

## Relatório do Trabalho Prático 3

# Replicação de Serviços Web

### Identificação (Grupo 11)

Ana Beatriz Rodrigues Torres - 201920241

Larissa Narciso Oliveira - 201920349

### Descrição Geral da Solução

Neste trabalho usamos o **NGINX**, que é um servidor Web que possui diversos recursos, onde podemos utilizá-lo como um proxy reverso, servidor de cache e um balanceador de carga.

O Balanceador de Carga controla para qual servidor deve ir uma requisição garantindo desempenho. Existem diversos métodos, como: Round Robin, Weighted Round Robin, IP Hash, Least Connection e Weighted Least Connection.

Neste trabalho, realizamos a primeira configuração de balanceamento de carga com o método **Least Connection**, onde o balanceamento de carga distribui a requisição para o servidor com menor carga naquele momento. Caso todos os servidores possuam a mesma quantidade de conexões, ele funciona como o **Round Robin**, onde as requisições são distribuídas de forma uniforme. Utilizamos o **Backup** no servidor 3, onde o NGINX apenas o utilizará caso os outros dois servidores estejam indisponíveis.

Também demonstramos como ficaria com o método **Weighted Least Connection**, onde o peso é aplicado apenas para determinar a proporção de solicitações que cada servidor receberá, mas não afeta a ordem em que as solicitações são encaminhadas para os servidores, dessa forma o método **Least Connection** continua sendo utilizado.

Por último, também mostramos com o método **IP Hash**, onde o servidor para qual a solicitação é enviada é determinado a partir do endereço IP. Com esse método de balanceamento, as solicitações do mesmo cliente sempre serão direcionadas para o mesmo servidor, garantindo que o estado da sessão seja mantido corretamente.

Foi utilizado os arquivos (**server-1.js**, **server-2.js**, **server-3.js**) disponibilizados pelo professor, onde está implementado 3 versões de serviços para identificar onde está sendo encaminhada cada uma das requisições. Sendo necessário instalar o **Node.js**.

## Executando os Servidores

Para rodar os arquivos (*Server-1.js*, *Server-2.js*, *Server-3.js*) deverá instalar o Node.js. Abre 3 terminais na pasta dos arquivos, e executar um comando em cada terminal:

```
node server-1.js
node server-2.js
node server-3.js
```

## Configuração NGINX

Foi instalado o NGINX no Linux, com o seguinte comando:

```
sudo apt install nginx
```

Para iniciar o NGINX, foi digitado o seguinte comando:

```
service nginx start
```

## Configuração Método: Least Connection

Para configurar o balanceamento de carga com o método Least Connection, foi editado o arquivo **nginx.conf** dentro da pasta */etc/nginx*, utilizado o comando:

```
nano etc/nginx/nginx.conf
```

E adicionado as seguintes informações dentro do arquivo:

```
upstream backend_servers {
    least_conn;
    server localhost:3001;
    server localhost:3002;
    server localhost:3003 backup;
}

server {
    listen 80;
    server_name localhost;
    location / {
        proxy_pass http://backend_servers;
    }
}
```

Dessa forma, foi configurado o método **Least Connection** para o balanceamento de carga. E o servidor *localhost:3003* servirá como backup para caso os servidores *localhost:3001* e *localhost:3002* fique indisponível.

## Configuração Método: Weighted Least Connection

Foi editado o arquivo **nginx.conf** dentro da pasta */etc/nginx* e realizado as seguintes alterações:

```
upstream backend_servers {  
    least_conn;  
    server localhost:3001 weight=3;  
    server localhost:3002 weight=2;  
    server localhost:3003;  
}
```

Dessa forma o servidor *Localhost:3001* receberá o triplo de requisições do que o *Localhost:3003*. E o *Localhost:3002* receberá o dobro de requisições que o *Localhost:3003*.

Foi realizado mais uma mudança no arquivo **nginx.conf** para demonstrar mais resultados:

```
upstream backend_servers {  
    least_conn;  
    server localhost:3001 backup;  
    server localhost:3002 weight=2;  
    server localhost:3003;  
}
```

Dessa forma o servidor *Localhost:3001* será utilizado como *backup*, sendo utilizado apenas se os outros dois servidores pararem, e o *Localhost:3002* receberá o dobro de requisições do *Localhost:3003*.

## Configuração Método: IP Hash

Foi editado o arquivo **nginx.conf** dentro da pasta */etc/nginx* e realizado as seguintes alterações:

```
upstream backend_servers {  
    ip_hash;  
    server localhost:3001;  
    server localhost:3002;  
    server localhost:3003;  
}
```

## Requisições

Para realizar as 15 requisições web, foi utilizado o comando **curl**, que irá mandar 15 requisições para os servidores conforme configurado o balanceamento de carga no *localhost:80*.

```
for i in {1..15}; do curl localhost:80; done
```

Para realizar requisições em um loop infinito (até que escreva ctrl+c) foi utilizado o comando:

```
while sleep 0.5; do curl localhost:80; done
```

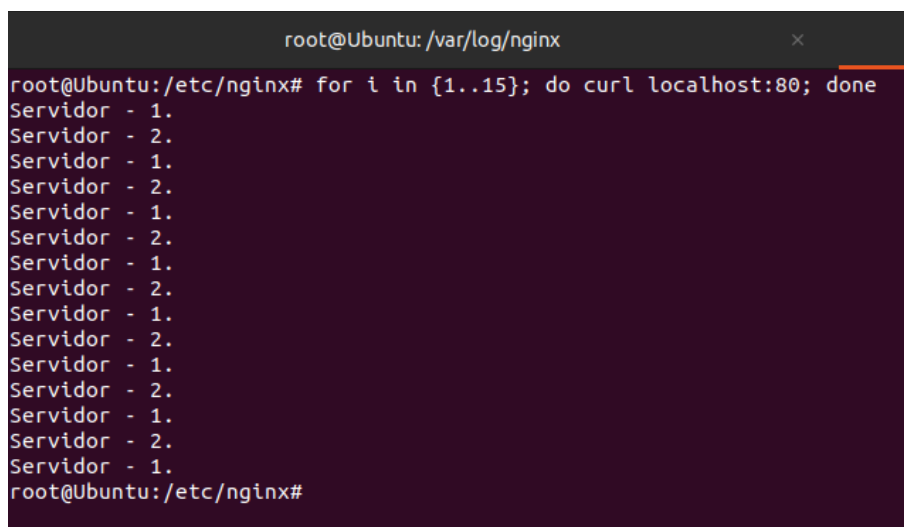
Observamos os resultados pelo arquivo *access.log* da pasta */var/log/nginx/*, onde é registrado todas as solicitações recebidas pelo servidor NGINX.

## Resultados

Será mostrado os resultados utilizando balanceamento **Least Connection**. Em seguida os resultados após realizar a mudança das políticas de balanceamento de carga para **Weighted Least Connection**. Por fim, será mostrado um comparativo entre os dois balanceamentos.

### Resultados Método: Least Connection

Foi iniciado os 3 servidores, conforme mostrado [aqui](#). Depois foi executado o comando para realizar 15 requisições no *localhost:80*, conforme mostrado [aqui](#).



```
root@Ubuntu: /var/log/nginx
root@Ubuntu:/etc/nginx# for i in {1..15}; do curl localhost:80; done
Servidor - 1.
Servidor - 2.
Servidor - 1.
Servidor - 2.
Servidor - 1.
Servidor - 2.
Servidor - 1.
Servidor - 2.
Servidor - 1.
Servidor - 2.
Servidor - 1.
Servidor - 2.
Servidor - 1.
Servidor - 2.
Servidor - 1.
root@Ubuntu:/etc/nginx#
```

Como podemos observar no log abaixo, houve uma distribuição uniforme das requisições entre *localhost:3001* e *localhost:3002*. A distribuição é uniforme pois ambos os servidores possuem a mesma quantidade de conexões ativas, então o método Least Connection age como um *Round Robin*.



Todas elas foram para o servidor 3 (*Backup*)

```
127.0.0.1 - - [21/Feb/2023:22:50:50 -0300] "GET / HTTP/1.1" 200 34 "-" "curl/7.81.0" ms upstream_addr 127.0.0.1:3002, 127.0.0.1:3001, 127.0.0.1:3003
127.0.0.1 - - [21/Feb/2023:22:50:50 -0300] "GET / HTTP/1.1" 200 34 "-" "curl/7.81.0" ms upstream_addr 127.0.0.1:3003
127.0.0.1 - - [21/Feb/2023:22:50:50 -0300] "GET / HTTP/1.1" 200 34 "-" "curl/7.81.0" ms upstream_addr 127.0.0.1:3003
127.0.0.1 - - [21/Feb/2023:22:50:50 -0300] "GET / HTTP/1.1" 200 34 "-" "curl/7.81.0" ms upstream_addr 127.0.0.1:3003
127.0.0.1 - - [21/Feb/2023:22:50:50 -0300] "GET / HTTP/1.1" 200 34 "-" "curl/7.81.0" ms upstream_addr 127.0.0.1:3003
127.0.0.1 - - [21/Feb/2023:22:50:50 -0300] "GET / HTTP/1.1" 200 34 "-" "curl/7.81.0" ms upstream_addr 127.0.0.1:3003
127.0.0.1 - - [21/Feb/2023:22:50:50 -0300] "GET / HTTP/1.1" 200 34 "-" "curl/7.81.0" ms upstream_addr 127.0.0.1:3003
127.0.0.1 - - [21/Feb/2023:22:50:50 -0300] "GET / HTTP/1.1" 200 34 "-" "curl/7.81.0" ms upstream_addr 127.0.0.1:3003
127.0.0.1 - - [21/Feb/2023:22:50:50 -0300] "GET / HTTP/1.1" 200 34 "-" "curl/7.81.0" ms upstream_addr 127.0.0.1:3003
127.0.0.1 - - [21/Feb/2023:22:50:50 -0300] "GET / HTTP/1.1" 200 34 "-" "curl/7.81.0" ms upstream_addr 127.0.0.1:3003
127.0.0.1 - - [21/Feb/2023:22:50:50 -0300] "GET / HTTP/1.1" 200 34 "-" "curl/7.81.0" ms upstream_addr 127.0.0.1:3003
127.0.0.1 - - [21/Feb/2023:22:50:50 -0300] "GET / HTTP/1.1" 200 34 "-" "curl/7.81.0" ms upstream_addr 127.0.0.1:3003
127.0.0.1 - - [21/Feb/2023:22:50:50 -0300] "GET / HTTP/1.1" 200 34 "-" "curl/7.81.0" ms upstream_addr 127.0.0.1:3003
127.0.0.1 - - [21/Feb/2023:22:50:50 -0300] "GET / HTTP/1.1" 200 34 "-" "curl/7.81.0" ms upstream_addr 127.0.0.1:3003
127.0.0.1 - - [21/Feb/2023:22:50:50 -0300] "GET / HTTP/1.1" 200 34 "-" "curl/7.81.0" ms upstream_addr 127.0.0.1:3003
root@Ubuntu:/var/log/nginx#
```

Na primeira requisição, tenta se comunicar com o *Localhost:3001* e *Localhost:3002*, e ao falhar ele manda a requisição para o *Localhost:3003* (backup). Dessa forma, o balanceamento de carga funcionou como o esperado.

## Resultados Método: Weighted Least Connection (1)

Foi realizada a primeira mudança no arquivo *nginx.conf*, conforme mostrado [aqui](#). Foi iniciado os 3 servidores, conforme mostrado [aqui](#).

Foi executado o comando para realizar 15 requisições, conforme mostrado [aqui](#).

```
root@Ubuntu: /var/log/nginx
root@Ubuntu:/etc/nginx# for i in {1..15}; do curl localhost:80; done
Servidor - 1. (Peso 3)
Servidor - 2. (Peso 2)
Servidor - 1. (Peso 3)
Servidor - 3.
Servidor - 2. (Peso 2)
Servidor - 1. (Peso 3)
Servidor - 1. (Peso 3)
Servidor - 2. (Peso 2)
Servidor - 1. (Peso 3)
Servidor - 3.
Servidor - 2. (Peso 2)
Servidor - 1. (Peso 3)
Servidor - 1. (Peso 3)
Servidor - 2. (Peso 2)
Servidor - 1. (Peso 3)
root@Ubuntu:/etc/nginx#
```

Como podemos observar nos logs abaixo:

```
root@Ubuntu: /var/log/nginx x root@Ubuntu: /etc/nginx
root@Ubuntu:/var/log/nginx# cat /var/log/nginx/access.log
127.0.0.1 - - [22/Feb/2023:00:24:54 -0300] "GET / HTTP/1.1" 200 34 "-" "curl/7.81.0" ms upstream_addr 127.0.0.1:3001
127.0.0.1 - - [22/Feb/2023:00:24:54 -0300] "GET / HTTP/1.1" 200 34 "-" "curl/7.81.0" ms upstream_addr 127.0.0.1:3002
127.0.0.1 - - [22/Feb/2023:00:24:54 -0300] "GET / HTTP/1.1" 200 34 "-" "curl/7.81.0" ms upstream_addr 127.0.0.1:3001
127.0.0.1 - - [22/Feb/2023:00:24:54 -0300] "GET / HTTP/1.1" 200 24 "-" "curl/7.81.0" ms upstream_addr 127.0.0.1:3003
127.0.0.1 - - [22/Feb/2023:00:24:54 -0300] "GET / HTTP/1.1" 200 34 "-" "curl/7.81.0" ms upstream_addr 127.0.0.1:3002
127.0.0.1 - - [22/Feb/2023:00:24:54 -0300] "GET / HTTP/1.1" 200 34 "-" "curl/7.81.0" ms upstream_addr 127.0.0.1:3001
127.0.0.1 - - [22/Feb/2023:00:24:54 -0300] "GET / HTTP/1.1" 200 34 "-" "curl/7.81.0" ms upstream_addr 127.0.0.1:3001
127.0.0.1 - - [22/Feb/2023:00:24:54 -0300] "GET / HTTP/1.1" 200 34 "-" "curl/7.81.0" ms upstream_addr 127.0.0.1:3002
127.0.0.1 - - [22/Feb/2023:00:24:54 -0300] "GET / HTTP/1.1" 200 34 "-" "curl/7.81.0" ms upstream_addr 127.0.0.1:3001
127.0.0.1 - - [22/Feb/2023:00:24:54 -0300] "GET / HTTP/1.1" 200 24 "-" "curl/7.81.0" ms upstream_addr 127.0.0.1:3003
127.0.0.1 - - [22/Feb/2023:00:24:54 -0300] "GET / HTTP/1.1" 200 34 "-" "curl/7.81.0" ms upstream_addr 127.0.0.1:3002
127.0.0.1 - - [22/Feb/2023:00:24:54 -0300] "GET / HTTP/1.1" 200 34 "-" "curl/7.81.0" ms upstream_addr 127.0.0.1:3001
127.0.0.1 - - [22/Feb/2023:00:24:54 -0300] "GET / HTTP/1.1" 200 34 "-" "curl/7.81.0" ms upstream_addr 127.0.0.1:3001
127.0.0.1 - - [22/Feb/2023:00:24:54 -0300] "GET / HTTP/1.1" 200 34 "-" "curl/7.81.0" ms upstream_addr 127.0.0.1:3002
127.0.0.1 - - [22/Feb/2023:00:24:54 -0300] "GET / HTTP/1.1" 200 34 "-" "curl/7.81.0" ms upstream_addr 127.0.0.1:3001
127.0.0.1 - - [22/Feb/2023:00:24:54 -0300] "GET / HTTP/1.1" 200 34 "-" "curl/7.81.0" ms upstream_addr 127.0.0.1:3001
127.0.0.1 - - [22/Feb/2023:00:24:54 -0300] "GET / HTTP/1.1" 200 34 "-" "curl/7.81.0" ms upstream_addr 127.0.0.1:3002
127.0.0.1 - - [22/Feb/2023:00:24:54 -0300] "GET / HTTP/1.1" 200 34 "-" "curl/7.81.0" ms upstream_addr 127.0.0.1:3001
root@Ubuntu:/var/log/nginx#
```

*Localhost:3001* recebeu 8 requisições

*Localhost:3002* recebeu 5 requisições

*Localhost:3003* recebeu 2 requisições

Ele funcionou de forma correta, o *localhost:3002* tem aproximadamente o dobro de requisições do *localhost:3003* e o *localhost:3001* tem aproximadamente o triplo de requisições do *localhost:3003*.

Podemos observar que como o *Localhost:3001* estava com uma carga menor que os outros servidores, por isso ele recebeu mais solicitações do que o esperado do peso (apesar de se aproximar). Mas o resultado está dentro do esperado.

## Resultados Método: Weighted Least Connection (2)

Foi realizada a segunda mudança no arquivo *nginx.conf*, conforme mostrado [aqui](#). E executado o comando para realizar 15 requisições no *localhost:80*, conforme mostrado [aqui](#).

```
root@Ubuntu:/etc/nginx# for i in {1..15}; do curl localhost:80; done
Servidor - 2. (Peso 2)
Servidor - 3.
Servidor - 2. (Peso 2)
Servidor - 2. (Peso 2)
Servidor - 3.
Servidor - 2. (Peso 2)
Servidor - 2. (Peso 2)
Servidor - 3.
Servidor - 2. (Peso 2)
Servidor - 2. (Peso 2)
Servidor - 3.
Servidor - 2. (Peso 2)
Servidor - 2. (Peso 2)
Servidor - 3.
Servidor - 2. (Peso 2)
root@Ubuntu:/etc/nginx#
```

Localhost:3003 recebeu 5 requisições

## Resultados Método: IP Hash

[illegible][illegible]



## Least Connection vs Weighted Least Connection vs IP Hash

O Least Connection consegue enviar novas solicitações para servidores com menos cargas.

O Weighted Least Connection é útil quando a capacidade de processamento dos servidores não é a mesma, podendo por um peso maior e receber proporção de solicitações maior.

O IP Hash garante que as solicitações de um cliente específico estejam sempre direcionadas para o mesmo servidor. Como usamos o mesmo computador para fazer as requisições, foram direcionadas para o mesmo servidor.

## Dificuldades

Ambas as integrantes do grupo utilizam o Windows como Sistema Operacional padrão, mas encontramos dificuldades de executar os comandos necessários e realizar as instalações, por isso resolvemos utilizar a máquina virtual para rodar o Linux. Dessa forma, conseguimos realizar todas as instalações necessárias para fazer as configurações de balanceamento de cargas.

As documentações recomendadas pelo professor, bem como os arquivos dos servidores ajudaram no andamento do trabalho. Pela documentação conseguimos entender como realizar a configuração correta do balanceamento de carga, sem ter dificuldades.

Houve dificuldades para conseguir interpretar os resultados pelo log, visto que o *Least Connection* age como *Round Robin* quando os servidores possuem a mesma quantidade de carga, dessa forma, demoramos um pouco para perceber que este era o resultado esperado. Mas em um cenário real essa configuração de balanceamento funcionaria de forma adequada.