|  | **BỘ CÔNG THƯƠNG**  **TRƯỜNG ĐẠI HỌC CÔNG NGHIỆP HÀ NỘI**  **—------------------------------------** | |
| --- | --- | --- |
| **ĐỒ ÁN TỐT NGHIỆP ĐẠI HỌC**  **NGÀNH CÔNG NGHỆ THÔNG TIN** | |
| **NGHIÊN CỨU VÀ THIẾT KẾ HỆ THỐNG ĐIỆN TOÁN ĐÁM MÂY PRIVATE CLOUD CHO NỘI BỘ SỬ DỤNG OPENSTACK** | |
| **Sinh viên:**  **Mã sinh viên:**  **CBHD:** | **Đỗ Viết Huy**  **2017604249**  **TS. Lê Thị Anh** |
| **Hà Nội - Năm 2024** | |

# LỜI CẢM ƠN

Qua thời gian 4 năm tích lũy và trau dồi kiến thức, rèn luyện kỹ năng tại trường Đại học Công nghiệp Hà Nội, em đã hoàn thành chương trình đào tạo với các học phần trên lớp. Nhiệm vụ học tập cuối cùng tại trường là thực hiện Đồ án tốt nghiệp, kết quả nghiên cứu của đề tài là thể hiện cho quá trình học tập-rèn luyện đó. Quãng thời gian ấy tuy không dài nhưng có ý nghĩa vô cùng với cá nhân em, với tình cảm sâu sắc, chân thành, cho phép em được bày tỏ lòng biết ơn sâu sắc đến gia đình và bạn bè đã luôn đồng hành, khích lệ, động viên, tạo điều kiện giúp đỡ em xuyên suốt quá trình học tập và nghiên cứu đề tài.

Em cũng xin phép gửi tới các thầy cô khoa Công nghệ thông tin trường Đại học Công nghiệp Hà Nội lời chào trân trọng, lời chúc sức khỏe và lời cảm ơn sâu sắc. Với sự quan tâm, dạy dỗ, chỉ bảo tận tình chu đáo của thầy cô, đến nay em đã có thể hoàn thành đề tài: “Nghiên cứu và thiết kế hệ thống điện toán đám mây Private Cloud cho nội bộ sử dụng Openstack”.

Đặc biệt em xin gửi lời cảm ơn chân thành nhất tới giảng viên – TS. Lê Thị Anh đã tận tình, nhiệt tình quan tâm giúp đỡ, hướng dẫn em hoàn thành đề tài này trong thời gian qua.

Em rất mong nhận được sự chỉ bảo, đóng góp ý kiến của các thầy cô để em có điều kiện bổ sung, nâng cao trình độ bản thân hơn nữa để có thể đưa kiến thức đã lĩnh hội được vào những công việc và lĩnh vực thực tế trong tương lai.

*Em xin chân thành cảm ơn!*

Đỗ Viết Huy

# LỜI NÓI ĐẦU

Trong bối cảnh công nghệ thông tin phát triển nhanh chóng, điện toán đám mây đã trở thành một trong những giải pháp hàng đầu cho việc lưu trữ, xử lý dữ liệu và cung cấp dịch vụ qua mạng. Các doanh nghiệp ngày càng có nhu cầu lớn về việc sử dụng các hệ thống đám mây riêng tư (Private Cloud) nhằm đảm bảo tính bảo mật, quyền kiểm soát và khả năng tùy chỉnh theo yêu cầu nội bộ của họ. Đặc biệt, OpenStack - một nền tảng mã nguồn mở mạnh mẽ, đã chứng minh được tính hiệu quả trong việc triển khai các hệ thống đám mây với chi phí hợp lý và tính linh hoạt cao.

Với mục tiêu nghiên cứu và ứng dụng công nghệ điện toán đám mây vào thực tiễn, đồ án này tập trung vào việc nghiên cứu và thiết kế hệ thống điện toán đám mây Private Cloud cho nội bộ sử dụng OpenStack. Mục tiêu chính của đồ án là tìm hiểu sâu về cấu trúc, chức năng của OpenStack và cách thức triển khai một hệ thống đám mây riêng tư nhằm đáp ứng các nhu cầu cụ thể của tổ chức.

Trong quá trình thực hiện, đồ án sẽ đi sâu vào các bước triển khai, từ việc cài đặt, cấu hình hệ thống, đến việc thử nghiệm tính ổn định và bảo mật của hệ thống. Hy vọng rằng qua nghiên cứu này, các tổ chức, doanh nghiệp có thể thấy được tiềm năng và lợi ích của việc sử dụng OpenStack để triển khai hệ thống Private Cloud, từ đó nâng cao hiệu quả quản lý và vận hành hệ thống thông tin.

Cuối cùng, em xin gửi lời cảm ơn chân thành tới các thầy cô trong khoa Công nghệ thông tin Trường Đại học Công Nghiệp Hà Nội đã tận tình giảng dạy và thúc đẩy em trong suốt quãng thời gian vừa qua. Giảng viên hướng dẫn đề tài – Tiến sĩ Lê Thị Anh, giảng viên Khoa Công nghệ thông tin Trường Đại học Công Nghiệp Hà Nội – đã luôn sẵn lòng giúp đỡ, đưa ra nhiều chỉ dẫn tận tình để giúp em hoàn thành được đồ án. này.

# MỤC LỤC

[**LỜI CẢM ƠN 1**](#_ob6um98j1t6w)

[**LỜI NÓI ĐẦU 2**](#_x0rgdn1g723i)

[**MỤC LỤC 3**](#_l0rckixvv789)

[**DANH MỤC CÁC CHỮ VIẾT TẮT 4**](#_swbpddgms3p9)

[**DANH MỤC BẢNG 4**](#_mit8qc9x01za)

[**DANH MỤC HÌNH ẢNH 5**](#_g1nafosr6g0u)

[**MỞ ĐẦU 6**](#_x0ajtjgsu4s0)

[1. Tên đề tài 6](#_kx84gqhqrbxx)

[2. Lý do chọn đề tài 6](#_ah3a9cyinkqx)

[3. Mục tiêu đề tài 7](#_bknbpuyrq5vz)

[4. Đối tượng và phạm vi của đề tài 8](#_fj54rat191ct)

[4.1. Đối tượng của đề tài 8](#_glkczhumvji2)

[4.2. Phạm vi của đề tài 8](#_axksnc2sfr13)

[5. Kết quả đạt được 9](#_xkjdob7yzqxw)

[6. Cấu trúc đề tài 9](#_m222hqj548k6)

[**CHƯƠNG 1: TỔNG QUAN VỀ ĐIỆN TOÁN ĐÁM MÂY VÀ NỀN TẢNG OPENSTACK 10**](#_ax1nquvc8dbb)

[1.1. TỔNG QUAN VỀ ĐIỆN TOÁN ĐÁM MÂY 10](#_c86vvl5kvbcf)

[1.1.1. Lịch sử về điện toán đám mây 10](#_6zad7lyg1fo1)

[1.1.2. Khái niệm về điện toán đám mây 11](#_ihg1bph0kwpl)

[1.1.3. Các mô hình dịch vụ trong điện toán đám mây 12](#_vuk8f76li7r8)

[1.1.4. Các mô hình triển khai điện toán đám mây 13](#_bj4ubgesfxsd)

[1.1.5. Ưu điểm và thách thức của điện toán đám mây 14](#_1iymszri57li)

[1.1.6. Ứng dụng của điện toán đám mây trong doanh nghiệp 14](#_k9qjfgk5flj0)

[1.2. NỀN TẢNG OPENSTACK 15](#_4q0ijiw759aw)

[1.2.1. Giới thiệu về Openstack 15](#_osx9wpz5f279)

[1.2.2. Kiến trúc tổng quan của Openstack 15](#_wmfnbchrqacu)

[1.2.3. Các thành phần chính của Openstack 16](#_9e5u771iwl3o)

[1.2.4. Ưu điểm của Openstack trong triển khai Private Cloud 17](#_51dap4clu2h6)

[1.2.5. So sánh Openstack với các nền tảng điện toán đám mây khác 18](#_2pyy016j6aqx)

[**1.3. Tổng kết chương 1 19**](#_bw2iygr44vxe)

[**CHƯƠNG 2: THIẾT KẾ HỆ THỐNG PRIVATE CLOUD 20**](#_9l51ohu5drn6)

[2.1. Hệ thống phần cứng hiện có 20](#_3n9xpxlgcpx)

[2.2. Bài toán quy hoạch máy chủ 21](#_shvmxmh0hz3o)

[2.2.1. Mô hình triển khai tham chiếu 21](#_ka0r6phg7gaj)

[2.2.2. Bài toán quy hoạch máy chủ 23](#_20quq0r9fr09)

[2.2.3. Triển khai Openstack trên nền tảng cơ sở hạ tầng sẵn có 29](#_le87l43znjwi)

[2.2.4. Triển khai Controller node theo mô hình PhyComp-PhyCon 30](#_pyerq0ad4060)

[2.2.5. Triển khai Compute node theo mô hình PhyComp-PhyCon 33](#_fc3t1ux9u6uq)

[2.3. Sử dụng Openstack trong quản trị hệ thống Private Cloud 35](#_in8j476y6gpq)

[**CHƯƠNG 3: TRIỂN KHAI VÀ KIỂM THỬ HỆ THỐNG 37**](#_l14m8muxzano)

[3.1. Triển khai Openstack 37](#_p61ydn9tmtio)

[3.1.1. Triển khai Controller node 37](#_39i0u87xpnea)

[3.1.2. Triển khai Compute node 58](#_hqhpdx21hfdw)

[3.2. Kiểm thử hệ thống 64](#_spde1hdjo59n)

[**KẾT LUẬN 70**](#_5gir9klsbnhh)

# DANH MỤC CÁC CHỮ VIẾT TẮT

| **STT** | **Từ viết tắt** | **Giải thích** |
| --- | --- | --- |
| 1 | CPU | Bộ xử lý trung tâm |
| 2 | RAM | Bộ nhớ tạm của máy chủ |
| 3 | Switch | Bộ chuyển mạch |
| 4 | VM | Máy ảo |
| 5 | SaaS | Software as a Service |
| 6 | PaaS | Platform as a Service |
| 7 | IaaS | Infrastructure as a Service |
| 8 | PhyComp-PhyCon | Physical Compute and Physical Controller |
| 9 | PhyComp-VirCon | Physical Compute and Virtual Controller |
| 10 | PhyComp-CoPhyCon | Physical Compute and Colocatable Physical Controller |
| 11 | Load Balancer | Cân bằng tải |
| 12 | DBaaS | Cung cấp dịch vụ database dưới dạng service |

# DANH MỤC BẢNG

[Bảng 1.1: So sánh Openstack và VMware 18](#_c9elnxp3ryjf)

[Bảng 2.1: các server và số lượng 21](#_scm6uvthwf9v)

[Bảng 2.2: cấu hình server 21](#_hc9l6ppdgk2o)

[Bảng 2.3: Các dịch vụ cài đặt trong Controller node 22](#_vwo3xy9zfmam)

[Bảng 2.4: Các dịch vụ trong Compute node 23](#_5ay45eh51ewb)

[Bảng 2.5: Các dịch vụ trong Storage node 23](#_b90h5dsewk74)

[Bảng 2.6: Bảng so sánh các mô hình quy hoạch máy chủ quy hoạch triển khai máy chủ theo mô hình PhyComp-PhyCon 28](#_irsxv62h66hw)

# DANH MỤC HÌNH ẢNH

[Hình 1.1: Các ứng dụng sử dụng SaaS 12](#_c4y0eqtb86m)

[Hình 1.2: Các nền tảng sử dụng PaaS 13](#_fmq2ymiqdjo8)

[Hình 1.3: Nền tảng sử dụng IaaS 13](#_l23jblfetm7)

[Hình 1.4: Kiến trúc cơ bản của Openstack 16](#_3fwuf7309y43)

[Hình 1.6: Các thành phần của Openstack 17](#_2d7f5azs9hm)

[Hình 2.1: Mô hình triển khai tham chiếu của Openstack 22](#_5kgj9msr9rr5)

[Hình 2.2:Mô hình PhyCom-VirtCon 24](#_gtky49580dwp)

[Hình 2.3: Mô hình Phycomp-CoPhyCon 25](#_q7q36suf9wsw)

[Hình 2.4: Mô hình PhyCon-PhyCom 27](#_6tvqbfl25t8y)

[Hình 2.5: Mô hình triển khai mạng cho Openstack theo mô hình PhyComp-PhyCon 29](#_ieoxym7ee1o3)

[Hình 2.6: Kiến trúc logic của hệ thống 30](#_aj59p6m1lwjv)

[Hình 2.7: Sơ đồ quy hoạch Controller node 30](#_vkel9oug6err)

[Hình 2.8: Các module được triển khai cho Controller node 31](#_a0nz9utzza3g)

[Hình 2.9: Sơ đồ quy trình cài đặt Compute node 34](#_barpcqn7zhes)

[Hình 2.10: Các module được triển khai cho Compute node 34](#_gw36sm87z4s3)

[Hình 2.11: Mô hình quản trị Openstack 35](#_xj4asnmwkhs0)

[Hình 3.1: Khởi tạo một project 65](#_smuvw2u4hnt5)

[Hình 3.2: Xóa một project 65](#_d1dy46of5gx7)

[Hình 3.3: Khởi tạo một role 66](#_r4bluce8am8k)

[HÌnh 3.4: Khởi tạo một người dùng 66](#_wxkdbgrnj4cc)

[Hình 3.5: Upload image lên hệ thống 67](#_32gkb739xpzu)

[Hình 3.6: Khởi tạo cấu hình cho máy ảo 68](#_jnjigqg5ootu)

# MỞ ĐẦU

## **Tên đề tài**

Đề tài “**Nghiên cứu và thiết kế hệ thống điện toán đám mây Private Cloud cho nội bộ sử dụng Openstack**”

## Lý do chọn đề tài

Trong thời đại công nghệ thông tin phát triển mạnh mẽ, dữ liệu và khả năng xử lý dữ liệu đã trở thành yếu tố quan trọng quyết định sự thành công của các tổ chức, doanh nghiệp. Điện toán đám mây, với khả năng cung cấp tài nguyên linh hoạt, hiệu quả về chi phí và tính mở rộng, đã trở thành xu hướng hàng đầu trong việc triển khai hạ tầng công nghệ thông tin. Tuy nhiên, đối với các doanh nghiệp và tổ chức có yêu cầu cao về tính bảo mật, kiểm soát dữ liệu và khả năng tùy chỉnh, việc sử dụng điện toán đám mây công cộng có thể không hoàn toàn phù hợp. Thay vào đó, hệ thống điện toán đám mây riêng tư (Private Cloud) là giải pháp lý tưởng để đáp ứng các yêu cầu này.

Một trong những nền tảng điện toán đám mây mã nguồn mở phổ biến và mạnh mẽ nhất hiện nay là OpenStack. OpenStack cung cấp giải pháp toàn diện cho việc xây dựng và quản lý hạ tầng đám mây riêng tư, với khả năng mở rộng, tùy chỉnh và tích hợp nhiều công nghệ khác nhau. Đồng thời, OpenStack không chỉ giúp giảm chi phí triển khai mà còn mang lại sự linh hoạt và tự do cho người dùng trong việc quản lý tài nguyên hệ thống.

Chính vì vậy, đề tài “Nghiên cứu và thiết kế hệ thống điện toán đám mây Private Cloud cho nội bộ sử dụng OpenStack” được chọn với những lý do chính sau:

* Nhu cầu thực tiễn của doanh nghiệp và tổ chức: Ngày càng nhiều doanh nghiệp cần một hệ thống đám mây riêng tư để đảm bảo quyền kiểm soát dữ liệu, bảo mật và khả năng tùy chỉnh theo yêu cầu cụ thể.
* OpenStack là nền tảng mã nguồn mở tiềm năng: OpenStack đã được công nhận rộng rãi với khả năng triển khai các hệ thống đám mây mạnh mẽ, dễ mở rộng và phù hợp với nhiều môi trường khác nhau. Việc nghiên cứu và áp dụng OpenStack giúp giảm chi phí đầu tư và phụ thuộc vào các nhà cung cấp dịch vụ đám mây bên ngoài.
* Thúc đẩy khả năng ứng dụng công nghệ cao: Đề tài này mang lại cơ hội để tìm hiểu, nghiên cứu và triển khai một công nghệ tiên tiến, đóng góp vào việc nâng cao trình độ chuyên môn và khả năng tiếp cận với những công nghệ mới trong lĩnh vực điện toán đám mây.
* Tiềm năng phát triển và ứng dụng trong thực tiễn: Private Cloud không chỉ là xu hướng công nghệ hiện đại mà còn có khả năng ứng dụng cao trong việc xây dựng hệ thống công nghệ thông tin nội bộ cho nhiều loại hình doanh nghiệp, từ nhỏ đến lớn, đáp ứng nhu cầu phát triển bền vững.

Với những lý do trên, đề tài này hứa hẹn sẽ mang lại giá trị cao trong nghiên cứu và ứng dụng công nghệ điện toán đám mây, đặc biệt là trong việc triển khai hệ thống đám mây riêng tư bằng OpenStack.

## Mục tiêu đề tài

Đề tài “Nghiên cứu và thiết kế hệ thống điện toán đám mây Private Cloud cho nội bộ sử dụng Openstack” đáp ứng các mục tiêu sau:

Tìm hiểu và phân loại các mô hình đám mây (Public Cloud, Private Cloud, Hybrid Cloud). Phân tích các ưu, nhược điểm của từng mô hình, đặc biệt tập trung vào Private Cloud và lý do lựa chọn mô hình này cho môi trường doanh nghiệp.

Tìm hiểu và nghiên cứu nền tảng OpenStack: Phân tích cấu trúc, chức năng và các thành phần chính của OpenStack, bao gồm Compute (Nova), Storage (Swift và Cinder), Networking (Neutron), Identity Service (Keystone), và các thành phần hỗ trợ khác.

Thiết kế và triển khai hệ thống Private Cloud cho nội bộ: Lên kế hoạch thiết kế kiến trúc hệ thống điện toán đám mây Private Cloud, phù hợp với nhu cầu và quy mô của doanh nghiệp. Tiến hành cài đặt và cấu hình các thành phần OpenStack trên hệ thống, đảm bảo hệ thống vận hành ổn định, an toàn và hiệu quả.

Tiến hành thử nghiệm hệ thống, đánh giá hiệu suất của các dịch vụ trong môi trường nội bộ.

## Đối tượng và phạm vi của đề tài

### Đối tượng của đề tài

Điện toán đám mây Private Cloud: Tập trung nghiên cứu mô hình điện toán đám mây riêng tư, với mục tiêu cung cấp các dịch vụ hạ tầng công nghệ thông tin như tài nguyên máy chủ, lưu trữ và mạng cho các doanh nghiệp mà không chia sẻ với bên ngoài.

Nền tảng OpenStack: Tìm hiểu và nghiên cứu các thành phần chính của OpenStack (như Nova, Neutron, Swift, Cinder, Keystone, v.v.) để xây dựng và quản lý hệ thống điện toán đám mây Private Cloud.

### Phạm vi của đề tài

#### Phạm vi về lý thuyết:

* Nghiên cứu tổng quan về các mô hình điện toán đám mây (Public, Private, Hybrid Cloud).
* Tìm hiểu chi tiết về kiến trúc, tính năng và các thành phần của OpenStack.
* Nghiên cứu các kỹ thuật triển khai và quản lý hệ thống Private Cloud, bao gồm các vấn đề về bảo mật, quản lý tài nguyên và tính mở rộng.

#### Phạm vi thực tiễn

* Triển khai mô hình Private Cloud sử dụng OpenStack trên một hệ thống mạng nội bộ với số lượng tài nguyên giới hạn.
* Cấu hình các dịch vụ cơ bản của OpenStack, bao gồm quản lý tính toán (Compute), lưu trữ (Storage), và mạng (Networking).
* Kiểm thử hệ thống sau khi triển khai, đánh giá hiệu suất và tính ổn định của hệ thống trong môi trường thực tế.

## Kết quả đạt được

Triển khai mô hình Private Cloud sử dụng OpenStack trên một hệ thống mạng nội bộ

## Cấu trúc đề tài

Trong báo cáo đồ án này, cấu trúc của đề tài được chia thành 5 chương:

* Chương 1: Tổng quan về điện toán đám mây và nền tảng Openstack
* Chương 3: Thiết kế hệ thống Private Cloud sử dụng Openstack
* Chương 3: Triển khai và kiểm thử hệ thống

# CHƯƠNG 1: TỔNG QUAN VỀ ĐIỆN TOÁN ĐÁM MÂY VÀ NỀN TẢNG OPENSTACK

## TỔNG QUAN VỀ ĐIỆN TOÁN ĐÁM MÂY

### Lịch sử về điện toán đám mây

Điện toán đám mây đã trải qua các giai đoạn lịch sử khác nhau từ những bước đầu sơ khai cho đến thời kỳ hiện đại với công nghệ tiên tiến và đang phát triển, đổi mới không ngừng. Sau đây là quá trình phát triển của điện toán đám mây:

#### Thời kỳ đầu 1960 - 1990

Ý tưởng ban đầu: Năm 1961, John McCarthy, một nhà khoa học máy tính người Mỹ, đã dự đoán rằng máy tính sẽ trở thành một tiện ích công cộng giống như điện và nước. Điều này đã mở ra một hướng suy nghĩ về khả năng tính toán theo yêu cầu.

Virtualization (Ảo hóa): Trong những năm 1970, IBM đã phát triển công nghệ ảo hóa để chia sẻ tài nguyên phần cứng thành nhiều máy ảo, cho phép nhiều người dùng truy cập cùng một phần cứng mà không xung đột.

Client-Server Model: Vào thập niên 1990, mô hình client-server (máy khách-máy chủ) bắt đầu trở nên phổ biến, trong đó máy khách yêu cầu và máy chủ cung cấp dịch vụ.

#### Sự ra đời của điện toán đám mây năm 2000

AWS (Amazon Web Services): Vào năm 2006, AWS đã ra mắt dịch vụ EC2 (Elastic Compute Cloud), dịch vụ cung cấp tài nguyên tính toán trên đám mây theo yêu cầu. Đây là một bước đột phá quan trọng trong việc thương mại hóa dịch vụ điện toán đám mây.

Google & Microsoft: Sau sự thành công của Amazon, các công ty lớn khác như Google và Microsoft cũng bắt đầu phát triển các dịch vụ đám mây của riêng mình. Google App Engine (2008) và Microsoft Azure (2010) đều là các nền tảng đáng chú ý trong lĩnh vực này.

#### Phát triển và mở rộng năm 2010 đến nay

SaaS, PaaS và IaaS: Các mô hình dịch vụ điện toán đám mây bao gồm SaaS (Software as a Service - Phần mềm dưới dạng dịch vụ), PaaS (Platform as a Service - Nền tảng dưới dạng dịch vụ) và IaaS (Infrastructure as a Service - Cơ sở hạ tầng dưới dạng dịch vụ) trở thành xu hướng chính.

* SaaS: Các phần mềm như Google Workspace (Google Docs, Google Drive), Microsoft 365, Salesforce đều là các ví dụ về SaaS.
* PaaS: Google App Engine và Microsoft Azure cung cấp các nền tảng phát triển ứng dụng mà không cần quản lý cơ sở hạ tầng.
* IaaS: AWS, Google Cloud, Microsoft Azure đều cung cấp dịch vụ IaaS, cho phép khách hàng thuê tài nguyên tính toán, lưu trữ, và mạng từ đám mây.

Multi Cloud và Hybrid Cloud: Các công ty bắt đầu kết hợp nhiều dịch vụ đám mây từ nhiều nhà cung cấp (multi cloud) hoặc kết hợp giữa đám mây công cộng và đám mây riêng (hybrid cloud) để tối ưu hóa chi phí và tính linh hoạt.

Edge Computing: Gần đây, xu hướng mới là tính toán tại rìa mạng (edge computing) cho phép xử lý dữ liệu gần với nguồn phát sinh dữ liệu hơn, nhằm giảm độ trễ và tải lưu lượng mạng.

### Khái niệm về điện toán đám mây

Điện toán đám mây (Cloud Computing) là mô hình cung cấp các tài nguyên công nghệ thông tin dưới dạng dịch vụ qua mạng Internet. Các tài nguyên này bao gồm máy chủ (server), lưu trữ (storage), cơ sở dữ liệu (database), mạng (networking), phần mềm (software) và các dịch vụ khác. Thay vì đầu tư, quản lý và vận hành hạ tầng phần cứng trực tiếp, các tổ chức/doanh nghiệp có thể sử dụng các dịch vụ này thông qua các nhà cung cấp đám mây, giảm bớt chi phí và phức tạp trong việc quản lý hệ thống công nghệ thông tin.

Khái niệm này đã phát triển nhanh chóng, đặc biệt trong những năm gần đây, và trở thành nền tảng cơ bản cho nhiều ứng dụng và hệ thống thông tin hiện đại.

### Các mô hình dịch vụ trong điện toán đám mây

Điện toán đám mây cung cấp ba mô hình dịch vụ chính, giúp người dùng có thể lựa chọn và triển khai theo nhu cầu

#### Software as a Service

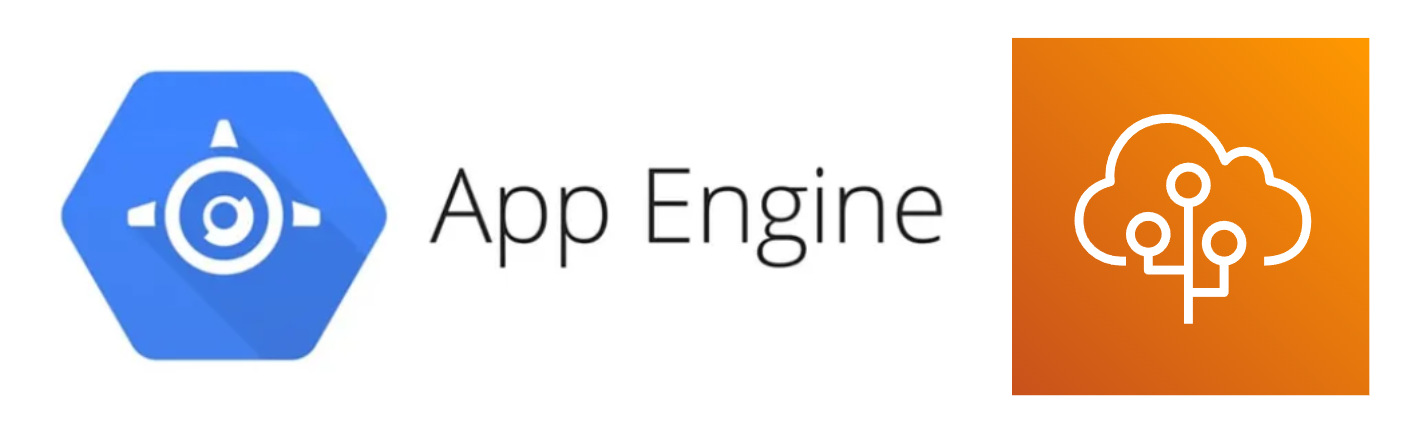
Software as a Service (SaaS): Mô hình này cung cấp các phần mềm và ứng dụng trực tiếp qua Internet, giúp người dùng có thể sử dụng các ứng dụng mà không cần phải cài đặt hay quản lý cơ sở hạ tầng. Ví dụ như Google Workspace, Microsoft 365.



###### Hình 1.1: Các ứng dụng sử dụng SaaS

#### Platform as a Service

Mô hình PaaS cung cấp một nền tảng hạ tầng để phát triển, kiểm thử và triển khai các ứng dụng mà không cần quản lý phần cứng cơ bản. Các máy ảo được cài đặt sẵn môi trường, người dùng chỉ tập trung vào phát triển ứng dụng và dịch vụ. Ví dụ như Google App Engine, AWS Elastic Beanstalk



###### Hình 1.2: Các nền tảng sử dụng PaaS

#### Infrastructure as a Service

IaaS cung cấp tài nguyên hạ tầng như máy chủ, lưu trữ và mạng qua Internet. Người dùng có thể sử dụng tài nguyên này để tự quản lý hệ điều hành, ứng dụng và dữ liệu theo ý mình. Ví dụ, Amazon Web Services (AWS EC2), Google Cloud (Compute Engine),...



###### Hình 1.3: Nền tảng sử dụng IaaS

### Các mô hình triển khai điện toán đám mây

Có ba mô hình triển khai chính trong điện toán đám mây:

* Public Cloud: Đây là mô hình đám mây công cộng, nơi tài nguyên và dịch vụ được cung cấp bởi các nhà cung cấp bên ngoài thông qua Internet. Mô hình này có chi phí thấp và khả năng mở rộng cao, nhưng khả năng kiểm soát và bảo mật dữ liệu bị hạn chế.
* Private Cloud: Đám mây riêng tư được triển khai cho một tổ chức hoặc doanh nghiệp duy nhất, cung cấp toàn quyền kiểm soát tài nguyên và dữ liệu. Private Cloud giúp đảm bảo tính bảo mật cao hơn so với Public Cloud và phù hợp với các doanh nghiệp yêu cầu bảo mật dữ liệu nội bộ.
* Hybrid Cloud: Hybrid Cloud kết hợp cả Public Cloud và Private Cloud, cho phép tổ chức tận dụng được lợi ích của cả hai mô hình. Các doanh nghiệp có thể sử dụng Private Cloud cho dữ liệu nhạy cảm và Public Cloud cho các ứng dụng hoặc dữ liệu ít quan trọng hơn.

### Ưu điểm và thách thức của điện toán đám mây

Điện toán đám mây mang lại nhiều ưu điểm nổi bật như:

* Tiết kiệm chi phí: Không cần đầu tư quá nhiều vào hạ tầng phần cứng, người dùng chỉ cần trả phí dựa trên tài nguyên đã sử dụng.
* Tính linh hoạt và mở rộng: Tài nguyên có thể dễ dàng mở rộng hoặc thu hẹp theo nhu cầu của người dùng.
* Tính khả dụng cao: Hệ thống đám mây thường đảm bảo khả năng hoạt động liên tục và có dự phòng để giảm thiểu thời gian gián đoạn.

Tuy nhiên, điện toán đám mây cũng có một số thách thức:

* Bảo mật dữ liệu: Việc lưu trữ và xử lý dữ liệu qua các dịch vụ đám mây có thể dẫn đến lo ngại về bảo mật, đặc biệt khi sử dụng Public Cloud.
* Quản lý và kiểm soát: Với một số doanh nghiệp, việc mất đi quyền kiểm soát trực tiếp hạ tầng có thể là một vấn đề lớn.

### Ứng dụng của điện toán đám mây trong doanh nghiệp

Điện toán đám mây được ứng dụng trong nhiều lĩnh vực khác nhau của doanh nghiệp, từ quản lý dữ liệu, lưu trữ thông tin, triển khai phần mềm, đến các ứng dụng thương mại điện tử và dịch vụ khách hàng. Sự linh hoạt và khả năng đáp ứng nhanh chóng các nhu cầu về tài nguyên đã giúp điện toán đám mây trở thành một phần quan trọng trong chiến lược phát triển công nghệ của các tổ chức hiện đại.

## NỀN TẢNG OPENSTACK

### Giới thiệu về Openstack

Openstack là một nền tảng phần mềm mã nguồn mở Cloud Computing, được phát triển theo mô hình Infrastructure as a Service (IaaS) quản lý tài nguyên hệ thống máy tính và cung cấp tài nguyên (các server ảo và các tài nguyên khác) cho người dùng.Nền tảng phần mềm bao gồm một nhóm các chức năng liên quan đến nhau điều khiển xử lý các nhóm phần cứng, lưu trữ và hệ thống mạng trong data center. Người sử dụng quản lý thông qua một dashboard dựa trên nền web, các công cụ dòng lệnh hoặc thông qua các API RESTful.

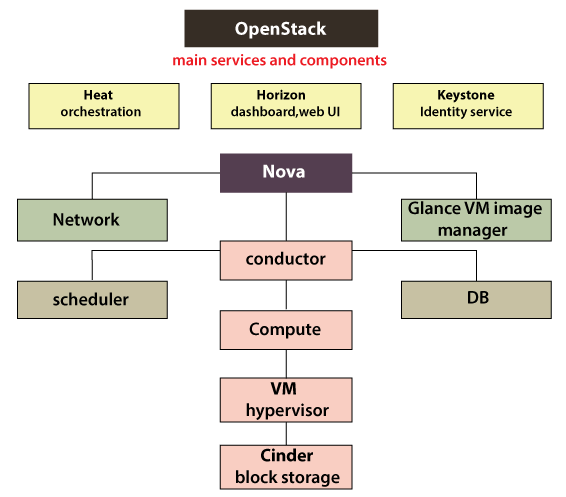
Openstack.org là đơn vị phát hành Openstack dựa theo các điều khoản của giấy phép Apache.

OpenStack là một dự án chung của Rackspace Hosting và của NASA vào năm 2010. Kể từ năm 2016, OpenStack được quản lý bởi OpenStack Foundation (một tổ chức phi lợi nhuận) được thành lập vào tháng 9 năm 2012 để quảng bá phần mềm OpenStack và cộng đồng người dùng và hơn 500 công ty đã tham gia dự án.

OpenStack được hỗ trợ bởi một cộng đồng lớn các nhà phát triển và doanh nghiệp, OpenStack cung cấp một giải pháp linh hoạt, hiệu quả và dễ dàng mở rộng cho việc quản lý tài nguyên hạ tầng như máy chủ, lưu trữ và mạng. Nền tảng này đã trở thành lựa chọn phổ biến trong việc triển khai các hệ thống Private Cloud nhờ tính mở, khả năng tùy chỉnh cao và khả năng tương thích với nhiều loại phần cứng khác nhau.

### Kiến trúc tổng quan của Openstack

Kiến trúc của OpenStack bao gồm nhiều thành phần khác nhau hoạt động đồng bộ để cung cấp các dịch vụ hạ tầng như tính toán, lưu trữ, quản lý mạng và bảo mật. Mỗi thành phần đảm nhiệm một chức năng cụ thể trong hệ thống, giúp quản lý và phân bổ tài nguyên hiệu quả hơn. OpenStack được xây dựng dựa trên kiến trúc module hóa, cho phép người dùng có thể lựa chọn và tích hợp những thành phần cần thiết cho hệ thống của mình, từ đó tăng tính linh hoạt và khả năng mở rộng.

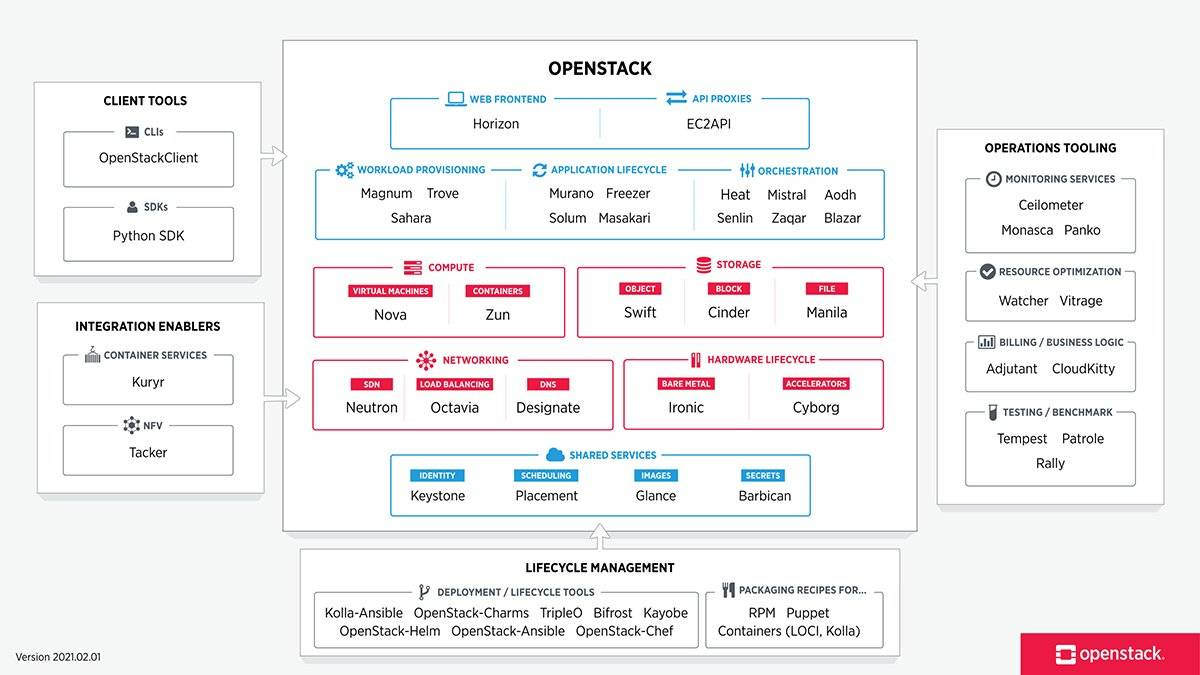


###### Hình 1.4: Kiến trúc cơ bản của Openstack

### Các thành phần chính của Openstack

OpenStack bao gồm nhiều thành phần chức năng chính, mỗi thành phần đóng một vai trò quan trọng trong việc xây dựng và vận hành hệ thống điện toán đám mây:

* Compute (Nova): Quản lý và phân phối tài nguyên tính toán, cho phép khởi tạo và quản lý máy ảo.
* Networking (Neutron): Cung cấp khả năng quản lý mạng linh hoạt, từ việc cấu hình mạng ảo đến quản lý các tài nguyên mạng phức tạp.
* Storage (Cinder): Swift hỗ trợ lưu trữ đối tượng, trong khi Cinder quản lý các khối lưu trữ, giúp đáp ứng nhu cầu lưu trữ của máy ảo và dữ liệu.
* Identity Service (Keystone): Cung cấp dịch vụ xác thực và phân quyền người dùng, đảm bảo bảo mật hệ thống và quản lý quyền truy cập hiệu quả.
* Image Service (Glance): Quản lý hình ảnh máy ảo, cho phép lưu trữ và phân phối các hình ảnh (image) cho việc khởi tạo máy ảo mới.
* Dashboard (Horizon): Giao diện web giúp người dùng và quản trị viên dễ dàng tương tác, quản lý tài nguyên và giám sát hệ thống.
* Orchestration (Heat): Quản lý và tự động hóa việc triển khai các ứng dụng và tài nguyên hạ tầng.
* Telemetry (Ceilometer): Thu thập và giám sát dữ liệu về hiệu suất và tài nguyên sử dụng trong hệ thống.



###### Hình 1.6: Các thành phần của Openstack

### Ưu điểm của Openstack trong triển khai Private Cloud

OpenStack nổi bật với một số ưu điểm vượt trội trong việc triển khai Private Cloud:

* Mã nguồn mở: OpenStack hoàn toàn miễn phí và có khả năng tùy chỉnh cao, giúp doanh nghiệp có thể xây dựng hệ thống theo yêu cầu riêng mà không bị ràng buộc bởi các nhà cung cấp dịch vụ.
* Tính mở rộng: Với kiến trúc module hóa, OpenStack dễ dàng mở rộng và tích hợp thêm các thành phần khác khi có nhu cầu tăng cường tài nguyên.
* Hỗ trợ đa dạng phần cứng: OpenStack có thể hoạt động trên nhiều loại phần cứng khác nhau, từ đó giúp tối ưu hóa chi phí đầu tư hạ tầng.
* Hỗ trợ cộng đồng: Với một cộng đồng phát triển mạnh mẽ, OpenStack luôn được cập nhật và cải thiện liên tục, cung cấp nhiều tài liệu và hỗ trợ từ cộng đồng.

### So sánh Openstack với các nền tảng điện toán đám mây khác

So với các giải pháp đám mây khác như VMware vCloud hay Microsoft Azure Stack, OpenStack có một số điểm khác biệt:

* Tính linh hoạt và tùy chỉnh: Trong khi các nền tảng khác thường bị giới hạn bởi phần cứng hoặc phần mềm của nhà cung cấp, OpenStack cung cấp tính linh hoạt cao hơn nhờ tính chất mã nguồn mở và khả năng tương thích với nhiều hệ thống khác nhau.
* Chi phí: OpenStack miễn phí và giảm đáng kể chi phí cấp phép phần mềm, trong khi các giải pháp khác như VMware vCloud có chi phí bản quyền cao.
* Phức tạp: Mặc dù OpenStack mang lại nhiều lợi ích, việc triển khai và quản lý OpenStack có thể phức tạp hơn, đòi hỏi nhiều kiến thức kỹ thuật hơn so với các nền tảng đám mây thương mại khác.

##### Bảng 1.1: So sánh Openstack và VMware

|  | Openstack | VMware |
| --- | --- | --- |
|  |  |  |
| Mô hình | Mã nguồn mở | Có chủ sở hữu |
| Kiến trúc | Cấu trúc phân tán | Cấu trúc tập trung |
| Hypervisor | Không bắt buộc | ESXi |
| Tính năng | Đa dạng, có thể tùy chỉnh | Tập trung vào tính sẵn sàng |
| Chi phí | Thấp | Cao |
| Khả năng mở rộng | Tốt | Tốt |
| Quản trị | Khó quản trị hơn | Dễ để quản trị hơn |
| Hỗ trợ | Cộng đồng | Từ VMware |

##### 

## Tổng kết chương 1

Trong chương 1, em đã trình bày cái nhìn tổng quan về khái niệm và các mô hình dịch vụ trong điện toán đám mây bao gồm SaaS, PaaS, IaaS. Điện toán đám mây đã trở thành một trong những xu hướng công nghệ quan trọng, giúp các doanh nghiệp tiết kiệm chi phí, tối ưu hóa tài nguyên và gia tăng khả năng mở rộng. Bên cạnh đó, các mô hình triển khai như Public Cloud, Private Cloud và Hybrid Cloud đã được giới thiệu, với Private Cloud đặc biệt phù hợp cho các doanh nghiệp yêu cầu cao về bảo mật và kiểm soát dữ liệu.

Ngoài các ưu điểm nổi bật của điện toán đám mây như tính linh hoạt và khả năng mở rộng, chương này cũng nêu rõ các thách thức cần phải đối mặt. Tiếp theo, em cũng cũng giới thiệu được các thành phần cơ bản của Openstack để có thể triển khai hệ thống Private Cloud. Những kiến thức cơ bản này sẽ là nền tảng quan trọng để xây dựng hệ thống Private Cloud trong chương tiếp theo.

# CHƯƠNG 2: THIẾT KẾ HỆ THỐNG PRIVATE CLOUD

Chương này sẽ trình bày cách thức triển khai hệ thống Private Cloud cho các ứng dụng đào tạo và thực hành dựa trên Openstack. Cụ thể chương này sẽ trình bày quy trình triển khai hệ thống Openstack. Trong quá trình cài đặt triển khai, các bài toán phát sinh cần được giải quyết để tối ưu hóa hệ thống gồm: Khảo sát hệ thống phần cứng sẽ triển khai, quy hoạch và thiết lập mô hình cài đặt và triển khai, quản trị hệ thống.

Hệ thống phần cứng dùng để cài đặt triển khai trên cụm ba server vật lý và mỗi server được cài đặt hệ điều hành Ubuntu 22.04 LTS. Trên mỗi máy vật lý cấu hình cho 2 card mạng gồm: một card mạng thiết lập kết nối ra Internet để quản trị viên có thể kết nối từng dịch vụ trong hệ thống và một card mạng thiết lập kết nối nội bộ (LAN) các máy với nhau thông qua các switch vật lý.

Mong muốn sau khi triển khai cài đặt Openstack, các server hoạt động với hiệu suất cao nhất do sự tối ưu tài nguyên của hệ thống mang lại. Ngoài ra, hệ thống cũng cho phép quản trị viên có thể tạo lập cơ chế quản trị một cách hiệu quả cũng như thiết lập kết nối từ bên ngoài đến từng máy ảo (hay cụm máy ảo) hoặc các kết nối các máy ảo với nhau.

## Hệ thống phần cứng hiện có

Hệ thống phần cứng dùng để triển khai Openstack gồm ba server được cài đặt hệ điều hành Ubuntu 22.04 được cấu hình sẵn để có thể kết nối tới nhau. Ngoài ra, các server này còn được cấu hình để có thể remote hay ssh với nhau. Mỗi server sẽ được mở cổng để cho phép hệ thống có thể kết nối với Internet nhằm giúp quản trị viên có thể quản trị hệ thống ngay cả khi không ở gần các server.

Để thực hiện triển khai giải pháp Private Cloud, hệ thống server được sử dụng bao gồm:

###### 

##### Bảng 2.1: các server và số lượng

| STT | Tên server | Số lượng |
| --- | --- | --- |
| 1 | Dell Inc. PowerEdge R650 | 3 |

##### 

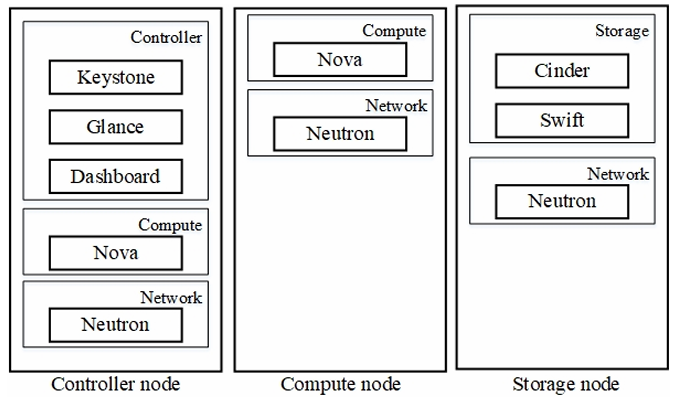
##### Bảng 2.2: cấu hình server

| STT | Tên server | CPU | Core | RAM (GB) | Storage  (GB) | Card mạng |
| --- | --- | --- | --- | --- | --- | --- |
| 1 | Dell Inc. PowerEdge R650 | Silver 4314 CPU @ 2.40GHz | 12 | 24 | 100 | 2 |
| 2 | Dell Inc. PowerEdge R650 | Silver 4314 CPU @ 2.40GHz | 2 | 4 | 30 | 2 |
| 3 | Dell Inc. PowerEdge R650 | Silver 4314 CPU @ 2.40GHz | 12 | 24 | 100 | 2 |

## Bài toán quy hoạch máy chủ

### Mô hình triển khai tham chiếu

Theo yêu cầu của giải pháp Openstack, một mô hình Private Cloud sử dụng Openstack cần có: (1) 01 Controller node; (2) nhiều Compute node; (3) có thể có thêm các Storage node hoặc các thành phần phụ trợ khác. Trong đó, Controller node chịu trách nhiệm quản lý điều phối hoạt động của các Compute node và các thành phần khác phục vụ các tác vụ quản lý máy ảo (tạo, hủy, cấu hình máy ảo), quản lý network ảo. Còn các Compute node là nơi thực hiện việc tạo các máy ảo, host các máy ảo, cung cấp tài nguyên cho các máy ảo hoạt động. Ngoài ra, Storage node chịu trách nhiệm cấp phát tài nguyên để lưu trữ (các máy ảo, file chia sẻ)



###### Hình 2.1: Mô hình triển khai tham chiếu của Openstack

Các dịch vụ chính được cài trong Controller node gồm:

##### Bảng 2.3: Các dịch vụ cài đặt trong Controller node

| **STT** | **Các module** | **Chức năng** |
| --- | --- | --- |
| 1 | Keystone | Là dịch vụ dùng để quản lý việc xác thực người dùng khi truy cập và sử dụng các dịch vụ của Openstack |
| 2 | Glance | Là dịch vụ dùng để quản lý image liên quan đến ổ đĩa ảo và cấu hình cài đặt trên các ổ đĩa ảo khi sử dụng Openstack |
| 3 | Dashboard | Là dịch vụ dùng để quản lý việc truy xuất cơ sở dữ liệu và phục vụ các tác vụ liên quan tới máy ảo |
| 4 | Nova (nova-api, nova-schedule, nova-conductor) | Là dịch vụ dùng để quản lý việc truy xuất cơ sở dữ liệu và phục vụ các tác vụ liên quan tới máy ảo |
| 5 | Neutron (neutron- server, neutron-agent) | Là dịch vụ dùng để quản lý kết nối mạng giữa Controller node với các node khác trong Openstack |

Các dịch vụ chính được cài đặt trong Compute node gồm:

##### Bảng 2.4: Các dịch vụ trong Compute node

| **STT** | **Các module** | **Chức năng** |
| --- | --- | --- |
| 1 | Nova (nova-compute) | Là dịch vụ dùng để quản lý các máy ảo cũng như cập nhật trạng thái của các máy ảo trong hệ thống Openstack |
| 2 | Neutron (neutron-agent) | Là dịch vụ dùng để quản lý kết nối mạng giữa Compute node với các node khác trong Openstack |

##### Bảng 2.5: Các dịch vụ trong Storage node

| **STT** | **Các module** | **Chức năng** |
| --- | --- | --- |
| 1 | Cinder | Là dịch vụ dùng để quản lý các thiết bị lưu trữ khối và cung cấp cho người dùng các API tự phục vụ theo yêu cầu và nhu cầu sử dụng |
| 2 | Neutron (neutron-agent) | Là dịch vụ dùng để quản lý kết nối mạng giữa Storage node với các node khác trong Openstack |

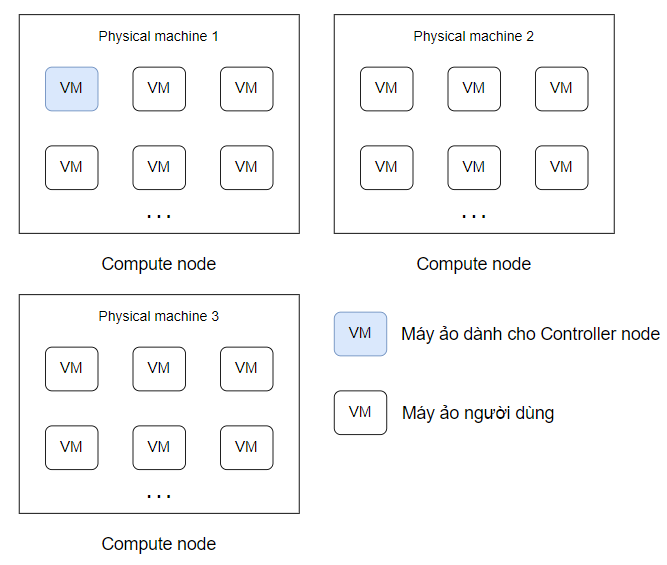
Tuy nhiên, khi triển khai hệ thống Openstack, Storage node có thể được cài đặt chung với Compute node. Đồng thời, Controller node cũng cung cấp dịch vụ như SQL Database để lưu trữ thông tin về hệ thống, MQ (đặc biệt là RabbitMQ) để trao đổi thông tin với Compute node hay NTP (Network Time Protocol) để đồng bộ thời gian giữa các máy ảo với máy host. Ngoài ra, Compute node cũng cung cấp các dịch vụ tường lửa để đảm bảo an toàn cho các máy ảo

### Bài toán quy hoạch máy chủ

Dựa theo mô hình triển khai tham chiếu đã nêu ở trên cũng như thông tin về các server nêu ở mục 2.1, ta có thể thấy rằng việc bố trí cài đặt các thành phần chức năng (controller node, compute node,…) vào server nào với hình thức và phương pháp nào là một vấn đề quan trọng ảnh hưởng tới năng lực xử lý của hệ thống và hiệu suất sử dụng tài nguyên. Để minh họa sự ảnh hưởng của việc bố trí cài đặt controller node, compute node lên server vật lý tới năng lực xử lý và hiệu suất sử dụng tài nguyên, chúng ta xem xét các mô hình triển khai thông thường sau:

#### PhyComp-VirCon

Trong mô hình này, một máy ảo đặc biệt được thiết lập trước, tách biệt tài nguyên với phần còn lại của một trong ba máy vật lý. Compute node sẽ được cài đặt trên ba máy vật lý. Trên máy ảo đặc biệt, Controller node được cài đặt. Như vậy với mô hình triển khai này, các nút chức năng sẽ được cài đặt tách biệt nhau. Mô hình này dễ cài đặt các chức năng, việc cài đặt Controller node và Compute node chỉ cần tuân thủ đúng theo hướng dẫn cài đặt của tài liệu tham khảo Openstack.

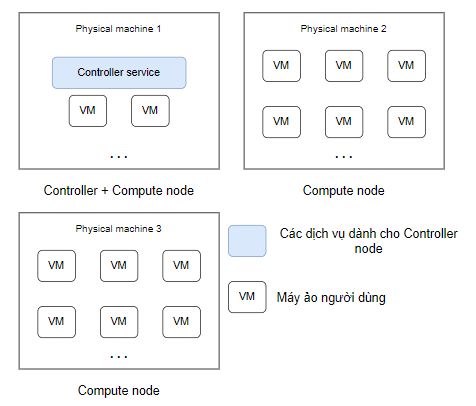


###### Hình 2.2:Mô hình PhyCom-VirtCon

Việc cài đặt tách biệt trên các máy vật lý giúp cho tài nguyên quản lý, điều phối của hệ thống Cloud (sử dụng bởi Controller node) và tài nguyên cung cấp cho các máy ảo người dùng (quản lý bởi Compute node) là tuyệt đối tách biệt, không xung đột và gây ảnh hưởng lẫn nhau. Điều đó khiến hệ thống hoạt động ổn định. Điểm khác biệt duy nhất giữa mô hình PhyComp-PhyCon và PhyComp-VirCon là Controller Node được cài lên một server ảo. Việc cấu hình tài nguyên cho máy ảo khá đơn giản, nhanh chóng và khi cần thiết có thể dễ dàng mở rộng. Khi cài đặt máy ảo, quản trị viên sẽ cài đặt máy ảo KVM trên Ubuntu 22.04, cấu hình cho KVM tối thiểu là 2 core và 4GB RAM và dùng libvirt để điều khiển máy ảo KVM. Khi cần tăng core và RAM để xử lý hệ thống, quản trị viên sẽ sửa file chứa cấu hình và khởi động lại máy ảo. Thêm vào đó, các máy vật lý được cài đặt trên các máy vật lý sẽ tận dụng toàn bộ tài nguyên phần cứng hiện có (CPU, RAM, Core và Storage).

#### Mô hình PhyComp-CoPhyCon

Trong mô hình này, các nút chức năng được triển khai trên cùng một máy vật lý. Trong số các máy vật lý có duy nhất một máy được cài đặt đầy đủ các chức năng. Như vậy với mô hình triển khai này, trong ba máy vật lý hiện có thì một máy sẽ được triển khai đồng thời Controller node và Compute node, hai máy còn lại sẽ chỉ được cài đặt Compute node.



###### Hình 2.3: Mô hình Phycomp-CoPhyCon

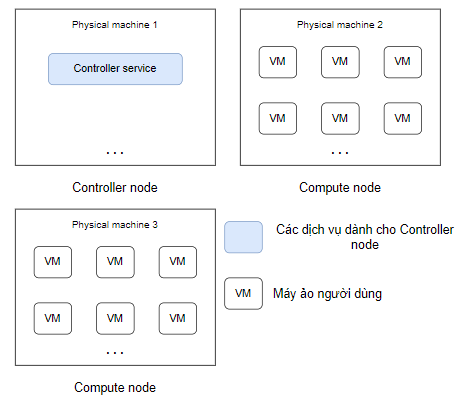
Mô hình này dễ tiếp cận trong giai đoạn đầu của việc triển khai thử nghiệm thực tế lần đầu, việc cài đặt Controller node và Compute node trên một máy sẽ không đòi hỏi nhiều trong việc chuẩn bị tài nguyên. Việc cài đặt chung các chức năng trên cùng một máy vật lý còn giúp cho hệ thống khắc phục, giải quyết được sự lãng phí về tài nguyên cung cấp cho nhu cầu xử lý (CPU, RAM, Core) và lưu trữ (của hệ thống, việc cài đặt Controller node và Compute node trên cùng một máy sẽ sử dụng tối đa tài nguyên của hệ thống.

Tuy nhiên, mô hình PhyComp-CoPhyCon là mô hình khó cấu hình, việc cấu hình Controller node và Compute node trên cùng một máy vật lý có thể dẫn tới việc xung đột. Việc cài đặt chung trên máy vật lý làm cho tài nguyên quản lý, điều phối của hệ thống Cloud (sử dụng bởi Controller node) và tài nguyên cung cấp cho các máy ảo người dùng (quản lý bởi Compute node) có thể ảnh hưởng lẫn nhau do không được tách biệt và gây ảnh hưởng tới hiệu suất hoạt động của hệ thống. Như vậy, với máy server cài chung chức năng Controller node và Compute node, hệ thống sẽ ưu tiên cấp phát tài nguyên cho Controller node nhằm đảm bảo nhu cầu hoạt động.

Tài nguyên CPU và RAM cấp phát cho các máy ảo trong Compute node sẽ bị chia sẻ dẫn tới hiệu suất hoạt động của các máy ảo bị giảm sút hoặc có thể một số máy ảo vì không được cung cấp tối thiểu CPU và RAM dẫn tới bị treo hệ thống cài đặt trên các máy ảo. Tương tự, khi số lượng máy ảo lớn cần hoạt động thì hệ thống có thể sẽ ưu tiên cấp phát tài nguyên như RAM và CPU cho các máy ảo trong Compute node hoạt động ổn định. Như vậy, tài nguyên CPU và RAM cấp phát Controller node sẽ bị chia sẻ hoặc khi cần thêm tài nguyên cho nhu cầu xử lý các tác vụ, hệ thống sẽ không cấp phát thêm làm giảm hiệu suất hoạt động của toàn bộ hệ thống.

#### Mô hình PhyComp-PhyCon

Trong mô hình này, các nút chức năng được triển khai trên các máy vật lý tách biệt nhau. Tại mỗi một máy vật lý chỉ có duy nhất một chức năng được thực hiện cài đặt. Như vậy với mô hình triển khai này, trong ba máy vật lý hiện có thì một máy sẽ được triển khai Controller node, hai máy còn lại sẽ được cài đặt Compute node.



###### Hình 2.4: Mô hình PhyCon-PhyCom

Mô hình này đơn giản, dễ cài đặt, việc cài đặt Controller node và Compute node chỉ cần tuân thủ đúng theo hướng dẫn cài đặt của tài liệu tham khảo Openstack. Việc cài đặt tách biệt trên các máy vật lý giúp cho tài nguyên quản lý, điều phối của hệ thống Cloud (sử dụng bởi Controller node) và tài nguyên cung cấp cho các máy ảo người dùng (quản lý bởi Compute node) là tuyệt đối tách biệt, không xung đột và gây ảnh hưởng lẫn nhau. Điều đó khiến hệ thống hoạt động ổn định

Tuy nhiên, nếu triển khai theo mô hình PhyComp-PhyCon, tài nguyên phần cứng của hệ thống sẽ bị lãng phí và không hiệu quả nếu ta không tìm được máy chủ vật lý có cấu hình phù hợp. Controller node sẽ được cài đặt lên server có cấu hình 2 core và 4GB RAM. Như vậy, Controller node sẽ được toàn quyền sử dụng 2 core và 4GB RAM. Ngoài ra, với cấu hình server khác tương tự, server sẽ được cài đặt một Compute node (chiếm 12 core và 24GB RAM) và khoảng 10 máy ảo (mỗi máy ảo sử dụng 1 core và 2GB RAM).

#### So sánh ba mô hình triển khai

Qua ba mô hình vừa trình bày bên trên, chúng ta thấy các thành phần chức năng được bố trí vào server và phương thức triển khai là một vấn đề quan trọng và là một thách thức trong bài toán quy hoạch máy chủ sao cho hệ thống hoạt động tối ưu về năng lực xử lý và hiệu suất sử dụng. Để giải quyết vấn đề tận dụng tối đa tài nguyên phần cứng của các máy chủ vật lý nhưng vẫn tách biệt được tài nguyên dùng cho quản lý hệ thống Cloud (Controller node) và tài nguyên cho các máy ảo người dùng (Compute node), Openstack có thêm mô hình triển khai PhyComp-VirCon trong đó Controller node được triển khai vào một máy ảo dành riêng ký sinh trên một máy vật lý có cài Compute node. Cụ thể mô hình PhyComp-VirCon được mô tả như dưới đây.

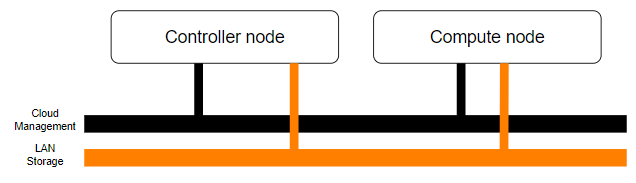
##### Bảng 2.6: Bảng so sánh các mô hình quy hoạch máy chủ quy hoạch triển khai máy chủ theo mô hình PhyComp-PhyCon

| **STT** | **Mô hình** | **Ưu điểm** | **Nhược điểm** |
| --- | --- | --- | --- |
| 1 | PhyComp - PhyCon | * Mô hình tách biệt, các node hoạt động độc lập, đơn giản, dễ cài đặt * Hệ thống sau khi triển khai hoạt động ổn định * Dễ để bảo trì hệ thống * Dễ dàng mở rộng hệ thống | * Gây sự lãng phí tài nguyên nếu không tìm được máy chủ có cấu hình yêu cầu |
| 2 | PhyComp - CoPhyCon | * Dễ tiếp cận hệ thống * Khắc phục sự lãng phí tài nguyên | * Mô hình khó cấu hình cho các node * Khi bổ sung thêm compute node mới, khó cấu hình * Khó để bảo trì hệ thống do tất cả dịch vụ chạy trên cùng một máy chủ vật lý |
| 3 | PhyComp - VirCon | * Mô hình tách biệt, các nút chức năng hoạt động độc lập. * Tối ưu tài nguyên hệ thống * Dễ dàng mở rộng hệ thống | * Mô hình phức tạp, quản trị cần có kinh nghiệm về libvirt và quản trị hệ thống * Khó khăn trong công việc quản trị và kết nối đến máy ảo |

### Triển khai Openstack trên nền tảng cơ sở hạ tầng sẵn có

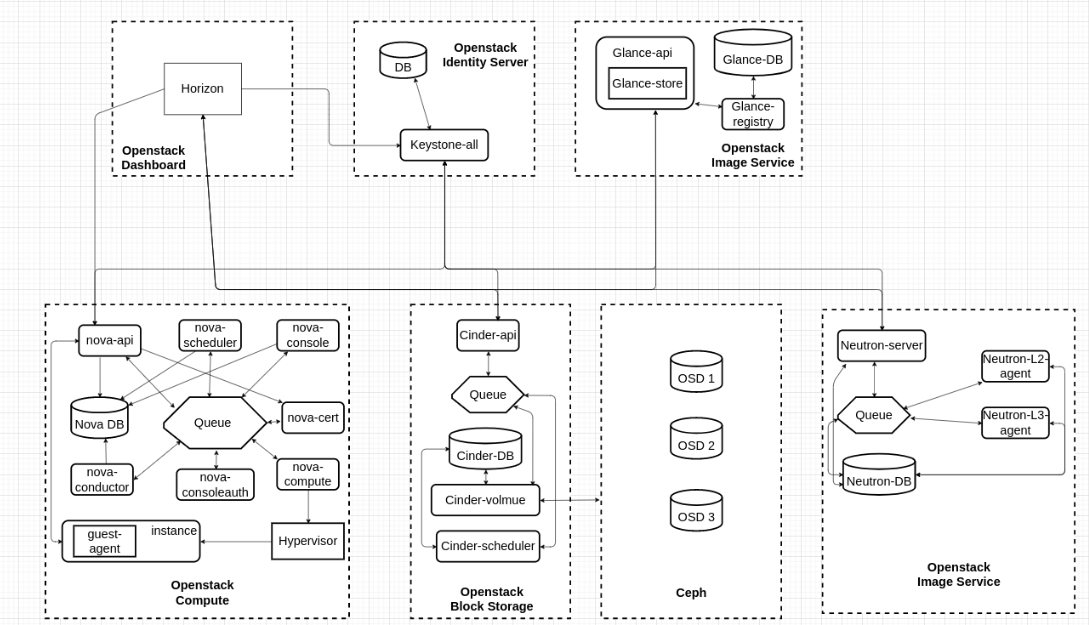
Để triển khai mô hình PhyComp-PhyCon, em triển khai mạng như hình 2.3. Cách bố trí như vậy mang lại các lợi ích như sau:

* Các dịch vụ trong máy chủ vật lý có thể sử dụng đầy đủ băng thông mạng mà không bị chiếm băng thông bởi một số dịch vụ khác (mô hình PhyComp-VirCon và PhyComp-CoPhyCon).
* Tách biệt các dịch vụ với nhau và dành tối đa tài nguyên cho các máy ảo người dùng



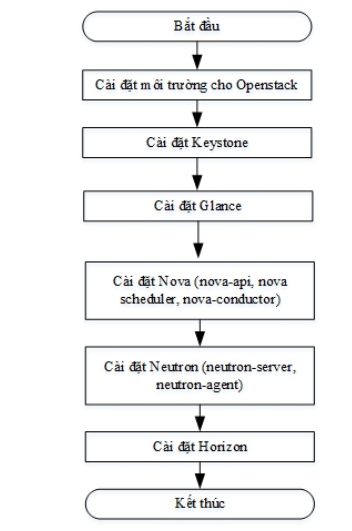
###### Hình 2.5: Mô hình triển khai mạng cho Openstack theo mô hình PhyComp-PhyCon

Trong mô hình PhyComp-VirCon, server ảo sẽ được thiết lập để cài đặt làm Controller node và server vật lý sẽ được thiết lập để cài đặt làm Compute node. Trên server ảo, các module được cài đặt cho Controller node bao gồm: OpenStack Identity (Keystone), OpenStack Compute (Nova), OpenStack Networking (Neutron), OpenStack Image (Glance) và OpenStack Dashboard (Horizon). Trên các máy server vật lý, các module được cài đặt cho Compute node bao gồm: OpenStack Compute (Nova), OpenStack Networking (Neutron). Dưới đây là chi tiết quy trình triển khai và cài đặt Controller node và Compute node.



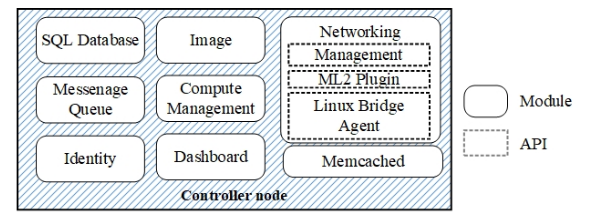
###### Hình 2.6: Kiến trúc logic của hệ thống

### Triển khai Controller node theo mô hình PhyComp-PhyCon



###### Hình 2.7: Sơ đồ quy hoạch Controller node

Để triển khai cài đặt Controller node, cần cài đặt các môi trường nền phục vụ cho việc triển khai các module của hệ thống. Sau khi cài đặt xong môi trường nền, sẽ tiến hành cài đặt các module theo tuần tự theo tài liệu hướng dẫn của Openstack:Keystone → Glance → Nova → Neutron → Horizon.

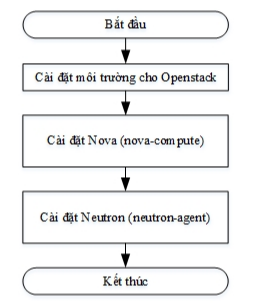


###### Hình 2.8: Các module được triển khai cho Controller node

Các phần mềm được cài đặt để tạo môi trường cho Openstack gồm Openstack Repository (phiên bản được sử dụng là phiên bản Openstack Zed), Openstack Client, SQL Database (làMySQL) , Messenger Queue (là RabbitMQ) và Memcached. Trong quá trình cài đặt các module chính của Openstack việc đầu tiên cần khởi tạo và gán quyền cho các database của “keystone”, “glance”, “nova”, “neutron” trong Controller node.

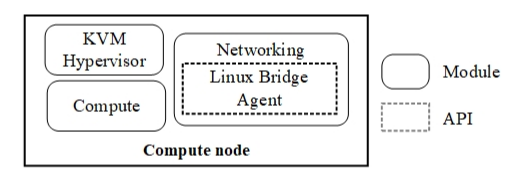
Trong Controller node, module đầu tiên là module Keystone. Keystone được cài đặt trong gói phần mềm openstack-keystone và được tinh chỉnh các thành phần trong file keystone.conf cho phép kết nối với database cũng như cài đặt keystone-manage nhằm khởi tạo và cập nhật dữ liệu trong Keystone để cung cấp dịch vụ định danh. Tiếp theo, cấu hình của Apache HTTP nhằm thiết lập và cung cấp dịch vụ web bằng cách cài đặt gói phần mềm httpd và mod\_wsgi, tinh chỉnh các thành phần trong file httpd.conf và chạy dịch vụ httpd đồng thời cấu hình tài khoản Admin với file admin-openrc. Dịch vụ định danh cung cấp tính năng xác thực cho mọi dịch vụ của Openstack. Tính năng xác thực được dùng kết hợp với các domain, project, user và role. Như vậy, các domain, project, user và role có thể được khởi tạo bởi người dùng, bước đầu tiên cần khởi tạo domain và dựa trên cơ sở các domain được khởi tạo, các project và user Keystone cũng sẽ được khởi tạo và đồng thời khởi tạo các role. Các role được thêm vào cho các project và các user Keystone. Để kết thúc việc cài đặt, Keystone cần được kiểm tra và xác thực thông tin lưu trữ trong database so với thông tin trong các bước khởi tạo, nếu kết quả trùng khớp thì xác lập việc cài đặt thành công và chuyển sang bước cài đặt tiếp theo. Ngoài môi trường tập lệnh mặc định của Openstack dành cho admin, Openstack cho phép người dùng tạo và cấu hình môi trường tập lệnh Openstack client để tương tác với dịch vụ định danh. Module thứ hai là module Glance. Dựa trên việc tạo user Glance và các project Keystone sẽ gán tới các role, ngoài ra, các service glance cần được khởi tạo để quản lý các image đồng thời khởi tạo các API endpoint cho Glance. Glance cài đặt gói phần mềm và tinh chỉnh các thành phần trong các file glance-api.conf, glance-registry.conf để kết nối với database cũng như cài đặt glance-manage để quản lý và cấu hình việc cài đặt Glance với database, tiếp đó cần khởi động lại các Image service. Để kết thúc việc cài đặt, glance cần được kiểm tra và xác nhận việc hoạt động của Image service để phục vụ vấn đề quản lý và lưu trữ chính xác các thông tin liên quan image máy ảo so với thông tin trong các bước khởi tạo, nếu kết quả trùng khớp thì xác lập việc cài đặt thành công và chuyển sang bước cài đặt tiếp theo. Module thứ ba là module Nova. Nova cần được cấu hình theo tài khoản admin với file admin-openrc. Khởi tạo các user Nova đồng thời thêm các role cho user Nova và các project Keystone, khởi tạo các service và các API endpoint cho Nova. Bước tiếp sẽ cài đặt gói phần mềm và tinh chỉnh các thành phần trong file cấu hình của Nova (bao gồm Nova API, Nova Scheduler và Nova Conductor) và khi cài đặt và cấu hình xong sẽ khởi động lại các dịch vụ của Nova. Khi các Compute node được thiết lập xong Controller node cần được quét và cập nhật các danh sách vào database qua service của Nova. Để kết thúc việc cài đặt, Nova cần được kiểm tra và xác thực thông tin lưu trữ trong database so với thông tin trong các bước khởi tạo, nếu kết quả trùng khớp thì xác lập việc cài đặt thành công và chuyển sang bước cài đặt tiếp theo. Module thứ tư là module Neutron. Neutron cũng cần được cấu hình theo tài khoản admin. Khởi tạo các user Neutron đồng thời thêm các role cho user Neutron và các project Keystone cũng như khởi tạo các service và các API endpoint cho Neutron. Sau đó, gói phần mềm được cài đặt và tinh chỉnh các thành phần trong các file neutron.conf, ml2\_conf.ini của Neutron. Các API được cài đặt và tinh chỉnh gồm API Openstack Neutron và API Openstack Neutron ML2 dành cho Neutron server nhằm cung cấp các dịch vụ liên quan tới Networking API còn API Openstack Neutron Linuxbridge dành cho Neutron Agent nhằm quản lý cấu hình các switch ảo (vswitch).Sau cùng, Neutron service cần được khởi động lại. Để kết thúc việc cài đặt, Neutron cần được kiểm tra và xác thực thông tin lưu trữ trong database so với thông tin trong các bước khởi tạo, nếu kết quả trùng khớp thì xác lập việc cài đặt thành công và chuyển sang bước cài đặt tiếp theo. Module cuối cùng là module Horizon. Gói phần mềm phục vụ Horizon là openstack-dashboard cần được cài đặt, tinh chỉnh các filelocal\_settings.py và openstack-dashboard.conf theo hướng dẫn tài liệu của Openstack.Sau đó, hệ thống cần khởi động lại web server và session storage service. Để kết thúc việc cài đặt, horizon cần được kiểm tra và xác nhận việc hoạt động bằng việc truy cập qua địa chỉ http://controller/dashboard đồng thời xác thực user và mặc định tên miền.Trong toàn bộ các file cấu hình để cung cấp mã token cần thiết lập thông qua port 11211, để truy nhập xác thực thông qua port 5000 khi kết nối với Keystone. Ngoài Ra để truy cập vào Glance để thao tác với image thông qua port 9292.

### Triển khai Compute node theo mô hình PhyComp-PhyCon



###### Hình 2.9: Sơ đồ quy trình cài đặt Compute node

Để triển khai cài đặt Compute node, cần cài đặt các môi trường nền phục vụ cho việc triển khai các module của hệ thống. Sau khi cài đặt xong môi trường nền, sẽ tiến hành cài đặt các module theo tuần tự theo tài liệu hướng dẫn của Openstack: Nova →Neutron.



###### Hình 2.10: Các module được triển khai cho Compute node

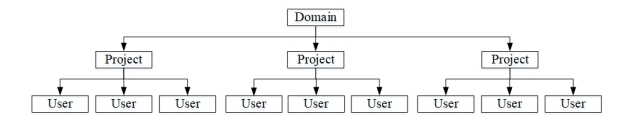
Các phần mềm được cài đặt để tạo môi trường cho Openstack gồm Openstack Repository (phiên bản được sử dụng là phiên bản Openstack Zed), Openstack Client,.

Trong Compute node, module đầu tiên là module Nova. Nova được cài đặt trong gói phần mềm openstack-nova-compute và được cấu hình trong file nova.conf và khi cấu hình xong sẽ khởi động lại các dịch vụ của Nova. Việc quản lý các tác vụ liên quan đến máy ảo sau khi máy ảo (KVM Hypervisor) được cài đặt trên Compute node được phụ trách bởi Controller node. Openstack cho phép người dùng có thể truy cập từ xa bằng công cụ VNC thông qua port 6080. Module đầu tiên là module Neutron. Các API được cài đặt và tinh chỉnh gồm API Openstack Neutron Linuxbridge dành cho Neutron Agent nhằm quản lý cấu hình các switch ảo (vswitch) phục vụ cho việc kết nối các máy ảo với nhau trong các file nova.conf, neutron.conf và linuxbridge\_agent.ini. Sau cùng,Neutron service và Nova service cần được khởi động lại.

Sau khi cài đặt xong các module của Openstack, quản trị viên sẽ quản trị hệ thống qua OpenStack Dashboard (Horizon) thông qua website. Tuy nhiên, quản trị viên cấu hình địa chỉ IP cho website quản trị có thể truy cập theo địa chỉ ứng với tên miền Internet. Sau khi đã cấu hình xong và quản trị viên đăng nhập vào hệ thống qua tài khoản admin, quản trị viên tạo ra các máy ảo và được cài sẵn hệ điều hành để người dùng có thể triển khai các ứng dụng (cụ thể là các phòng ban sẽ cài đặt các dịch vụ và ứng dụng liên quan tới đào tạo và thực hành ứng với các nhiệm vụ chức năng như công tác quản lý, giảng dạy và thực hành…). Quản trị viên cũng có thể tạo ra các rule để thiết lập bảo mật cũng như kết nối cho toàn bộ các máy ảo hay từng máy ảo trên Openstack.

## Sử dụng Openstack trong quản trị hệ thống Private Cloud

Sử dụng Openstack trong quản trị hệ thống Private Cloud cho trường đại học Hệ thống Openstack được quản trị theo domain. Các domain được quản trị bởi tài khoản admin trong domain Default. Ngoài ra, mỗi domain có tài khoản quản trị riêng của domain đó và tách biệt độc lập với các domain khác. Khi quản trị một domain, tài khoản quản trị được phép tạo nhiều project và nhiều user khác nhau được phân quyền(role) khác nhau (như admin, user, …) để tương tác với hệ thống Cloud.



###### Hình 2.11: Mô hình quản trị Openstack

Với việc triển khai hệ thống Private Cloud cùng hệ thống phần cứng hiện có để cung cấp tài nguyên cho các đơn vị trong trường học với mục đích sử dụng các ứng dụng đào tạo và thực hành, vai trò phân cấp chức năng của hệ thống quản trị cũng cần phải tương ứng với mô hình cơ cấu quản lý đào tạo giáo dục của nhà trường.

Mỗi domain được vận hành tương ứng với một cloud dùng để phục vụ cho các đơn vị trong nhà trường như khối hành chính, các khoa – viện – bộ môn. Mỗi domain cung cấp dịch vụ cho phép triển khai công tác nghiệp vụ, quản lý cũng như học tập giảng dạy của các đơn vị trong nhà trường. Mỗi domain hoạt động tách biệt với nhau có được kiểm soát bởi một tài khoản quản trị.

Mỗi project được vận hành tương ứng với cụm server được dùng cho từng phòng ban trong khối hành chính như phòng Đào tạo, phòng Kế toán… hay các dự án của khoa– viện, các môn học trong từng bộ môn. Mỗi project được quản trị nhằm phục vụ cho việc triển khai các hệ thống đặc thù của từng đơn vị. Mỗi domain hoạt động tách biệt với nhau và được sử dụng bởi một hay nhiều user với các quyền khác nhau.

Việc tạo các role và user trong hệ thống đáp ứng được nhu cầu phân cấp người dùng, cụ thể như trong một môn học, giảng viên có thể tạo ra các hệ thống thực nghiệm(gồm những máy ảo, cụm máy ảo đã được thiết lập sẵn môi trường và điều kiện thực hành, nghiên cứu) là các bài tập cho phép sinh viên, học viên vào để học tập, nghiên cứu theo như yêu cầu môn học mà không cần phải thiết lập lại hệ thống hoặc giảng viên có thể kiểm tra việc thực hành của người học theo yêu cầu hay cho phép học viên, sinh viên có quyền quản trị, theo dõi, giám sát và thao tác với hệ thống nhỏ (là những máy ảo, cụm máy ảo đã được thiết lập sẵn môi trường và các dịch vụ theo yêu cầu) trong dự án, đề tài do bộ môn, khoa – viện đang triển khai.

# CHƯƠNG 3: TRIỂN KHAI VÀ KIỂM THỬ HỆ THỐNG

## Triển khai Openstack

### Triển khai Controller node

#### Chuẩn bị cài đặt

* Cài đặt Openstack client

apt install python3-openstackclient

* Cài đặt MySQL

apt install mysql-server

systemctl start mysqld

mysql\_secure\_installation

* Cài đặt Memcached

apt install memcached python3-memcached

systemctl enable memcached.service

systemctl start memcached.service

* Cài đặt RabbitMQ

apt-get install rabbitmq-server

systemctl enable rabbitmq-server.service

systemctl start rabbitmq-server.service

rabbitmqctl add\_user openstack RABBIT\_PASS

rabbitmqctl set\_permissions openstack ".\*" ".\*" ".\*"

#### Cài đặt Keystone

* Tạo database và user “keystone” và gán quyền cho keystone

CREATE DATABASE keystone;

GRANT ALL PRIVILEGES ON keystone.\* TO 'keystone'@'localhost' IDENTIFIED BY 'KEYSTONE\_DBPASS';

GRANT ALL PRIVILEGES ON keystone.\* TO 'keystone'@'%' IDENTIFIED BY 'KEYSTONE\_DBPASS';

* Cài đặt và cấu hình các thành phần

apt install openstack-keystone httpd mod\_wsgi

* Sửa file cấu hình keystone

vim /etc/keystone/keystone.conf

{

[database]

...

connection=mysql+pymysql://keystone:KEYSTONE\_DBPASS@controller/keystone

[token]

...

provider=fernet

}

* Cài đặt keystone-manage

su -s /bin/sh -c "keystone-manage db\_sync" keystone

keystone-manage fernet\_setup --keystone-user keystone \

--keystone-group keystone

keystone-manage credential\_setup --keystone-user keystone \

--keystone-group keystone

keystone-manage bootstrap --bootstrap-password ADMIN\_PASS \

--bootstrap admin-url http://controller:5000/v3/ \

--bootstrap-internal-url http://controller:5000/v3/ \

--bootstrap-public-url http://controller:5000/v3/ \

--bootstrap-region-id RegionOne

* Bật service Apache HTTP

systemctl enable httpd.service

systemctl start httpd.service

* Config tài khoản ADMIN

vim admin-openrc

{

export OS\_USERNAME=admin

export OS\_PASSWORD=ADMIN\_PASS

export OS\_PROJECT\_NAME=admin

export OS\_USER\_DOMAIN\_NAME=Default

export OS\_PROJECT\_DOMAIN\_NAME=Default

export OS\_AUTH\_URL=http://controller:5000/v3

export OS\_IDENTITY\_API\_VERSION=3

}

. admin-openrc

* Tạo project, user, roles
* Tạo project

openstack project create --domain default \

--description "Service Project" service

openstack project create --domain default \

--description "Demo Project" myproject

* Tạo user

openstack user create --domain default --password-prompt myuser

* Thêm role, project cho user

openstack role add --project myproject --user myuser myrole

#### Cài đặt Glance

* Tạo database và user "glance". Gán quyền cho glance

CREATE DATABASE glance;

GRANT ALL PRIVILEGES ON glance.\* TO 'glance'@'localhost'

IDENTIFIED BY 'GLANCE\_DBPASS';

GRANT ALL PRIVILEGES ON glance.\* TO 'glance'@'%' IDENTIFIED

BY 'GLANCE\_DBPASS';

* Tạo user glance trong Openstack

openstack user create --domain default --password-prompt glance

* Thêm role “admin” cho user “glance” và project “service”

openstack role add --project service --user glance admin

* Tạo service glance

openstack service create --name glance \

--description "OpenStack Image" image

* Tạo API endpoint

openstack endpoint create --region RegionOne image public

http://controller:9292;

openstack endpoint create --region RegionOne image internal

http://controller:9292;

openstack endpoint create --region RegionOne image admin

http://controller:9292;

* Cài đặt và cấu hình các thành phần

apt install openstack-glance

vim /etc/glance/glance-api.conf

{

[database]

...

connection=mysql+pymysql://glance:GLANCE\_DBPASS@controller

/glance

[keystone\_authtoken]

...

www\_authenticate\_uri =http://controller:5000

auth\_url=http://controller:5000

memcached\_servers=controller:11211

auth\_type=password

project\_domain\_name=Default

user\_domain\_name=Default

project\_name=service

username=glance

password=GLANCE\_PASS

[paste\_deploy]

...

flavor=keystone

[glance\_store]

...

stores=file,http

default\_store=file

filesystem\_store\_datadir=/var/lib/glance/images/

}

* Cấu hình glance-registry

vi /etc/glance/glance-registry.conf

{

[database]

...

connection=mysql+pymysql://glance:GLANCE\_DBPASS@controller

/ glance

[keystone\_authtoken]

...

www\_authenticate\_uri=http://controller:5000

auth\_url=http://controller:5000

memcached\_servers=controller:11211

auth\_type=password

project\_domain\_name=Default

user\_domain\_name=Default

project\_name=service

username=glance

password=GLANCE\_PASS

[paste\_deploy]

...

flavor=keystone

}

* Cài đặt glance-manage

su -s /bin/sh -c "glance-manage db\_sync" glance

* Khởi động và cấu hình Image service

systemctl enable openstack-glance-api.service \

openstack-glance-registry.service

systemctl start openstack-glance-api.service \

openstack-glance-registry.service

* Verify operation

. admin-openrc

wget http://cloud-images.ubuntu.com/xenial/current/xenial-server-cloudimg\

amd64-disk1.img

openstack image create "ubuntu" \

--file xenial-server-cloudimg-amd64-disk1.img \

--disk-format qcow2 --container-format bare --public

openstack image list

#### Cài đặt Nova

* Tạo database và user "nova". Gán quyền cho nova

CREATE DATABASE nova\_api;

CREATE DATABASE nova;

CREATE DATABASE nova\_cell0;

CREATE DATABASE placement;

GRANT ALL PRIVILEGES ON nova\_api.\* TO 'nova'@'localhost'

IDENTIFIED BY 'NOVA\_DBPASS';

GRANT ALL PRIVILEGES ON nova\_api.\* TO 'nova'@'%' IDENTIFIED

BY 'NOVA\_DBPASS';

GRANT ALL PRIVILEGES ON nova.\* TO 'nova'@'localhost'

IDENTIFIED BY 'NOVA\_DBPASS';

GRANT ALL PRIVILEGES ON nova.\* TO 'nova'@'%' IDENTIFIED BY

'NOVA\_DBPASS';

GRANT ALL PRIVILEGES ON nova\_cell0.\* TO 'nova'@'localhost'

IDENTIFIED BY 'NOVA\_DBPASS';

GRANT ALL PRIVILEGES ON nova\_cell0.\* TO 'nova'@'%' IDENTIFIED

BY 'NOVA\_DBPASS';

GRANT ALL PRIVILEGES ON placement.\* TO 'placement'@'localhost'

IDENTIFIED BY 'PLACEMENT\_DBPASS';

GRANT ALL PRIVILEGES ON placement.\* TO 'placement'@'%'

IDENTIFIED BY 'PLACEMENT\_DBPASS';

* Tạo user ở Openstack

openstack user create --domain default --password-prompt nova

* Thêm role “admin” cho user “nova”

openstack role add --project service --user nova admin

* Tạo service compute, placement

openstack service create --name nova \

--description "OpenStack Compute" compute

openstack service create --name placement \

--description "Placement API" placement

* Tạo API endpoint cho nova

openstack endpoint create --region RegionOne \

compute public http://controller:8774/v2.1

openstack endpoint create --region RegionOne \

compute internal http://controller:8774/v2.1

openstack endpoint create --region RegionOne \

compute admin http://controller:8774/v2.1

* Thêm role “admin” vào project “service”

openstack user create --domain default --password-prompt placement

* Cài đặt và cấu hình các thành phần

apt install openstack-nova-api openstack-nova-conductor openstack-nova \

console openstack-nova-novncproxy openstack-nova-scheduler \

openstack-nova-placement-api

* Sửa file nova.conf

vim /etc/nova/nova.conf

{

[DEFAULT]

...

enabled\_apis=osapi\_compute,metadata

[api\_database]

...

connection=mysql+pymysql://nova:NOVA\_DBPASS@controller/nova \_api

[database]

...

connection=mysql+pymysql://nova:NOVA\_DBPASS@controller/nova

[placement\_database]

...

connection=mysql+pymysql://placement:PLACEMENT\_DBPASS@controller/placement

[DEFAULT]

...

transport\_url=rabbit://openstack:RABBIT\_PASS@controller

[api]

...

auth\_strategy=keystone

[keystone\_authtoken]

...

auth\_url=http://controller:5000/v3

memcached\_servers=controller:11211

auth\_type=password

project\_domain\_name=default

user\_domain\_name=default

project\_name=service

username=nova

password=NOVA\_PASS

[DEFAULT]

...

my\_ip=10.10.0.33

[DEFAULT]

...

use\_neutron=true

firewall\_driver=nova.virt.firewall.NoopFirewallDriver

[vnc]

enabled=true

...

server\_listen=$my\_ip

server\_proxyclient\_address=$my\_ip

[glance]

...

api\_servers=http://controller:9292

[oslo\_concurrency]

...

lock\_path=/var/lib/nova/tmp

[placement]

...

region\_name=RegionOne

project\_domain\_name=Default

project\_name=service

auth\_type=password

user\_domain\_name=Default

auth\_url=http://controller:5000/v3

username=placement

password=PLACEMENT\_PASS

}

* Sửa file 00-nova-placement-api.conf

vim /etc/httpd/conf.d/00-nova-placement-api.conf

{

<Directory /usr/bin>

<IfVersion >= 2.4>

Require all granted

</IfVersion>

<IfVersion < 2.4>

Order allow,deny

Allow from all

</IfVersion>

</Directory>

}

su -s /bin/sh -c "nova-manage api\_db sync" nova

su -s /bin/sh -c "nova-manage cell\_v2 map\_cell0" nova

su -s /bin/sh -c "nova-manage cell\_v2 create\_cell --name=cell1 --verbose"

su -s /bin/sh -c "nova-manage db sync" nova

su -s /bin/sh -c "nova-manage cell\_v2 list\_cells" nova

systemctl enable openstack-nova-api.service openstack-nova-consoleauth

openstack-nova-scheduler.service openstack-nova-conductor.service

openstack-nova-novncproxy.service

systemctl start openstack-nova-api.service openstack-nova-consoleauth

openstack-nova-scheduler.service openstack-nova-conductor.service

openstack-nova-novncproxy.service

* Tìm kiếm compute hosts

su -s /bin/sh -c "nova-manage cell\_v2 discover\_hosts --verbose" nova

vim /etc/nova/nova.conf

{

[scheduler]

discover\_hosts\_in\_cells\_interval=300

}

#### Cài đặt Neutron

* Tạo database và user "neutron". Gán quyền cho neutron

CREATE DATABASE neutron;

GRANT ALL PRIVILEGES ON neutron.\* TO 'neutron'@'localhost'

IDENTIFIED BY 'NEUTRON\_DBPASS';

GRANT ALL PRIVILEGES ON neutron.\* TO 'neutron'@'%' IDENTIFIED

BY 'NEUTRON\_DBPASS';

* Tạo user neutron

openstack user create --domain default --password-prompt neutron

* Thêm role “admin” cho user “nova”

openstack role add --project service --user neutron admin

* Tạo service network

openstack service create --name neutron \

--description "OpenStack Networking" network

* Tạo API endpoint

openstack endpoint create --region RegionOne \

network public http://controller:9696

openstack endpoint create --region RegionOne \

network internal http://controller:9696

openstack endpoint create --region RegionOne \

network admin http://controller:9696

* Cài đặt neutron

apt install openstack-neutron openstack-neutron-ml2 \

openstack-neutron linuxbridge ebtables

* Cấu hình neutron

vim /etc/neutron/neutron.conf

{

[database]

...

connection=mysql+pymysql://neutron:NEUTRON\_DBPASS@control

ler/neutron

[DEFAULT]

...

core\_plugin=ml2

service\_plugins =

[DEFAULT]

...

transport\_url=rabbit://openstack:RABBIT\_PASS@controller

[DEFAULT]

...

auth\_strategy=keystone

[keystone\_authtoken]

...

www\_authenticate\_uri=http://controller:5000

auth\_url=http://controller:5000

memcached\_servers=controller:11211

auth\_type=password

project\_domain\_name=default

user\_domain\_name=default

project\_name=service

username=neutron

password=NEUTRON\_PASS

}

* Cấu hình neutron-db-manage

su -s /bin/sh -c "neutron-db-manage --config-file /etc/neutron/neutron.conf \

--config-file /etc/neutron/plugins/ml2/ml2\_conf.ini upgrade head" neutron

* Khởi chạy networking service

systemctl enable neutron-server.service neutron-linuxbridge-agent.service \

neutron-dhcp-agent.service neutron-metadata-agent.service

systemctl start neutron-server.service neutron-linuxbridge-agent.service \

neutron-dhcp-agent.service neutron-metadata-agent.service

systemctl enable neutron-l3-agent.service

systemctl start neutron-l3-agent.service

#### Cài đặt Dashboard

* Cài đặt các thành phần

apt install openstack-dashboard

* Sửa file cấu hình local\_setting.py

vim /etc/openstack-dashboard/local\_settings.py

{

OPENSTACK\_HOST="controller"

ALLOWED\_HOSTS=['one.example.com', 'two.example.com']

SESSION\_ENGINE='django.contrib.sessions.backends.cache'

CACHES={

'default': {

'BACKEND':

'django.core.cache.backends.memcached.MemcachedC

ache',

'LOCATION': 'controller:11211',

}

}

OPENSTACK\_KEYSTONE\_URL="http://%s:5000/v3" %

OPENSTACK\_HOST

OPENSTACK\_KEYSTONE\_MULTIDOMAIN\_SUPPORT=True

OPENSTACK\_API\_VERSIONS={

"identity": 3,

"image": 2,

"volume": 2,

}

OPENSTACK\_KEYSTONE\_DEFAULT\_DOMAIN="Default"

OPENSTACK\_KEYSTONE\_DEFAULT\_ROLE="user"

OPENSTACK\_NEUTRON\_NETWORK={

...

'enable\_router': False,

'enable\_quotas': False,

'enable\_distributed\_router': False,

'enable\_ha\_router': False,

'enable\_lb': False,

'enable\_firewall': False,

'enable\_vpn': False,

'enable\_fip\_topology\_check': False,

}

TIME\_ZONE="TIME\_ZONE"

}

* Sửa file openstack-dashboard.conf

vim /etc/httpd/conf.d/openstack-dashboard.conf

{

WSGIApplicationGroup %{GLOBAL}

}

* Khởi động lại web server và session storage service

systemctl restart httpd.service memcached.service

* Verify operation

<http://controller/dashboard>.

Xác thực user admin hay demo có thể login

#### Cài đặt Cinder

* Tạo database và user " cinder". Gán quyền cho cinder

CREATE DATABASE cinder;

GRANT ALL PRIVILEGES ON cinder.\* TO 'cinder'@'localhost'

IDENTIFIED BY 'CINDER\_DBPASS';

GRANT ALL PRIVILEGES ON cinder.\* TO 'cinder'@'%' IDENTIFIED

BY 'CINDER\_DBPASS';

* Tạo user cinder

openstack user create --domain default --password-prompt cinder

* Thêm role “admin” cho user “cinder”

openstack role add --project service --user cinder admin

* Tạo service volumev2, volumev3

openstack service create --name cinderv2 \

--description "OpenStack Block Storage" volumev2

openstack service create --name cinderv3 \

--description "OpenStack Block Storage" volumev3

* Tạo API endpoint

openstack endpoint create --region RegionOne \

volumev2 public http://controller:8776/v2/%\(project\_id\)s

openstack endpoint create --region RegionOne \

volumev2 internal http://controller:8776/v2/%\(project\_id\)s

openstack endpoint create --region RegionOne \

volumev2 admin http://controller:8776/v2/%\(project\_id\)s

openstack endpoint create --region RegionOne \

volumev3 public http://controller:8776/v3/%\(project\_id\)s

openstack endpoint create --region RegionOne \

volumev3 internal http://controller:8776/v3/%\(project\_id\)s

openstack endpoint create --region RegionOne \

volumev3 admin http://controller:8776/v3/%\(project\_id\)s

* Cài đặt các thành phần

apt install openstack-cinder

* Sửa file cấu hình cinder.conf

vim /etc/cinder/cinder.conf

{

[database]

...

connection =

mysql+pymysql://cinder:CINDER\_DBPASS@controller/cinder

[DEFAULT]

...

transport\_url = rabbit://openstack:RABBIT\_PASS@controller

[DEFAULT]

...

auth\_strategy = keystone

[keystone\_authtoken]

...

www\_authenticate\_uri = http://controller:5000

auth\_url = http://controller:5000

memcached\_servers = controller:11211

auth\_type = password

project\_domain\_id = default

user\_domain\_id = default

project\_name = service

username = cinder

password = CINDER\_PASS

[DEFAULT]

...

my\_ip = 10.0.0.11

[oslo\_concurrency]

...

lock\_path = /var/lib/cinder/tmp

}

* Sửa file cấu hình nova.conf

vim /etc/nova/nova.conf

{

[cinder]

os\_region\_name = RegionOne

}

* Khởi động lại các service

systemctl restart openstack-nova-api.service

systemctl enable openstack-cinder-api.service \

openstack-cinder-scheduler.service

systemctl start openstack-cinder-api.service \

openstack-cinder-scheduler.service

### Triển khai Compute node

#### Cài đặt Nova

* Cài đặt các thành phần

apt install openstack-nova-compute

* Sửa file nova.conf

vim /etc/nova/nova.conf

{

[DEFAULT]

...

enabled\_apis=osapi\_compute,metadata

[DEFAULT]

...

transport\_url=rabbit://openstack:RABBIT\_PASS@controller

[api]

...

auth\_strategy=keystone

[keystone\_authtoken]

...

auth\_url=http://controller:5000/v3

memcached\_servers=controller:11211

auth\_type=password

project\_domain\_name=default

user\_domain\_name=default

project\_name=service

username=nova

password=NOVA\_PASS

[DEFAULT]

...

my\_ip=MANAGEMENT\_INTERFACE\_IP\_ADDRESS

[DEFAULT]

...

use\_neutron=true

firewall\_driver=nova.virt.firewall.NoopFirewallDriver

[vnc]

...

enabled=true

server\_listen=0.0.0.0

server\_proxyclient\_address=$my\_ip

novncproxy\_base\_url=http://controller:6080/vnc\_auto.html

[glance]

...

api\_servers=http://controller:9292

[oslo\_concurrency]

...

lock\_path=/var/lib/nova/tmp

[placement]

...

region\_name=RegionOne

project\_domain\_name=Default

project\_name=service

auth\_type=password

user\_domain\_name=Default

auth\_url=http://controller:5000/v3

username=placement

password=PLACEMENT\_PASS

[libvirt]

...

virt\_type=qemu

}

* Khởi động lại dịch vụ

systemctl enable libvirtd.service openstack-nova-compute.service

systemctl start libvirtd.service openstack-nova-compute.service

#### Cài đặt Neutron

* Cài đặt các thành phần

apt install openstack-neutron-linuxbridge ebtables ipset

* Sửa file nova.conf

vim /etc/neutron/neutron.conf

{

[DEFAULT]

...

transport\_url=rabbit://openstack:RABBIT\_PASS@controller

[DEFAULT]

...

auth\_strategy=keystone

[keystone\_authtoken]

...

www\_authenticate\_uri=http://controller:5000

auth\_url=http://controller:5000

memcached\_servers=controller:11211

auth\_type=password

project\_domain\_name=default

user\_domain\_name=default

project\_name=service

username=neutron

password=NEUTRON\_PASS

[oslo\_concurrency]

...

lock\_path=/var/lib/neutron/tmp

}

* Sửa file linuxbridge\_agent.ini

vim /etc/neutron/plugins/ml2/linuxbridge\_agent.ini

{

[linux\_bridge]

physical\_interface\_mappings=provider:PROVIDER\_INTERFACE\_N

AME

[vxlan]

enable\_vxlan=false

[securitygroup]

...

enable\_security\_group=true

firewall\_driver=neutron.agent.linux.iptables\_firewall.IptablesFirewall

Driver

}

* Sửa file nova.conf

vim /etc/nova/nova.conf

{

[neutron]

...

url=http://controller:9696

auth\_url=http://controller:5000

auth\_type=password

project\_domain\_name=default

user\_domain\_name=default

region\_name=RegionOne

project\_name=service

username=neutron

password=NEUTRON\_PASS

}

* Khởi động lại dịch vụ

modprobe br\_netfilter

sysctl -p

systemctl enable openstack-nova-compute.service

systemctl restart openstack-nova-compute.service

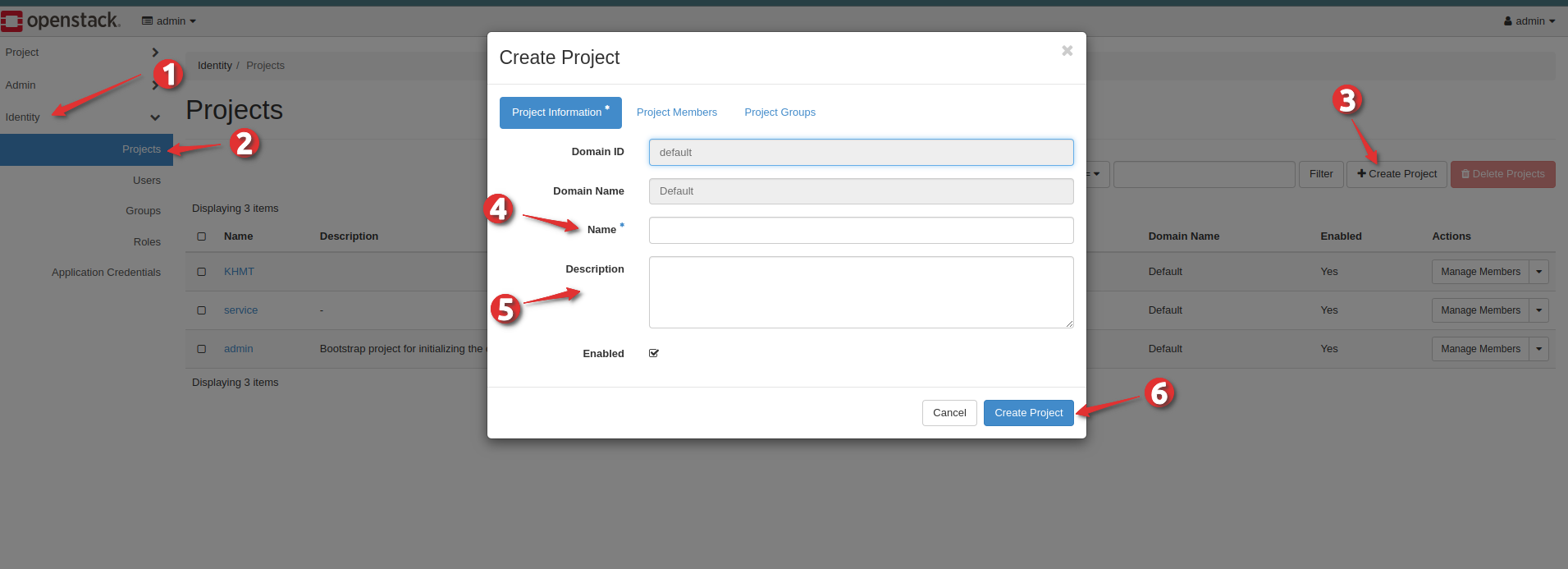
systemctl enable neutron-linuxbridge-agent.service

systemctl start neutron-linuxbridge-agent.service

## Kiểm thử hệ thống

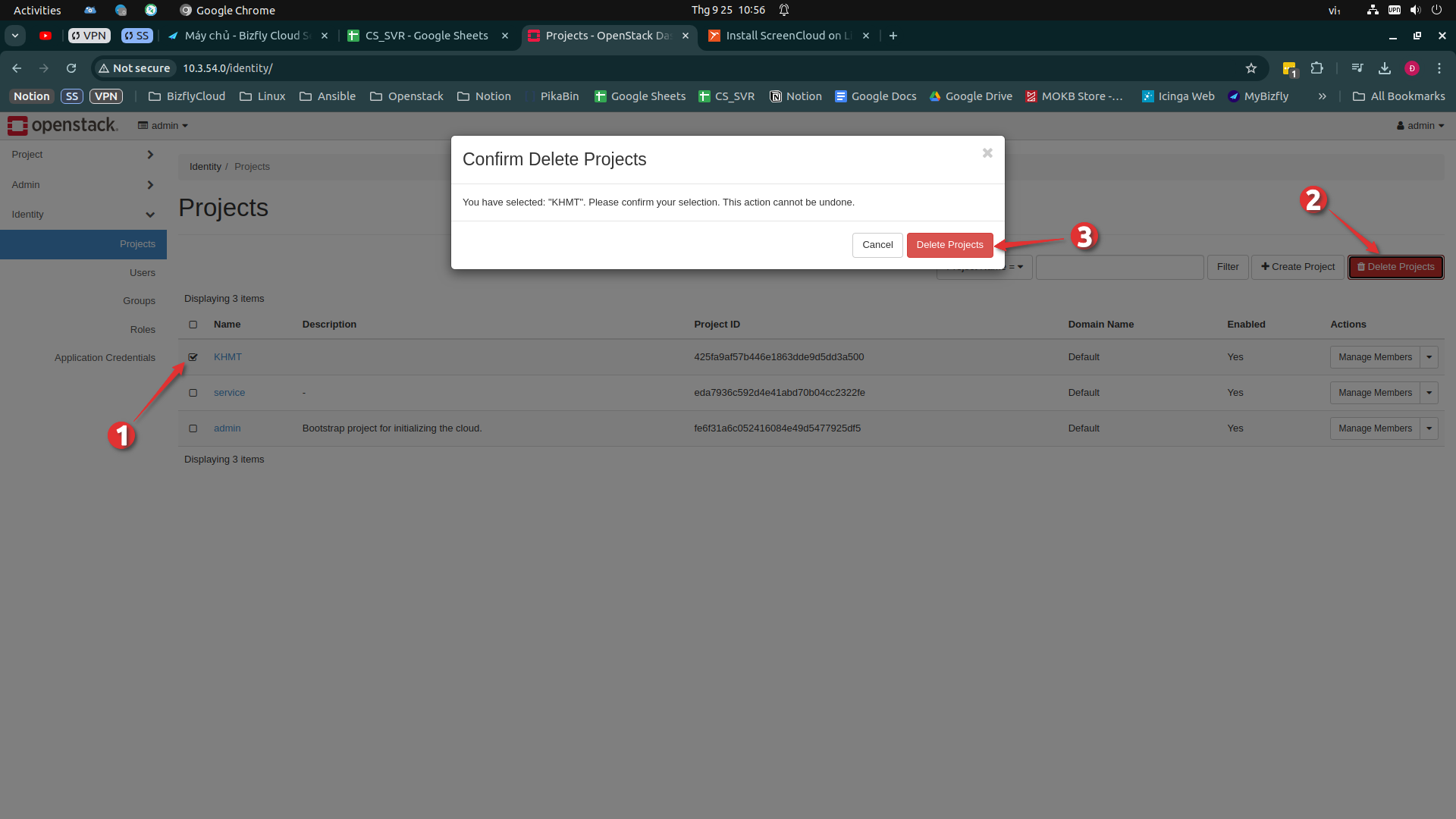
Để quản trị hệ thống Openstack, quản trị viên cần tiến hành việc khởi tạo và quản trị với hệ thống lần lượt theo thứ tự domain → project → role → user. Sau khi cài đặt thành công Openstack, quản trị viên sẽ đăng nhập với domain Default cùng với username và password là tài khoản được cấu hình trong file admin-openrc. Ngoài Domain mặc định Default để có thể quản trị hệ thống nhằm cung cấp các dịch vụ trên Openstack, quản trị viên có thể khởi tạo domain khác cho hệ thống. Domain được quản trị viên khởi tạo bằng giao diện dòng lệnh(cli) trong Controller node.

Để khởi tạo một project, quản trị viên cần chọn menu Identity → Projects và chọn Create Project đồng thời nhập thông tin về tên, mô tả về project như hình bên dưới.



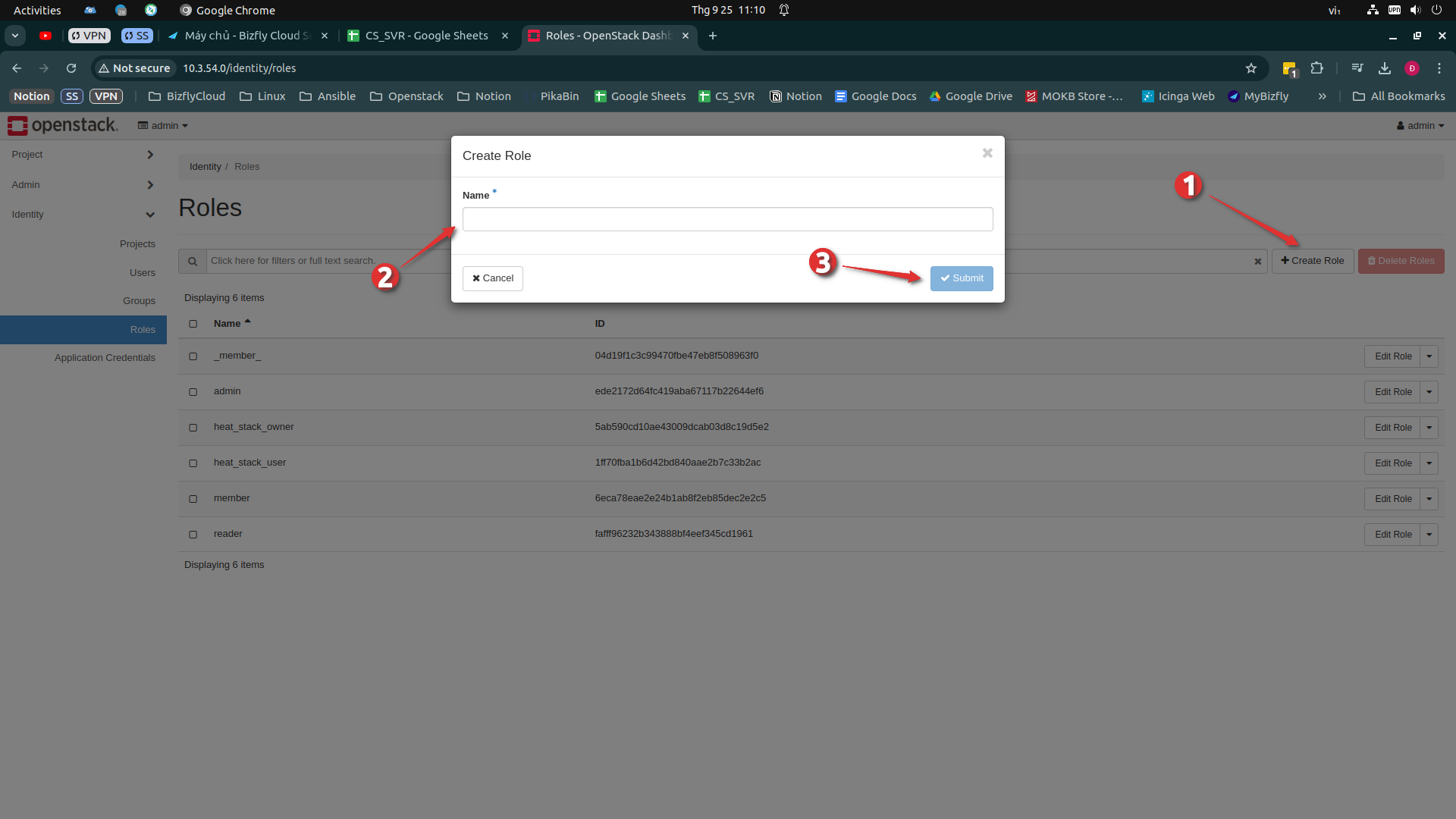
###### Hình 3.1: Khởi tạo một project

Ngoài ra, trong quá trình quản trị project, quản trị viên có thể thay đổi hay xóa từng project.



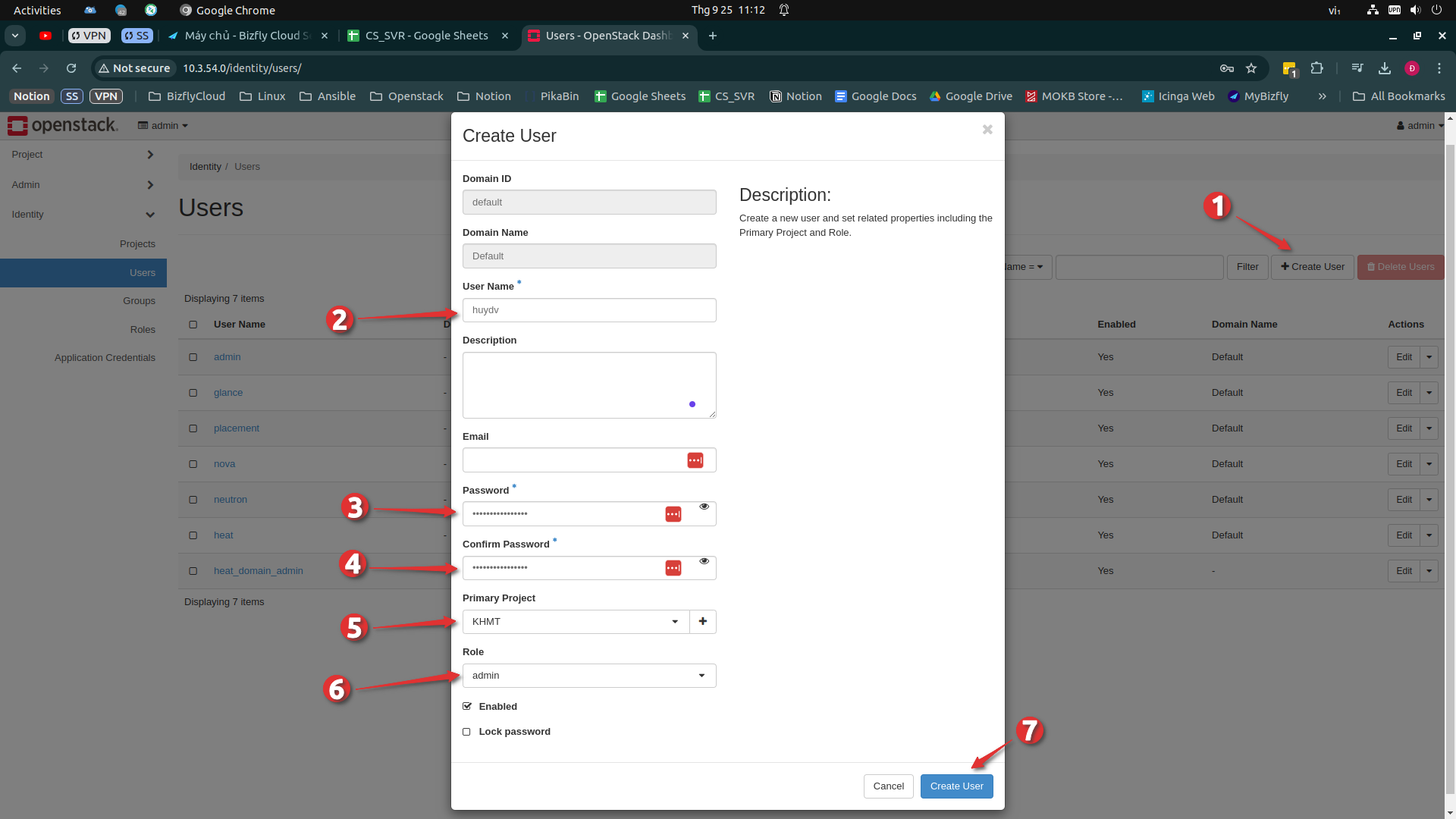
###### Hình 3.2: Xóa một project

Với mỗi project, quản trị viên cũng sẽ tạo và quản lý nhiều user với các role khác nhau. Sau khi khởi tạo project, quản trị viên cần khởi tạo các quyền nhằm gán quyền khi tạo tài khoản người dùng. Role được quản trị viên khởi tạo bằng cách chọn menu Identity → Roles và chọn Create Role và nhập thông tin về tên Role.



###### Hình 3.3: Khởi tạo một role

Để khởi tạo một User, quản trị viên cần chọn menu Identity → Users và chọn Create User đồng thời nhập thông tin về tên, mô tả, mật khẩu đăng nhập của user, project của user cũng như quyền (role) tương ứng của người dùng. Trong quá trình tạo User, quản trị viên cần biết chính xác Project mà tài khoản người dùng cần khởi tạo cũng như vai trò của người dùng trong Project thông qua role.

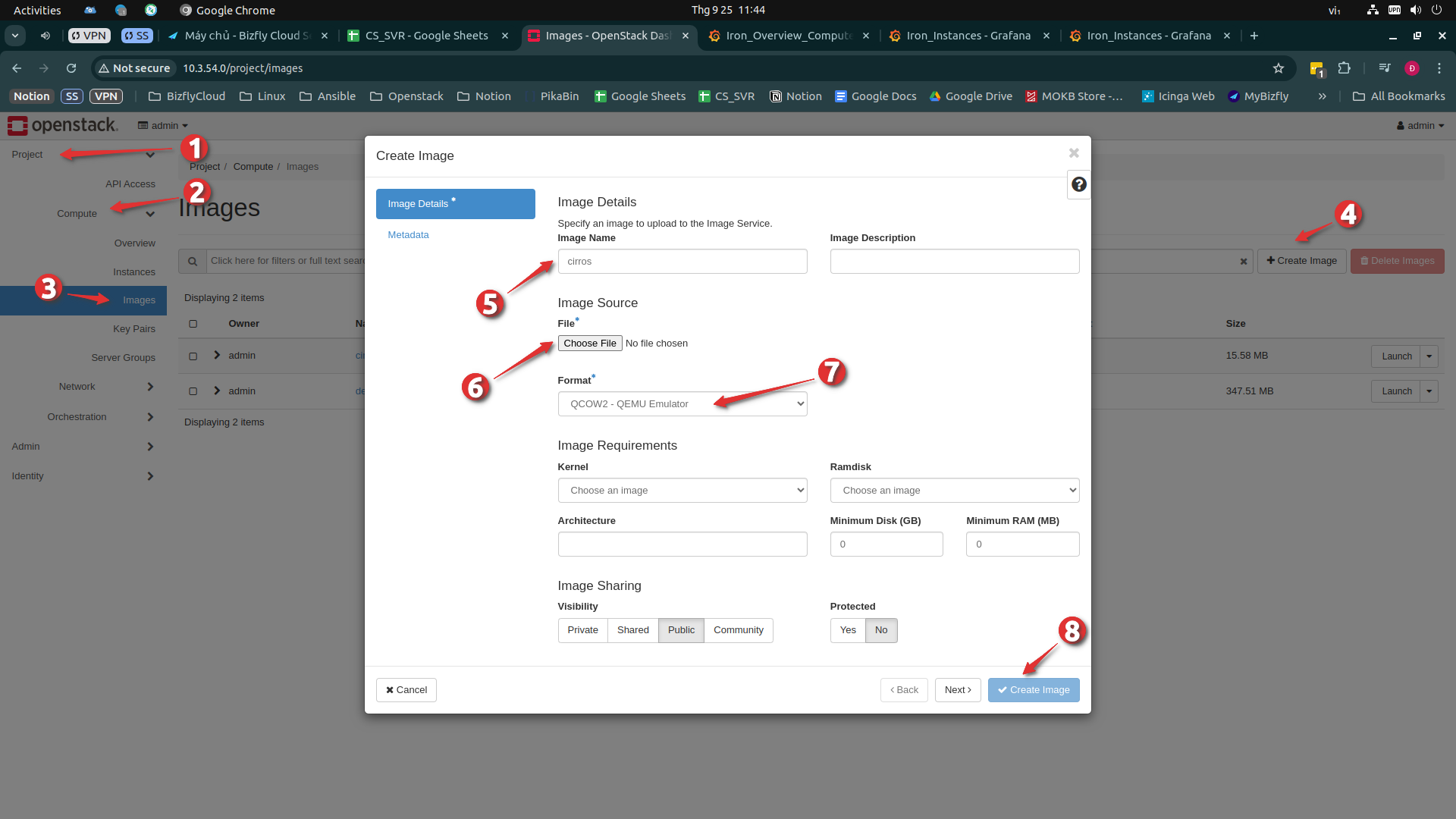


###### HÌnh 3.4: Khởi tạo một người dùng

Sau khi đã khởi tạo một user xong, người dùng sẽ đăng nhập vào Openstack, điền chính xác tên domain được khởi tạo để quản lý user và thông tin tài khoản của người dùng được khởi tạo. Trong quá trình quản trị, quản trị viên có thể thay đổi role cho user cũng như có thể vô hiệu hóa tài khoản user mà không nhất thiết phải xóa tài khoản user nhằm đảm bảo tính an toàn của hệ thống.

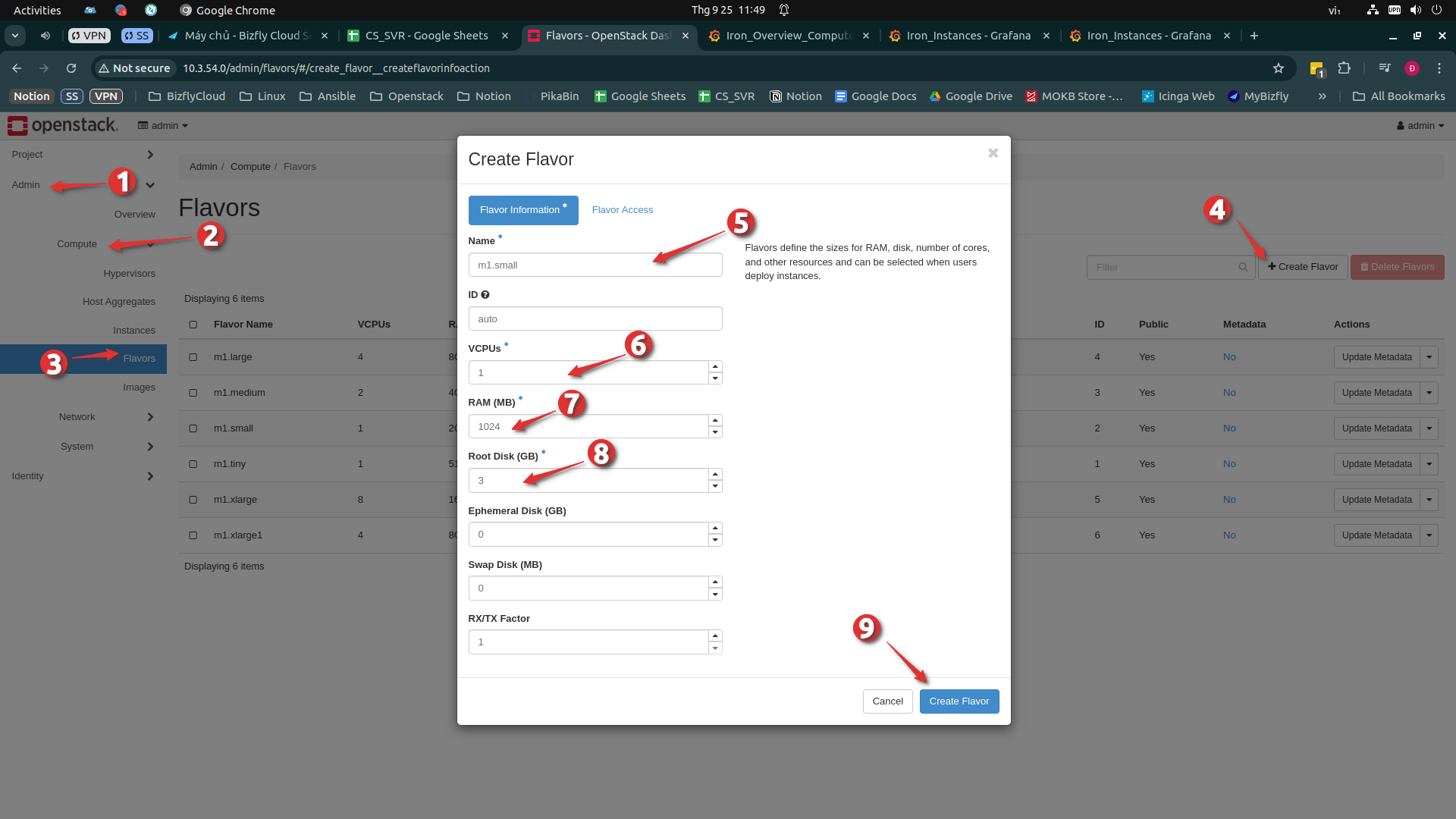
Ngoài việc quản trị theo mô hình phân cấp chức năng của hệ thống Cloud Openstack còn cung cấp các cơ chế để quản trị theo mô hình hướng người dùng. Người dùng có thể quy hoạch tạo và thay đổi hệ thống các cụm server do bản thân quản lý bằng việc thiết lập hệ thống mạng bao gồm việc kết nối cho từng máy ảo, việc đóng mở các port, các luật nhằm bảo mật kết nối và thiết lập hệ thống máy ảo bao gồm lựa chọn hệ điều hành, cấu hình máy ảo, vị trí máy ảo trong hệ thống.

Trong mỗi tài khoản người dùng, có thể tạo một hay nhiều máy ảo. Người dùng khi được cấp quyền cho phép truy nhập vào các project khác nhau cũng có thể thao tác với các máy ảo được tạo có trong các project đó. Để có thể cài đặt máy ảo, quản trị viên cần khởi tạo source image là các bản cài đặt hệ điều hành cho máy ảo. Để khởi tạo Image, quản trị viên chọn menu Project → Compute → Image, chọn Create Image để nhập thông tin cũng như tải file image là các bản cài đặt hệ điều hành và để nên định dạng của image là QCOW2.



###### Hình 3.5: Upload image lên hệ thống

Để định dạng cấu hình cho việc cấp phát tài nguyên cho máy ảo, quản trị viên cần tạo các flavor. Để khởi tạo Flavor, quản trị viên chọn menu Admin → Compute → Flavors, chọn Create Flavor và nhập thông tin về tên flavor, cấu hình cho máy ảo (CPU, RAM, dung lượng lưu trữ)



###### Hình 3.6: Khởi tạo cấu hình cho máy ảo

Để khởi tạo các kết nối mạng trong hệ thống,quản trị viên chọn menu Project → Network → Networks. Tại đây, quản trị viên có thể thiết lập tên mạng, cấu hình dải mạng (subnet) cho máy ảo hay cụm máy ảo kết nối với nhau.

Ngoài ra, khi đã thiết lập các dải mạng, để kết nối chúng với nhau hay tạo lập kết nối với dải mạng cho phép truy cập Internet, quản trị viên tạo và cấu hình cho các router qua thao tác chọn menu Project → Network → Routers. Router được thiết lập để kết nối với đường mạng cho phép kết nối với Internet. Nếu muốn kết nối router với đường mạng LAN trong hệ thống, quản trị viên chọn router cần kết nối và chọn Interface →Add Interface và chọn subnet cần kết nối

Openstack hỗ trợ việc kết nối ssh với các công cụ như putty,openssh… bằng việc tạo key pair. Để khởi tạo Key pair, quản trị viên chọn menu Compute → Key Pairs, chọn Create Key Pair và nhập tên key pair. Hệ thống sẽ tự động cho phép người dùng tải key pair ngay sau khi khởi tạo. Cuối cùng là thao tác khởi tạo các máy ảo (instance). Để khởi tạo Instance, quản trị viên chọn menu Compute →Instances, chọn Launch Instance và nhập các thông tin về tên cũng như mô tả cho instance, số lượng instance cần tạo, hình thức boot hệ điều hành vào máy ảo, dung lượng ổ ảo, hệ điều hành cần tạo cho instance, loại flavor, dải mạng sẽ kết nối, các cơ chế bảo mật cho instance, key pair (nếu sử dụng ssh). Sau khi tạo instance, quản trị viên có thể kiểm tra thông tin của instance như port ssh của máy ảo, nội dung hoạt động của máy ảo và các thao tác người dùng trên máy ảo, cũng như truy cập vào máy ảo trên nền web qua console.

Thông qua việc khởi tạo các instance (chính là các máy ảo) trong Project một cách đơn giản và nhanh chóng, quản trị viên có thể thiết lập các hệ thống mạng khác nhau gồm các cụm máy ảo nhằm phục vụ việc triển khai cài đặt các ứng dụng.

# KẾT LUẬN

Đề tài “**Nghiên cứu và thiết kế hệ thống điện toán đám mây Private Cloud cho nội bộ sử dụng Openstack**” xuất phát từ những nhu cầu thực tế mà ngày nay trong các doanh nghiệp, nhà trường cần có. Qua quá trình thực hiện đề tài, em đã khảo sát hệ thống phần cứng hiện có dùng để triển khai, xây dựng mô hình triển khai tham chiếu cũng như đưa ra các bài toán quy hoạch máy chủ. Trong quá trình xem xét các mô hình PhyComp-PhyCon và PhyComp-CoPhyCon, PhyComp-VirCon khi triển khai trên các máy chủ vật lý cũng như việc em đề xuất mô hình PhyComp-PhyCon để đánh giá năng lực xử lý của hệ thống và hiệu suất sử dụng tài nguyên khi bố trí các khối chức năng vào vị trí cũng như phương thức triển khai trên các server hiện có. Qua so sánh các ưu điểm và nhược điểm của các mô hình, em đã lựa chọn mô hình PhyComp-PhyCon là mô hình để giải quyết bài toán quy hoạch máy chủ. Dựa trên mô hình PhyComp-PhyCon, em đã thực hiện triển khai thực nghiệm máy chủ vật lý Controller node lên trên một trong các máy vật lý sẽ cài đặt Compute node cũng như mô tả quy trình cài đặt bằng sơ đồ, mô hình các module được cài đặt cho Controller node và Compute node cùng với mô tả chi tiết các bước cài đặt theo tài liệu hướng dẫn của Openstack. Ngoài ra, em cũng đã đưa ra quy trình quản trị hệ thống Private Cloud sử dụng Openstack, từ việc mô tả phân cấp chức năng của hệ thống gồm domain, project, user, role cũng như phân tích và áp dụng mô hình phân cấp chức năng vào trong mô hình triển khai các ứng dụng đào tạo và thực hành trong môi trường đại học thông qua việc phân tích quy trình ứng dụng hệ thống Openstack cũng như các thao tác kỹ thuật liên quan tới cài đặt và quản trị hệ thống.

* Kết quả đạt được:
  + Xây dựng được hệ thống private cloud sử dụng Openstack để cung cấp dịch vụ máy ảo dưới dạng cung cấp dịch vụ IaaS
  + Hệ thống private cloud để người dùng có thể tương tác trực tiếp với máy ảo
* Hướng phát triển trong tương lai:
  + Tích hợp Octavia để cung cấp dịch vụ Load Balancer
  + Tích hợp Trove để cung cấp dịch vụ DBaaS

# TÀI LIỆU THAM KHẢO

[1]. <https://docs.openstack.org/zed/>

[2]. What is a private cloud? – Microsoft Azure (azure.microsoft.com)

[3]. Cloud computing, Virtualization, Hypervisor, Openstack – Wikipedia (wikipedia.org).