

**Escola Superior de Tecnologia de Tomar**

Relatório do Projeto 1 de

Computação Cloud e Virtualização

Projeto Realizado por:

José Santos 21892

Rui Paiva 12754

Conteúdo

[Introdução 3](#_Toc139199229)

[Componente A 4](#_Toc139199230)

[Objetivo 4](#_Toc139199231)

[Solução teórica 5](#_Toc139199232)

[Load Balancing (Web) 6](#_Toc139199233)

[Web Server 6](#_Toc139199234)

[Service Discovery 6](#_Toc139199235)

[Load Balancing (DB) 6](#_Toc139199236)

[WebSockets 6](#_Toc139199237)

[Solução desenvolvida 7](#_Toc139199238)

[Resultados 7](#_Toc139199239)

[Componente B 10](#_Toc139199240)

# Introdução

Este projeto tem como objetivo a criação de infraestruturas com o intuito de possibilitar escalabilidade, disponibilidade de serviços elevada e provisionamento dos mesmos. O projeto está dividido em duas componentes, a primeira (A) baseia-se no uso de máquinas virtuais e a segunda (B) no uso de containers.

Link Parte A Github: <https://github.com/Joselpsantos/MEI-Cloud-Computing-A/>

Link Parte B Github: <https://github.com/Joselpsantos/MEI-Cloud-Computing-B/>

# Componente A

Na primeira componente do projeto obtivemos duas soluções, uma teórica e uma prática que foi implementada.

## Objetivo

Esta solução procura propor uma solução com enfase na escalabilidade horizontal, elasticidade e redundância. A aplicação proposta faz uso de diversos serviços, tais como: base de dados relacional, WebSockets e sessões.

## Solução teórica

Figura Solução Ideal

Na solução apresentada na Figura 1, procurámos separar os serviços do nó web fornecido, desta forma é possível garantir uma maior redundância visto que se um nó web falhar os outros serviços não são afetados. Na grande parte das camadas da infraestrutura existe redundância à exceção da distribuição de carga das bases de dados. A versão apresentada contém as seguintes camadas:

* Load balancing;
* Web servers;
* Service discovery;
* Databases;
* WebSockets;
* Provisioning;

### Load Balancing (Web)

Primeiramente o load balancer, este permite obter uma maior performance do serviço web visto que a carga é distribuída por um ou mais nós. Ao existir mais que um nó eliminamos um ponto único de falha, e com mecanismos de alta disponibilidade (ex. keepalived) um dos nós (previamente em standby) automaticamente entra em ação em caso de falha do outro. Uma vez que é a última camada da infraestrutura pode também assumir um papel da segurança, dado que, permite isolar os serviços que estão para trás desta camada de ataques (ex. prevenção DDOS).

### Web Server

Os nós web possuem um servidor web que servem os pedidos vindos do load balancer.

### Service Discovery

De modo a garantirmos elasticidade foram incluídos dois nós servidores de service discovery. Estes possuem o registo de todos os nós existentes na infraestrutura e dos respetivos serviços, o que é útil para anunciar ao load balancer e ao os nós web disponíveis. Também possui a vantagem de funcionar como servidor dns e remove a necessidade de utilizar endereços de IP fixos.

### Load Balancing (DB)

O pgbouncer assume as funções de distribuição de carga nas bases de dados com a divisão de instruções de escrita e de leitura por diferentes nós. Garante também redundância caso o nó principal ou caso o high-availability proxy, ou os seus dois nós replicas falhem.

### WebSockets

Por fim para garantir redundância no serviço de websockets existem dois nós atrás de um high availability proxy.

### Provisioning

Para realizar operações na infraestrutura existe um nó de provisionamento, a partir do qual é possível realizar manutenção dos nós.

## Solução desenvolvida

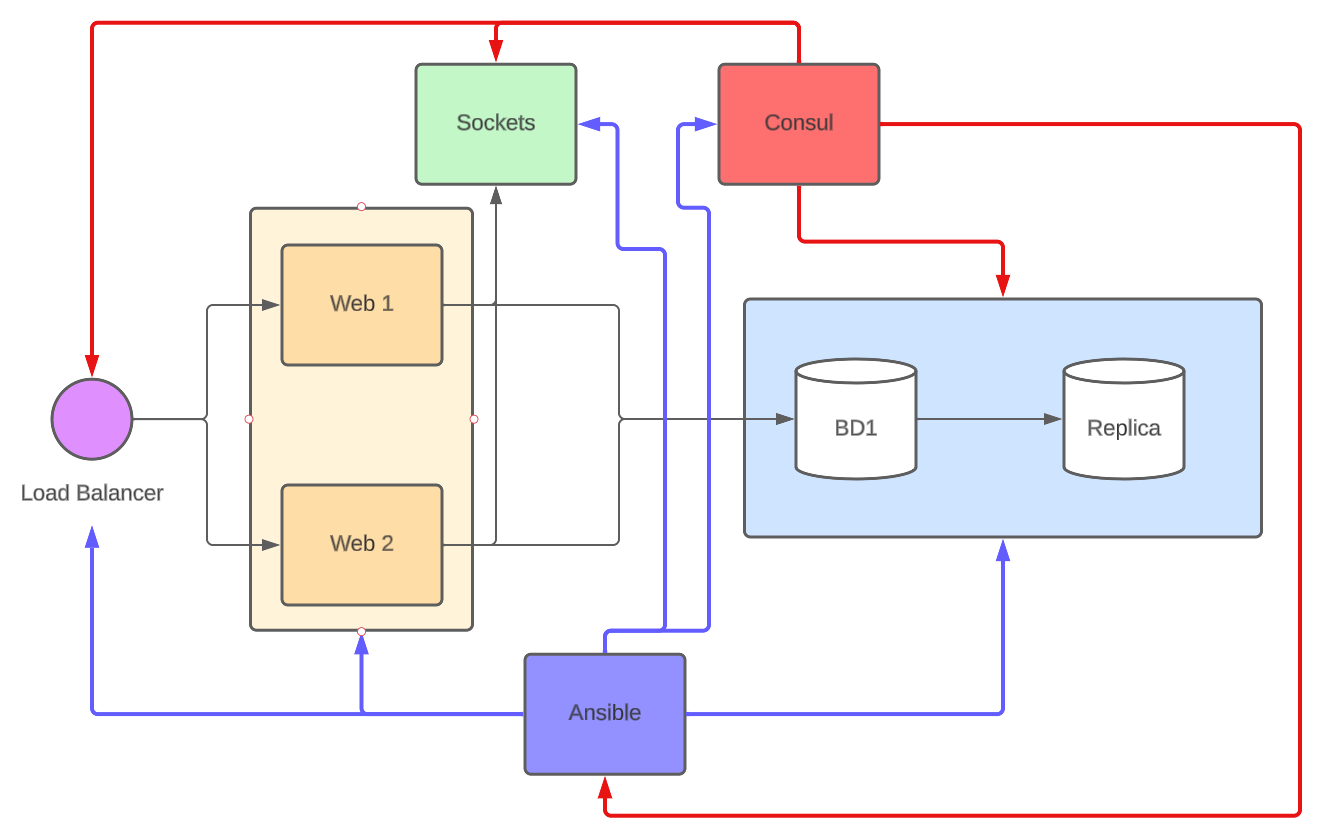


Figura Solução Desenvolvida

Com base na solução pensada anteriormente (Figura 1)

### Resultados

Testando o aumento de performance das chamadas aos servidores web com a ferramenta *hey*, com a qual testámos a velocidade 200 pedidos. Neste ponto considerámos a medida de performance mais relevante a quantidade de pedidos que o servidor consegue responder por segundo e o valor de tempo de resposta médio.

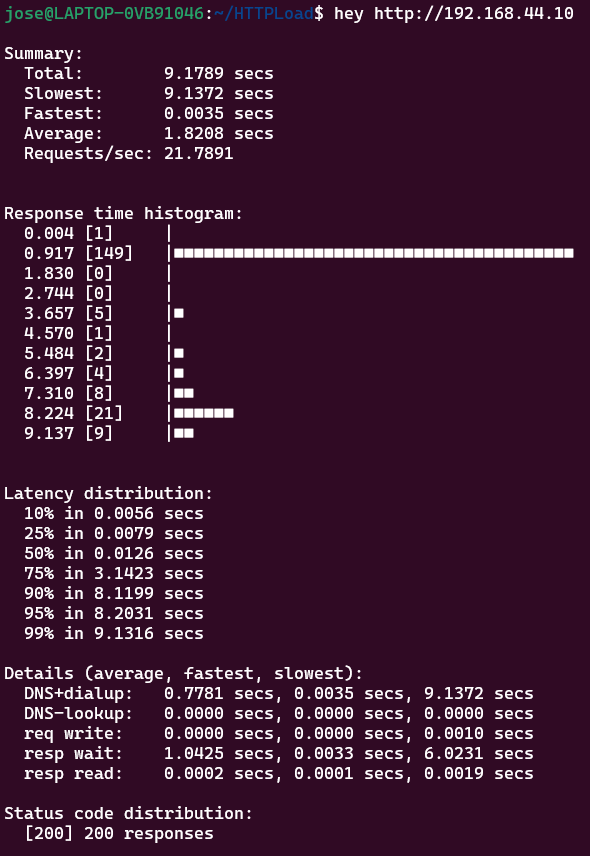


Figura Performance sem load balancing

Primeiro efetuámos as chamadas a um nó web com o ip 192.168.44.10, os resultados obtidos (visível na Figura 4) a velocidade média foi de 1.8 segundos. Devido a uma razão desconhecida 49 pedidos demoraram mais do que 3.6 segundos, por isso o teste foi repetido, desta vez foram efetuados 400 pedidos.

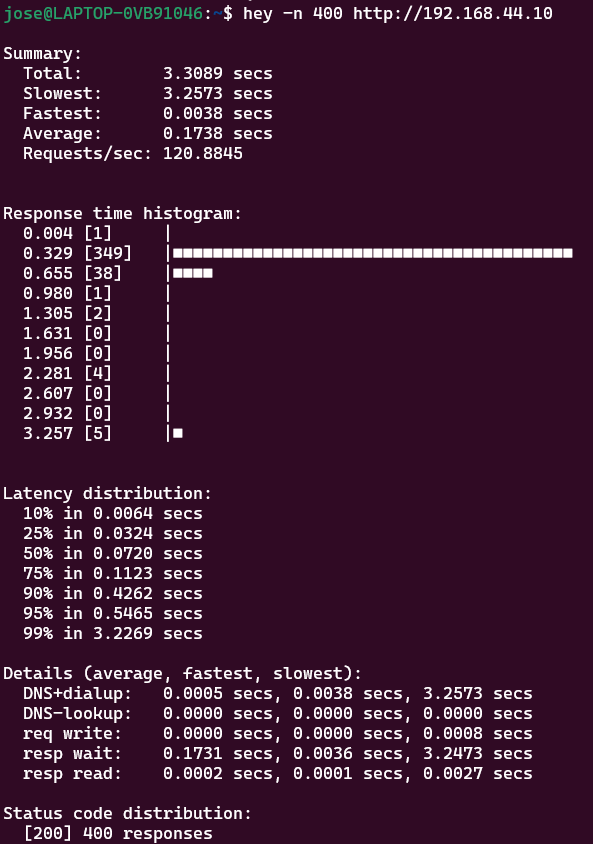


Figura resultados Web server 400 pedidos

Desta vez a discrepância diminuiu e os resultados que o servidor consegue responder por segundo passou para 120 e o tempo médio de resposta diminuiu para 0.17 segundos.

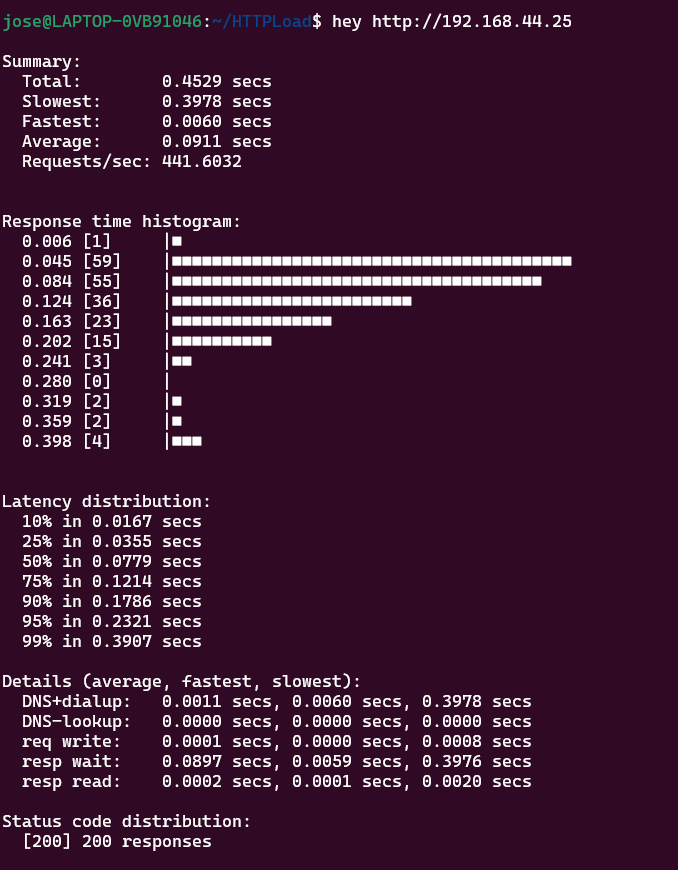


Figura - Performance com LB

Com a adição de mais um nó web e do load balancer (IP: 192.168.44.25) a velocidade média dos pedidos desceu significativamente para 0.09 (0.17 sem load balancer) segundos e o número de pedidos que a infraestrutura aguenta aumentou entre 3 a 4 vezes. Está de acordo com o esperado, uma vez que os pedidos estão a ser distribuídos por 2 servidores web ao invés de 1.

# Componente B