Hội Tin học Việt Nam

Olympic Tin học 2016
Trường ĐH Nha Trang, 29/11/2016 - 02/12/2016

Nội dung : Phần mềm nguồn mở
Ngày : 01/12/2016

Thời gian 480'

A. Đề thi

Dựa trên nền tảng mã nguồn mở **OpenStack**, hãy triển khai dự án **OpenStack với các project core (nova, neutron, glance, keystone, horizon)** trên mô hình devstack theo mô hình như sau:

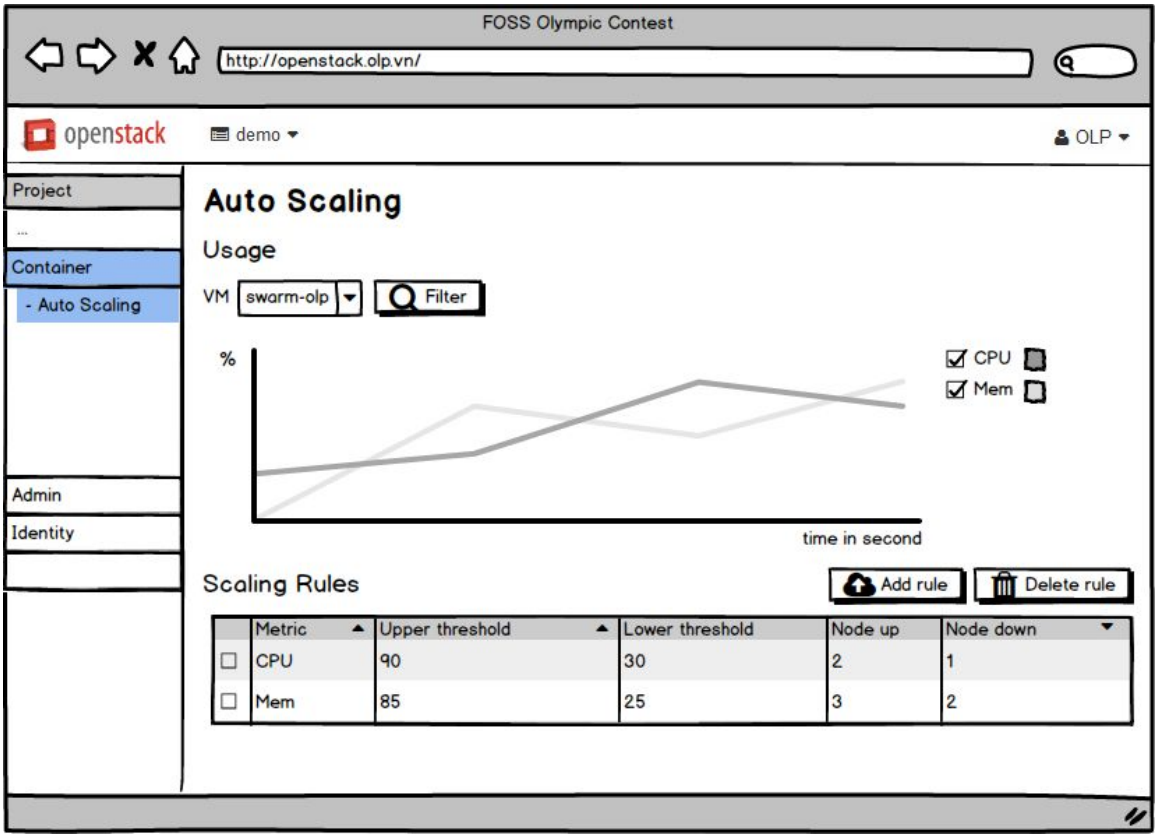
- DevStack All-in-One, branch **stable/mitaka**.
- Triển khai ít nhất **01 VM** với cấu hình tối thiểu **01 GB RAM** và **02 vCPU**, cài đặt **Docker v1.12 với Swarm mode** trên VM đó.
- Cài đặt và khởi chạy **cAdvisor** trên VM(s) đã cài đặt Docker Swarm mode.
- Gán tag **olp2016** cho VM trên.

Có thể sử dụng các image có sẵn Docker engine như CoreOS hoặc RancherOS (lưu ý cần enable **Configuration Drive** khi tạo VM cho cả 2 distro trên, với RancherOS cần flavor tối thiểu 8GB root disk).

Hiện tại, nền tảng mã nguồn mở OpenStack chưa có tính năng cho phép co giãn tài nguyên tự động (auto-scaling) đối với container trên VM, cụ thể khi một host chạy container có CPU và Memory vượt quá ngưỡng cho phép (ví dụ 90% CPU), OpenStack sẽ tự động cấp phát thêm node để giảm tải.

Đây là bài toán mô phỏng quá trình auto-scaling trong thực tế. Thực tiễn triển khai, auto-scaling yêu cầu mô hình phân tán và có các ràng buộc về hạ tầng mạng và lưu trữ phức tạp hơn.

Lập trình module quản lý cấu hình tự động co giãn tài nguyên (Auto Scaling) tương tự theo hình dưới đây:

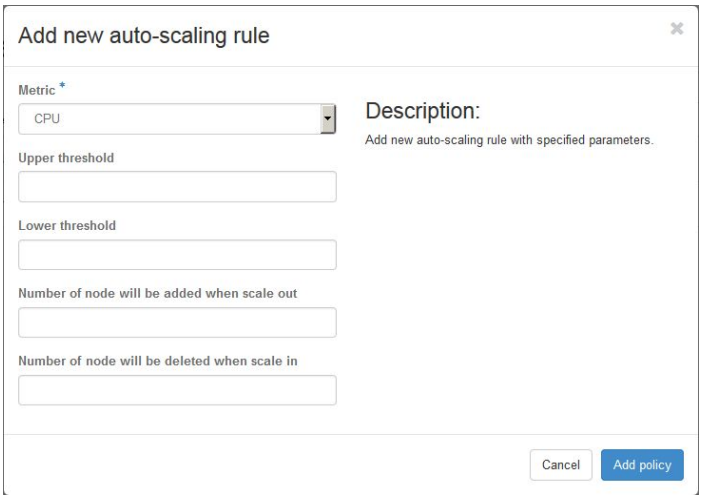


Trong đó:

- Panel **Container** và tab **Auto Scaling**: là panel và tab mới được cần được lập trình thêm vào Horizon. Table **Scaling Rules** trong tab **Auto Scaling** bao gồm các phần sau:
  - Quản lý tài nguyên của cụm container cluster đang cấp phát (Usage), gồm các thành phần nhỏ:
    - ❏ **VM**: combo-box liệt kê danh sách các VM **đã cài đặt Docker Swarm mode được gán tag olp2016** cần giám sát và cấu hình luật auto-scaling.
    - ❏ Biểu đồ thể hiện tài nguyên đang sử dụng theo thời gian **15 phút trước tính từ thời điểm hiện tại.**
    - ❏ Nút **Filter** để thực hiện thao tác cập nhật dữ liệu sử dụng tài nguyên mới nhất.
  - Bảng quản lý các **luật** (rule) cấu hình khả năng co giãn tự động cho **VM đã cài đặt Docker Swarm mode** lựa chọn từ combo-box trên, chi tiết các cột:
    - ❏ Tham số quyết định thực hiện việc co/giãn (metric): giám sát theo CPU hoặc Memory.
    - ❏ Ngưỡng mức trên (Upper Threshold): là ngưỡng thực hiện việc giãn thêm tài nguyên theo đơn vị %.

- ❏ Ngưỡng mức dưới (Lower Threshold): là ngưỡng thực hiện việc co lại tài nguyên theo đơn vị %.
- ❏ Node Up: số lượng swarm node (VM) sẽ thêm vào khi giãn tài nguyên.
- ❏ Node Down: số lượng node minion (VM) sẽ loại bỏ khi co tài nguyên.

- Nút **Delete Rule**: cho phép thực hiện việc xóa một hoặc nhiều luật đã chọn.
- Nút **Add Rule**: hiển thị giao diện dialog cho phép nhập thông tin của luật auto-scaling như sau:



Câu 1 (35 điểm): Lập trình và tạo giao diện dự án Horizon trong OpenStack tương tự như hình mô tả ở trên. Lưu ý:

- Biểu đồ thống kê lượng tài nguyên sử dụng hiển thị cho 2 mức tài nguyên là CPU và Memory theo đơn vị % và theo khung thời gian tính theo đơn vị giây.
- Toàn bộ các thao tác với nút bấm (Filter, Add Rule, Delete Rule) phải tuân thủ theo coding convention của Horizon (Django) về việc reload (postback) tuỳ ngữ cảnh và các thao tác với nghiệp vụ bên dưới.
- Mã nguồn chỉ cần chứng minh việc đã lập trình tạo các thành phần giao diện như yêu cầu trên, chưa cần thiết phải có thông tin dữ liệu.
- Mô tả cách thức đóng OpenStack Image có sẵn Docker Swarm mode, cAdvisor, cách thức khởi chạy Docker Engine cùng cAdvisor mỗi khi start VM và tự động join vào Swarm Cluster mỗi khi khởi tạo.
- **Điểm thưởng:** Nhóm sinh viên có điểm thưởng nếu hoàn thành các yêu cầu sau:
  - Có thể filter chỉ hiển thị biểu đồ theo CPU hoặc Memory, sử dụng ô check-box bên cạnh biểu đồ.

Câu 2 (25 điểm): Lập trình logic cho giao diện đã xây dựng từ câu 01.

- Connect vào Docker Remote API v1.24 (tương ứng Docker 1.12) để lấy ra các service hiển thị trên UI.
- Connect vào port của cAdvisor để query các thông số monitor của service và hiển thị lên UI.
- Combo-box chỉ hiển thị các VM có tags: **olp2016**

Lưu ý:

- Trình bày hướng tiếp cận, cách giải quyết, sơ đồ khối, sơ đồ luồng và cách thức tích hợp vào hệ thống OpenStack vào tài liệu riêng, nộp cùng mã nguồn bài thi.

Điểm thưởng: Nhóm sinh viên có điểm thưởng nếu hoàn thành các yêu cầu sau.

- Biểu đồ hiện thị tài nguyên đang sử dụng được hiện thị dưới dạng động, cập nhật theo giây.

Câu 3 (20 điểm): Lập trình logic xử lí việc auto-scaling

- Thiết kế database và thiết kế logic cho phần quản lí thêm, sửa, xoá các luật Auto-scaling trong Horizon. Viết tài liệu mô tả thiết kế và các thành phần.
- Các node VM mới được tạo cần đảm bảo cAdvisor và Docker Swarm mode được khởi chạy ngay khi VM boot.
- Các node VM mới được tạo cần đảm bảo có tags: **olp2016**.

Điểm thưởng: Nhóm sinh viên có điểm thưởng nếu hoàn thành các yêu cầu sau.

- Xây dựng logic xử lí việc auto-scaling, sử dụng thông tin monitor, áp dụng các luật để ra quyết định scale số lượng instances của Docker Swarm Cluster.

Lưu ý:

- Trình bày hướng tiếp cận, cách giải quyết vấn đề, sơ đồ khối, sơ đồ luồng của module auto-scaling và cách thức tích hợp trong hệ thống OpenStack vào tài liệu riêng, nộp cùng mã nguồn bài thi.

Quy cách nộp bài: thí sinh sử dụng git sinh ra file .diff hoặc .patch lưu toàn bộ thay đổi về mã nguồn so với các project con trong OpenStack trên nhánh /stable/mitaka, sau đó nộp file .diff hoặc .patch kèm với tài liệu mô tả hướng tiếp cận giải quyết bài toán và cấu trúc, thiết kế của chương trình.

B. Tham khảo

#	Nội dung	Tham khảo
1	**DevStack**	
	Hướng dẫn cài đặt	http://docs.openstack.org/developer/devstack/guides/single-machine.html http://docs.openstack.org/developer/devstack/
	Hướng dẫn sử dụng	http://www.rushiagr.com/blog/2014/04/03/openstack-in-an-hour-with-devstack/
2	**Django**	
	Framework Django	https://www.djangoproject.com/
	Dự án OpenStack Horizon	http://docs.openstack.org/developer/horizon/
3	**cAdvisor**	
	Kho tài liệu cAdvisor	https://github.com/google/cadvisor/tree/master/docs
	API của cAdvisor	https://github.com/google/cadvisor/blob/master/docs/api.md
4	**Các thư viện phụ thuộc sử dụng trong quá trình thi**	
	D3js - thư viên để mô hình hóa và hiển thị dữ liệu sau quá trình phân tích	https://d3js.org/
5	**Git**	
	Công cụ quản lý mã nguồn	http://git-scm.com/
	Chức năng nộp bài Git-diff	http://www.git-scm.com/docs/git-diff
6	**Phong cách lập trình nguồn mở**	
	Tham khảo phong cách lập trình nguồn mở.	http://techblog.vn/lap-trinh-vien/danh-gia-kha-nang-phat-trien-cua-mot-du-an-phan-mem-tu-do-nguon-mo-2649/
7	**OpenStack Image có sẵn Docker 1.12**	
	Core OS Beta hoặc Alpha 1192.2.0 trở lên	https://coreos.com/os/docs/latest/booting-on-openstack.html
	RancherOS v0.7 trở lên	https://github.com/rancher/os#iso

C. Chuẩn bị

- Chuẩn bị sẵn môi trường DevStack AIO cài đặt OpenStack nhánh Mitaka stable với các project core: Nova, KeyStone, Glance, Neutron.

- Cài đặt sẵn Git và công cụ soạn thảo LibreOffice (OpenOffice) để soạn thảo tài liệu thiết kế.

- Chuẩn bị IDE phù hợp: Eclipse Pydev, PyCharm Community version.

D. Hướng dẫn chấm

Tổng điểm: 100 điểm

Điểm làm bài: 80 điểm,

Điểm cho phong cách lập trình: 5 điểm dành cho việc trình bày và viết mã trong sáng rõ ràng.

Điểm thưởng: tổng ***15 điểm***, trong đó câu 01 có 10 điểm thưởng, câu 02 có 5 điểm thưởng.

Tiêu chí chấm điểm được cho các câu được trình bày chi tiết theo từng câu như sau:

Câu 1: ***35 điểm***

- Xây dựng được tab Auto Scaling: **10 điểm.**
- Xây dựng các control thực hiện thao tác Filtering: combo-box lựa chọn VM, button filter: **5 điểm.**
- Xây dựng được cơ chế hiển thị biểu đồ: 1**0 điểm.**
- Hiển thị được bảng thông tin chi tiết luật auto-scaling theo đúng template của Horizon (Django) và nút Delete Rule, Add Rule: **5 điểm.**
- Mô tả cách thức đóng image có sẵn Docker Swarm mode, cAdvisor và cách thức khởi chạy Docker Engine cùng cAdvisor mỗi khi start VM: **5 điểm.**
- Điểm thưởng: **10 điểm**
    - Filter được biểu đồ chỉ hiển thị CPU hoặc Memory: **10 điểm.**

Câu 2: **25 điểm.**

- Tài liệu mô tả hướng tiếp cận, giải quyết bài toán và thiết kế, cấu trúc chương trình: **10 điểm**.
- Lập trình theo tài liệu thiết kế, hiển thị được đầy đủ dữ liệu trên biểu đồ tài nguyên đang sử dụng. **15 điểm.**

Điểm thưởng: **5 điểm.**

- Xây dựng được biểu đồ hiện thị tài nguyên đang sử dụng dưới dạng động, cập nhật theo giây.

Câu 3: **20 điểm.**

- Tài liệu mô tả hướng tiếp cận, giải quyết bài toán và thiết kế, cấu trúc chương trình: **5 điểm.**
- Cho phép thêm, sửa, xoá luật auto-scaling vào DB: **5 điểm.**
- Lập trình theo tài liệu thiết kế, chạy stress test và chứng minh module auto-scaling hoạt động: **10 điểm.**