

Docker 內 container 如何使用 Docker

在 Docker 容器內部直接執行宿主機上的 Docker 命令有一定的限制，因為 Docker 容器本身是與宿主機隔離的。以下是一些實現方法，根據不同需求選擇適合的解法：

方法 1：將 Docker socket 映射到容器

將宿主機的 Docker socket (`/var/run/docker.sock`) 映射到容器內部，這樣容器內部的 Docker CLI 可以直接與宿主機的 Docker daemon 通訊。

步驟

1. 啟動容器時掛載 Docker socket：

```
docker run -it --rm \
-v /var/run/docker.sock:/var/run/docker.sock \
-v $(which docker):/usr/bin/docker \
your-image
```

2. 在容器內使用 Docker 命令：容器內的 Docker CLI 可以直接操作宿主機的 Docker，執行例如 `docker ps`、`docker run` 等指令。
3. 注意安全性：授予容器訪問 Docker socket 的權限會帶來安全風險，因為容器內的進程可以完全控制宿主機的 Docker。僅在信任的環境下使用此方法。

方法 2：使用 Docker-in-Docker (DinD)

如果希望在容器內運行獨立的 Docker daemon，可以使用 Docker 官方的 `docker:dind` 映像。

步驟

1. 啟動 DinD 容器：

```
docker run --privileged --name dind-container -d docker:dind
```

2. 進入容器：

```
docker exec -it dind-container sh
```

3. 在容器內執行 Docker 命令（這是獨立的 Docker daemon，不會影響宿主機的 Docker）。

方法 3：使用 Docker CLI 並通過 API 操作

宿主機的 Docker daemon 提供 REST API，可以在容器內安裝 Docker CLI 並通過 API 操作。

步驟

1. 確保宿主機的 Docker daemon 開啟遠程 API，編輯 `/etc/docker/daemon.json`：

```
{
  "hosts": ["unix:///var/run/docker.sock", "tcp://0.0.0.0:2375"]
}
```

然後重啟 Docker：

```
sudo systemctl restart docker
```

2. 啟動容器，並配置環境變數 `DOCKER_HOST` 指向宿主機 API：

```
docker run -it --rm \
-e DOCKER_HOST=tcp://<host-ip>:2375 \
your-image
```

3. 在容器內執行 Docker 命令：

```
docker ps
```

注意事項

- 1. **安全風險：**
 - 授予容器訪問 Docker socket 或 Docker daemon 的權限可能導致安全漏洞。
 - 確保只在受信任的環境中使用，並採取必要的隔離措施（例如：限制網路訪問、使用 TLS 保護 API）。
- 2. **性能影響：**
 - 如果容器內需要頻繁操作宿主機的 Docker，可能會影響宿主機的性能。
- 3. **權限問題：**
 - 確保容器內的進程有足夠的權限訪問 `/var/run/docker.sock` 或 Docker API。

希望這些方法能幫助你解決問題！如果需要更詳細的操作指引，可以告訴我具體的需求或環境設定。