Linux Ubuntu

1. Shell Linux

1.1 Tổng quan

1.2 Lịch sử

1.3 Các command căn bản

Cd: lệnh để chuyển đến thư mục làm việc mong muốn

Ls: Liệt kê danh sách file và thư mục có trong thư mục hiện tại

Cp: Copy nội dung file và thư mục tham số

Mv: Di chuyển file đến vị trí mong muốn, đổi tên file nếu trong cùng thư mục

Mkdir: Lệnh tạo thư mục mới

Cat: Hiển thị nội dung file

Head: hiển thị số dòng chỉ định trong nội dung file

1.4 Các command liên quan đến quyền

Chmod: Dùng để thay đổi quyền truy cập vào tập tin hoặc thư mục

Chown: Thay đổi chủ sở hữu của tập tin và thư mục

Ls –l: Hiển thị chi tiết danh sách file và thư mục có trong file bao gồm các thông tin về quyền truy cập, chủ sở hữu, nhóm, kích thước, thời gian chỉnh sửa

1.5 Các command thực hiện song song

Sử dụng ký tự “ | “ giữa 2 lệnh để thực hiện như là một toán tử đường ống. đầu ra của lệnh trước đó được lấy làm đầu vào cho lệnh tiếp theo

Wc: đếm sô dòng, số từ, số ký tự trong một file, wc –l chỉ đếm số dòng

Grep: thực hiện tìm kiếm chuỗi ký tự và hiển thị những dòn chứa cụm đó trong file

Echo thực hiện ghi dữ liệu vào file chỉ định

1.6 Sử dụng vim

Sử dụng lệnh vim để tiến hành tạo file hoặc thực hiện truy hập vào nội dung file để sửa đổi. Thực hiện lệnh :w để lưu tác vụ sửa đổi. Lệnh :q để thoát và :q! để thoát mà không thay đổi nội dụng file.

1.7 Lệnh quản lý tiến trình

Htop để thực hiện hiển thị các tiến trình đang chạy, tài nguyên sử dụng và tương tác với chúng

Ps aux: Cung cấp thông tin chi tiết về các tiến trình đang chạy bao gồm danh sách tiến trình, ID, tệp thực thi, người dùng,…

Kill -9: Lệnh thực hiện cưỡng chế dừng hoạt động của một tiến trình kể cả khi công việc của tiến trình chưa thực hiện được

2 Docker

2.1 Tổng quan

Docker là một nền tảng mở để phát triển, vận chuyển và chạy ứng dụng. Với Docker, bạn có thể tách ứng dụng khỏi hạ tầng và nhanh chóng triển khai phần mềm. Điều này giúp bạn quản lý hạ tầng và ứng dụng dễ dàng hơn. Docker cho phép bạn đóng gói và chạy ứng dụng trong các container cô lập. Các container này nhẹ và chứa đầy đủ mọi thứ để chạy ứng dụng, giúp bạn không phụ thuộc vào việc cài đặt môi trường trên máy chủ. Bạn cũng có thể chia sẻ các container này và đảm bảo rằng mọi người sử dụng cùng một container hoạt động giống nhau. Docker cung cấp công cụ và nền tảng để quản lý vòng đời của các container. Bạn có thể phát triển và kiểm thử ứng dụng sử dụng container, sau đó triển khai ứng dụng vào môi trường sản xuất. Điều này hoạt động giống nhau cho các môi trường địa phương và điện toán đám mây.

2.2 Chức năng

Docker có thể dùng để thực hiện các công việc sau:

1. Docker giúp phát triển và triển khai ứng dụng nhanh chóng và nhất quán bằng cách sử dụng các container địa phương, đồng thời hỗ trợ quy trình CI/CD.

2. Với Docker, bạn có thể di chuyển và tỷ lệ mở rộng ứng dụng dễ dàng trên nhiều môi trường như máy tính cá nhân, máy chủ vật lý hoặc ảo, đám mây hoặc môi trường kết hợp.

3. Docker cho phép chạy nhiều công việc trên cùng phần cứng một cách linh hoạt và hiệu quả, giúp tiết kiệm tài nguyên và chi phí.

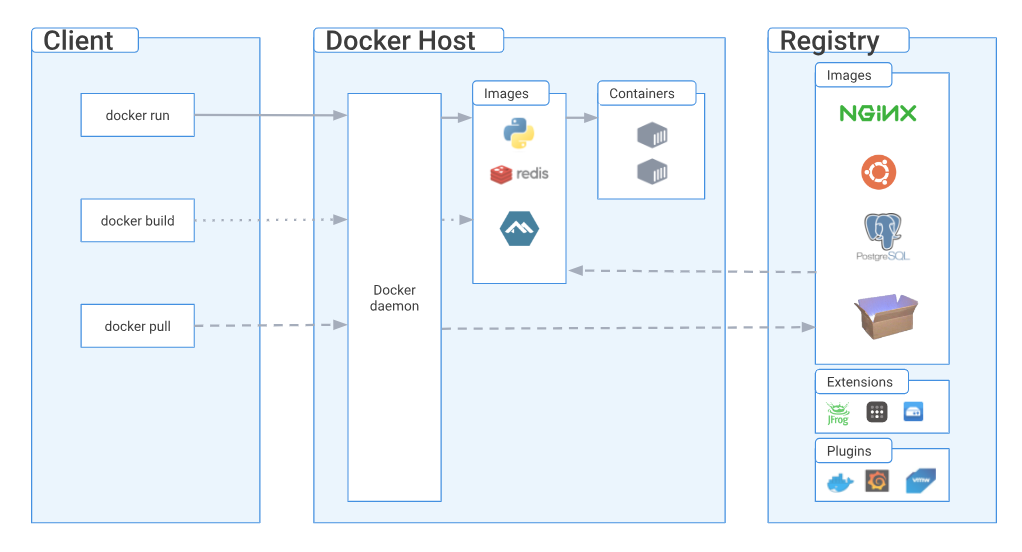
2.3 Kiến trúc

Kiến trúc

Kiến trúc Docker

Docker sử dụng một kiến trúc client-server. Docker client trò chuyện với Docker daemon, làm nhiệm vụ xây dựng, chạy và phân phối các container Docker. Docker client và daemon có thể chạy trên cùng một hệ thống hoặc bạn có thể kết nối một Docker client tới một Docker daemon từ xa. Docker client và daemon giao tiếp sử dụng REST API, qua socket UNIX hoặc giao diện mạng. Một client Docker khác là Docker Compose, cho phép bạn làm việc với các ứng dụng bao gồm một tập hợp các container.

Sơ đồ kiến trúc Docker



Daemon Docker

Daemon Docker (dockerd) lắng nghe các yêu cầu Docker API và quản lý các đối tượng Docker như images, container, networks và volumes. Một daemon cũng có thể truyền thông với các daemon khác để quản lý các dịch vụ Docker.

Client Docker

Client Docker (docker) là cách chính mà nhiều người dùng Docker tương tác với Docker. Khi bạn sử dụng các lệnh như docker run, client gửi các lệnh này tới dockerd, và dockerd thực hiện chúng. Lệnh docker sử dụng Docker API. Client Docker có thể kết nối với nhiều daemon.

Docker Desktop

Docker Desktop là một ứng dụng dễ cài đặt cho môi trường Mac, Windows hoặc Linux, cho phép bạn xây dựng và chia sẻ các ứng dụng và dịch vụ được đóng gói thành container. Docker Desktop bao gồm Docker daemon (dockerd), Docker client (docker), Docker Compose, Docker Content Trust, Kubernetes và Credential Helper. Để biết thêm thông tin, xem Docker Desktop.

Docker registries

Docker registry lưu trữ các Docker images. Docker Hub là một registry công cộng mà ai cũng có thể sử dụng, và Docker mặc định tìm kiếm images trên Docker Hub. Bạn cũng có thể chạy một registry riêng.

Khi bạn sử dụng lệnh docker pull hoặc docker run, Docker tải các image cần thiết từ registry đã được cấu hình của bạn. Khi bạn sử dụng lệnh docker push, Docker đẩy image của bạn lên registry đã được cấu hình.

Docker Object

Các đối tượng Docker là các thành phần quan trọng khi sử dụng Docker. Có những đối tượng chính như images và containers.

Images là các mẫu chỉ đọc để tạo ra các containers. Chúng được xây dựng từ các images khác, có thể tùy chỉnh bổ sung theo nhu cầu. Để tạo một image riêng, ta sử dụng một Dockerfile chứa các bước cần thiết để tạo ra image. Mỗi bước tạo ra một layer trong image, giúp image trở nên nhẹ và nhanh hơn.

Containers là phiên bản chạy của các image. Ta có thể tạo, khởi động, dừng, di chuyển và xóa containers. Containers có thể được kết nối với mạng và lưu trữ, và dễ dàng tạo ra các image mới dựa trên trạng thái hiện tại của container. Mỗi container được xác định bởi image và các tùy chọn cấu hình.

Ngoài images và containers, Docker còn có các đối tượng khác như network, volume và plugin. Chúng giúp quản lý cách container giao tiếp với môi trường bên ngoài và chia sẻ dữ liệu.

Công nghệ cơ bản

Docker được viết bằng ngôn ngữ lập trình Go và tận dụng nhiều tính năng của nhân Linux để cung cấp các chức năng của nó. Docker sử dụng một công nghệ được gọi là namespaces để cung cấp không gian làm việc cách ly được gọi là container. Khi bạn chạy một container, Docker tạo ra một tập hợp các namespaces cho container đó.

Các namespaces này cung cấp một lớp cách ly. Mỗi mặt của một container chạy trong một namespace riêng biệt và quyền truy cập của nó chỉ giới hạn trong namespace đó.