Insper

Computação em Nuvem

Cap. 2 - Deployment Orchestration - 4 Aulas

Raul Ikeda - rauligs@insper.edu.br

Grupo:

Objetivos:

1. Entender os conceitos básicos sobre uma plataforma de gerenciamento de aplicações distribuídas.
2. Entender os conceitos básicos de comunicação entre aplicações e serviços.

Pré-requisitos:

1. Terminar o capítulo anterior (Bare metal)
2. Realizar a leitura sobre o Juju. [https://jaas.ai/how-it-works].
3. Realizar a leitura sobre redes de computadores. [Kavis - Cap. 6. Tanenbaum & Steen - Cap. 1 e Cap. 12]

Instalando Juju

- Acesse o maas via SSH e instale o Juju
  - $ sudo snap install juju --classic
- Verifique se o Juju enxerga o MaaS como um provedor de recursos
  - $ juju clouds
- Caso não possua o MaaS como opção, deve adicioná-lo
  - Criar um arquivo de configuração: maas-juju.yaml (atenção com a formatação)

```
clouds:
  maas:
    type: maas
    auth-types: [oauth1]
    endpoint: http://192.168.0.3:5240/MAAS/
```

  - $ juju add-cloud maas maas-juju.yaml
  - $ juju add-credential maas
  - Obs: OAuth é o **TOKEN** (not key) gerado no MaaS
- Vamos criar agora o *juju controller*:
  - $ juju bootstrap maas main --to juju
- Para verificar o status do *controller* utilize:
  - $ juju status
- Para acessar o Dashboard (via browser):
  - $ juju gui
- Note que o endereço do *dashboard* usa o IP interno da subrede. Para acessarmos de qualquer lugar, teríamos que criar uma nova entrada no NAT do roteador.
- Mas existe uma outra opção criando um **SSH TUNNEL** do maas:
  - Do seu terminal (não conecte no maas)
  - $ ssh cloud@[IP Roteador] -L 8080:[IP do Juju GUI]:17070
  - Do seu browser: https://localhost:8080/gui/login/u/admin/default

1. Qual o S.O. utilizado na máquina Juju? Quem o instalou?

2. O programa juju client roda aonde? E o juju service? Como eles interagem entre si?

Deploying Wordpress Low-cost

- Dispobilizar o próprio *controller* como recurso:
  - $ juju switch main:controller
- Finalmente o deploy:
  - $ juju deploy cs:~rikeda/bionic/wordpress --to 0
  - $ juju deploy cs:xenial/mysql --to lxd:0
  - $ juju add-relation wordpress mysql
  - Para acompanhar algum processo de implantação, utilize:
    $ watch -c juju status --color
  - $ juju expose wordpress
  - Após finalizar, configure e acesse o Wordpress. Poste um *Hello World!*.

3. Vocês utilizaram apenas 1 máquina e instalaram o MySQL em um *container* LXD. o que é LXD e LXC?

Deploying Wordpress agora com *Load Balancing*

- Finalizando o *controller* anterior:
  - $ juju kill-controller main
- Faça novamente o *bootstrap*:
  - $ juju bootstrap maas main
- Faça novamente a instalação do Wordpress, mas em máquinas separadas:
  - Acompanhe todo o processo no *Dashboard* e via *juju status*
  - $ juju deploy cs:~rikeda/bionic/wordpress
  - $ juju deploy cs:~rikeda/bionic/mysql
  - $ juju add-relation wordpress mysql
- Agora fazendo escalabilidade horizontal e *load balancing*
  - $ juju add-unit wordpress
  - $ juju deploy cs:bionic/haproxy
  - $ juju add-relation wordpress:website haproxy:reverseproxy
  - $ juju expose haproxy
- Acesse o Wordpress via Browser (com qual IP agora?)
- Destrua o *controller*

4. Explique o conceito por traz do HAProxy (*Reverse Proxy*). Vocês já fizeram algo parecido?

5. Na instalação, o Juju alocou automaticamente 4 máquinas físicas, duas para o Wordpress, uma para o Mysql e uma para o HAProxy. Considerando que é um Hardware próprio, ao contrário do modelo *Public Cloud*, isso é uma característica boa ou ruim?

6. Crie um roteiro de implantação do Wordpress no seu hardware sem utilizar o Juju.

Protótipo I

- Façam o seguinte tutorial **individualmente** na AWS:
  - https://jaas.ai/docs/getting-started-with-charm-development
  - Criar uma t2.micro *client* e fazer o *bootstrap*
  - Há um problema no tutorial, rastreie *issues/pull requests* para solucionar.
  - Use majoritariamente a versão (*series*) Ubuntu Trusty (12 LTS).
- Crie um repositório no Git e coloque o material.

- **Usuários/Repositórios**:

Garbage Collector

- Limpe todo o ambiente, deixando as máquinas livres
- Crie um novo *controller*:
  - $ juju bootstrap maas main

Questões Complementares

1. Juju é uma aplicação distribuída? E o MaaS?

2. Qual a diferença entre REST e RPC?

3. O que é SOAP?

Concluindo

1. O que é e o que faz um *Deployment Orchestrator*? Cite alguns exemplos.

2. Como é o o processo de interação entre o MaaS e o Juju?

3. Defina Aplicação Distribuída, Alta Disponibilidade e *Load Balancing*?

Conclusão: O Juju utilizou o MaaS como provedor de recursos. O MaaS por sua vez forneceu o que havia disponível no rack. Você acha que seria necessária uma máquina de 32Gb para rodar um *Apache Webserver* ou um *Load Balancer*? Extrapole a resposta para um Datacenter real, onde as máquinas possuem configurações **muito** superiores. Como resolver esse problema?