ĐẠI HỌC QUỐC GIA HÀ NỘI

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ

BÁO CÁO THỰC TẬP

NGÀNH: KHOA HỌC MÁY TÍNH

ĐỀ TÀI: XÂY DỰNG HỆ THỐNG PRIVATE CLOUD VỚI OPENSTACK

Giảng viên hướng dẫn: ***TS. TRẦN TRÚC MAI***

Sinh viên: NGUYỄN ĐỨC LÂM

Mã sinh viên: 17021280

Lớp: K62-CACLC-2

Hà Nội, tháng 10 năm 2020

Mục lục

Lời cảm ơn

Để hoàn thành báo cáo thực tập này trước hết em xin gửi lời chân thành cảm ơn đến khoa CNTT và giảng viên hướng dẫn Trần Trúc Mai cùng với sự hỗ trợ của giảng viên Hoàng Xuân Tùng đã tạo điều kiện giúp đỡ em trong suốt quá trình thực tập này. Sau 8 tuần thực tập ngắn ngủi nhưng là thời gian quí báu để cho em tổng hợp và hệ thống hóa lại kiến thức đã được học, đồng thời kết hợp với thực tế để nâng cao kiến thức chuyên môn. Tuy chỉ trong 8 tuần thực tập nhưng thông qua làm việc cùng thầy, em đã được mở rộng tầm nhìn và tiếp thu được nhiều kiến thức thực tế. Từ đó em nhận thấy việc cọ sát thực tế là vô cùng quan trọng – là yếu tố giúp cho sinh viên xây dựng được nền tảng kiến thức học ở nhà trường vững chắc hơn. Em mong rằng từ những kiến thức, kinh nghiệm đã tiếp thu được sẽ xây dựng nền tảng vững vàng để em có thể bước ra ngoài xã hội và không ngừng hoàn thiện phát triển bản thân.

Trong quá trình làm việc với các thầy do kinh nghiệm và kĩ năng còn chưa tốt nên em khó tránh khỏi các sai sót, vì vậy em rất mong các thầy có thể bỏ qua. Đồng thời do trình độ lý luận cũng như kinh nghiệm thực tiễn còn hạn chế nên báo cáo của em không thể tránh khỏi những thiếu sót, em mong sẽ nhận được góp ý từ các thầy để em có thể hoàn thiện tốt hơn những báo cáo trong tương lai.

I. Giới thiệu chung.

1. Giới thiệu công việc.

Sau khi kết thúc kì học 2019-2020, nhà trường đã tạo điều kiện cho các sinh viên được tham gia trải nghiệm thực tập với các doanh nghiệp hoặc giảng viên trong các khoa để sinh viên có thêm kinh nghiệm và kĩ năng cần thiết cho công việc sau khi ra trường. Em đã tham gia đăng kí thực tập với giảng viên Trần Trúc Mai trong Khoa CNTT, bộ môn Mạng và Truyền thông máy tính và tham gia làm một dự án nhỏ dưới sự hướng dẫn của thầy Mai cùng với sự hỗ trợ của thầy Hoàng Xuân Tùng. Em được sắp xếp nơi làm việc là phòng dự án của bộ môn và thực tập full-time 8 tiếng / ngày. Tiếp xúc và làm việc trực tiếp với các thầy tạo điều kiện thuận lợi cho em được hướng dẫn tỉ mỉ, giải đáp những khúc mắc vấn đề mà bản thân gặp phải, qua đó giúp bản thân hiểu rõ được bản chất công việc mình làm và dùng những giải pháp để giải quyết vấn đề.

2. Giới thiệu bài toán.

Tiết kiệm chi phí và tối ưu hóa hạ tầng CNTT là điều mà các doanh nghiệp quan tâm, đặc biệt là các doanh nghiệp lớn. Một trong những giải pháp hiệu quả để giải quyết vấn đề trên là sử dụng công nghệ ảo hóa. Công nghệ ảo hóa giúp doanh nghiệp nâng cao năng lực bảo vệ dữ liệu, tăng cường khả năng khôi phục hoạt động sau thảm họa, nâng cao tính linh hoạt và cắt giảm chi phí đầu tư cho doanh nghiệp CNTT. Ví dụ như phải cập nhật liên tục các phần mềm và các tính năng mới trên nhiều máy chủ vật lí. Trường đại học Công nghệ đang có hàng ngàn sinh viên theo học, việc trang bị các máy tính phần cứng phục vụ cho sinh viên đã được nhà trường triển khai. Tuy nhiên Trường đại học Công Nghệ cũng đang tìm kiếm một giải pháp ảo hóa để có thể nâng cao tính linh hoạt, cắt giảm những chi phí phần cứng, tránh được nhiều rủi ro để phục vụ sinh viên học tập một cách tốt nhất trong ngữ cảnh sinh viên tuyển vào trường mỗi năm rất đông. Các giảng viên trong bộ môn Mạng và Truyền thông máy tính đã có một ý tưởng về việc xây dựng một hệ thống đám mây ảo hóa được áp dụng trong môi trường của nhà trường, hệ thống này có khả năng tạo những máy tính ảo để sinh viên có tương tác như sử dụng với máy tính vật lí hàng ngày. Cụ thể bài toán cần giải quyết là đưa ra một giải pháp về việc xây

dựng và quản trị một hệ thống hạ tầng cung cấp dịch vụ đám mây (IaaS). Hệ thống này tương tự trong một số phần chức năng cơ bản của các nhà cung cấp đám mây hiện nay như AWS, Azure, Google Cloud. Yêu cầu cơ bản của bài toán là phải xây dựng được một hệ thống private cloud, có thể chạy những máy ảo phục vụ nhu cầu sử dụng của người dùng. Bên cạnh đó người quản trị hệ thống phải có kiến thức trong việc vận hành hệ thống, phân quyền sử dụng tài nguyên cho người dùng, dự án trong hệ thống.

II. Yêu cầu công việc.

Bài toán yêu cầu sinh viên biết cách thiết kế hệ thống, xây dựng được hệ thống đám mây và biết cách quản trị hệ thống đó dựa vào các tiêu chí:

1. Để xây dựng được hệ thống, yêu cầu đầu tiên với sinh viên là viên phải nắm rõ được khái niệm cơ bản về công nghệ ảo hóa – đây là nền tảng để sinh viên hiểu được hệ thống phát triển sử dụng công nghệ ảo hóa như thế nào trong ngữ cảnh của dự án. Sinh viên còn phải thành thạo với các công cụ quan trọng liên quan như phần mềm tạo môi trường ảo hóa (***qemu, libvirt***), phần mềm tự động cấu hình máy ảo khi khởi động (***cloud-init***), phần mềm chỉnh sửa image (***guestfish***) Những phần mềm nói trên cung cấp những dịch vụ hỗ trợ cho tầng ảo hóa của hệ thống và yêu cầu sinh viên phải thành thạo trong việc sử dụng các phần mềm này để có thể hiểu và vận hành hệ thống.
2. Yêu cầu thứ hai là sinh viên phải thành thạo với Linux. Kĩ năng sử dụng Linux là một kĩ năng quan trọng và không thể thiếu đối với một sinh viên CNTT. Trong ngữ cảnh của hệ thống cần phát triển, sinh viên phải có kiến thức kiến thức cơ bản về Linux system, nắm vững và có thể làm việc với Linux networking (bridge networking, zero-configuration, namespace, iptables …). Sinh viên cũng được khuyến thích sử dụng thành thạo trình soạn thảo Vim, lập trình bash và một số ngôn ngữ bậc cao (Python , C++ , Java) để hỗ trợ công việc.

3. Yêu cầu thứ ba là sinh viên phải tìm hiểu về nền tảng mã nguồn mở **Openstack**. Đây là nền tảng công nghệ xây dựng nên hệ thống private cloud. Sinh viên phải nắm vững những khái niệm cơ bản trong Openstack như máy ảo, dự án, người dùng… cùng các dịch vụ liên quan được sử dụng trong Openstack như dịch vụ tính toán, dịch vụ lưu trữ, dịch vụ mạng, dịch vụ xác thực … Sinh viên sẽ sử dụng nền tảng Openstack để thiết kế ra một hệ thống cung cấp dịch vụ đám mây theo yêu cầu của bài toán đưa ra và triển khai xây dựng thành hệ thống có thể sử dụng được.
4. Sau khi xây dựng được hệ thống, sinh viên được yêu cầu phải hiểu được cách vận hành và quản trị hệ thống. Sinh viên phải biết cách tạo những dự án, thêm người dùng và phân quyền cho các người dùng trong dự án trên hệ thống đám mây đã xây dựng. Sinh viên biết cách giới hạn các tài nguyên trong mỗi dự án, tạo được các máy ảo và cung cấp dịch vụ mạng cho người dùng sử dụng. Khi hệ thống xảy ra vấn đề, sinh viên biết cách khắc phục lỗi của hệ thống sinh ra.

III. Thiết kế và xây dựng hệ thống.

1. Khái niệm về công nghệ ảo hóa và công cụ liên quan.

a) Công nghệ ảo hóa

Virtualization hay còn được gọi là ảo hóa, là công nghệ được dung để thiết kế tầng trung gian giữa hệ thống phần cứng và phần mềm chạy trên nó. Ý tưởng của công nghệ ảo hóa là từ một máy chủ vật lý đơn lẻ có thể tạo thành nhiều máy ảo độc lập. Mỗi máy ảo đều có một thiết lập nguồn hệ thống riêng rẽ, sử dụng hệ điều hành riêng và các ứng dụng chạy độc lập. Bản chất của ảo hóa là phân chia ổ đĩa của máy chủ vật lí thành nhiều máy chủ logic chạy hệ điều hành riêng và các ứng dụng chạy độc lập với nhau.

b) QEMU (Quick Emulator)

QEMU là một phần mềm hoạt động như một "trình biên dịch lại" đặc biệt để chuyển mã nhị phân được viết cho một bộ xử lí nhất định thành một bộ xử lí khác (ví dụ như mô phỏng ARM trong máy x86_64). Không chỉ mô phỏng được bộ xử lí, QEMU có thể giả lập các thiết bị đĩa, mạng, PCI, USB, VGA, các cổng nối tiếp / song song.

c) KVM (Kernel Virtualization Machine)

KVM là một module hạt nhân Linux, hiện nay được tích hợp trong các dòng máy Linux chính, nó chuyển nhân Linux thành một hypervisor có khả năng tạo ra các trạng khái khách (máy ảo). KVM phụ thuộc vào bộ xử lí bên dưới của máy như AMD, VT-X (ví dụ như chỉ có thể mô phỏng x86_64 nếu bộ xử lí của máy là x86_64). KVM và QEMU khá tương đồng về chức năng, hệ thống xây dựng sẽ sử dụng kỹ thuật kết hợp hai công nghệ này vào mới nhau để tăng hiệu năng.

Tại sao hệ thống phát triển lại sử dụng cả KVM và QEMU?

KVM là hypervisor loại một nên có thể cung cấp khả năng tăng tốc phần cứng cho máy ảo nhưng nó vẫn cần QEMU để giả lập mọi thứ trông hệ điều hành. QEMU là hypervisor loại hai, nó có thể cài đặt độc lập trên một hệ điều hành và có khả năng giả lập một hệ điều hành khác mà không cần đến KVM, tuy nhiên QEMU không có khả năng tăng tốc phần cứng như KVM. Vì vậy sử dụng QEMU kết hợp với KVM sẽ cho ta ***hiệu năng tốt hơn***, cụ thể KVM sẽ phân quyền truy cập vào CPU và bộ nhớ. QEMU sẽ mô phỏng các tài nguyên phần cứng (đĩa cứng, video, USB). Khi làm việc một mình, QEMU mổ phỏng cả CPU lần phần cứng.

d) Cloud-init

Cloud init là một phần mềm cấu hình cho máy ảo khi khởi động dựa bằng tác tương tác với đĩa ảo tạo nên máy ảo trên môi trường ảo hóa. Ví dụ như chỉnh sửa hostname, cấu hình mạng, chạy các file scripts hoặc thêm ssh key vào bên trong image... Mục đích của cloud-init là giúp người tạo máy ảo bỏ qua việc khởi tạo thủ công trên số lượng nhiều các image, từ đó nâng cao hiệu suất, tránh gây ra lỗi, tiết kiệm thời gian.

e) Guestfish

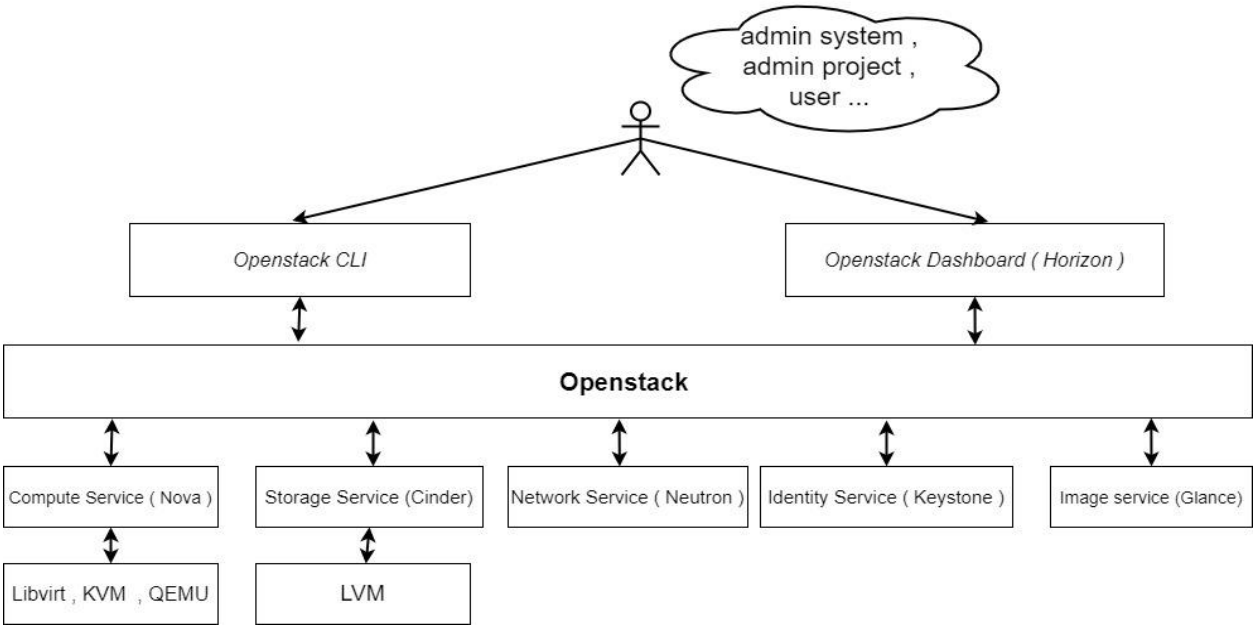
Guestfish là phần mềm cung cấp giao diện shell để người dùng có thể dùng có thể truy cấp vào các đĩa ảo. Nhờ vào công cụ này, người lập trình có thể trực tiếp thay đổi dữ liệu trong tệp đĩa ảo.

f) Openstack

Openstack là một nền tảng mã nguồn mở điện toán đám mây, nó bao gồm một loạt các dự án tương quan cung cấp các thành phần khác nhau cho các giải pháp cơ sở hạ tầng đám mây. Openstack cung cấp một hệ điều hành đám mây

có khả năng mở rộng bao gồm các nhóm dịch vụ tính toán, lưu trữ, mạng, lưu trữ đĩa ảo, xác thực…

2. Thiết kế kiến trúc.
   Tổ chức kiến trúc hệ thống được mô tả bên dưới như sau:



Kiến trúc hệ thống

Người dùng ở đây có thể là người quản trị hệ thống, người quản trị trong một dự án, người dùng trong một dự án.. Mỗi người dùng đều có một vai trò riêng trong hệ thống. Người dùng có thể tương tác với hệ thống qua Openstack API hoặc qua GUI dashboard

Các dịch vụ của nền tảng Openstack xây dựng hệ thống bao gồm:

+ *Identity Service (Keystone)*: là dịch vụ xác thực trong Openstack, keystone giới hạn quyền của người dùng và xác định các tài nguyên mà người đó có quyền truy cập. Khi client gửi yêu cầu truy cập, keystone sẽ kiểm tra thông tin người dùng gửi lên (username, password, key…). Khi xác nhận thành công, keystone sẽ cấp cho client một mã token, người dùng sử dụng token này để gửi cho các yêu cầu tiếp theo mà không cần phải cung cấp thông tin xác thực nữa.

+ *Compute Service (Nova)*: là dịch vụ tính toán trong Openstack cho phép người dùng kiểm soát nền tảng điện toán đám mây cơ sở hạ tầng dưới dạng dịch vụ IaaS. Nó cho phép vận hành các máy ảo, kiểm soát mạng, tương tác với tài nguyên và quản lí quyền truy cập tới đám mây của người dụng và dự án. Dịch vụ này không bao gồm phần mềm ảo hóa, thay vì đó nó tạo ra trình điều khiển tương tác với các công cụ ảo hóa chạy trên hệ điều hành máy chủ thông qua API.

+ *Block Storage Service (Cinder)*: là dịch vụ lưu trữ dữ liệu dạng khối, nó có khả năng cung cấp dữ liệu tồn tại độc lập bên cạnh các máy ảo và không bị mất khi máy ảo bị hủy.

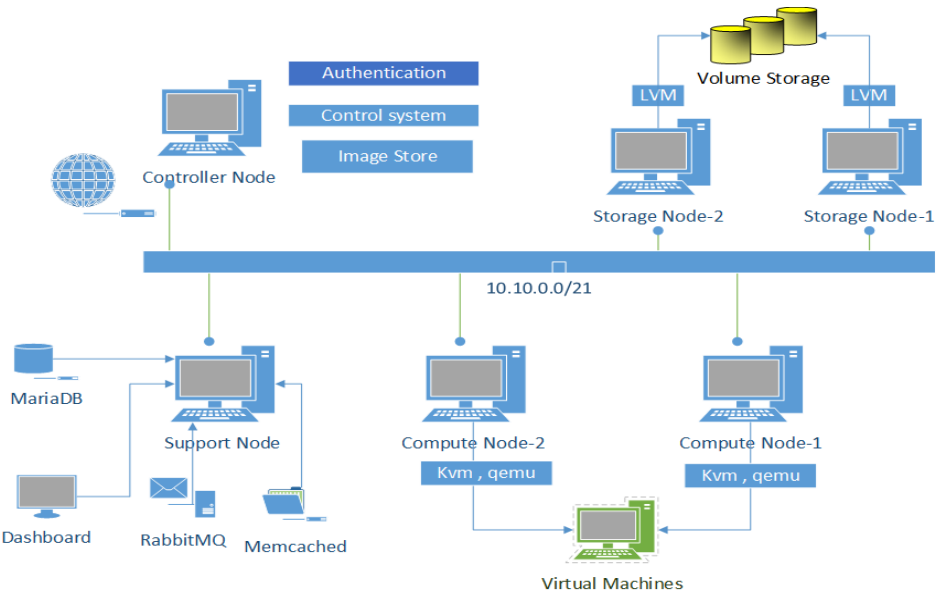
+ *Network Service (Neutron)*: là dịch vụ mạng cung cấp API cho phép người dùng thiết lập và xác định kết nối mạng cũng như địa chỉ trong đám mây. Nó xử lí việc tạo và quản lí cơ sở hạ tầng mạng ảo, bao gồm mạng, switch, subnet, router. Các dịch vụ nâng cao như tường lửa, vpn cũng có thể được sử dụng.

+ *Image Service (Glance)*: là dịch vụ đĩa ảo cho phép người dùng tải và khám phá các dữ liệu được sử dụng với các dịch vụ khác. Hiện nay bao gồm lưu trữ các đĩa ảo và metadata hỗ trợ quá trình khởi tạo máy ảo.

Đánh giá giải pháp

Đây là một giải pháp hiệu quả để xây dựng một hệ thống đám mây. Openstack còn rất nhiều dịch vụ hỗ trợ cho việc tự xây dựng một hệ thống dịch vụ đám mây. Với kiến trúc này, chúng ta có thể xây dựng một hệ thống dịch vụ đám mây cho người dùng với các dịch vụ hỗ trợ cho hệ thống như dịch vụ xác thực (keystone), dịch vụ tính toán (nova), dịch vụ mạng (neutron), dịch vụ lưu trữ (cinder), dịch vụ lưu trữ đĩa ảo (glance), dashboard web-based (Horizon). Các dịch vụ theo dạng component service cho nên khi ta muốn tích hợp thêm các service khác vào để thêm các dịch vụ cho hệ thống thì có thể thêm dễ dàng mà không làm ảnh hưởng tới kiến trúc của hệ thống.

3. Cài đặt hệ thống



Hệ thống triển khai

Hệ thống đã triển khai bao gồm những thành phần sau:

1. *Controller Node:*

   Đây là trung tâm điều khiển hoạt động của hệ thống, có trách nhiệm xác thực người dùng và các quyền truy cập tài nguyên của người dùng. Đây cũng là nơi điều phối mạng, lên lịch các máy ảo và lưu trữ đĩa ảo. Các dịch vụ được cài ở trên máy Controller bao gồm Nova, Cinder, Neutron, Glance, Keystone.
2. *Storage Node:*

   Đây là nơi lưu trữ các dữ liệu dưới dạng khối block (volume, snapshot, snapshot volume). Sử dụng 2 máy storage để tăng tính sẵn sàng cho hệ thống, cụ thể khi một máy storage node bị vấn đề như hỏng hoặc hết đĩa cứng thì hệ thống sẽ sử dụng máy storage còn lại. Dịch vụ được cài đặt trên các máy Storage gồm Cinder.
3. *Compute Node:*

   Đây là nơi khởi tạo vận hành các máy ảo sử dụng các phần mềm ảo hóa cài trên các máy đó: KVM, QEMU. Khi khởi tạo máy ảo, máy compute sao chép image từ image store hoặc sử dụng image volume từ máy Storage, sử dụng metadata từ Glance (các thông số khởi tạo máy ảo), sau đó tạo nên các máy ảo từ những thành phần trên. Các máy ảo được tạo trên các máy Compute và được tương tác qua máy Controller. Việc sử dụng hai máy tính toán để tăng tính sẵn sàng cho hệ thống tương tự như với máy Storage.

4. Support Node:

Đây là máy có nhiệm vụ hỗ trợ nhằm đơn giản và giảm tải cho hệ thống. Các phần mềm được cài trên máy này là:

+ *Rabbit MQ*: hệ thống sử dụng message queue để điều phối thông tin giữa các service với nhau.
+ *Memcached:* dịch vụ xác thực sử dụng cache để lưu tokens.
+ *Mariadb*: nơi lưu trữ thông tin của các service trong hệ thống Openstack.
+ *Etcd*: cơ sở lưu trữ dữ liệu dạng key-value cho phân bố key, lưu trữ cấu hình, lưu vết tình trạng của các service.

IV. Quản trị hệ thống.

1. *Người dùng, dự án và quyền.*

Đầu tiên ta phải hiểu rõ những khái niệm người dùng, dự án và quyền trong hệ thống. Quản trị viên hệ thống là người có thể quản lí các dự án, người dùng và các luật. Dự án là một tổ chức đơn vị trong Openstack. Một dự án gồm một nhóm người dùng và tài nguyên chỉ định thuộc về dự án đó (máy ảo, mạng, storage…)

Một người dùng có thể là thuộc về một hoặc nhiều dự án. Ít nhất một người dùng, một dự án và một vai trò được gắn liên kết với nhau. Một vai trò thể hiện giới hạn sử dụng tài nguyên mà người đó có thể sử dụng hoặc hành động người dùng đó có thể làm trong dự án đó. Vai trò cũng có thể được gắn cho 1 nhóm người dùng trong một dự án thể hiện hành động chung mà nhóm đó có thể làm.

Dưới đây là ví dụ để người quản trị viên quản lí người dùng, dự án và quyền thông qua Openstack API:

a. *Quản lí dự án.*

Là một quản trị viên hệ thống, bạn có thể thêm, xóa, chỉnh sửa các dự án trong hệ thống của mình thông qua CLI hoặc dashboard.

Liệt kê thông tin các dự án trong hệ thống :

```
$ openstack project list
```

Tùy vào nhu cầu sử dụng mà quản trị viên hệ thống có thể tạo nhiều dự án:

```
$ openstack project create --description 'my project' new-project \   --domain default
```

b. *Quản lí người dùng.*

Đối với người dùng, quản trị viên hệ thống có quyền để thêm, xóa, chỉnh sửa người dùng thông qua CLI, dashboard tương tự như dự án, dự án và người dùng được quản lí riêng biệt.

Liệt kê các người dùng trong hệ thống:

```
$ openstack user list
```

Tạo một người dùng mới gắn với một project đã tồn tại:

```
$ openstack user create --project new-project –password pass_word new-user
```

Khi tạo người dùng liên kết với project, quản trị viên cũng có thể thêm người dùng đó vào các project khác thông qua option -set. Một người dùng cũng có thể bị vô hiệu hóa qua option -disable.

c. *Quản lí quyền.*

Openstack sử dụng cơ chế kiểm soát truy cập dựa trên vai trò (RBAC) để quản lí quyền truy cập vào các tài nguyên của hệ thống. Quyền xác định những hành động mà người dùng có thể thực hiện. Theo mặc định có hai vai trò được xác định trước: vai trò thành viên được gắn với người đi thuê và vai trò quản trị để cho phép người dùng không phải quản trị viên quản lí môi trường. Quản trị viên có thể thêm, chỉnh sửa và xóa các quyền.

Liệt kê các quyền trong hệ thống:

```
$ openstack role list
```

Tạo quyền mới tên là "new-role":

```
$ openstack role create new-role
```

Gán quyền new-role cho user "A" trong dự án B:

```
$ openstack role add --user A –project B new-role
```

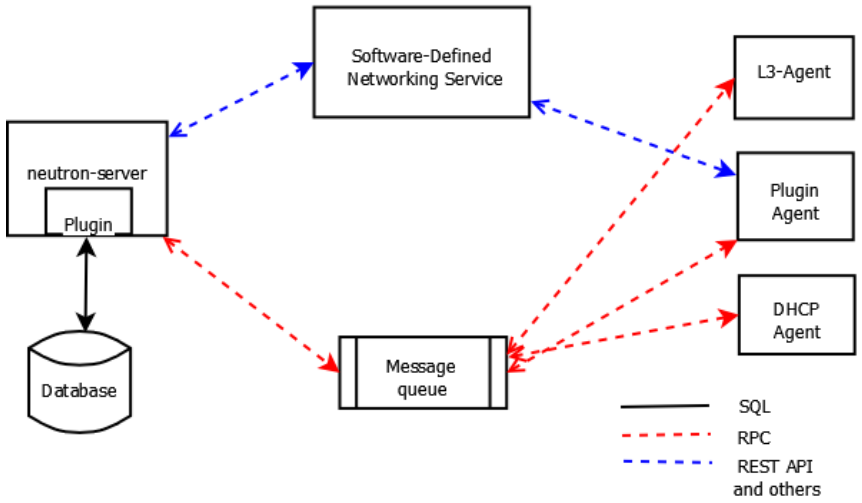
Để kiểm tra thao tác gán quyền cho user đã thành công hay chưa , ta có thể kiểm tra bằng:

```
$ openstack role assignment list --user USER_NAME \ --project PROJECT_ID –names
```

2. *Dịch vụ mạng (Neutron).*

Dịch vụ mạng trong Openstack là một dịch vụ độc lập thường được triển khai một số qui trình trên một số nút. Các quy trình này tương tác với nhau và các dịch vụ Openstack khác. Quy trình chính của dịch vụ mạng là neutron-server, một luồng điều khiển chạy python để điều phối các yêu cầu của người dùng đến các plugin để xử lí.

Dịch vụ mạng cho phép tạo và gắn các giao diện của thiết bị tới mạng. Neutron bao gồm các thành phần:



+ *Neutron server:* là nơi chấp nhận và định tuyến các lời gọi api tới những plugin network tương ứng để thực thi hành động.
+ *Openstack Networking plug-in and agents:* là dịch vụ chạy trên mỗi nút tính toán để quản lí cấu hình chuyển mạch ảo cục bộ, dịch vụ này có quyền truy cập vào hang đợi tin nhắn và phụ thuộc vào plugin đang sử dụng. Nó có chức năng gắn và gỡ các port, tạo mạng, mạng con, cung cấp địa chỉ IP, L3 agents, DHCP agents…
+ *Message Queue*: định tuyến thông tin giữa neutron server và các agents.

+ *DHCP agent*: cung cấp dịch vụ DHCP cho người dùng và chịu trách nhiệm duy trì cấu hình DHCP.
+ *L3 agent (neutron L3 agent)*: cung cấp chuyển tiếp L3/ NAT để truy cập mạng ngoài của máy ảo trên mạng nội bộ của người dùng.
+ *Network provide service:* cung cấp những dịch vụ mạng bổ sung cho người sử dụng và tương tác với những dịch vụ mạng khác.

Quản trị viên hệ thống hoặc quản trị viên của một dự án đều có quyền thêm, xóa, chỉnh sửa thao tác với các thành phần mạng như network, subnet, router, interface…

Để tạo một mạng flat để kết nối với mạng vật lí bên ngoài:

```
$ openstack network create  --share --external \
  --provider-physical-network provider \
  --provider-network-type flat provider
```

Option –-share cho phép các dự án sử dụng mạng lưới ảo.

Option –external định nghĩa mạng lưới ảo được cho ra ngoài, nếu chỉ muốn tạo mạng riêng thì có thể sử dụng –internal để thay thế, giá trị mặc định là internal.

Option –provider-physical-network provider và --provider-network-type flat kết nối mạng ảo flat tới mạng vật lí qua interface mạng.

Một lưu ý khi khởi tạo mạng flat là khi kết nối với mạng vật lí bên ngoài, trong quá trình tạo máy ảo, Openstack sẽ tạo ra một virtual bridge và cố gắng để thêm mạng vật lí bên ngoài để các máy ảo được kết nối thông qua bridge đó để thông ra bên ngoài. Địa chỉ của bridge sẽ lấy từ địa chỉ mạng vật lí bên ngoài nhưng nếu card mạng vật lí đó có địa chỉ IPv6 thì không chuyển được địa chỉ, sẽ dẫn đến thất bại khi tạo máy ảo, vì vậy giải pháp tạm thời của em là tắt IPv6 đi.

Quản trị viên có thể quản lí các subnet có trong mạng vừa tạo:

```
 $ openstack subnet create --network provider \
 --allocation-pool start=START_IP_ADDRESS,end=END_IP_ADDRESS \
 --dns-nameserver DNS_RESOLVER --gateway PROVIDER_NETWORK_GATEWAY \
 --subnet-range PROVIDER_NETWORK_CIDR provider
```

Subnet được tạo ra gồm một range các IP có thể sử dụng trong mạng, DNS, default gateway, CIDR…

Để kết nối giữa mạng ngoài và mạng trong của người dùng, quản trị viên có thể tạo những router và kết nối hai mạng với nhau:

```
$ openstack router create router
```

Gắn router vào một subnet:

```
$ openstack router add subnet [router] [subnet]
```

Cài đặt gateway trên router kết nối với mạng ngoài flat:

```
$ openstack router set router --external-gateway provider
```

3. *Dịch vụ tính toán (Nova)*

Khi thao tác với dịch vụ tính toán, ta sẽ gặp những khái niệm như flavor, keypair, security groups … Chúng là những cấu hình cơ bản cho việc tạo những máy ảo.

Flavor: định nghĩa khả năng tính toán, bộ nhớ và lưu trữ cho một máy ảo, hiểu một cách đơn giản, flavor là một cấu hình phần cứng cho một máy ảo. Nó định nghĩa bộ nhớ ảo của một máy ảo khi khởi động lên.

Keypair: là public key được sử dụng để truy cập vào máy ảo qua giao thức ssh (security shell). Người dùng sẽ cung cấp keypair cho người quản trị dự án để người quản trị dự án thiết lập keypair cho máy ảo, sau khi khởi tạo xong thì người dùng có thể truy cập vào máy ảo.

Security groups: là những bộ luật lọc IP được áp dụng cho máy ảo, định nghĩa cách thức truy cập mạng vào các máy ảo. Ví dụ như cho phép kết nối vào ở cổng 22 hoặc kết nối ra ở cổng 20 , …vv

Người quản trị có thể quản lí: thêm, chỉnh sửa và xóa các flavor, keypair, security group. Để tạo một flavor, ta dùng câu lệnh sau, trong đó option -id cho biết id của flavor đó là gì, mặc định auto thì openstack sẽ sinh ra một id bất kì, số lượng cpus, ram và ổ cứng của máy ảo.

```
$ openstack flavor create --id auto --vcpus [cpus] --ram [mbs] --disk [gbs] [name-flavor]
```

Đầu tiên người dùng, người thuê phải cung cấp cho quản trị viên keypair để sau khi việc tạo máy ảo hoàn thành, người dùng có thể truy cập vào máy ảo thông qua giao thức ssh (secure shell). Người dùng không thể truy cập vào máy ảo qua console vì chức năng này chỉ dành cho các thành viên thuộc group admin.

Sinh một ssh-key và tạo keypair, lưu trữ trên hệ thống:

```
$ ssh-keygen -q -N ""

# create key pair
$ openstack keypair create --public-key ~/.ssh/id_rsa.pub mykey
```

Mặc định khi tạo một dự án, dự án đó có một security group mặc định nhưng người quản trị hệ thống / dự án có quyền quản lí: thêm, chỉnh sửa hoặc xóa các security groups. Có thể sử dụng nhiều security groups, hệ thống sẽ kiểm tra các kết nối thỏa mãn toàn bộ các security group được thiết lập trong quá trình tạo máy ảo.

Thêm luật vào security groups đã tồn tại:

Cho phép giao thức ping (ICMP)

```
$ openstack security group rule create --proto icmp default
```

Cho phép giao thức ssh (Permit secure shell)

```
$ openstack security group rule create --proto tcp --dst-port 22 default
```

Người quản trị, người dùng có quyền để tạo một máy ảo dựa vào cấu hình flavor, image, mạng, sercurity groups, keypair theo mong muốn của mình.

Tạo một máy ảo với image cirros:

```
$ openstack server create --flavor m1.nano --image cirros \
  --nic net-id=PROVIDER_NET_ID --security-group default \
  --key-name mykey [instance-name]
```

Kiểm tra máy ảo vừa được tạo:

```
$ openstack server show [instance-name]
```

Máy ảo được tạo sẽ hiện lên trạng thái khởi tạo của chính nó, nếu trạng thái là success thì có nghĩa máy ảo đã được tạo thành công, nếu trạng thái là error thì có nghĩa là máy ảo khởi tạo thất bại .

Quản trị viên cũng có thể truy cập máy ảo qua VNC console:

```
$ openstack console url show [instance-name]
```

Một máy ảo được tạo trong mạng riêng của người dùng nên không thể kết nối tới mạng bên ngoài để ra ngoài internet. Một cách để cho phép kết nối ra bên ngoài là gắn một địa chỉ nổicho máy ảo đó, lúc đó máy ảo đó sẽ sử dụng hai địa chỉ: một địa chỉ trong và một địa chỉ ngoài.

```
$ openstack server add floating ip selfservice-instance 203.0.113.104
```

Ngoài ra còn có nhiều thao tác cơ bản của một người quản trị đối với một máy ảo như bật, tắt, khởi động lại hoặc những thao tác nâng cao như di chuyển một máy ảo từ dự án này sang dự án khác, tạo máy ảo thông qua một volume image, tạo snapshot của máy ảo đó.

4. *Dịch vụ lưu trữ (Cinder).*

Cinder là dịch vụ lưu trữ dữ liệu dưới dạng khối (block). Mặc định dịch vụ lưu trữ backend mà cinder sử dụng là LVM (Logical volume manager) với nhân linux. Cinder bao gồm những service sau:

+ *cinder api*: chuyển tiếp các yêu cầu api tới cinder-scheduler để khởi tạo các volume.

+ *cinder scheduler*: quyết định nơi sẽ tạo các volume và chuyển yêu cầu đến cinder-volume.

+ *cinder volume*: nhận những thông tin yêu cầu từ cinder-api và cinder-scheduler, định tuyến những yêu cầu đó tới những dịch vụ lưu trữ backend.

Trong quá trình tạo máy ảo, máy ảo đó sẽ được tạo trên các compute node những nơi lưu trữ dữ liệu lại ở các storage node để tiết kiệm tài nguyên cũng như giảm thiểu rủi ro cho người dùng. Những khối dữ liệu được gọi là các volume, người quản trị có thể quản lí: thêm, chỉnh sửa và xóa các volume.

Để tạo một volume sử dụng câu lệnh sau:

```
$ openstack volume create --size 1 volume1
```

Liệt kê danh sách các volume:

```
$ openstack volume list
```

Trong quá trình sử dụng máy ảo, người dùng có nhu cầu sử dụng thêm các khối dữ liệu thì có thể thêm các volume vào máy ảo:

```
$ openstack server add volume INSTANCE_NAME VOLUME_NAME
```

Bên cạnh đó nếu như một volume trong máy ảo sắp bị đầy dữ liệu, người quản trị có thể mở rộng linh hoạt size của volume đó:

```
$openstack volume extend [volume_id] size
```

Dịch vụ lưu trữ cinder cũng cung cấp những chức năng như tạo image từ một, tạo một snapshot volume hoặc migrate một volume.

5. *Dịch vụ image (Glance)*

Glance là image service cung cấp khả năng tìm kiếm, đăng ký, thu thập các image cho máy ảo. Trong dịch vụ image, các image được lưu dưới dạng các template để sử dụng khởi tạo các máy ảo.

Hệ thống cloud đang sử dụng chỉ cho phép quản trị viên hệ thống có quyền upload các image lên image store. Quản trị viên hệ thống có thể tải image thông

qua câu lệnh image-create, quản trị viên cũng có quyền để quản lí, chỉnh sửa image data, tạo những bản sao, snapshot của image đó. Một image được tải lên thì không thể chỉnh sửa được.

Liệt kê danh sách image:

```
$ openstack image list
```

Xem thông tin chi tiết của một image:

```
$ openstack image show [image_id]
```

Ta có thể tạo một image từ một image khác, ví dụ như tạo image centos 6 mới từ image centos cũ.

```
$ openstack image create --disk-format qcow2 --container-format bare \
  --public --file ./centos63.qcow2 centos63-image
```

6. *Giám sát tình trạng hệ thống.*

Do hệ thống gồm rất nhiều các dịch vụ, ta cần phải có một cách để có thể kiểm soát và nắm được tình trạng của hệ thống. Khi hệ thống xảy ra vấn đề, việc kiểm tra tình trạng của các service trong hệ thống thủ công làm mất thời gian, quản trị viên có thể tiết kiệm thời gian, tăng hiệu năng để lần vết vấn đề và giám sát tình trạng của hệ thống:

- Kiểm tra trạng thái các compute service, kết quả hiện ra những service chạy trên compute, tình trạng và trạng thái của chúng để quản trị viên có thể kiểm soát được.

```
$ openstack compute service list
```

```
+----+----------------+---------------+----------+---------+-------+----------------------------+
| ID | Binary         | Host          | Zone     | Status  | State | Updated At                 |
+----+----------------+---------------+----------+---------+-------+----------------------------+
|  1 | nova-conductor | os-controller | internal | enabled | up    | 2020-10-08T04:30:04.000000 |
|  5 | nova-scheduler | os-controller | internal | enabled | up    | 2020-10-08T04:29:59.000000 |
|  7 | nova-compute   | os-compute    | nova     | enabled | up    | 2020-10-08T04:29:59.000000 |
|  8 | nova-compute   | mmt-66        | nova     | enabled | up    | 2020-10-08T04:29:58.000000 |
+----+----------------+---------------+----------+---------+-------+----------------------------+
```

- Kiểm tra trạng thái các neutron-service, kết quả hiện ra những service network, tình trạng và trạng thái của chúng để quản trị viên có thể kiểm soát được.

```
$openstack network service list
```

```
+--------------------------------------+--------------------+--------------+-------------------+-------+-------+----------------------------+
| ID                                   | Agent Type         | Host         | Availability Zone | Alive | State | Binary                     |
+--------------------------------------+--------------------+--------------+-------------------+-------+-------+----------------------------+
| 27c38250-068a-4b12-83e0-d7aab85417dd | Linux bridge agent | os-compute   | None              | :-)   | UP    | neutron-linuxbridge-agent  |
| 31d6704b-dc83-4585-b9e5-3db804658ca1 | DHCP agent         | os-controller| nova              | :-)   | UP    | neutron-dhcp-agent         |
| 5fc17b21-eb41-49a3-b31d-e395ac114fa5 | Linux bridge agent | os-controller| None              | :-)   | UP    | neutron-linuxbridge-agent  |
| 8c013b7e-da95-43e1-838b-027b364a7826 | Metadata agent     | os-controller| None              | :-)   | UP    | neutron-metadata-agent     |
| d0a484d8-0ec9-4575-86dd-e8596df36fa8 | L3 agent           | os-controller| nova              | :-)   | UP    | neutron-l3-agent           |
| fa64b0c9-c621-4899-9532-927aeab43cfc | Linux bridge agent | mmt-66       | None              | :-)   | UP    | neutron-linuxbridge-agent  |
+--------------------------------------+--------------------+--------------+-------------------+-------+-------+----------------------------+
```

- Kiểm tra sử dụng của các dự án, đây là cách để kiểm tra dự án đã sử dụng tài nguyên của hệ thống, ví dụ như thông qua số máy ảo đã tạo, tổng số giờ ram, cpu, disk đã sử dụng

```
$ openstack usage list
```

```
+----------+---------+--------------+-----------+---------------+
| Project  | Servers | RAM MB-Hours | CPU Hours | Disk GB-Hours |
+----------+---------+--------------+-----------+---------------+
| mmt      |       6 |  11347232.72 |   5303.73 |           0.0 |
| DoPrj    |       6 |    797236.47 |    433.83 |           0.0 |
| crawling |      10 |   2323108.82 |   1516.16 |           0.0 |
| gns3     |       5 |    427462.35 |    104.6  |           0.0 |
| i2g      |      11 |   6074286.21 |   3249.81 |           0.0 |
+----------+---------+--------------+-----------+---------------+
```

- Trên các máy compute, kiểm soát những tài nguyên mà các máy ảo đã sử dụng.

```
$ openstack host show [host_name]
```

```
+--------+----------------------------------+-----+-----------+---------+
| Host   | Project                          | CPU | Memory MB | Disk GB |
+--------+----------------------------------+-----+-----------+---------+
| mmt-66 | (total)                          |  80 |    257429 |      49 |
| mmt-66 | (used_now)                       |  19 |     34816 |       0 |
| mmt-66 | (used_max)                       |  19 |     34304 |       0 |
| mmt-66 | 75d62bc67a024c14b6d3d9c920627bb7 |   4 |      5120 |       0 |
| mmt-66 | 9c516f73cd4c4793883e95568ea1b8ec |   7 |     12800 |       0 |
| mmt-66 | 78c4bea6929444f99ffcbb45e9bc0383 |   4 |      8192 |       0 |
| mmt-66 | 2e17440f89d643bfb89f06e24f80c7d7 |   4 |      8192 |       0 |
+--------+----------------------------------+-----+-----------+---------+
```

- Trên các máy storage, kiểm soát số lượng volume đang sử dụng, tài nguyên còn lại của host storage đó.

```
+-----------------------------------------------------------+------+--------------+----------------+----------------------------------+------------+
| reference                                                 | size | safe_to_manage | reason_not_safe | cinder_id                        | extra_info |
+-----------------------------------------------------------+------+--------------+----------------+----------------------------------+------------+
| {'source-name': 'volume-bb9a35aa-17c2-434a-8f76-4b2e8aad76d4'} | 5    | False        | already managed | bb9a35aa-17c2-434a-8f76-4b2e8aad76d4 | -          |
| {'source-name': 'volume-92a36651-503e-4b97-ad00-e19007b0c9a5'} | 1000 | False        | already managed | 92a36651-503e-4b97-ad00-e19007b0c9a5 | -          |
| {'source-name': 'volume-6911fef9-6880-4bd9-ab25-15bc0255386f'} | 8    | True         | -               | -                                | -          |
| {'source-name': 'cinder-volumes-pool'}                    | 1415 | False        | volume in use   | -                                | -          |
+-----------------------------------------------------------+------+--------------+----------------+----------------------------------+------------+
[root@os-controller centos]#
```

7. *Oslo policy.*

Oslo policy là một thư viện python để các dịch vụ Openstack sử dụng định nghĩa các chính sách RBAC (Role Base Access Control). Chính sách này xác định một người dùng có thể truy cập tới tài nguyên tương ứng với quyền của người đó. Thư viện này chứa việc triển khai ngôn ngữ chính sách mà người dung tạo dịch vụ sẽ sử dụng để tạo các giá trị mặc định thích hợp về những gì được cho phép bởi mỗi endpoint của dịch vụ. Ngôn ngữ mà chính sách oslo-policy dựa vào định dạng json hoặc yaml.

Mỗi dịch vụ trong openstack đều chính sách riêng của nó và có trách nhiệm định nghĩa ra chính sách của chính nó. Ví dụ như ở dịch vụ xác thực keystone sẽ định nghĩa chính sách của nó qua một file cấu hình. File này đề ra những chính sách về quyền người dùng, những dịch vụ mà người đó có thể truy cập tới. Trong quá khứ, các chính sách được mặc định lưu ở trong thư mục cài đặt của mỗi dịch vụ. Sau đó các nhà phát triển đã tích hợp chính sách mặc định vào trong mã nguồn để người vận hành có trải nghiệm tốt. Nếu người vận hành không cần thay đổi chính sách mặc định thì bọn họ không cần thay đổi bất cứ điều gì, mọi thứ sẽ được vận hành đúng như với chính sách mặc định. Nếu người vận hành có nhu cầu thay đổi chính sách thì họ chỉ cần tạo ra các file chính sách và ghi các giá trị cần thay đổi vào file đó, khi đó các dịch vụ của openstack sẽ đọc những giá trị mới đó và ghi đè vào các chính xác cũ. Với cơ chế như thế sẽ giúp người vận hành dễ quan sát, chỉnh sửa và bảo trì hệ thống.

Cách viết policy:

Mỗi luật chính sách xác định những gì cần được thực hiện để cho phép hoạt động và chúng được định nghĩa theo định dạng**: "< target > ": "<rule>"**

Target có thể là các aliases (bí danh) hoặc là actions (hành động):

+ actions: lời gọi API hoặc các toán tử (and, or, not...) Ví dụ: create user.

+ aliases: sử dụng lại rule, các rule chứa các toán tử dài.

Ví dụ: ghi thêm vào file nova.policy trong /etc/nova/

```
"os_compute_api:servers:create": "rule:admin"
```

Chính sách này có ý nghĩa chỉ những người nào có role là admin mới có quyền để tạo máy ảo. Khi tạo máy ảo, dịch vụ tính toán nova sẽ kiểm tra người dùng tạo có đủ quyền để tạo máy ảo hay không thông qua chính sách này.

8. *Thiết lập hạn ngạch.*

Tài nguyên trong hệ thống hạ tầng đám mây bao gồm những ổ đĩa, cpu, ram, địa chỉ IP, số lượng các máy ảo, router. Việc qui hoạch, sử dụng các tài nguyên này một cách hợp lí là một việc quan trọng trong qui trình quản trị hệ thống, nếu người quản trị không biết cách phân bố tài nguyên hợp lí cho các domain và các dự án thì tài nguyên sẽ bị lãng phí, không tận dụng được tối đa và dẫn đến hậu quả tài nguyên của hệ thống sẽ bị cạn kiệt.

Quản trị viên có quyền thiết lập hạn ngạch cho một dự án. Mỗi dự án được cấp phát tài nguyên và người dùng dự án được cấp phát quyền truy cập để sử dụng các tài nguyên này. Một tập hợp hạn ngạch bao gồm số lượng các cpu, version, số ram, floating ip có thể chỉ định cho người thuê (hoặc dự án) và người thuê – người dùng. Ví dụ ta có thể đặt hạn ngạch cho một dự án A với hạn ngạch số lượng cpu tối đa là 20 cpu, số lương máy ảo tạo tối đa là 10 máy.

Quản trị viên cũng có thể tùy chỉnh hạn ngạch mặc định được áp dụng sẵn cho hệ thống. Ví dụ muốn thay đổi hạn ngạch mặc định cho dịch vụ lưu trữ, truy cập vào file cinder.conf và tùy chỉnh các thuộc tính hạn ngạch ở field [quota].

Việc điều chỉnh hạn ngạch không có sẵn trên Openstack Dashboard, vì thế để điều chỉnh được hạn ngạch thì ta có một cách là sử dụng qua CLI:

```
openstack quota set --QUOTA_NAME QUOTA_VALUE PROJECT_ID
```

Trong đó:

--QUOTA_NAME: tên các thuộc tính

--QUOTA_VALUE: giá trị thay đổi

--PROJECT_ID: dự án cần chỉnh hạn ngạch

Dưới đây là bảng các tên các giá trị thuộc tính quota thường sử dụng để thiết lập hạn ngạch cho dự án

Compute settings	
--core	Number core
--fixed-ips	Number fixed ips
[--floating-ips	num-floating-ips
[--injected-file-size	injected-file-bytes
[--injected-files	num-injected-files
[--instances	num-instances
[--key-pairs	num-key-pairs
[--properties	num-properties
[--ram	ram-mb

[--server-groups	num-server-groups
[--server-group-members	num-server-group-members
Block Storage settings	
[--backups	new-backups
[--backup-gigabytes	new-backup-gigabytes
[--gigabytes	new-gigabytes
[--per-volume-gigabytes	new-per-volume-gigabytes
[--snapshots	new-snapshots
[--volumes	new-volumes
[--volume-type	volume-type
# Network settings	
[--floating-ips	num-floatingips
--secgroup-rules	num-security-group-rules
[--secgroups	num-security-groups
[--networks	num-networks
[--subnets	num-subnets
[--ports	num-ports
[--routers	num-routers
[--rbac-policies	num-rbac-policies
[--subnetpools	num-subnetpools

Ví dụ thiết lập hạn ngạch cho dự án ABC, tối đa được sử dụng 20 core cpu, 50GB lưu trữ và 10 máy ảo:

```
$ openstack quota set --core 20 --gigabytes 50 --instances 10 ABC
```

V. Kết quả đạt được và hướng phát triển tiếp theo.

1. *Kết quả đạt được.*

Sau thời gian thực tập với các thầy, được triển khai dự án đã giúp em có điều kiện được va chạm với thực tế, có thể vận dụng kiến thức mình đã học ở nhà trường để sử dụng giải quyết vấn đề thực tế là điều rất quan trọng, cần thiết và bổ ích cho những sinh viên sắp ra trường làm việc như em. Kết thúc quá trình thực tập hè, em đã luyện được cho bản thân khả năng tìm tòi đọc hiểu công nghệ, nắm chắc được những khái niệm nền tảng ảo hóa và các phần mềm chuyên dụng liên quan, đồng thời em đã được cải thiện thêm kĩ năng về hệ điều hành Unix, có thêm tư duy về lập trình. Quan trọng hơn là được học hỏi tiếp thu những kiến thức kinh nghiệm từ các thầy để mở rộng vốn tri thức của mình đã khiến em nâng cao trình

độ hiểu biết và thực tế về công việc. Bên cạnh đó đồng thời em cũng học hỏi thêm được tác phong làm việc chuyên nghiệp, các giao tiếp ứng xử … góp phần để bản thân trở nên tốt hơn. Tuy nhiên vì thời gian có hạn, bản thân vẫn còn những thiếu sót, chưa có nhiều kĩ năng, kiến thức thực tế cho nên kết quả đạt được vẫn chưa đáp ứng được kết quả kế hoạch ban đầu đã định, em vẫn mong muốn được nhận ý kiến đóng góp từ thầy để em có thể hoàn thiện bản thân tốt hơn.

2. *Hướng phát triển tiếp theo*.

Hệ thống phát triển mặc dù đáp ứng được những yêu cầu mà bài toán đặt ra nhưng bản thân em thấy vẫn còn một số vấn đề trong hệ thống mà ta có thể cải thiện. Openstack bao gồm rất nhiều các dịch vụ với khả năng đa dạng khác nhau, tùy vào mục đích sử dụng mà ta có thể cài đặt thêm để mở rộng dịch vụ của Openstack, ví dụ như OVS Swift (dịch vụ lưu trữ dữ liệu dưới dạng object), Heat (dịch vụ monitor tình trạng các service trong hệ thống). Bên cạnh đó trong quá trình thực hành với Openstack Dashboard, em thấy bảng điều khiển có một chút khó sử dụng cho những người dùng mới sử dụng nó. Vì vậy nên có một số cải tiến để dashboard có thể sử dụng dễ hơn. Một hướng phát triển khác là vấn đề giám sát tài nguyên có trong Openstack, việc kiểm tra mỗi dự án và tính toán thủ công bằng tay rất mất thời gian để kiểm soát xem hệ thống chúng ta còn lại bao nhiêu tài nguyên: ram, cpu, ổ đĩa. Để khắc phục hạn chế này em đưa ra đề xuất tạo ra một dịch vụ có thể truy xuất được dữ liệu tài nguyên trong Openstack, qua đó đưa ra một thống kê khách quan cho quản trị viên về tổng quan tài nguyên hệ thống, giúp quá trình quản trị trở nên dễ dàng hơn. Một vấn đề tiếp theo là việc sử dụng địa chỉ IP internet ngoài đang là một phương án tạm thời nhưng không phải là một phương án tối ưu vì khi muốn truy cập từ internet vào một máy ảo trong mạng private, ta phải gán cho nó một floating ip trong mạng ngoài để có thể truy cập từ bên ngoài vào trong mạng trong đó, như vậy sẽ rất nhanh làm cạn kiệt bể địa chỉ IP ngoài thông với internet, vì vậy em đã sử dụng giải pháp là thiết lập một mạng trong khác làm bể IP để gán địa chỉ floating IP cho các máy ảo đó , sau đó thực hiện cơ chế NAT để có thể kết nối internet từ ngoài vào máy ảo, tuy nhiên việc thực hiện đó vẫn là thủ công, không kiểm soát được các cổng mở ra để thực hiện NAT, cho nên việc đưa ra một dịch vụ PaaS giúp mở cổng

NAT và quản lí các cổng đang được mở để kết nối internet từ ngoài vào mạng trong là quan trọng (Proxy as a Service)

VI. Tài liệu tham khảo

Trong quá trình thực tập, em đã sử dụng những tài liệu tham khao bên dưới để hỗ trợ công việc của mình:

1. Tài liệu sử dụng Virsh tool : https://libvirt.org/sources/virshcmdref/html/
2. Tài liệu về định dạng XML format để tạo các domain: https://libvirt.org/format.html
3. Libvirt Networking: https://wiki.libvirt.org/page/Networking
4. Tài liệu về cloud-init: https://cloudinit.readthedocs.io/en/latest/
5. Tài liệu về qemu-tools: https://qemu.readthedocs.io/en/latest/
6. Tài liệu về Openstack phiên bản train : https://docs.openstack.org/train/index.html
7. Quản trị openstack https://docs.openstack.org/ussuri/admin/
8. Hướng dẫn cài openstack train : https://www.serverworld.info/en/note?os=CentOS_7&p=openstack_train2&f=2