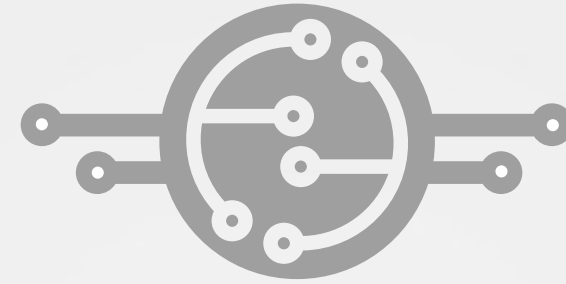


November 2023



Alta disponibilidade e tolerância a falhas

Com o uso do OpenNebula

@gabrielrih
gabrielrih@gmail.com

Alta Disponibilidade

Alta disponibilidade significa que um sistema, ou componente, pode operar em um alto nível, de forma contínua, sem intervenção por um determinado período de tempo.

Busca **reduzir downtime**.



99.999%

Tolerância a falhas

É a capacidade de um sistema continuar funcionando mesmo quando algum componente falhar.

Busca **zerar downtime**.



Elementos básicos

Redundância

Garantindo que cada elemento crítico do sistema tenha um componente adicional que possa tomar o seu lugar em caso de falha.

Failover

Mecanismo, manual ou automatizado, que permite “trocar” o componente atual pelo componente adicional (backup).

Monitoramento

Coleta dados do sistema e detecta quando um componente falhar ou parar de responder.

HA no OpenNebula

Falha em host físico (nodes)

O OpenNebula permite que hooks sejam disparados a partir da alteração do estado de um host físico. Esse hook por sua vez pode **executar um script** (escrito normalmente em Ruby).

Um exemplo de uso é **migrar automaticamente todas as VMs** assim que este host fique com o estado ERROR. Reduzindo o downtime e intervenção manual.

Falha em máquina virtual


Usando a mesma lógica de hooks, ações podem ser aplicadas automaticamente nas VMs de clientes.

Um exemplo de uso é **iniciar as VMs** caso algum host reinicie e a VM não suba automaticamente.

HA para os serviços do OpenNebula

Garante Alta Disponibilidade dos serviços de Frontend do OpenNebula (opennebula e opennebula-scheduler). O serviço precisa ser configurado em múltiplas VMs e utiliza um protocolo distribuído que possui mecanismo de tolerância a falhas.



The background of the slide is a complex, abstract network diagram. It consists of numerous small, dark grey circular nodes connected by thin, light grey lines. The nodes are distributed across the entire frame, with a higher density in the upper left and lower right areas. The lines form a web-like structure, with some nodes having multiple connections, suggesting a distributed or interconnected system. The overall aesthetic is technical and modern, typical of cloud computing or network infrastructure presentations.

HA para os serviços do OpenNebula

Principais componentes do OpenNebula

OpenNebula Daemon

oned: Serviço core da plataforma. Gerencia os nós do cluster, usuários,, grupos, redes virtuais e mais.

Scheduler

Responsável pelo planejamento dos deploys de VMs.

Database

É onde o estado do OpenNebula é persistido.

MySQL.

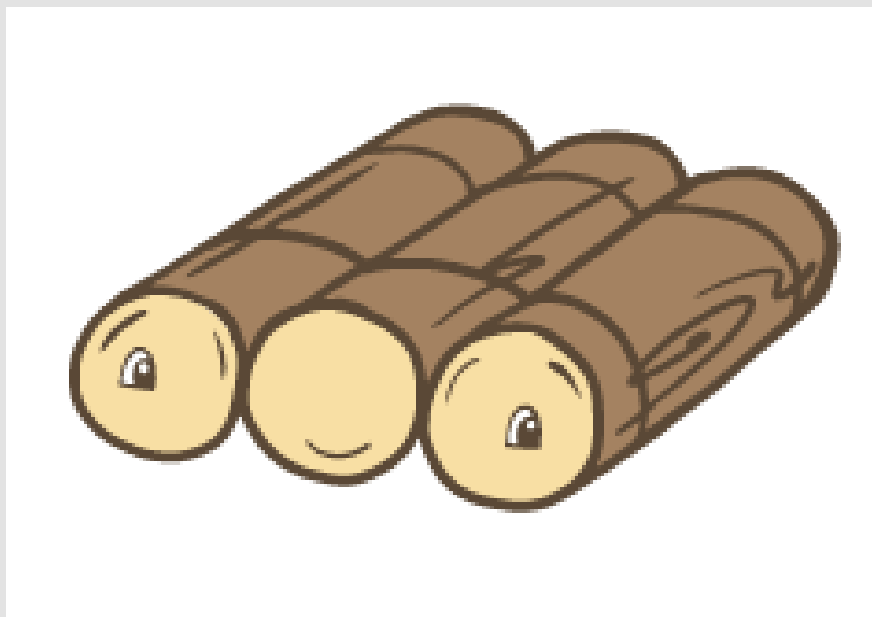
Sunstone

WebUI (Acesso gráfico). Frontend.

Algoritmo Raft

Raft é um algoritmo de consenso projetado para que seja de simples entendimento.

- Leader e followers.
- As modificações são replicadas.
- O leader se mantém ativo com o envio de heartbeats.



Requisitos

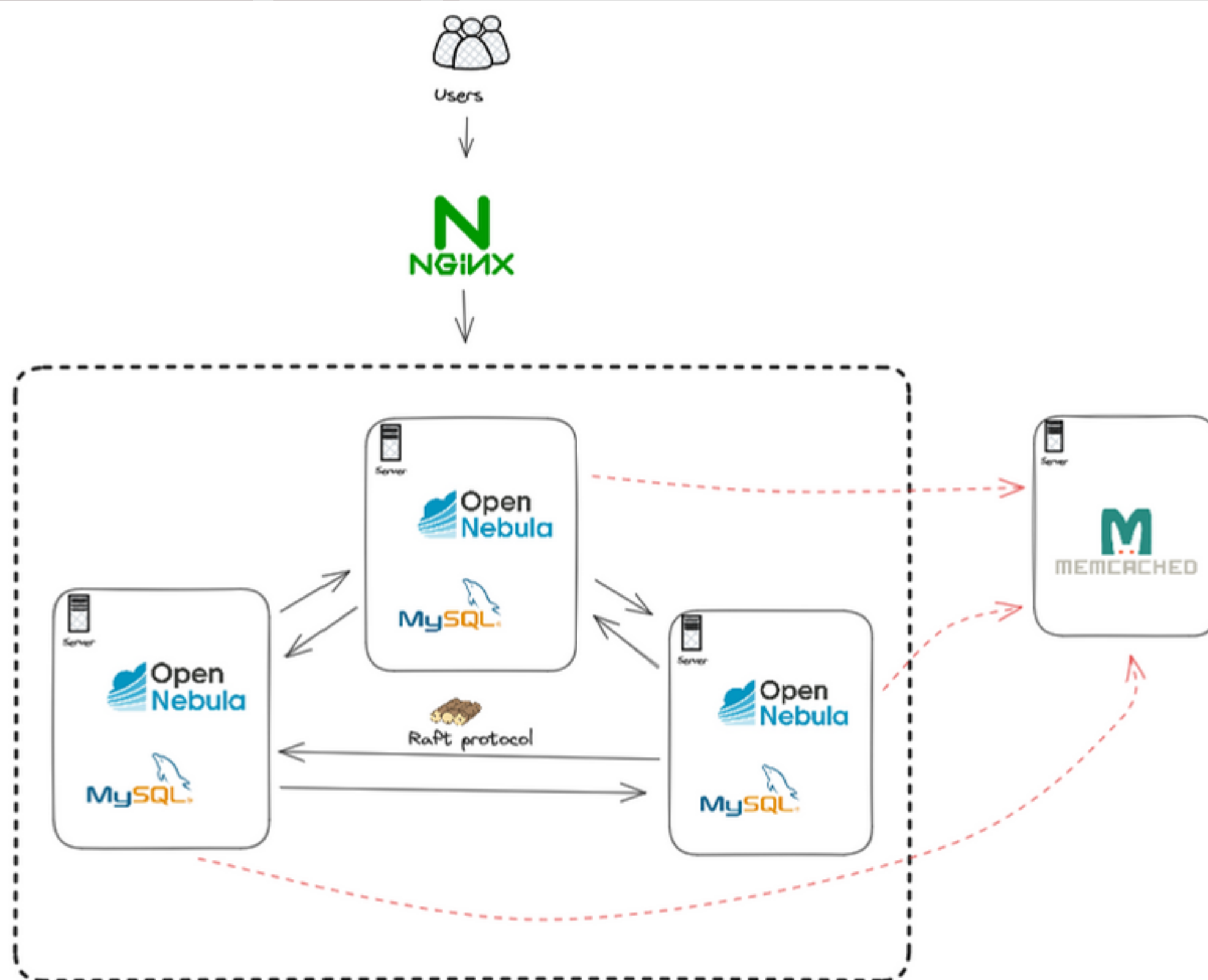
Every HA cluster requires:

- An odd number of servers (three is recommended).
- (Recommended) identical server capacities.
- The same software configuration of the servers. (The sole difference would be the `SERVER_ID` field in `/etc/one/oned.conf`.)
- A working database connection of the same type. MySQL is recommended.
- All the servers must share the credentials.
- Floating IP which will be assigned to the *leader*.
- A shared filesystem.

The servers should be configured in the following way:

- Sunstone (with or without Apache/Passenger) running on all the nodes.
- Shared datastores must be mounted on all the nodes.

Arquitetura proposta



Tecnologias:

- Memcached: Salvar sessão de usuário
- Nginx: Balanceador de carga
- MySQL: Banco de dados para OpenNebula
- Raft: Algoritmo de consenso

Memcached e Nginx são necessários somente para HA do Sunstone.

Cronograma de execução



Configurar o nginx em uma VM, de preferência separada, e balancear o tráfego entre os três nós.

2

Subir uma VM com o memcached corretamente configurado.

memcached

4

nginx



1

oned

Subir os três nós no cluster OpenNebula com os serviços principais executando:

- oned
- scheduler
- mysql

Validar shared filesystem.

3

Sunstone

Configurar o serviço do sunstone nas três VMs utilizando o memcached.



Vantagens: Porque HA?

Sistema tolerante a falhas

Garante que o sistema continuará operando mesmo se um componente falha.

Manutenção

Possibilita a manutenção de componentes sem downtime no sistema.

- Atualização de versão do sistema.
- Patches de segurança.
- Novas funcionalidades.

Escalabilidade

O balanceador de tráfico permite um maior escalabilidade do sistema já que posso incluir novos nós com zero indisponibilidade.

Distribuição de carga


Com o balanceador de carga posso distribuir requisições de leituras para os followers reduzindo a carga do leader.



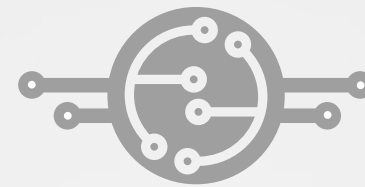


Referências

s

- [OpenNebula overview](#)
 - [OpenNebula FrontEnd HA](#)
 - [Sunstone for Large Deployments](#)
 - [The Raft consensus algorithm](#)
- 

November 2023



OBRIGADO!

@gabrielrih
gabrielrih@gmail.com