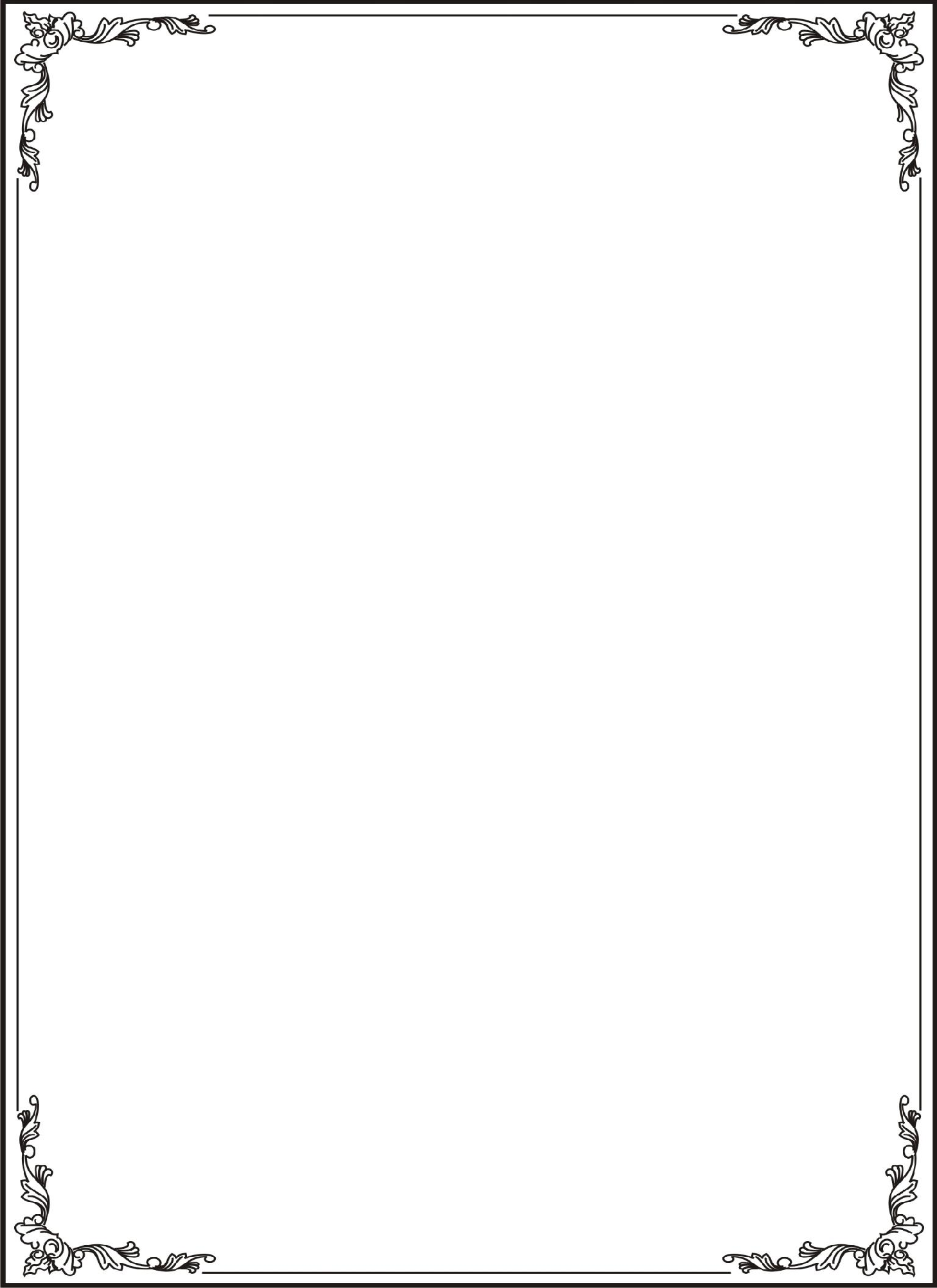
**BỘ TÀI NGUYÊN VÀ MÔI TRƯỜNG**



**TRƯỜNG ĐẠI HỌC TÀI NGUYÊN VÀ MÔI TRƯỜNG TP.HCM**

**KHOA: HỆ THỐNG THÔNG TIN VÀ VIẾN THÁM**



**BÁO CÁO ĐỒ ÁN MÔN BẢO MẬT MẠNG MÁY TÍNH VÀ HỆ THỐNG**

NGHIÊN CỨU GIẢI PHÁP BẢO VỆ DỮ LIỆU ỨNG DỤNG TRÊN NỀN TẢNG OPENSTACK

Giảng viên hướng dẫn: **ThS. Phạm Trọng Huynh**

Thành viên của nhóm:

**Thái Bình Dương 0850080013**

**Nguyễn Thanh Tùng 0850080052**

**Đoàn Dzoãn Tường Anh 0850080002**

Lớp: **08-ĐH-CNPM**

**TP. Hồ Chí Minh, tháng 3 năm 2023**

**MỞ ĐẦU**

Hiên nay các mối đe dọa bảo mật không ngừng phát triển và các yêu cầu tuân thủ ngày càng trở nên phức tạp. Các tổ chức phải tạo ra một chính sách bảo mật thông tin toàn diện để giải quyết cả hai thách thức. Chính sách bảo mật thông tin giúp phối hợp và thực thi chương trình bảo mật và truyền đạt các biện pháp bảo mật cho bên thứ ba và kiểm toán viên bên ngoài.

Để điều hành một tổ chức an toàn, thành công, các nhà lãnh đạo CNTT cần có các chính sách được ghi chép đầy đủ để giải quyết các vấn đề bảo mật tiềm ẩn và giải thích cách quản lý các vấn đề này trong công ty. Các chính sách này cũng là nền tảng cho quy trình kiểm tra CNTT, vì chúng thiết lập các biện pháp kiểm soát có thể được kiểm tra và xác thực.

Mọi tổ chức cần phải có các biện pháp và chính sách bảo mật để bảo vệ dữ liệu của mình. Cùng với các kế hoạch quản lý rủi ro và mua hợp đồng bảo hiểm, việc có một chính sách bảo mật thông tin hiệu quả (và luôn cập nhật chính sách này) là một trong những cách tốt nhất và quan trọng nhất để bảo vệ dữ liệu, nhân viên, khách hàng và doanh nghiệp.

Nhận thức được tầm quan trọng của các chính sách bảo mật hệ thống thông tin, nhóm chúng em đã chọn đề tài “**Nghiên cứu giải pháp bảo vệ dữ liệu ứng dụng trên nền tảng OpenStack**”

**Nhận xét**

……………………………………………………………………………………………

…………………………………………………………………………………………….

…………………………………………………………………………………………….

…………………………………………………………………………………………….

…………………………………………………………………………………………….

…………………………………………………………………………………………….

…………………………………………………………………………………………….

…………………………………………………………………………………………….

…………………………………………………………………………………………….

…………………………………………………………………………………………….

…………………………………………………………………………………………….

…………………………………………………………………………………………….

…………………………………………………………………………………………….

…………………………………………………………

…………..,ngày….tháng….năm……

GIẢNG VIÊN

*(ký tên)*

**MỤC LỤC**

[**CHƯƠNG I. TỔNG QUAN 1**](#_Toc135573818)

[**1.1 Tổng quan đề tài 1**](#_Toc135573819)

[**1.2 Lý do chọn đề tài 1**](#_Toc135573820)

[**1.3 Phạm vi của đề tài 1**](#_Toc135573821)

[**CHƯƠNG II. CƠ SỞ LÝ THUYẾT 2**](#_Toc135573822)

[**2.1 OpenStack là gì? 2**](#_Toc135573823)

[**2.2 OpenStack dùng để làm gì? 2**](#_Toc135573824)

[**2.3 Nguyên lý hoạt động 3**](#_Toc135573825)

[**2.4 Cấu trúc hệ thống OpenStack 3**](#_Toc135573826)

[**2.5 Các Components trong OpenStack 4**](#_Toc135573827)

[**2.6 Các thành phần chức năng chính của OpenStack 6**](#_Toc135573828)

[**2.7 Các module chính được cung cấp trong OpenStack 6**](#_Toc135573829)

[**2.7.1 Openstack identity module 6**](#_Toc135573830)

[**2.7.2 Openstack compute module 7**](#_Toc135573831)

[**2.7.3 Openstack network module 7**](#_Toc135573832)

[**2.7.4 Openstack storage module 8**](#_Toc135573833)

[**2.7.5 Openstack image module 9**](#_Toc135573834)

[**2.7.6 Openstack dashboard module 9**](#_Toc135573835)

[**2.8 Các Framework, API 10**](#_Toc135573836)

[**2.8.1 Framework 10**](#_Toc135573837)

[**2.8.2 API 11**](#_Toc135573838)

[**2.9 Triển khai làm việc với Dashboard (Horizon) trên OpenStack 11**](#_Toc135573839)

[**2.10 Giới thiệu về dự án Karbor 12**](#_Toc135573840)

[**2.10.1 Sứ mệnh của Karbor 12**](#_Toc135573841)

[**2.10.2 Khái niệm Karbor 12**](#_Toc135573842)

[**2.10.3 Các thuật ngữ trong Karbor 13**](#_Toc135573843)

[**2.10.4 Điểm nổi bật của Karbor 15**](#_Toc135573844)

[**CHƯƠNG III. CÀI ĐẶT THỬ NGHIỆM 20**](#_Toc135573845)

[**3.1 Cài đặt và cấu hình OpenStack 20**](#_Toc135573846)

[**3.2 Giao diện Openstack Dashboard 24**](#_Toc135573847)

[**3.3 Bảo vệ dữ liệu trên Openstack 32**](#_Toc135573848)

[**CHƯƠNG IV. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN 41**](#_Toc135573849)

[**4.1 Kết luận 41**](#_Toc135573850)

[**4.1.1 Ưu điểm 41**](#_Toc135573851)

[**4.1.2 Nhược điểm 41**](#_Toc135573852)

[**4.1.3 Cách khắc phục 41**](#_Toc135573853)

[**4.2 Hướng phát triển 41**](#_Toc135573854)

[**TÀI LIỆU THAM KHẢO 42**](#_Toc135573855)

**DANH MỤC HÌNH ẢNH**

[Hình 1 Cấu trúc hệ thống OpenStack 4](#_Toc135573776)

[Hình 2 Các component trong OpenStack 5](#_Toc135573777)

[Hình 3 Ứng dụng đám mây ba tầng 14](#_Toc135573778)

[Hình 4 Biểu đồ phụ thuộc 14](#_Toc135573779)

[Hình 5 Kiến trúc mở 16](#_Toc135573780)

[Hình 6 Các thiết bị bảo vệ 16](#_Toc135573781)

[Hình 7 Karbor APIs 17](#_Toc135573782)

[Hình 8 Karbor Architecture 18](#_Toc135573783)

[Hình 9 Tạo người dùng và gán quyền sudo 20](#_Toc135573784)

[Hình 10 Chuyển sang người dùng 20](#_Toc135573785)

[Hình 11 Clone git openstack 21](#_Toc135573786)

[Hình 12 Cấu hình DevStack 22](#_Toc135573787)

[Hình 13 Cấu hình thành công 22](#_Toc135573788)

[Hình 14 Cài đặt OpenStack và DevStack 23](#_Toc135573789)

[Hình 15 Trang đăng nhập OpenStack 24](#_Toc135573790)

[Hình 16 Trang Overview 24](#_Toc135573791)

[Hình 17 Trang quản lý Instances 25](#_Toc135573792)

[Hình 18 Trang quản lý Imnages 25](#_Toc135573793)

[Hình 19 Trang quản lý Volumes 26](#_Toc135573794)

[Hình 20 Trang quản lý Volumes Snapshots 26](#_Toc135573795)

[Hình 21 Trang quản lý Networks 27](#_Toc135573796)

[Hình 22 Giao diện tạo máy ảo 27](#_Toc135573797)

[Hình 23 Giao diện chọn Images cho máy ảo 28](#_Toc135573798)

[Hình 24 Giao diện chọn Flavor 28](#_Toc135573799)

[Hình 25 Giao diện chọn Network cho máy ảo 29](#_Toc135573800)

[Hình 26 Giao diện thông tin máy ảo 30](#_Toc135573801)

[Hình 27 Giao diện chạy console 30](#_Toc135573802)

[Hình 28 Giao diện add Interfaces 31](#_Toc135573803)

[Hình 29 Network Topology 31](#_Toc135573804)

[Hình 30 Network Graph 32](#_Toc135573805)

[Hình 31 Kết quả ping giữa 2 máy 32](#_Toc135573806)

[Hình 32 Tạo Key Pairs 33](#_Toc135573807)

[Hình 33 Private Key 34](#_Toc135573808)

[Hình 34 Thông tin Key 34](#_Toc135573809)

[Hình 35 Tạo Snapsot 35](#_Toc135573810)

[Hình 36 Tạo Snapsot Images và Volume 36](#_Toc135573811)

[Hình 37 Giao diện quản lý User 36](#_Toc135573812)

[Hình 38 Giao diện quản lý Project 37](#_Toc135573813)

[Hình 39 Giao diện Roles 38](#_Toc135573814)

[Hình 40 Thiết lập Roles 38](#_Toc135573815)

[Hình 41 Giao diện người dùng 39](#_Toc135573816)

[Hình 42 Tạo Rule cho Sercurity Groups 40](#_Toc135573817)

# **CHƯƠNG I. TỔNG QUAN**

1. **Tổng quan đề tài**

Tên đề tài: Nghiên cứu giải pháp bảo vệ dữ liệu trên nền tảng OpenStack

1. **Lý do chọn đề tài**

Các lý do nhóm chọn đề tài “Nghiên cứu giải pháp bảo vệ dữ liệu ứng dụng trên nền tảng OpenStack”. Nghiên cứu các yêu cầu và tiêu chuẩn an toàn thông tin hiện tại .

Để nâng cao hiểu biết hơn về an toàn bảo mật hệ thống thông tin và bảo vệ dữ liệu. Để có cơ hội được nghiên cứu về các chính sách bảo mật an toàn hệ thống thông tin cũng như về bảo vệ dữ liệu.

Hiện nay việc mà các cuộc tấn công đánh cấp thông tin của các tin tặc dẫn đến việc thông tin bị rò rỉ, xuất hiện các vụ lừa đảo khách hàng từ các thông tin đã bị tin tặc đánh cấp theo nhiều hình thức và mục đích khác nhau thì nhóm chọn đề tài để có thể nghiên cứu và thực hiện xây dựng một chính sách để bảo mật hệ thống thông tin cho một doanh nghiệp để hiểu rõ hơn các doanh nghiệp làm thế nào để ngăn chặn sự xâm nhập, tấn công vào mạng lưới internet của doanh nghiệp bằng cách sử dụng các công cụ hoặc xây dựng các chính sách phù hợp. giúp các doanh nghiệp giảm thiểu thiệt hại về doanh thu, uy tín cũng như bảo vệ khách hàng của họ.

1. **Phạm vi của đề tài**

- Tìm hiểu cấu trúc hệ thống OpenStack

- Các dịch vụ của OpenStack

- Framework, API

- Dự án Karbor

- Triển khai công cụ đã nghiên cứu

# **CHƯƠNG II. CƠ SỞ LÝ THUYẾT**

1. **OpenStack là gì?**

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.

1. **OpenStack dùng để làm gì?**

Để tạo môi trường điện toán đám mây, các tổ chức thường xây dựng cơ sở hạ tầng ảo hóa của mình bằng cách sử dụng các hệ thống nổi tiếng như VMware vSphere, Microsoft Hyper-V hoặc KVM. Tuy nhiên, điện toán đám mây ngoài cung cấp ảo hóa public cloud và private cloud, cung cấp khả năng tự động hóa vòng đời tự phục vụ của người dùng, báo cáo chi phí và thanh toán, đồng bộ hóa và các khả năng khác.

Cài đặt phần mềm OpenStack trên môi trường ảo tạo ra một cloud operating system. Các công ty có thể sử dụng nó để tổ chức, đặt cấu hình và quản lý các nhóm tài nguyên mạng, storage và network resources khác nhau, trong khi quản trị viên CNTT thường định cấu hình và quản lý tài nguyên trong môi trường ảo truyền thống.

Cơ sở hạ tầng ảo hóa được xây dựng thông qua OpenStack có rất nhiều tác dụng, bao gồm tạo web hosting. Server chứa các dự án dữ liệu lớn, các phần mềm dưới dạng dịch vụ

OpenStack cạnh tranh trực tiếp với các nền tảng đám mây mã nguồn mở khác, ví dụ như Eucalyptus và Apache CloudStack, một số người cũng coi nó như một giải pháp thay thế cho các nền tảng đám mây miễn phí như Amazon Web Services hoặc Microsoft Azure, với một số công ty về hosting, cloud nhỏ hơn sử dụng OpenStack làm nền tảng gốc dịch vụ của mình

1. **Nguyên lý hoạt động**

OpenStack không phải là một ứng dụng theo nghĩa truyền thống, nhưng nó là một nền tảng được tạo thành từ hàng chục thành phần độc lập được gọi là *projects*, hoạt động cùng nhau thông qua các API. Các tổ chức chỉ có thể cài đặt các thành phần được chọn để tạo ra các tính năng và chức năng mong muốn trong môi trường đám mây.

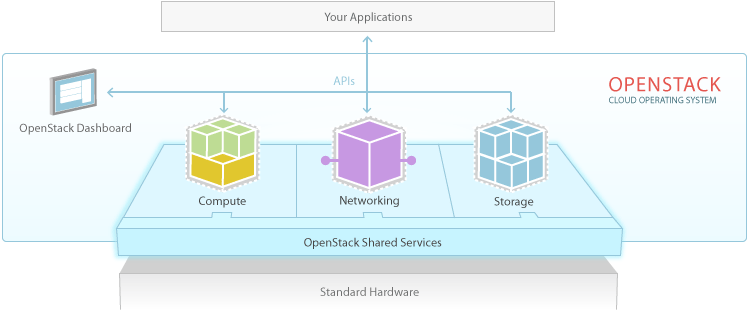
OpenStack cũng dựa vào hai công nghệ nền tảng: hệ điều hành cơ bản (như Linux) và nền tảng ảo hóa (như VMware hoặc Citrix), một hệ điều hành xử lý các hướng dẫn và dữ liệu được trao đổi từ OpenStack, trong khi công cụ ảo hóa quản lý các tài nguyên phần cứng ảo được sử dụng bởi projects OpenStack.

Sau khi hệ điều hành, nền tảng ảo hóa và các thành phần OpenStack được triển khai và cấu hình đúng cách, quản trị viên có thể định cấu hình và quản lý các tài nguyên được tạo sẵn mà ứng dụng cần. Các hành động và yêu cầu được thực hiện thông qua trang tổng quan tạo ra một loạt các lệnh gọi API được xác thực bởi dịch vụ bảo mật và được chuyển đến thành phần đích. thực hiện các nhiệm vụ liên quan.

**Ví dụ:** administrator đăng nhập vào OpenStack và quản lý môi trường đám mây thông qua dashboard. Administrator có thể tạo mới và kết nối các phiên bản điện toán và storage mới cũng như đặt cấu hình network behaviors. Administrator cũng có thể kết nối với các dịch vụ khác, chẳng hạn như giám sát hiệu năng phiên bản được cung cấp và sử dụng tính phí và bồi hoàn tài nguyên cho dung lượng lưu trữ.

Phạm vi rộng của nền tảng OpenStack và số lượng các thành phần được kết nối với nhau có thể gây nhầm lẫn và khó khăn. Hầu hết người dùng OpenStack bắt đầu với một số lượng nhỏ các yếu tố cơ bản và dần dần triển khai các phần tử khác. Theo thời gian, để xây dựng các khả năng hoạt động và kinh doanh của đám mây của họ.

1. **Cấu trúc hệ thống OpenStack**

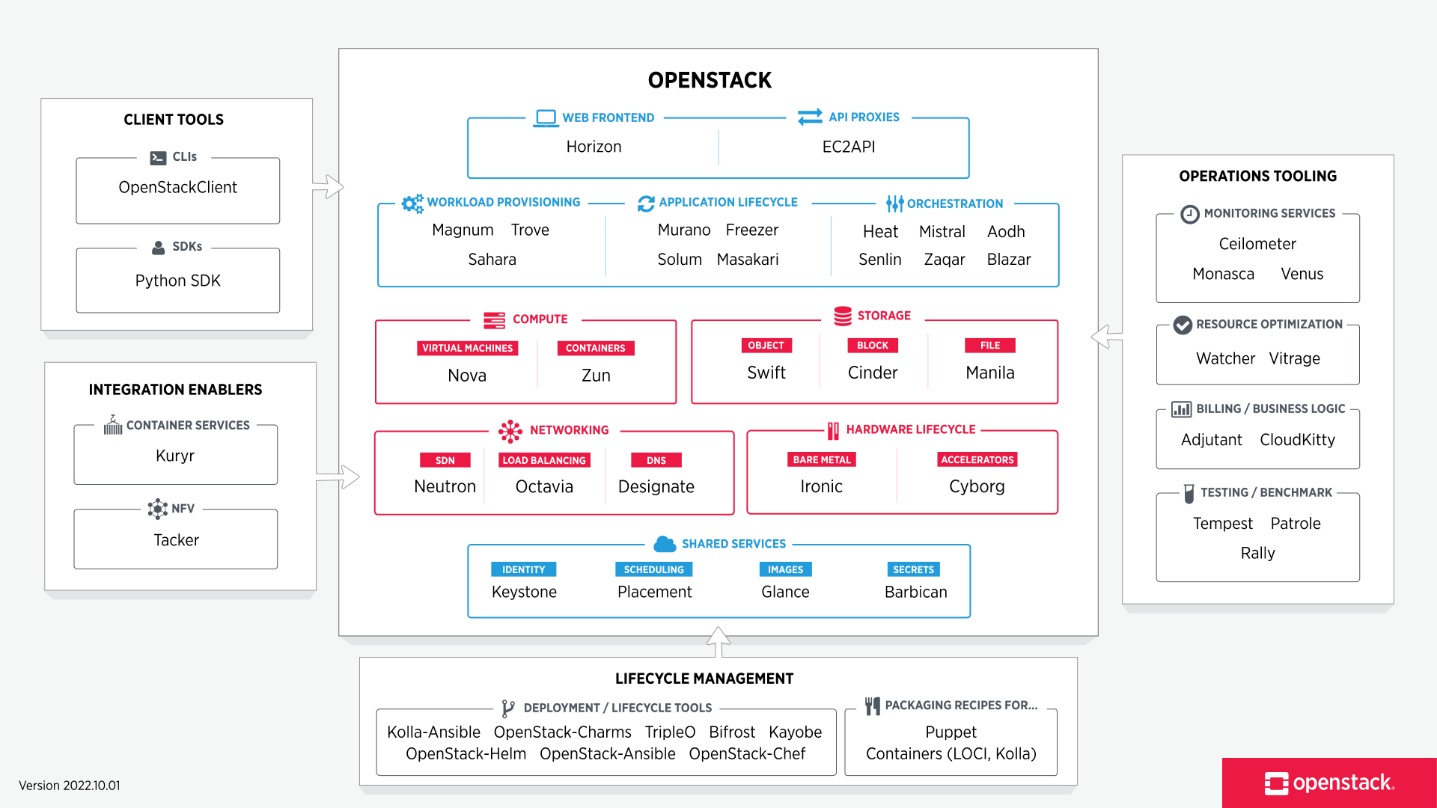


Hình 1 Cấu trúc hệ thống OpenStack

Kiến trúc tổng quan của OpenStack được chia thành 3 tầng

* Tầng ứng dụng (Your Application): Các ứng dụng/phần mềm sử dụng OpenStack
* Tầng Hypervisor (Standard Hardware): Phần ứng máy chủ đã được ảo hóa để chia sẻ cho người dùng.
* Dịch vụ OpenStack (Openstack Shared Services): Các thành phần cơ bản như Dashboard, Compute, Networking, API, Storage.

1. **Các Components trong OpenStack**



Hình 2 Các component trong OpenStack

OpenStack là một nền tảng đám mây mã nguồn mở được xây dựng trên kiến trúc máy chủ phân tán (distributed architecture) và bao gồm các thành phần chính như sau:

* Compute (Nova): Là thành phần quản lý tài nguyên tính toán, cho phép triển khai và quản lý máy ảo trên hạ tầng đám mây OpenStack.
* Networking (Neutron): Là thành phần quản lý mạng trong hạ tầng đám mây OpenStack, cung cấp các dịch vụ về mạng như tạo và quản lý mạng ảo, tạo và quản lý địa chỉ IP, cấu hình tường lửa, ...
* Storage (Cinder, Swift): Thành phần quản lý lưu trữ trên hạ tầng đám mây OpenStack. Cinder là thành phần quản lý lưu trữ khối (block storage), cho phép triển khai và quản lý các ổ đĩa ảo được gán vào máy ảo. Swift là thành phần quản lý lưu trữ đối tượng (object storage), cho phép lưu trữ và quản lý dữ liệu phi cấu trúc.
* Identity (Keystone): Là thành phần quản lý xác thực và phân quyền, cho phép quản lý người dùng, nhóm người dùng, quyền truy cập và đăng nhập vào hạ tầng đám mây OpenStack.
* Dashboard (Horizon): Là giao diện web cho người dùng cuối, cung cấp cho người dùng một cách thức để tương tác và quản lý các tài nguyên trên hạ tầng đám mây OpenStack.

Ngoài các thành phần chính này, OpenStack còn có các thành phần phụ như Telemetry (thống kê và giám sát), Orchestration (tự động hóa triển khai ứng dụng), Database (quản lý cơ sở dữ liệu) và Messaging (quản lý tin nhắn).

1. **Các thành phần chức năng chính của OpenStack**

Dựa trên các dịch vụ chính, Openstack đưa ra mô tả chi tiết các thành phần chức năng như sau:

*Controller node* là một node dùng để cài đặt hầu hết các dịch vụ liên quan đến quản trị, xác thực của Openstack cũng như các dịch vụ quản lý database cần thiết liên quan đến các image và các máy ảo cho hay là một control plane trong môi trường Openstack. *Controller node* chứa các module Keystone, Glance và Horizon.

*Compute node* là một node dùng đề cài đặt các dịch vụ quản lý các máy ảo. Compute node chứa module Nova.

*Network node* là một node dùng để cài đặt các dịch vụ quản lý đến hệ thống mạng và địa chỉ IP trong Openstack. Network node chứa module Neutron.

*Storage node* là một node dùng để cài đặt các dịch vụ liên quan đến quản lý lưu trữ các image, các máy ảo cũng như các file trong Openstack. *Storage node* chứa Cinder hoặc Swift hoặc cả Cinder lẫn Swift.

1. **Các module chính được cung cấp trong OpenStack**
2. **Openstack identity module**

OpenStack Identity (Keystone) cung cấp một danh mục các user được ánh xạ tới các dịch vụ Openstack để người dùng có thể truy nhập. OpenStack Identity hoạt động như một hệ thống xác thực chung trên toàn bộ hệ thống và có thể tích hợp với các dịch vụ có danh mục phụ trợ hiện có như Lightweight Directory Access Protocol (LDAP) hay Pluggable authentication module (PAM). OpenStack Identity hỗ trợ nhiều hình thức xác thực bao gồm thông tin đăng nhập tên người dùng và mật khẩu tiêu chuẩn, hệ thống token và phương thức truy nhập của AWS (tức là Amazon Web Services). Ngoài ra, OpenStack Identity còn cung cấp một danh sách có thể truy vấn của tất cả các dịch vụ được triển khai trên OpenStack trong một khởi tạo bình thường. Người dùng và các công cụ của bên thứ ba có thể xác lập tài nguyên được cấp quyền có thể truy cập thông qua OpenStack Identity.

1. **Openstack compute module**

OpenStack Compute (Nova) là một module điều khiển Cloud computing, là phần chính của hệ thống Openstack được phát triển theo mô hình dịch vụ Infrastructure as a Service (IaaS). OpenStack Compute được thiết kế để quản lý và tự động tối ưu các tài nguyên máy tính cũng như có thể hoạt động với sự mở rộng các công nghệ ảo hóa có sẵn bao gồm các cấu hình của bare-metal server hay cấu hình của siêu máy tính. KVM, VMware và Xen là những lựa chọn có sẵn sử dụng công nghệ hypanneror (màn hình máy ảo), cùng với công nghệ Hyper-V và Linux container là LXC .

OpenStack Compute phân tán các tác vụ hoạt động độc lập và riêng biệt như sau:

|  |  |  |
| --- | --- | --- |
| **STT** | **API** | **Chức năng** |
| 1 | nova-api | Tương tác giữa các API của hệ thống với người dùng |
| 2 | nova-compute | Cho phép người dùng có thể tạo và thực thi các VM instance với các API hypervisor (Libvirt KVM, Vmware API của Vmware) |
| 3 | nova-network | Cho phép người dùng quản lý các tác vụ liên quan đế mạng |
| 4 | nova - scheduler | Cho phép tối ưu các hoạt động của VM instance |
| 5 | nova - conductor | Cho phép người dùng truy nhập vào các node thông qua database |

1. **Openstack network module**

OpenStack Networking (Neutron) có chức năng quản lý mạng và địa chỉ IP. OpenStack Networking đảm bảo mạng không bị tắc nghẽn hoặc thắt cổ chai trong khi triển khai Cloud và cung cấp cho người dùng khả năng cấu hình nội bộ và cấu hình qua mạng Internet.

OpenStack Networking cung cấp các mô hình mạng cho các ứng dụng hoặc nhóm người dùng khác nhau. Các mô hình tiêu chuẩn bao gồm các flat network hoặc VLAN để phân tách các server với nhau và lưu lượng truyền dẫn.

OpenStack Networking quản lý địa chỉ IP, hỗ trợ cả địa chỉ IP tĩnh hoặc địa chỉ IP động. Địa chỉ Floating IP cho phép lưu lượng truy cập được định tuyến lại một cách linh hoạt bất kỳ tài nguyên nào trong cơ sở hạ tầng, do đó người dùng có thể chuyển hướng lưu lượng trong quá trình bảo trì hoặc trong trường hợp xảy ra lỗi.

Người dùng có thể tạo các mạng nội bộ, điều khiển lưu lượng, thiết lập kết nối tới các server và các thiết bị trong một hoặc nhiều mạng. Quản trị viên có thể sử dụng các công nghệ software-defined networking (SDN) như OpenFlow để hỗ trợ tối đa multi-tenancy và triển khai quy mô rộng. OpenStack Networking cung cấp một framework mở rộng có thể triển khai và quản lý các dịch vụ mạng thêm vào như hệ thống phát hiện xâm nhập (IDS), cân bằng tải, tường lửa và mạng riêng ảo (VPN).

1. **Openstack storage module**

Openstack storage có 2 loại lưu trữ là Block storage (Cinder) và Object storage (Swift)

* *OpenStack Block Storage (Cinder)*

OpenStack Block Storage (Cinder) là một hệ thống lưu trữ block-level để sử dụng với các OpenStack compute instance. Hệ thống block storage quản lý việc tạo, gắn và tách các khối thiết bị trên các server. Các phân vùng block storage được tích hợp hoàn toàn vào OpenStack Compute và Dashboard cho phép người dùng cloud quản lý lưu trữ cần thiết của người dùng. Ngoài lưu trữ trên server Linux cục bộ, block storage có thể sử dụng các nền tảng lưu trữ bao gồm Ceph, CloudByte, Coraid, EMC (ScaleIO, VMAX, VNX and XtremIO), GlusterFS, Hitachi Data Systems, IBM Storage (IBM DS8000, Storwize family, SAN Volume Controller, XIV Storage System, and GPFS), Linux LIO, NetApp, Nexenta, Nimble Storage, Scality, SolidFire, HP (StoreVirtual, 3PAR StoreServ families) và Pure Storage. Block storage cũng được sử dụng cho các trường hợp phức tạp liên quan tới hiệu suất như lưu trữ cơ sở dữ liệu, hệ thống file mở rộng hoặc cung cấp cho server quyền truy cập vào block-level storage. Sự quản lý snapshot cung cấp hiệu quả chức năng để sao lưu dữ liệu được lưu trữ trên phân vùng block storage. Snapshot có thể được khôi phục hoặc tạo mới một phân vùng block storage.

* *OpenStack Object Storage (Swift)*

OpenStack Object Storage (Swift) là một hệ thống lưu trữ dự phòng có thể mở rộng. Các object và file được ghi trên nhiều ổ đĩa trải đều các server trong data center với phần mềm OpenStack chịu trách nhiệm đảm bảo sao chép và toàn vẹn dữ liệu thông qua cluster. Các cluster lưu trữ phân bố đều khi thêm các server mới. Nếu server hoặc ổ cứng bị lỗi, OpenStack sẽ sao chép nội dung của nó từ các node hoạt động khác sang các vị trí mới trong cluster. Vì OpenStack sử dụng tính logic trong phần mềm để đảm bảo sao chép và phân tán dữ liệu trên các thiết bị khác nhau nên ổ cứng và server được sử dụng không cần đắt tiền.

Vào tháng 8 năm 2009, Rackspace đã bắt đầu phát triển OpenStack Object Storage như một sự thay thế hoàn toàn cho sản phẩm Cloud Files. Nhóm phát triển ban đầu bao gồm chín nhà phát triển. SwiftStack , một công ty phần mềm object storage, 28 hiện là nhà phát triển hàng đầu cho Swift với những đóng góp đáng kể từ HP, Red Hat, NTT, NEC, IBM….

1. **Openstack image module**

OpenStack Image (Glance) cung cấp dịch vụ trải nghiêm, tạo lập và cho phép sử dụng các image (ổ đĩa ảo). Các image lưu trữ được sử dụng như một template. OpenStack Image cũng có thể được sử dụng để lưu trữ và lập danh mục không giới hạn số lần sao lưu. Image Service có thể lưu trữ image trong nhiều loại back-end, bao gồm Swift. Image Service API cung cấp giao diện REST tiêu chuẩn để truy vấn thông tin về image ổ đĩa và cho phép các client tải các image sang server mới.

OpenStack Image thêm nhiều cải tiến cho cơ sở hạ tầng truyền thống. Nếu được tích hợp với VMware, OpenStack Image giới thiệu các tính năng nâng cao cho tập các vSphere như vMotion, tính sẵn sàng cao và lập lịch tài nguyên động (DRS). vMotion là một công nghệ cho phép di chuyển trực tiếp một VM đang chạy, từ server vật lý này sang server vật lý khác mà không bị gián đoạn dịch vụ. Do đó, OpenStack Image cho phép một datacenter tự tối ưu việc tự động và điều phối, cho phép bảo trì phần cứng cho các server hoạt động kém hiệu suất mà không bị gián đoạn.

Các module OpenStack khác cần tương tác với các image như Heat, phải giao tiếp với images metadata thông qua Glance. Ngoài ra, Nova có thể tiếp nhận thông tin về các image và sự thay đổi cấu hình trên image để tạo ra một instance. Tuy nhiên, Glance là module duy nhất có thể thêm, xóa, chia sẻ hoặc sao chép image.

1. **Openstack dashboard module**

OpenStack Dashboard (Horizon) cung cấp cho quản trị viên và người dùng giao diện đồ họa để truy cập, cung cấp và triển khai tự động các tài nguyên cloud-based. Mô hình chứa các sản phẩm và dịch vụ của bên thứ ba như thanh toán, giám sát và các công cụ quản lý bổ sung. OpenStack Dashboard cũng có khả năng tạo sự khác biệt trong cách sử dụng cho các nhà cung cấp dịch vụ và các nhà cung cấp thương mại khác. OpenStack Dashboard là một trong các cách người dùng có thể tương tác với tài nguyên OpenStack. Các nhà phát triển có thể tự động truy cập hoặc xây dựng các công cụ để quản lý tài nguyên bằng API OpenStack gốc hoặc API tương thích EC2.

1. **Các Framework, API**
2. **Framework**

OpenStack là một hệ thống phần mềm mở và linh hoạt, được xây dựng trên nhiều framework khác nhau để cung cấp các tính năng khác nhau. Dưới đây là một số framework phổ biến trong OpenStack:

* Django: Django là một framework phát triển ứng dụng web Python phổ biến, được sử dụng trong dịch vụ giao diện người dùng Horizon của OpenStack.
* SQLAlchemy: SQLAlchemy là một framework ORM (Object-Relational Mapping) Python, được sử dụng trong dịch vụ quản lý cơ sở dữ liệu Trove của OpenStack.
* Twisted: Twisted là một framework mạng Python, được sử dụng trong dịch vụ tính toán Nova của OpenStack.
* Celery: Celery là một framework phân tán cho các tác vụ bất đồng bộ, được sử dụng trong dịch vụ giám sát Ceilometer của OpenStack.
* Flask: Flask là một framework phát triển ứng dụng web Python nhẹ, được sử dụng trong dịch vụ giám sát Aodh của OpenStack.
* AMQP (Advanced Message Queuing Protocol): AMQP là một giao thức truyền tải tin nhắn phân tán, được sử dụng trong các dịch vụ như Messaging (Zaqar) và Notification (Swift) của OpenStack.
* libvirt: libvirt là một thư viện ảo hóa đa nền tảng, được sử dụng trong dịch vụ tính toán Nova của OpenStack.

Các framework này cùng với các công nghệ khác nhau được tích hợp trong OpenStack để tạo nên một hệ thống đa nền tảng, mở và linh hoạt để triển khai và quản lý các ứng dụng đám mây.

1. **API**

OpenStack cung cấp một số API khác nhau để quản lý và tương tác với các dịch vụ và tài nguyên trong hệ thống. Sau đây là một số API phổ biến của OpenStack:

* Identity API (Keystone): Được sử dụng để quản lý xác thực và phân quyền cho người dùng và các dịch vụ.
* Compute API (Nova): Được sử dụng để quản lý các máy ảo, khởi động, tắt, xóa và kiểm soát các tài nguyên tính toán.
* Networking API (Neutron): Được sử dụng để quản lý các mạng, máy chủ và dịch vụ mạng.
* Block Storage API (Cinder): Được sử dụng để quản lý các tài nguyên lưu trữ khối, như các ổ đĩa cứng và các máy chủ lưu trữ.
* Object Storage API (Swift): Được sử dụng để quản lý các tài nguyên lưu trữ đối tượng, như các tệp và thư mục.
* Image Service API (Glance): Được sử dụng để quản lý các hình ảnh máy ảo, bao gồm các hình ảnh của các hệ điều hành và phần mềm ứng dụng.
* Orchestration API (Heat): Được sử dụng để quản lý các tài nguyên được triển khai bằng các mẫu cấu hình và quản lý phân phối ứng dụng.
* Telemetry API (Ceilometer): Được sử dụng để thu thập và lưu trữ thông tin về các hoạt động và sự kiện trong hệ thống.

Các API này đều được thiết kế để có thể tích hợp với nhau để tạo ra một hệ thống đám mây đầy đủ tính năng. Bằng cách sử dụng các API này, người dùng có thể tùy chỉnh và quản lý các tài nguyên của họ trong OpenStack.

1. **Triển khai làm việc với Dashboard (Horizon) trên OpenStack**

Horizon là giao diện web cho OpenStack, cho phép người dùng tạo và quản lý các tài nguyên của OpenStack một cách trực quan và dễ dàng hơn. Để làm việc với Horizon, có thể thực hiện các bước sau:

* Đăng nhập vào Horizon: Truy cập vào địa chỉ IP của Horizon trên trình duyệt web, sau đó đăng nhập bằng tên người dùng và mật khẩu được cấp.
* Khám phá giao diện Horizon: Sau khi đăng nhập, bạn sẽ thấy giao diện của Horizon với các menu và chức năng. Bạn có thể khám phá giao diện này để tìm hiểu các tính năng và tài nguyên của OpenStack mà bạn muốn quản lý.
* Tạo và quản lý máy ảo: Bằng cách sử dụng Horizon, bạn có thể tạo và quản lý các máy ảo trên OpenStack. Bạn có thể tạo máy ảo mới, xem thông tin về máy ảo hiện có, điều chỉnh cấu hình máy ảo, v.v.
* Quản lý mạng: Horizon cho phép bạn quản lý các mạng và subnet trong OpenStack. Bạn có thể tạo, xóa, sửa đổi mạng và subnet, cũng như gán các máy ảo vào mạng tương ứng.
* Quản lý ổ đĩa: Horizon cho phép bạn quản lý các ổ đĩa trong OpenStack. Bạn có thể tạo và xóa các ổ đĩa, cũng như gán các ổ đĩa cho máy ảo tương ứng.
* Giám sát tài nguyên: Horizon cung cấp các tính năng giám sát tài nguyên để theo dõi và quản lý các tài nguyên của OpenStack. Bạn có thể xem thông tin về CPU, bộ nhớ, băng thông mạng, v.v.
* Quản lý người dùng và phân quyền: Horizon cũng cho phép bạn quản lý người dùng và phân quyền trong OpenStack. Bạn có thể tạo người dùng mới, thêm họ vào các project, và chỉ định quyền truy cập cho họ.

1. **Giới thiệu về dự án Karbor**
2. **Sứ mệnh của Karbor**

Để bảo vệ dữ liệu và dữ liệu meta bao gồm ứng dụng được triển khai OpenStack chống lại tổn thất/thiệt hại (ví dụ: sao lưu, sao chép) bằng cách cung cấp một khung API và dịch vụ tiêu chuẩn cho phép các nhà cung cấp cung cấp các plugin thông qua giao diện thống nhất.

1. **Khái niệm Karbor**

Karbor liên quan đến việc bảo vệ dữ liệu và dữ liệu meta bao gồm một ứng dụng được triển khai OpenStack (cái được gọi là "dự án" trong thuật ngữ Keystone) chống lại tổn thất/thiệt hại (ví dụ: sao lưu, sao chép)-không bị nhầm lẫn với bảo mật ứng dụng hoặc DLP . Nó thực hiện điều đó bằng cách cung cấp một khung tiêu chuẩn của API và dịch vụ cho phép các nhà cung cấp giới thiệu các dịch vụ bảo vệ dữ liệu khác nhau vào một luồng thống nhất và thống nhất cho người dùng. Koala là một loại thú có marsupial ăn cỏ ở Úc và Karbor là tên bản địa của Koala. Karbor có một túi tự nhiên để bảo vệ con cái của họ. Karbor của chúng tôi được thiết kế để cung cấp trải nghiệm đơn giản và thân thiện với người dùng để bảo vệ dữ liệu của người dùng. Khái niệm chính đằng sau Karbor là cung cấp sự bảo vệ của toàn bộ dự án OpenStack, trên các trang web OpenStack (hoặc với một trang web cục bộ duy nhất).

1. **Các thuật ngữ trong Karbor**

*Protectable* Một loại yếu tố mà Karbor có thể bảo vệ. Đổi mới các tài nguyên OpenStack nổi bật nhất (khối lượng, dự án, máy chủ, v.v.), ví dụ thực tế của bảo vệ được đặt tên là tài nguyên

*Protection Plugin*: một plugin, chịu trách nhiệm thực hiện bảo vệ và khôi phục một vật liệu bảo vệ có thể bảo vệ hoặc nhiều

*Bank Plugin*: một plugin, chịu trách nhiệm về dữ liệu tồn tại giữa các hoạt động bảo vệ và khôi phục và giữa các trang web khác nhau

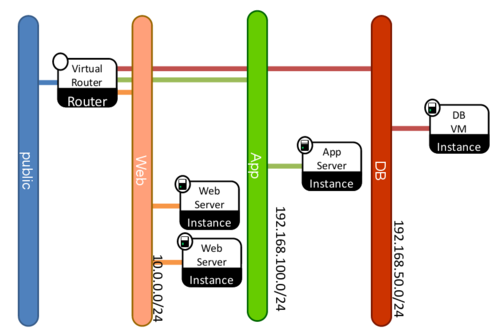
*Protection Provider*: Một thành phần của các plugin bảo vệ bao gồm các loại tài nguyên được hỗ trợ và một plugin ngân hàng để tồn tại dữ liệu

*Checkpoint*: trạng thái, dữ liệu và siêu dữ liệu của các tài nguyên được bảo vệ, trong một thời gian

*Protection Plan*: Nhà cung cấp bảo vệ được áp dụng cho một bộ tài nguyên, với các tham số người dùng

Ví dụ:

Hãy lấy một ứng dụng đám mây 3 tầng điển hình:



Hình 3 Ứng dụng đám mây ba tầng

Ứng dụng Cấp 3 mẫu Để bảo vệ hoàn toàn việc triển khai như vậy (ví dụ: Khôi phục thảm họa), chúng tôi sẽ phải bảo vệ nhiều tài nguyên, có một số phụ thuộc giữa chúng.

Biểu đồ sau đây cho thấy cây phụ thuộc như vậy có thể trông như thế nào:

Diagram, schematic

Description automatically generated

Hình 4 Biểu đồ phụ thuộc

Biểu đồ phụ thuộc của ứng dụng mẫu trong Karbor

Chúng tôi đã xác định một công cụ plugin tải plugin bảo vệ cho từng loại tài nguyên. Sau đó, chúng tôi cho phép người dùng tạo một kế hoạch bảo vệ, bao gồm tất cả các tài nguyên cô ấy muốn bảo vệ.

Các tài nguyên này có thể được chia thành các nhóm, mỗi nhóm được xử lý bởi một plugin khác nhau trong Karbor:

+ Volume - Thông thường, một khối dữ liệu được ánh xạ/gắn vào VM và được sử dụng để đọc/viết

+ VM - một đơn vị khối lượng công việc được triển khai, thường là bao gồm một số siêu dữ liệu (cấu hình, tùy chọn) và mạng ảo (phụ thuộc) được kết nối

+ Mạng ảo - Lớp phủ mạng ảo nơi VM chạy

+ Dự án - Một nhóm VMS và các tài nguyên được chia sẻ của họ (ví dụ: mạng, khối lượng, hình ảnh, v.v.)

+ Hình ảnh - Gói phân phối phần mềm được sử dụng để khởi chạy VM

1. **Điểm nổi bật của Karbor**
2. **Kiến trúc mở**

Các nhà cung cấp tạo ra các plugin thực hiện các cơ chế bảo vệ cho các tài nguyên OpenStack khác nhau.

- Quan điểm người dùng: Bảo vệ triển khai ứng dụng

Người dùng định cấu hình và quản lý các kế hoạch bảo vệ tùy chỉnh trên các tài nguyên được triển khai (cấu trúc liên kết, VM, tập, hình ảnh, trên mạng). Người dùng chọn một "nhà cung cấp bảo vệ" từ một lựa chọn các nhà cung cấp bảo vệ có sẵn, được quản trị viên duy trì và quản lý.

- Quan điểm của quản trị viên: Cấu hình các nhà cung cấp bảo vệ

Quản trị viên xác định nhà cung cấp bảo vệ nào có sẵn cho người dùng. Một "nhà cung cấp bảo vệ" về cơ bản là một gói các plugin bảo vệ mỗi tài nguyên và một ngân hàng, được quản lý từ tổng số các plugin bảo vệ có sẵn và các plugin ngân hàng. Ngoài ra, quản trị viên định cấu hình tài khoản ngân hàng cho mỗi người dùng (người thuê).

Graphical user interface, text, application, chat or text message

Description automatically generated

Hình 5 Kiến trúc mở

1. **Có sẵn các thiết bị bảo vệ**

Như bạn có thể nhận thấy từ thiết kế linh hoạt của nó. Karbor có khả năng thêm bất kỳ loại tài nguyên mới nào. Điều đó được nói rằng hầu hết người dùng có thể sẽ muốn chỉ sử dụng các thiết bị bảo vệ mặc định được cung cấp với Karbor.

Đối với Phiên bản 1.0, chúng tôi đang nhắm đến việc hỗ trợ các thiết bị bảo vệ này: Đây là sự phụ thuộc giữa các vật liệu bảo vệ trong phân phối mặc định của Karbor. Các mũi tên, giống như kế thừa, chỉ vào cha mẹ vì các thiết bị bảo vệ xác định loại nào có thể phụ thuộc vào chúng để điều này phản ánh ai chịu trách nhiệm kết nối.

Graphical user interface, diagram

Description automatically generated

Hình 6 Các thiết bị bảo vệ

1. **Karbor APIs**

Diagram

Description automatically generated with medium confidence

Hình 7 Karbor APIs

*API tài nguyên* (có thể bảo vệ) cho phép người dùng Karbor truy cập thông tin về các loại tài nguyên nào có thể bảo vệ được (nghĩa là có thể được bảo vệ bởi Karbor). Ngoài ra, cho phép người dùng nhận thêm thông tin về từng loại tài nguyên, chẳng hạn như danh sách các phiên bản thực tế và phụ thuộc của chúng.

*API Plan* cho phép người dùng Karbor tạo hoặc chỉnh sửa các gói bảo vệ bằng cách sử dụng nhà cung cấp bảo vệ được chọn, cũng như truy cập tất cả các tham số của gói.

*Nhà cung cấp API* cho phép người dùng Karbor liệt kê các nhà cung cấp có sẵn và nhận các tham số và kết quả Lược đồ siêu đặt hàng cho tất cả các plugin của một nhà cung cấp cụ thể.

*API điểm kiểm tra* cho phép người dùng Karbor truy cập và quản lý các điểm kiểm tra bảo vệ, cũng như liệt kê và truy vấn các điểm kiểm tra hiện có trong một nhà cung cấp. Ngoài ra, cung cấp quyền truy cập đọc điểm kiểm tra vào API khôi phục, khi khôi phục dữ liệu của ứng dụng được bảo vệ. Gọi API điểm kiểm tra (POST) sẽ bắt đầu một quy trình bảo vệ sẽ tạo một kho tiền trong tài khoản ngân hàng của người dùng, trên ngân hàng được gán cho nhà cung cấp. Quá trình này sau đó sẽ chuyển kho tiền trên một cuộc gọi đến hành động bảo vệ trên mỗi plugin bảo vệ được gán cho nhà cung cấp, vì vậy mỗi plugin sẽ viết siêu dữ liệu của mình vào kho tiền. Nó được để lại cho việc triển khai plugin để quyết định nơi lưu trữ dữ liệu thực tế (nghĩa là trong kho tiền hoặc ở một nơi khác).

*Lịch trình API* hoạt động cho phép người dùng Karbor tạo ánh xạ giữa các định nghĩa kích hoạt và hoạt động.

*Khôi phục API* cho phép gọi và theo dõi các hoạt động khôi phục. Khôi phục hoạt động sẽ thực hiện một trạm kiểm soát trong ngân hàng và cung cấp nó trong một đám mây.

1. **Karbor Architecture**

Graphical user interface, timeline

Description automatically generated

Hình 8 Karbor Architecture

Dịch vụ API của Karbor: các API liên kết ở phía bắc cấp cao nhất này để hiển thị các dịch vụ bảo vệ dữ liệu ứng dụng cho người dùng Karbor. Mục đích của các dịch vụ là tối đa hóa tính linh hoạt và phù hợp với (hy vọng) bất kỳ loại bảo vệ nào cho bất kỳ loại tài nguyên nào, cho dù đó là tài nguyên OpenStack cơ bản (như VM, Volume, Image, v.v.) hoặc một số tài nguyên phụ trợ bên trong Một hệ thống ứng dụng không được quản lý trong OpenStack (chẳng hạn như thiết bị phần cứng, cơ sở dữ liệu bên ngoài, v.v.).

Dịch vụ lịch trình của Karbor: hệ thống con này chịu trách nhiệm lên lịch và sắp xếp việc thực hiện các kế hoạch bảo vệ. Việc thực hiện có thể được thay thế bằng bất kỳ giải pháp bên ngoài nào khác. Tất cả các hoạt động liên quan đến bảo vệ thực tế được quản lý thông qua các API về phía bắc, để hỗ trợ:

- Ghi lại việc duy trì tất cả các hoạt động trong cơ sở dữ liệu Karbor (để điều khiển API trạng thái hoạt động)

- Tách rời việc triển khai bộ lập lịch từ việc triển khai Dịch vụ Bảo vệ

Dịch vụ bảo vệ Karbor: Hệ thống con này chịu trách nhiệm xử lý các nhiệm vụ sau:

- Hoạt động thực thi

- Bảo vệ (Tài nguyên)

- Bảo vệ Plugin Bảo vệ Nhà cung cấp

- Bảo vệ Plugin Quản lý Plugin

- Bank Quản lý Trình kiểm tra Ngân hàng

- Ngân hàng Dịch vụ phụ

# **CHƯƠNG III. CÀI ĐẶT THỬ NGHIỆM**

## **3.1 Cài đặt và cấu hình OpenStack**

**Bước 1: Cập nhật và nâng cấp hệ thống**

Để bắt đầu, hãy đăng nhập vào hệ thống Ubuntu bằng giao thức SSH, cập nhật và nâng cấp kho hệ thống bằng lệnh sau:

*apt update -y && apt upgrade -y*

**Bước 2: Tạo người dùng Stack và gán đặc quyền sudo**

Phương pháp hay nhất yêu cầu rằng devstack phải được chạy bằng một tài khoản người dùng thông thường với các đặc quyền sudo.

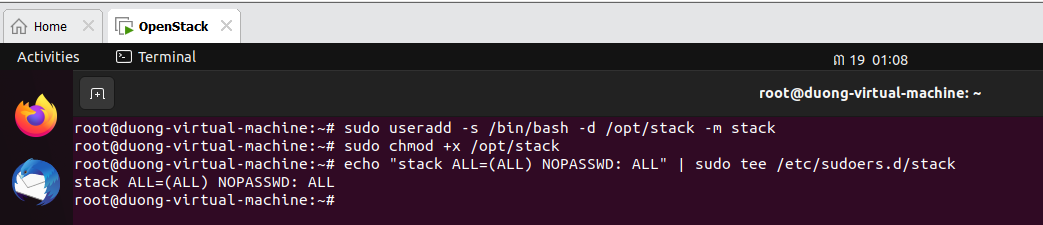
Do đó, bài viết sẽ thêm một người dùng mới được gọi là **"stack"** và chỉ định các đặc quyền sudo.

Để tạo người dùng stack, hãy thực thi lệnh:

*sudo adduser -s /bin/bash -d /opt/stack -m stack*

Tiếp theo, chạy lệnh dưới đây để gán các đặc quyền sudo cho người dùng stack:

*echo "stack ALL=(ALL) NOPASSWD: ALL" | sudo tee /etc/sudoers.d/stack*

**

Hình 9 Tạo người dùng và gán quyền sudo

**Bước 3: Cài đặt git và tải DevStack**

Khi bạn đã tạo thành công người dùng stack và chỉ định các đặc quyền sudo, hãy chuyển sang người dùng này bằng cách sử dụng lệnh.

*sudo -u stack -i*

*Text

Description automatically generated*

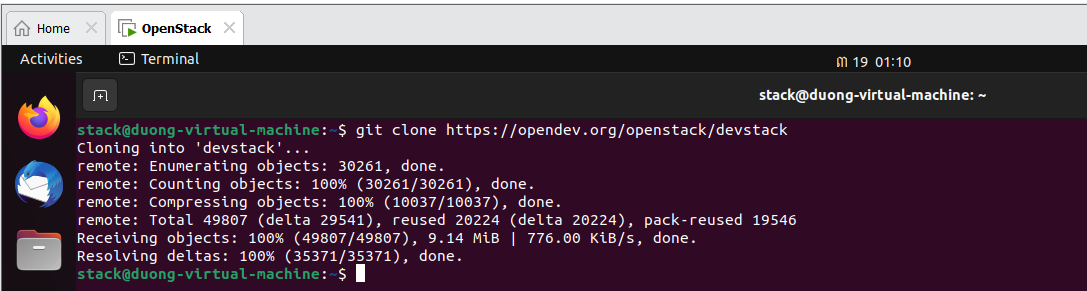
Hình 10 Chuyển sang người dùng

Trong hầu hết các hệ thống Ubuntu, git đã được cài đặt sẵn. Nếu ngẫu nhiên git bị thiếu, hãy cài đặt nó bằng cách chạy lệnh sau:

*sudo apt install git -y*

Sử dụng git, hãy clone (sao chép) kho lưu trữ git của Devstack như hình.

*git clone* [*https://git.openstack.org/openstack-dev/devstack*](https://git.openstack.org/openstack-dev/devstack)

**

Hình 11 Clone git openstack

**Bước 4: Tạo file cấu hình Devstack**

Trong bước này, điều hướng đến thư mục devstack:

*cd devstack*

Sau đó, cấu hình file **local.conf** trong thư mục samples:

*vim samples/local.conf*

Dán nội dung sau:

*[[local|localrc]]*

*FLOATING\_RANGE=192.168.100.0/24*

*FIXED\_RANGE=10.11.12.0/24*

*FIXED\_NETWORK\_SIZE=256*

*FLAT\_INTERFACE=ens33*

*# Password for KeyStone, Database, RabbitMQ and Service*

*ADMIN\_PASSWORD=admin*

*DATABASE\_PASSWORD=$ADMIN\_PASSWORD*

*RABBIT\_PASSWORD=$ADMIN\_PASSWORD*

*SERVICE\_PASSWORD=$ADMIN\_PASSWORD*

*Text

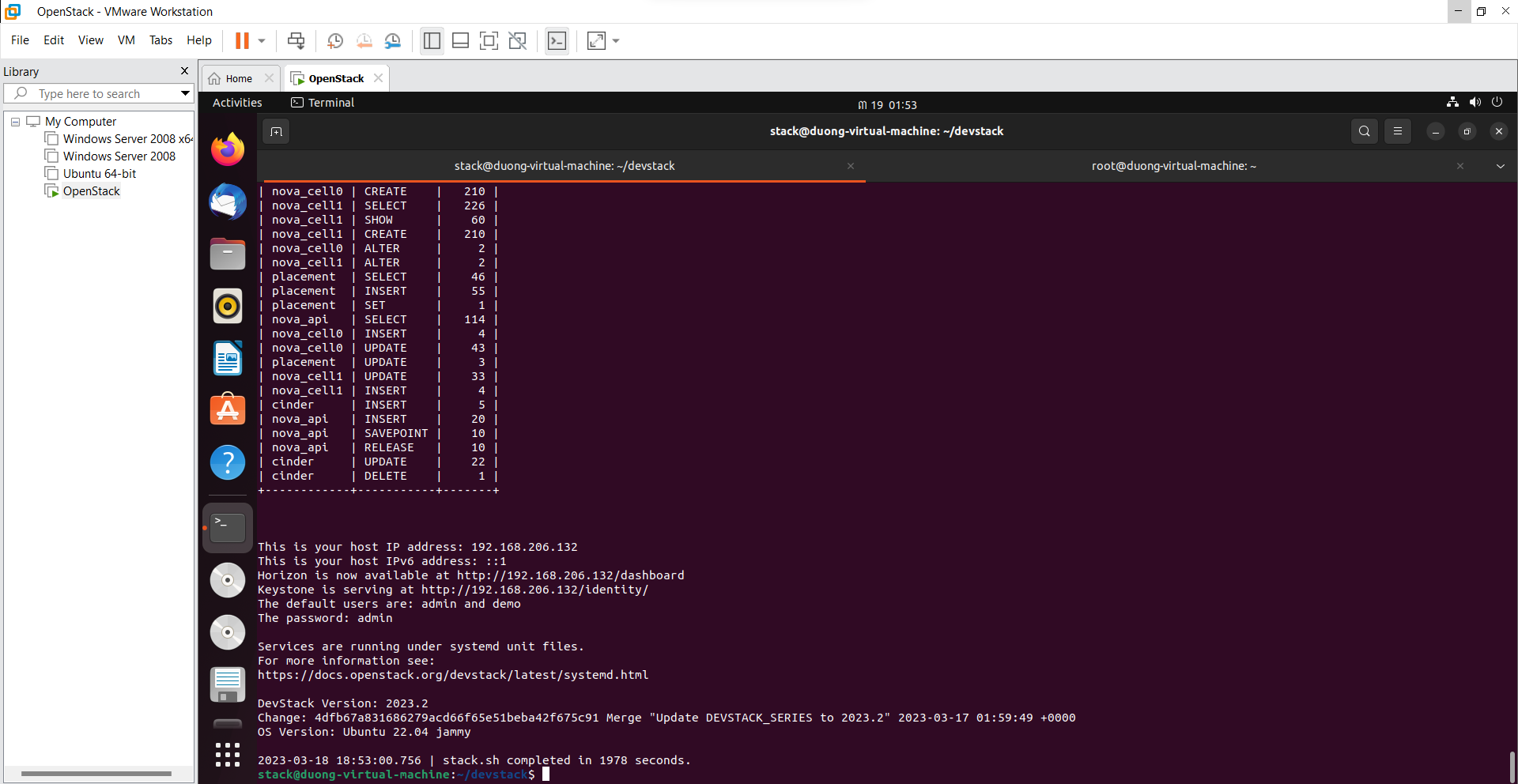
Description automatically generated*

Hình 12 Cấu hình DevStack

Lưu ý:

1. **ADMIN\_PASSWORD** là mật khẩu mà bạn sẽ sử dụng để đăng nhập vào trang đăng nhập OpenStack. Tên người dùng mặc định là **admin**.

2. **HOST\_IP** là địa chỉ IP hệ thống có được bằng cách chạy các lệnh **ifconfig** hoặc **ip addr.**

****

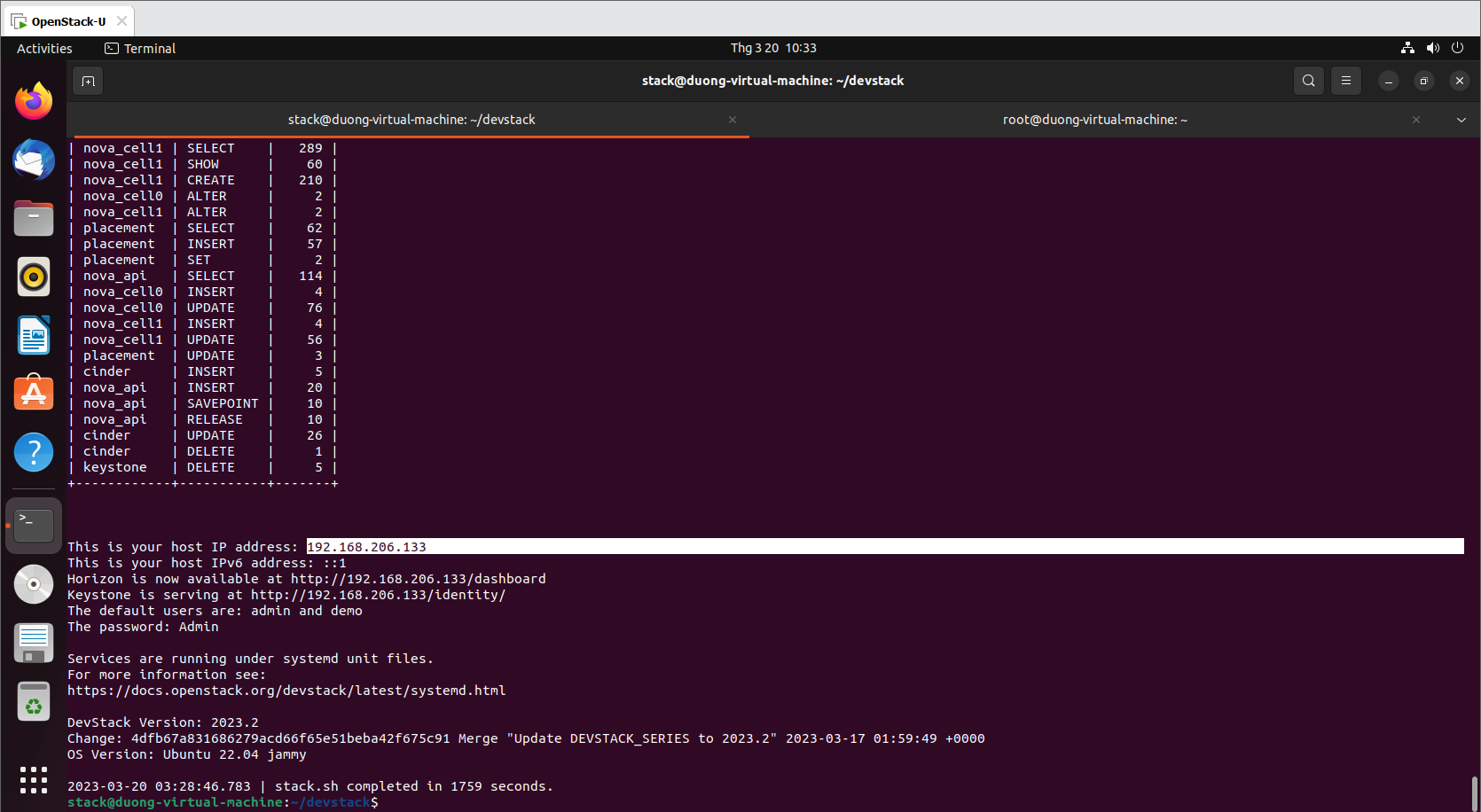
Hình 13 Cấu hình thành công

**Bước 5: Cài đặt OpenStack với Devstack**

Để bắt đầu cài đặt OpenStack trên Ubuntu, hãy chạy script bên dưới có trong thư mục devstack.

*./stack.sh*

Màn hình sau khi cài đặt thành công

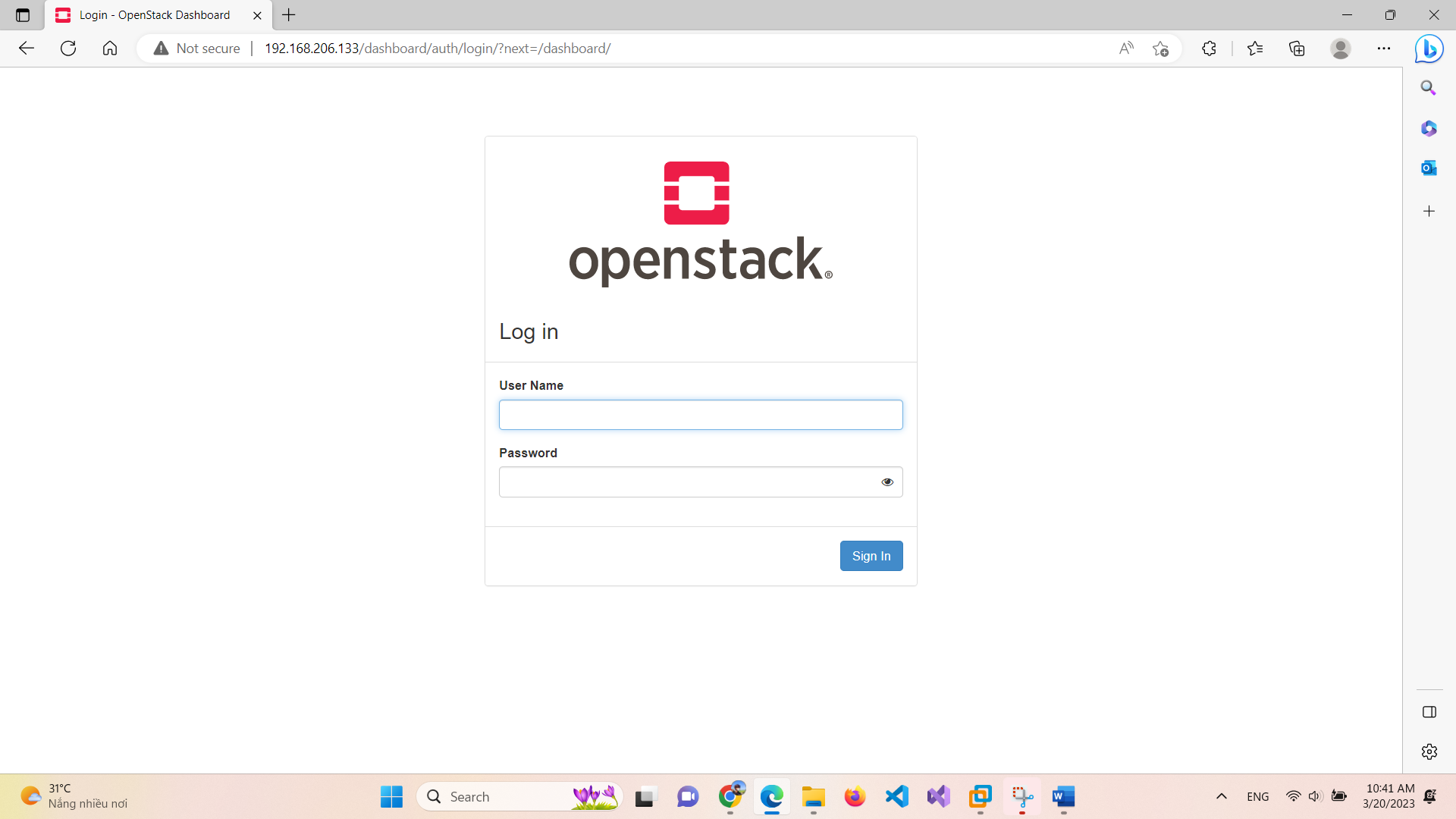
**

Hình 14 Cài đặt OpenStack và DevStack

Các tính năng sau sẽ được cài đặt:

* Horizon – OpenStack Dashboard
* Nova – Compute Service
* Glance – Image Service
* Neutron – Network Service
* Keystone – Identity Service
* Cinder – Block Storage Service
* Placement – Placement API

**Bước 6: Truy cập OpenStack trên trình duyệt web (Dashboard)**



Hình 15 Trang đăng nhập OpenStack

## **3.2 Giao diện Openstack Dashboard**

Dashboard Overview

Graphical user interface, chart, bubble chart

Description automatically generated

Hình 16 Trang Overview

Quản lý Instances

Graphical user interface, text, application

Description automatically generated

Hình 17 Trang quản lý Instances

Quản lý Images

Graphical user interface, application, email

Description automatically generated

Hình 18 Trang quản lý Imnages

Quản lý Volumes

Graphical user interface, text, application, email

Description automatically generated

Hình 19 Trang quản lý Volumes

Quản lý Volume Snapshots

Graphical user interface, text, application, email

Description automatically generated

Hình 20 Trang quản lý Volumes Snapshots

Quản lý Networks

Graphical user interface, application

Description automatically generated

Hình 21 Trang quản lý Networks

Tạo 2 máy ảo có tên là may-1 và may-2 chạy trên mạng net-1 và net-2 vừa tạo

Graphical user interface, application

Description automatically generated

Hình 22 Giao diện tạo máy ảo

Chọn Image cho máy ảo

Graphical user interface, text, application, email

Description automatically generated

Hình 23 Giao diện chọn Images cho máy ảo

Chọn Flavor

Graphical user interface, application

Description automatically generated

Hình 24 Giao diện chọn Flavor

Chọn network cho máy ảo

Graphical user interface

Description automatically generated

Hình 25 Giao diện chọn Network cho máy ảo

Xem thông tin máy ảo

Graphical user interface, text, application, email

Description automatically generated

Hình 26 Giao diện thông tin máy ảo

Chạy Console

Graphical user interface, text

Description automatically generated

Hình 27 Giao diện chạy console

Add Inteface may-1 và may-2 cho router:

Table

Description automatically generated

Hình 28 Giao diện add Interfaces

Kết quả kiểm tra sơ đồ mạng: đã kết nối được mạng net-1 và net-2 cho hai máy với mạng Public qua router

A picture containing chart

Description automatically generated

Hình 29 Network Topology

Text

Description automatically generated with medium confidence

Hình 30 Network Graph

Ping từ may-1 qua may-2 để kiểm tra kết nối mạng

A screenshot of a computer

Description automatically generated with medium confidence

Hình 31 Kết quả ping giữa 2 máy

## **3.3 Bảo vệ dữ liệu trên Openstack**

OpenStack cung cấp nhiều giải pháp bảo vệ dữ liệu như mã hóa dữ liệu, sao lưu dữ liệu và kiểm soát truy cập dữ liệu

* Mã hóa dữ liệu là một giải pháp bảo vệ dữ liệu trên OpenStack. Mã hóa dữ liệu là quá trình chuyển đổi thông tin sang một định dạng khác để ngăn chặn người không có quyền truy cập thông tin.

OpenStack cung cấp một cặp khóa công khai / riêng tư để mã hóa dữ liệu trên các máy chủ ảo của bạn. Công khai OpenStack ssh key có thể được chèn vào một instance khi khởi chạy, để sẵn sàng cho bạn truy cập bằng cách sử dụng khóa riêng tư. Nếu sau đó bạn thiết lập ssh để từ chối xác thực mật khẩu và thay vào đó yêu cầu khóa, bạn sẽ cung cấp cho instance của mình một lớp bảo mật mạnh hơn rất nhiều.

Tạo Key Pairs: có thể chọn kiểu mã hóa SSH key hoặc kiểu khác

Graphical user interface, application

Description automatically generated

Hình 32 Tạo Key Pairs

Sau khi tạo key thì hệ thống sẽ tự động download cho máy chủ một file private key và đồng thời lưu trên doashboard một public key.

Private key được lưu về máy

A screen shot of a computer

Description automatically generated with medium confidence

Hình 33 Private Key

Dưới đây là thông tin của public key vừa tạo và chúng ta có thể bảo mật khi khởi tạo máy ảo mới bằng cách gán key vừa tạo cho máy ảo

Graphical user interface, text, application, email

Description automatically generated

Hình 34 Thông tin Key

Sao lưu dữ liệu là một giải pháp bảo vệ dữ liệu trên OpenStack. Sao lưu dữ liệu là quá trình sao chép dữ liệu sang một nơi khác để đảm bảo rằng dữ liệu của bạn được lưu trữ an toàn và có thể khôi phục được khi cần thiết.

Snapshot là một công nghệ sao lưu dữ liệu được sử dụng để tạo ra một bản sao lưu của hệ thống hoặc một phần của hệ thống tại một thời điểm cụ thể. Nó có thể được sử dụng để bảo vệ dữ liệu khỏi các sự cố như lỗi phần cứng hoặc phần mềm, tấn công độc hại và lỗi người dùng.

Tạo Snapshot sao lưu dữ liệu của may-1

Graphical user interface, text, application, email

Description automatically generated

Hình 35 Tạo Snapsot

Tạo được một Snapshot Image (đồng thời Volume của Image cũng tự động tạo Snapshot mới)

Graphical user interface, text, application

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Hình 36 Tạo Snapsot Images và Volume

Từ Snapshot đó có thể Lauch một instance mới chứa thông tin Image snapshot của máy cũ

Kiểm soát truy cập dữ liệu là một giải pháp bảo vệ dữ liệu trên OpenStack. Kiểm soát truy cập dữ liệu là quá trình kiểm soát quyền truy cập vào dữ liệu của bạn để đảm bảo rằng chỉ những người được ủy quyền mới có thể truy cập vào dữ liệu của bạn

OpenStack cung cấp một số công cụ để kiểm soát truy cập dữ liệu như Keystone. Keystone là một dịch vụ xác thực và ủy quyền trong OpenStack, nó quản lý danh sách người dùng và vai trò của họ và xác định quyền truy cập của từng người dùng.

Quản lý các user

Graphical user interface, text, application, email

Description automatically generated

Hình 37 Giao diện quản lý User

Quản lý các Projects: Mỗi Project có thể gán cho nhiều user và người lại

Graphical user interface, text, application, email

Description automatically generated

Hình 38 Giao diện quản lý Project

Phân quyền trên Openstack:

- Admin: quyền cao nhất, có thể truy cập để tạo user và quản lý được toàn bộ dashboard, xem được những máy ảo nào đã tạo (chứa thông tin máy ảo đó thuộc Project nào và User nào đã tạo ra nó …)

- Member: được hiểu là quyền chỉ cho phép thao tác giới hạn trong phạm vi Project của riêng user đó

- Reader: chỉ cho phép xem

- Service: quản lý các dịch vụ

- Và một số quyền khác

Danh sách các quyền cơ bản của Openstack và chúng ta có thể tạo thêm quyền mới bằng tài khoản admin

Graphical user interface, text, email

Description automatically generated with medium confidence

Hình 39 Giao diện Roles

Ở danh mục Users tạo tải khoản người dùng mới. Sau đó gán Project cho người dùng và thiết lập Role cho người dùng.

Graphical user interface, application

Description automatically generated

Hình 40 Thiết lập Roles

Đăng nhập vào tài khoản có quyền member: các máy ảo và thông tin, dữ liệu của Project khác sẽ không được hiển thị. Do quyền member chỉ thực thi các tác vụ bên trong phạm vi Project của user đó thôi.

Graphical user interface

Description automatically generated

Hình 41 Giao diện người dùng

Ngoài ra, OpenStack còn có một số công cụ khác để kiểm soát truy cập dữ liệu như Security Groups bạn có thể sử dụng Security Groups để kiểm soát truy cập dữ liệu trong OpenStack. Security Groups là một tính năng của OpenStack, cho phép bạn tạo các quy tắc để kiểm soát truy cập vào các máy chủ ảo của bạn. Bạn có thể tạo các quy tắc để cho phép hoặc từ chối truy cập từ một địa chỉ IP hoặc một nhóm địa chỉ IP.

Tạo Rule cho Sercurity Groups và Add thêm Rule mới để kiểm soát truy cập

Graphical user interface, application

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Hình 42 Tạo Rule cho Sercurity Groups

# **CHƯƠNG IV. KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN**

1. **Kết luận**
2. **Ưu điểm**

- Tiết kiệm chi phí: OpenStack mà phần mềm mã nguồn mở được phát hành miễn phí theo giấy phép Apache 2.0.

- Độ tin cậy cao: OpenStack gồm có nhiều mô đun cho phép các doanh nghiệp xây dựng và vận hành đám mây riêng hoặc công cộng với nhiều khả năng như mở rộng lưu trữ, nâng cao hiệu suất, bảo mật dữ liệu và quy mô sử dụng lớn.

- Nhà cung cấp trung lập: không có bất kỳ hạn chế nào bởi OpenStack là một phần mềm mã nguồn mở.

1. **Nhược điểm**

- Việc triển khai OpenStack đòi hỏi nhiều kỹ năng gây tốn thời gian và chi phí.

- Gây khó khăn trong việc hỗ trợ quản lý chất lượng các dự án ngoài cộng đồng mã nguồn mở.

- Ngừng hỗ trợ các phiên bản thành phần cũ khi có phiên bản mới thay thế.

1. **Cách khắc phục**

- Khắc phục mức độ phức tạp khi triển khai OpenStack các tổ chức đã chuyển sang sử dụng bản phân phối được tích hợp với một số kỹ thuật hỗ trợ.

- Một số bản phân phối OpenStack phổ biến bao gồm các nền tảng như Red Hat OpenStack, đám mây Mirantis và đám mây riêng Rackspace OpenStack.

1. **Hướng phát triển**

**-** Kết hợp với các công nghệ khác vd như kubernetes để quản lý các ứng dụng và dịch vụ trên đám mây

- Triển khai các dịch vụ đám mây cho cộng đồng, doanh nghiệp, cá nhân

- Triển khai các dịch vụ đám mây để cho thuê

# **TÀI LIỆU THAM KHẢO**

1. Barkat, A., dos Santos, A. D., & Ho, T. T. N. (2014, September). Open stack and cloud stack: Open source solutions for building public and private clouds. In 2014 16th International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (pp. 429-436). IEEE.
2. Böhme, U., & Buytenhenk, L. (2000). Linux BRIDGE− STP− HOWTO. Revision 0.4. The Linux Documentation Project.
3. Chirammal, H. D., Mukhedkar, P., & Vettathu, A. (2016). Mastering KVM virtualization. Packt Publishing Ltd.
4. Cloud computing, Virtualization, Hypervisor, Openstack – Wikipedia (wikipedia.org).
5. Cloudstack – Cloudstack (cloudstack.apache.org).
6. Desai, A., Oza, R., Sharma, P., & Patel, B. (2013). Hypervisor: A survey on concepts and taxonomy. International Journal of Innovative Technology and Exploring Engineering, 2(3), 222-225.
7. Khedher, O., & Chowdhury, C. D. (2017). Mastering OpenStack. Packt Publishing Ltd.
8. Kumar, R., Jain, K., Maharwal, H., Jain, N., & Dadhich, A. (2014). Apache cloudstack: Open source infrastructure as a service cloud computing platform. Proceedings of the International Journal of advancement in Engineering technology, Management and Applied Science, 111-116.
9. Makita, T. (2014). Virtual switching technologies and linux bridge.
10. Mell, P., & Grance, T. (2011). The NIST definition of cloud computing.
11. Openstack – Openstack (openstack.org).
12. What is a private cloud? – Microsoft Azure (azure.microsoft.com).