ĐẠI HỌC QUỐC GIA TP.HCM
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

NT132.P12.ANTT

TÌM HIỂU VÀ TRIỂN KHAI CÔNG CỤ ẢO HOÁ KVM

Nhóm 6

NỘI DUNG

1. Tổng quan
2. Chi tiết đề tài
3. So sánh KVM và VMWare
4. Mô hình, kịch bản triển khai
5. Tổng kết và hướng phát triển

1. TỔNG QUAN

Xu hướng:

- **Ảo hóa** là công nghệ quan trọng trong trung tâm dữ liệu và điện toán đám mây, mang lại sự linh hoạt, hiệu quả và bảo mật.

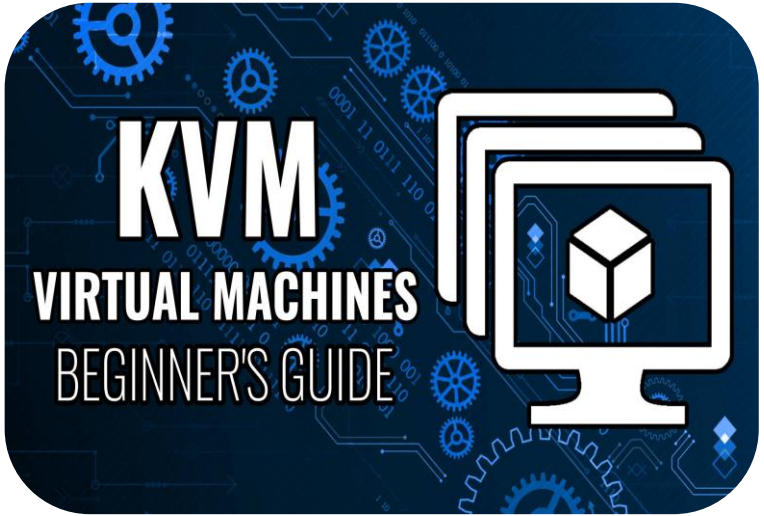
- Các công nghệ phổ biến bao gồm: VMware, Microsoft Hyper-V, Xen và KVM

1. TỔNG QUAN

Phạm vi đề tài:

- Hiểu về khái niệm và cơ chế hoạt động của **KVM** (Kernel-based Virtual Machine).

- Nắm vững các khái niệm cơ bản như ảo hóa phần cứng, hypervisor, và cách KVM tận dụng các công nghệ



2. CHI TIẾT ĐỀ TÀI

Giới thiệu về KVM:

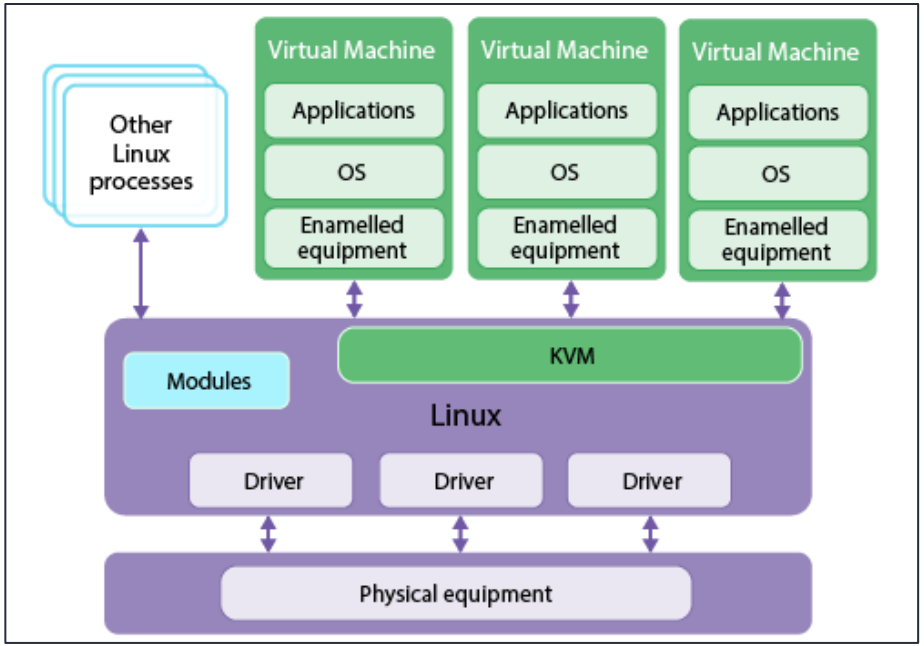
- KVM (Kernel-based Virtual Machine) là một giải pháp ảo hóa mã nguồn mở, tích hợp trực tiếp vào nhân Linux, ra mắt năm 2007.

- **KVM** giúp chuyển đổi kernel Linux thành một nền tảng quản lý và triển khai máy ảo, nơi mà mỗi máy ảo có thể hoạt động độc lập như một hệ thống hoàn chỉnh.

- Các máy ảo này có thể chia sẻ tài nguyên phần cứng như CPU, RAM, ổ cứng và mạng với các hệ điều hành khác chạy trên cùng một máy chủ vật lý, nhưng vẫn đảm bảo sự cô lập giữa chúng.

2. CHI TIẾT ĐỀ TÀI

Giới thiệu về KVM:



2. CHI TIẾT ĐỀ TÀI

Hypervisor:

- **Type 1:** Chạy trực tiếp trên phần cứng, hiệu suất cao.
  VD: KVM, VMware ESXi, Hyper-V …

- **Type 2:** Chạy trên hệ điều hành chủ, dễ sử dụng nhưng hiệu suất thấp hơn.
  VD: VMware Workstation, VirtualBox …

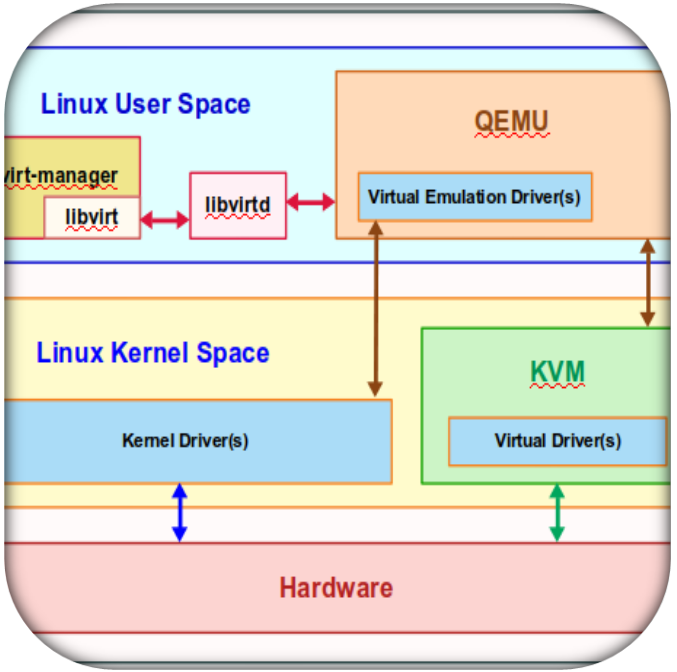
2. CHI TIẾT ĐỀ TÀI

Hypervisor:

- **KVM là hypervisor loại 1** tích hợp trong Linux, tận dụng công nghệ ảo hóa phần cứng (Intel VT-x, AMD-V) để cải thiện hiệu suất và độ ổn định.

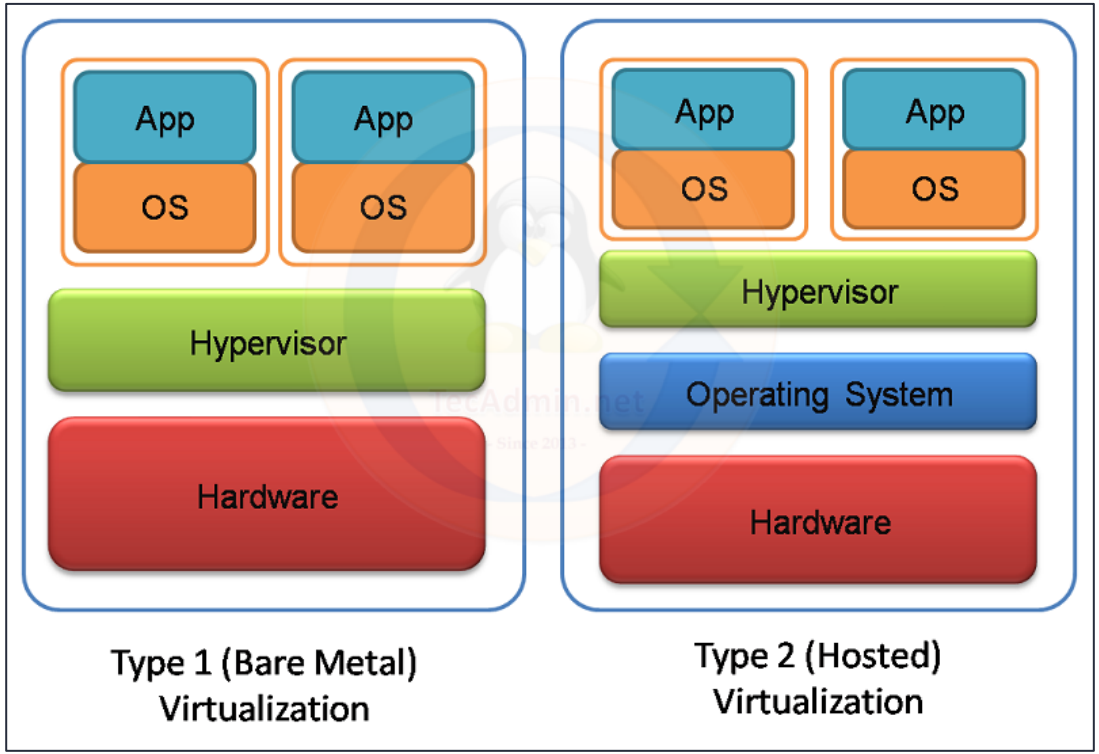
- **KVM kết hợp** với:
  + **QEMU** (giả lập phần cứng)
  + **libvirt** (quản lý tài nguyên)
  + **Virt-Manager** (giao diện đồ họa)

⇒ Quản lý máy ảo hiệu quả hơn.



2. CHI TIẾT ĐỀ TÀI

Hypervisor:



2. CHI TIẾT ĐỀ TÀI

Ưu điểm của KVM:

- Tích hợp trực tiếp vào nhân Linux, tận dụng công nghệ ảo hóa phần cứng, **đạt hiệu suất cao** gần tương đương hệ thống vật lý (bare-metal).

- Phát hành theo giấy phép GNU GPL, **không phí bản quyền**, linh hoạt và tùy chỉnh dễ dàng.

- **Triển khai linh hoạt với nhiều OS** như: Linux, Windows, macOS, FreeBSD, Solaris… trên cùng một nền tảng.

2. CHI TIẾT ĐỀ TÀI

Tính năng nổi bật:

- **Ảo hóa tốt phần cứng**, tối ưu cho việc tạo và quản lý nhiều máy ảo.

- **Hiệu suất I/O được tối ưu hoá**, phân bổ tài nguyên động tốt.

- **Hỗ trợ mạng ảo hóa**, tích hợp với các công cụ quản lý hệ thống.

- **Bảo mật và linh hoạt**.

2. CHI TIẾT ĐỀ TÀI

Ứng dụng:

- Triển khai hạ tầng mạng

- Hỗ trợ phát triển và thử nghiệm

- Quản lý mạng nâng cao

3. SO SÁNH KVM VÀ VMWARE

Thành phần Benchmark:

- So sánh hiệu năng giữa hai phương pháp ảo hóa:
    + KVM: Chạy trên Linux (DualBoot, không có lớp trung gian)
    + VMware: Chạy trên Windows (sử dụng hệ điều hành chủ)

- Các tiêu chí đánh giá: Hiệu năng **CPU**, **RAM**, **Disk I/O**, **Network I/O.**

3. SO SÁNH KVM VÀ VMWARE

Môi trường thử nghiệm:

- Phần cứng: **Laptop Lenovo Gaming LOQ 2024, CPU Intel Core i5-12450HX, RAM 24GB DDR5, SSD NVMe 512GB.**

CPU	Intel Core i5-12450HX – 8 nhân (4P-core, 4E-core), 12 luồng
RAM	24GB DDR5
Lưu trữ	512GB SSD NVMe PCIe
GPU	Nvidia RTX 3050 6GB
Chuẩn Wifi	Wifi 6
CPU	Intel Core i5-12450HX – 8 nhân (4P-core, 4E-core), 12 luồng
Điều kiện thí nghiệm	Nhiệt độ phòng được duy trì ổn định ở mức 26°C. Sau mỗi kịch bản thí nghiệm, hệ thống được nghỉ 30 phút để hạ nhiệt.

3. SO SÁNH KVM VÀ VMWARE

Môi trường thử nghiệm:

- **Phần mềm:**

+ KVM: Chạy trên Kali Linux 2024.3, sử dụng máy ảo Ubuntu Server 22.04 dùng 1 Processor, 4 cores CPU, 4GB RAM, 64GB Disk

+ VMware Workstation Pro: Chạy trên Windows 11 Home 23H2, sử dụng máy ảo Ubuntu Server 22.04 dùng 1 Processor, 4 cores CPU, 4GB RAM, 64GB Disk

3. SO SÁNH KVM VÀ VMWARE

Thực nghiệm:

Kịch bản 1: Truy vấn cơ sở dữ liệu lớn (100.000 - 1 triệu bản ghi)

- **Về đối tượng:**

+ Thực hiện truy vấn trên 1 Database gồm 3 bảng là staffs (1 triệu bảng ghi), products (300 000 bảng ghi) và partner (100 000 bảng ghi).

+ Trong đó, staffs.id sẽ là khoá chính của bảng sfatfs, đồng thời cũng được 2 bảng products (poducts.manager_id) và parner (parner.manager_id) tham chiếu đến.

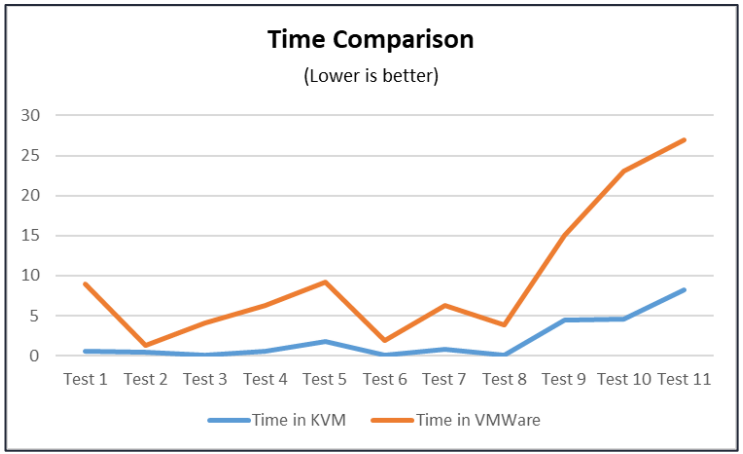
- **Về các bài test:** gồm 11 bài, độ phức tạp tăng dần từ những câu truy vấn đơn giản như hiển thị bảng, cho đến các phép toán logic như giao, hợp, kết, điều kiện,...

3. SO SÁNH KVM VÀ VMWARE

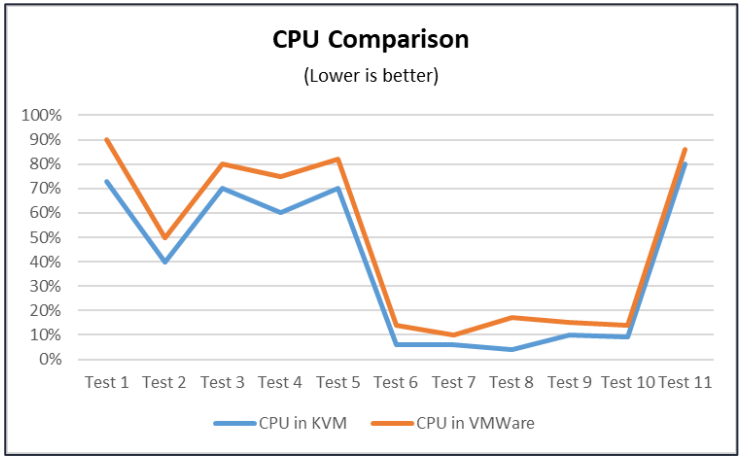
Thực nghiệm:

Kịch bản 1: Truy vấn cơ sở dữ liệu lớn (100.000 - 1 triệu bản ghi)

Thời gian xử lí:



Hiệu suất CPU:

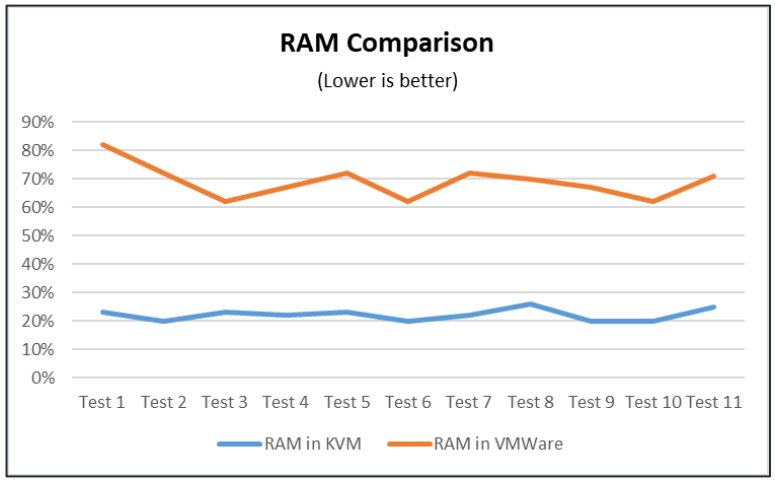


3. SO SÁNH KVM VÀ VMWARE

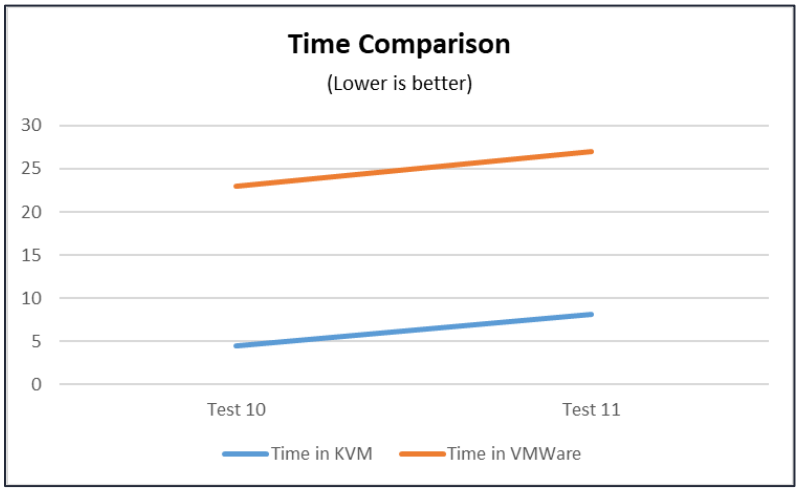
Thực nghiệm:

Kịch bản 1: Truy vấn cơ sở dữ liệu lớn (100.000 - 1 triệu bản ghi)

Hiệu suất RAM:



Tăng trưởng thời gian ở các bài kiểm tra phức tạp:



3. SO SÁNH KVM VÀ VMWARE

Thực nghiệm:

Kịch bản 1: Truy vấn cơ sở dữ liệu lớn (100.000 - 1 triệu bản ghi)

- **Kết luận:**

+ **KVM:** Là lựa chọn ưu việt cho các tác vụ xử lý nặng như truy vấn cơ sở dữ liệu phức tạp, nhờ khả năng tận dụng trực tiếp và hiệu quả tài nguyên phần cứng. Tuy nhiên, việc triển khai Dual-boot trên máy thật chạy Linux có độ phức tạp cao hơn, đòi hỏi kiến thức về hệ thống và khả năng xử lý sự cố.

+ **VMware:** Dễ dàng cài đặt và sử dụng hơn trên Windows, phù hợp với các tác vụ không yêu cầu hiệu năng cao, nhưng tiêu tốn tài nguyên hơn do phụ thuộc vào lớp trung gian của phần mềm ảo hóa

3. SO SÁNH KVM VÀ VMWARE

Thực nghiệm:

Kịch bản 2: Sử dụng SSH và build ứng dụng web

- Về đối tượng: Ứng dụng được sử dụng trong kịch bản này là llama-chat, một ứng dụng web mã nguồn mở được clone từ GitHub (dung lượng: 4.17MB).

- Về các bài test: Kịch bản được lặp lại 3 lần, mỗi lần bao gồm hai bài test:

+ Test x.1: Cài đặt các dependency của ứng dụng thông qua lệnh npm install.
+ Test x.2: Khởi chạy ứng dụng bằng lệnh npm run dev.

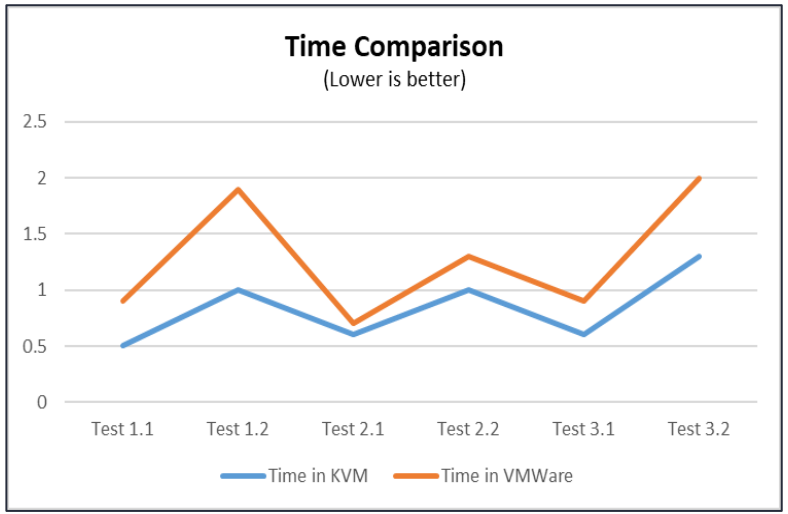
(với x là lần lặp thứ i với $i \in (1,2,3)$)

3. SO SÁNH KVM VÀ VMWARE

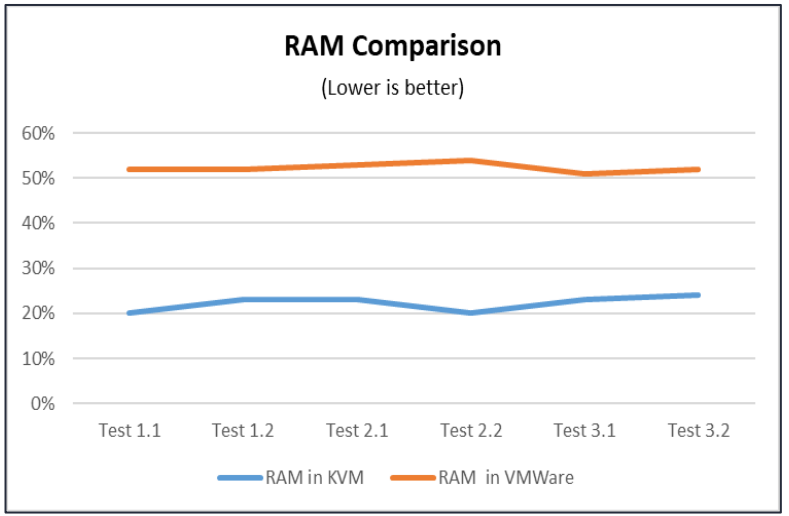
Thực nghiệm:

Kịch bản 2: Sử dụng SSH và build ứng dụng web

Thời gian xử lí:



Hiệu suất CPU:

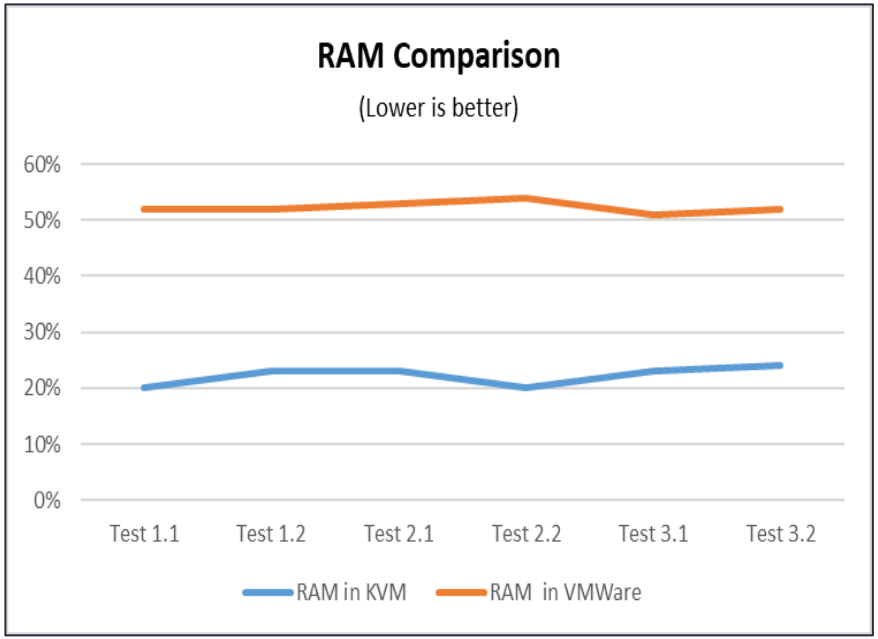


3. SO SÁNH KVM VÀ VMWARE

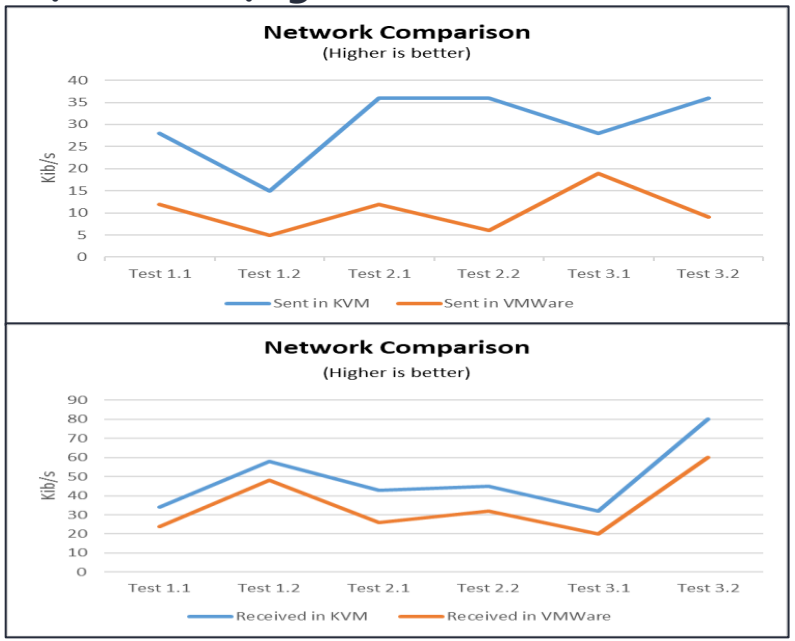
Thực nghiệm:

Kịch bản 2: Sử dụng SSH và build ứng dụng web

Hiệu suất RAM



Hiệu suất mạng:



3. SO SÁNH KVM VÀ VMWARE

Thực nghiệm:

Kịch bản 2: Sử dụng SSH và build ứng dụng web

- **Kết luận:**

+ KVM tỏ ra vượt trội với hiệu suất CPU, RAM và mạng ổn định, giúp giảm thời gian thực hiện các tác vụ.

+ VMWare mặc dù linh hoạt hơn trong việc quản lý nhiều môi trường, nhưng hiệu suất bị ảnh hưởng đáng kể bởi lớp ảo hóa.

+ Đối với các tác vụ yêu cầu tốc độ và hiệu quả tài nguyên cao, KVMS là lựa chọn ưu tiên. Ngược lại, VMWare phù hợp với các nhu cầu thử nghiệm nhiều hệ điều hành hoặc môi trường khác nhau

4. MÔ HÌNH, KỊCH BẢN TRIỂN KHAI

Kịch bản 1: Tạo máy ảo thông qua GUI Virt Manager

- Virt Manager là công cụ giao diện đồ họa thân thiện, hỗ trợ quản lý máy ảo trên Linux.

- <u>Các bước triển khai:</u>

+ Bước 1: Chuẩn bị máy chủ vật lý
+ Bước 2: Cài đặt KVM và các công cụ hỗ trợ
+ Bước 3: Tải file ISO Ubuntu Server cho máy ảo
+ Bước 4: Tạo máy ảo mới bằng GUI của Virt Manager

4. MÔ HÌNH, KỊCH BẢN TRIỂN KHAI

Kịch bản 1: Tạo máy ảo thông qua GUI Virt Manager

- **Kết quả:** Sau khi hoàn tất các bước trên, chúng ta sẽ có một máy ảo chạy hệ điều hành Ubuntu Server.
    + Máy ảo hoạt động ổn định với cấu hình đúng như yêu cầu
    + Dễ dàng quản lý qua giao diện GUI của Virt Manager

- **Ứng dụng:**
    + Ứng dụng trong phát triển phần mềm
    + Mô phỏng hạ tầng Network và IT
    + Ứng dụng trong học tập và nghiên cứu
    + Ứng dụng trong doanh nghiệp

DEMO
KỊCH BẢN 1
Tạo máy ảo thông qua GUI Virt Manager

4. MÔ HÌNH, KỊCH BẢN TRIỂN KHAI

Kịch bản 2: Tạo máy ảo có User Data thông qua CLI Virt Manager

- Sử dụng script Bash để tự động hóa quá trình tạo máy ảo với thông tin cấu hình User Data và Metadata.

- **Mô hình bao gồm:** Máy thật chạy hệ điều hành Kali Linux, có sẵn KVM và libvirt được config. Máy ảo tạo dựa trên 2 tệp ISO cùng các tệp metadata và userdata.

4. MÔ HÌNH, KỊCH BẢN TRIỂN KHAI

Kịch bản 2: Tạo máy ảo có User Data thông qua CLI Virt Manager

Các bước chính bao gồm 2 phần:

- Phần 1: Tạo seed ISO
    + Tạo thư mục và nhập thông tin user
    + Sinh tệp user-data, sinh tệp meta data
    + Tạo file ISO
- Phần 2: Tạo máy ảo
    + Thu thập thông tin máy ảo
    + Tạo ổ đĩa ảo và khởi động máy ảo
    + Mở virt manager để quản lý máy ảo

4. MÔ HÌNH, KỊCH BẢN TRIỂN KHAI

Kịch bản 2: Tạo máy ảo có User Data thông qua CLI Virt Manager

- Ứng dụng:

+ Tự động hóa cấu hình ban đầu => tránh sai sót và rủi ro
+ Tiện lợi trong quá trình quản lý nhiều máy ảo
+ Ứng dụng trong DevOps và CI/CD
+ Tăng cường tính bảo mật và đồng nhất
+ Triển khai theo nhu cầu

DEMO
KỊCH BẢN 2
Tạo máy ảo có User Data thông qua CLI Virt Manager

4. MÔ HÌNH, KỊCH BẢN TRIỂN KHAI

Kịch bản 3: SSH giữa 2 máy ảo trong KVM

- Tạo ra 2 máy ảo Ubuntu, mô phỏng tình huống khi dung máy ảo 1 để kết nối đến máy ảo 2 và triển khai, thực hiện cấu hình trên đó

- Các bước triển khai:

+ Bước 1: Tạo 2 máy ảo, cấu hình mạng và thông số cho 2 máy ảo
+ Bước 2: Khởi động máy ảo và cài đặt hệ điều hành Ubuntu 22.04
+ Bước 3: Cài đặt SSH trên 2 máy ảo đến kết nối với nhau
+ Bước 4: Kết nối máy ảo 2 đến máy ảo 1
+ Bước 5: Xác minh kết nối
+ Bước 6: Thực hiện clone và build 1 project trên github

4. MÔ HÌNH, KỊCH BẢN TRIỂN KHAI

Kịch bản 3: SSH giữa 2 máy ảo trong KVM

- **Kết quả:**

    + 2 máy ảo có thể connect SSH mà không gặp vấn đề về mạng

    + Build thành công dự án github trên máy ảo 1 thông qua việc SSH từ máy ảo 1 đến máy ảo 2.

- **Ứng dụng:**

    + Mô phỏng môi trường phát triển phần mềm phân tán

    + Quy trình DevOps cơ bản

    + Phân quyền và bảo mật trong hệ thống phân tán

    + Tối ưu hóa tài nguyên máy chủ ảo hóa

    + Kiểm tra tính khả thi của dự án trước khi triển khai

DEMO
KỊCH BẢN 3
SSH giữa 2 máy ảo trong KVM

4. MÔ HÌNH, KỊCH BẢN TRIỂN KHAI

Kịch bản 4: Triển khai CSDL MySQL và Backup data CSDL

- Mô hình triển khai mô hình gồm 1 máy chủ là **Database Server** là nơi lưu trữ chính, cho phép client thực hiện Crate, Read, Update, Delete. Máy ảo còn lại là **Backup Server**, chỉ có quyền Read và liên tục sao lưu dữ liệu.

- **Triển khai:**

  + Bước 1: Tạo chứng chỉ SSL cho MySQL
  + Bước 2: Thêm quyền cho các chứng chỉ SSL được tạo
  + Bước 3: Thực hiện cấu hình MySQL
  + Bước 4: Tạo người dung cho Backup Server chỉ cấp quyền thực hiệu query
  + Bước 5: Thực hiện lưu các chứng chỉ từ Database Server về Backup Server
  + Bước 6: Thực hiện kết nối vào MySQL bằng SSL/TLS trên Backup Server
  + Bước 7: Tạo file .sh để config thông tin của Database Server
  + Bước 8: Vào Cron table và thêm các dòng cấu hình

4. MÔ HÌNH, KỊCH BẢN TRIỂN KHAI

Kịch bản 4: Triển khai CSDL MySQL và Backup data CSDL

Kết quả:

- Từ phía Database Server:
    + CSDL luôn sẵn sàng phục vụ client với các thao tác CRUD
    + Backup Server có thể kết nối và truy xuất dữ liệu không gặp lỗi
    + Kết nối giữa Database Server và các user được mã hóa
    + Có hệ thống phân quyền và giới hạn IP => không có người dung trái phép truy cập
- Từ phía Backup Server:
    + Lấy toàn bộ dữ liệu từ Database Server và lưu dưới dạng .zip
    + File backup được đặt trong thư mục mặc định với tên rõ rang
    + Không thể thay đổi dữ liệu trên Database Server

4. MÔ HÌNH, KỊCH BẢN TRIỂN KHAI

Kịch bản 4: Triển khai CSDL MySQL và Backup data CSDL

- **Ứng dụng:**
  + Đảm Bảo Tính Sẵn Sàng Của Dữ Liệu
  + Hỗ trợ khôi phục dữ liệu nhanh chóng trong nhiều trường hợp
  + Tăng Cường Bảo Mật Dữ Liệu
  + Hỗ Trợ Phân Tích Dữ Liệu
  + Phục Vụ Sao Lưu Dài Hạn

DEMO
KỊCH BẢN 4
Triển khai CSDL MySQL và sao lưu dữ liệu

4. MÔ HÌNH, KỊCH BẢN TRIỂN KHAI

Kịch bản 5: Xây dựng mạng nội bộ gồm WebServer, DataServer, Clients

- **Web Server:** Một máy ảo KVM chạy hệ điều hành Ubuntu 20.04. Chịu trách nhiệm cung cấp dịch vụ web cho các ứng dụng nội bộ, đảm bảo truy cập nhanh chóng và bảo mật.

- **Data Server:** Một máy ảo KVM chạy hệ điều hành Ubuntu Server 20.04 chứa cơ sở dữ liệu nội bộ, lưu trữ thông tin và phục vụ dữ liệu cho các ứng dụng web.

- **Client:** Máy ảo KVM khác chạy hệ điều hành Ubuntu 20.04.06, có thể truy cập vào các website nội bộ để thực hiện công việc được giao.

4. MÔ HÌNH, KỊCH BẢN TRIỂN KHAI

Kịch bản 5: Xây dựng mạng nội bộ gồm WebServer, DataServer, Clients

- **Triển khai:**

+ Chuẩn bị trước khi triển khai Web Server: Thiết lập môi trường ảo Python, cài đặt các gói cần thiết Gunicorn và Whitenoise

+ Cấu hình Django Project

+ Cấu hình static files và media files

+ Cấu hình Nginx

+ Cấu hình firewall

+ Cấu hình tên miền nội bộ

+ Cập nhật lại cấu hình Nginx

+ Cấu hình HTTPS

4. MÔ HÌNH, KỊCH BẢN TRIỂN KHAI

Kịch bản 5: Xây dựng mạng nội bộ gồm WebServer, DataServer, Clients

- **Ứng dụng:**

+ Quản lý tài nguyên nội bộ: Cung cấp cổng thông tin tập trung, giúp truy cập và chia sẻ tài liệu, dữ liệu nội bộ nhanh chóng và hiệu quả.

+ Phát triển và thử nghiệm: Môi trường lý tưởng để phát triển, kiểm thử ứng dụng web, giảm rủi ro trước khi triển khai thực tế.

+ Đào tạo và học tập: Hỗ trợ thực hành cấu hình, bảo mật và vận hành hạ tầng web trong môi trường thực tế.

+ Bảo mật dữ liệu: Hạn chế truy cập từ bên ngoài, kết hợp SSL và tường lửa, đảm bảo an toàn dữ liệu tổ chức.

4. MÔ HÌNH, KỊCH BẢN TRIỂN KHAI

Kịch bản 5: Xây dựng mạng nội bộ gồm WebServer, DataServer, Clients

- **Kết quả:**

+ Người dùng truy cập ứng dụng qua tên miền nội bộ nhom6.local với giao thức HTTPS, đảm bảo an toàn dữ liệu.

+ Triển khai Database Server: Hoạt động ổn định, MySQL được triển khai, xử lý truy vấn nhanh và ổn định. Cổng 3306 được mở có kiểm soát, bảo mật qua cấu hình tường lửa chặt chẽ.

+ Web Server (Gunicorn và Nginx) giao tiếp hiệu quả với Data Server. Gunicorn và Nginx phối hợp giảm thời gian phản hồi. Data Server đáp ứng tải cao với độ trễ thấp.

+ Bảo mật: Chỉ mở các cổng cần thiết (443, 3306), tăng cường bảo vệ hệ thống.

DEMO
KỊCH BẢN 5
Xây dựng mạng nội bộ gồm Web Server, Data Server và các Client

5. TỔNG KẾT VÀ HƯỚNG PHÁT TRIỂN

Kết luận:

- KVM là một công cụ mạnh mẽ cho các môi trường ảo hóa.

- Mang lại hiệu suất cao và nhiều ứng dụng trong thực tiễn.

- Việc triển khai KVM không chỉ cải thiện hiệu quả sử dụng tài nguyên mà còn hỗ trợ tốt cho các hoạt động học tập, nghiên cứu và triển khai hệ thống thực tế.

5. TỔNG KẾT VÀ HƯỚNG PHÁT TRIỂN

Hạn chế:

- Cấu hình phức tạp.

- Yêu cầu kiến thức cao từ người quản trị.

- Cần thêm các công cụ hỗ trợ để quản lý hệ thống lớn.

5. TỔNG KẾT VÀ HƯỚNG PHÁT TRIỂN

Hướng phát triển trong tương lai:

- Cải thiện giao diện quản lý.

- Tích hợp với các công cụ container hiện đại như Docker.

- Triển khai trên quy mô lớn hơn thông qua các nền tảng quản lý tập trung như OpenStack.

Tóm lại, KVM là một giải pháp ảo hóa hiệu quả, phù hợp cho nhiều mục đích từ học tập, thử nghiệm đến triển khai hạ tầng doanh nghiệp.

THANKS FOR LISTENING !