Ciclo 2 - Relatório final

Resumo

O conceito de computação em nuvem (em inglês, *cloud computing*) refere-se à utilização da memória, capacidade de armazenamento e poder computacional dos computadores interligados por meio da *Internet*, seguindo o princípio da computação em grade. Com a subutilização destes recursos computacionais em equipamentos modernos, surgiu a necessidade de compartilhar recursos de forma isolada, tornando assim uma máquina física em várias máquinas virtuais, cada uma servindo para um determinado propósito e com seus recursos bem definidos, tentando obter o máximo de aproveitamento dos recursos computacionais.

Prover uma infraestrutura de máquinas virtuais para clientes de forma escalável é hoje uma necessidade mundial e um desafio tecnológico. Este relatório tem por objetivo exibir as características e requisitos necessários para instalar e configurar uma infraestrutura de Cloud para prover máquinas virtuais aos usuários do projeto **EduCloud**.

Introdução

O projeto **EduCloud** é uma iniciativa da **Universidade Estadual Paulista - UNESP**, em parceria com a **Secretaria de Educação do estado de São Paulo**, **Intel** e **ACCEPT**. Esta iniciativa visa fornecer uma infraestrutura de computação em nuvem para hospedar sistemas de apoio ao ensino para as escolas da rede pública do estado de São Paulo.

Uma das grandes vantagens da computação em nuvem é a possibilidade de utilizar *softwares* remotamente, oferecidos na forma de serviços, mais comumente chamado de: *Software as a Service (SaaS)*, sem a necessidade de que esses estejam instalados no disco local do computador do usuário. Todavia o leque de vantagens trazidos pelo *SaaS* é mais amplo, não limitando-se unicamente à economia do recurso de disco do usuário[1]:

- Independência de plataforma, ou seja, é possível acessar a nuvem virtualmente a partir de qualquer computador e sistema operacional;
- As atualizações dos *softwares* são feitas de forma transparente ao usuário, sem necessidade que este intervenha;
- O trabalho colaborativo torna-se mais dinâmico e simples, uma vez que os arquivos criados pelo grupo são editados na nuvem, onde existe um controle de acessos simultâneos para garantir consistência das informações, e instantaneamente disponibilizados para todos os membros da equipe;
- Os *softwares* e os dados podem ser acessados em qualquer lugar, bastando que haja acesso à *Internet*, não estando mais restritos ao ambiente local de computação, nem dependendo da sincronização de mídias removíveis;
- O usuário tem um melhor controle de gastos ao usar aplicativos, pois a maioria dos sistemas de computação em nuvem fornece aplicações gratuitamente, e mesmo quando estas não são gratuitas, são pagas somente pelo tempo de utilização dos recursos. Não é necessário pagar por uma licença integral de uso de *software*;
- Diminuição dos gastos com manutenção da infraestrutura física de redes locais dos usuários, bem como da instalação dos *softwares* nos computadores corporativos, pois esta fica a cargo do provedor da estrutura de nuvem, bastando que os computadores clientes tenham acesso à *Internet*;
- A infraestrutura necessária para uma solução de cloud computing é mais enxuta do que uma solução tradicional de hosting ou collocation[17], consumindo menos energia, refrigeração e espaço físico e consequentemente contribuindo para preservação e uso racional dos recursos naturais.

Para implementar uma infraestrutura de nuvem, o **Núcleo de Computação Científica - NCC**, da **UNESP** pesquisou as principais ferramentas livres[2] do mercado para que juntas, executem esta tarefa.

Para prover uma infraestrutura de nuvem, faz-se atender as seguintes necessidades:

1. Disponibilizar máquinas virtuais de acordo com a demanda de forma automática e rápida, com diferentes configurações de memória, disco e *cpu*;
2. Entregar unidades persistentes de armazenamento de dados, juntamente às máquinas virtuais disponibilizadas;
3. Suportar controle de acesso de usuários e grupos que podem utilizar os recursos da nuvem, bem como um mecanismo de quota de recursos por usuário/grupo;
4. Armazenar as máquinas virtuais e os dados de forma descentralizada, possibilitando também que a arquitetura seja escalável facilmente.

Alguns *softwares* existem para suprir estas necessidades. No capítulo 1 falaremos sobre o projeto *OpenStack*[3], responsável por disponibilizar uma infraestrutura de máquinas virtuais e unidades de armazenamento para as máquinas virtuais. No capítulo 2 falaremos sobre o projeto *Hadoop*[4], responsável por gerenciamento e manipulação de grandes volumes de dados em ambientes largamente distribuídos. No capítulo 5, falaremos sobre os balanceadores de conexões e de máquinas virtuais, fundamental para balancear a carga da nuvem entre os nós *"workers"*, no capítulo 6 falaremos sobre o *OCCI*[5], uma interface de comunicação entre os serviços de cloud.

OpenStack

O OpenStack consiste em uma coleção de projetos open source criada para fornecer uma infraestrutura de computação em nuvem e armazenamento distribuído. A empresa Rackspace[6] e a NASA[7] são os principais contribuidores do conjunto de projetos. Hoje o consórcio possui mais de 100 membros, dentre as principais temos: Rackspace, AT&T, NASA, DreamHost, Safew Labs, Citrix, Dell, AMD, Intel, Canonical, Yahoo e HP. O projeto mantém uma página[8] com uma lista dos principais contribuidores do projeto.

No OpenStack, existem 3 principais componentes:

- Nova - Compute Infrastructure

Nova é um controlador de poder computacional para o OpenStack. Todas as atividades necessárias durante o ciclo de vida de uma instância (uma máquina virtual sendo executada) com o OpenStack são gerenciadas pelo Nova. Isto faz dele uma plataforma de gerenciamento que manipula recursos computacionais (e.g. rede), além de manter o controle de acesso sobre os recursos, e zelar pela escabilidade da infraestrutura de máquinas virtuais. O Nova não provê nenhuma funcionalidade de virtualização por si só; ele usa a interface libvirt[9] para interagir com os hypervisors[10] suportados.

- Swift - Storage Infrastructure

O Swift provê um armazenamento para os objetos (dados) de forma distribuída. Ele é análogo ao serviço da Amazon: Simple Storage Service, ou S3[13]. O Swift é capaz de armazenar bilhões de objetos distribuindos entre os nós. Ele possui um mecanismo de redundância e *failover* e é capaz de armazenar e transmitir os dados. Além disso o Swift é capaz de escalar arquivos considerados muito grandes (petabytes), mesmo que estes sejam em grande quantidade (número de objetos).

- Glance - Imaging Service

O Glance é um sistema de busca e recuperação de imagens de máquinas virtuais. Ele é responsável por gerenciar as imagens das máquinas virtuais, armazenando-as no sistema de arquivos local ou via o OpenStack Object Store - Swift.

Outros componentes, como um front end web (Horizon) e um serviço de autenticação (keystone), também fazem parte do projeto. O OpenStack está disponível no site do projeto[10] e está dispobilizado sobre a licença Apache[11].

O OpenStack atende as necessidades do projeto EduCloud por se tratar de um conjunto de ferramentas que permitem montar uma infra estrutura de cloud utilizando ferramentas livres. Cada subprojeto do OpenStack tem suas próprias características, mas de modo geral as principais vantagens do OpenStack são:

- Escalabilidade.
- Robustez.
- VNC proxy via web browser.
- API com suporte a REST nativo.
- Projeto com apoio de grande empresas.
- Larga adoção na iniciativa privada e comunidade do Software Livre.
- API com *"rate limite"*, ou seja, pode-se estabelecer quotas máximas de requisições na API.
- Comunidade ativa.
- Licença Livre.

As principais desvantagens são:

- Documentação dispersa.
- Muitos subprojetos, o que potencialmente dificulta a instalação e configuração.

Requisitos de Hardware

O OpenStack é uma infraestrutura complexa que exige um conjunto mínimo de algumas máquinas. Embora uma única máquina pode ser utilizada para fins de desenvolvimento, os requisitos mínimos para um ambiente de produção exige uma quantidade maior de máquinas.

A arquitetura mínima é suficiente para testes de staging. Uma vez que o EduCloud tenha passado por essa fase, faz-se necessário a adoção do cenário médio para que sejam tratadas as requisições do treinamento remoto oferecido pela FDE hoje. O cenário mínimo é utilizado apenas fins de desenvolvimento. O cenário médio é robusto o suficiente para substituir a atual arquitetura sobre o qual roda o treinamento hoje. Por fim, quando chegar o momento de se oferecer serviços adicionais através da infraestrutura do projeto EduCloud, tais como ferramentas de ensino à distância, será necessário expandir o cenário médio até que atinja-se o porte do cenário grande. Dessa forma a infraestrutura será suficiente para, além de oferecer o treinamento remoto já disponibilizado, suportar novos serviços educacionais a serem oferecidos a educadores e estudantes da rede pública de ensino do estado de São Paulo.

As recomendações de hardware foram retiradas do site do projeto OpenStack[11]. Abaixo temos 3 propostas de arquiteturas para o projeto EduCloud, com suas respectivas configurações:

Arquitetura proposta - mínima

https://ncc.unesp.br/its/attachments/189/openstack_small.png

Nesta arquitetura proposta, temos as seguintes máquinas:

- 05 Computer Nodes;
- 03 Storage Nodes;
- 04 Standard Servers (Storage Proxy, Nova Server, API Server e Network Controller)
- 04 Switches 10 GBoE;

Com as seguintes configurações:

- Computer Nodes: 2 x 64 bits QuadCore x86 CPU with HT support | 16 GB RAM | 2 x 2TB SATA | 2 x 10GBoE NICs
- Storage Nodes: 2 x 64 bits QuadCore x86 CPU with HT support | 16 GB RAM | 24 2T SATA | 2 x 10GBoE NICs
- Standard Servers: 2 x 64 bits QuadCore x86 CPU with HT support | 16 GB RAM | 2 x 2TB SATA | 2 x 10GBoE NICs

Arquitetura proposta - caso médio

https://ncc.unesp.br/its/attachments/190/openstack_medium.png

Nesta arquitetura proposta, temos as seguintes máquinas:

- 10 Computer Nodes;
- 05 Storage Nodes;
- 08 Standard Servers (Storage Proxy, Nova Server, API Server, Network Controller, VPN, etc..)
- 04 Switches 10 GBoE;

Com as seguintes configurações:

- Computer Nodes: 2 x 64 bits QuadCore x86 CPU with HT support | 32 GB RAM | 2 x 2TB SATA | 2 x 10GBoE NICs
- Storage Nodes: 2 x 64 bits QuadCore x86 CPU with HT support | 32 GB RAM | 24 2T SATA | 2 x 10GBoE NICs
- Standard Servers: 2 x 64 bits QuadCore x86 CPU with HT support | 32 GB RAM | 2 x 2TB SATA | 2 x 10GBoE NICs

Arquitetura proposta - melhor caso

https://ncc.unesp.br/its/attachments/191/openstack_big.png

Nesta arquitetura proposta, temos as seguintes máquinas:

- 20 Computer Nodes;
- 10 Storage Nodes;
- 13 Standard Servers (Storage Proxy, Nova Server, API Server, Network Controller, VPN, etc..)
- 08 Switches 10 GBoE;

Com as seguintes configurações:

- Computer Nodes: 2 x 64 bits QuadCore x86 CPU with HT support | 32 GB RAM | 2 x 2TB SATA | 2 x 10GBoE NICs
- Storage Nodes: 2 x 64 bits QuadCore x86 CPU with HT support | 32 GB RAM | 24 2T SATA | 2 x 10GBoE NICs
- Standard Servers: 2 x 64 bits QuadCore x86 CPU with HT support | 32 GB RAM | 2 x 2TB SATA | 2 x 10GBoE NICs

Tecnologias alternativas

Várias tecnologias podem ser utilizadas para provêr uma infraestrutura de máquinas virtuais. Algumas delas na forma de solução livre, podendo ser instalada em infraestrutura própria, e outras na forma de serviço web, sendo executada na infraestrutura da empresa que a provê.

Entre as tecnologias livres mais conhecidas, temos:

1. Abiquo (na versão mantida pela comunidade).
2. Nimbus.
3. OpenStack.
4. Open Nebula.
5. Eucalyptus.

Entre as tecnologias funcionando como serviço web, temos:

1. Amazon Web Service.
2. Linode.
3. Rackspace.
4. Gogrid.

Estas são apenas as principais alternativas que atendem os requisitos do projeto, levando em consideração popularidade, quantidade de downloads/mês e nível de maturidade do projeto.

Balanceador de conexões HTTP

Um balanceador de conexões é o elemento responsável por redirecionar o fluxo de pacotes de redes vindo da Internet para os *worker nodes* na rede interna, dividindo as conexões de forma mais justa possível para evitar sobre carga em um nó específico. Desta forma, se temos 10 servidores web, o balanceador recebe as conexões e encaminha para os 10 balanceadores dependendo do algoritmo por ele suportado.

Como as conexões HTTP são *stateless*, é fundamental que o balanceador tenha a capacidade de enviar sempre para o mesmo servidor conexões vindas de um mesmo cliente. O balanceador também é responsável por adicionar ou remover *workernodes* da fila de máquinas sempre que necessário.

Nginx vs HAProxy

Durante esta etapa foi feita uma pesquisa inicial sobre os projetos Nginx[14] e HAProxy[15]. Todas as duas tecnologias se encaixam, com vantagens e desvantagens no projeto EduCloud.

O Nginx (pronuncia-se em inglês: Engine X) é um servidor HTTP + proxy reverso e está disponibilizado pela licença BSD. Ele funciona no modo de *loadbalance* via um módulo externo.

Já o HAProxy nao é um servidor HTTP (ele não serve páginas estáticas), foi feito exclusivamente para ser um proxy e *loadbalance* não só HTTP mas para qualquer protocolo TCP. Seu código está licenciado sob a GPL.

Nenhuma das tecnologias conflita com demais componentes do projeto EduCloud, porém o Nginx possui algumas limitações.

Nginx é um servidor e proxy reverso HTTP de alta performance, gratuito e livre, bem como um servidor proxy para IMAP/POP3.

Foi criado pelo programador russo Igor Sysoev em 2005. Segundo pesquisa da Alexa, reportada pela W3Techs. Em presquisa realizada pela Netcraft em Julho de 2010[16], o Nginx é usado em 5.21% dos domínios, tornando-se o quarto servidor web mais popular do mundo.

E funciona em UNIX, Linux, BSD, Mac OS, Solaris, e Microsoft Windows.

Todas as duas tecnologias possui um desenvolvimento ativo.

O HAProxy é usado por empresas como Twitter, Farmville, Fedora, github, Reddit, Stack Overflow, Tumblr, entre outras empresas.

Todos os dois projetos existem uma boa documentação, embora elas estejam espalhadas.

Desvantagens do Nginx

- Possui um algoritmo simples de *round-robin*. Se as requisições são pequenas e retornam rapidamente, o ngnix atende perfeitamente. Porém se a página demora para ser atendida, o Ngnix enviará novas conexões para máquinas que ainda estão atendendo requisições, fazendo com que algumas máquinas fiquem sobrecarregadas e outras não.
- Existem módulos de terceiros para o Ngnix que torce esse balanceamento mais justo, mas ainda não está maduro.
- O Ngnix apenas reconhece um host como *down*, quando o servidor retorna um erro com código 500.
- Apenas atende HTTP.

Desvantagens do HAProxy

Durante a pesquisa inicial a única desvantagem encontrada no HAProxy é que nativamente ele não suporta HTTPS. Porém alguns módulos extras permitem adicionar esta funcionalidade.

Referências

1. O que é computação em nuvem?
2. Definição de software livre
3. http://www.gnu.org/philosophy/free-sw.html
4. Projeto Hadoop

5. Open Cloud Compute Interface - OCCI
6. Rackspace
7. Nasa
8. OpenStack - Empresas Participantes
9. The virtualization API
10. Hypervisor
11. Projeto OpenStack
12. Apache License
13. Amazon Simple Storage Service
14. nginx [engine x] an HTTP and reverse proxy server
15. HAProxy, The Reliable, High Performance TCP/HTTP Load Balancer
16. Netcraft, web server survey
17. Cloud Colocation and data center real estate analysis

Files

openstack_small.png	65.7 KB	30 Apr 2012	Beraldo Leal
openstack_medium.png	95 KB	30 Apr 2012	Beraldo Leal
openstack_big.png	157 KB	30 Apr 2012	Beraldo Leal