

instalando o OpenStack com o DevStack

Após gastar uns 15 dias [brincando com o Mirantis Fuel](#), resolvi instalar o OpenStack usando o DevStack.

A instalação é a mais simples de todas. Basta baixar os arquivos do repositório e executar o instalador. Eu segui exatamente desta forma, como em quase todos os exemplos que vi na internet. O problema é que o resultado é uma instalação extremamente pobre. Dá para ver umas poucas opções, iniciar umas instâncias e não é possível ver o Neutron funcionando (configurar redes e roteadores). Já estava acostumado a ver o Neutron e o Murano, então não achei legal fazer um *test-drive* em algo tão limitado.

Por sorte inventaram o Google. Após algumas buscas, descobri umas dicas para incrementar minha instalação. Primeiro baixei o DevStack do repositório. Informei o *branch* **mitaka** só para garantir.

Meu usuário padrão possui permissão para sudo. Fui para pasta home dele, mas sem sudo.

```
git clone https://git.openstack.org/openstack-dev/devstack -b stable/mitaka
```

Para a instalação padrão, bastaria agora executar

```
./stack.sh
```

porém eu preciso fazer algumas considerações antes. É possível usar um arquivo de configuração que é bem flexível e possibilita controlar diversos aspectos da instalação, mas é necessário criá-lo na pasta raiz do instalador (mesma pasta do stack.sh).

```
cd devstack
```

```
vim local.conf
```

É obrigatório iniciar o arquivo com a seguinte linha:

```
[[local|localrc]]
```

Na instalação padrão, o instalador irá pedir as senhas que serão posteriormente usadas no sistema. É possível definir estas senhas no arquivo de configuração.

```
ADMIN_PASSWORD=suasenha
MYSQL_PASSWORD=suasenha
RABBIT_PASSWORD=suasenha
SERVICE_PASSWORD=suasenha
SERVICE_TOKEN=xynMtSp9KS59SPHp
SWIFT_HASH=JRBj7ukxMG4tckek
```

As senhas não precisam ser as mesmas.

Quando eu tentava instalar recebia um erro informando que não era possível encontrar a URL de autenticação e a instalação parava. Após procurar muito, encontrei a solução. Editei o arquivo *stack.sh* e localizei a parte onde são definidas as variáveis para acesso ao Keystone:

```
# Set up password auth credentials now that Keystone is
bootstrapped
export OS_IDENTITY_API_VERSION=3
export OS_AUTH_URL=$KEYSTONE_AUTH_URI
export OS_USERNAME=admin
export OS_USER_DOMAIN_ID=default
export OS_PASSWORD=$ADMIN_PASSWORD
export OS_PROJECT_NAME=admin
export OS_PROJECT_DOMAIN_ID=default
export OS_REGION_NAME=$KEYSTONE_REGION_NAME
```

Comente as linhas:

```
# Set up password auth credentials now that Keystone is
bootstrapped
export OS_IDENTITY_API_VERSION=3
export OS_AUTH_URL=$KEYSTONE_AUTH_URI
export OS_USERNAME=admin
# export OS_USER_DOMAIN_ID=default
export OS_PASSWORD=$ADMIN_PASSWORD
export OS_PROJECT_NAME=admin
# export OS_PROJECT_DOMAIN_ID=default
export OS_REGION_NAME=$KEYSTONE_REGION_NAME
```

Agora eu preciso do Neutron e do Murano. Para isso eu usei a configuração que

encontrei [aqui](#). Esta configuração ainda me brindou com o Sahara e o Trove que eu queria tanto! Agora eu pude executar o instalador.

```
./stack.sh
```

Isso demora! Obviamente as coisas não foram bem logo da primeira vez e precisei reinstalar tudo várias vezes. Dizem as más línguas que basta executar

```
./unstack.sh
```

para limpar a instalação e começar novamente. Eu tive alguns problemas com isso e resolvi apagar todo o diretório onde os arquivos são instalados:

```
./unstack.sh  
rm -R /opt/stack/*  
./stack.sh
```

Isso é bem radical e faz as coisas demorarem bem mais, porém garante uma reinstalação mais limpa.

O resultado é um ambiente *tudo-em-um*. Todos os serviços ficam disponíveis na mesma máquina. Isso é ótimo para testar, mas é péssimo se você quer entender a infra do OpenStack.

Não sei se é devido a minha máquina estar conectada à rede por uma interface wifi, mas quando reiniciei o servidor eu perdi o acesso ao Horizon (conexão recusada). Pensei que fosse devido às mexidas que fiz na configuração da rede e reinstalei a bagaça toda mais uma vez só para descobrir que o problema acontece mesmo sem eu mexer em nada. Mais uma vez não fiquei correndo atrás de solução. Não acho que vale a pena empregar tempo para consertar uma coisa que não vou usar.

Resumo: apesar de ser a única instalação que me deu o Trove, eu não gostei. Vou retomar minha instalação no VirtualBox Windows para poder seguir adiante com meus testes pois o caminho é longo. Só lamento o Fuel não instalar o Trove. Eles se amarraram com a Tesora (que é pago). Lamentável.

Mirantis Fuel OpenStack : Resumo

Após terminar de [instalar o OpenStack no VirtualBox em um Ubuntu Server usando o Fuel](#), eu resolvi postar algumas observações:

A [instalação no Windows](#) é de longe a mais prática, mas o desempenho não é dos melhores. Precisei comprar mais 8GB de memória para ficar com 16GB e poder ficar tranquilo.

A [instalação no Ubuntu Server](#) é um pouco chata, mas não é difícil. O desempenho é muito melhor, mas como não há interface gráfica, é preciso alguns macetes para poder acessar o Horizon e o painel do Fuel. Não pude acessar as instâncias das VM por causa do erro que dá quando tento colocar um IP flutuante. Não me incomodei muito em tentar resolver o erro, mas achei uns [vídeos excelentes](#) que explicam muito bem como o Neutron funciona. É capaz da solução estar ali.

A conclusão: se for para testar o OpenStack usando o Fuel, é muito melhor usar o Windows. Deixe o Linux para quando for jogar pra valer.

Apesar disso, vou testar o DevStack.

Criando rotas para Máquinas Virtuais no Ubuntu Server (IPTables)

Eu criei máquinas virtuais no Ubuntu Server que aceitam acesso somente da máquina host (Host Only Interface) . O problema é que o Ubuntu Server não possui browser para que eu acesse estas máquinas e eu não podia modificar as

interfaces para *bridge* porque foram geradas automaticamente. Então precisei criar um redirecionamento de portas para que eu pudesse usar outro computador com Windows em minha rede. Dessa forma, foi só trocar o IP da VM pelo IP da máquina host. No caso da máquina host já usar uma porta com mesmo número que a VM está usando (os dois estão rodando um Apache Server na porta 80) então é necessário colocar a máquina host para escutar em outra porta (8066) e passar o tráfego para a porta 80 da VM.

Mesma porta:

```
iptables -I FORWARD -d 10.20.0.2 -m comment --comment "Accept to forward Fuel DashBoard traffic" -m tcp -p tcp --dport 8443 -j ACCEPT
```

```
iptables -I FORWARD -m comment --comment "Accept to forward Fuel DashBoard return traffic" -s 10.20.0.2 -m tcp -p tcp --sport 8443 -j ACCEPT
```

```
iptables -t nat -I PREROUTING -m tcp -p tcp --dport 8443 -m comment --comment "redirect pkts to virtual machine" -j DNAT --to-destination 10.20.0.2:8443
```

```
iptables -t nat -I POSTROUTING -m comment --comment "NAT the src ip" -d 10.20.0.2 -o vboxnet0 -j MASQUERADE
```

Portas diferentes:

```
iptables -I FORWARD -d 172.16.0.3 -m comment --comment "Accept to forward Horizon DashBoard traffic" -m tcp -p tcp --dport 8066 -j ACCEPT
```

```
iptables -I FORWARD -m comment --comment "Accept to forward Horizon DashBoard return traffic" -s 172.16.0.3 -m tcp -p tcp --sport 80 -j ACCEPT
```

```
iptables -t nat -I PREROUTING -m tcp -p tcp --dport 8066 -m comment --comment "redirect pkts to Horizon Dashboard" -j DNAT --to-destination 172.16.0.3:80
```

```
iptables -t nat -I POSTROUTING -m comment --comment "NAT the src ip" -d 172.16.0.3 -o vboxnet1 -j MASQUERADE
```

[Video] OpenStack Neutron

Encontrei estes excelentes vídeos explicativos sobre a infraestrutura de rede no OpenStack (Neutron):

Os direitos pertencem aos seus respectivos proprietários. Para mais detalhes acesse as respectivas páginas YouTube.

Criando um Servidor de Mapas: Instalando o PostgreSQL

Após [instalar o GeoServer](#), é necessário instalar o gerenciador de banco de dados PostgreSQL juntamente com o PostGIS. Existe um pacote contendo tudo o que você precisa, mas não está no repositório oficial do Ubuntu. Mais informações no [site do PostGIS](#). Não é o escopo deste artigo ensinar como instalar o PostgreSQL, pois o método de instalação envolve muitos passos que podem mudar drasticamente com as versões, o que tornaria meu artigo obsoleto rapidamente. Você pode acompanhar [este tutorial para realizar a instalação](#). Basicamente, é necessário ver a versão do Ubuntu que você está usando (*codename*), adicionar a URL do repositório na sua lista do *apt* e efetuar a instalação. Como estamos em um terminal sem interface gráfica (pelo menos eu recomendo dessa forma) não é necessário instalar o *PgAdmin*.

Não execute o passo a seguir sem antes ter [adicionado o repositório do apt.postgresql.org](http://apt.postgresql.org) ([alternativo](#)) na sua lista do *apt*. Existem outras alternativas, como instalar o PostgreSQL e depois instalar o PostGIS e o pgRouting, mas eu acho esta forma mais fácil. [Neste site](#) você encontra informações úteis sobre instalação e atualização.

```
apt-get install postgresql-9.5-postgis-2.2
```

Já instalei com o PostGIS, que é uma extensão que permite usar dados georreferenciados. Verifique se foi tudo bem com o PostgreSQL. Vou puxar um *sudo* interativo para esta situação, mas você não deve fazer disso um hábito para não causar acidentes:

```
$ sudo -i (não faça disso um hábito)
```

```
$ su postgres
```

```
$ psql
```

```
$> \l ( barra + letra L minúscula : lista os bancos de dados existentes )
```

```
$> \q ( barra + letra Q minúscula : sair do psql )
```

```
$ exit
```

```
$ exit ( do sudo )
```

Aproveita que está no psql e já troca logo a senha do usuário *postgres*. Troque esta senha fantástica pela sua própria (e lembre-se dela!).

```
$ su postgres
```

```
$ psql
```

```
$> ALTER USER Postgres WITH PASSWORD 'zebrasemlista';
```

```
$\q
```

Se tudo foi bem, você deverá ter o PostgreSQL instalado em sua máquina.

Eu alterei a configuração do PostgreSQL para permitir o acesso externo ao servidor e o acesso local sem precisar fornecer senha. Edite o seguinte arquivo :

```
$ vi /etc/postgresql/9.3/main/pg_hba.conf
```

Altere os itens conforme a seguir:

Troque:

local all all peer

Por:

local all all trust

Troque:

host all all peer

Por:

host all all trust

Para melhorar o desempenho durante a importação e a exibição de mapas, modifique estes parâmetros na configuração do PostgreSQL:

```
$ vi /etc/postgresql/9.3/main/postgresql.conf
```

| option | default | recommended |
|--------------------------|-------------|----------------|
| shared_buffers | 24 MB | 4 GB |
| work_mem | 1 MB | 100MB |
| maintenance_work_mem | 16 MB | 4096 MB |
| fsync | on | off |
| autovacuum | on | off (*) |
| checkpoint_segments | | 60 |
| random_page_cost | 4.0 | 1.1 |
| effective_io_concurrency | 1 | 2 |
| temp_tablespace | '' | 'tablespace_1' |
| listen_address | 'localhost' | '*' |

Vamos precisar de uma pasta para o *table space*: um banco de dados para o OpenStreetMap (OSM) e as extensões *postgis*, *postgis_topology* e *hstore*.

```
$ mkdir -p /media/osm/postgres/tablespace_1
```

```
$ chown postgres /media/osm/postgres/tablespace_1
```

```
$ su postgres
```

```
$ psql
```

```
$> create tablespace tablespace_1 location
```

```
'/media/osm/postgres/tablespace_1';  
$>  
CREATE DATABASE osm WITH OWNER boundless tablespace tablespace  
_1;  
$> connect osm;  
$> CREATE EXTENSION postgis;  
$> CREATE EXTENSION postgis_topology;  
$> CREATE EXTENSION hstore;  
$> SELECT postgis_full_version();  
$> \q  
$ exit
```

Reinicie o PostgreSQL:

```
$ service postgresql restart
```

A ferramenta de importação dos dados do OpenStreetMap para o banco de dados do PostgreSQL é o *osm2pgsql*:

```
apt-get install osm2pgsql
```

No próximo post: [baixando o arquivo de dados do OpenStreetMap](#).

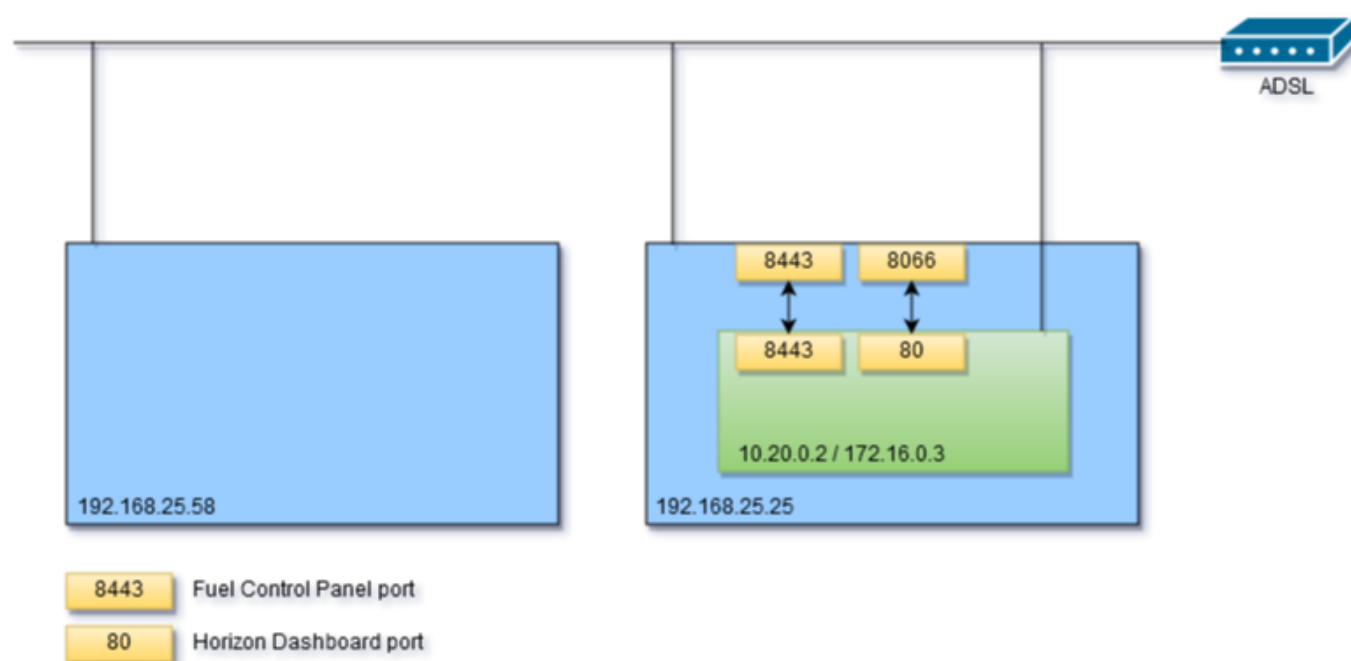
[Deploy do OpenStack no Ubuntu Server](#)

Após [instalar o Mirantis Fuel](#) em um ambiente virtualizado hospedado no Ubuntu Server, chegou a hora de instalar o OpenStack (novamente). O procedimento não é [diferente do anterior](#), hospedado em ambiente Windows. Esperei bastante tempo para que o processo terminasse.

No meio do caminho surgiu um problema: a instalação do Ubuntu no slave que estava destinado ao Cinder deu erro. Não consegui descobrir que tipo de erro foi, apesar de existir um sistema de log bem completo. Parti logo para a solução. Como a instalação nos outros dois nós foi tranquila e não tinha nenhuma opção de instalar somente no nó defeituoso, cliquei em “Deploy” (aba “Dashboard”) novamente. Achei que ia refazer tudo, mas ele teve a decência de instalar somente o nó que deu erro, sem mexer nos outros dois.

Após o Ubuntu ter sido “instalado com sucesso” nos 3 nós, o instalador iniciou o *deploy* do OpenStack. Nesse ponto você pode ir visitar seu tio na Noruega, porque vai demorar (pelo menos na minha máquina demorou).

Após o *deploy*, o link para o Horizon foi exibido. Mesmo problema de acesso para o painel do Fuel: o endereço do Horizon não é exposto para a minha rede local, então preciso redirecionar portas novamente. Nesse caso, como eu tenho um Apache instalado na máquina host ([O PHPVirtualBox lembra?](#)) preciso colocar o host para escutar em outra porta, mas enviar para a porta 80 da máquina onde está o Horizon.



Vou aplicar a mesma regra *iptables* da outra vez, mas trocando as portas. Para ser sincero, não lembro agora se é a mesma máquina que está usando os dois endereços (10.20.0.2 e 172.16.0.3). Vou verificar e edito o post depois. O certo é que a rede 10.20.0.x é a rede administrativa do Fuel (PXE). A 172.16.0.x é a pública e *floating* do OpenStack. Como estamos em um ambiente VirtualBox , é

necessário criar esta ponte da rede externa da máquina host para a rede pública/*floating* do OpenStack. sempre que se desejar um acesso de fora para dentro (de dentro para fora ele já se resolve, pois já vi que os nós são capazes de acessar a internet pela minha rede doméstica).

```
iptables -I FORWARD -d 172.16.0.3 -m comment --comment "Accept to forward Horizon DashBoard traffic" -m tcp -p tcp --dport 8066 -j ACCEPT
```

```
iptables -I FORWARD -m comment --comment "Accept to forward Horizon DashBoard return traffic" -s 172.16.0.3 -m tcp -p tcp --sport 80 -j ACCEPT
```

```
iptables -t nat -I PREROUTING -m tcp -p tcp --dport 8066 -m comment --comment "redirect pkts to Horizon Dashboard" -j DNAT --to-destination 172.16.0.3:80
```

```
iptables -t nat -I POSTROUTING -m comment --comment "NAT the src ip" -d 172.16.0.3 -o vboxnet1 -j MASQUERADE
```

Atualizei minha cópia do *iptables* para que isso não se perca ao reiniciar o servidor:

```
sudo iptables-save > /etc/iptables.conf
```

Feito isso, pude acessar o Horizon a partir do meu notebook, assim como faço com o painel de controle do Fuel.

<http://192.168.25.25:8066/>

No meu caso, tive um problema ao acessar o Horizon:

The server encountered an internal error or misconfiguration and was unable to complete your request.

Eu usei a Conexão de Área de Trabalho Remota do Windows para conectar no nó onde está o Horizon. Se não for a porta 5001 então tente outro. Não lembro de foi um bom palpite ou se eu verifiquei em algum lugar.:

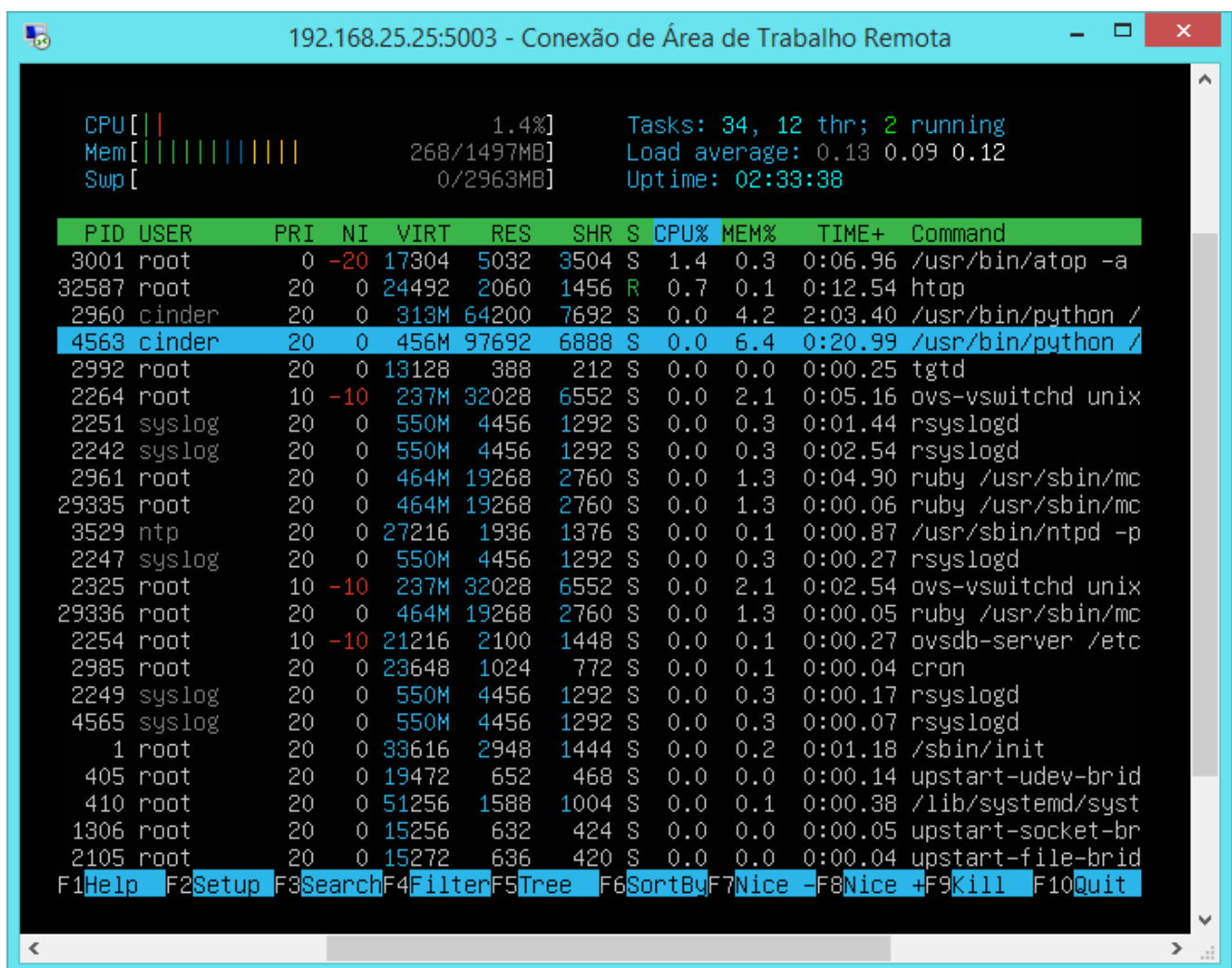
<IP_DO_HOST_VIRTUALBOX>:5001

Uma vez logado como root (senha **r00tme**) eu simplesmente reiniciei o Apache:

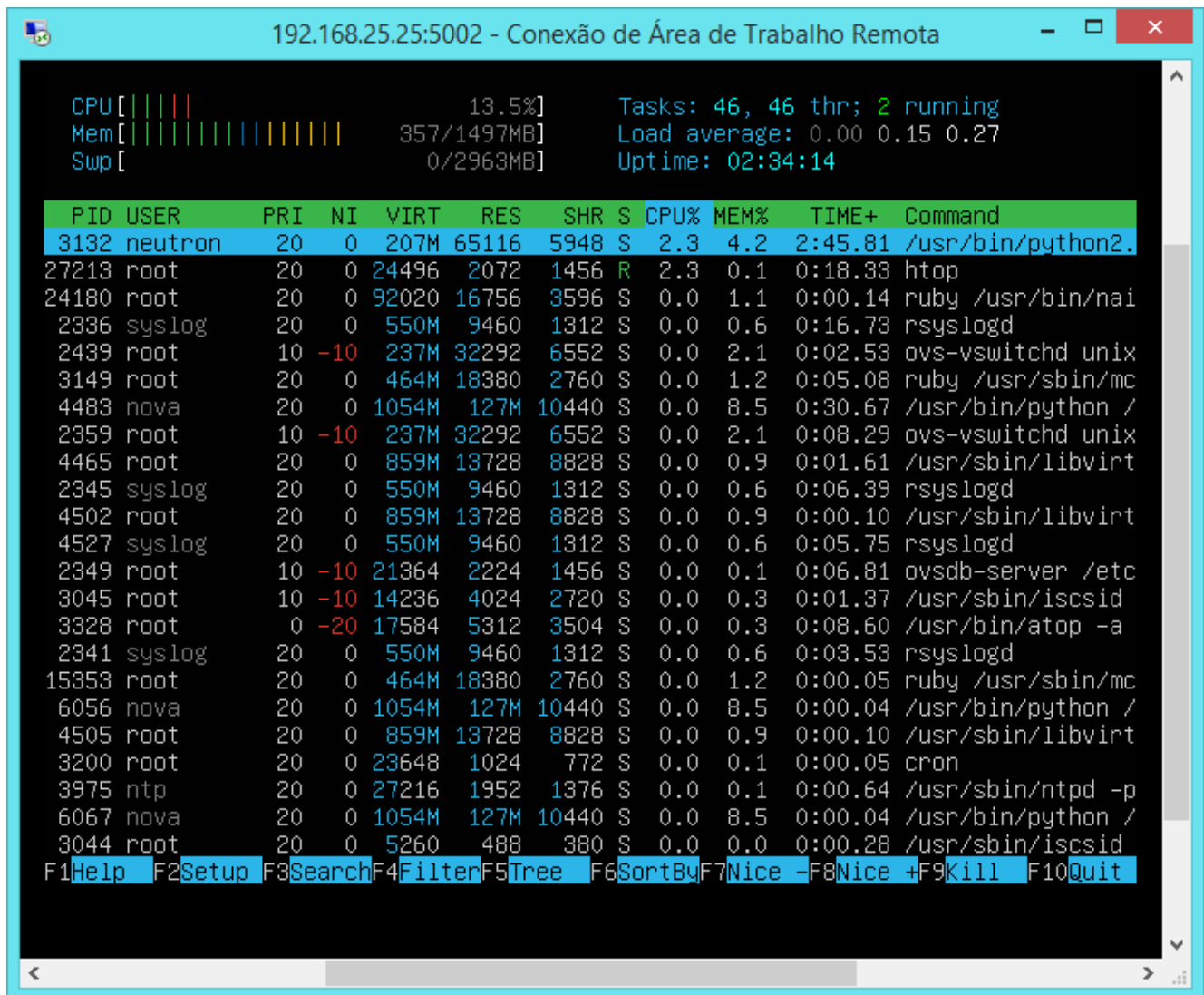
```
service apache2 restart
```

e tudo funcionou como deveria.

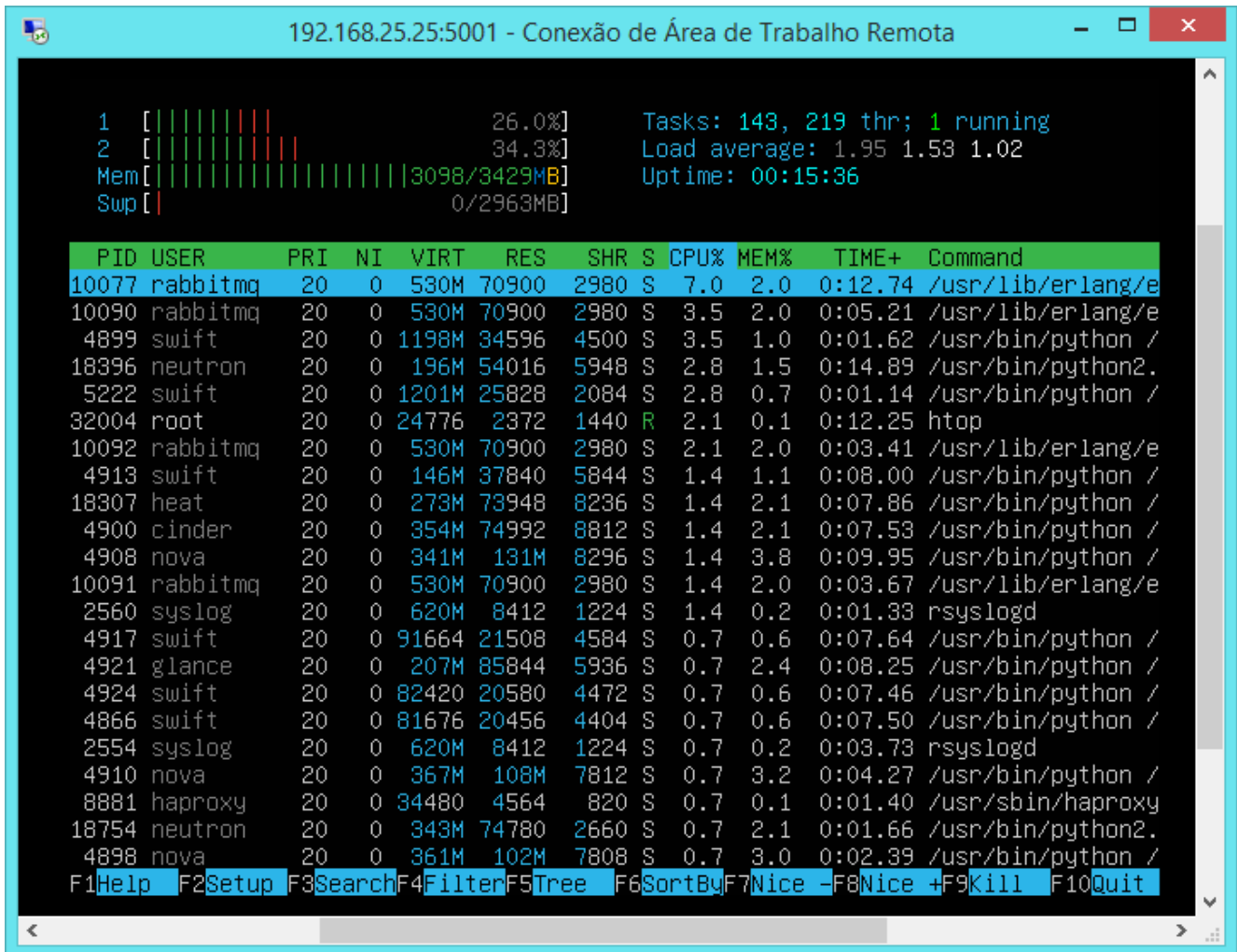
Só por curiosidade eu acessei cada máquina e executei o aplicativo htop do Ubuntu para monitorar a utilização de recursos. Percebi que o Controller estava com 1.3 GB de RAM (assim como todos os outros) mas estava paginando demais e consumindo toda a memória disponível. Apesar de não estar nos manuais, resolvi disponibilizar mais um processador para esta VM e também resolvi aumentar a memória para 3.5 GB de RAM. Após isso, reiniciei a VM do Controller e percebi que o acesso ao Horizon ficou mais fluido e que não estava paginando tanto. Monitorei também a memória disponível na máquina host para ver o impacto da minha modificação. Tudo dentro do normal, pelo menos sem executar nada no OpenStack.



Cinder



Compute



Controller

Após tudo isso, tentei executar uma instância. Escolhi a imagem "TestVM" que veio na instalação, tamanho m1.tiny.

Quando eu seleciono a rede "admin_internal_net" tudo vai bem. A máquina sobe, mas não consigo acessá-la de nenhum lugar (o IP da rede interna não é exposto). Quando eu escolho a rede "admin_floating_net" ocorre um erro:

No valid host was found. There are not enough hosts available.

Não sei bem o que é isso. Procurei pela internet, mas cada lugar dá uma solução diferente e nenhuma resolveu o problema. Aloquei alguns endereços IP flutuantes e criei uma chave, mas também não resolveu. Quando descobrir o que houve eu posto aqui.

No valid host was found. There are not enough hosts available.

File `"/usr/lib/python2.7/dist-`

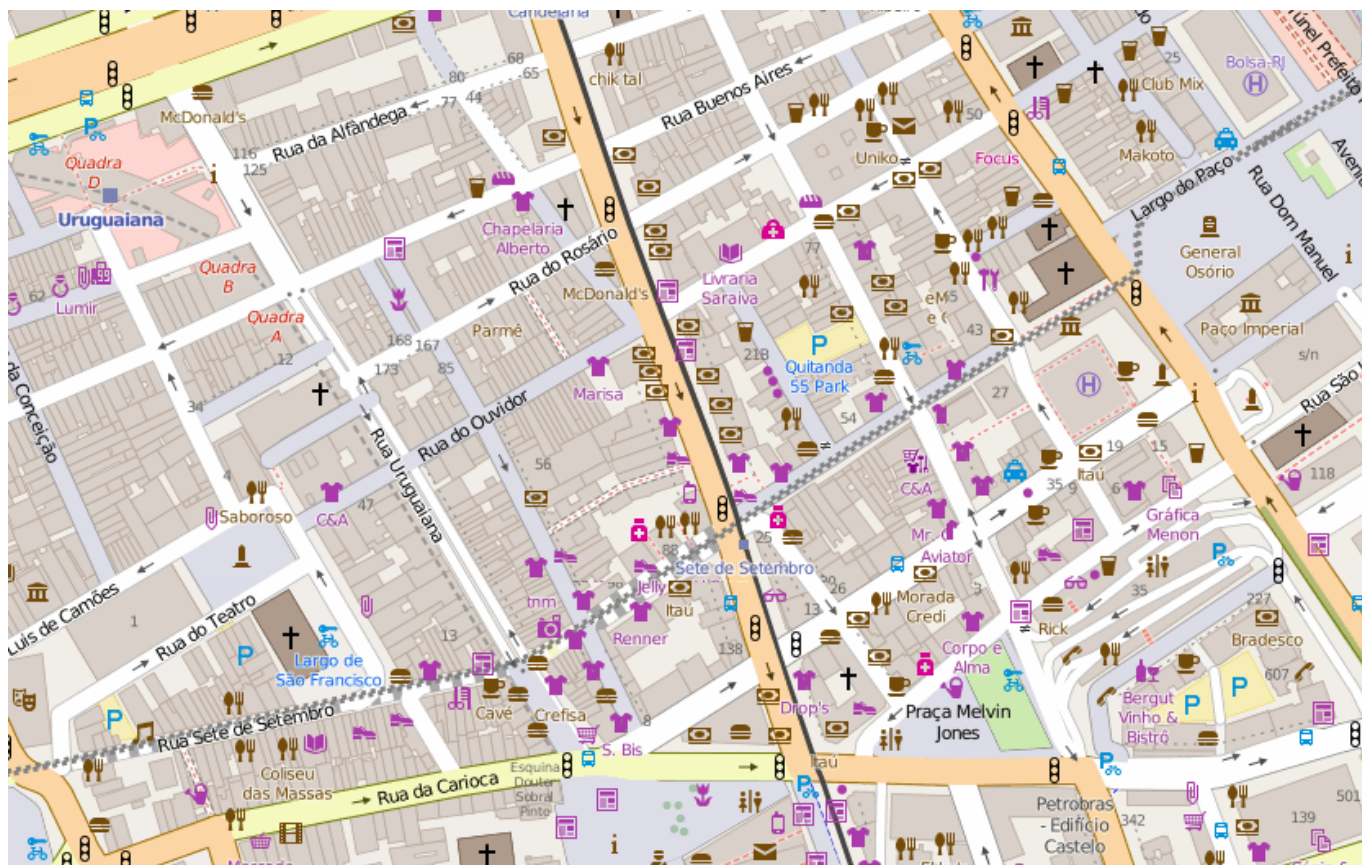
```
packages/nova/conductor/manager.py", line 739, in
build_instances request_spec, filter_properties) File
"/usr/lib/python2.7/dist-packages/nova/scheduler/utils.py",
line 343, in wrapped return func(*args, **kwargs) File
"/usr/lib/python2.7/dist-
packages/nova/scheduler/client/__init__.py", line 52, in
select_destinations context, request_spec, filter_properties)
File
"/usr/lib/python2.7/dist-
packages/nova/scheduler/client/__init__.py", line 37, in
__run_method return getattr(self.instance, __name)(*args,
**kwargs) File
"/usr/lib/python2.7/dist-
packages/nova/scheduler/client/query.py", line 34, in
select_destinations context, request_spec, filter_properties)
File
"/usr/lib/python2.7/dist-
packages/nova/scheduler/rpcapi.py", line 120, in
select_destinations request_spec=request_spec,
filter_properties=filter_properties) File
"/usr/lib/python2.7/dist-
packages/oslo_messaging/rpc/client.py", line 158, in call
retry=self.retry) File
"/usr/lib/python2.7/dist-
packages/oslo_messaging/transport.py", line 90, in _send
timeout=timeout, retry=retry) File
"/usr/lib/python2.7/dist-
packages/oslo_messaging/_drivers/amqpdriver.py", line 533, in
send retry=retry) File
"/usr/lib/python2.7/dist-
packages/oslo_messaging/_drivers/amqpdriver.py", line 524, in
_send raise result
```

Edit:

[A solução para este erro é a seguinte](#): eu estava designando a instância para uma rede externa sem que ela tenha um IP válido na rede fixa (por desconhecimento meu na infraestrutura de rede do Neutron). Toda instância DEVE possuir um IP fixo na rede interna designado pelo Nova e só então receber um IP flutuante externo. A expressão *"not enough hosts available"* significa que ele não encontrou um IP fixo (host) disponível (available) da rede interna designado para a instância que possa ser mapeado para o IP da rede externa.

Criando um servidor de Mapas

Nesta série de posts vou mostrar os passos que segui para criar um servidor de mapas [OpenStreetMap](#) em uma máquina local.



O OpenStreetMap (OSM) possui um incrível banco de dados com várias informações georreferenciadas do mundo inteiro, atualizado diariamente pela comunidade mundial de voluntários. Existe ainda um [vasto catálogo de informações](#) com ilustrações, onde é possível ver os tipos de elementos disponíveis e como consultá-los no banco.

Para possuir um desses, você precisará de um bom hardware. É imprescindível ter um linux (preferencialmente o Ubuntu).

Mínimo:

i5 ou equivalente.

16 GB RAM (8 até roda, mas vai ficar lento).

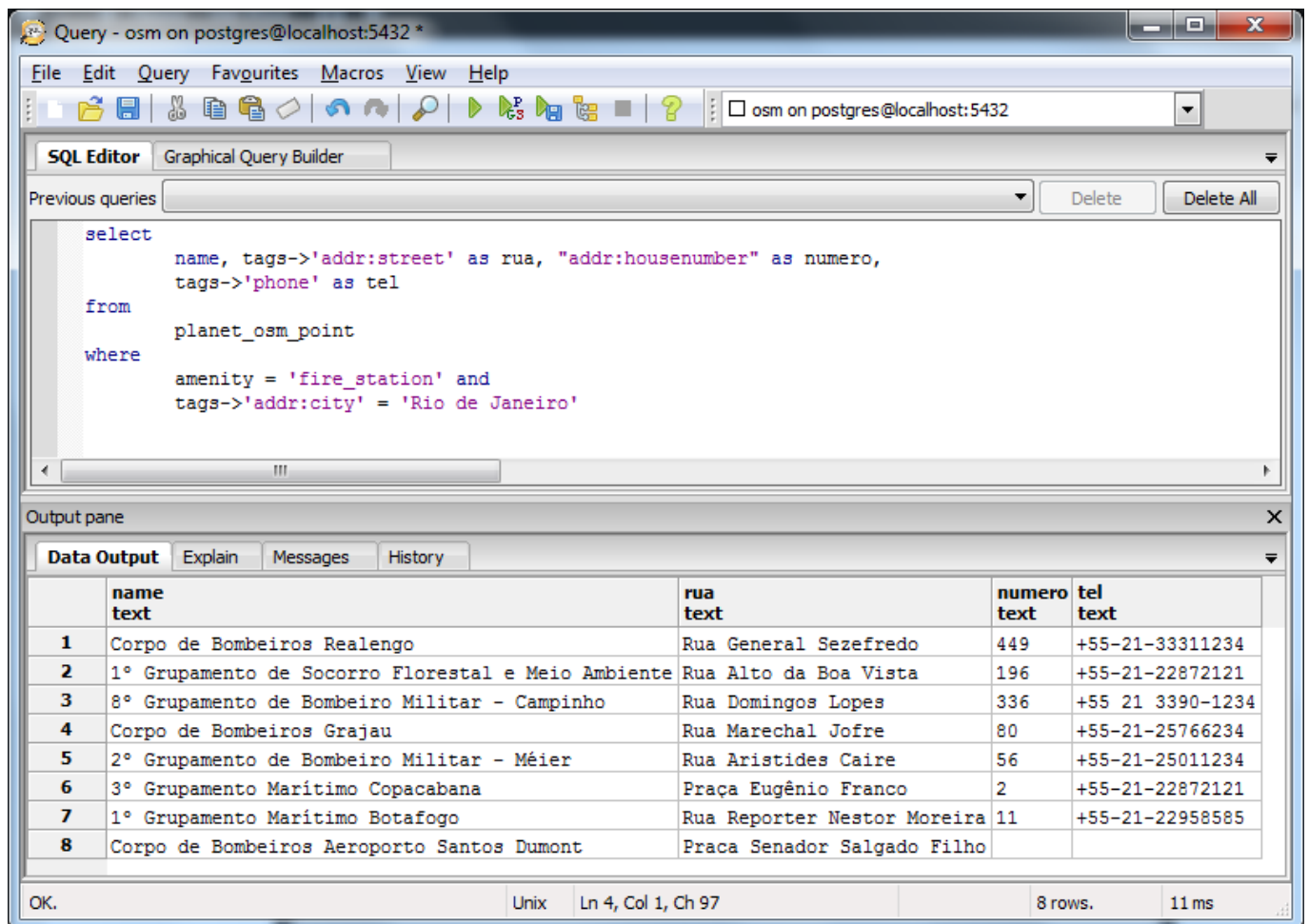
60 GB de HD livres.

Ubuntu 14.04

Internet

Na consulta a seguir eu usei a informação na página 11 do catálogo:

amenity | fire_station



The screenshot shows a PostgreSQL SQL Editor window titled "Query - osm on postgres@localhost:5432 *". The window has a menu bar (File, Edit, Query, Favourites, Macros, View, Help) and a toolbar. The SQL Editor tab is active, displaying the following query:

```
select
  name, tags->'addr:street' as rua, "addr:housenumber" as numero,
  tags->'phone' as tel
from
  planet_osm_point
where
  amenity = 'fire_station' and
  tags->'addr:city' = 'Rio de Janeiro'
```

Below the SQL Editor is the "Output pane" with tabs for "Data Output", "Explain", "Messages", and "History". The "Data Output" tab is selected, showing a table with 8 rows and 5 columns: "name text", "rua text", "numero text", and "tel text".

| | name text | rua text | numero text | tel text |
|---|--|-----------------------------|-------------|------------------|
| 1 | Corpo de Bombeiros Realengo | Rua General Sezefredo | 449 | +55-21-33311234 |
| 2 | 1º Grupamento de Socorro Florestal e Meio Ambiente | Rua Alto da Boa Vista | 196 | +55-21-22872121 |
| 3 | 8º Grupamento de Bombeiro Militar - Campinho | Rua Domingos Lopes | 336 | +55 21 3390-1234 |
| 4 | Corpo de Bombeiros Grajaú | Rua Marechal Jofre | 80 | +55-21-25766234 |
| 5 | 2º Grupamento de Bombeiro Militar - Méier | Rua Aristides Caire | 56 | +55-21-25011234 |
| 6 | 3º Grupamento Marítimo Copacabana | Praça Eugênio Franco | 2 | +55-21-22872121 |
| 7 | 1º Grupamento Marítimo Botafogo | Rua Reporter Nestor Moreira | 11 | +55-21-22958585 |
| 8 | Corpo de Bombeiros Aeroporto Santos Dumont | Praça Senador Salgado Filho | | |

At the bottom of the window, a status bar shows "OK.", "Unix", "Ln 4, Col 1, Ch 97", "8 rows.", and "11 ms".

As 3 tabelas do OSM (*planet_osm_line*, *planet_osm_point* e *planet_osm_polygon*) possuem a mesma estrutura, o que significa que a mesma consulta pode ser aplicada para pontos, polígonos e linhas. Mas como saber onde está o que eu preciso? no catálogo, existe um símbolo que indica em quais tabelas a mesma informação está presente. No caso de *fire_station* (página 11), posso encontrar um quartel de bombeiros como um ponto (latitude, longitude) ou como um polígono (área que o quartel ocupa no mapa).

Query - osm on postgres@localhost:5432 *

File Edit Query Favourites Macros View Help

osm on postgres@localhost:5432

SQL Editor Graphical Query Builder

Previous queries [v] [Delete] [Delete All]

```

select
  name, tags->'addr:street' as rua, "addr:housenumber" as numero,
  tags->'phone' as tel
from
  planet_osm_polygon
where
  amenity = 'fire_station' and
  tags->'addr:city' = 'Rio de Janeiro'

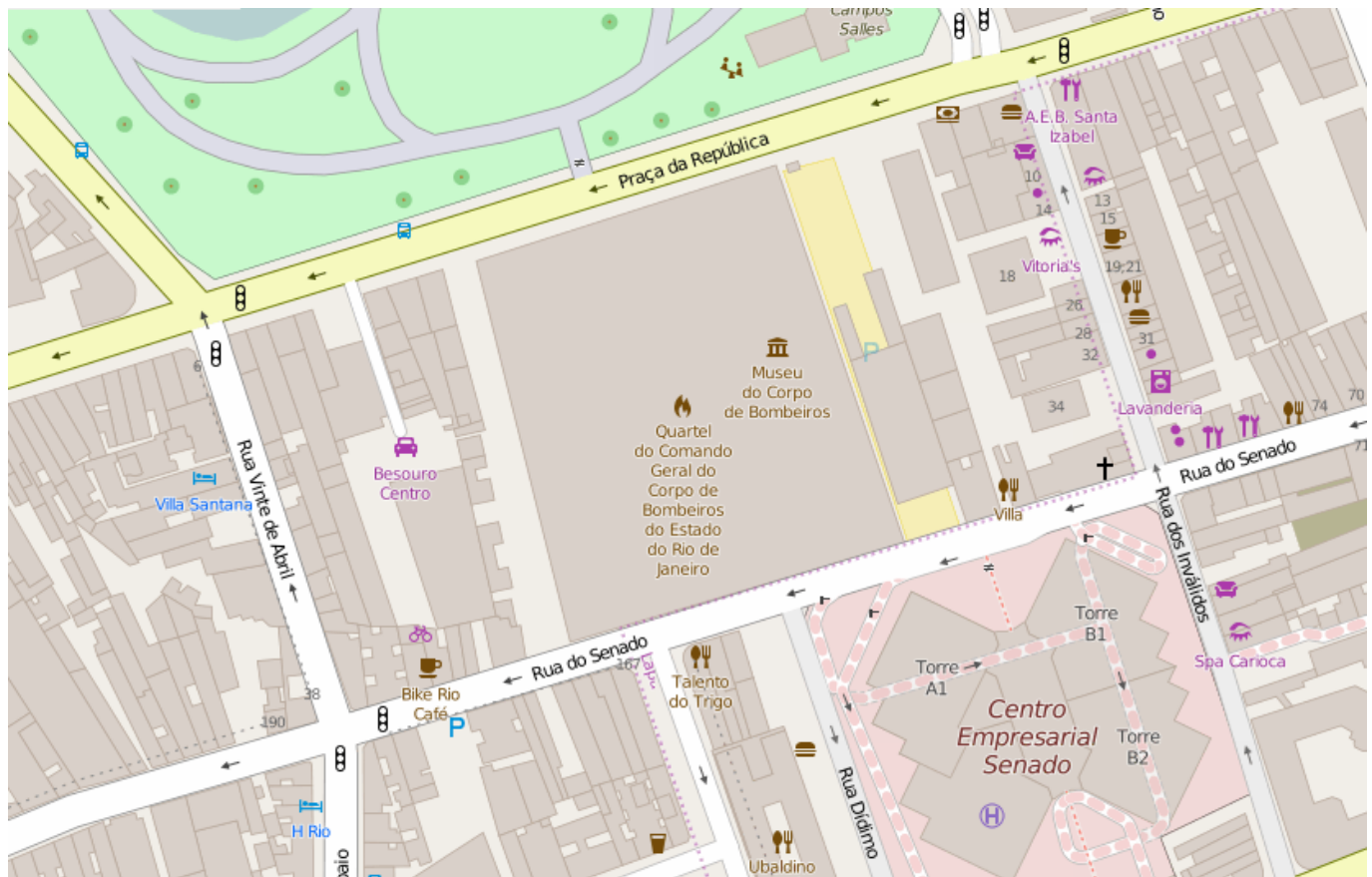
```

Output pane

Data Output Explain Messages History

| | name text | rua text | numero text | tel text |
|---|--|----------------------|----------------|--------------------|
| 1 | 11° Grupamento de Bombeiro Militar - Vila Isabel | Rua Oito de Dezembro | 456 | +55 (21) 8596-9606 |
| 2 | Grupamento de Socorro de Emergência | Praça São Salvador | 4 | +55 21 2285 6386 |
| 3 | Quartel do Comando Geral do Corpo de Bombeiros do Estado | Praça da República | 45 | +55 21 3399-4069 |

OK. Unix Ln 5, Col 27, Ch 122 3 rows. 221 ms



Oportunamente, vou mostrar como usar este mesmo banco de dados para criar um sistema de cálculo de rotas.

Recomendo que você possua uma instalação limpa do Ubuntu com o Java instalado. Se você não lembra como instalar o Java:

```
$ sudo add-apt-repository ppa:webupd8team/java
$ sudo apt-get update
$ sudo apt-get install oracle-java8-installer
```

```
magno@AKRAB:~$ java -version
```

```
java version "1.8.0_91"
Java(TM) SE Runtime Environment (build 1.8.0_91-b14)
Java HotSpot(TM) 64-Bit Server VM (build 25.91-b14, mixed
mode)
```

```
$ sudo apt-get install oracle-java8-set-default
```

O próximo passo é [instalar o Tomcat](#):

```
sudo apt-get install tomcat7
```

Agora é só instalar o GeoServer, que é, em curtas palavras, um software capaz de transformar dados georreferenciados em imagens de mapas que podem ser acessadas usando um navegador de internet.

Vá no site e baixe a versão mais atual:

<http://geoserver.org/release/stable/>

Existem algumas opções de download que você deve considerar:

- Web Archive : É um arquivo Java WAR que você pode instalar em um servidor web (Tomcat, por exemplo). É a minha preferida.
- Windows Installer: É o famoso “deixa que eu faço, mas não te digo o que fiz.”
- Platform Independent Binary: A opção para quem não tem um servidor Tomcat já pronto e também não quer ter trabalho. Ele executa seu próprio servidor web.

Vamos escolher o WAR. Baixe o arquivo e copie para a pasta

`/var/lib/tomcat7/webapps/ .`

```
$ wget  
http://sourceforge.net/projects/geoserver/files/GeoServer/2.9.  
0/geoserver-2.9.0-war.zip
```

URL alternativa:

```
http://ufpr.dl.sourceforge.net/project/geoserver/GeoServer/2.9  
.0/geoserver-2.9.0-war.zip
```

Versão mais recente (não testada):

```
http://sourceforge.net/projects/geoserver/files/GeoServer/2.10  
.0/geoserver-2.10.0-war.zip  
http://ufpr.dl.sourceforge.net/project/geoserver/GeoServer/2.1  
0.0/geoserver-2.10.0-war.zip
```

```
$ unzip geoserver-2.9.0-war.zip  
$ cp geoserver.war /var/lib/tomcat7/webapps
```

Reinicie o Tomcat

```
sudo service tomcat7 restart
```

Se tudo deu certo, você pode apontar seu navegador para

`http://localhost:8080/geoserver`

Faça login em seu novo servidor:

Usuário: admin

Senha: geoserver

Servidor

- Status do servidor
- Logs do GeoServer
- Informações de contato
- Sobre o GeoServer

Dados

- Visualizador de Camada
- Espacios de trabajo
- Almacenes
- Camadas
- Grupos de camadas
- Estilos

Servicios

- WFS
- WCS
- WMS

Ajustes

- Global
- JAI
- Acceso de cobertura

Tile Caching

- Tile Layers
- Configuración de cache

Bem-vindo

Bem-vindo

Este GeoServer pertence a The ancient geographies INC.

1.420 Camadas

+ Agregar capas

35 Almacenes

+ Agregar almacenes

18 Espacios de trabajo

+ Agregar espacios de trabajo

⚠ Por favor leia o arquivo `/opt/geoserver/data_dir/security/masterpw.info` e remova após a leitura. Este arquivo é um **risco de segurança**.

⚠ O serviço de usuário/grupo padrão deve utilizar codificação digest para a senha.

⚠ No strong cryptography available, installation of the unrestricted policy jar files is recommended

Esta instância GeoServer está sendo executada na versão `$(versão)`. Para obter mais informações, por favor entre em contato com o administrador pelo email [administrador](#).

Serviços Disponibilizados

WCS

1.1.0

1.1.1

Não, você não pode trocar o idioma. Ele detecta do seu browser e o mais perto do português que chegará é o espanhol.

Não é o escopo deste post ensinar a usar o GeoServer, mas você estiver curioso, poderá ir em “Visualizador de Camada” e clicar em “OpenLayers” ao lado de cada nome para ver um dos mapas de exemplo.

[No próximo post](#) vou mostrar como instalar o banco de dados PostgreSQL e como colocar todos os quiosques de cachorro-quente da América do Sul na palma de sua mão.

OpenStack: Mirantis Fuel no

Ubuntu Server

Tendo conseguido [instalar uma versão do OpenStack usando o Mirantis Fuel](#) em um ambiente VirtualBox hospedado em uma máquina Windows, meu próximo passo foi fazer o mesmo em um ambiente VirtualBox hospedado em uma máquina Ubuntu 14.04.

Eu sei, já existe versão LTS mais recente. Me deixe em paz, panela velha é que faz comida boa.

Peguei um HD de 300 e poucos GB de um notebook velho que tinha aqui e pluguei na mesma máquina de antes (um Core i5 com 8GB de RAM). É o que o sistema oferece. Tive o cuidado de remover meu HD com o Windows. Só Jesus salva.

Com meu Ubuntu fresquinho, fiz a prece do recém-instalado:

```
sudo apt-get update
sudo apt-get dist-upgrade
sudo apt-get autoremove
```

Instalando o VirtualBox com Interface Gráfica WEB:

Parti então para a instalação do VirtualBox. Para tanto, usei [este tutorial](#). Só modifiquei as versões usadas. Vou repetir os passos usados:

```
apt-get install build-essential dkms
```

Crie o seguinte arquivo:

```
vi /etc/apt/sources.list.d/virtualbox.list
```

e coloque nele o seguinte conteúdo:

```
deb http://download.virtualbox.org/virtualbox/debian trusty
contrib
```

Baixe e instale a chave:

```
wget http://download.virtualbox.org/virtualbox/debian/oracle_vbox.
sc -0- | sudo apt-key add -
```

e instale o VirtualBox. A minha versão difere do tutorial. Para verificar a versão mais recente vá no site do VirtualBox e troque o número da versão de acordo.

```
sudo apt-get update && sudo apt-get install VirtualBox-5.1
```

Minha versão do Extension Pack também é diferente (vá para seu diretório *home*):

```
wget
```

```
http://download.virtualbox.org/virtualbox/5.1.2/Oracle_VM_VirtualBox_Extension_Pack-5.1.2-108956.vbox-extpack
```

instale:

```
VBoxManage                                extpack                                install  
Oracle_VM_VirtualBox_Extension_Pack-5.1.2-108956.vbox-extpack
```

Esse próximo passo é importante. É necessário criar um usuário para acessar o VirtualBox pelo PHP. Eu não tinha me ligado nisso e quando instalei o OpenStack usei o meu usuário como root em modo sudo (*sudo -i*) então não pude ver as máquinas virtuais criadas. Tive que fazer tudo denovo do início. Use o mesmo usuário aqui e na instalação do OpenStack. Para instalar o OpenStack ele precisará ter poderes de root. Eu usei o usuário padrão da instalação do Ubuntu (primeiro usuário).

Neste ponto o tutorial manda criar um usuário, mas eu vou usar o meu usuário padrão do Ubuntu.

Verifique a instalação do VirtualBox:

```
sudo /etc/init.d/vboxdrv status
```

Caso algo esteja errado, execute:

```
sudo /etc/init.d/vboxdrv setup
```

É necessário um servidor Apache e o PHP. Se já tiver, pule esta etapa e verifique a pasta de destino se confere com sua instalação (/var/www/html/). Não esqueça também de conferir se os demais pacotes do PHP estão instalados:

```
sudo apt-get install apache2 php5 php5-common php-soap php5-gd
```

Baixei o PHPVirtualBox. Minha versão é diferente da do tutorial (vá para seu diretório *home*).

```
wget
http://sourceforge.net/projects/phpvirtualbox/files/phpvirtual
box-5.0-5.zip
```

Descompacte o arquivo. Se não tiver o unzip, o apt-get resolve seu problema.

```
unzip phpvirtualbox*.zip
```

Copiar o PHPVirtualBox para o servidor Apache.

```
sudo mv phpvirtualbox-5.0-5 /var/www/html/phpvirtualbox
```

Criar o arquivo de configuração e configurar o usuário que irá controlar o VirtualBox e suas máquinas. Precisa ser o mesmo que irá instalar o OpenStack (com poderes sudo).

```
sudo cp /var/www/html/phpvirtualbox/config.php-example
/var/www/html/phpvirtualbox/config.php
sudo vi /var/www/html/phpvirtualbox/config.php
```

Modifique as seguintes linhas:

```
[...]
var $username = 'magno';
var $password = 'minhasenha';
[...]
```

O próximo passo é informar ao VirtualBox este usuário. Crie o seguinte arquivo:

```
sudo vi /etc/default/virtualbox
```

e coloque a seguinte linha nele (claro que você vai trocar meu nome pelo usuário que existe na sua instalação):

```
VBOXWEB_USER=magno
```

Reinicie o VirtualBox WebService:

```
sudo /etc/init.d/vboxweb-service start
```

agora é só navegar para o seu novo servidor:

```
http://<SEU_IP>/phpvirtualbox/
```

Usuário: admin

Senha: admin

Na minha instalação eu recebi o seguinte erro após logar no painel do PhpVirtualBox:

```
This version of phpVirtualBox (5.0-5) is incompatible with
VirtualBox 5.1.2. You probably need to download the latest
phpVirtualBox 5.1-x.
```

O caso é que coloquei a versão mais recente do VirtualBox e do PhpVirtualBox mas elas não estavam ainda equiparadas. Mas até agora não tive maiores problemas com isso.

Mas tudo isso é desnecessário se você for fera na linha de comando do VirtualBox. Eu não sou e preciso muito da interface gráfica do VirtualBox. O mais legal disso é que você poderá pegar a console de qualquer máquina virtualizada usando o Console Remoto do Windows apontando para o IP da máquina host (sim da host, não da máquina virtual). O VirtualBox te dirá qual máquina está escutando em que porta).

Instalando o Fuel:

A partir daqui, o procedimento foi igual ao descrito [no primeiro post da série](#). Copiei a imagem ISO para a pasta *iso* do pacote de scripts e executei dessa vez a versão para 8GB, pois agora estou ostentando recursos.

Precisei informar ao script de instalação que agora não há mais interface gráfica no VirtualBox. Isso é muito importante, pois o script não vai conseguir iniciar as VM corretamente. Para tanto, editei o arquivo **config.sh**:

Localize a linha contendo o código a seguir:

```
# Set to 1 to run VirtualBox in headless mode
headless=0
```

e troque para

```
headless=1
```

Após isso, executei a instalação.

```
./launch_8GB.sh
```

Fiquei impressionado com a melhora no tempo de instalação. Incrivelmente mais rápido. Após tudo instalado, o sistema me informou o mesmo modo de acesso de antes:

`http://10.20.0.2:8443`

Só tem um probleminha: essa rede não está exposta para minha rede local e não posso modificar nada do ambiente de rede do Fuel sem danificar nada. A interface que controla a rede 10.20.0.0 é *host only*. Logo, só posso acessar o painel do Fuel a partir da máquina host e não tenho interface gráfica para isso.

Deixei uns dois dias esse problema de molho na minha cabeça e concluí que a solução não estava no VirtualBox, mas no próprio Ubuntu (e é uma solução até bem simples, se você sabe para onde olhar).

Resolvi criar um redirecionamento de portas da interface de rede da máquina host (que dá para minha rede local) para a interface de rede virtual que controla a rede 10.20.0.0. Na verdade, só tive a ideia porque tenho conhecimentos de infraestrutura de redes, mas não tinha a menor ideia de como fazer isso, pois não domino muito bem o *iptables* do linux. Tá bom, eu confesso: não entendo uma vírgula disso.

Recorri ao pessoal do [Unix StackExchange](https://unix.stackexchange.com/) e a resposta veio rápido. Primeiro postei a configuração das interfaces de rede da máquina host:

```
root@AKRAB:~# ifconfig
lo          Link encap:Loopback Local
            inet end.: 127.0.0.1  Masc:255.0.0.0
            endereço inet6: ::1/128 Escopo:Máquina
            UP LOOPBACK RUNNING  MTU:65536  Métrica:1
            pacotes RX:19685 erros:0 descartados:0 excesso:0
quadro:0
            Pacotes TX:19685 erros:0 descartados:0 excesso:0
portadora:0
            colisões:0 txqueuelen:0
            RX bytes:7674590 (7.6 MB) TX bytes:7674590 (7.6 MB)

vboxnet0    Link encap:Ethernet      Endereço de HW
0a:00:27:00:00:00
            inet end.: 10.20.0.1    Bcast:10.20.0.255
Masc:255.255.255.0
```

```
    endereço inet6: fe80::800:27ff:fe00:0/64 Escopo:Link
    UP BROADCAST RUNNING MULTICAST MTU:1500 Métrica:1
    pacotes RX:0 erros:0 descartados:0 excesso:0
quadro:0
    Pacotes TX:167 erros:0 descartados:0 excesso:0
portadora:0
    colisões:0 txqueuelen:1000
    RX bytes:0 (0.0 B) TX bytes:22260 (22.2 KB)

vboxnet1    Link encap:Ethernet    Endereço de HW
0a:00:27:00:00:01
    inet end.: 172.16.0.254    Bcast:172.16.0.255
Masc:255.255.255.0
    endereço inet6: fe80::800:27ff:fe00:1/64 Escopo:Link
    UP BROADCAST RUNNING MULTICAST MTU:1500 Métrica:1
    pacotes RX:0 erros:0 descartados:0 excesso:0
quadro:0
    Pacotes TX:437 erros:0 descartados:0 excesso:0
portadora:0
    colisões:0 txqueuelen:1000
    RX bytes:0 (0.0 B) TX bytes:137886 (137.8 KB)

vboxnet2    Link encap:Ethernet    Endereço de HW
0a:00:27:00:00:02
    inet end.: 172.16.1.1    Bcast:172.16.1.255
Masc:255.255.255.0
    endereço inet6: fe80::800:27ff:fe00:2/64 Escopo:Link
    UP BROADCAST RUNNING MULTICAST MTU:1500 Métrica:1
    pacotes RX:0 erros:0 descartados:0 excesso:0
quadro:0
    Pacotes TX:464 erros:0 descartados:0 excesso:0
portadora:0
    colisões:0 txqueuelen:1000
    RX bytes:0 (0.0 B) TX bytes:150336 (150.3 KB)

wlan0        Link encap:Ethernet    Endereço de HW
00:13:46:94:18:c1
    inet end.: 192.168.25.25    Bcast:192.168.25.255
Masc:255.255.255.0
    endereço inet6: fe80::213:46ff:fe94:18c1/64
Escopo:Link
```

```
UP BROADCAST RUNNING MULTICAST  MTU:1500  Métrica:1
    pacotes RX:2354945 erros:0 descartados:4 excesso:0
quadro:0
    Pacotes TX:1237088 erros:0 descartados:0 excesso:0
portadora:0
    colisões:0 txqueuelen:1000
    RX bytes:3455421823 (3.4 GB) TX bytes:103231994
(103.2 MB)

root@AKRAB:~#
```

O que me interessa é a interface exposta na minha rede local (*wlan0*) e a interface que responde pela rede 10.20.0.0 (*vboxnet0*). Então a dica para o redirecionamento de pacotes que recebi foi:

```
iptables -I FORWARD -d 10.20.0.2 -m comment --comment "Accept
to forward Fuel DashBoard traffic" -m tcp -p tcp --dport 8443
-j ACCEPT
```

```
iptables -I FORWARD -m comment --comment "Accept to forward
Fuel DashBoard return traffic" -s 10.20.0.2 -m tcp -p tcp --
sport 8443 -j ACCEPT
```

```
iptables -t nat -I PREROUTING -m tcp -p tcp --dport 8443 -m
comment --comment "redirect pkts to virtual machine" -j DNAT -
-to-destination 10.20.0.2:8443
```

```
iptables -t nat -I POSTROUTING -m comment --comment "NAT the
src ip" -d 10.20.0.2 -o vboxnet0 -j MASQUERADE
```

Isso ainda não resolve o problema. É necessário informar ao linux que eu quero ativar o *IP Forwarding*:

```
echo 1 > /proc/sys/net/ipv4/ip_forward
```

Ainda não basta. Ao reiniciar o servidor tudo isso será perdido. É necessário editar o arquivo

```
/etc/sysctl.conf
```

e remover o comentário da linha

```
net.ipv4.ip_forward=1
```

Ainda, o *iptables* vai evaporar também ao reiniciar o servidor. Para tornar permanente as regras de redirecionamento, é necessário gravar as regras atuais para um arquivo

```
sudo iptables-save > /etc/iptables.conf
```

e então editar o seguinte arquivo:

```
/etc/rc.local
```

e colocar o seguinte conteúdo nele (antes do “exit 0”):

```
# Load iptables rules from this file
iptables-restore < /etc/iptables.conf
```

Pronto. Agora foi possível acessar o painel de controle do Fuel através do endereço

```
https://<IP_DO_SERVIDOR>:8443
```

Ele irá redirecionar o acesso para a máquina virtual *master* no IP 10.20.0.2 na mesma porta.

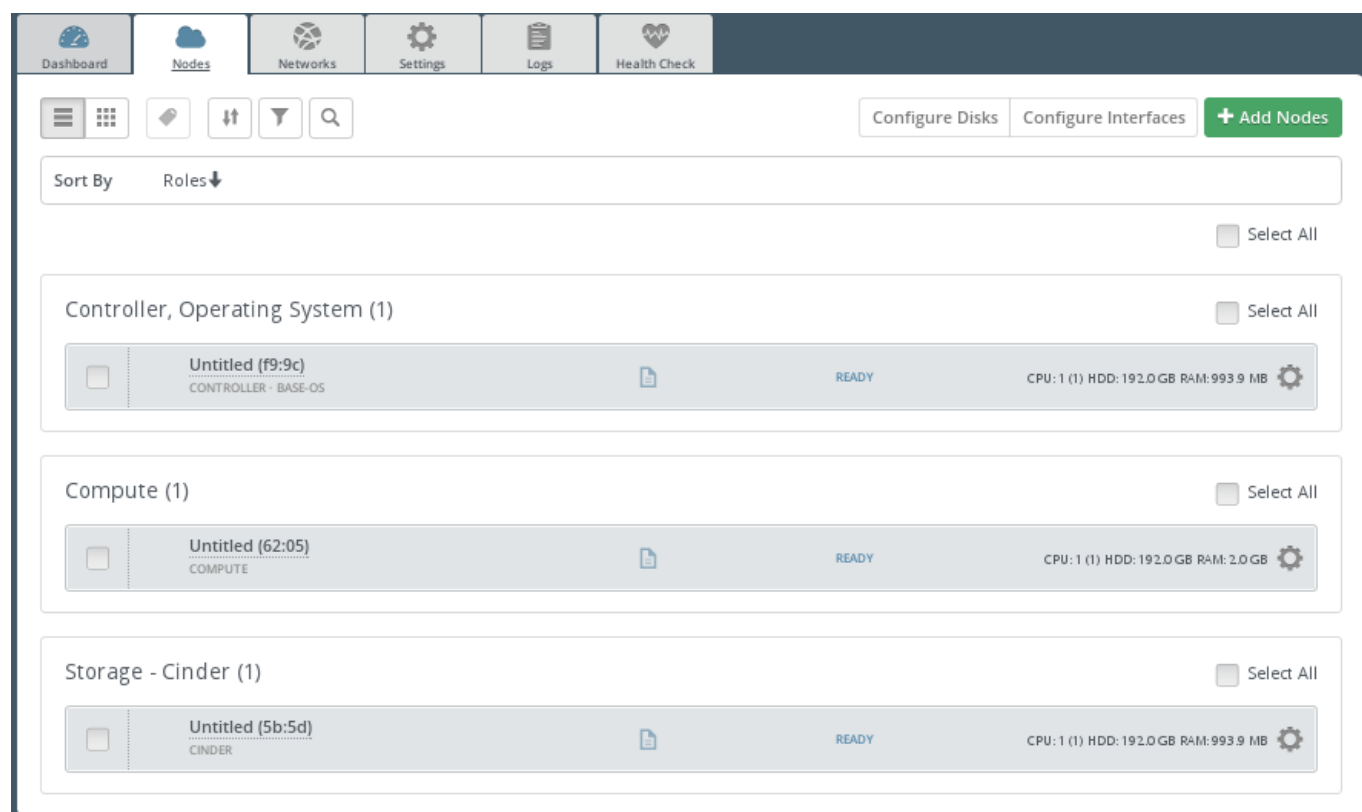
Não sei se isso é algum problema da instalação do VirtualBox, mas o caso é que, ao reiniciar o servidor, minha interface *wlan0* sobe mas não conecta com a rede. Estou precisando executar um *ifdown* seguido de *ifup* em *wlan0* para ela conectar. Fica a dica se você tiver somente uma interface Wi-Fi e acontecer aí.

[No próximo post](#): e agora José?

OpenStack: Criação da Nuvem

Após [instalar o Mirantis Fuel em um VirtualBox no Windows](#), o próximo passo foi criar a minha nuvem. A minha primeira pergunta foi: o que fazer com as 3

máquinas virtuais que me foram entregues?



A resposta eu encontrei no arquivo **config.sh** (configuração mínima):

Uma configuração sem HA (Alta Disponibilidade) : 1 Controller e 1 Compute (a que sobra coloca como Controller se quiser).

Uma configuração sem HA com Cinder : 1 Controller, 1 Compute e 1 Cinder

Uma configuração HA é necessário no mínimo 3 Controllers e 1 Compute.

Encontrei nesse arquivo algumas dicas de hardware, visto que uma das máquinas tinha configuração de memória diferente, mas todas possuem disco igual. A máquina com memória diferente seria para quem?

Um Controller requer no mínimo 1.5 G de RAM

Um Compute requer no mínimo 1 G de RAM ou as instâncias de VM

não rodam.

Se tiver um Cinder dedicado, este precisará de 768M e o Ceph, 1 G.

Recomendo a leitura do arquivo **config.sh** para obter informações valiosas sobre configuração.

De posse destas informações e do vídeo do primeiro post eu pude fazer meu primeiro *deploy* do OpenStack. Tentei um Controller, um Compute e um Cinder, como no vídeo.

Demorou bastante. Primeiro ele instala um Ubuntu em todos os *slaves* e depois faz a instalação dos programas necessários para rodar o OpenStack. Na primeira tentativa tive um problema com um dos nós: deu um erro após a instalação do Ubuntu porque aparentemente o nó ficou *offline* durante a instalação. Para não ter uma instalação contaminada eu apaguei todas as VM e fiz tudo novamente desde o início.

Ao final do processo eu tinha meu ambiente OpenStack instalado. O link para acessar o Horizon foi entregue na aba “*Dashboard*” do painel de controle do Fuel (imagem acima).

Pude brincar um pouco e até instanciar uma máquina, mas algumas perguntas logo apareceram: e depois? este ambiente é o bastante para algo sério? todo o ambiente está encapsulado na máquina host que executa o VirtualBox. como acessar alguma coisa de fora desta máquina se nenhuma interface de rede está exposta na minha rede doméstica? Estas perguntas reforçaram minha teoria de que o Mirantis Fuel, embora seja uma ferramenta excelente para simplificar o processo de instalação do OpenStack e oferecer uma boa oportunidade para ver funcionando sem precisar de um parque computacional completo, é apenas um ambiente de teste.

Também oferece uma oportunidade de estudar a complexa configuração de rede do OpenStack. Quando for criar um ambiente mais sério acompanhando os manuais oficiais, isso será de grande valia para saber quantas interfaces de rede cada máquina terá, qual será o papel de cada uma e onde vão se conectar nos diversos roteadores que serão usados para montar as diferentes redes.

Outra observação: não é possível tirar máximo proveito do ambiente hospedando-o em uma máquina Windows. Partirei para o próximo passo, que será instalar o VirtualBox em um servidor Ubuntu e instalar tudo novamente, assim poderei ter mais recursos de hardware disponíveis (minha máquina não é lá essas coisas). Depois de ficar satisfeito, partirei para a instalação do Trove, que infelizmente não vem no Fuel. Precisaré fazer isso “na mão”.

Após tudo isso, usarei meus conhecimentos em Java para acessar a API do OpenStack e manipular os recursos da nuvem programaticamente, pois tenho planos para criar um Sistema Gerenciador de Workflows Científicos nativo para o OpenStack.

Estou bem empolgado para acessar o Trove e o Swift usando Java, mas até lá o caminho me parece longo ainda. Como dizem, *toda grande jornada começa com o primeiro passo*.

OpenStack : Primeiros passos

Semana passada resolvi me aventurar a entender o OpenStack. Tudo começou quando meu amigo Ricardo Campisano do Laboratório Interinstitucional de e-Astronomia me perguntou se eu conhecia. Bastou isso para ativar minha curiosidade.

Primeiro tentei instalar com o AutoPilot do Ubuntu, mas a infraestrutura exigida não é para amadores. Precisava de alguma coisa mais prática e rápida.

Achei o [Mirantis Fuel](#). É perfeito para quem está começando. Futuramente vou dar uma olhada a versão do [DevStack](#), que possui a opção de instalar o OpenStack diretamente em uma máquina física.

Do site <http://docs.mirantis.com>:

You can install Fuel on VirtualBox and use that to deploy a Mirantis OpenStack

environment for demonstration and evaluation purposes. Mirantis provides scripts that create and configure all the VMs required for a test environment, including the Master node and Slave nodes.

Baixei a versão 8.0 poucos dias antes de lançarem a 9.0. Trata-se de um arquivo ISO e um arquivo ZIP contendo alguns scripts bash. Não tenho um ambiente Ubuntu com virtualbox pronto, mas felizmente as [instruções de instalação](#) permitem usar o cygwin.

Copiei o arquivo ISO para a pasta “iso” que veio junto com os scripts e coloquei tudo no cygwin. Após isso, executei o comando de instalação. Existem 3 versões do script de execução: normal, 8GB e 16GB. Rodei o normal, que promete 3 slaves e um master. O master não é propriamente o OpenStack, mas o controle de *deploy* do Fuel.

Não entrei em detalhes sobre as outras opções. Quando a gente ganha um brinquedo de natal quer logo sair brincando, então digitei:

```
./launch.sh
```

Esperei um bom tempo. Minha máquina é um Core i5 com 8GB de RAM. Não ficou tão pesado quanto eu achei, mas o OpenStack ainda não estava instalado, só o Fuel (que, a propósito, é algo como um “criador de nuvens”, permitindo criar uma infraestrutura OpenStack de acordo com suas necessidades).

Resolvi me apoiar em algum vídeo do Youtube, que possui uma vasta coleção de vídeos sobre o assunto. Escolhi este, que apesar de estar desatualizado, serviu.

Como prometido, recebi 3 máquinas slave e uma master.

O usuário de todas elas é root,
Senha: r00tme (zero no lugar da letra "o")

Fui instruído a acessar o painel de controle do Fuel na máquina master através do endereço

<https://10.20.0.2:8443/>



Usuário: admin

Senha: admin

Percebi que as máquinas *slave* são apenas “cascas vazias” com boot PXE. Acredito que se eu iniciar uma máquina física com boot PXE na mesma rede ela vá fazer parte do cluster. Vou testar isso ainda. Eu clonei um dos nós para testar e ele foi acrescentado ao cluster. Obviamente eu resetei o endereço MAC dele e a descrição, que é o próprio endereço MAC. Apesar da tentação de ter mais um slave, apaguei a nova máquina, afinal, tartaruga não sobe em árvore. Se decidiram por apenas 3, que seja.

Por falar em rede, a infraestrutura de rede criada me deixou um pouco confuso. Procurei por material na internet que explique como cada máquina está posicionada na rede, mas não tive muita paciência em me aprofundar nisso por enquanto. Quando for passar para o mundo físico eu vou enfrentar isso de frente. Basta saber que foram criadas 3 novas interfaces de rede no VirtualBox: 2 Host Only e 1 NAT e nenhuma delas expõe a rede interna para a sua rede doméstica. Fiquei tentado a mudar uma das redes no arquivo de configuração para coincidir com minha rede doméstica, mas ele criou um gateway com o mesmo endereço do meu Modem ADSL e acabou tirando minha máquina da internet. Resolvi que deveria instalar o produto exatamente como manda o manual e parei de experimentar coisas antes de conhecer tudo. Apaguei as VM e instalei tudo novamente.

Pelo pouco que entendi, uma rede é a rede administrativa, outra é a rede pública e outra é a rede interna usada para comunicação do cluster e boot PXE.

Encontrei [este site com boas informações](#) sobre a rede e requerimentos:

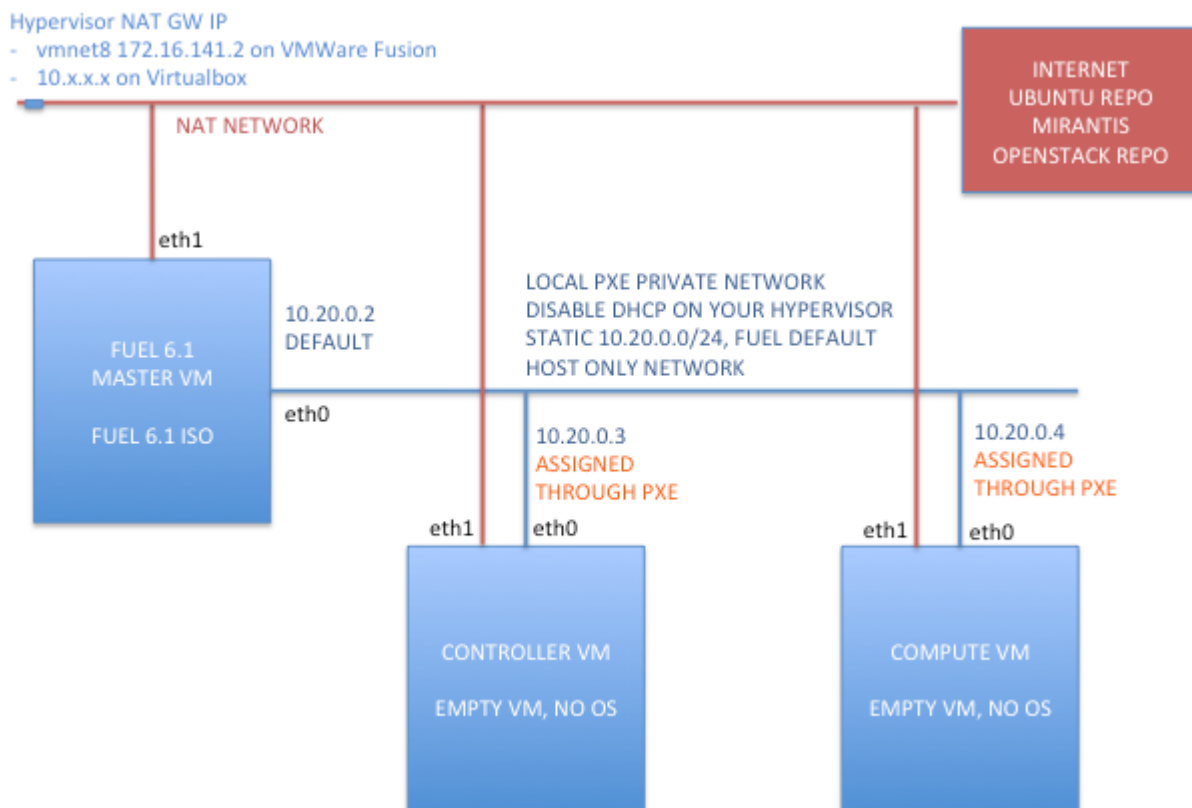
Hypervisor (Virtualbox, VMWare Fusion). This guide is based on VMWare Fusion on Mac (3 GHz i7, 16GB RAM), but the steps are similar for Virtualbox

Fuel ISO for Fuel Master VM, download from Mirantis Website (min 41 GB HDD), use 50GB VM Storage, 2 vCPU, 4GB RAM
2 network adapters: 1 for NAT, 1 for host to host/host-only
first boot on Fuel ISO

Node 1 -> for OpenStack Controller VM
50 GB space, 2vCPU, 2GB RAM
Empty VM with no OS, OS will be provisioned by Fuel Master, OpenStack role will be deployed by Fuel Master
2 network adapters: 1 for NAT, 1 for host to host/host-only
first boot through PXE on eth0

Node 2 -> for OpenStack Compute VM
50 GB space, 2vCPU, 2GB RAM
Empty VM with no OS, OS will be provisioned by Fuel Master, OpenStack role will be deployed by Fuel Master
2 network adapters: 1 for NAT, 1 for host to host/host-only
first boot through PXE on eth0

Connectivity from all 3 VM to Internet (Ubuntu Repo and Mirantis OpenStack Repo).



No [próximo post](#) vou continuar descrevendo a instalação de uma nuvem OpenStack.