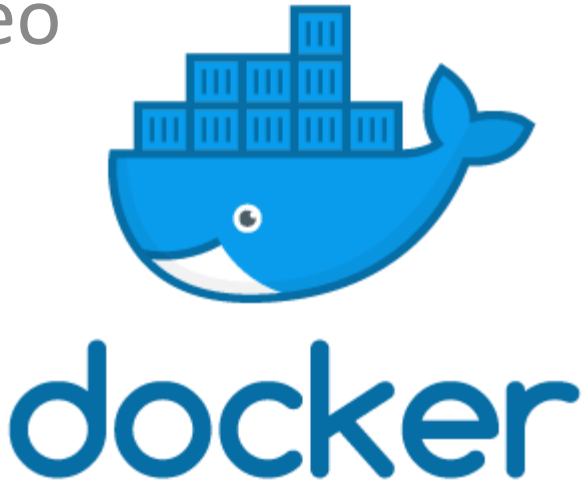
Docker

Riccardo Cattaneo



Storage Drivers

I Container necessitano di uno spazio sul disco locale per poter effettuare la stratificazione delle immagini e per poter montare il filesystem.

La componente che si occupa della gestione di questo spazio su disco è lo STORAGE DRIVERS. E' uno spazio sul disco NON persistente e dura per l'intero ciclo di vita del container.

I Volumi

In docker abbiamo la possibilità di memorizzare i dati in modo persistente e non persistente. La scelta dipende dall'utilizzo che dobbiamo fare dei dati.

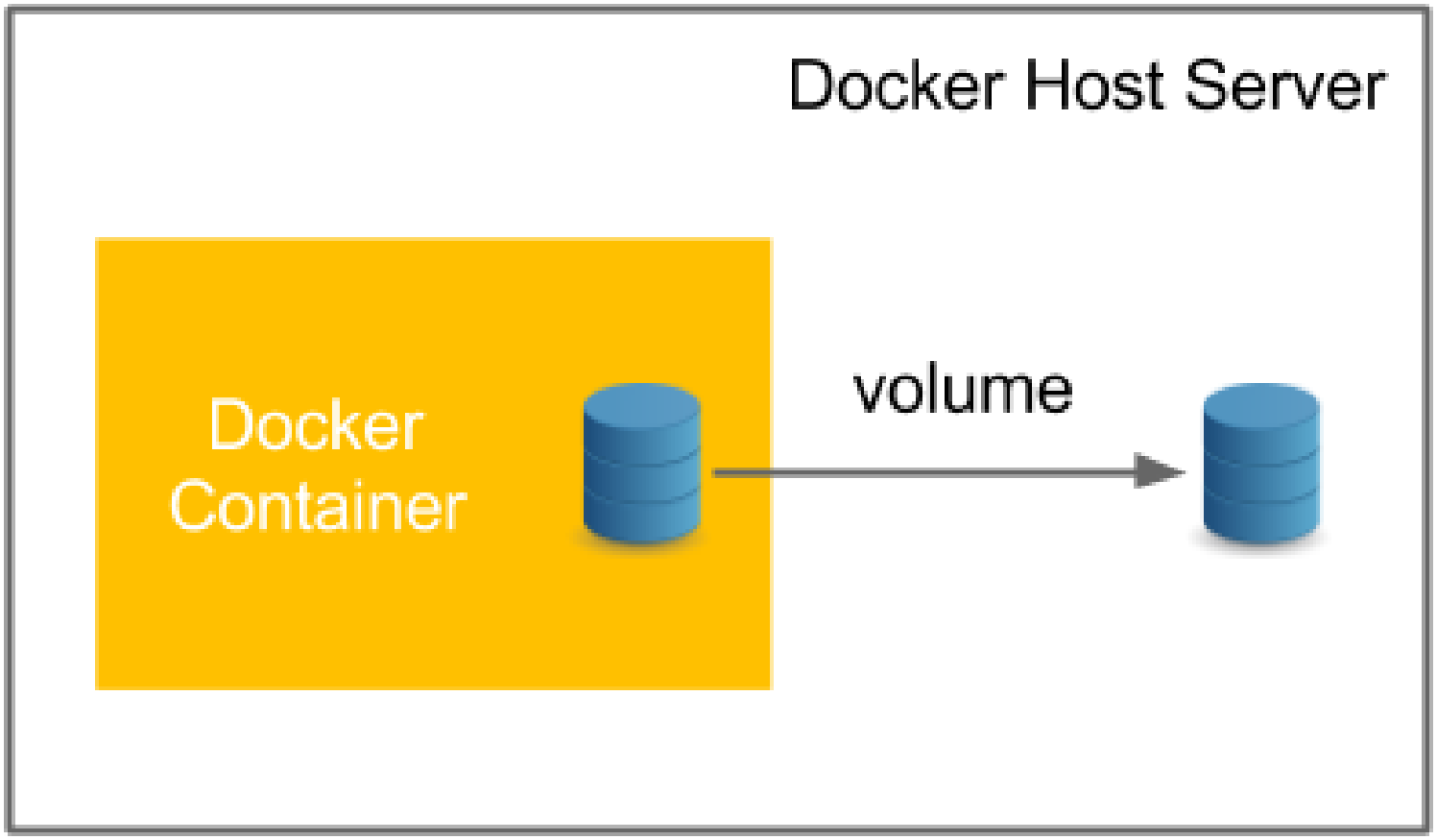
I Container docker hanno uno spazio creato automaticamente per la memorizzazione non persistente dei dati che dura l'intero ciclo di vita del container che, come abbiamo visto avviene grazie allo Storage Driver.

Se la nostra esigenza è avere a disposizione i dati oltre la durata del container, allora dobbiamo introdurre il concetto di **Volume**.

Quali sono i vantaggi? Separare i container dallo storage. Condividere un volume tra container differenti. Non perdere i dati ad eliminazione del container.

Questi sono i passaggi

- Creare il Volume
- Creare il container
- Montiamo il volume all'interno del container
- In fase di «Montaggio» dobbiamo specificare una cartella del filesystem del container
- I dati saranno scritti in questa directory
- Se il container sarà eliminato i dati continueranno ad esistere.


Docker Host Server
Docker
Container
volume

I Comandi Principali

I comandi principali per la gestione dei volumi sono :

- **docker volume create nomeVolume** (per creare un volume)
- **docker volume ls** (per visualizzare i volumi creati)
- **docker volume inspect nomeVolume** (per ispezionare un volume)
- **docker volume rm nomeVolume** (per eliminare uno specifico volume)
- **docker volume prune** (per eliminare tutti i volumi non utilizzati)

Esempio

Se digitiamo dalla nostra console il seguente comando :

docker volume - -help

Otteniamo l'elenco di tutti i comandi che possiamo utilizzare con volume :

- create
- inspect
- ls
- prune
- rm

- docker volume create primovolume
- docker volume ls
- docker volume inspect primovolume
- docker volume rm primovolume
- docker volume ls

Una volta fatto l'esempio andiamo a vedere come è possibile utilizzare un volume in un container.

Volume - Container

Per creare un volume all'interno di un container bisogna utilizzare l'opzione **–v** nel momento in cui creo il container. Andiamo a creare per prima cosa il volume :

docker volume create volumeprimo

docker run –it - -name containerprimo –v volumeprimo:/test ubuntu bash

Cosa abbiamo fatto ? Con l'opzione –v abbiamo associato il volume «volumeprimo» ad una cartella del container ubuntu (in questo caso la cartella test) e i due punti servono da separatore tra il volume e dove deve essere visualizzato.

In questo caso, nel «montare» il volume su un container, posso specificare un percorso locale di windows. In questo caso posso scrivere :

```
docker run –it - -name containerprimo –v
d:\Docker:/test ubuntu bash
```

Una volta lanciato il comando possiamo verificare che è stata creata la cartella test all'interno del nostro container tramite il comando «ls». A questo punto possiamo creare un file all'interno di questo volume, quindi :

```
cd test
touch prova.txt
```

Adesso proviamo a stoppare e ad eliminare questo container e vedere se i dati continuano a persistere.

Comando EXIT

Vediamo ora qualche scorciatoia. Per uscire è sufficiente che al suo interno scrivo il comando **exit**.

Una volta digitato il comando premere invio e ci ritroveremo nel nostro terminale. Procediamo ora all'eliminazione del container con il comando

docker rm containerprimo

Dopo aver eliminato il container verifichiamo che il volume esiste ancora con il seguente comando

docker volume ls

Possiamo ora verificare che il volume esiste e che è presente anche il nostro file di prova :

docker volume inspect volumeprimo

Attenzione

E’ possibile creare direttamente il volume durante la creazione del container. Il comando è sempre lo stesso, in automatico docker controlla se il volume esiste, se non esiste lo crea, quindi potevo direttamente scrivere il seguente comando :

docker run –it - -name containerprimo –v volumesecondo:/test ubuntu bash

Condivisione dei Volumi

Si può condividere il volume su più container. Non bisogna fare nulla di particolare, è sufficiente che, quando si creano i container si fa riferimento allo stesso volume. Ad esempio :

docker run –it - -name container1 –v volumeprincipale:/test ubuntu bash

docker run –it - -name container2 –v volumeprincipale:/test ubuntu bash

Possiamo ora verificare che se creo un file in uno dei due container, lo trovo anche se eseguo il secondo container. Proviamo ad eseguire questi comandi :

- docker exec –it container1 bash
- cd test
- touch pippo.txt
- CTRL+P CTRL+Q
- docker exec –it container2 bash
- cd test
- ls

Mapping verso una Location Fisica su Linux

Abbiamo la possibilità di utilizzare per la persistenza dei dati i volumi oppure di mappare anche una location fisica del nostro host tramite sempre il comando -v.

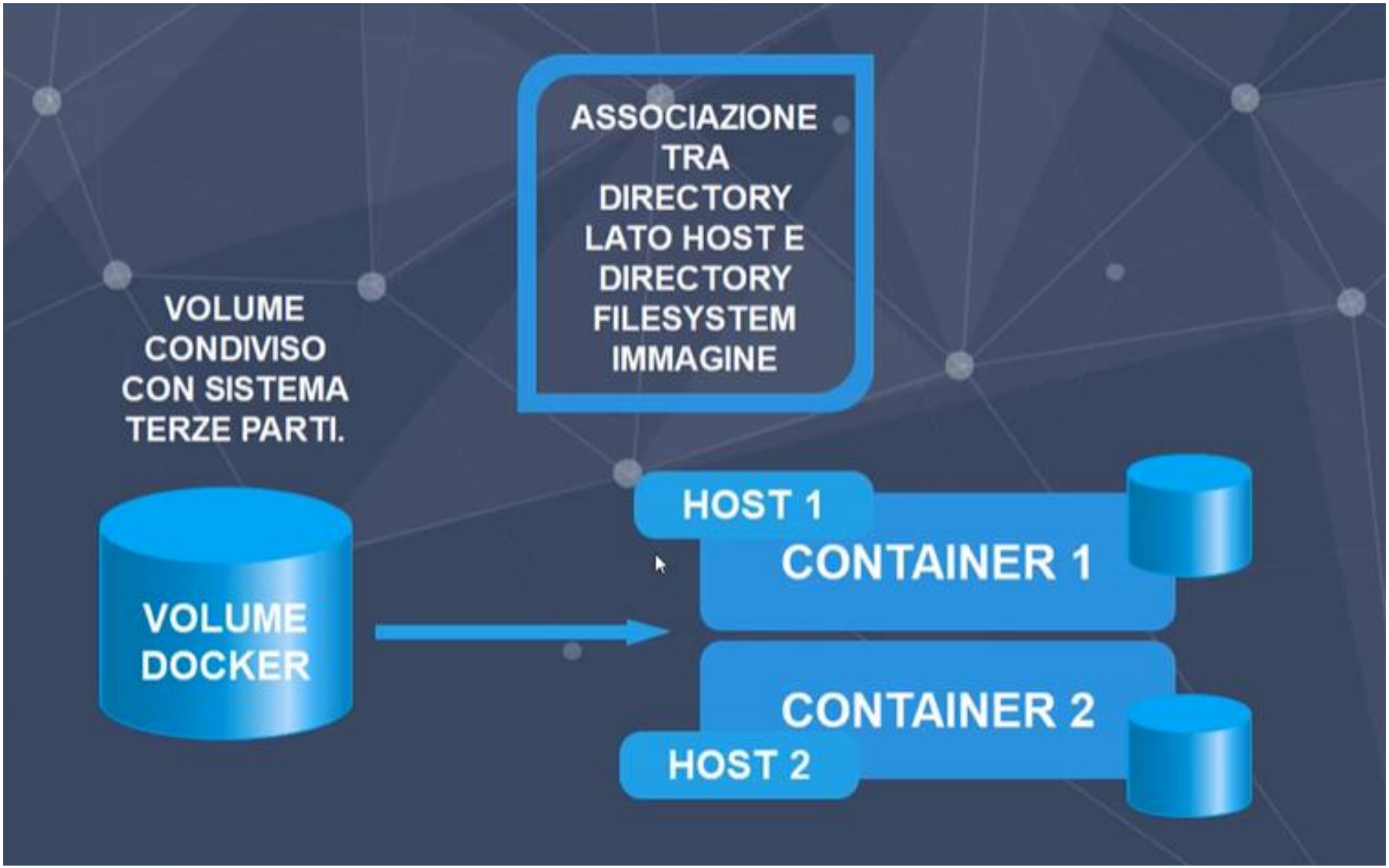
Se inseriamo nel comando –v solo il nome, automaticamente utilizzeremo il volume indicato e se non esiste lo crea. Invece utilizzando il path assoluto, quindi il percorso del nostro pc locale, utilizzeremo una directory del nostro host. Facciamo subito un esempio :

Per fare un esempio andiamo a creare una cartella all'interno del nostro host, ad esempio sotto la nostra directory principale andiamo a creare una cartella «cartellaTest»

- **pwd** (per vedere il percorso)
- **cd Documenti** (entriamo nella cartella documenti)
- **mkdir cartellaTest** (creo la cartella «cartellaTest»)
- **docker run –it - -name container3 –v /home/riccardo/cartellaTest:/test ubuntu bash**

Una volta avviato il nostro container andiamo nella cartellaTest e proviamo a creare un file. Dopo averlo creato usciamo dal container (CTRL P + CTRL Q) ed andiamo a controllare che nella cartella del nostro host fisico sia presente il file appena creato nel container.

Possiamo provare anche a fare il contrario, cioè creare un file dal nostro host, poi accedere al container con exec e verificare che il file sia presente.


ASSOCIAZIONE TRA DIRECTORY LATO HOST E DIRECTORY FILESYSTEM IMMAGINE
VOLUME CONDIVISO CON SISTEMA TERZE PARTI.
VOLUME DOCKER
HOST 1
CONTAINER 1
CONTAINER 2
HOST 2