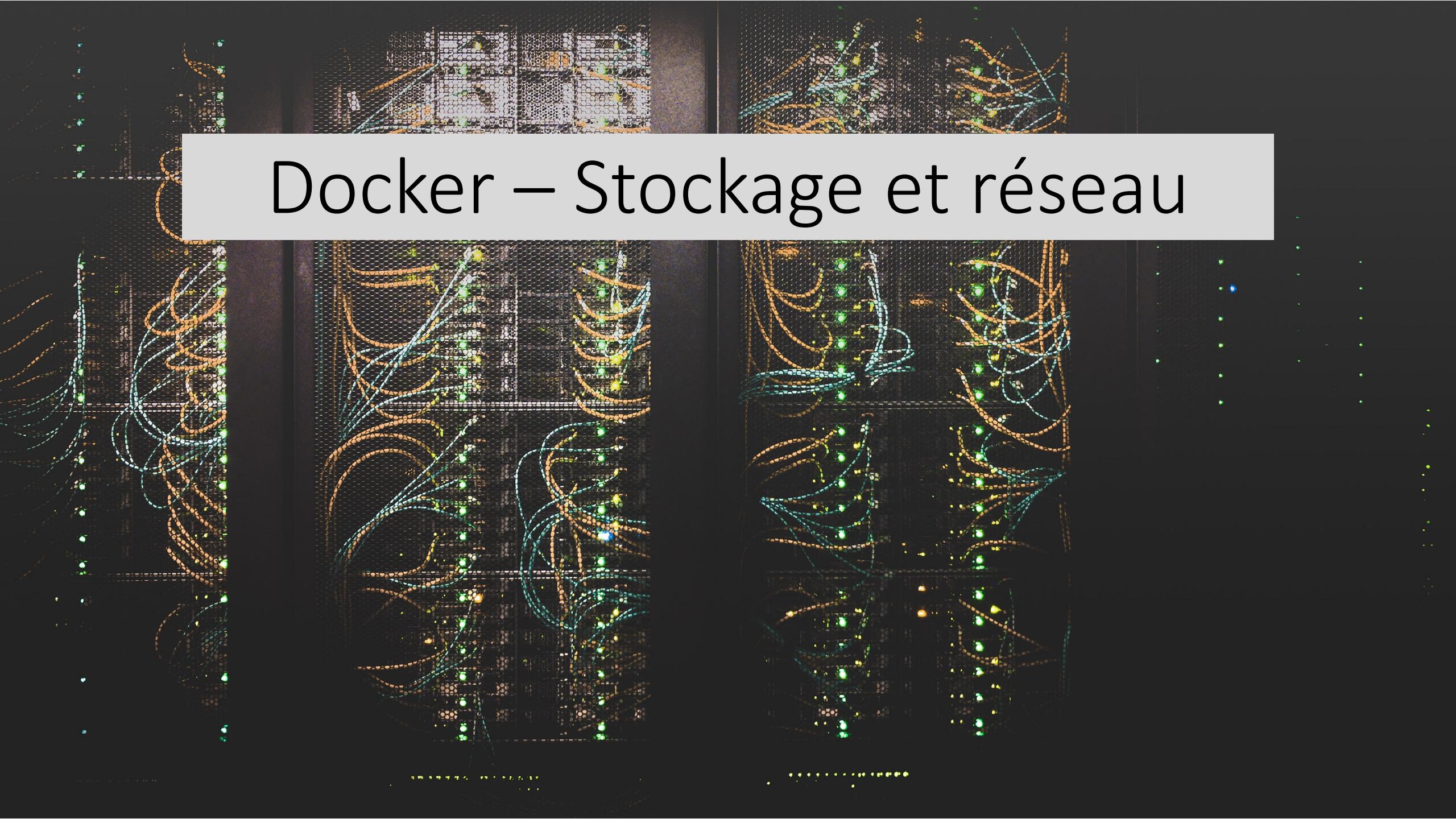


Docker – Stockage et réseau



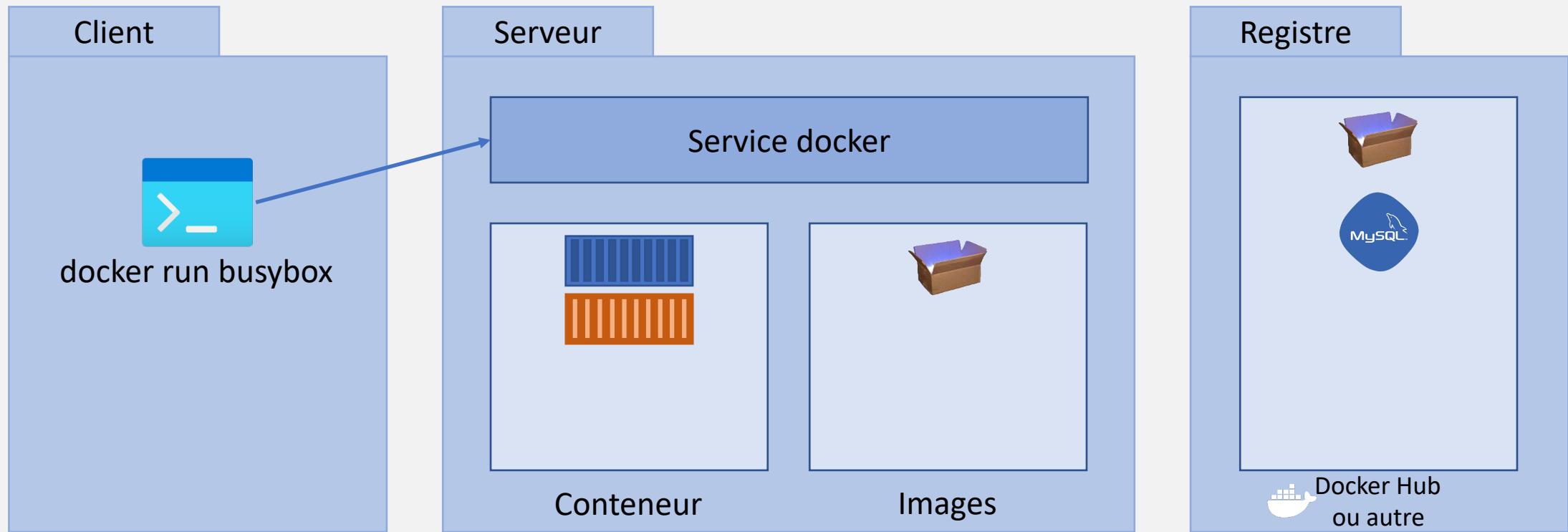
Objectifs

- Retour docker et ses utilisations
- Réseau
- Stockage

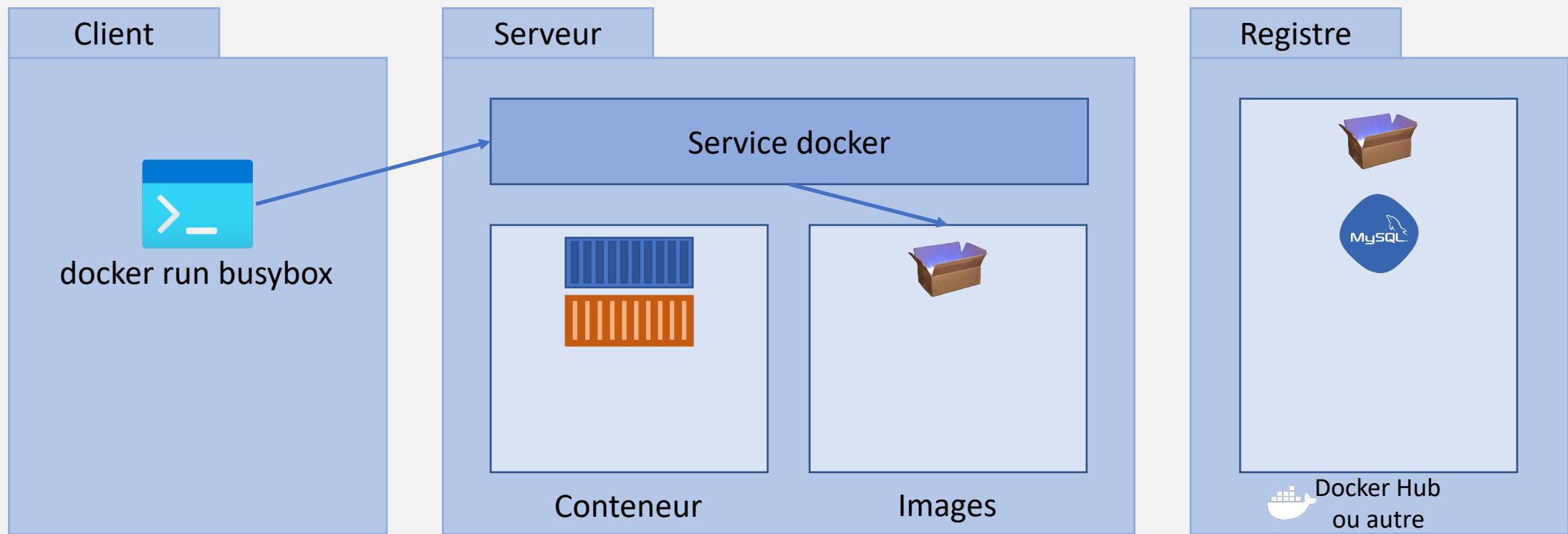
Utilisations

- Déploiement :
 - Léger : ne contient que l'application et ses dépendances
 - Autonome : contient toutes les dépendances
 - Sécurité : isolement du système de fichiers et du réseau
- Développement :
 - Accéder rapidement à un service sans l'installer sur sa machine et risquer de la déstabiliser
 - Dans un environnement complexe, par exemple plusieurs applications qui communiquent ensemble, on peut les déployer et se concentrer sur l'application que l'on développe sans passer beaucoup de temps à installer des pré-requis

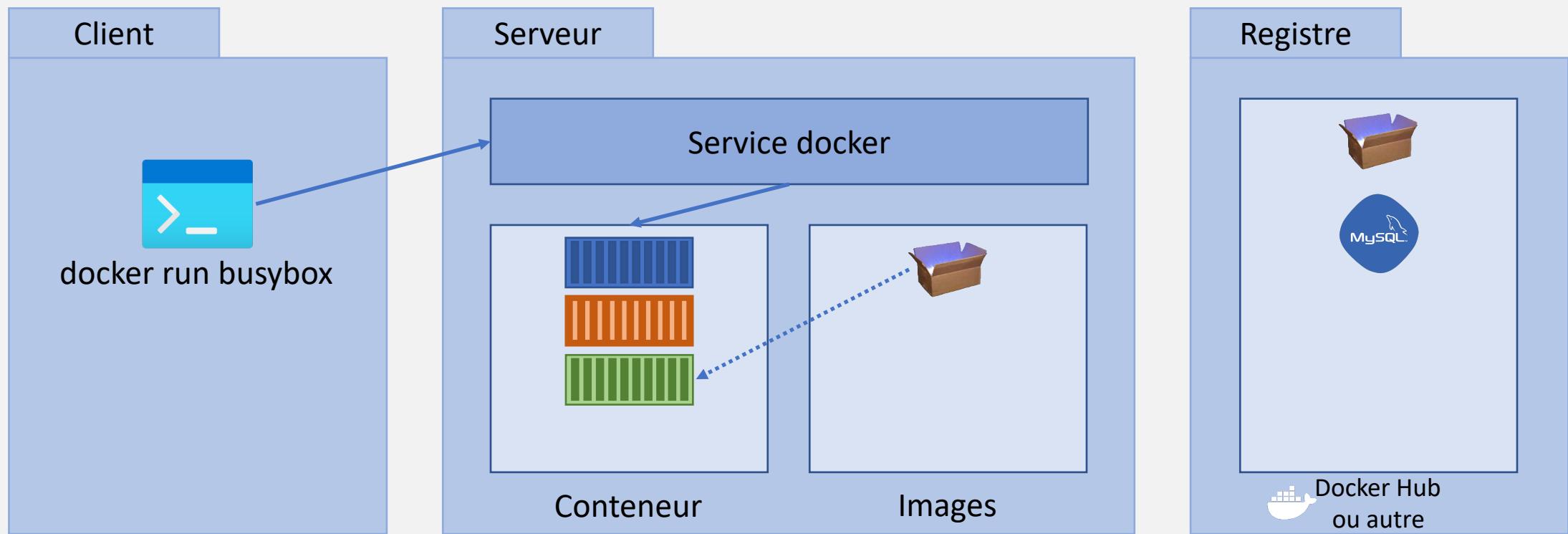
Docker – run



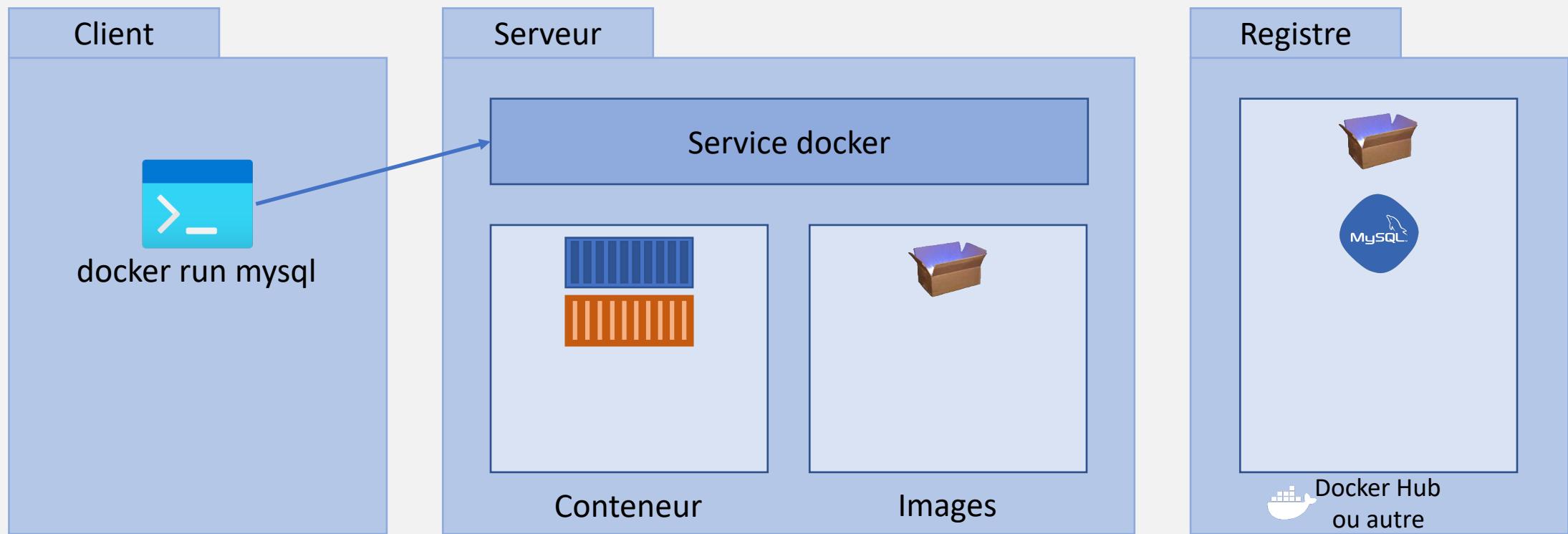
Docker – run



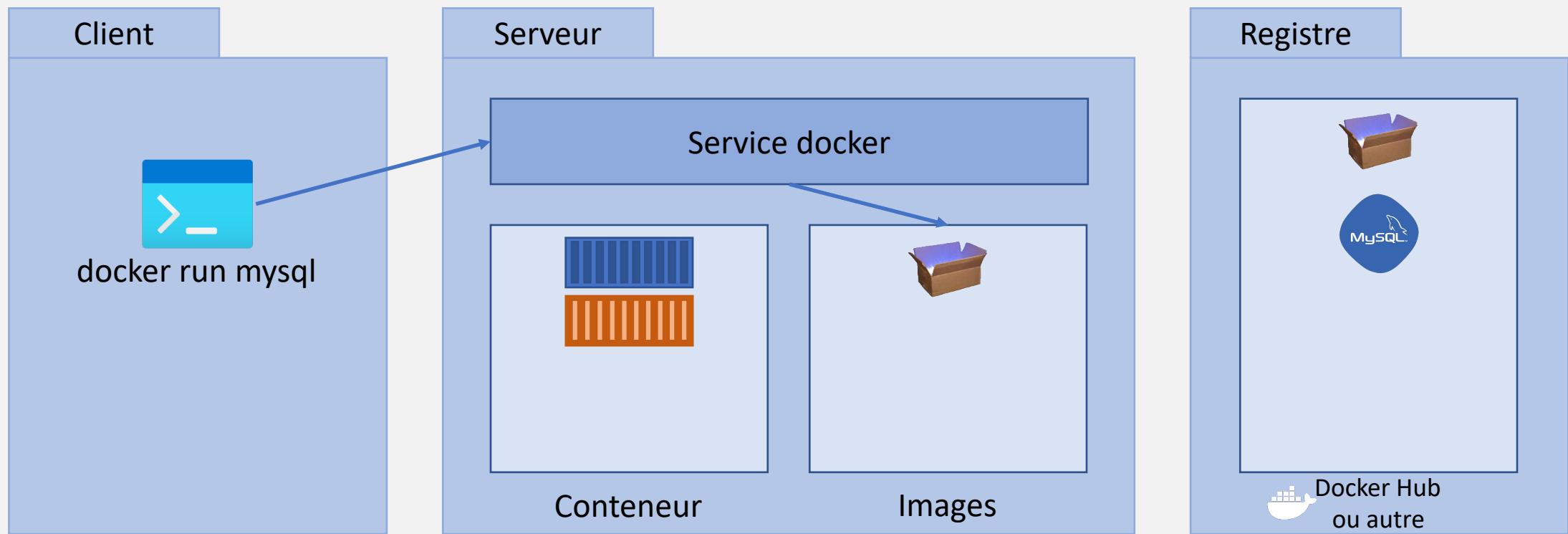
Docker – run



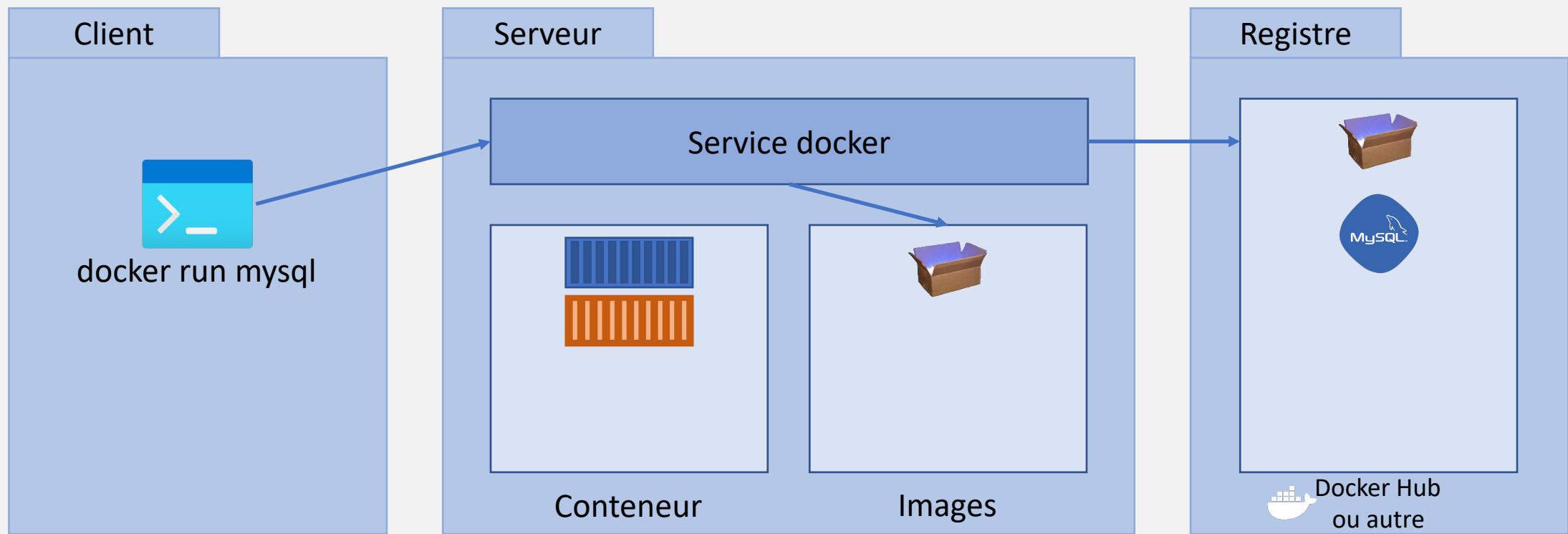
Docker – run



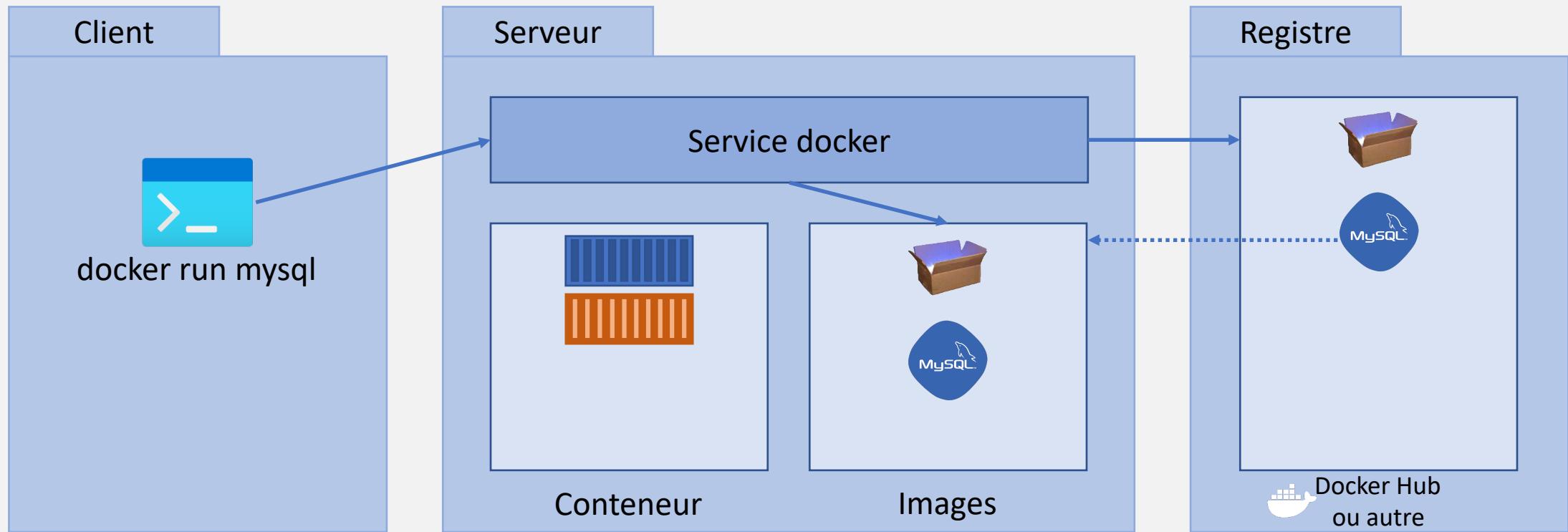
Docker – run



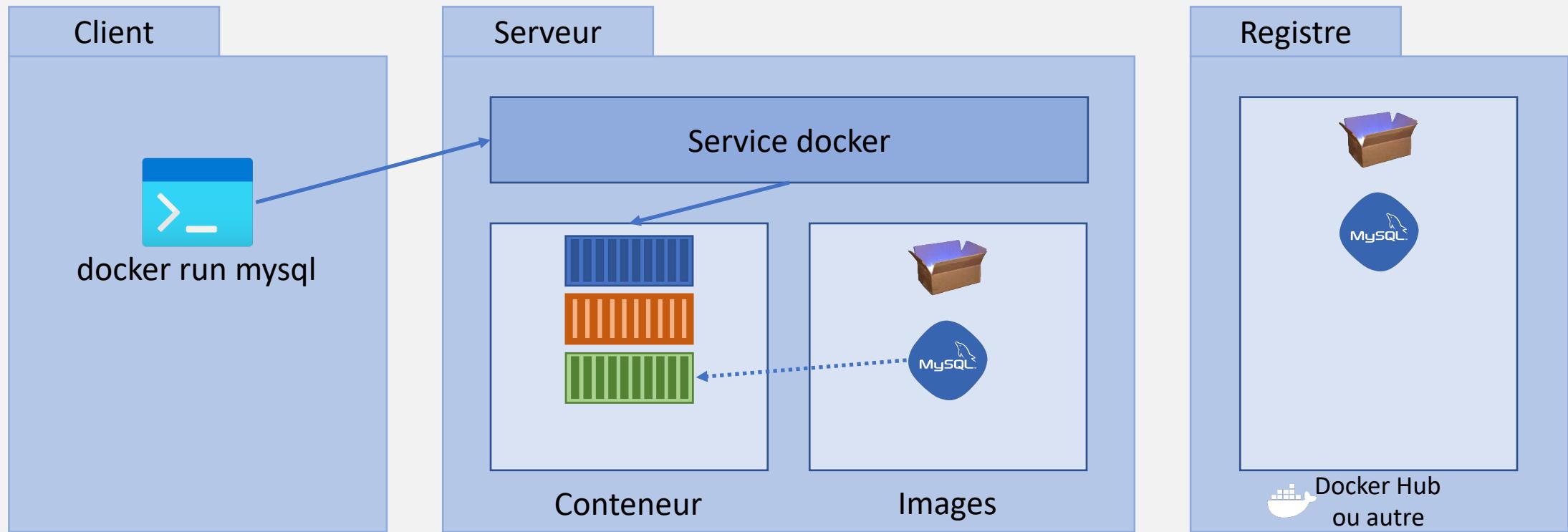
Docker – run



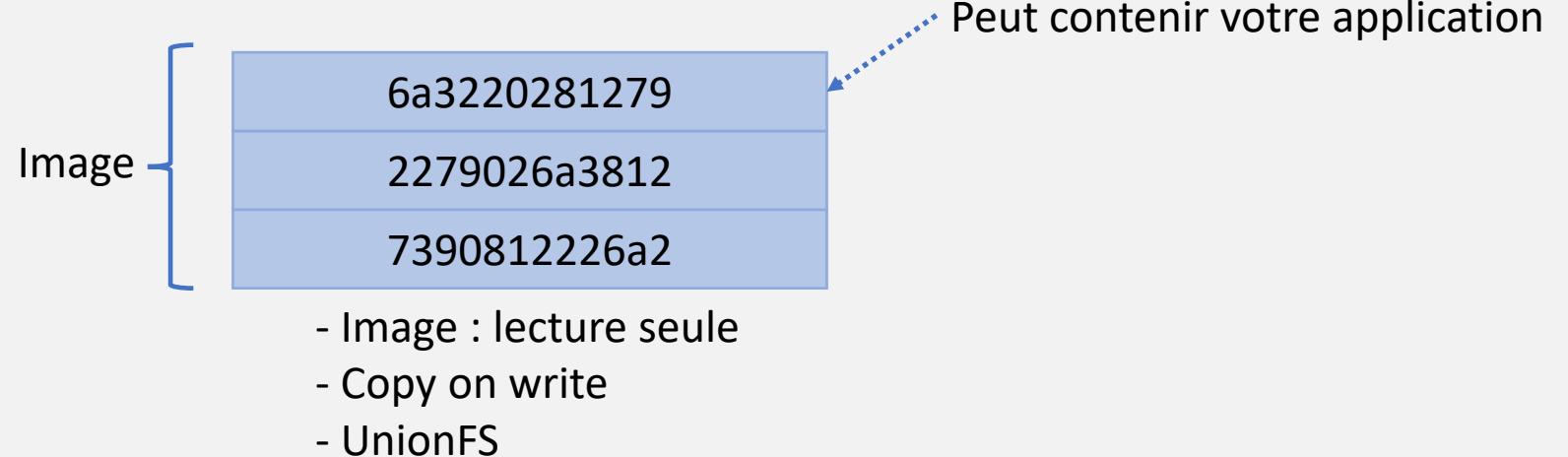
Docker – run



Docker – run



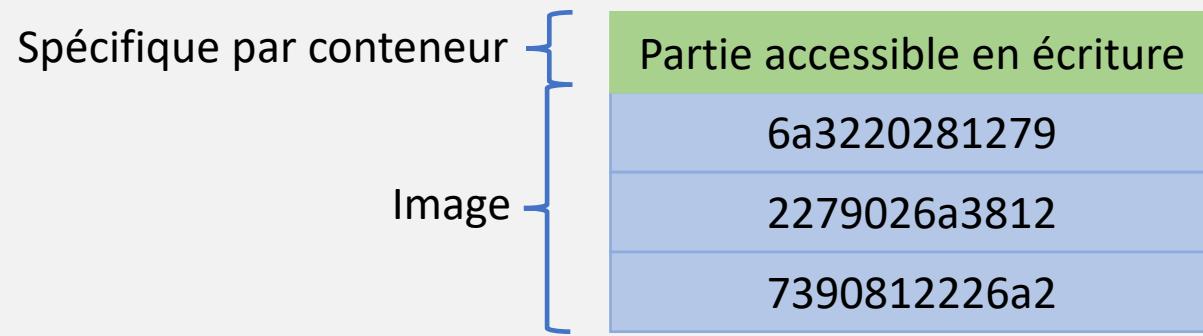
Docker – Image



Création des images à partir d'un Dockerfile

Possibilité de l'envoyer dans un registre comme Docker hub

Docker – Conteneur

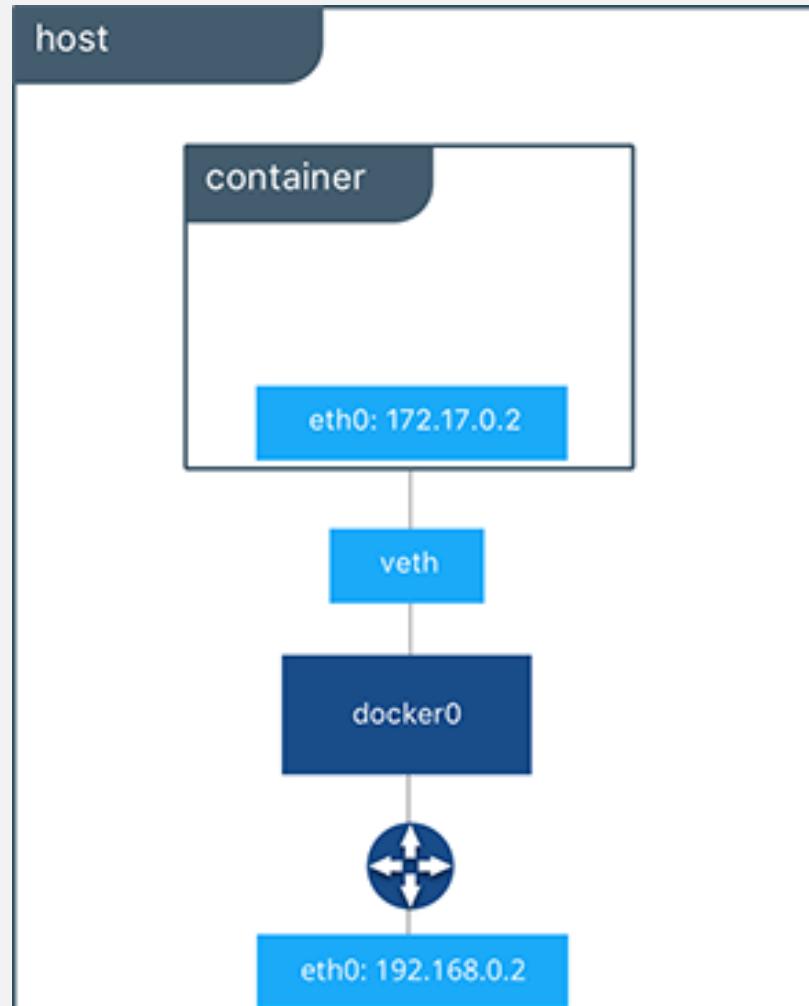


Docker – Réseau

- Plusieurs drivers :
 - Liste : `docker network ls`
 - Propriétés : `docker network inspect bridge`
 - Autre : create, rm, etc.
- Par défaut docker vient avec 3 drivers :
 - bridge : valeur par défaut si non spécifiée, pont – 172.17.0.0/16 (172.17.0.1)
 - host : utilise l'IP de l'hôte, **n'est plus isolé de l'hôte**
 - none : pas de réseau

> docker network ls			
NETWORK ID	NAME	DRIVER	SCOPE
ac4a6903602e	bridge	bridge	local
81bda30d730b	host	host	local
4393a605bac2	none	null	local

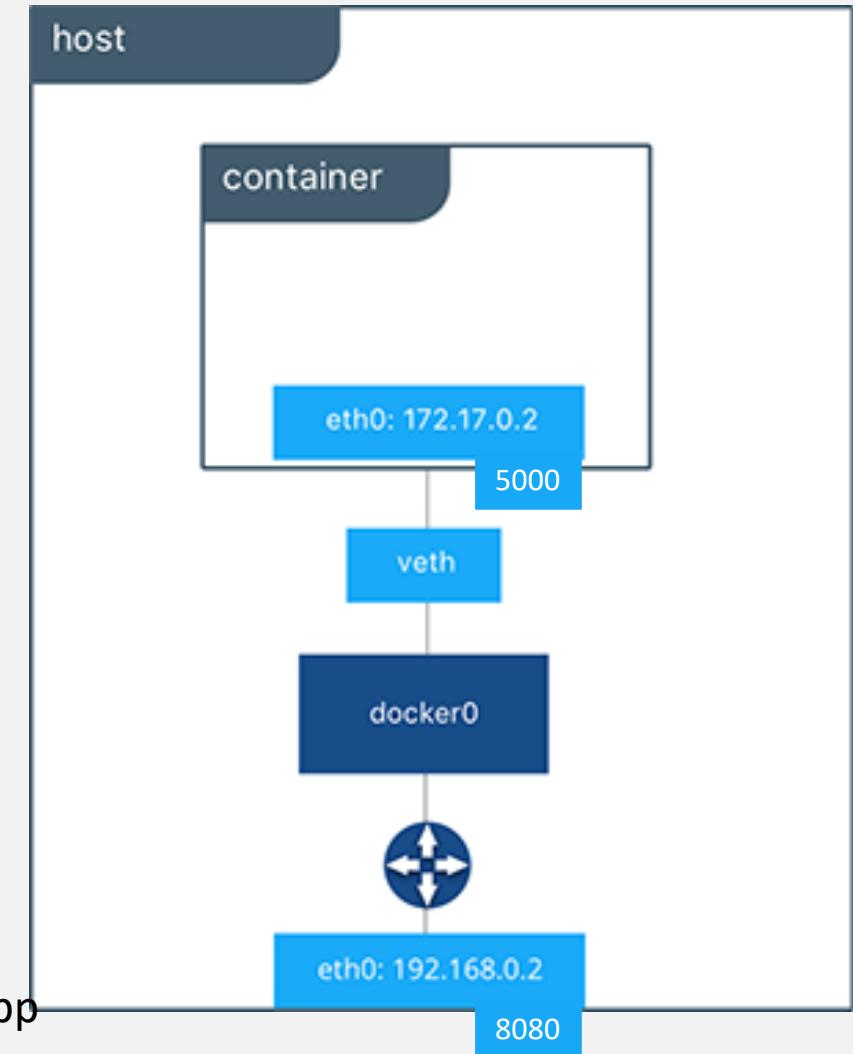
Docker – Réseau / conteneur



Docker – Réseau / conteneur / redirection

```
docker run -p 8080:5000 --name container mon_image
```

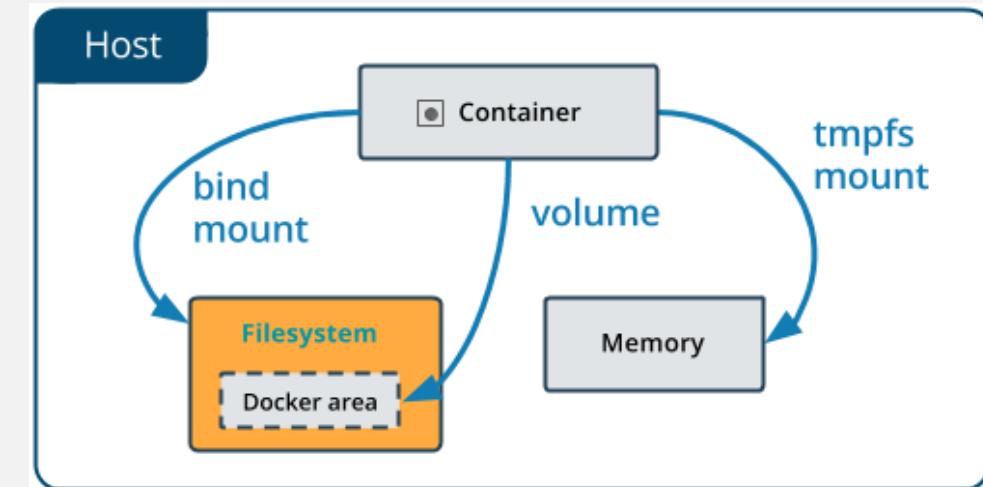
```
docker run --rm -it -p 8000:80  
mcr.microsoft.com/dotnet/core/samples:aspnetapp
```



<https://github.com/dotnet/dotnet-docker/tree/master/samples/aspnetapp>
<https://docs.docker.com/engine/tutorials/networkingcontainers/>

Docker – Stockage persistant

- Docker gère trois types de stockage :
 - Point de montage : monter un répertoire dans le conteneur
 - Volume : garder des données indépendamment du conteneur, spécifique à docker
 - Mémoire : tmpfs



<https://docs.docker.com/storage/bind-mounts/>

Docker – Stockage persistant

- Utilisation des points de montage :

```
docker run -v /mon_repertoire/a_partager:/point_montage  
busybox ls -l /point_montage
```

```
> mkdir /tmp/mon_repertoire_a_partager  
> touch /tmp/mon_repertoire_a_partager/un_fichier  
> touch /tmp/mon_repertoire_a_partager/un_autre_fichier  
> docker run -v /tmp/mon_repertoire_a_partager:/point_montage busybox ls -l /point_montage  
  
total 0  
-rw-r--r--    1 root      root          0 Sep 10 10:54 un_autre_fichier  
-rw-r--r--    1 root      root          0 Sep 10 10:54 un_fichier
```

Docker – Stockage persistant

- Crédation de volume :
 - docker volume create mon_volume
- Lister les volumes :
 - docker volume ls
- L'utilisation :
 - docker run -v mon_volume:/point_montage busybox ls -l /point_montage
- D'autres commandes :
 - prune, rm, inspect

Docker – Stockage persistant

```
> docker volume ls
DRIVER          VOLUME NAME
> docker volume create mon_volume
mon_volume
> docker volume ls
DRIVER          VOLUME NAME
local           mon_volume

> docker run -v mon_volume:/point_montage busybox touch /point_montage/un_fichier
> docker run -v mon_volume:/point_montage busybox ls -l /point_montage
total 0
-rw-r--r--    1 root      root            0 Sep 10 11:12 un_fichier

> docker run -v mon_volume:/point_montage busybox rm /point_montage/un_fichier
> docker run -v mon_volume:/point_montage busybox ls -l /point_montage
total 0
```