m6i.4xlarge
64 GB, 16 CPU
Worker

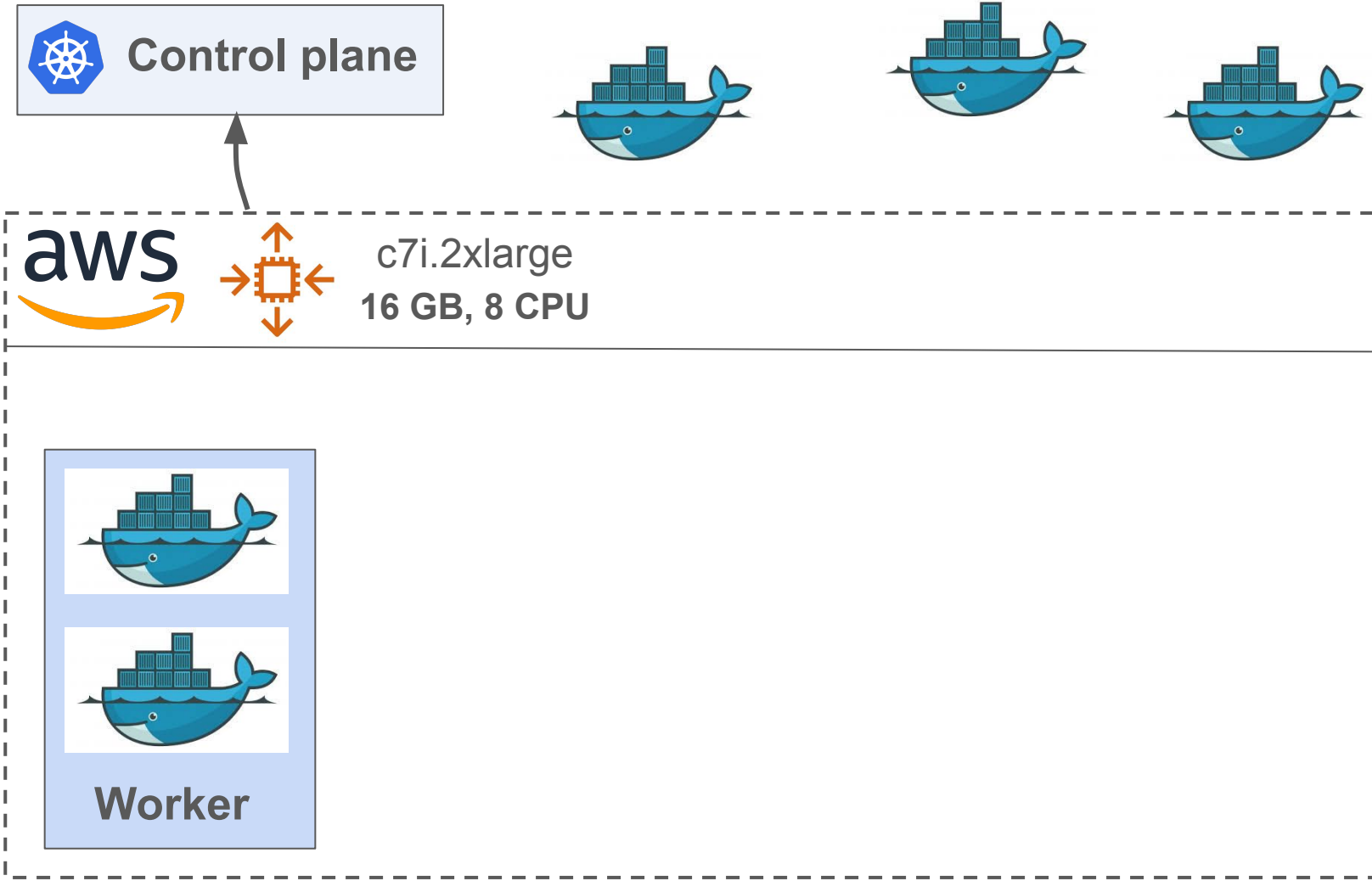

Control plane
c7i.2xlarge
16 GB, 8 CPU
c7i.4xlarge
32 GB, 16 CPU
m6i.4xlarge
64 GB, 16 CPU
c7i.2xlarge
16 GB, 8 CPU
c7i.4xlarge
32 GB, 16 CPU
m6i.4xlarge
64 GB, 16 CPU
c7i.2xlarge
16 GB, 8 CPU
c7i.4xlarge
32 GB, 16 CPU
m6i.4xlarge
64 GB, 16 CPU
eu-west-3a
eu-west-3b
eu-west-3c

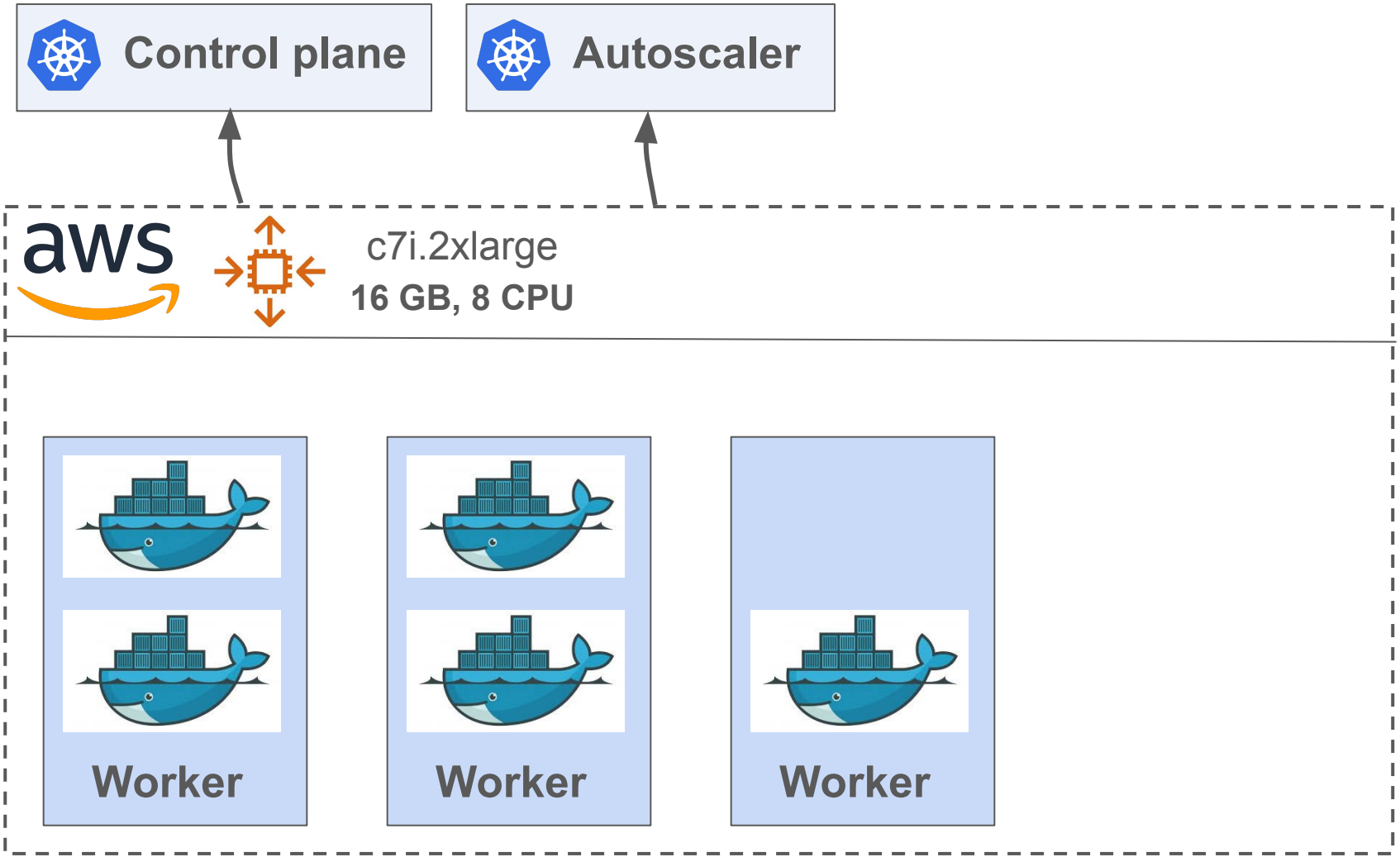


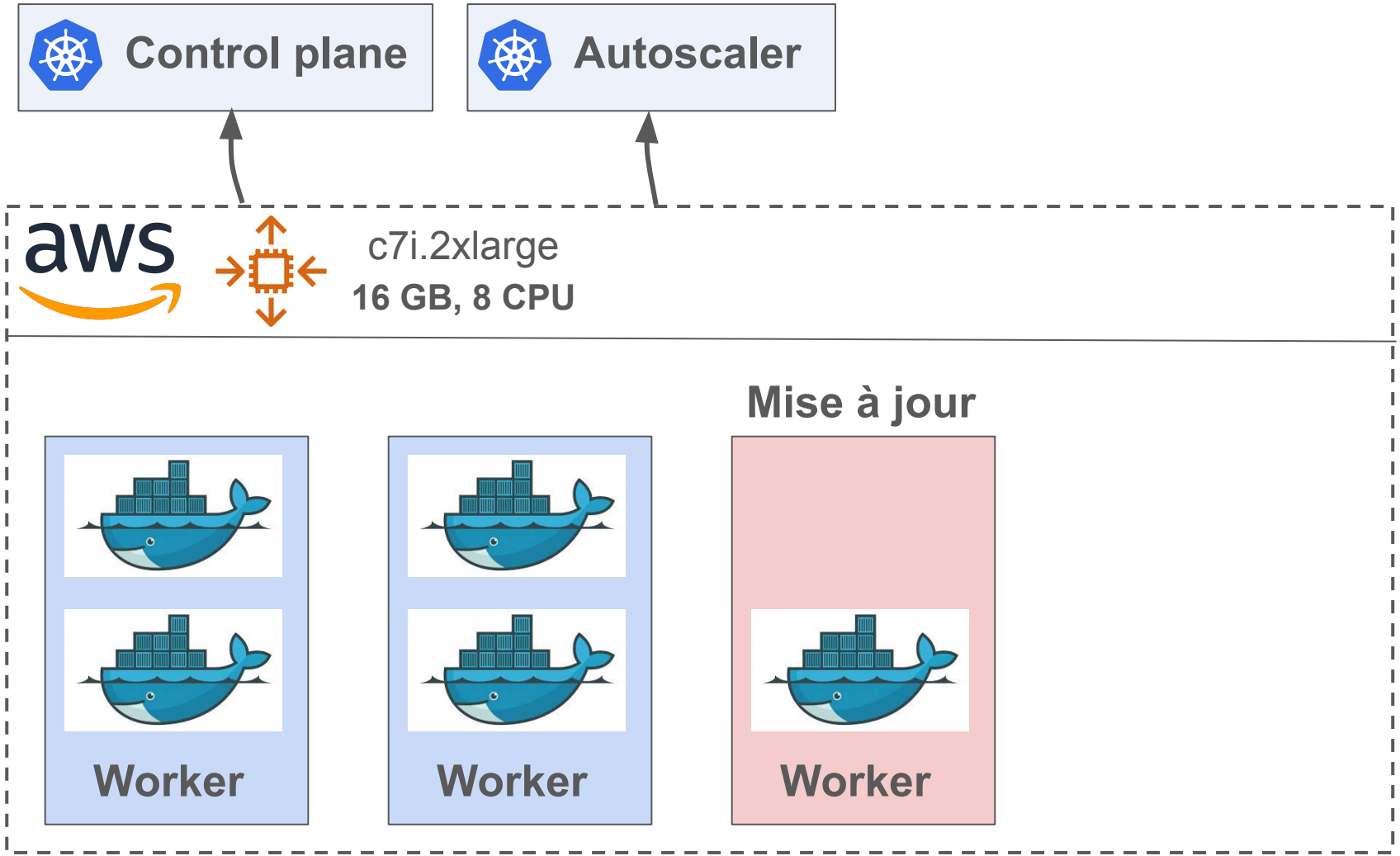
Instances “Spot” ? Instances ARM ?

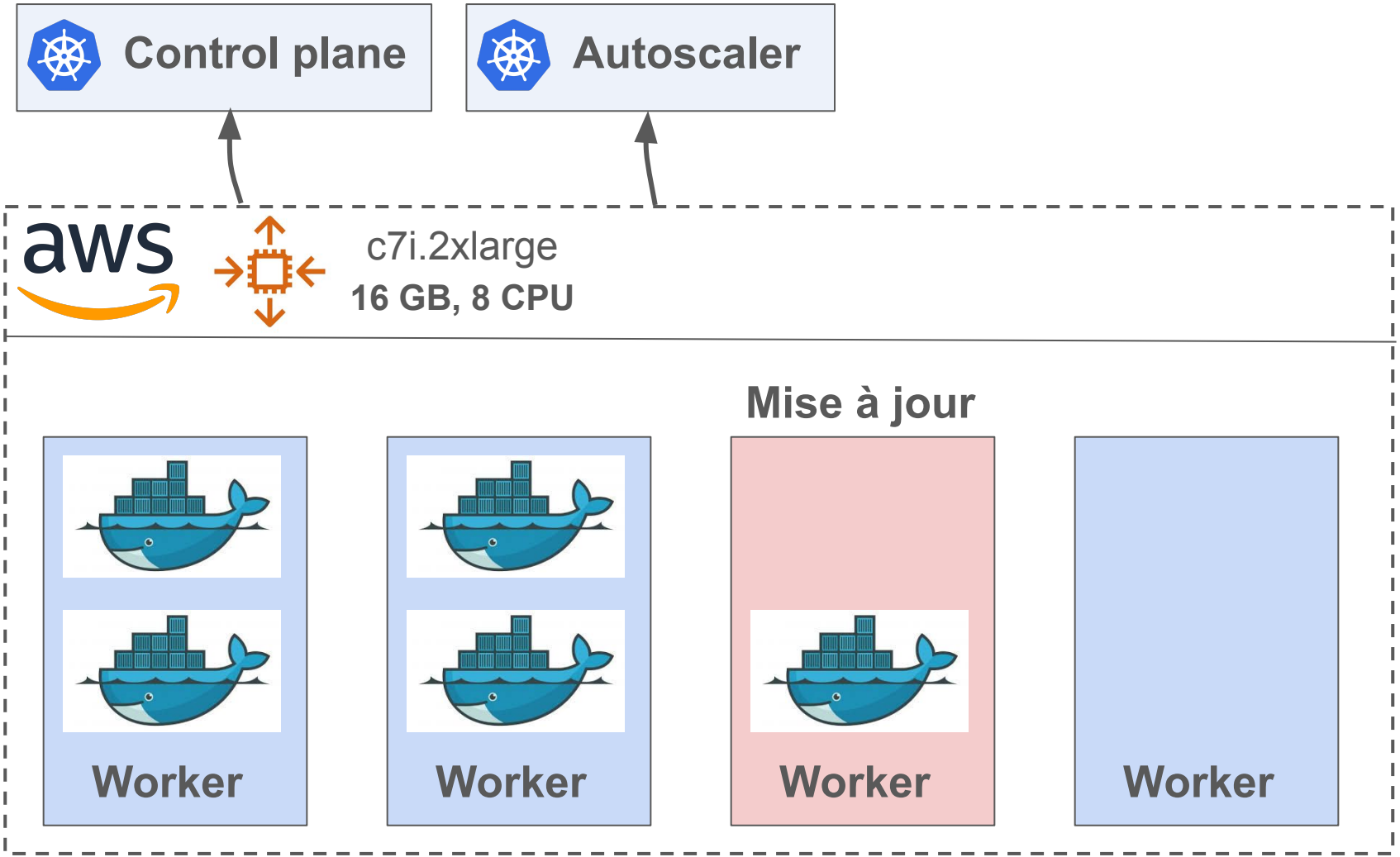
L’équipe SRE et le fichier cluster.tf

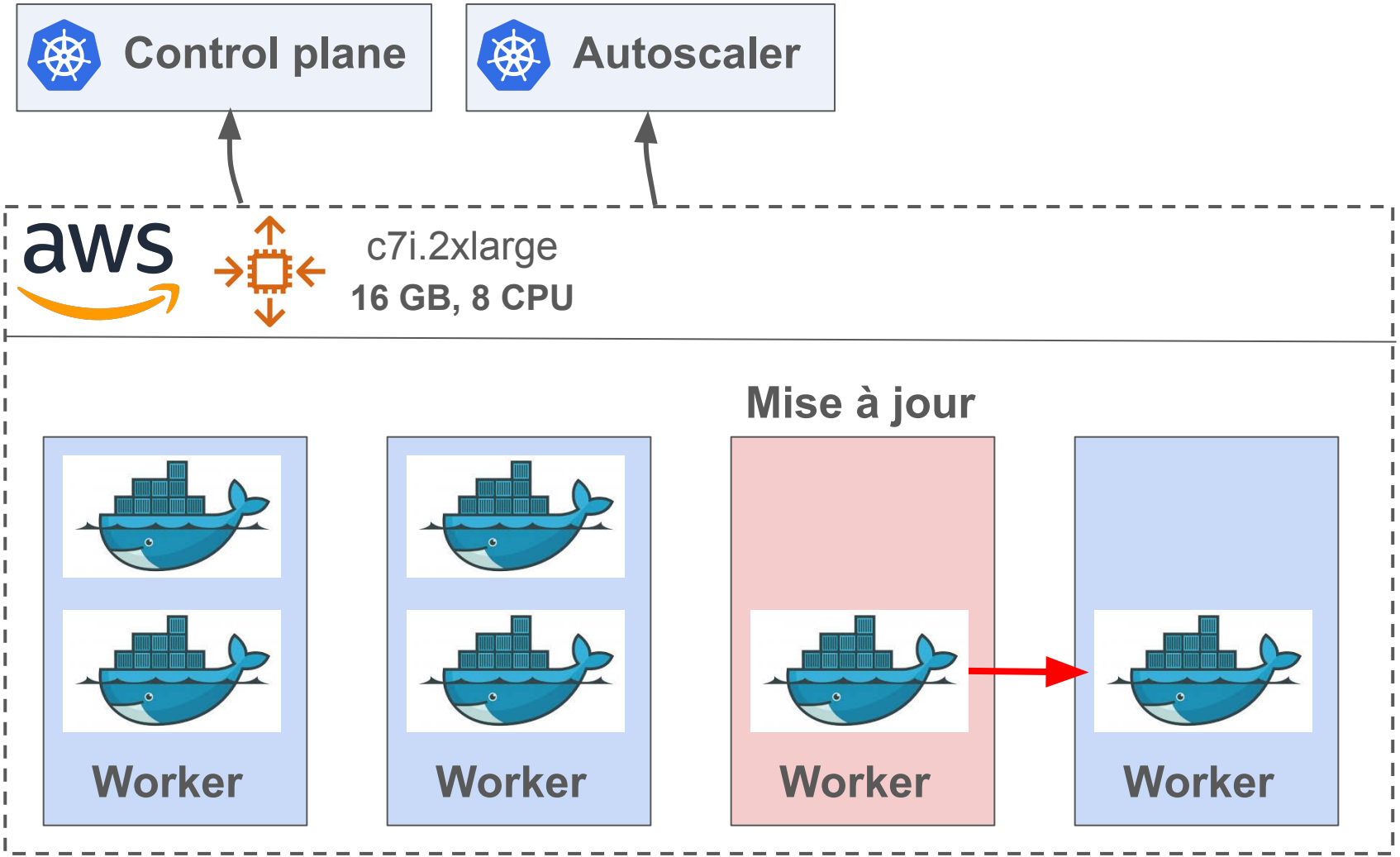

Control plane
aws
c7i.2xlarge
16 GB, 8 CPU
Worker

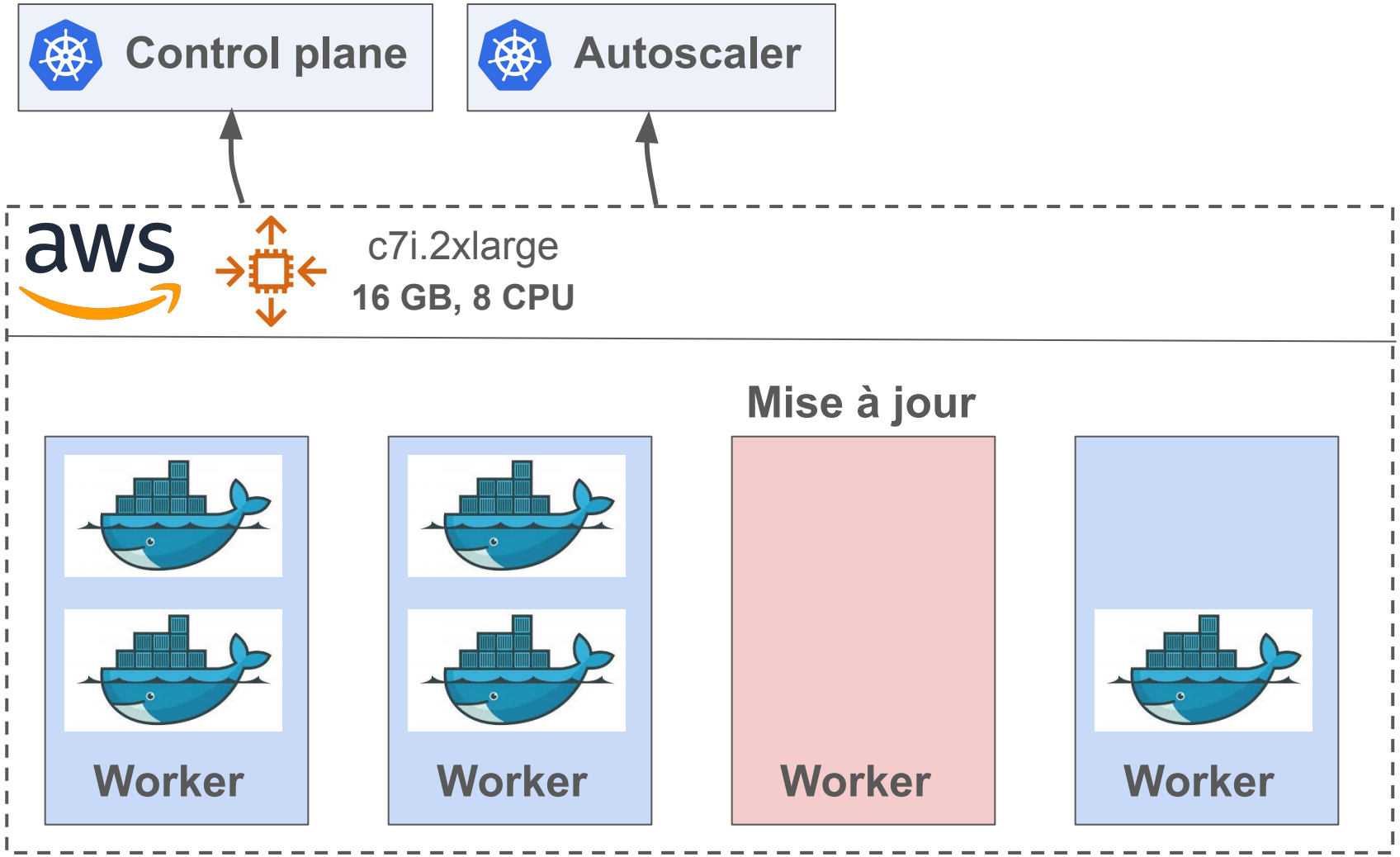

Control plane
aws
c7i.2xlarge
16 GB, 8 CPU
Worker

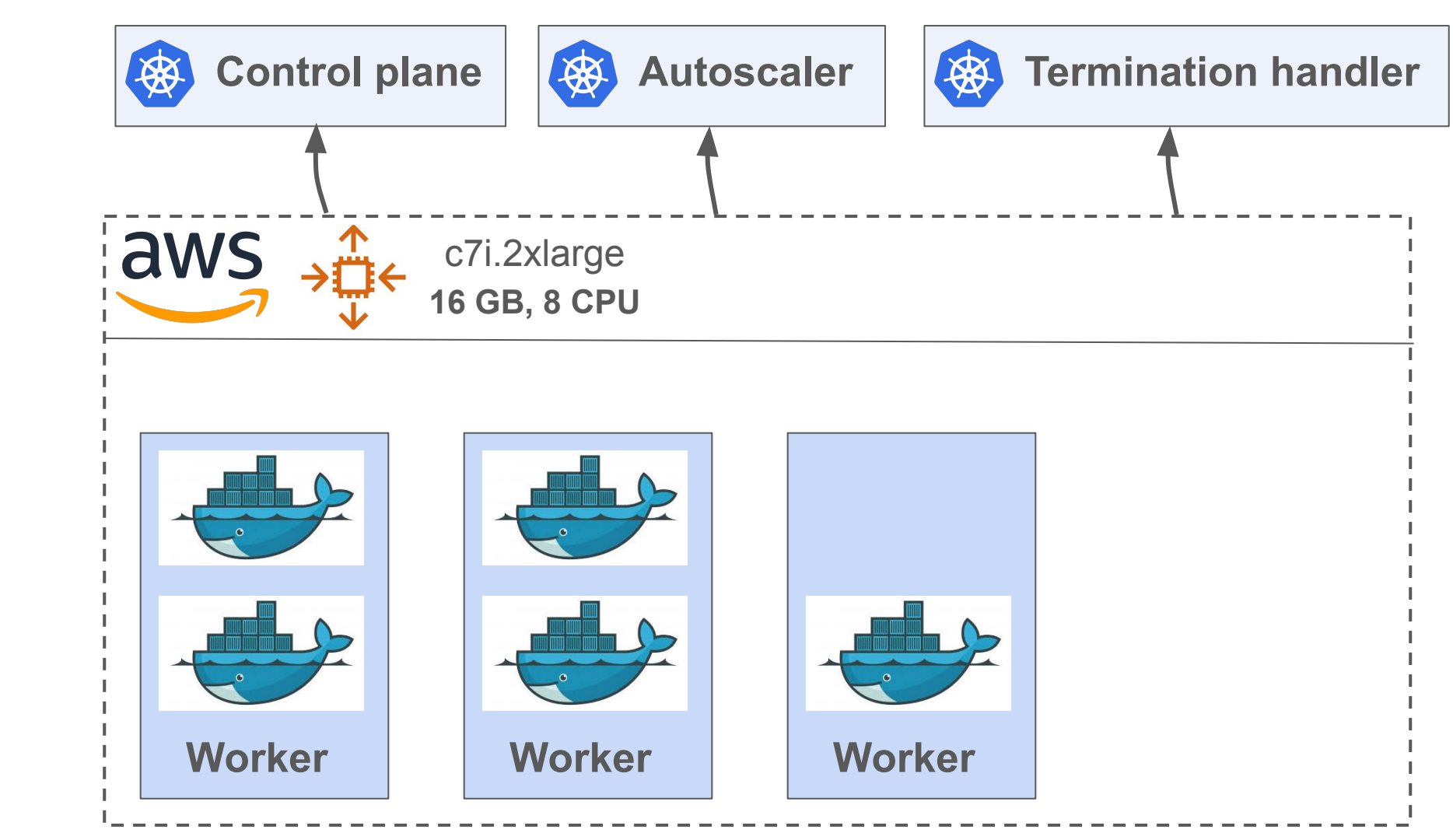

Control plane
Autoscaler
aws
c7i.2xlarge
16 GB, 8 CPU
Worker
Worker
Worker

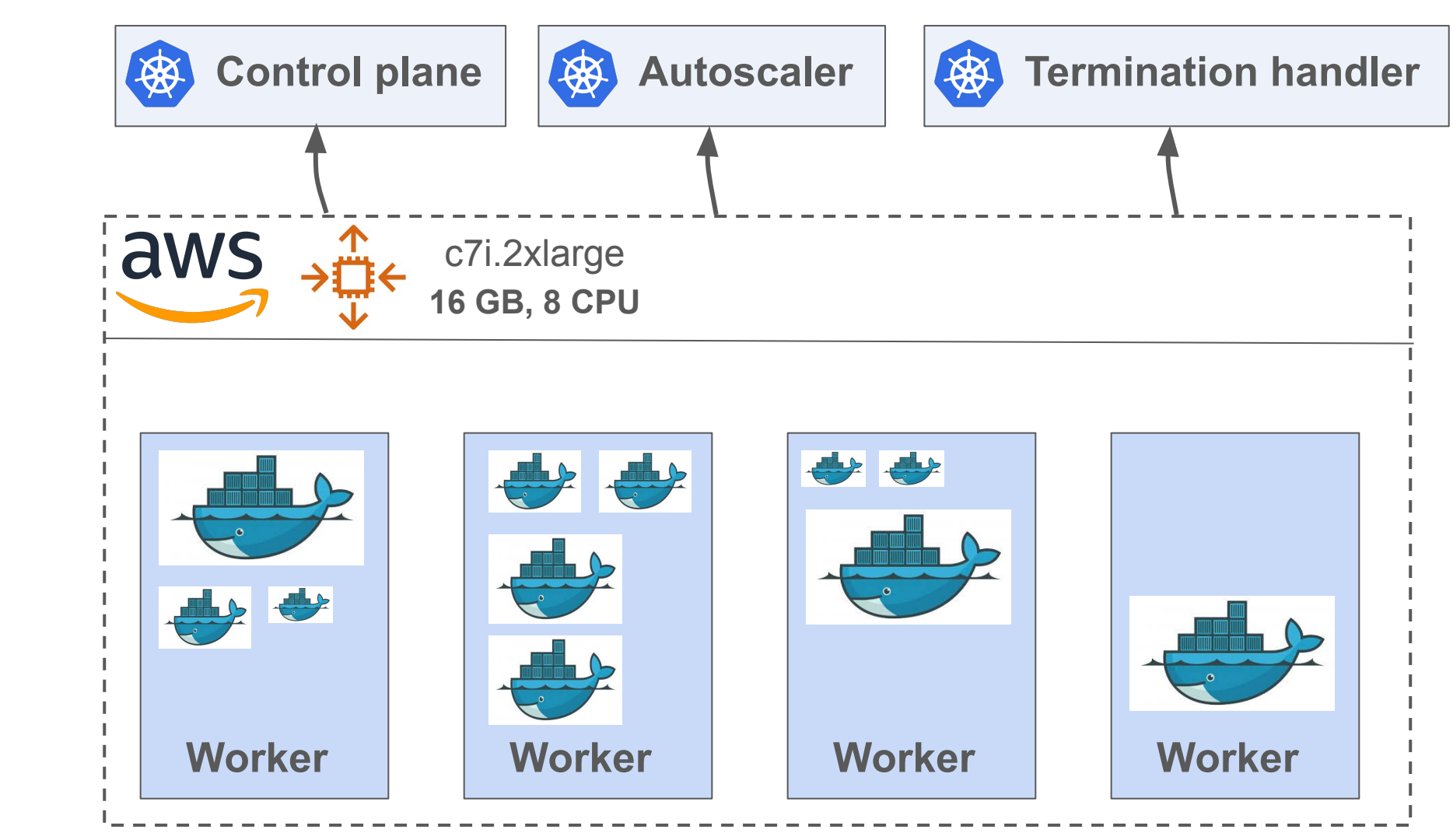

Control plane
Autoscaler
aws
c7i.2xlarge
16 GB, 8 CPU
Mise à jour
Worker
Worker
Worker

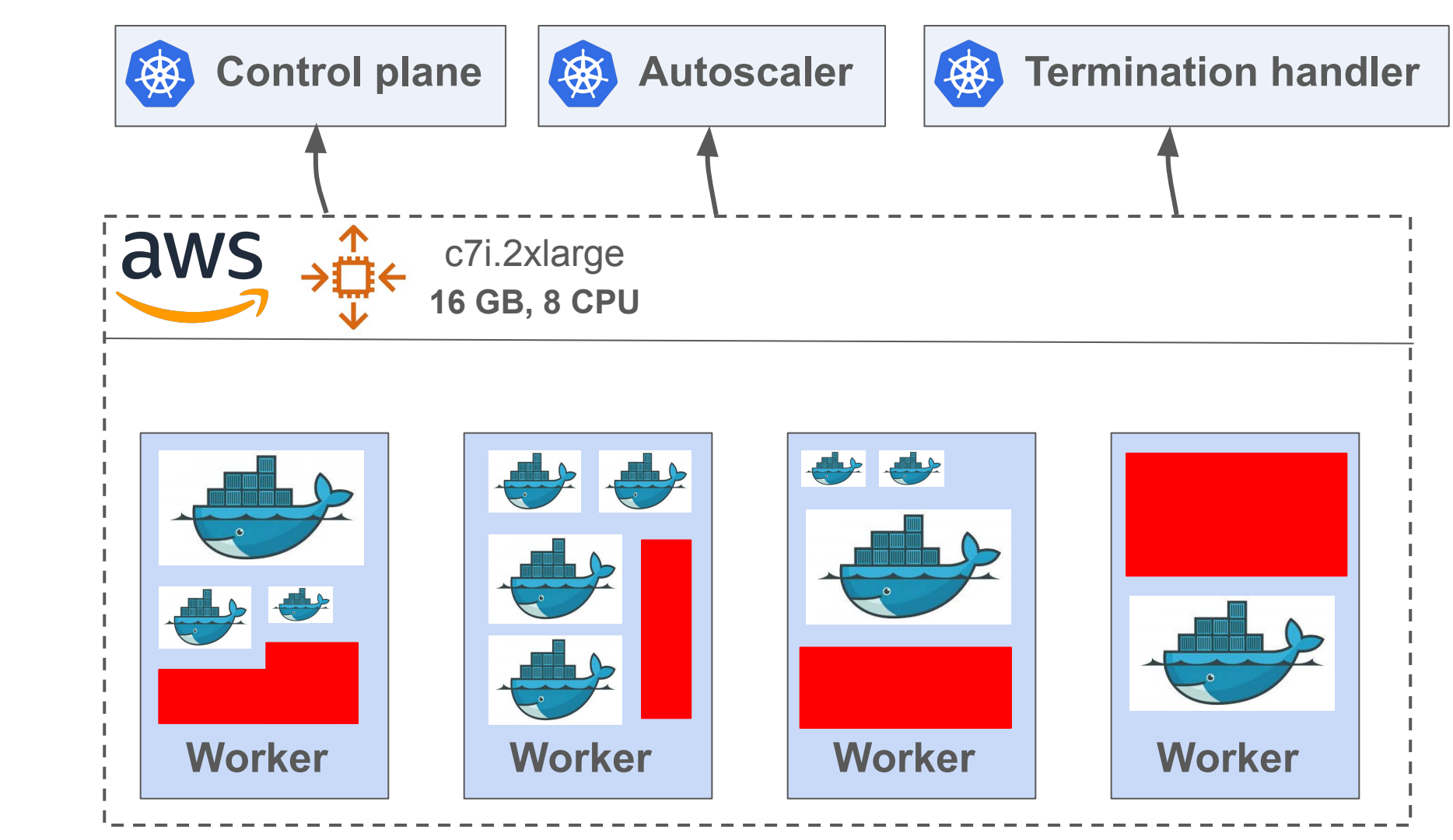

Control plane
Autoscaler
aws
c7i.2xlarge
16 GB, 8 CPU
Mise à jour
Worker
Worker
Worker
Worker


Control plane
Autoscaler
aws
c7i.2xlarge
16 GB, 8 CPU
Mise à jour
Worker
Worker
Worker
Worker


Control plane
Autoscaler
aws
c7i.2xlarge
16 GB, 8 CPU
Mise à jour
Worker
Worker
Worker
Worker


Control plane
Autoscaler
Termination handler
aws
c7i.2xlarge
16 GB, 8 CPU
Worker
Worker
Worker


Control plane
Autoscaler
Termination handler
aws
c7i.2xlarge
16 GB, 8 CPU
Worker
Worker
Worker
Worker


Control plane
Autoscaler
Termination handler
aws
c7i.2xlarge
16 GB, 8 CPU
Worker
Worker
Worker
Worker

Mettre à jour des dizaines d’autoscaling group est lent et pénible


Karpenter

NodeClass

une CRD spécifique à votre cloud

EC2NodeClass

Configuration des noeuds chez AWS

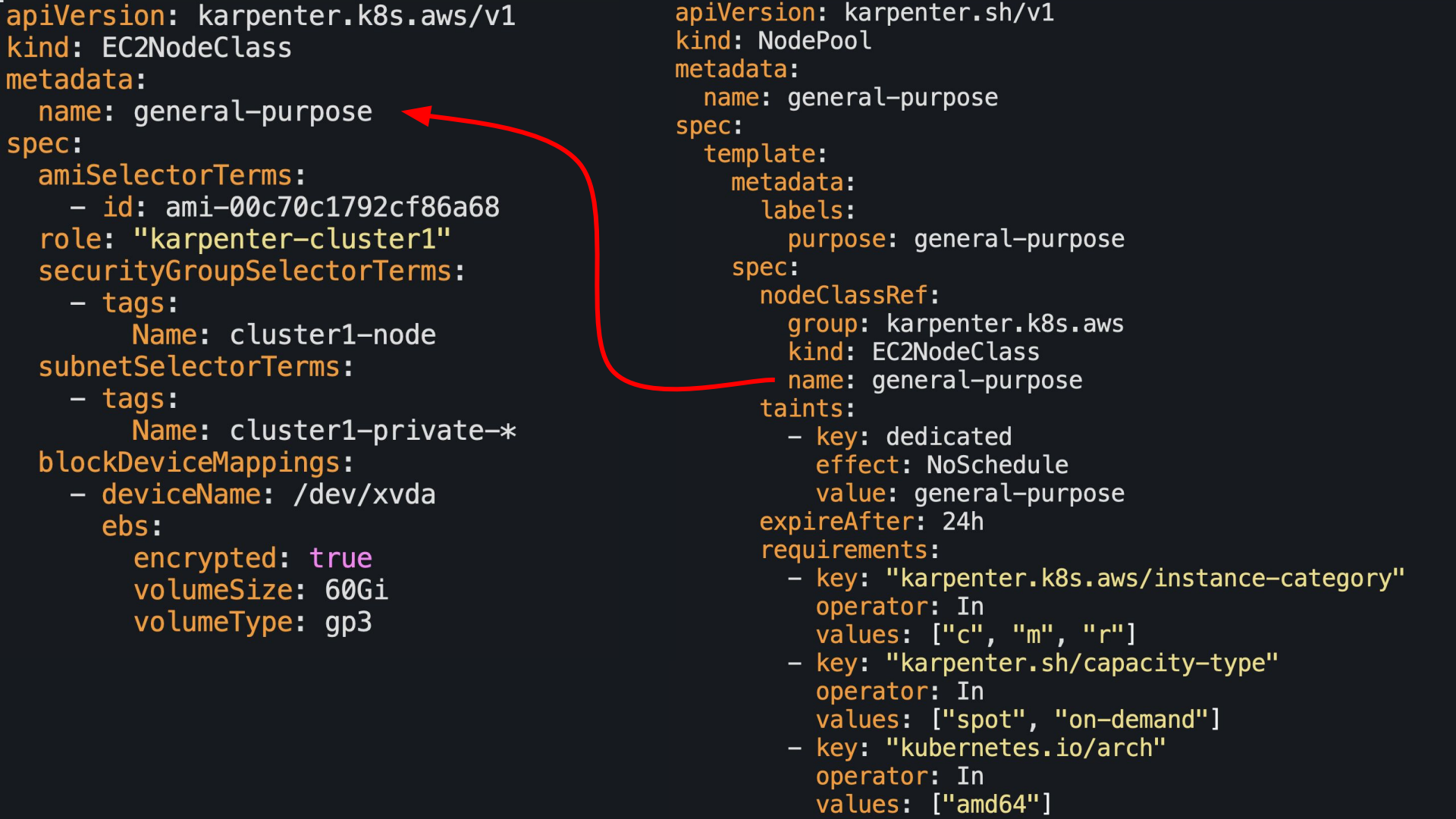
- Choix de l'AMI (image de base)
  - Version fixe ou sélection automatique
- Configuration réseau
  - VPC, security group…
- Permissions (IAM)
- Userdata (cloud-init)
- Configuration de Kubelet
- …

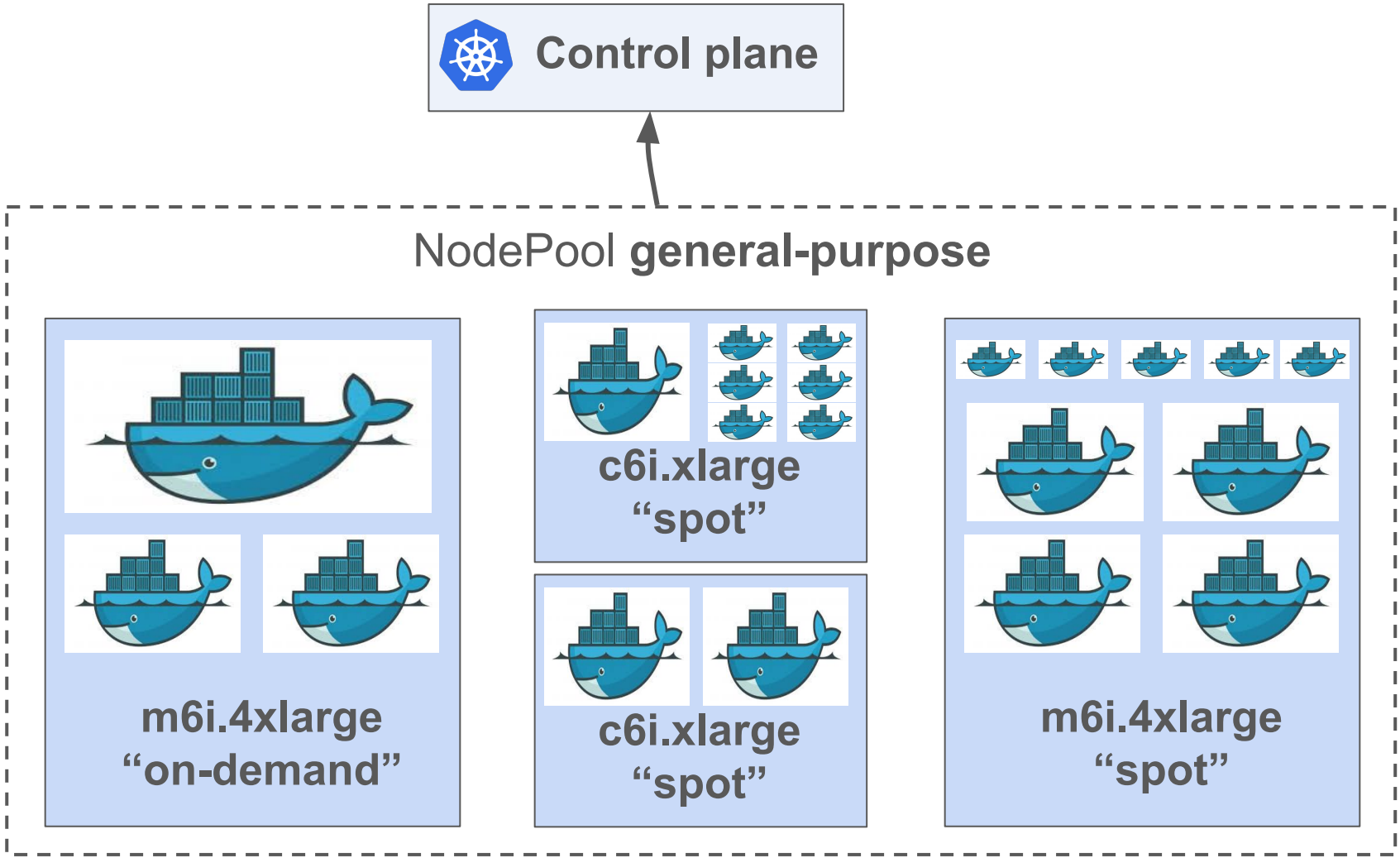
```yaml
apiVersion: karpenter.k8s.aws/v1
kind: EC2NodeClass
metadata:
  name: general-purpose
spec:
  amiSelectorTerms:
    - id: ami-00c70c1792cf86a68
  role: "karpenter-cluster1"
  securityGroupSelectorTerms:
    - tags:
        Name: cluster1-node
  subnetSelectorTerms:
    - tags:
        Name: cluster1-private-*
  blockDeviceMappings:
    - deviceName: /dev/xvda
      ebs:
        encrypted: true
        volumeSize: 60Gi
        volumeType: gp3
```

NodePool

- Labels, annotations, taints

- Requirements (optionnels)
  - Architecture
  - Taille min et max de l’instance
  - Génération d’instance
  - Capacité réseau
  - Type d’instance (spot ou on demand)
  - …


apiVersion: karpenter.k8s.aws/v1
kind: EC2NodeClass
metadata:
name: general-purpose
spec:
amiSelectorTerms:
- id: ami-00c70c1792cf86a68
role: "karpenter-cluster1"
securityGroupSelectorTerms:
- tags:
Name: cluster1-node
subnetSelectorTerms:
- tags:
Name: cluster1-private-*
blockDeviceMappings:
- deviceName: /dev/xvda
ebs:
encrypted: true
volumeSize: 60Gi
volumeType: gp3
apiVersion: karpenter.sh/v1
kind: NodePool
metadata:
name: general-purpose
spec:
template:
metadata:
labels:
purpose: general-purpose
spec:
nodeClassRef:
group: karpenter.k8s.aws
kind: EC2NodeClass
name: general-purpose
taints:
- key: dedicated
effect: NoSchedule
value: general-purpose
expireAfter: 24h
requirements:
- key: "karpenter.k8s.aws/instance-category"
operator: In
values: ["c", "m", "r"]
- key: "karpenter.sh/capacity-type"
operator: In
values: ["spot", "on-demand"]
- key: "kubernetes.io/arch"
operator: In
values: ["amd64"]


Control plane
NodePool general-purpose
m6i.4xlarge
“on-demand”
c6i.xlarge
“spot”
c6i.xlarge
“spot”
m6i.4xlarge
“spot”

Gestion des noeuds

Karpenter draine proprement les noeuds en cas de suppression

Consolidation

Karpenter optimise votre facture cloud tout seul !

Choix et **optimisation continue** des noeuds en fonction des workloads

(en tenant compte des contraintes de scheduling)

Consolidation + “expireAfter”

Des dizaines de noeuds créés et supprimés par jour !

Une très bonne façon de commencer le chaos engineering

Détection de drift

Mise à jour automatique des noeuds en cas de changement de configuration

Plages de maintenance

Pour vos workloads pénibles qui doivent se mettre à jour à des moments spécifiques !

```yaml
disruption:
  consolidationPolicy: WhenEmptyOrUnderutilized
  budgets:
    - nodes: "10%"
    - nodes: "0"
      schedule: "0 23 * * *"
      duration: 20h
```

Karpenter: un nouveau standard?

- Supérieur en tout point au cluster autoscaler
- Déjà disponible chez AWS et Azure
- Utilisable on-premise

Karpenter sera bientôt partout et sera un composant indispensable pour Kubernetes

Merci