BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC DÂN LẬP HẢI PHÒNG

-------o0o-------

ISO 9001:2015

# ĐỒ ÁN TỐT NGHIỆP

## NGÀNH CÔNG NGHỆ THÔNG TIN

HẢI PHÒNG 2019

BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC DÂN LẬP HẢI PHÒNG

-------o0o-------

# TÌM HIỂU GIẢI PHÁP ẢO HÓA DOCKER VÀ ỨNG DỤNG

ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC HỆ CHÍNH QUY
Ngành: Công nghệ Thông tin

HẢI PHÒNG - 2019

BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC DÂN LẬP HẢI PHÒNG

-------o0o-------

# TÌM HIỂU GIẢI PHÁP ẢO HÓA DOCKER VÀ ỨNG DỤNG

ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC HỆ CHÍNH QUY
Ngành: Công nghệ Thông tin

Sinh viên thực hiện: Đinh Hải Long
Mã số sinh viên: 1412101066
Giảng viên hướng dẫn: TS. Ngô Trường Giang

HẢI PHÒNG - 2019

BỘ GIÁO DỤC VÀ ĐÀO TẠO
TRƯỜNG ĐẠI HỌC DÂN LẬP HẢI PHÒNG
-------o0o-------

# NHIỆM VỤ THIẾT KẾ TỐT NGHIỆP

Sinh viên: Đinh Hải Long Mã số: 1412101099

Lớp: CT1802 Ngành: Công nghệ Thông tin

Tên đề tài: Tìm hiểu giải pháp ảo hóa docker và ứng dụng

# LỜI CẢM ƠN

Trong quá trình làm đồ án vừa qua vì được sự chỉ dẫn nhiệt tình của thầy TS. Ngô Trường Giang – Trường Đại học Dân lập Hải Phòng , em đã hoàn thành đồ án của mình. Mặc dù em đã cố gắng với sự tận tâm của thầy, nhưng vì thời gian và khả năng nên đồ án của em vẫn còn không tránh được những điều thiếu sót.

Em xin chân thành và bày tỏ lòng biết ơn sâu sắc đến thầy Ngô Trường Giang vì đã tận tình chỉ bảo, hướng dẫn và giành thời gian quý báu của mình cho em trong thời gian qua để em có thể hoàn thành đồ án của mình đúng thời hạn.

Em xin cảm ơn tất cả thầy cô giáo trong khoa Công nghệ thông tin vì đã truyền đạt cho em rất các kiến thức nền tảng, chuyên ngành, chuyên môn và chuyên sâu cực kì vững chắc trong những năm qua để em có thể hoàn thanh được đồ án này.

Em xin cảm ơn Trường Đại Học Dân Lập Hải Phòng vì không ngừng hỗ trợ và tạo những điều kiện tốt nhất trong những năm vừa qua để em có thể học và thực hiện tốt đồ án.

Em xin cảm ơn gia đình, bạn bè đã hỗ trợ và cổ vũ cho em trong suốt quá trình học tập cũng như làm đồ án để em có thể hoàn thành khóa học và đồ án theo quy định.

Em xin chân thành cảm ơn!

# MỤC LỤC

# DANH MỤC HÌNH MINH HỌA

# MỞ ĐẦU

Ảo hóa là công nghệ cho phép chạy đồng thời nhiều VM (Virtual Machine) trên cùng phần cứng vật lý . Cùng chia sẽ tài nguyên phần cứng và được quản lý bởi lớp ảo hóa (Hypervisor), được quản lý cấp phát tài nguyên hợp lý, tránh lãng phí …

Một số trung tâm dữ liệu chỉ sử dụng 10% đến 30% năng lực xử lý hiện có của họ. Ảo hóa đã giúp nhiều tổ chức có thể chia sẻ các tài nguyên công nghệ thông tin theo cách tốn ít giá thành nhất, làm cho cơ sở hạ tầng công nghệ thông tin trở nên linh động và bảo đảm cung cấp một cách tự động với những nhu cầu cần thiết. Các doanh nghiệp luôn tìm giải pháp để tiết kiệm hơn, đây cũng là lúc công nghệ ảo hóa tìm được chỗ đứng vững chắc trong lĩnh vực công nghệ thông tin trên thế giới. Sử dụng công nghệ ảo hóa đã đem đến cho người dùng sự tiện ích, có thể chạy nhiều hệ điều hành, nhiều hệ thống đồng thời trên cùng một hệ thống phần cứng máy chủ, mở rộng khả năng lưu trữ, cung cấp tài nguyên phần cứng.

Công nghệ ảo hóa Docker là công nghệ mới, có khả năng phát triển mạnh mẽ trong tương lai. Đây là lý do em chọn đề tài “Tìm hiểu giải pháp ảo hóa Docker và ứng dụng” để triển khai.

Đồ án này trình bày về giải pháp ảo hóa Docker và ứng dụng, nội dung của đồ án bao gồm:

Chương 1: Tổng quan về công nghệ ảo hóa.

Chương 2: Công nghệ ảo hóa Docker.

Chương 3: Thử nghiệm ảo hóa ứng dụng với Docker.

# CHƯƠNG 1: Tổng quan về ảo hóa

## 1.1 Tổng quan về ảo hóa

### 1.1.1 Ảo hóa

Ảo hóa là việc chia phần cứng vật lý thành nhiều phần cứng ảo. Vì vậy, có thể nói ảo hóa là việc chia một máy vật lý thành nhiều máy con ảo.

Công nghệ ảo hóa là một công nghệ thực hiện ảo hóa trên máy tính, bao gồm các kỹ thuật và quy trình thực hiện ảo hóa. Các kỹ thuật và quy trình này để tạo ra một tầng trung gian giữa hệ thống phần cứng máy tính và phần mềm chạy trên nó. Ý tưởng ban đầu của công nghệ ảo hóa là từ một máy vật lý đơn lẻ có thể tạo thành nhiều máy ảo độc lập. Nó cho phép tạo nhiều máy ảo trên một máy chủ vật lý, mỗi một máy ảo cũng được cấp phát tài nguyên phần cứng như máy thật gồm có RAM, CPU, Card mạng, ổ cứng, các tài nguyên khác và hệ điều hành riêng. Khi chạy ứng dụng, người sử dụng không nhận biết được ứng dụng đó chạy trên lớp phần cứng ảo. người sử dụng chỉ chú ý tới khái niệm logic về tài nguyên máy tính hơn là khái niệm vật lí về tài nguyên máy tính[2].

Máy chủ trong các hệ thống CNTT ngày nay thường được thiết kế để chạy một hệ điều hành và một ứng dụng. Điều này không khai thác triệt để hiệu năng của hầu hết các máy chủ rất lớn. Ảo hóa cho phép ta vận hành nhiều máy chủ ảo trên cùng một máy chủ vật lý, dùng chung các tài nguyên của một máy chủ vật lý qua nhiều môi trường khác nhau. Các máy chủ ảo khác nhau có thể vận hành nhiều hệ điều hành và ứng dụng khác nhau trên cùng một máy chủ vật lý.

Kỹ thuật ảo hóa đã không còn xa lạ kể từ khi Vmware giới thiệu sản phẩm Vmware Workstation đầu tiên vào năm 1999. Sản phẩm này ban đầu được thiết kế để hỗ trợ việc phát triển và kiểm tra phần mềm. Nó đã trở lên phổ biến nhờ khả năng tạo ra những máy tính "ảo" chạy đồng thời nhiều hệ điều hành khác nhau trên cùng một máy tính "thực"(khác với chế độ "khởi

động kép ” - máy tính được cải nhiều hệ điều hành và có thể chọn lúc khởi động nhưng mỗi lúc chỉ làm việc được với một hệ diều hành).

Vmware đã được EMC – hãng chuyên về lĩnh vực thiết bị lưu trữ mua lại vào tháng 12 năm 2003. EMC đã mở rộng tầm hoạt động lĩnh vực ảo hóa từ máy tính để bàn đến máy chủ và hiện hãng vẫn giữ vai trò thống lĩnh thị trường ảo hóa, tuy nhiên Vmware không giữ vị trí “độc tôn” mà phải cạnh tranh với rất nhiề sản phẩm ảo hóa các hãng khách như Virtualization Engine của IBM, Hyper V – Microsoft, Virtuozzo của SWSoft và vitual iron của iron software… và ảo hóa cũng không còn bó hệp trong một lĩnh vực mà đã mở rộng cho toàn bộ hạ tầng công nghệ thông tin, từ phần cứng như chip xử lý cho đến hệ thống máy chủ và cả hệ thống mạng.

Hiện nay, Vmware là hãng đẫn đầu thị trường ảo hóa nhưng không phải là hãng tiên phong, vai trò thuộc về IBM với hệ thống ảo hóa VM/370 nổi tiếng được công bố vào năm 1972 và “ảo hóa” vẫn đang hiện diện trong các hệ thống máy chủ của IBM.

Giữa năm 1960, IBM’s Cambridge Scientific Center đã tiến hành phát triển sản phẩm CP-40, sản phẩm đầu tiên của dòng CP/CMS. Nó được chính thức đưa vào sản xuất vào tháng 1 năm 1967. Ngay từ khi thiết kế CP-40 đã đặt mục đích phải sử dụng ảo hóa đầy đủ. Để làm được vấn đề này nó yêu cầu phần cứng và đoạn mã của S/360-40 phải kết hợp hoàn chỉnh với nhau, nó phải cung cấp cách truy cập địa chỉ vung nhớ, tập lệnh CPU và các tính năng ảo hóa.

Năm 1970 IBM công bố sản phẩm System 370. Nhưng điều khiến người dùng thất vọng nhất về sản phẩm này do nó không có tính năng Virtial memory.

Vào tháng 8 năm 1999, Vmware giới thiệu sản phẩm ảo hóa đầu tiên hoạt dộng trên nên tảng x86. Vmware Virtual Platform…

Trước đây chúng ta phải mất tiền mua bản quyền sử dụng của Vmware's Workstion. Nhưng năm 2005 Vmware đã quyết định cung cấp sản phẩm ảo hóa chất lượng cao cho người dùng miễn phí. Tuy nhiên chức năng tạo máy chủ ảo và các tính năng phụ khác nhằm mục đích tăng hiệu suất sử dụng máy ảo đã bị lược bỏ.

Năm 2006 đây là năm ảo hóa có một bước tiến mới trong quá trình phát triển, đó là sự ra dời của Application Virtualization và Application Streaming.

Năm 2008, Vmware giới thiệu phiên bản Vmware workstation 6.5 beta, sản phẩm đầu tiên cho phép các chương trình cuả windows và linux được sử dụng Direct X9 để tăng tốc xử lý hình ảnh trong máy ảo Windows XP[12].

### 1.1.2 Phân loại ảo hóa

#### 1.1.2.1 Ảo hóa mạng

Ảo hóa hệ thống mạng là một tiến trình hợp nhất tài nguyên, thiết bị mạng cả phần cứng lẫn phần mềm thành một hệ thống mạng ảo. Sau đó, các tài nguyên này sẽ được phân chia thành các channel và gắn với một máy chủ hoặc một thiết bị nào đó.

Có nhiều phương pháp để thực hiện việc ảo hóa hệ thống mạng. Các phương pháp này tùy thuộc vào các thiết bị hỗ trợ, tức là các nhà sản xuất thiết bị đó, ngoài ra còn phụ thuộc vào hạ tầng mạng sẵn có, cũng như nhà cung cấp dịch vụ mạng (ISP).Sau đây là một vài mô hình ảo hóa hệ thống mạng:

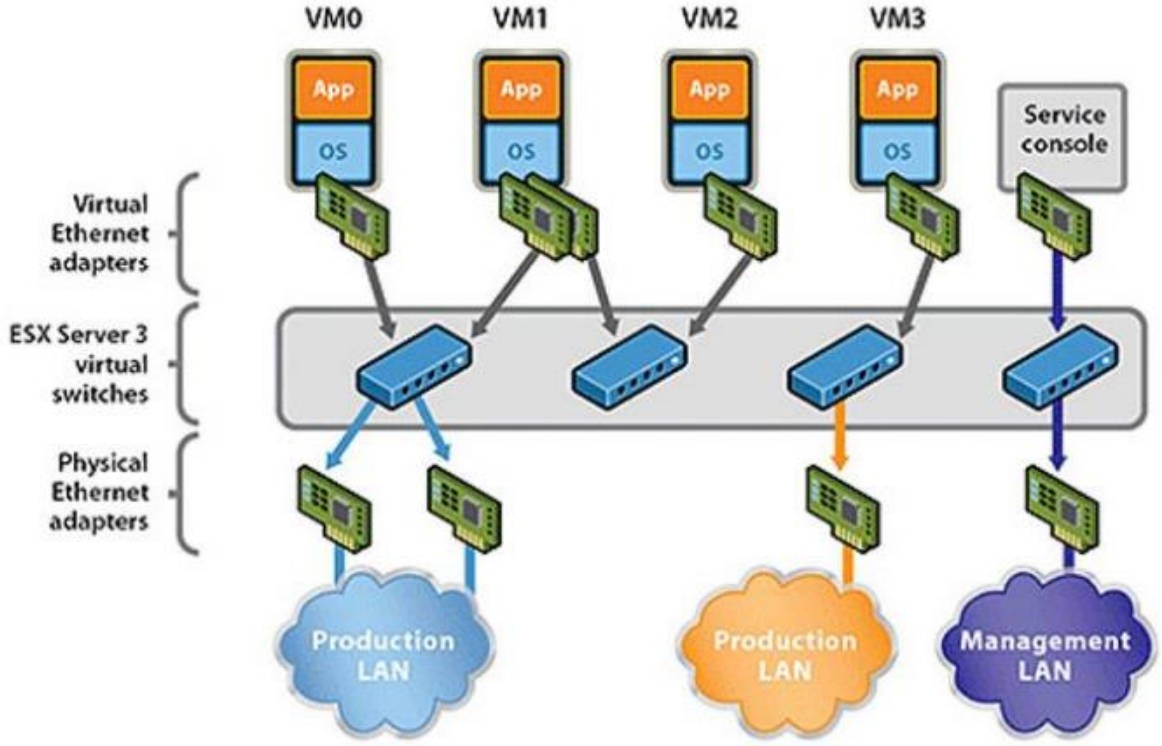
- Ảo hóa lớp mạng (Virtualized overlay network): Trong mô hình này, nhiều hệ thống mạng ảo sẽ cùng tồn tại trên một lớp nền tài nguyên dùng chung. Các tài nguyên đó bao gồm các thiết bị mạng như router, switch, các dây truyền dẫn, NIC (network interface card). Việc thiết lập nhiều hệ thống mạng ảo này sẽ cho phép sự trao đổi thông suốt giữa các hệ thống mạng khác nhau, sử dụng các giao thức và phương tiện

truyền tải khác nhau, ví dụ như mạng Internet, hệ thống PSTN, hệ thống Voip.

- Mô hình ảo hóa của Cisco: đó là phân mô hình ảo hóa ra làm 3 khu vực, với các chức năng chuyên biệt. Mỗi khu vực sẽ có các liên kết với các khu vực khác để cung cấp các giải pháp đến tay người dùng 1 cách thông suốt.

Các thành phần mạng trong cơ sở hạ tầng mạng như Switch, Card mạng, được ảo hoá một cách linh động. Switch ảo cho phép các máy ảo trên cùng một máy chủ có thể giao tiếp với nhau bằng cách sử dụng các giao thức tương tự như trên thiết bị chuyển mạch vật lý mà không cần phần cứng bổ sung. Chúng cũng hỗ trợ VLAN tương thích với việc triển khai VLAN theo tiêu chuẩn từ nhà cung cấp khác, chẳng hạn như Cisco.

Một máy ảo có thể có nhiều card mạng ảo, việc tạo các card mạng ảo này rất đơn giản và không giới hạn số card mạng tạo ra.Ta có thể nối các máy ảo này lại với nhau bằng một Switch ảo. Điều đặc biệt quan trọng, tốc độ truyền giữa các máy ảo này với nhau thông qua các switch ảo được truyền với tốt độ rất cao theo chuẩn GIGABITE(1GB), đẫn đến việc đồng bộ giữa các máy ảo với nhau diễn ra rất nhanh.




**Hình 1-1 Ảo hóa network**

### 1.1.2.2 Ảo hóa lưu trữ

Ảo hóa hệ thống lưu trữ về cơ bản là sự mô phỏng, giả lập việc lưu trữ từ các thiết bị lưu trữ vật lý. Các thiết bị này có thể là băng từ, ổ cứng hay kết hợp cả 2 loại. Việc làm này mang lại các ích lợi như việc tăng tốc khả năng truy xuất dữ liệu, do việc phân chia các tác vụ đọc, viết trong mạng lưu trữ. Ngoài ra, việc môt phỏng các thiết bị lưu trữ vật lý cho phép tiết kiệm thời gian hơn thay vì phải định vị xem máy chủ nào hoạt động trên ổ cứng nào để truy xuất.

Hiện nay các nhà lưu trữ đã cung cấp giải pháp lưu trữ hiệu suất cao cho khách hàng của họ trong một thời gian kha khá. Trong hình thức cơ bản nhất của nó, lưu trữ ảo hóa tồn tại trong việc ta lắp ráp nhiều ổ đĩa vật lý thành một thực thể duy nhất để các máy chủ lưu trữ và chạy hệ điều hành chẳng hạn như triển khai RAID. Điều này có thể được coi là ảo bởi vì tất cả các ổ đĩa được sử dụng và tương tác như một ổ đĩa logic duy nhất, mặc dù bao gồm hai hoặc nhiều ổ đĩa trong.

Một công nghệ ảo hoá lưu trữ khá nổi bật là SAN (Storeage Area Network – lưu trữ qua mạng). SAN là một mạng được thiết kế cho việc thêm các thiết bị lưu trữ cho máy chủ một cách dễ dàng như: Disk Aray Controllers, hay Tape Libraries.

Với những ưu điểm nổi trội SAN đã trở thành một giải pháp rất tốt cho việc lưu trữ thông tin của doanh nghiệp hay tổ chức. SAN cho phép kết nối từ xa tới các thiết bị lưu trữ trên mạng như: Disks và Tape drivers. Các thiết bị lưu trữ trên mạng, hay các ứng dụng chạy trên đó được thể hiện trên máy chủ như một thiết bị của máy chủ (as locally attached divices).

Có hai đặc điểm cơ bản trong các thành phần của SAN:

- Mạng (network) có tác dụng truyền thông tin giữa thiết bị lưu trữ và hệ thống máy tính. Một SAN bao gồm một cấu trúc truyền tin, nó cung cấp kết nối vật lý, và quản lý các lớp, tổ chức các kết nối, các thiết bị lưu trữ, và hệ thống máy tính sao cho dữ liệu truyền trên đó với tốc độ cao và tính bảo mật;
- Một hệ thống lưu trữ bao gồm các thiết bị lưu trữ, hệ thống máy tính, hay các ứng dụng chạy trên nó, và một phần rất quan trọng là các phần mềm điều khiển, quá trình truyền thông tin qua mạng.

#### 1.1.2.3 Ảo hóa máy chủ

Một máy chủ riêng ảo tiếng anh Virtual Private Server, máy chủ ảo hoá là một phương pháp phân vùng một máy chủ vật lý thành nhiều máy chủ ảo, mỗi máy chủ có khả năng của riêng của mình chạy trên máy tính dành riêng. Mỗi máy chủ ảo riêng của nó có thể chạy hệ điều hành khác nhau, và mỗi máy chủ ảo có thể được khởi động lại độc lập.

Lợi thế của ảo hóa máy chủ:

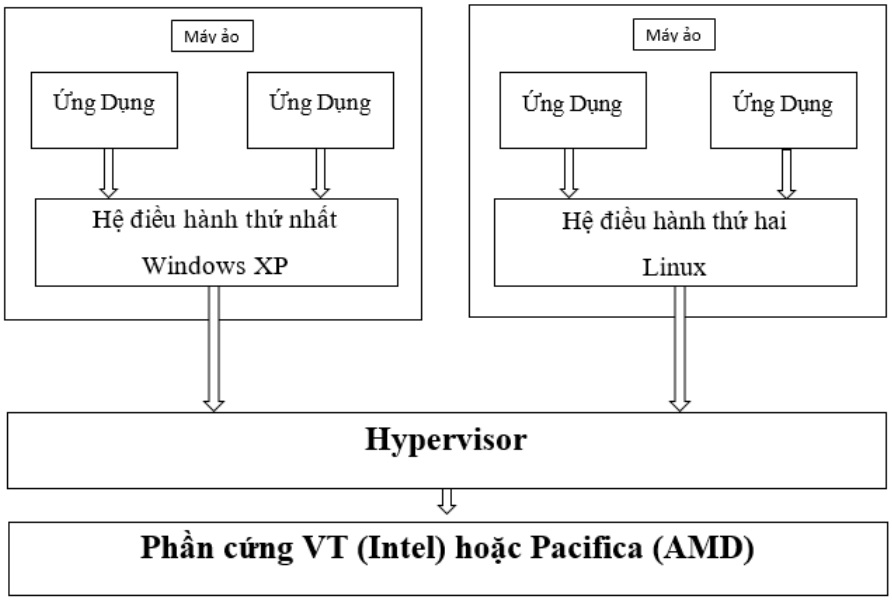
- Tiết kiệm được chi phí đầu tư máy chủ ban đầu;
- Hoạt động hoàn toàn như một máy chủ riêng;
- Có thể dùng máy chủ ảo hoá cài đặt các ứng dụng khác tùy theo nhu cầu của doanh nghiệp;
- Bảo trì sửa chữa nâng cấp nhanh chóng và dễ dàng;
- Dễ dàng nâng cấp tài nguyên RAM, HDD, băng thông khi cần thiết;
- Có thể cài lại hệ điều hành từ 5-10 phút;
- Không lãng phí tài nguyên.

Có hai môi trường máy chủ ảo hoá: ảo hoá toàn phần (Full virtualization) và ảo hoá một nửa (Paravirtualization).

#### 1.1.2.3.1 Full-virtualization

Phần cứng được mô phỏng để mở rộng chạy những hệ điều hành khách trên nền tảng ảo hóa.Điều này có nghĩa rằng các thiết bị phần cứng khác nhau đều được mô phỏng.Thông thường, có nhiều nền tảng ảo hóa cố gắng chạy nhiều sự ủy nhiệm trên CPU chính (chạy nhanh hơn nhiều so với CPU mô phỏng) nhằm nắm bắt và xử lý các sự ủy nhiệm một cách thích hợp.

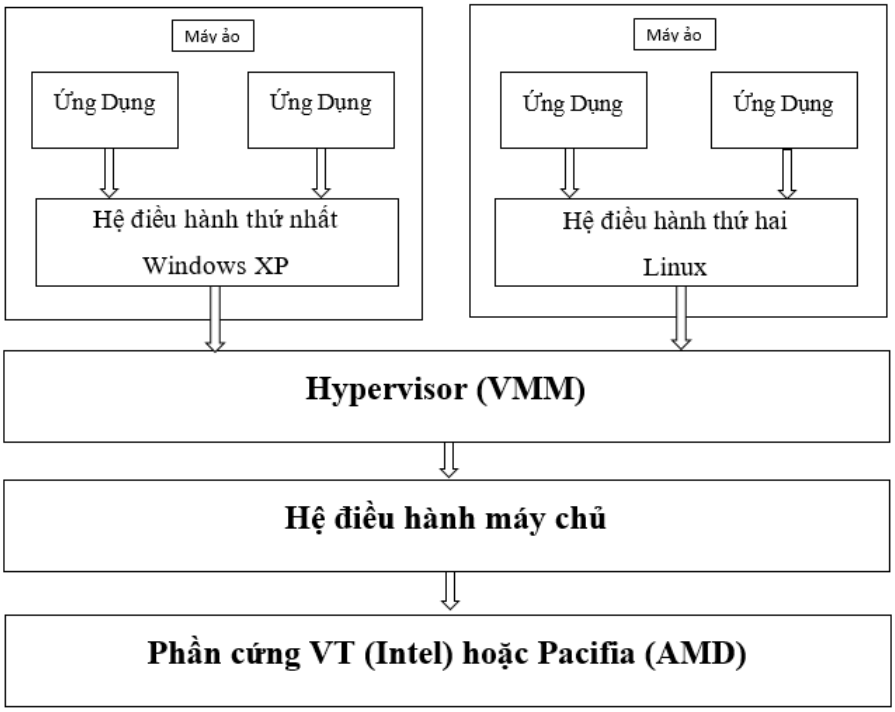
Một số nền tảng ảo hóa hỗ trợ hoặc yêu cầu CPU mở rộng để hỗ trợ ảo hóa. Trên 1 số những dòng chíp mới như x86 và x86_64 CPUs được cung cấp thông qua VT-X (Intel) và AMD-V (AMD).Chúng được gọi là Phần Cứng Hỗ Trợ Ảo Hóa (hardware-assisted full-virtualization).




**Hình 1-2 Full-virtualization**

#### 1.1.2.3.2 Paravirtualization

là một phương pháp ảo hóa máy chủ khác. Với phương pháp ảo hóa này, thay vì mô phỏng một môi trường phần cứng hoàn chỉnh, phần mềm ảo hóa này là một lớp mỏng (Hypervisor) dồn các truy cập các hệ điều hành máy chủ vào tài nguyên máy vật lý cơ sở.

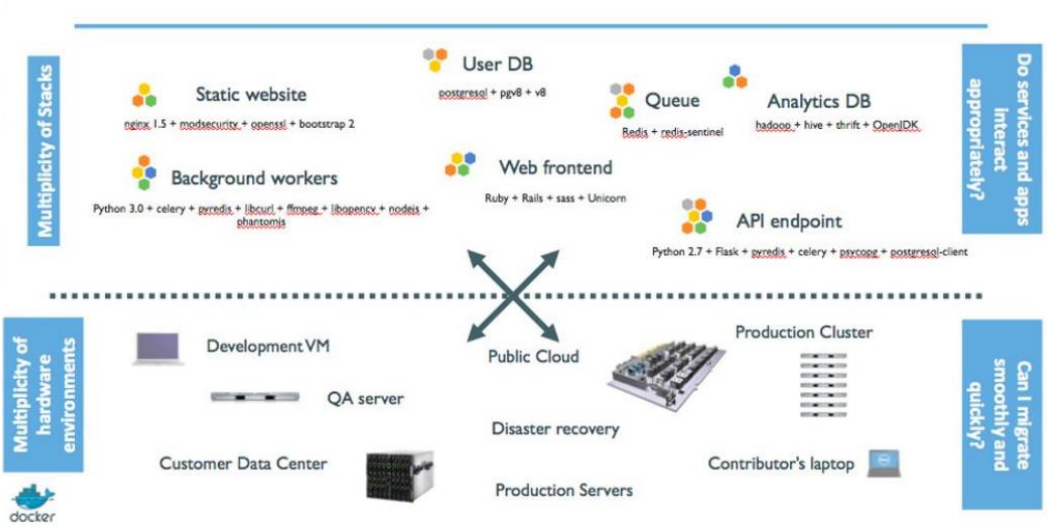



**Hình 1-3 Paravirtualization**

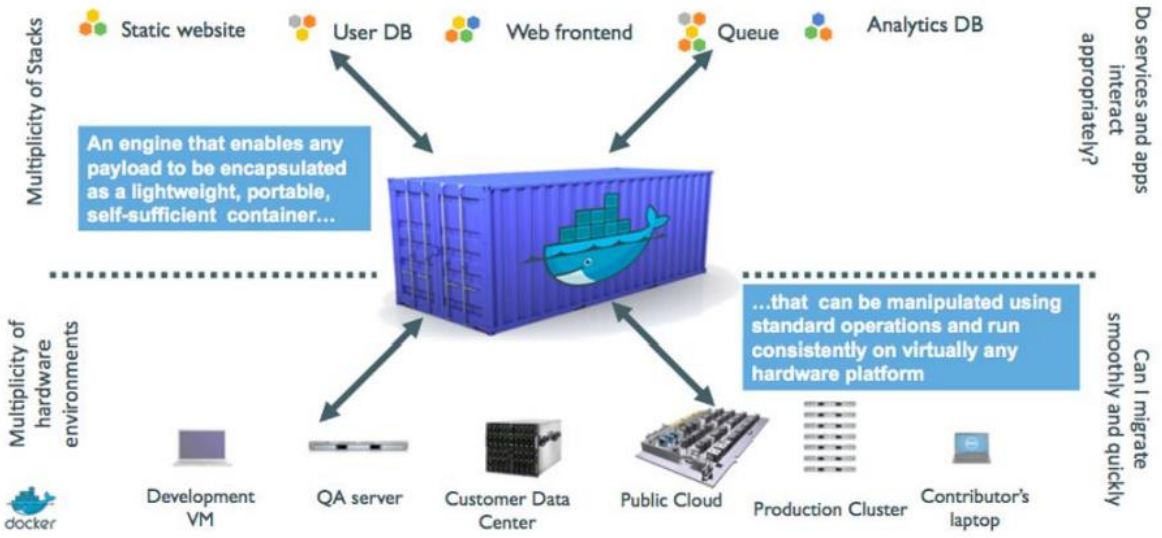
### 1.1.2.4 Ảo hóa ứng dụng

Ảo hóa ứng dụng là một dạng công nghệ ảo hóa khác cho phép chúng ta tách rời mối liên kết giữa ứng dụng và hệ điều hành và cho phép phân phối lại ứng dụng phù hợp với nhu cầu user. Môt ứng dụng được ảo hóa sẽ không được cài đặt lên máy tính một cách thông thường, mặc dù ở góc độ người sử dụng, ứng dụng vẫn hoạt động một cách bình thường. Việc quản lý việc cập nhật phần mềm trở nên dễ dàng hơn, giải quyết sự đụng độ giữa các ứng dụng và việc thử nghiệm sự tương thích của chúng cũng trở nên dễ dàng hơn. Hiện nay đã có khá nhiều chương trình ảo hóa ứng dụng như VirtualBox, Vagrant, Docker, Citrix XenApp, Microsoft Application Virtualization, Vmware ThinApp…

Ảo hóa ứng dụng là giải pháp tiến đến công nghệ "điện toán đám mây" cho phép ta sử dụng phần mềm của công ty mà không cần phải cài phần mềm này vào bất cứ máy tính con nào[11].




**Hình 1-4 Hypervisor**




**Hình 1-5 Docker**

Giải pháp Ảo Hóa Ứng Dụng cho ta những lợi ích nổi trội sau:

- Tất cả các máy tính đều có thể sử dụng phần mềm ảo như đang cài trên máy tính của mình mà không phải lo về cấu hình (ví dụ chạy Photoshop trên máy P4 chỉ có 512 MB RAM). Tốc độ phần mềm luôn ổn định và ko phụ thuộc vào cấu hình từng máy;
- Các máy tính con luôn ở trong tình trạng sạch và chạy nhanh hơn. Loại bỏ hòan tòan việc phải sửa lỗi phần mềm do virus, spyware hoặc do người dùng sơ ý;
- Cho phép sử dụng phần mềm mà không phải quan tâm đến hệ điều hành ta đang sử dụng (ví dụ: ta có thể dùng Microsoft Office 2007 ngay trong Linux, Windows 98 hoặc MAC-OS);
- Ta có thể phân phối phần mềm 1 cách linh động đến 1 số cá nhân hoặc nhóm có nhu cầu sử dụng thay vì cài vào tất cả mọi máy như cách phổ thông. Việc phân phối hoặc gỡ bỏ phần mềm ra các máy tính có thể diễn ra chỉ trong vòng chỉ vài giây thay vì hàng tuần nếu như công ty có hàng chục máy tính;
- Thông tin luôn luôn được lưu trữ an toàn ở server trung tâm thay vì có thể phân tán ra từng máy con. Cho dù ta ở bất cứ nơi nào (tại 1 máy tính

khác, tại nhà hay thậm chí ở internet cafe), việc truy nhập và sử dụng phần mềm của doanh nghiệp trở nên dễ dàng qua 1 hệ thống bảo mật hiện đại nhất.

Ảo hóa ứng dụng là giải pháp cho phép sử dụng và quản lý phần mềm doanh nghiệp 1 cách hiệu quả có hệ thống. Tiết kiệm tối đa chi phí bảo trì, hỗ trợ kỹ thuật và quản lý từng máy tính.

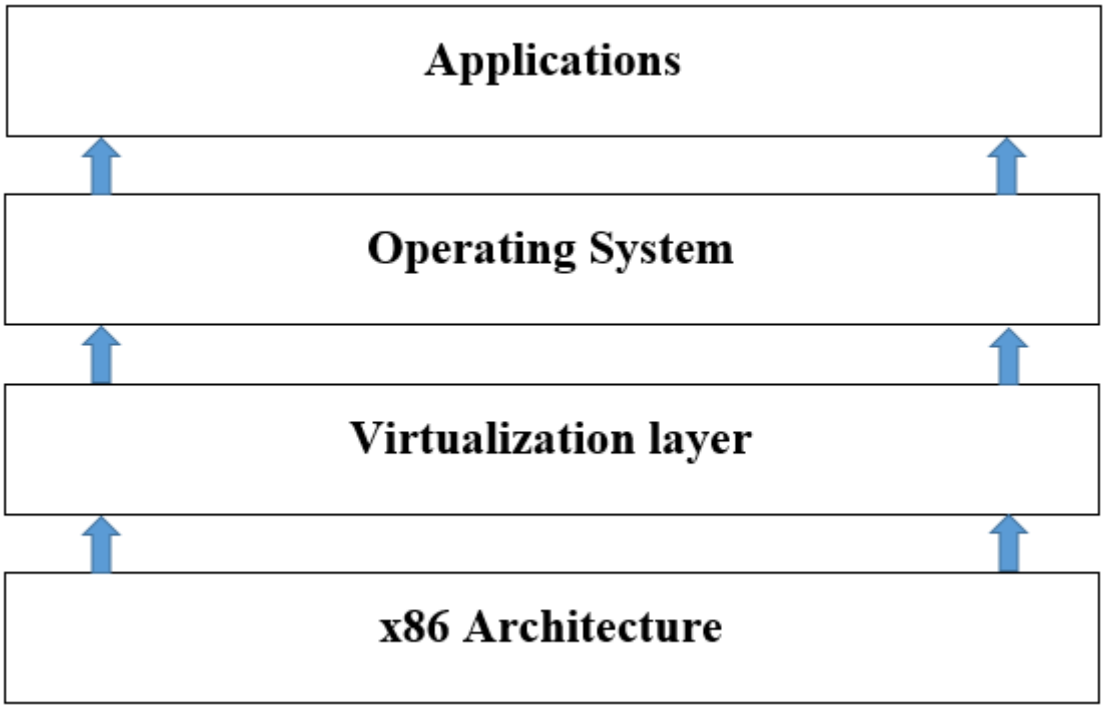
Trong ảo hóa ứng dụng thì công nghệ ảo hóa Docker đang được đánh giá là tương lai của công nghệ ảo hoá (future of virtualization), Công nghệ này là sản phẩn của một dự án phần mềm tự do nguồn mở phát hành theo giấy phép Apache. Khác biệt lớn của Docker và các công nghệ ảo hóa khác đó là tiết kiệm đáng kể nguồn lực sử dụng. Với docker có thể chạy 20 container (tương tự như một hệ điều hành nhỏ) trên cùng một máy host (host machine), mà nếu sử dụng công nghệ ảo hóa khác như Vagrant sẽ cần một máy chủ với cấu hình rất lớn. Docker làm được điều này là bởi vì khác với Virtual Machine ở chỗ thay vì tách biệt giữa hai môi trường guest và host, thì các container của Docker chia sẻ các resource với host machine[3].

### 1.1.3 Các công nghệ giúp ảo hóa hệ thống

#### 1.1.3.1 Công nghệ máy ảo

Máy ảo là một máy tính được cài trên một hệ điều hành khác hay một máy tính khác. Một máy ảo cũng bao gồm phần cứng, các ứng dụng phần mềm và hệ điều hành. Điều khác biệt ở đây là lớp phần cứng của máy ảo không phải là các thiết bị thường mà chỉ là một môi trường hay phân vùng mà ở đó nó được cấp phát một số tài nguyên như là chu kì CPU, bộ nhớ, ỗ đĩa….Công nghệ máy ảo cho phép cài và chạy nhiều máy ảo trên một máy tính vật lý. Mỗi máy ảo có một hệ điều hành máy khách riêng lẻ và được phân bố tài nguyên, ổ cứng, card mạng và các tài nguyên phần cứng khác một cách hợp lý. Việc phân bố tài nguyên này phụ thuộc vào nhu cầu của từng máy ảo ứng dụng và cũng tùy thuộc vào phương pháp ảo hóa được dùng. Đặc biệt khi

máy ảo cần truy xuất tài nguyên phần cứng thì nó hoạt động giống như một máy thật hoàn chỉnh. Vì chỉ là một tập tin được phân vùng trên ổ đĩa nên việc di chuyển các máy ảo từ máy chủ này sang máy chủ khác là rất dễ dàng và không cần quan tâm đến vấn đề tương thích phần cứng hay ảnh hưởng tới máy chủ.




**Hình 1-6 Mô hình các lớp tương tác trong hệ thống VMs**

Trong kiến trúc của một bộ xử lý ảo hóa được chia thành 4 lớp . Lớp 0 là lớp có quyền cao nhất có thể truy cập và can thiệp sâu nhất đến tài nguyên phần cứng.

Lớp 0 thường là các hệ điều hành chủ được cài trên chính máy chủ. Lớp 1 là lớp ảo hóa Hypervisor. Lớp này dùng để quản lý và phân phối tài nguyên đến các máy ảo. Lớp 2 là các hệ điều hành khách chạy trên các máy ảo. Để truy cập tài nguyên phần cứng nó phải liên lạc với lớp ảo hóa và phải qua hệ điều hành máy chủ. Lớp có quyền can thiệp thấp nhất đến tài nguyên là lớp 3 là các ứng dụng hoạt động trên các máy ảo.

Trong các hệ thống máy tính lớn dùng để xử lý các ứng dụng thương mại và khoa học (mainframe), hệ điều hành chạy trên phần cứng máy thực ở chế độ ưu tiên vì chỉ có hệ điều hành chủ mới được phép sửa đổi và can thiệp vào phần cứng bên dưới nó. Còn máy ảo làm việc ở chế độ giới hạn vì phần cứng mà nó nhìn thấy chỉ là các thiết bị ảo. Khi máy ảo yêu cầu các lệnh hoặc tiến trình thông thường thì hệ điều hành chủ sẽ chuyển tiếp chúng đến bô xử lý để thực thi trực tiếp, còn đối với các lệnh hoặc các tiến trình đặc biệt nhạy cảm can thiệp sâu đến phần cứng bên dưới sẽ bị chặn lại vì có thể làm ảnh hưởng tới hệ thống và máy ảo còn lại. Hệ điều hành chủ sẽ thực thi lệnh với bộ xử lý trên máy thực rồi sau đó mô phỏng kết quả rồi trả về cho máy ảo. Đây là cơ chế nhằm cách ly máy ảo với máy thực để đảm bảo an toàn hệ thống.

#### 1.1.3.2 Hệ thống cân bằng tải

Trong xu hướng công nghệ, máy chủ là trái tim của của mạng máy tính, nếu máy chủ mạng hỏng, hoạt động của hệ thống sẽ bị ngưng trệ. Do vậy, vấn đề đặt ra là cần có một giải pháp để đảm bảo cho hệ thống vẫn hoạt động tốt ngay cả khi có sự cố xảy ra đối với máy chủ mạng. Giải pháp hiệu quả được đưa ra là sử dụng một nhóm server cùng thực hiện một chức năng dưới sự điều khiển của một công cụ phân phối tải - Giải pháp cân bằng tải (Load Balancing). Có rất nhiều hãng đưa ra giải pháp cân bằng tải như Cisco, Coyote Point, Sun Microsystems... với rất nhiều tính năng phong phú.

Load Balancing là một công nghệ có khả năng chia tải và nâng cao khả năng chịu lỗi của hệ thống. Load Balancing không chỉ làm nhiệm vụ phân phối tải cho các server mà còn cung cấp cơ chế đảm bảo hệ thống server luôn khả dụng trước các client và nhu cầu truy cập từ internet.

Hiện nay có 2 loại cân bằng tải được áp dụng:

1.Cân bằng tải sử dụng phần cứng: sử dụng các mođun cắm thêm trên các thiết bị chuyên dụng như Bộ định tuyến (Router) hay bộ chuyển mạch

(Switch) để chia tải theo luồng, thường hoạt động từ layer 4 trở xuống. Vì sử dụng thiết bị chuyên dụng nên có hiệu năng cao, tính ổn định cao, khả năng mở rộng tốt hơn nhưng khó phát triển được tính năng bảo mật phức tạp. Thường sử dụng các thiết bị của Cisco, F5, Citrix,…

2.Cân bằng tải sử dụng phần mềm: Sử dụng phần mềm cài trên server để kết hợp nhiều server một cách chặt chẽ tạo thành một server ảo (virtual server). Cách này có ưu điểm là có thể chia sẻ được nhiều tài nguyên trong hệ thống, theo dõi được trạng thái của các máy chủ trong nhóm để chia tải hợp lý. Tuy nhiên, do sử dụng phần mềm trên server, tính phức tạp cao nên khả năng mở rộng của giải pháp này bị hạn chế, phức tạp khi triển khai cũng như khắc phục khi xảy ra sự cố, có rào cản về tính tương thích, khó có được những tính năng tăng tốc và bảo mật cho ứng dụng. Thường sử dụng các giải pháp Proxy, DNS load balancing, Round Robin NDS,…

#### 1.1.3.3 Hệ thống cân bằng tải mạng

Công nghệ cân bằng tải mạng (Load Balancing) là một công nghệ có khả năng chia tải và nâng cao khả năng chịu lỗi của hệ thống. Được dùng cho các ứng dụng Stateless applications (các ứng dụng hoạt động mang tính nhất thời) như Web, File Tranfer Protocol (FTP), Virtual Private Network (VPN)… Trong hệ thống NLB sẽ bao gồm các cụm server được cấu hình tương tự nhau (có thể được đặt rải rác ở nhiều nơi) cùng hoạt động để phân phối khối lượng công việc giữa các máy chủ trong hệ thống, giúp hệ thống giảm bớt gánh nặng khi phân bố tải.

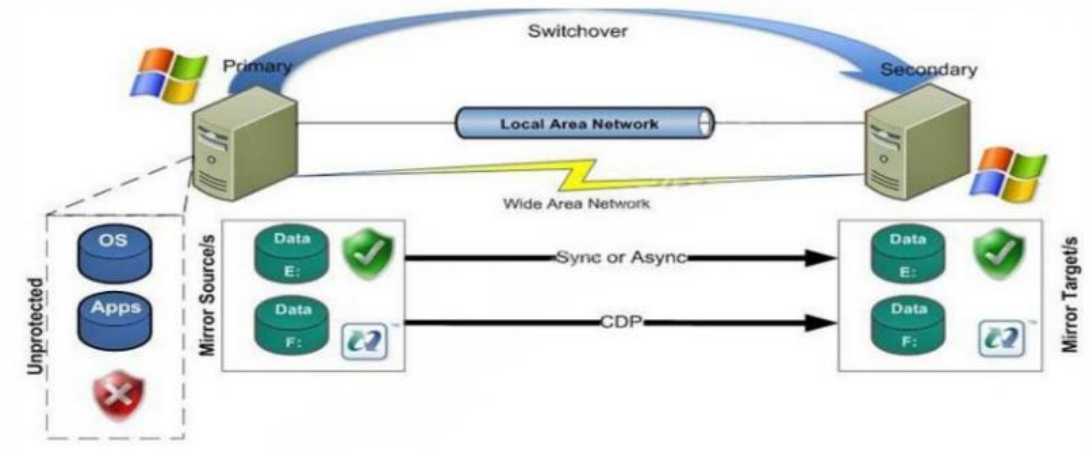
Nhược điểm của NLB là mỗi cụm server phải dùng riêng một nơi lưu trữ cục bộ (Local Storage) cho nên cần phải có quá trình đồng bộ hóa dữ liệu ở mỗi nơi lưu trữ, số lượng cụm server càng nhiều thì thời gian cho việc đồng bộ hóa càng lâu, chính vì điều này nên ta không nên triển khai các ứng dụng Stateful applications (các ứng dụng hoạt động thường xuyên trong thời gian dài) như các database server: Microsoft SQL Server, Microsoft Exchange

Server, File and Print Server… trên kỹ thuật NLB này nhằm đảm bảo tính chính xác của dữ liệu.

### 1.1.3.4 Hệ thống cân bằng tải clustering

Đây là công nghệ được dùng rộng rãi cho các hệ thống cần độ sẵn sàng phục vụ cao, đây là giải pháp được đặc biệt quan tâm do tính kinh tế, đa dạng và khả năng dịch vụ cao. Công nghệ này có thể sử dụng phần cứng chuyên dụng để cung cấp một môi trường với độ tin cậy cao đảm bảo cho các dịch vụ có thể hoạt động trơn tru mà không bị dừng bởi một vài lỗi nhỏ; hoặc cũng có thể được thiết kế để chạy trên các phần cứng thông dụng mà vẫn đạt được các yêu cầu:

- Tăng cường khả năng mở rộng;
- Nâng cao hiệu suất;
- Tính sẵn sàng cao và khắc phục sự cố.




**Hình 1-7 Mô hình cân bằng tải Clustering**

Với hệ thống sử dụng công nghệ clustering, trong quá trình khởi tạo cấu hình của hệ thống sẽ bao gồm một hệ thống với các nút chính chủ động (active primary node) và một hệ thống với các nút phụ bị động sao lưu (passive backup node). Trong hệ thống này các nút chủ động và bị động cần có cùng cấu hình và được sử dụng công nghệ lưu trữ SAN hoặc Raid (Raid 1)

Nút đang hoạt động (active node) sẽ đáp lại các yêu cầu về dịch vụ thông qua một địa chỉ IP ảo (Virtual IP hay VIP). Địa chỉ VIP là một địa chỉ IP và nó chỉ khác so với địa chỉ IP thông thường của một nút đang hoạt động.

Hệ thống bị động (gồm các nút không hoạt động) sẽ không trực tiếp chạy dịch vụ, thay vào đó nó quản lí các dịch vụ của nút chủ động đang hoạt động, và đảm bảo chắc chắn là nút đang hoạt động vẫn phải đang còn hoạt động. Nếu nút không hoạt động phát hiện ra 1 vấn đề nào đó với hoặc là nút hoạt động hoặc dịch vụ đang chạy trên nó, thì một thông báo lỗi sẽ được khởi tạo.Khi có lỗi, hệ thống clustering sẽ thực hiện các bước sau:

- Bước 1: Nút đang hoạt động sẽ trực tiếp ngắt hết các dịch vụ đang chạy và các kết nối;
- Bước 2: Nút không hoạt động sẽ khởi động các dịch vụ tương đương với các dịch vụ của máy chủ động;
- Bước 3: Nút đang hoạt động sẽ ngắt không sử dụng địa chỉ VIP;
- Bước 4: Nút không hoạt động bây giờ lại chuyển thành nút đang hoạt động, và ở chế độ sử dụng địa chỉ VIP;
- Bước 5: Nếu nó vẫn đang hoạt động và đang duy trì kết nối mạng, nút trước kia là chủ động thì bây giờ trở thành nút bị động, bắt đầu giám sát các dịch vụ của nút chủ động.

#### 1.1.3.5 Công nghệ RAID

Vào cuối những năm 1980 và đầu 1990, các nhà cung cấp dịch vụ công nghệ thông tin đã phải đối mặt với việc tăng nhanh một khối lượng khổng lồ các dữ liệu cần được lưu trữ. Các công nghệ lưu trữ đang trở nên rất đắt để đặt một số lượng lớn ổ cứng có khả năng cao trên các máy chủ. RAID ra đời đã giải quyết vấn đề trên.

RAID được định nghĩa như thế nào? Trước hết RAID là viết tắt của Redundant Array of Inexpensive Disks (Hệ thống đĩa dự phòng). Đây là hệ

thống hoạt động bằng cách kết nối một dãy các ổ cứng có chi phí thấp lại với nhau để hình thành một thiết bị nhớ đơn có dung lượng lớn hỗ trợ hiệu quả cao và đáng tin cậy hơn so với các giải pháp trước đây. RAID được sử dụng và triển khai thành phương pháp lưu trữ trong doanh nghiệp và các máy chủ, nhưng trong 5 năm sau đó RAID đã trở nên phổ biến đối với mọi người dùng.

Lợi thế của RAID:

Có 3 lý do chính để áp dụng RAID:

Dự phòng;

Hiệu quả cao;

Giá thành thấp.

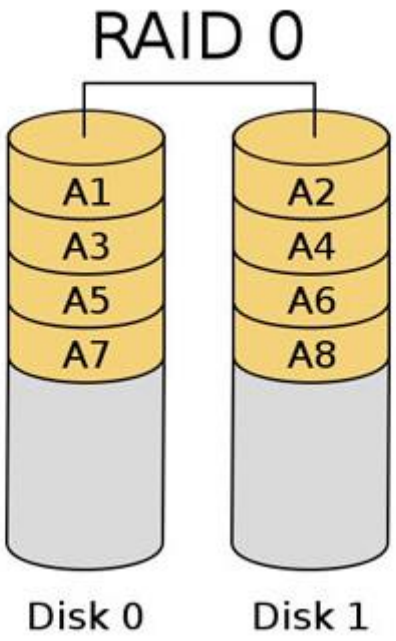
Sự dự phòng là nhân tố quan trọng nhất trong quá trình phát triển RAID cho môi trường máy chủ. Dự phòng cho phép sao lưu dữ liệu bộ nhớ khi gặp sự cố. Nếu một ổ cứng trong dãy bị trục trặc thì nó có thể hoán đổi sang ổ cứng khác mà không cần tắt cả hệ thống hoặc có thể sử dụng ổ cứng dự phòng. Phương pháp dự phòng phụ thuộc vào phiên bản RAID được sử dụng.

Khi áp dụng các phiên bản RAID mạnh bạn có thể thấy rõ hiệu quả tăng cao của nó. Hiệu quả cũng tùy thuộc vào số lượng ổ cứng được liên kết với nhau và các mạch điều khiển.

Tất cả các nhà quản lý những tập đoàn CNTT đều muốn giảm giá thành. Khi chuẩn RAID ra đời, giá thành là một vấn đề chủ chốt. Mục tiêu của các dãy RAID là cung cấp bộ nhớ tốt hơn cho hệ thống so với việc sử dụng riêng biệt các ổ đĩa có dung lượng lớn.

Có 3 cấp độ RAID sử dụng cho hệ thống máy tính để bàn là RAID 0, RAID 1 và RAID 5. Trong nhiều trường hợp thì chỉ hai trong ba cấp trên là có hiệu lực và một trong hai kỹ thuật được sử dụng không phải là một cấp độ của RAID.

**RAID 0**



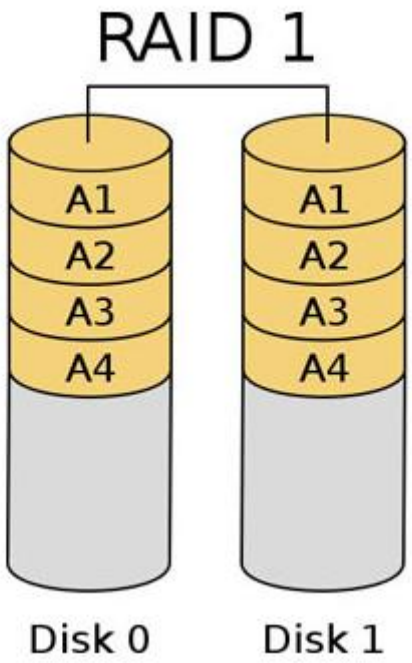

**Hình 1-8 RAID 0**

Đây là dạng RAID đang được người dùng ưa thích do khả năng nâng cao hiệu suất trao đổi dữ liệu của đĩa cứng. Đòi hỏi tối thiểu hai đĩa cứng, RAID 0 cho phép máy tính ghi dữ liệu lên chúng theo một phương thức đặc biệt được gọi là Striping. Ví dụ bạn có 8 đoạn dữ liệu được đánh số từ 1 đến 8, các đoạn đánh số lẻ (1,3,5,7) sẽ được ghi lên đĩa cứng đầu tiên và các đoạn đánh số chẵn (2,4,6,8) sẽ được ghi lên đĩa thứ hai. Để đơn giản hơn, bạn có thể hình dung mình có 100MB dữ liệu và thay vì dồn 100MB vào một đĩa cứng duy nhất, RAID 0 sẽ giúp dồn 50MB vào mỗi đĩa cứng riêng giúp giảm một nửa thời gian làm việc theo lý thuyết. Từ đó bạn có thể dễ dàng suy ra nếu có 4, 8 hay nhiều đĩa cứng hơn nữa thì tốc độ sẽ càng cao hơn. Tuy nghe có vẻ hấp dẫn nhưng trên thực tế, RAID 0 vẫn ẩn chứa nguy cơ mất dữ liệu. Nguyên nhân chính lại nằm ở cách ghi thông tin xé lẻ vì như vậy dữ liệu không nằm hoàn toàn ở một đĩa cứng nào và mỗi khi cần truy xuất thông tin (ví dụ một file nào đó), máy tính sẽ phải tổng hợp từ các đĩa cứng. Nếu một đĩa cứng gặp trục trặc thì thông tin (file) đó coi như không thể đọc được và

mất luôn. Thật may mắn là với công nghệ hiện đại, sản phẩm phần cứng khá bền nên những trường hợp mất dữ liệu như vậy xảy ra không nhiều.

Có thể thấy RAID 0 thực sự thích hợp cho những người dùng cần truy cập nhanh khối lượng dữ liệu lớn, ví dụ các game thủ hoặc những người chuyên làm đồ hoạ, video số.

**RAID 1**




**Hình 1-9 RAID 1**

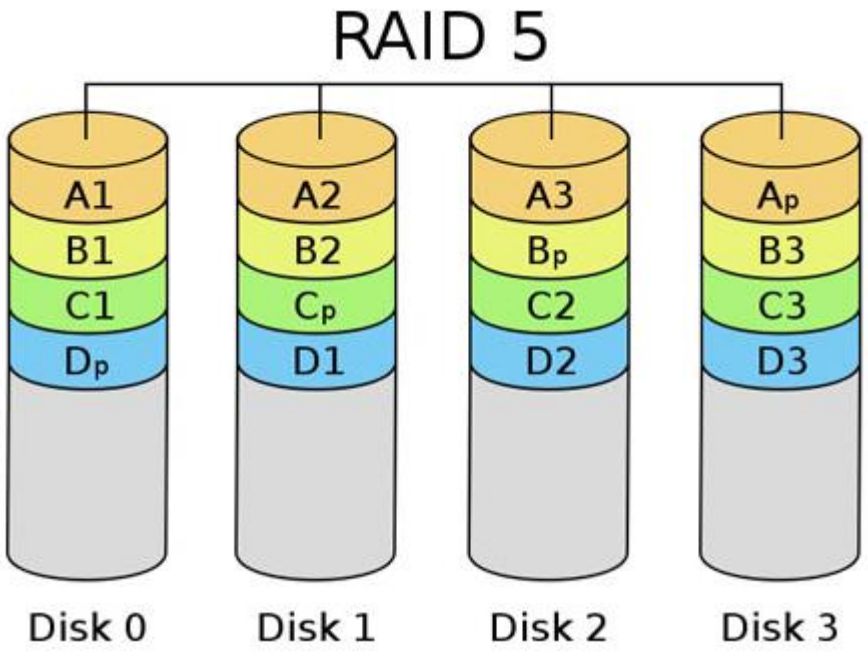
Đây là dạng RAID cơ bản nhất có khả năng đảm bảo an toàn dữ liệu. Cũng giống như RAID 0, RAID 1 đòi hỏi ít nhất hai đĩa cứng để làm việc. Dữ liệu được ghi vào 2 ổ giống hệt nhau (Mirroring). Trong trường hợp một ổ bị trục trặc, ổ còn lại sẽ tiếp tục hoạt động bình thường. Bạn có thể thay thế ổ đĩa bị hỏng mà không phải lo lắng đến vấn đề thông tin thất lạc. Đối với RAID 1, hiệu năng không phải là yếu tố hàng đầu nên chẳng có gì ngạc nhiên nếu nó không phải là lựa chọn số một cho những người say mê tốc độ. Tuy nhiên đối với những nhà quản trị mạng hoặc những ai phải quản lý nhiều thông tin quan trọng thì hệ thống RAID 1 là thứ không thể thiếu. Dung lượng cuối cùng của

hệ thống RAID 1 bằng dung lượng của ổ đơn (hai ổ 80GB chạy RAID 1 sẽ cho hệ thống nhìn thấy duy nhất một ổ RAID 80GB).

**RAID 0+1**

Người dùng luôn muốn có một hệ thống lưu trữ nhanh nhẹn như RAID 0, an toàn như RAID 1. Chính vì thế mà hệ thống RAID kết hợp 0+1 đã ra đời, tổng hợp ưu điểm của cả hai "đàn anh". Tuy nhiên chi phí cho một hệ thống kiểu này khá đắt, người dùng sẽ cần tối thiểu 4 đĩa cứng để chạy RAID 0+1. Dữ liệu sẽ được ghi đồng thời lên 4 đĩa cứng với 2 ổ dạng Striping tăng tốc và 2 ổ dạng Mirroring sao lưu. 4 ổ đĩa này phải giống hệt nhau và khi đưa vào hệ thống RAID 0+1, dung lượng cuối cùng sẽ bằng ½ tổng dung lượng 4 ổ, ví dụ bạn chạy 4 ổ 80GB thì lượng dữ liệu "thấy được" là (4*80)/2 = 160GB.

**RAID 5**




**Hình 1-10 RAID 5**

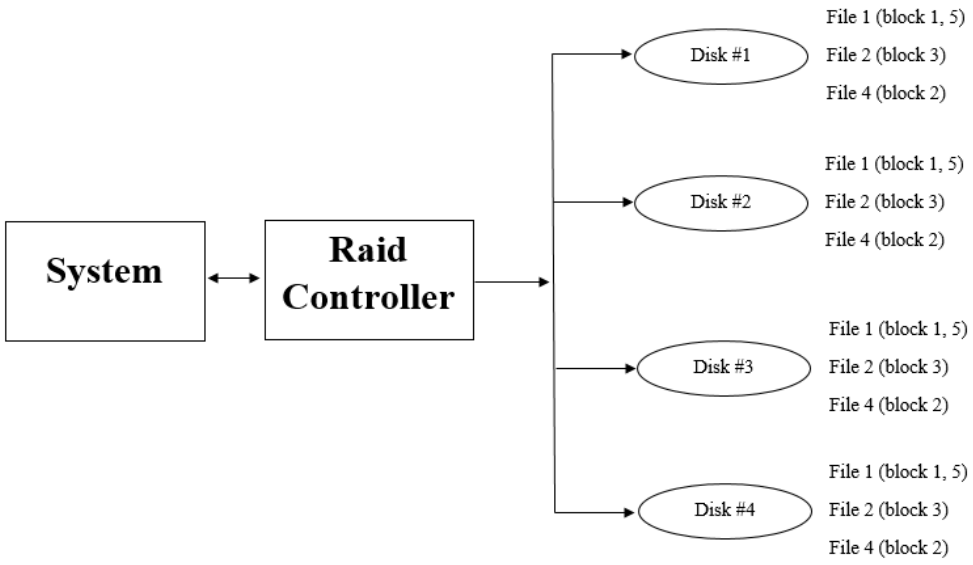
Đây có lẽ là dạng RAID mạnh mẽ nhất cho người dùng văn phòng và gia đình với 3 hoặc 5 đĩa cứng riêng biệt. Dữ liệu và bản sao lưu được chia lên tất cả các ổ cứng. Nguyên tắc này khá rối rắm. Chúng ta quay trở lại ví dụ về 8 đoạn dữ liệu (1-8) và giờ đây là 3 ổ đĩa cứng. Đoạn dữ liệu số 1 và số 2

sẽ được ghi vào ổ đĩa 1 và 2 riêng rẽ, đoạn sao lưu của chúng được ghi vào ổ cứng 3. Đoạn số 3 và 4 được ghi vào ổ 1 và 3 với đoạn sao lưu tương ứng ghi vào ổ đĩa 2. Đoạn số 5, 6 ghi vào ổ đĩa 2 và 3, còn đoạn sao lưu được ghi vào ổ đĩa 1 và sau đó trình tự này lặp lại, đoạn số 7,8 được ghi vào ổ 1, 2 và đoạn sao lưu ghi vào ổ 3 như ban đầu. Như vậy RAID 5 vừa đảm bảo tốc độ có cải thiện, vừa giữ được tính an toàn cao. Dung lượng đĩa cứng cuối cùng bằng tổng dung lượng đĩa sử dụng trừ đi một ổ. Tức là nếu bạn dùng 3 ổ 80GB thì dung lượng cuối cùng sẽ là 160GB.

**Striping**

Là một trong những chuẩn RAID có hiệu năng cao nhất, nó giúp ta tăng tốc độ truy cập lên tối đa bằng cách ghi song song dữ liệu lên các ổ đĩa này. Kỹ thuật này sẽ chia các tệp dự liệu ra và ghi đồng thời lên ổ đĩa cứng trong cùng một thời gian. Và khi đọc thì cũng đọc cùng lúc trên tất cả các ổ đĩa làm cho tốc độ đọc và hiệu suất cao.

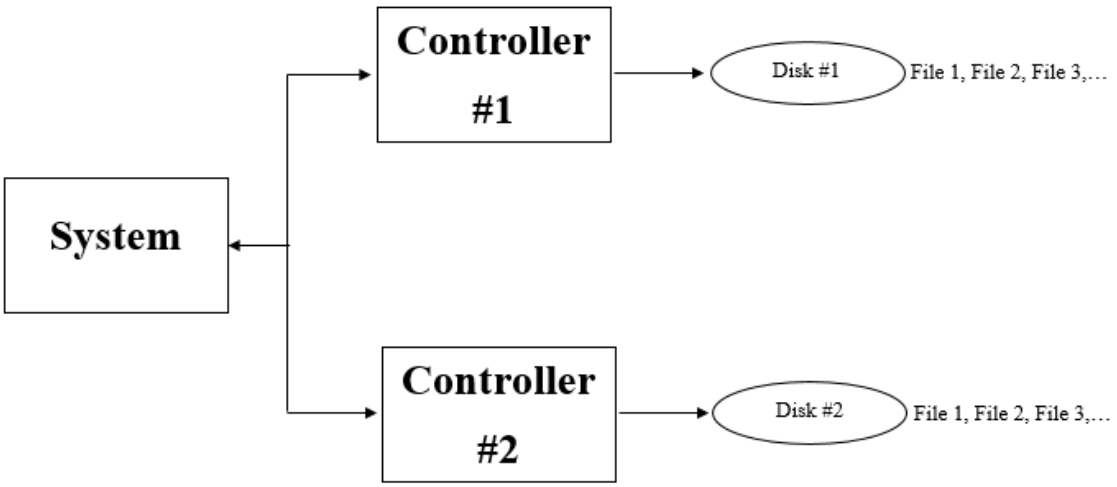
Ở cấp độ byte Striping chia ra thành từng gói nhỏ có kích thước một byte và bộ điều khiển sẽ ghi byte này lên ổ đĩa, trong cấp độ Block thì tập tin cũng bị chia nhỏ, lúc này chia nhỏ như thế nào thì tùy theo kích thước của Block nó như thế nào, tập tin sẽ được lưu và phân bổ trên các Block này.




**Hình 1-11 Raid song hành**

**Duplexing**

Đây là chuẩn mở rộng của Mirroring. Dữ liệu cũng được ghi trên hai ổ cứng nhưng phải có 2 bộ điều khiển RAID kết nối với 2 đĩa cứng. Từ đây ta đã thấy chuẩn này khá tốn kém. Nhưng có một đặc tính là Duplexing mang tính bảo mật cao hơn Mirroing vì ở đây nó dùng tới 2 card điêu khiển RAID.




**Hình 1-12 Raid ghép đôi (Mirror)**

**Parity RAID**

Đây là phương pháp bảo vệ an toàn cho dữ liệu, sử dụng các thông tin mang tính chẵn lẻ bằng cách lưu giữ một con số nhị phân 0 hoặc 1 cho biết tổng các bit trong gói tin là chẵn hay lẻ. Nếu dùng chuẩn này thì lợi ích lớn nhất của nó là không yêu cầu hệ thống RAID bớt đi một phần dung lượng để lưu trữ dữ liệu. Nhưng cũng có khuyết điểm của nó là phải yêu cầu hệ thống có một phần cứng thật mạnh.

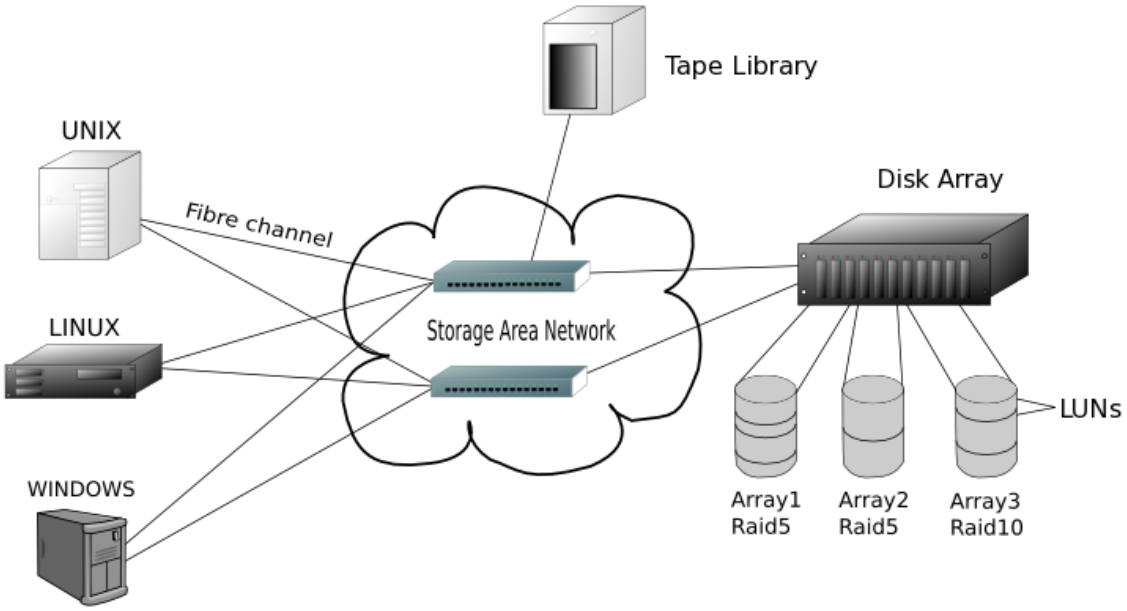
**JBOD**

JBOD được viết tắt từ "Just a Bunch of Disks" và không phải là hệ thống RAID chính thống, nó không có mục đích cải thiện hiệu suất của ổ cứng hay độ tin cậy. Nó dùng để ghép những ổ cứng có dung lượng khác nhau thành một dung lượng lưu trữ duy nhất.

Từ các chuẩn Raid trên cho ra đời các loại Raid như Raid 0, Raid 1, Raid 2, Raid 3, Raid 4, Raid 5, Raid 6, Raid 0+1, Raid 10,…là các ứng dụng dựa trên các công nghệ của những chuẩn RAID.

### 1.1.3.6 Công nghệ lưu trữ SAN

SAN là một hệ thống mạng lưu trữ chuyên dụng kết nối nhiều Server và nhiều thiết bị lưu trữ, với mục đích chính là truyền tải dữ liệu giữa hệ thống máy tính và phần tử lưu trữ và giữa các phần tử lưu trữ với nhau.




**Hình 1-13 Mô hình lưu trữ SAN**

Trong ảo hóa công nghệ lưu trữ mạng được dùng làm trung tâm của dữ liệu và cũng có thể làm nơi chứa các máy ảo khi cần thiết. Nó hỗ trợ các máy chủ có thể lấy dữ liệu từ nó để khởi động.

Lợi ích của SAN:

- Lưu trữ tập trung dữ liệu trên một hệ thống đơn nhất;
- Cho phép truy cập dữ liệu với tốc độ rất cao (2Gb/s; 4Gb/s tương lai lên tới 10Gb/s);
- Tính ổn định liên tục của thiết bị rất cao (99,999%);

- Dễ dàng nâng cấp, mở rộng trong tương lại (tăng thêm số lượng máy chủ, hay mở rộng dung lượng lưu trữ lên rất lớn và uyển chuyển): bảo vệ sự đầu tư thông qua khả năng tương thích ngược với các switch thế hệ cũ hơn, và khả năng nâng cấp firmware của thiết bị mà không phải dừng hoạt động hệ thống, đảm bảo mọi dịch vụ đều được duy trì liên tục 100%;
- Có khả năng sao lưu dữ liệu trong nội bộ hệ thống SAN, mà không phải dừng dịch vụ của máy chủ để sao lưu như các hệ thống lưu trữ khác, đồng thời không hề ảnh hưởng băng thông của mạng LAN khi thực hiện các thao tác backup;
- Bảo mật tốt: xác thực, xác quyền, điều khiển truy xuất và khả năng quản lý theo vùng tăng thêm mức bảo mật mạng;
- Hỗ trợ nhiều hệ điều hành và môi trường cluster, cho phép thiết kế linh động và bảo vệ sự đầu tư;
- Uyển chuyển trong khoảng cách, kết nối và hiệu suất, với khả năng hỗ trợ nhiều hơn 3000 port. Hỗ trợ cơ sở hạ tầng đa giao thức gồm FC, iSCSI, và FCIP.

## 1.2 Phần mềm tự do nguồn mở

Phần mềm tự do nguồn mở (tiếng Anh: Free and open-source software (Viết tắt là F/OSS, FOSS) hoặc free/libre/open-source software (viết tắt là FLOSS)) là phần mềm thỏa mãn cả hai yếu tố: tự do và nguồn mở.

Phần mềm tự do nguồn mở cung cấp cho người sử dụng 4 quyền: quyền: sử dụng (use), sao chép (copy), nghiên cứu (study), sửa đổi (change). Điều này đang được ngày càng nhiều người dùng cá nhân cũng như các doanh nghiệp thừa nhận.

Trong bối cảnh từ "free" trong tiếng Anh bị lẫn lộn giữa "miễn phí" và "tự do", tổ chức Free Software Foundation (Viết tắt là SFS) - một tổ chức ủng

hộ sáng kiến phần mềm nguồn mở - lưu ý rằng free hiểu theo nghĩa "tự do" (theo kiểu "độc lập - tự do - hạnh phúc") chứ không phải "miễn phí" (theo kiểu "miễn phí không mất tiền"), bởi "tự do" giá trị hơn "miễn phí".

FOSS là một thuật ngữ bao gồm bao gồm cả phần mềm tự do và phần mềm nguồn mở, mặc dù mô tả mô hình phát triển tương tự, nhưng khác nhau về văn hóa và triết lý sử dụng làm nền tảng. Phần mềm tự do tập trung vào triết lý về các quyền tự do mà nó mang lại cho người sử dụng, trong khi đó phần mềm nguồn mở tập trung vào các cảm nhận thế mạnh của mô hình phát triển ngang hàng của nó. FOSS là một thuật ngữ có thể được sử dụng mà không thiên vị đặc biệt đối với một trong hai cách tiếp cận chính.

### 1.2.1 Lịch sử hình thành

Trong những năm 50, 60, 70 thì người sử dụng máy tính đã có quyền tự do sử dụng các phần mềm miễn phí. Phần mềm miễn phí được những người sử dụng máy tính chia sẻ miễn phí với nhau và cũng do chính các nhà sản xuất chế tạo máy tính vì họ phấn khởi do có nhiều người đang cùng họ sáng tạo ra những phần mềm làm cho máy tính của họ có ích, không phải là những cục sắt vô dụng. Những tổ chức người tiêu dùng và nhà sản xuất được lập nên để tạo điều kiện cho việc trao đổi phần mềm ví dụ như SHARE. Vào những năm cuối của thập kỉ 60 thì xuất hiện những thay đổi đáng ngại: giá phần mềm tăng lên nhanh chóng, giữa nhà sản xuất phần cứng có cài đặt sẵn và nhà sản xuất phần mềm cũng xuất hiện sự cạnh tranh gay gắt để mở rộng thị trường vì khi đó phần mềm miễn phí vì chi phí của nó đã nằm trong giá phần cứng.Nhưng việc cài đặt sẵn những phần mềm như vậy lại không đem lại lợi ích gì cho việc bán phần mềm và người sử dụng đôi khi lại không cần những thứ được cài sẵn nên họ không muốn phải chi trả cho những thứ không xài tới. Trong bài "Nước Mỹ và IBM" United States vs. IBM, ngày 17/1/1969 chính quyền đã cho rằng việc cài đặt phần mềm đi kèm phần cứng khi bán ra thị trường là một kiểu cạnh tranhnkhông lành mạnh. Tuy rằng vẫn có nhiều

phần mềm là hoàn toàn miễn phí nhưng đa phần vẫn chỉ là những sản phẩm thương mại. Trong suốt quãng thời gian những năm 70 và thời kì đầu những năm 80,nền công nghệ phần mềmbắt đầu sử dụng các tiêu chuẩn về công nghệ (ví dụ như chỉ cho phân phối các phiên bản sử dụng, các bản sao nhị phân binary copies của chương trình máy tính) nhằm ngăn người sử dụng máy tính nghiên cứu và chỉnh sửa các phần mềm. Năm 1980 bộ luật quyền tác giả được mở rộng sang phần mềm máy tính.

Năm 1983, Richard Stallman, là thành viên lâu năm của cộng đồng hacker của MIT Artificial Intelligence Laboratory, chính ông cũng đã khởi xướng dự án GNU. Stallman nói rằng ông thấy chán nản vì những tác động thay đổi về văn hóa trong nền công nghiệp máy tính và người dùng máy. Sự phát triển các phần mềm cho hệ điều hành GNU,GNU operating system, bắt đầu từ 1/1984, và Tổ chức phần mềm tự do Free Software Foundation (FSF) được thành lập năm 1985. Ông đã phát triển một định nghĩa riêng cho phần mềm tự do và khái niệm "copyleft".

Và tiềm năng thương mại của các phần mềm tự do được các công ty lớn nhìn thấy như IBM, Red Hat, và Sun Microsystems. Cũng có rất nhiều công ty không thuộc lĩnh vực công nghệ thông tin chọn các phần mềm miễn phí để làm các trang web thông tin và thương mại của họ vì chi phí đầu tư thấp và khả năng tự do đóng gói dữ kiện của các phần mềm dạng này. Ngoài ra cũng có những công ty trong các ngành công nghiệp phi phần mềm sử dụng các công nghệ tương tự như công nghệ phát triển phần mềm tự do trong quá trình nghiên cứu và phát triển. Một ví dụ minh chứng là các nhà khoa học cũng luôn mong muốn có một quy trình nghiên cứu tiên tiến hơn những công nghệ hiện tại và đã xuất hiện nhiều thiết bị phần cứng như microchips với giấy phép copyleft. Creative Commons cũng bị ảnh hưởng rất nhiều bởi trào lưu phần mềm tự do.

### 1.2.2 Những lý do nên chọn phần mềm tự do nguồn mở

#### 1.2.2.1 Chi phí sở hữu

- Miễn phí bản quyền phần mềm (chỉ mất phí cho dịch vụ hỗ trợ và tùy biến sản phẩm).
- Miễn phí các phiên bản nâng cấp trong toàn bộ vòng đời sử dụng sản phẩm (chỉ mất phí cho dịch vụ nâng cấp).
- Giảm chi phí phát triển phần mềm đáp ứng theo yêu cầu nghiệp vụ (sử dụng phần mềm, mô-đun có sẵn để phát triển tiếp, sửa đổi điều chỉnh cho phù hợp với nghiệp vụ).
- Kéo dài thời gian sử dụng/tái sử dụng các phần cứng, thiết bị trong khi vẫn đảm bảo hiệu năng toàn hệ thống.
- Chi phí đầu tư, vận hành hệ thống tập trung cho các dịch vụ "hữu hình" đem lại giá trị trực tiếp, thiết thực cho tổ chức như: tư vấn, sửa đổi theo yêu cầu, triển khai, đào tạo, bảo trì, nâng cấp hệ thống...
- Mức chi phí tiết kiệm tới 75% so với phần mềm license ngay trong năm đầu tiên.

#### 1.2.2.2 Giảm tối đa sự phụ thuộc vào nhà cung cấp

(nguy cơ chính dẫn đến dịch vụ kém, do không có cạnh tranh), hoặc "bị ép" trong các trường hợp cần đàm phán về chi phí, dịch vụ (mỗi sản phẩm phần mềm tự do nguồn mở có thể có nhiều nhà cung cấp dịch vụ tương tự), nâng cấp phần mềm, mở rộng hệ thống (với mã nguồn trong tay, có thể dễ dàng nâng cấp, mở rộng hệ thống theo yêu cầu trong mỗi giai đoạn phát triển).

#### 1.2.2.3 Bổ sung thêm nhiều lựa chọn

Tiền bản quyền tốn kém là nguyên nhân chính khiến các doanh nghiệp công nghệ thông tin chuyển đổi sang phần mềm nguồn mở tự do. Đối với các hệ thống đang hoạt động không có bản quyền, chủ động thực hiện chuyển đổi

sẽ tránh được "nguy cơ" bị phạt vi phạm bản quyền và/hoặc bị "bắt buộc" mua license.

#### 1.2.2.4 Hiệu quả trong giáo dục

Phần mềm nguồn mở rất tốt trong giáo dục và học tập nhờ tính minh bạch, cho phép người học tiếp cận với mã nguồn gốc và học tập những công nghệ cao cấp từ trong nhân hệ thống chứ không phải thứ công nghệ nửa vời mà các nhà sản xuất nghĩ ra.

#### 1.2.2.5 An toàn và bảo mật

Theo báo cáo của Gartner & nhiều tổ chức phân tích độc lập, Phần mềm tự do nguồn mở giúp tăng cường độ tin cậy (có thể kiểm chứng không có mã độc, "cửa sau"... với mã nguồn được phân phối kèm), ổn định (tuân theo các chuẩn mở ứng dụng lâu dài), tính an toàn, bảo mật toàn hệ thống. Trong một báo cáo năm 2011 của Bộ quốc phòng Mỹ, đơn vị này khẳng định: Mở mới là an ninh, và sẽ không tồn tại phần mềm độc quyền trong quân đội và chính phủ nhờ sự phát triển của công nghệ mở. Một quốc gia không thể phụ thuộc vào một thứ không rõ ràng hoặc được độc quyền, và cách thoát khỏi nó chỉ có thể bằng phần mềm tự do nguồn mở. Những công ty hàng đầu thế giới đang sử dụng phần mềm tự do nguồn mở, những quốc gia hàng đầu thế giới cũng đang chuyển sang sử dụng phần mềm tự do nguồn mở. Và đó không phải lựa chọn ngẫu nhiên, đó là vì lợi ích của mỗi quốc gia, mỗi doanh nghiệp.

# CHƯƠNG 2: Công nghệ ảo hóa Docker

## 2.1 Khái niệm về công nghệ ảo hóa Docker

### 2.1.1 Container là gì

Các phần mềm sẽ được Container Engine (một công cụ ảo hóa tinh gọn, được cài trên host OS) Các phần mềm sẽ được Container Engine (một công cụ ảo hóa tinh gọn, được cài đặt trên host OS) cô lập bằng cách đóng gói chúng thành các container.

Container là giải pháp để chuyển giao phần mềm một cách đáng tin cậy (không phát sinh lỗi) giữa các môi trường máy tính khác nhau bằng cách:

Tạo ra một môi trường bị cô lập chứa mọi thứ mà phần mềm cần để có thể chạy được Không bị các yếu tố liên quan đến môi trường hệ thống làm ảnh hưởng tới Cũng như không làm ảnh hưởng tới các phần còn lại của hệ thống.

Các tiến trình (process) trong một container bị cô lập với các tiến trình của các container khác trong cùng hệ thống tuy nhiên tất cả các container này đều chia sẻ kernel của host OS (dùng chung host OS).

Container là một nền tảng mở dành cho các lập trình viên, quản trị hệ thống dùng để xây dựng, vận chuyển và chạy các ứng dụng dễ dàng hơn. (VD: Bạn sẽ có thể chạy một ứng dụng Java mà không cần cài JDK vào máy local bạn đang sử dụng.)

Ưu điểm:

- Linh động: Triển khai ở bất kỳ nơi đâu do sự phụ thuộc của ứng dụng vào tầng OS cũng như cơ sở hạ tầng được loại bỏ. Scale-up & scale-down: Do chia sẻ host OS nên container có thể được tạo gần như một cách tức thì.

- Nhẹ: Container cũng sử dụng chung các images nên cũng không tốn nhiều disks. Tính đồng nhất :Khi nhiều người cùng phát triển trong cùng một dự án sẽ không bị sự sai khác về mặt môi trường. Đóng gói: Có thể ẩn môi trường bao gồm cả app vào trong một gói được gọi là container. Có thể test được các container. Việc bỏ hay tạo lại container rất dễ dàng.
- Tạo OS ảo nên thời gian khởi động là rất nhanh. Điều này khác với vagrant, vagrant tạo môi trường ảo ở level phần cứng, nên khi khởi động mất nhiều thời gian hơn

Nhược điểm:

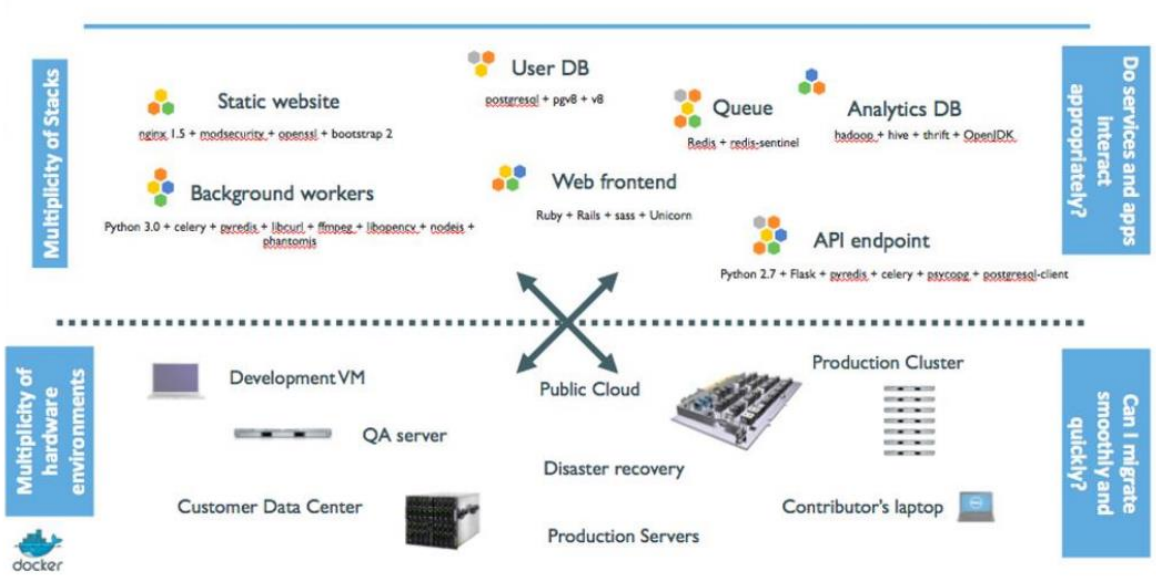
- Xét về tính an toàn: Do dùng chung OS nên nếu có lỗ hổng nào đấy ở kernel của host OS thì nó sẽ ảnh hưởng tới toàn bộ container có trong host OS đấy. Ngoài ra hãy thử tưởng tượng với host OS là Linux, nếu trong trường hợp ai đấy hoặc một ứng dụng nào đấy có trong container chiếm được quyền superuser, điều gì sẽ xảy ra? Về lý thuyết thì tầng OS sẽ bị crack và ảnh hưởng trực tiếp tới máy host bị hack cũng như các container khác trong máy đó (hacker sử dụng quyền chiếm được để lấy dữ liệu từ máy host cũng như từ các container khác trong cùng máy host bị hack chẳng hạn).

### 2.1.2 Công nghệ Docker

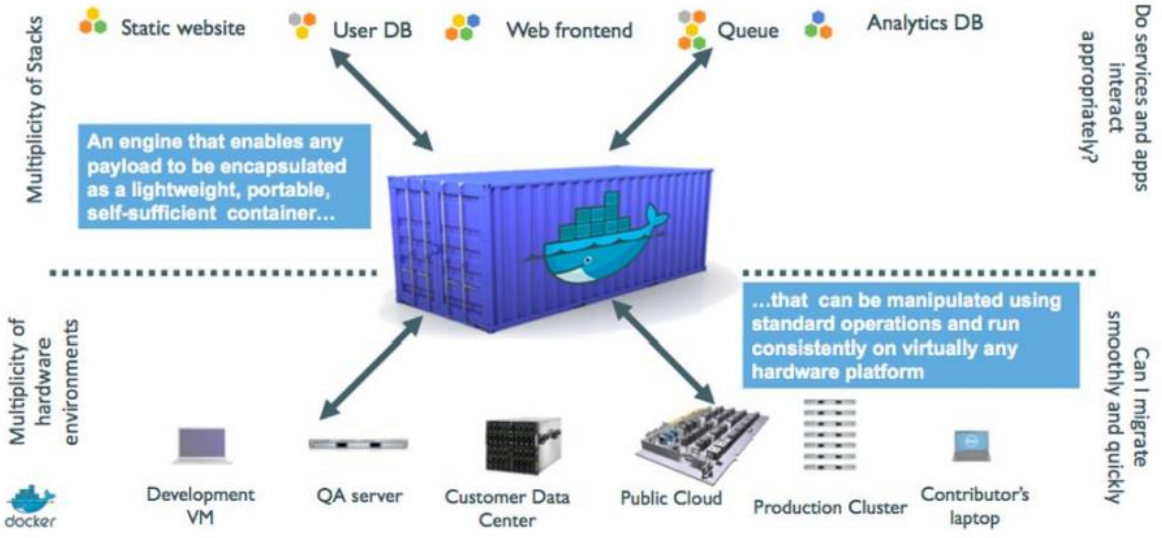
Docker là một nền tảng mở dành cho các lập trình viên, quản trị hệ thống dùng để xây dựng, vận chuyển và chạy các ứng dụng phân tán. Ban đầu viết bằng Python, hiện tại đã chuyển sang Go-lang.

Docker đưa ra một giải pháp mới cho vấn đề ảo hóa, thay vì tạo ra các máy ảo con chạy độc lập kiểu hypervisors (tạo phần cứng ảo và cài đặt hệ điều hành lên đó), các ứng dụng sẽ được đóng gói lại thành các Container (Công-ten-nơ) riêng lẻ. Các Container này chạy chung trên nhân hệ điều hành

qua LXC (Linux Containers), chia sẻ chung tài nguyên của máy mẹ, do đó, hoạt động nhẹ và nhanh hơn các máy ảo dạng hypervisors[6].




**Hình 2-1 Hypervisor**




**Hình 2-2 Docker**

### 2.1.3 Các thành phần chính

Các thành phần chính của Docker bao gồm:

- Docker Engine: là thành phần chính của Docker, như một công cụ để đóng gói ứng dụng;
- Docker Hub: là dịch vụ cloud để chia sẻ ứng dụng và tự động hóa chuỗi các công việc liên tục, có thể thao tác pull/push với các images.

### 2.1.4 Một số khái niệm

Một số khái niệm phổ biến về Docker:

#### 2.1.4.1 Docker images

Là một "read-only template". Tương tự như file .gho để ghost win, ở Docker thì đó được gọi là image, image này không phải là một file vật lý mà nó chỉ được chứa trong Docker. Một image bao gồm hệ điều hành (Windows, CentOS, Ubuntu, …) và các môi trường lập trình được cài sẵn (httpd, mysqld, nginx, python, git,… ).

#### 2.1.4.2 Docker registries

Là kho chứa images. Người dùng có thể tạo ra các images của mình và tải lên đây hoặc tải về các images được chia sẻ.

#### 2.1.4.3 Docker container

Hoạt động giống như một thư mục (directory), chứa tất cả những thứ cần thiết để một ứng dụng có thể chạy được. Mỗi một docker container được tạo ra từ một docker image. Các thao tác với một container: chạy, bật, dừng, di chuyển, và xóa. Các files và settings được sử dụng trong container được lưu và sử dụng lại, gọi chung là images của docker. Docker hub là nơi chia sẻ và lưu giữ các file images này (hiện có khoảng 300.000 images).

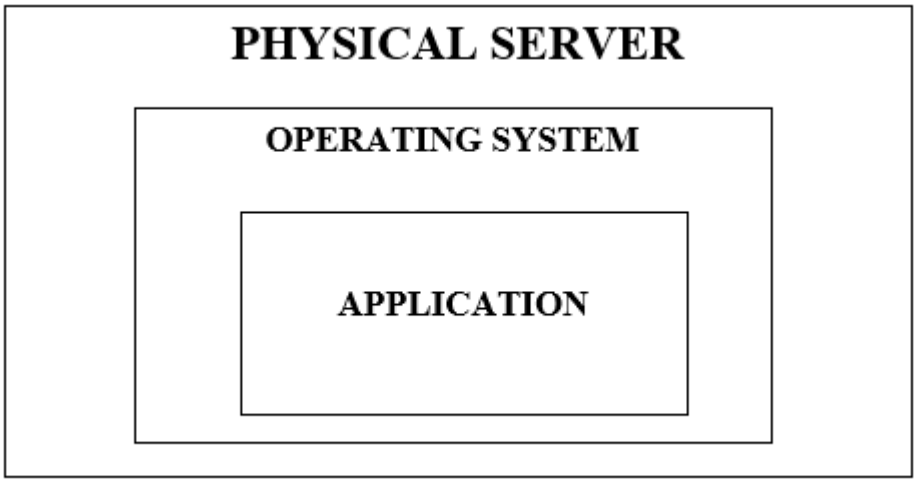
#### 2.1.4.4 Dockerfile

Là một file chứa tập hợp các lệnh để Docker có thể đọc và thực hiện để đóng gói một image theo yêu cầu người dùng.

#### 2.1.4.5 Orchestration

Là các công cụ, dịch vụ dùng để điều phối và quản lý nhiều containers sao cho chúng làm việc hiệu quả nhất.

Ngày xưa, mô hình máy chủ thường là 1 máy chủ vật lý + 1 hệ điều hành (OS) + 1 application.

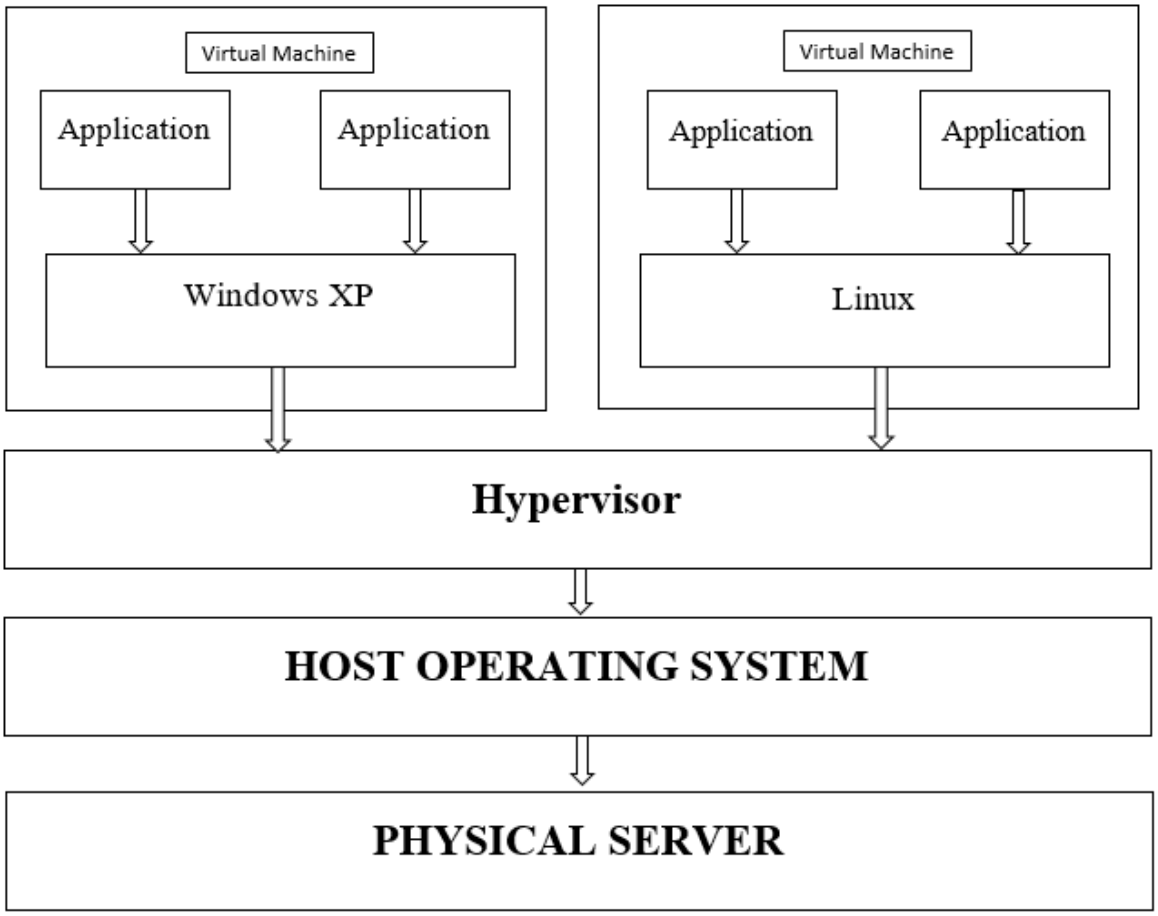



**Hình 2-3 Mô hình máy chủ bình thường**

Khi ứng dụng phát triển, mô hình này nảy sinh ra nhiều vấn đề, ví dụ:

- Lãng phí tài nguyên: mặc dù cấu hình máy khỏe, ổ cứng dung lượng lớn, nhưng hệ thống lại không tận dụng được hết lợi thế này;
- Khó khăn trong việc mở rộng hệ thống: muốn mở rộng phải thuê thêm server, cấu hình, cân bằng tải (load balacing), …

Lúc này, công nghệ ảo hóa (vitualization) ra đời.

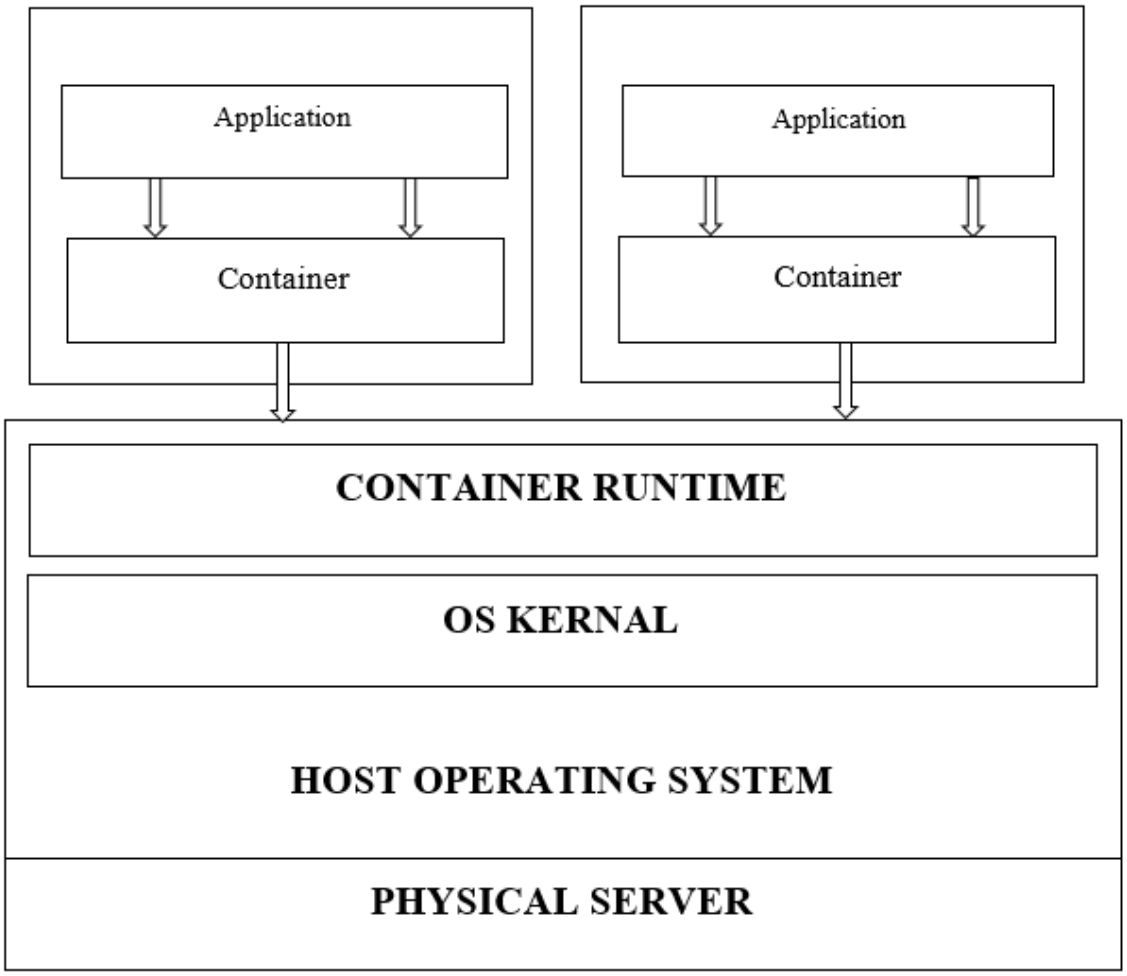



**Hình 2-4 Mô hình máy ảo VMs**

Với công nghệ ảo hóa, trên cùng 1 máy chủ vật lý, có thể tạo ra nhiều OS, tức là sẽ chạy được nhiều application. Vậy là tài nguyên của máy được tận dụng tốt hơn. Tuy nhiên, việc ảo hóa này lại nảy sinh vấn đề mới:

- Ngốn tài nguyên: khi chạy 1 máy ảo, nó sẽ luôn chiếm 1 phần tài nguyên cố định. Vd: máy chủ có 512GB SSD, 16GB RAM. Tạo ra 4 máy ảo Linux, mỗi máy cấp 64GB SSD và 2GB RAM. Như vậy, sẽ mất 256 GB SSD để chứa 4 máy ảo, và khi chạy cùng 4 máy ảo lên cùng lúc, chúng sẽ chiếm 8GB RAM. Mặc dù chỉ chạy lên để không đó thôi, chưa dùng gì cả nhưng nó vẫn chiếm từng đó;
- Tốn thời gian thực thi: thời gian khởi động, shutdown của các máy ảo sẽ lâu, thường là hàng phút;
- Cồng kềnh: việc phải chịu tải cho cả 1 nhóm máy ảo như vậy thì server không thể chạy hết hiệu suất được.

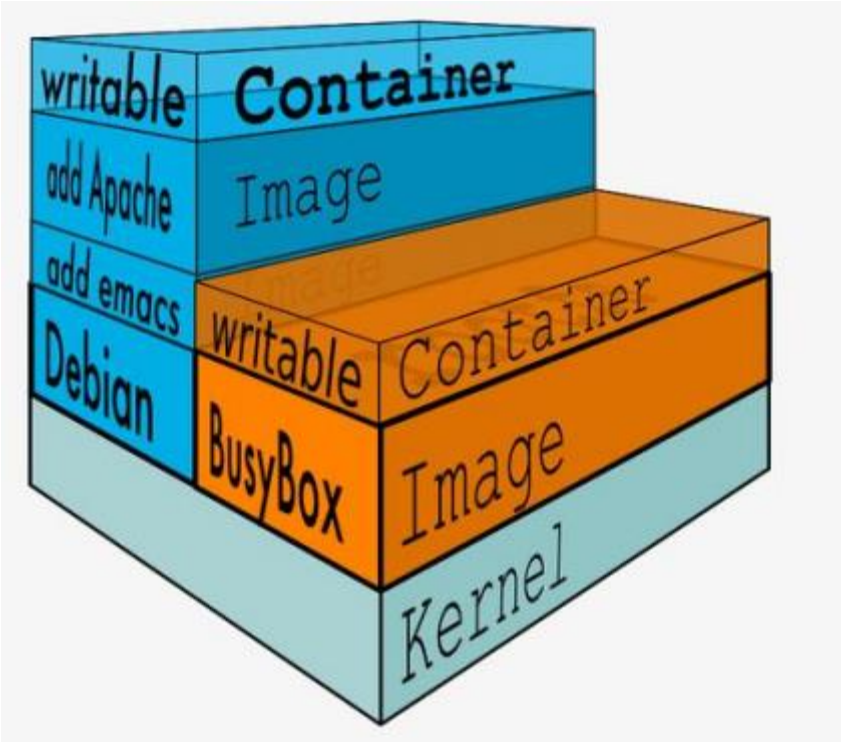
Bước tiến hóa tiếp theo, người ta phát minh ra containerlization.




**Hình 2-5 Mô hình ảo hóa container**

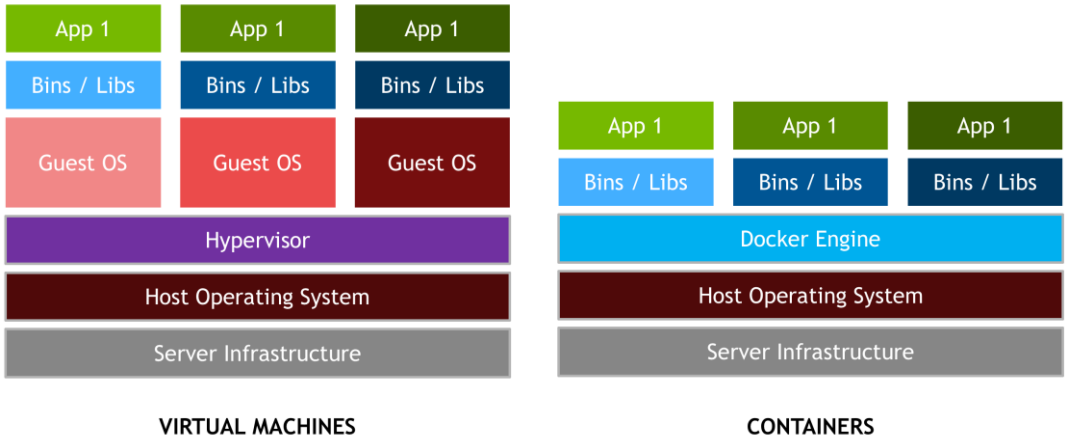
Với công nghệ này, trên một máy chủ, ta sẽ sinh ra được nhiều máy con (giống với ảo hóa), nhưng điều đặc biệt là các máy con (Guess OS) này đều dùng chung phần nhân của máy mẹ (host OS) và chia sẻ với nhau tài nguyên của máy mẹ (RAM chẳng hạn). Như vậy việc tận dụng tài nguyên sẽ được tối ưu hơn.

Ngoài ra, việc sử dụng hệ thống file cắt lớp (layer file system) sẽ khiến việc tối ưu tài nguyên hiệu quả hơn.




**Hình 2-6 Hệ thống file cắt lớp Container**

Cụ thể, mỗi máy con (container) mới, nó sẽ được xây dựng dựa trên 1 file ảnh (image) dạng chỉ đọc (read-only). Trong mỗi máy con sẽ có thêm 1 lớp bọc có thể ghi được (writabe-layer), các thay đổi trong máy con sẽ được ghi lên đây. Như vậy, từ 1 image ban đầu, ta có thể tạo nhiều máy con mà chỉ tốn rất ít dung lượng ổ đĩa.




**Hình 2-7 Sự khác biệt giữa Docker và VMs**

Điểm khác biệt chính là các containers sử dụng chung kernel với Host OS nên các thao tác bật, tắt rất nhẹ nhàng, nhanh chóng.

- Ưu điểm: nhanh, nhẹ, có thể chia sẻ dễ dàng qua DockerHub;
- Nhược điểm: mới, cập nhật thay đổi thường xuyên.

## 2.2 Cài đặt Docker

Docker hỗ trợ nhiều nền tảng hệ điều hành khác nhau bao gồm Linux, Windows và cả Mac. Ngoài ra, Docker còn hỗ trợ nhiều dịch vụ điện toán đám mây nổi tiếng như Microsoft Azure hay Amazon Web Services. Tuy vậy ban đầu nó được xây dựng trên nền tảng Linux. Vì Docker cần can thiệp vào phần lõi, nhân Kernel trong khi đó Linux là mã nguồn mở.

Khi nhận thấy tiềm năng của Docker, Windows đã tham gia, và thế là Docker với Windows bắt tay nhau nhưng có vẻ không khả quan bởi vì nhân Windows có nhưng thứ không public được.

Nên cho tới hiện tại khi cài Docker trên Windows hay Mac thì Docker sẽ cài một máy ảo Linux trên máy thật và Docker hoạt động dựa trên máy ảo Linux đó.

Docker có 2 phiên bản

- CE: Dành cho nhà phát triển, nhóm nhỏ;
- EE: Dành cho doanh nghiệp.

Docker image là nền tảng của container, có thể hiểu Docker image như là một template của container, nó sẽ tạo ra container khi thực hiện câu lệnh chạy image đó. Nếu nói với ngôn ngữ lập trình hướng đối tượng, Docker image là class, còn container là thực thể (instance) của class đó.

Docker hổ trợ cài đặt được trên nhiều OS (CentOS 7, Debian, Ubuntu, Fedora) tuy nhiên Ubuntu gần như được hỗ trợ nhiều nhất.

### 2.2.1 Cài đặt Docker

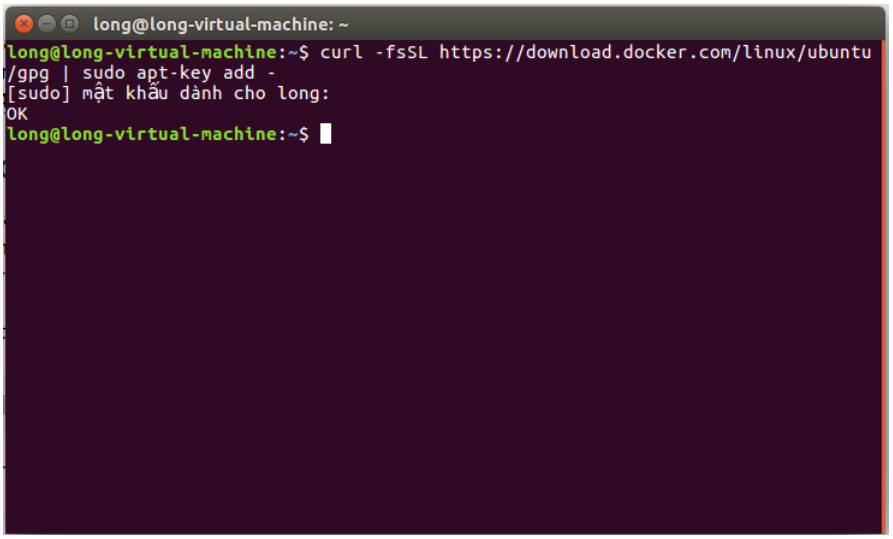
Gói cài đặt Docker có sẵn trong kho lưu trữ chính thức của Ubuntu 16.04 có thể không phải là phiên bản mới nhất. Để có được phiên bản mới

nhất và tốt nhất, nên cài đặt Docker từ kho lưu trữ Docker chính thức. Phần này cho thấy làm thế nào để làm điều đó.

Đầu tiên, thêm khóa GPG cho kho lưu trữ Docker chính thức vào hệ thống:

curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -

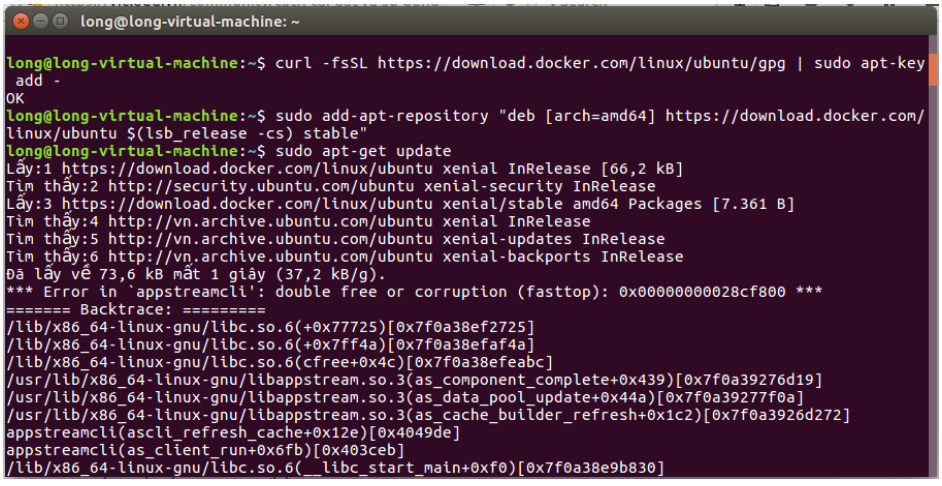
Sau đó nhập mật khẩu của người dùng




**Hình 2-8 Khóa GPG cho kho lưu trữ Docker**

Thêm kho lưu trữ Docker vào các nguồn APT:

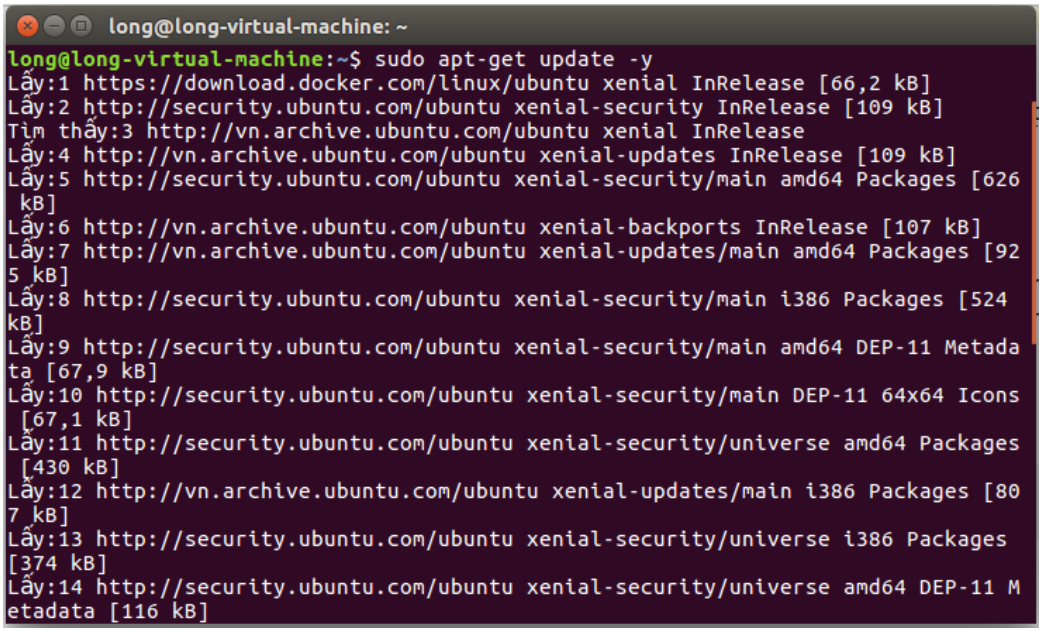
sudo add-apt-repository "deb [arch=amd64] https://
download.docker.com/linux/ubuntu $(lsb_release -cs) stable"



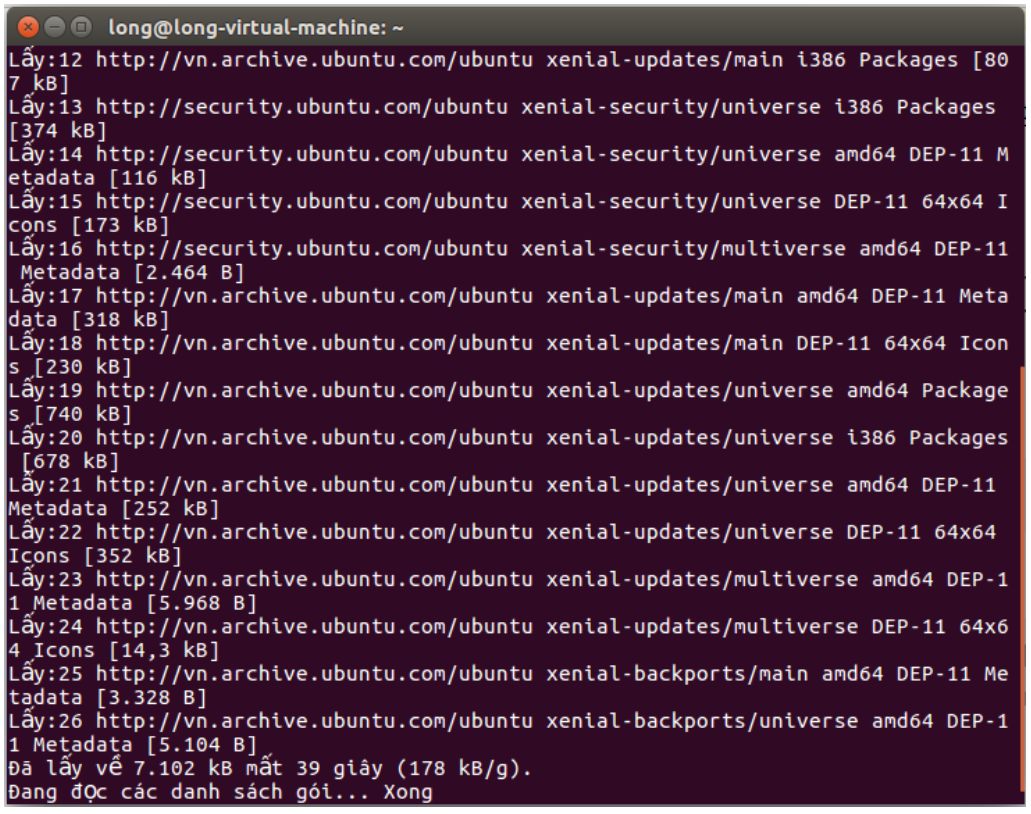

**Hình 2-9 Thêm kho lưu trữ Docker**

Tiếp theo, cập nhật cơ sở dữ liệu gói với các gói Docker từ repo mới được bổ sung:

sudo apt-get update -y



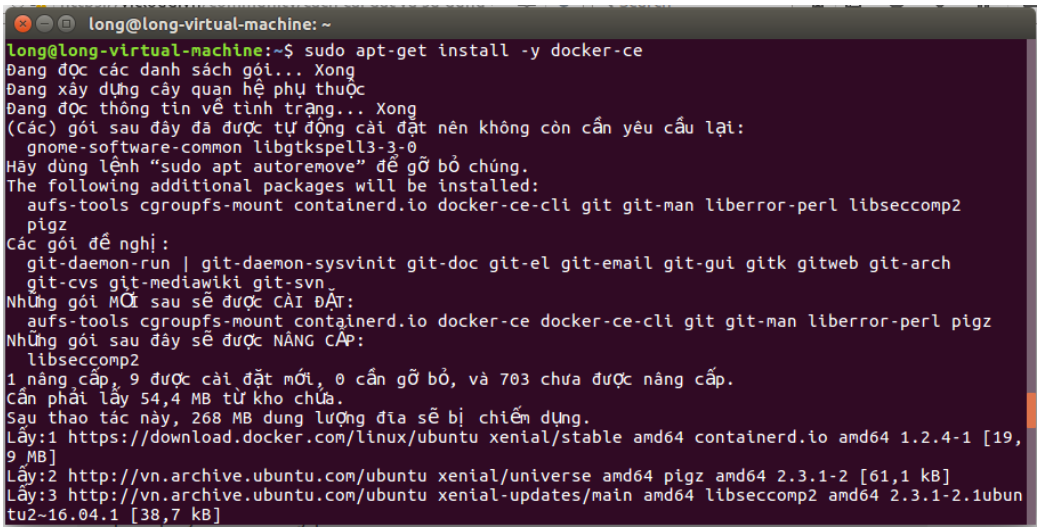

**Hình 2-10 Cập nhật cở sở dữ liệu**




**Hình 2-11 Cập nhật cơ sở dữ liệu**

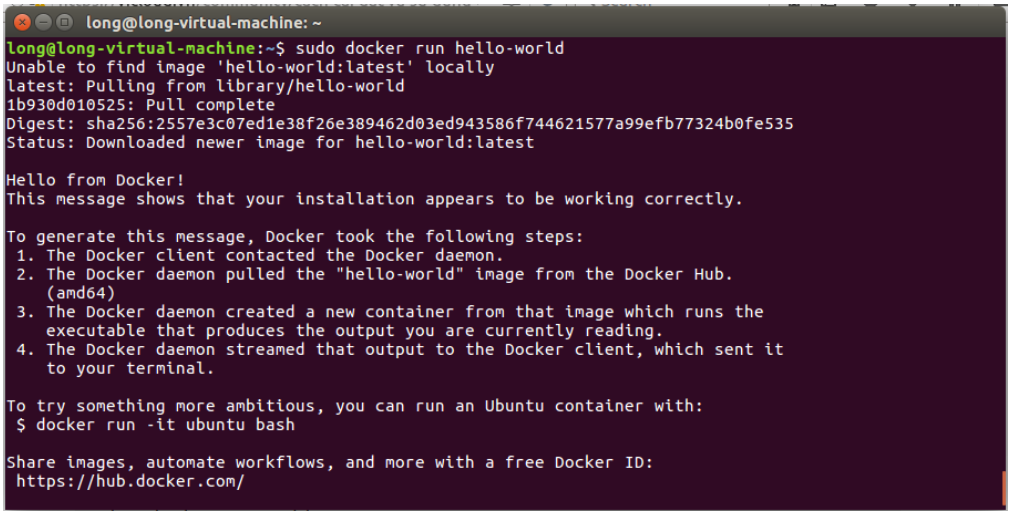
Tiếp theo, cài đặt Docker:

sudo apt-get install -y docker-ce



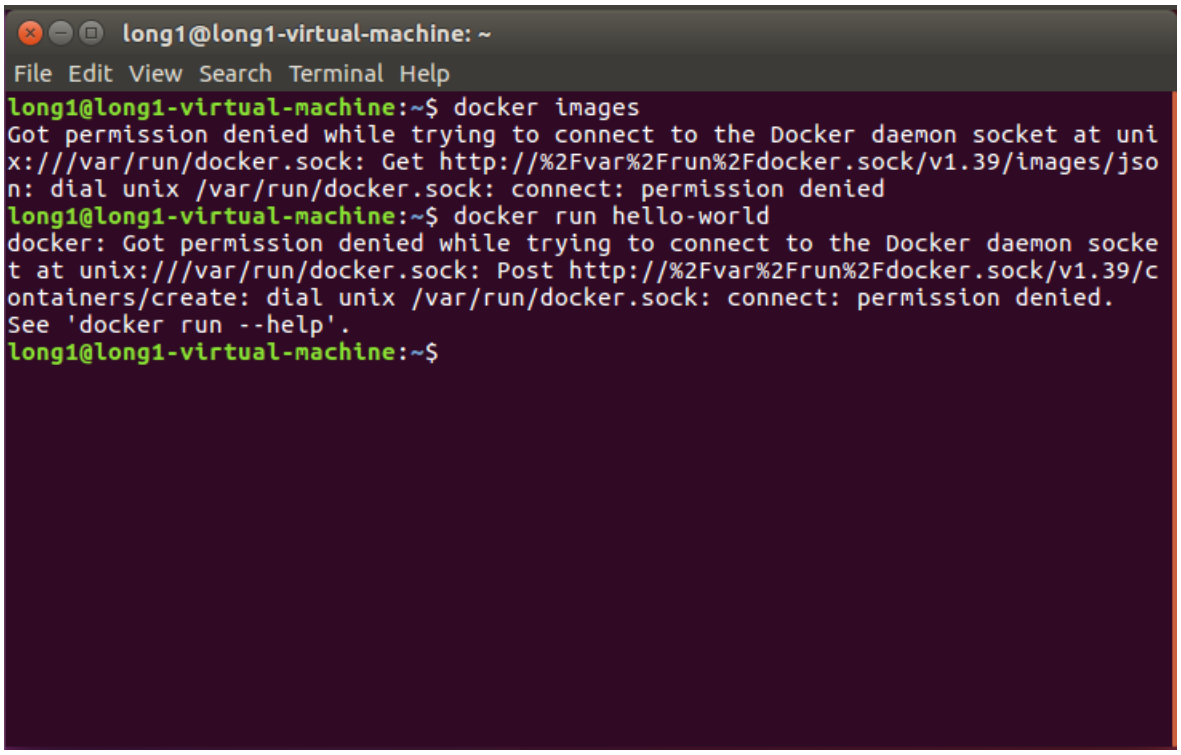

**Hình 2-12 Cài đặt Docker**

Chạy thử 1 chương trình đơn giản như hello-world để kiểm tra




**Hình 2-13 Hello-world**

Sau khi cài đặt docker, chỉ có thể chạy được Docker Command với user thông thường với quyền sudo hoặc user root. Điều này khá bất tiện vì phải gõ tiền tố sudo trước docker command. Còn nếu không gõ thì sẽ gặp phải lỗi như sau




**Hình 2-14 Lỗi khi không có tiền tố sudo**

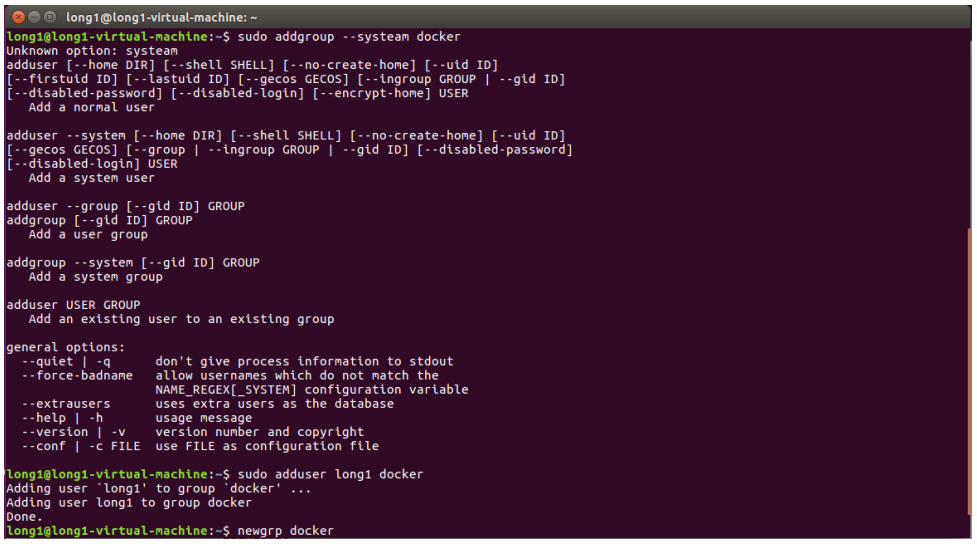
Giải pháp cho vấn đề này là tạo 1 group docker và thêm user vào group này. Để thực hiện điều đó cần phải nhập các command sau

sudo addgroup –system docker

sudo adduser (user name) docker

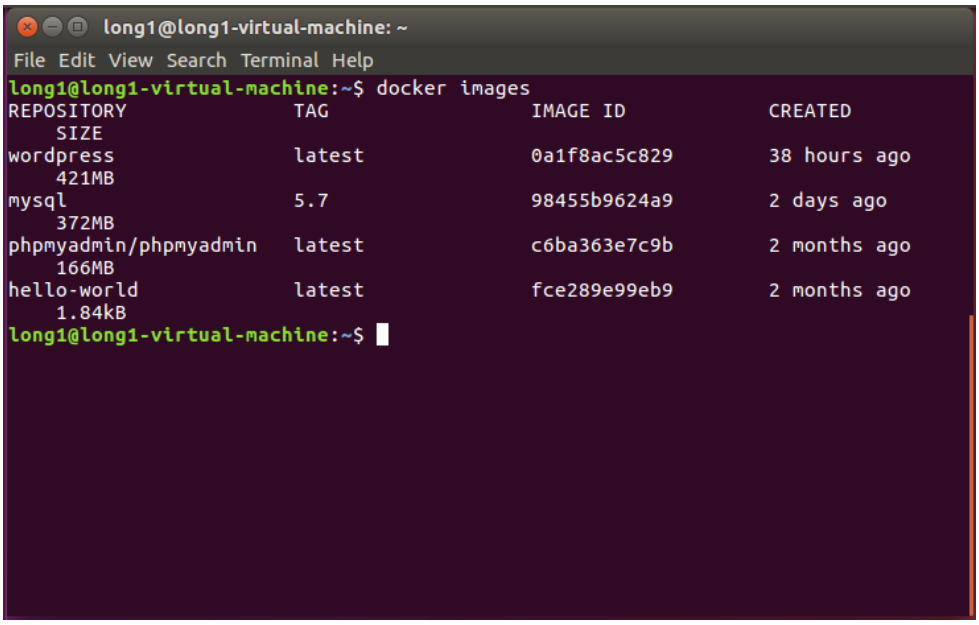
newgrp docker

Trong đó user name là user mà người dùng sử dụng




**Hình 2-15 Giải pháp cho tiền tố sudo**

Và giờ đây sẽ không còn phải gõ lệnh sudo trước mỗi docker command nữa




**Hình 2-16 Câu lệnh không cần tiền tố sudo**

### 2.2.2 Cài đặt Docker compose

#### 2.2.2.1 Docker Compose là gì

Compose là công cụ giúp định nghĩa và khởi chạy multi-container Docker applications. Trong Compose, chúng ta sử dụng Compose file để cấu hình application's services. Chỉ với một câu lệnh, lập trình viên có thể dễ dàng create và start toàn bộ các services phục vụ cho việc chạy ứng dụng.

Compose là một công cụ tuyệt với không chỉ dùng cho development, testing, staging environments, mà còn ứng dụng trong CI workflows. Việc sử dụng Docker Compose được tóm lược trong 3 bước cơ bản sau:

#### 2.2.2.2 Cài đặt Docker Compose

Bước 1: chạy dòng lệnh dưới để tải xuống bản ổn định nhất của docker compose

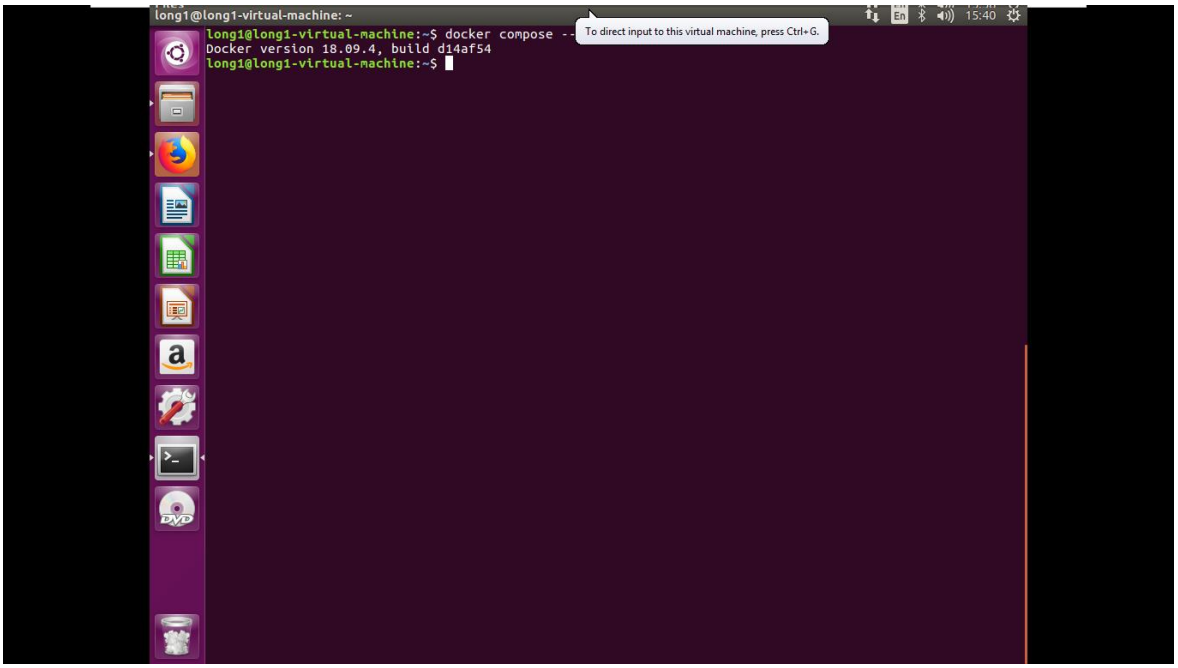
```
Sudo curl -L
"https://github.com/docker/compose/releases/download/1.24.0/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
```

Bước 2: Đặt quyền cho docker compose

sudo chmod +x /usr/local/bin/docker-compose

Bước 3: Kiểm tra phiền bản cài đặt

Docker-compose --version




**Hình 2-17 Kiểm tra version docker compose**

# CHƯƠNG 3: Thử nghiệm ảo hóa ứng dụng với Docker

## 3.1 Một số lệnh cơ bản với Docker

- Pull một image từ Docker Hub : docker pull {image_name};
- Liệt kê các images hiện có : docker images;
- Xóa một image : docker rmi {image_id/name};
- Liệt kê các container đang chạy : docker ps;
- Liệt kê các container đã tắt : docker ps –a;
- Xóa một container : docker rm -f {container_id/name};
- Đổi tên một container : docker rename {old_container_name} {new_container_name};
- Khởi động một container : docker start {new_container_name};
- docker exec -it {new_container_name} /bin/bash;
- Tạo mới một container, đồng thời khởi động với tùy chọn cổng và volume : docker run --name {container_name} -p {host_port}:{container_port} -v {/host_path}:{/container_path} -it {image_name} /bin/bash;
- Xem các thay đổi trên container : docker diff {container_name};
- Commit các thay đổi trên container và image : docker commit -m "message" {container_name} {image_name};
- Save image thành file .tar : docker save {image_name} > {/host_path/new_image.tar};
- Tạo một image mới từ file .tar : cat musashi.tar | docker import - {new_image_name}:latest;
- Xem lịch sử các commit trên image : docker history {image_name};

- Khôi phục lại images từ IMAGE_ID : docker tag {iamge_id} {image_new_name}:{tag};
- Build một image từ container : docker build -t {container_name} . ;

Dấu . ở đây ám chỉ Dockerfile đang nằm trong thư mục hiện tại[7].

## 3.2 Tạo một image và push lên hub.docker.com

Bước 1: Thiết lập tài khoản của docker.

Trước tiên cần tạo một tài khoản trên https://hub.docker.com/ để có thể lưu những image được tạo trên đó.

Sau đó login vào docker trên terminal bằng: docker login.

Bước 2: Tạo image

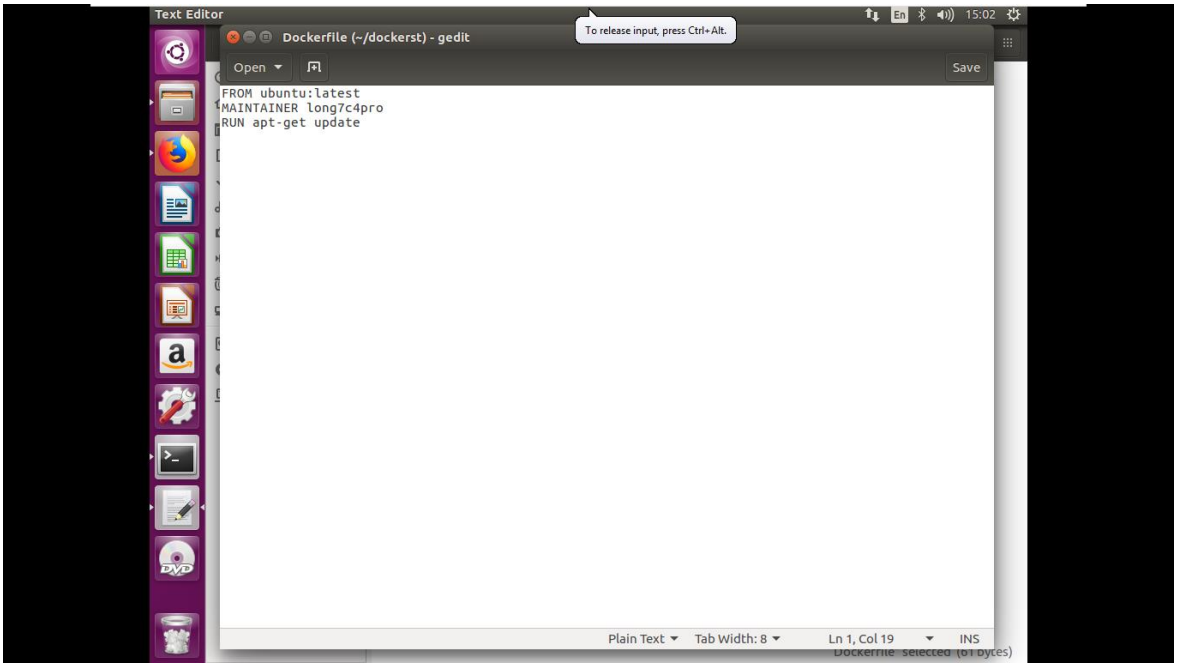
Với mỗi một project nên tạo một thư mục riêng bằng lệnh.

Mkdir dockerst.

Trong thư mục dockerst tạo 1 file Dockerfile.

Touch Dockerfile

Truy cập vào dockerfile và nhập nội dung.




**Hình 3-1 Dockerfile**

Trong đó:

- FROM ubuntu: latest là môi trường làm việc;
- Maintainer là người tiến hành xây dựng lên images;
- RUN apt-get update là nội dung để customize image ubuntu.

Sau đó vào terminal chạy lệnh

Sudo chmod 755 Dockerfile

Trong đó:

- Sudo để cấp quyền chạy lệnh
- CHMOD 755 cho các thư mục có nghĩa là:

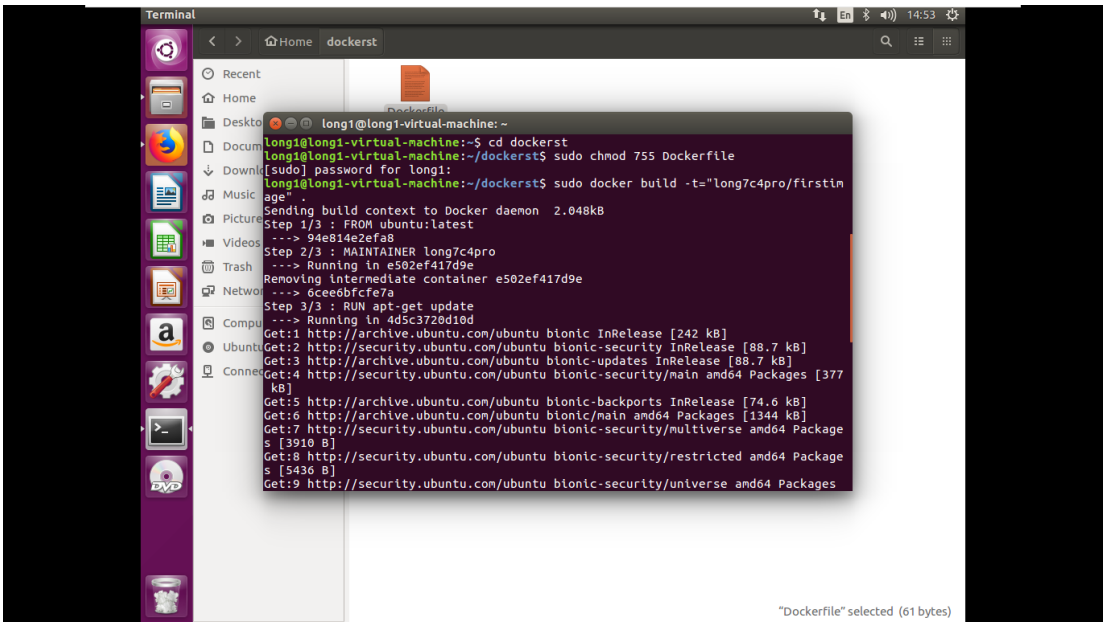
7 = 4 + 2 + 1: Người sở hữu thư mục có quyền đọc thư mục (read); chỉnh sửa thư mục (write); liệt kê các thư mục và file bên trong (execute);

5 = 4 + 0 + 1: Những người cùng nhóm chỉ có quyền đọc thư mục (read); liệt kê các thư mục và file bên trong (execute);

5 = 4 + 0 + 1: Những người còn lại chỉ có quyền đọc thư mục (read); liệt kê các thư mục và file bên trong (execute).

Tiếp tục nhập lệnh:

sudo docker build -t="long7c4pro/firstimage" .




**Hình 3-2 Build image**

Docker build để xây dựng image.

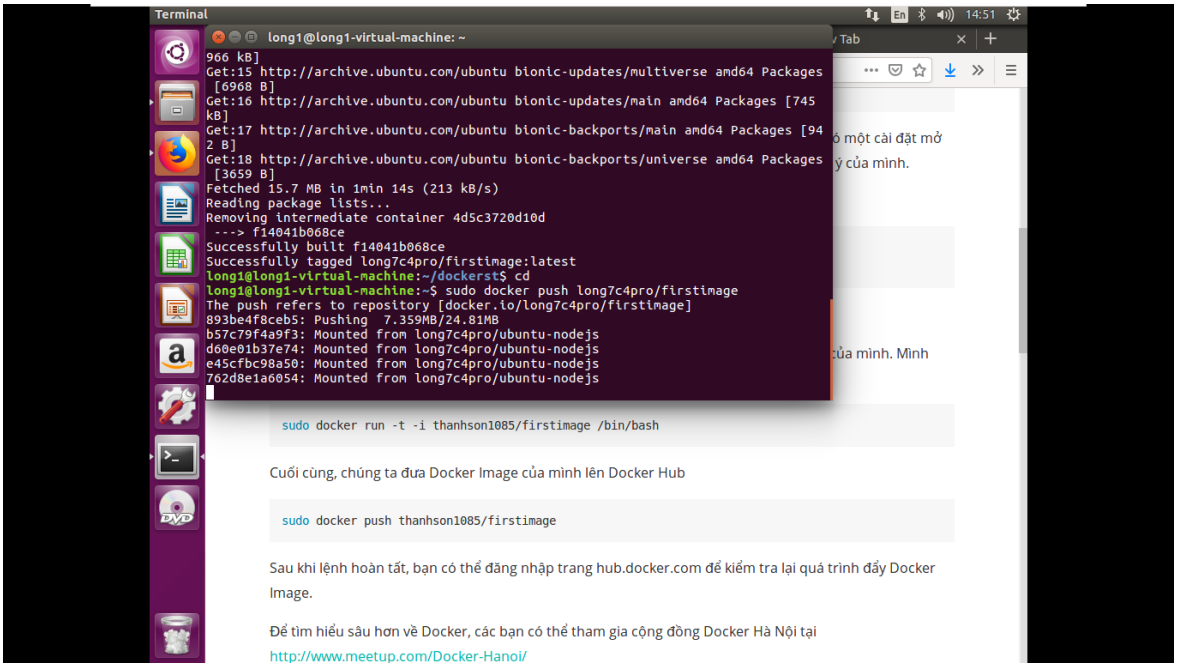
Long7c4pro/firstimage là tên của image.

Cuối cùng chạy thử image bằng câu lệnh:

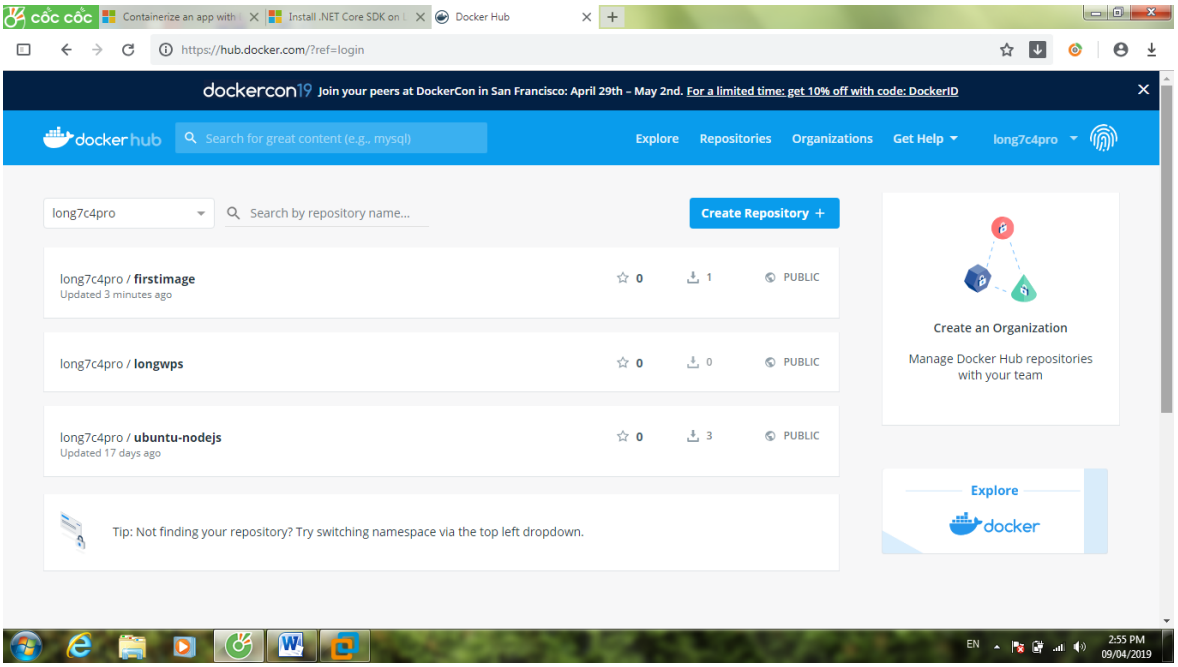
sudo docker run -t -i long7c4pro/firstimage /bin/bash

Bước 3: Push image lên trên hub.docker.com bằng lệnh

sudo docker push long7c4pro/firstimage




**Hình 3-3 Push image**




**Hình 3-4 Image được push lên hub.docker.com**

## 3.3 Chạy một image trên docker

Có 2 phương pháp cài đặt WordPress trên Docker. Phương pháp đầu tiên là thủ công, sử dụng CLI để chạy WordPress. Phương pháp thứ 2 là cleaner, một phương pháp có hệ thống hơn sử dụng Docker Compose.

Chạy Wordpress với Docker Compose

Đây là phương pháp được khuyên dùng để tạo WordPress container hoặc bất kỳ container nào trong Docker. Nó dùng công cụ chính thức Docker Compose của Docker. Mỗi container được tạo bằng Docker Compose sử dụng file config, cho nên rất dễ để port.

Bước 1: Tạo 1 thư mục wordpress bằng lệnh: mkdir my_wordpress

Bước 2: Vào trong thư mục my_wordpress tạo 1 file tên docker-compose.yml

Bước 3: Vào trong thư mục docker-compose.yml và nhập:

```
version: '3.3'
services:
  db:
```

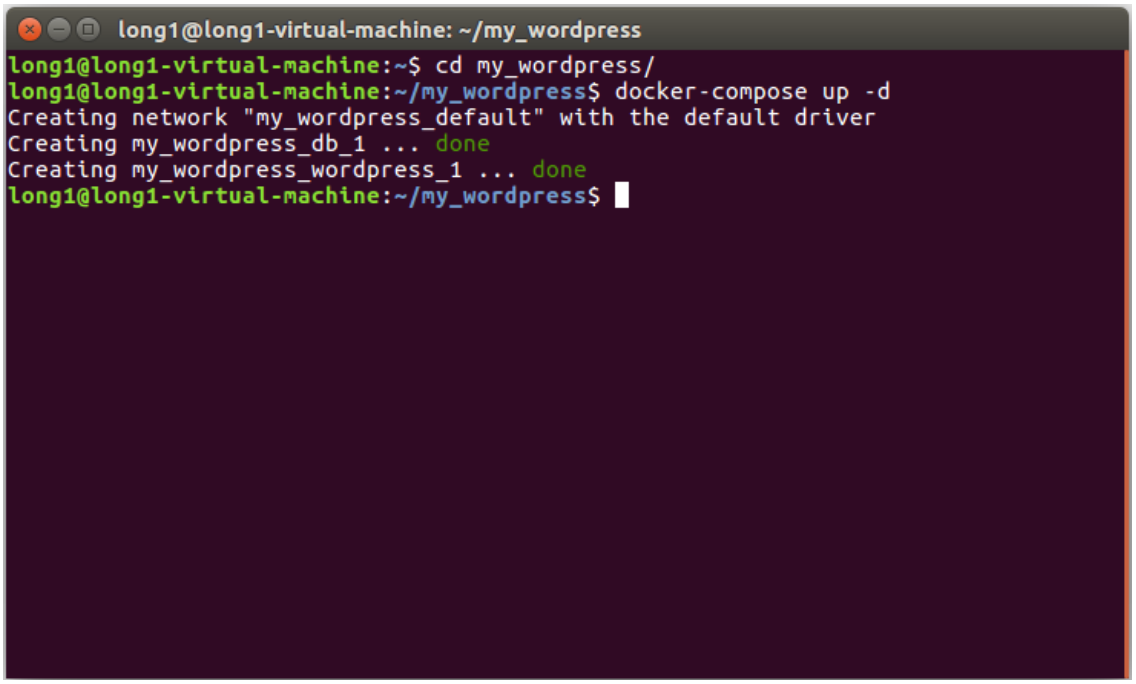
```
  image: mysql:5.7
  volumes:
  - db_data:/var/lib/mysql
  restart: always
  environment:
  MYSQL_ROOT_PASSWORD: somewordpress
  MYSQL_DATABASE: wordpress
  MYSQL_USER: wordpress
  MYSQL_PASSWORD: wordpress

 wordpress:
  depends_on:
  - db
  image: wordpress:latest
  ports:
  - "8000:80"
  restart: always
  environment:
  WORDPRESS_DB_HOST: db:3306
  WORDPRESS_DB_USER: wordpress
  WORDPRESS_DB_PASSWORD: wordpress
 volumes:
  db_data:
```

Bước 4: Vào Command line nhập:

```
cd my_wordpress/
```

Bước 5: Nhập lệnh để compose:

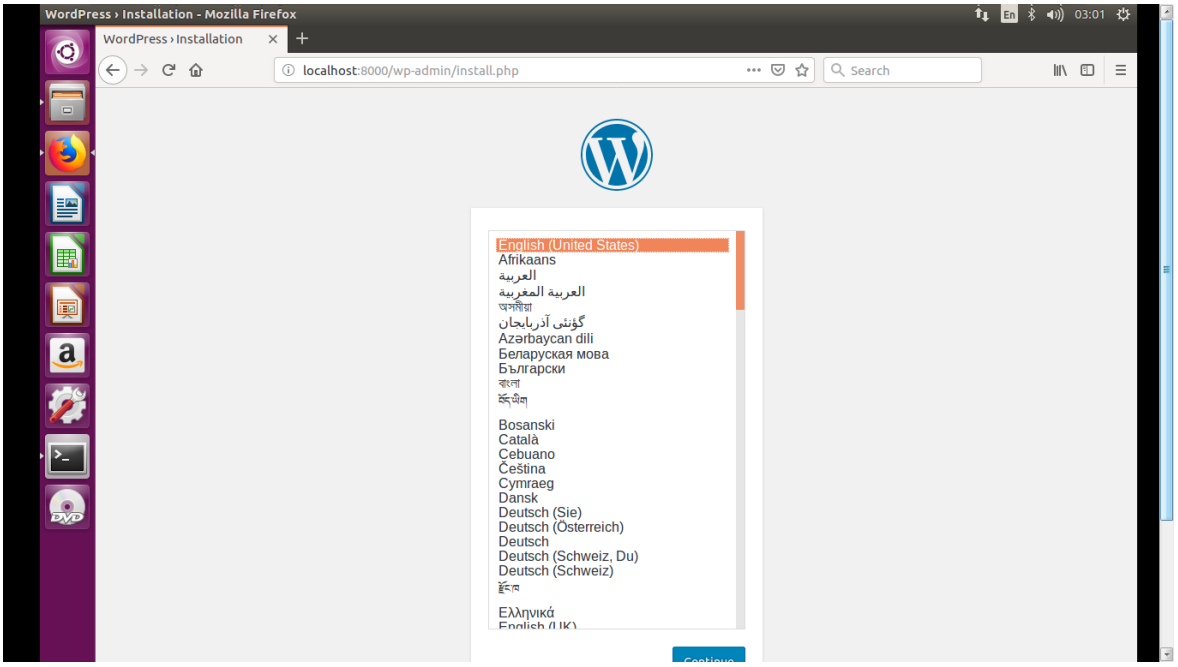
```
docker-compose up –d
```



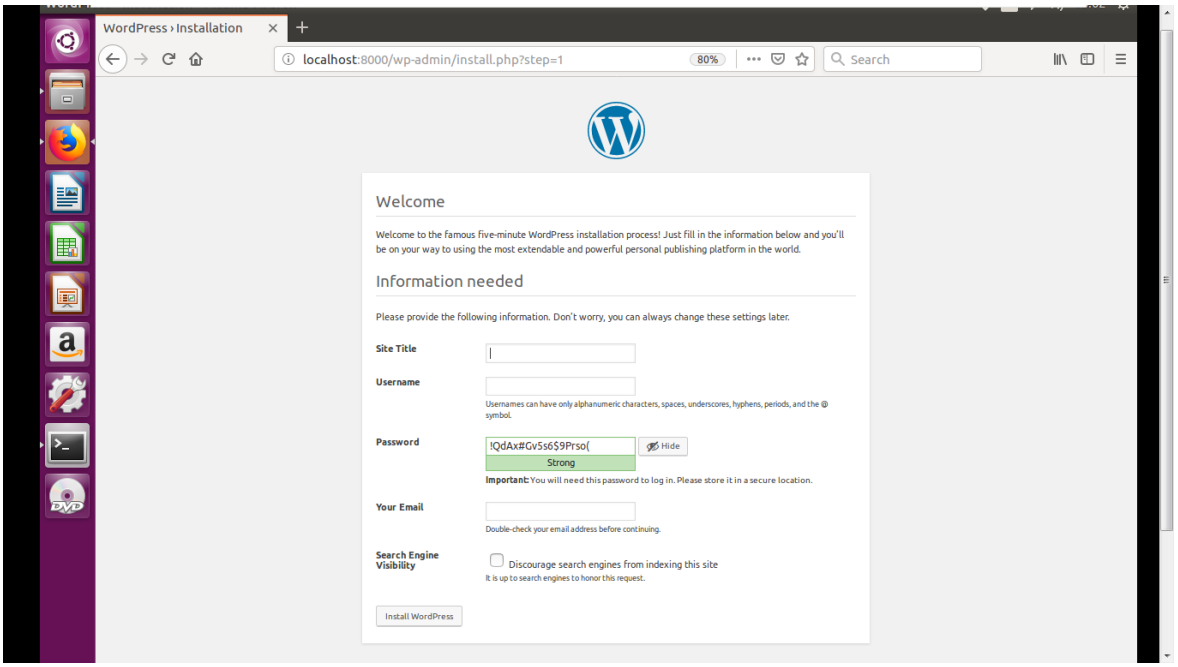

**Hình 3-5 Cài đặt wordpress**

Bước 6: Vào trình duyệt bất kỳ nhập localhost:8000

Kết quả




**Hình 3-6 Màn khởi động wordpress**




**Hình 3-7 Màn thông tin của trang chuẩn bị tạo**

# KẾT LUẬN

Trong khuôn khổ đồ án em đã trình bày một số những khái niệm cơ bản về docker, các hướng triển khai và một số bài thử nghiệm đã được giới thiệu

Đồ án tập trung vào viêc tìm hiểu và ứng dụng về công nghệ ảo hóa Docker, Tạo và push một image cùng với đó là triển khai một image là wordpress và chương trình làm việc ổn định

Kết quả đạt được của đồ án

Đồ án đã đạt được các kết quả như sau

- Tìm hiểu về khái niệm ảo hóa
- Tổng quan được về ảo hóa Docker
- Cài đặt Docker
- Tạo một image
- Push một image lên hub.docker.com
- Thiết lập được ứng dụng web bằng docker

Hạn chế của đồ án

- Các dữ liệu còn giả định
- Một số thông tin có thể chưa chính xác
- Chưa triển khai ứng dụng được rõ ràng hơn

Hướng phát triển

- Nghiên cứu sâu hơn về Docker áp dụng cho các mục đích khác nhau: nghiên cứu, phát triển ứng dụng, thương mại, …

# TÀI LIỆU THAM KHẢO

**A: Tiếng Việt**

[1.] Trần Văn Đoàn (2013), Luận văn thạc sỹ, Công nghệ ảo hoá và ứng dụng, Học viện công nghệ bưu chính viễn thông.

[2.] Trần Hải Phương (2015), Luận văn thạc sỹ, Nghiên cứu công nghệ ảo hóa và ứng dụng xây dựng hệ thống thông tin doanh nghiệp, Viện Đại học mở Hà Nội.

[3.] Vũ Trọng Chiến (2017), Luận văn thạc sỹ, Công nghệ ảo hóa docker và ứng dụng tại Đại Học Dân Lập Hải Phòng

**B: Tiếng Anh**

[4.] [5]https://medium.com/the-andela-way/docker-for-beginners-61e8e0ce6a19

**C: Internet**

[5.] https://docker-curriculum.com/

[6.] https://fullstackstation.com/docker-la-gi/

[7.] https://jobs.evolable.asia/eva-topics/huong-dan-su-dung-docker-co-ban/

[8.] https://techblog.vn/docker-chua-biet-gi-den-biet-dung-phan-1

[9.] https://techmaster.vn/khoa-hoc/25561/docker-vidu-thuc-te

[10.] http://luanvan.net.vn/luan-van/luan-van-tim-hieu-giai-phap-ao-hoa-cua-vmware-va-trien-khai-data-center-tren-nen-esx-server-28930/

[11.] https://luanbn.wordpress.com/2015/08/26/docker-part1-docker-containers-hypervisor/

[12.] https://vi.wikipedia.org/wiki/Docker_(phần_mềm)