# Relatório Preliminar - Balanceamento de carga em Servidores Web com HAProxy e Keepalived

Disponibilidade e Desempenho

2021 - 2022

**Bruno Teixeira**
a2019100036@isec.pt

Licenciatura de Engenharia Informática
22 de dezembro de 2021

# Conteúdo


**1 Introdução** **1**

**2 Desenvolvimento** **2**
2.1 Descrição das tecnologias usadas . . . 2
2.2 Flask, MariaDB e Galera Cluster . . . 2
2.3 HAProxy . . . 2
2.4 KeepAlived . . . 2
2.5 Ambiente para as experiências . . . 3
2.5.1 Aplicação Web . . . 3
2.5.2 Configuração do HAProxy . . . 4
2.5.3 Configuração do KeepAlived . . . 5
2.6 Experiências . . . 6
2.6.1 Experiência 01 . . . 6
2.6.2 Experiência 02 . . . 8

**3 Guião** **13**
3.1 HAProxy e KeepAlived . . . 13
3.2 Aplicação Web . . . 13
3.2.1 Estrutura da aplicação . . . 13
3.2.2 Variáveis de ambiente usadas . . . 14
3.2.3 requirements.txt . . . 14
3.2.4 sync_between_webservers.sh . . . 15
3.2.5 flask_app . . . 15
3.2.6 app . . . 15
3.2.7 static . . . 18
3.2.8 templates . . . 20
3.3 Base de Dados . . . 24
3.3.1 Criação da base de dados . . . 24
3.3.2 Criação das tabelas . . . 24
3.3.3 Inserção de dados nas tabelas . . . 25
3.3.4 Selecionar todos os artigos do carrinho do Cliente 1 . . . 25

# Lista de Figuras


2.1 Index - Aplicação Web . . . 3
2.2 Carrinho - Aplicação Web . . . 4
2.3 Configuração - HAProxy . . . 5
2.4 Estatísticas - HAProxy . . . 5
2.5 Configuração - Keepalived . . . 6
2.6 Esquema - Experiência 01 . . . 7
2.7 Wireshark - Experiência 01 . . . 7
2.8 Single Point of Failure - Experiência 01 . . . 8
2.9 Esquema inicial - Experiência 02 . . . 8
2.10 Esquema com *fail-over* - Experiência 02 . . . 9
2.11 Wireshark no 192.168.1.183 - Experiência 02 . . . 9
2.12 Estado inicial do KeepAlived em ambos os servidores - Experiência 02 . . . 10
2.13 Wireshark no 192.168.1.183 com o HAProxy desligado - Experiência 02 . . . 10
2.14 Estado atual do KeepAlived em ambos os servidores - Experiência 02 . . . 11
2.15 Wireshark no 192.168.1.183 com o HAProxy retomado - Experiência 02 . . . 11
2.16 Estado final do KeepAlived - Experiência 02 . . . 12
2.17 Single Point of Failure - Experiência 02 . . . 12

3.1 Estrutura da aplicação web - Guião . . . 13
3.2 Variáveis de ambiente - Guião . . . 14

# Capítulo 1

# Introdução

Este trabalho tem como objetivo estudar o balanceamento de carga e o *failover* em servidores *web* com o HAProxy e o KeepAlived.
O principal foco é, criar alguns cenários com possíveis falhas, analizar esses cenários, descobrir os pontos fracos e tentar sempre minimizar o *down-time*.

Para alcançar este objetivo foram então criadas algumas experiências de modo a perceber como é possível criar uma infraestrutura segura e com um *down-time* reduzido ou até mesmo nulo aplicando ao mesmo tempo o conceito de balanceamento de carga.

# Capítulo 2

# Desenvolvimento

## 2.1 Descrição das tecnologias usadas

## 2.2 Flask, MariaDB e Galera Cluster

O Flask é um *micro-framework* do *Python* destinado principalmente para pequenas aplicações com requisitos mais simples. O mesmo funciona bastante bem com bases de dados tendo sido este o escolhido para o desenvolvimento da aplicação *web* para este trabalho.
Para que a aplicação fosse *stateful* foi então usada uma base de dados em MariaDB, uma vez que a mesma tem uma comunidade enorme na Internet, tornando-a bastante simples de ser utilizada.

O *Galera Cluster* é um *cluster* virtual para MariaDB que permite a replicação entre diferentes bases de dados, mantendo assim uma disponibilidade e desempenho alto uma vez que existe mais do que uma base de dados para responder a diferentes *querys* dos clientes, sendo que todas as mudanças feitas numa base de dados são replicadas para a outra.

## 2.3 HAProxy

O HAProxy é um serviço Linux que garante o balanceamento de carga para HTTP e TCP. Na prática, o mesmo recebe as conexões dos utilizadores e atua como *proxy*, criando um canal de comunicação entre o utilizador e um dos *webservers*.

O HAProxy funciona em dois modos diferentes, HTTP ou TCP.

- Quando opera em TCP dizemos que é um *Proxy* de camada 4 (OSI)
  - Quando o HAProxy opera neste modo, o mesmo apenas tem acesso a qual IP e Porto o cliente está a tentar aceder, não conseguindo assim ver a informação trocada entre de ambas as partes.
- Quando opera em HTTP dizemos que é um *Proxy* de camada 7 (OSI)
  - Quando o HAProxy opera neste modo, o mesmo tem acesso a toda a informação, logo estamos a confiar no mesmo para ter acesso a esses dados, dados que transitam de um lado para o outro.

## 2.4 KeepAlived

O objetivo principal do KeepAlived é fornecer instalações simples para existir balanceamento de carga e alta disponibilidade (a alta disponibilidade é alcançada pelo protocolo VRRP) para sistemas baseados em Linux. O Keepalived agrega então um conjunto de servidores de balanceamento de carga (HAProxy), e consoante a saúde dos mesmos, ele toma decisões sobre pelo qual deverá passar a operacionalização.

## 2.5 Ambiente para as experiências

Para serem feitas algumas experiências foi criado um ambiente com várias máquinas virtuais, estando estas agregadas a um virtualizador ESXi, ou seja, todas as máquinas estão na mesma LAN.

- **webserver01** - 192.168.1.180
- **webserver02** - 192.168.1.181
- **mariadb01** - 192.168.1.182
- **mariadb02** - 192.168.1.186
- **haproxy01** - 192.168.1.183
- **haproxy02** - 192.168.1.184
- **haproxy03** - 192.168.1.185
- **haproxy04** - 192.168.1.187

### 2.5.1 Aplicação Web

Como descrito anteriormente, foi criada uma aplicação em Flask. Esta aplicação funciona como uma espécie de lista de compras em que o utilizador depois de fazer o *login*, consegue adicionar e eliminar produtos do seu cesto.

Figura 2.1: Index - Aplicação Web

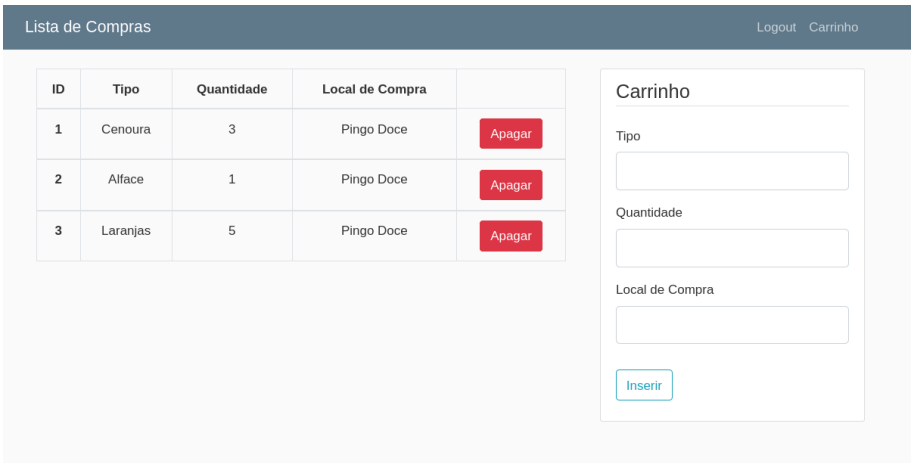



Figura 2.2: Carrinho - Aplicação Web

### 2.5.2 Configuração do HAProxy

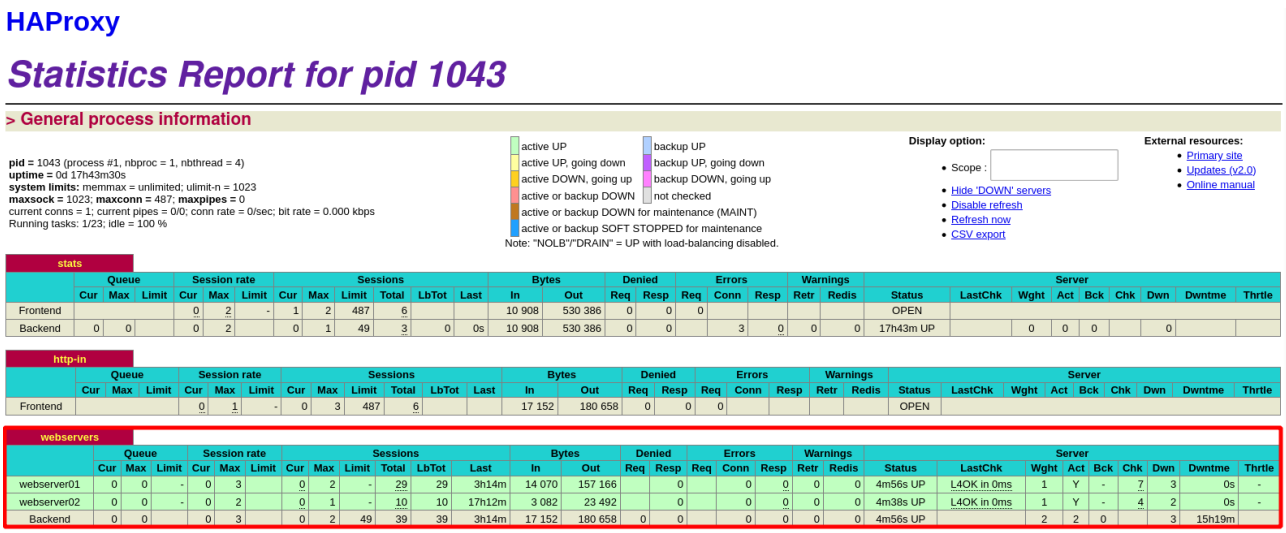
No ficheiro de configuração do HAProxy (localizado em ***/etc/haproxy/haproxy.cfg***) existem 5 secções, sendo que estas definem como é que o servidor se comporta, quais são as definições por omissão, e como é que o cliente faz pedidos e recebe respostas.

- **global**
  - Nesta primeira secção estão definidas as medidas em que o processo vai operar, sendo estas medidas de um nível mais baixo, ou seja, relacionadas com o sistema operativo.
- **defaults**
  - Esta secção não é obrigatória, no entanto permite reduzir a duplicação de comandos, uma vez que as configurações feitas aqui são aplicadas na secção ***frontend*** e ***backend***.
- **listen**
  - Aqui podemos combinar o **frontend** e **backend** ao mesmo tempo. Isto é útil, pois é aqui feito o redirecionamento para o ***endpoint*** de estatisticas.
- **frontend**
  - Nesta secção definimos como é que os pedidos dos utilizadores irão ser encaminhados para o **backend**.
- **backend**
  - Aqui definimos os ***webservers*** que vão operar na infraestrutura, definindo tambem o algoritmo de ***load balancing*** a ser utilizado

```
# O comando "stats socket" ativa a API do HAProxy sendo assim possível gerar um "endpoint" com todas as estatisticas do proxy e dos servidores web
global
        stats socket /run/haproxy/admin.sock mode 660 level admin

# Neste caso usamos o modo "http"
defaults
        mode http

# Configuração referente ao "endpoint" de estatisticas
# Por motivos de simplicidade o "auth" está sem  encriptação
listen stats
       bind 192.168.1.183:9999
       stats enable
       stats hide-version
       stats refresh 10s
       stats uri /stats
       stats auth haproxy:haproxy

# "http-in" é apenas um nome para o frontend
# Qualquer pessoa que se conecte a este servidor irá ser redirecionado para os "webservers" e fazemos isso usando "default_backend <nome do backend>"
frontend http-in
         bind 192.168.1.183:80
         default_backend webservers

# Damos o mesmo nome ao backend que demos no "default_backend"
# No "balance" escolhemos o algoritmo que quisermos utilizador na atribuição de "webservers" aos clientes (round-robin ou leastcoon)
# Fazemos um "check" com um intervalo de 500 milisegundos, se falhar 3 vezes retira esse "webserver" do grupo
backend webservers
        balance roundrobin
        server webserver01 192.168.1.180:5000 check inter 500 fall 3
        server webserver02 192.168.1.181:5000 check inter 500 fall 3
```

Figura 2.3: Configuração - HAProxy

Conforme foi configurado o HAProxy, ao acedermos a ***http://192.168.1.183:9999/stats*** conseguimos visualizar uma página *web* com várias estatísticas sobre os *webservers*.



HAProxy

Statistics Report for pid 1043

> General process information

**pid** = 1043 (process #1, nbproc = 1, nbthread = 4)
**uptime** = 0d 17h43m30s
**system limits:** memmax = unlimited; ulimit-n = 1023
**maxsock** = 1023; **maxconn** = 487; **maxpipes** = 0
current conns = 1; current pipes = 0/0; conn rate = 0/sec; bit rate = 0.000 kbps
Running tasks: 1/23; idle = 100 %

active UP | backup UP
active UP, going down | backup UP, going down
active DOWN, going up | backup DOWN, going up
active or backup DOWN | not checked
active or backup DOWN for maintenance (MAINT)
active or backup SOFT STOPPED for maintenance
Note: "NOLB"/"DRAIN" = UP with load-balancing disabled.

Display option:
- Scope :
- Hide 'DOWN' servers
- Disable refresh
- Refresh now
- CSV export

External resources:
- Primary site
- Updates (v2.0)
- Online manual

stats

| | Queue | | | Session rate | | | Sessions | | | | | | Bytes | | Denied | | Errors | | | Warnings | | Server | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Cur | Max | Limit | Cur | Max | Limit | Cur | Max | Limit | Total | LbTot | Last | In | Out | Req | Resp | Req | Conn | Resp | Retr | Redis | Status | LastChk | Wght | Act | Bck | Chk | Dwn | Dwntme | Thrtle |
| Frontend | | | | 0 | 2 | - | 1 | 2 | 487 | 6 | | | 10 908 | 530 386 | 0 | 0 | 0 | | | | | OPEN | | | | | | | | |
| Backend | 0 | 0 | | 0 | 2 | | 0 | 1 | 49 | 3 | 0 | 0s | 10 908 | 530 386 | 0 | 0 | | 3 | 0 | 0 | 0 | 17h43m UP | | 0 | 0 | 0 | | 0 | | |

http-in

| | Queue | | | Session rate | | | Sessions | | | | | | Bytes | | Denied | | Errors | | | Warnings | | Server | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Cur | Max | Limit | Cur | Max | Limit | Cur | Max | Limit | Total | LbTot | Last | In | Out | Req | Resp | Req | Conn | Resp | Retr | Redis | Status | LastChk | Wght | Act | Bck | Chk | Dwn | Dwntme | Thrtle |
| Frontend | | | | 0 | 1 | - | 0 | 3 | 487 | 6 | | | 17 152 | 180 658 | 0 | 0 | 0 | | | | | OPEN | | | | | | | | |

webservers

| | Queue | | | Session rate | | | Sessions | | | | | | Bytes | | Denied | | Errors | | | Warnings | | Server | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Cur | Max | Limit | Cur | Max | Limit | Cur | Max | Limit | Total | LbTot | Last | In | Out | Req | Resp | Req | Conn | Resp | Retr | Redis | Status | LastChk | Wght | Act | Bck | Chk | Dwn | Dwntme | Thrtle |
| webserver01 | 0 | 0 | - | 0 | 3 | | 0 | 2 | - | 29 | 29 | 3h14m | 14 070 | 157 166 | | 0 | | 0 | 0 | 0 | 0 | 4m56s UP | L4OK in 0ms | 1 | Y | - | 7 | 3 | 0s | - |
| webserver02 | 0 | 0 | - | 0 | 2 | | 0 | 1 | - | 10 | 10 | 17h12m | 3 082 | 23 492 | | 0 | | 0 | 0 | 0 | 0 | 4m38s UP | L4OK in 0ms | 1 | Y | - | 4 | 2 | 0s | - |
| Backend | 0 | 0 | | 0 | 3 | | 0 | 2 | 49 | 39 | 39 | 3h14m | 17 152 | 180 658 | 0 | 0 | | 0 | 0 | 0 | 0 | 4m56s UP | | 2 | 2 | 0 | | 3 | 15h19m | |

Figura 2.4: Estatísticas - HAProxy

### 2.5.3 Configuração do KeepAlived

No ficheiro de configuração do Keepalived (localizado em ***/etc/keepalived/keepalived.conf***), foi criado um **vrrp_script** com o intuito de verificar, com um intervalo de 2 em 2 milisegundos, se o haproxy está a funcionar corretamente. Se o mesmo não estiver a funcionar, o peso dele diminui em 10 reduzindo assim a sua prioridade tornando o BACKUP num MASTER.
Depois disto, foi criado um **vrrp_instance** que define uma instância individual do protocolo VRRP com alguns atributos.

```
# Define definições globais do processo
global_defs{
        enable_script_security
        script_user root
}

# Script que verifica o estado do HAProxy
vrrp_script check_haproxy {
            script "service haproxy status"
            interval 1
            weight -10
}

# Instancia que executa o script, nomeando o HAProxy01 (192.168.1.183) como MASTER
# Escolhemos a interface que queremos usar
# O estado inicial
# Identificação do virtual_router_id sendo que esta tem de ser a mesma no BACKUP
# Prioridade do servidor, esta será anunciada no grupo VRRP
# IP virtual do grupo VRRP
vrrp_instance V1_1{
               interface ens160
               state MASTER
               virtual_router_id 11
               priority 101
               authentication {
                 auth_type PASS
                 auth_pass algumacoisamuitocomplicada
               }
               virtual_ipaddress {
                 192.168.1.200
               }
               track_script {
                 check_haproxy
               }
}
```

Figura 2.5: Configuração - Keepalived

O mesmo foi feito para o segundo servidor de HAProxy, no entanto foi alterada a prioridade e o estado para definir que este seria o BACKUP.

Foi tambem necessário alterar o Ip, para onde faziamos ***bind*** inicialmente (**Ip do servidor HAProxy**), para o novo IP virtual (**192.168.1.200**) na secção de ***frontend*** do ficheiro de configuração do HAProxy.

Por fim foi preciso colocar ***net.ipv4.ip_nonlocal_bind=1*** no ficheiro ***/etc/sysctl.conf*** uma vez que no segundo servidor de HAProxy o IP virtual ainda não está ativo (só fica ativo quando esse for o MASTER), logo não é possível iniciar o *bind*.

Esta configuração do KeepAlived apenas foi feita a partir da segunda experiência, uma vez que na primeira ainda não é usado o mesmo para mostrar os riscos que isso tem.

## 2.6 Experiências

### 2.6.1 Experiência 01

Nesta experiência foi usado um servidor de balanceamento de carga (HAProxy), dois *webservers* e uma base de dados externa (MariaDB). Foi feita uma captura no HAProxy para perceber o que acontecia quando o utilizador fazia um pedido ao mesmo.

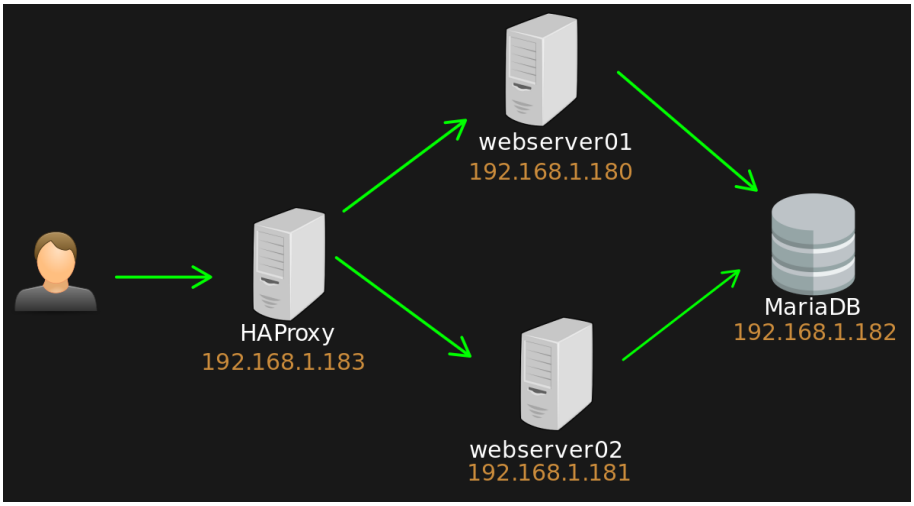
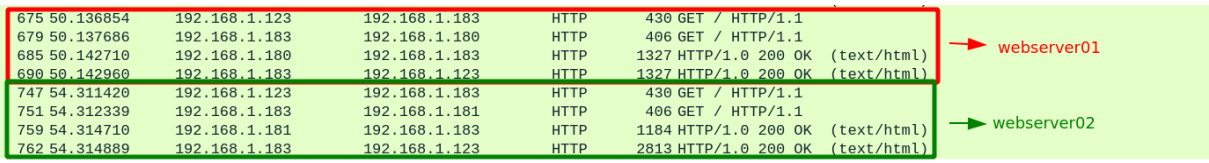



Figura 2.6: Esquema - Experiência 01

### Resultado

Conseguiu-se perceber que o cliente (**192.168.1.123**) envia um *HTTP GET Request* diretamente ao servidor HAProxy (**192.168.1.183**). De seguida, o servidor HAProxy faz um *HTTP GET Request* ao *webserver* disponível, que neste caso foi o *webserver01* (**192.168.1.180**). O *webserver01* responde com o *HTTP status code 200*, mostrando que a comunicação ocorreu sem falhas, sendo feito depois um redirecionamento do HAProxy para o cliente. Imediatamente a seguir foi feito outro pedido pelo mesmo cliente, no entanto consegue-se perceber que, como está a ser usar o algoritmo **round-robin**, quem respondeu foi o *webserver02*.



| | | | | | | |
|---|---|---|---|---|---|---|
| 675 | 50.136854 | 192.168.1.123 | 192.168.1.183 | HTTP | 430 | GET / HTTP/1.1 |
| 679 | 50.137686 | 192.168.1.183 | 192.168.1.180 | HTTP | 406 | GET / HTTP/1.1 |
| 685 | 50.142710 | 192.168.1.180 | 192.168.1.183 | HTTP | 1327 | HTTP/1.0 200 OK (text/html) |
| 690 | 50.142960 | 192.168.1.183 | 192.168.1.123 | HTTP | 1327 | HTTP/1.0 200 OK (text/html) |
| 747 | 54.311420 | 192.168.1.123 | 192.168.1.183 | HTTP | 430 | GET / HTTP/1.1 |
| 751 | 54.312339 | 192.168.1.183 | 192.168.1.181 | HTTP | 406 | GET / HTTP/1.1 |
| 759 | 54.314710 | 192.168.1.181 | 192.168.1.183 | HTTP | 1184 | HTTP/1.0 200 OK (text/html) |
| 762 | 54.314889 | 192.168.1.183 | 192.168.1.123 | HTTP | 2813 | HTTP/1.0 200 OK (text/html) |



Figura 2.7: Wireshark - Experiência 01

### Problemas encontrados na Experiência 01

É percetivel que a experiência feita anteriormente tem alguns problemas, como a existência de dois SPOFs (**Single Point of Failure**).

- Existe um SPOF no HAProxy.
- Existe um SPOF na Base de Dados.

Ou seja, caso o servidor de HAProxy ou a base de dados deixe de operar, o cliente deixa de ter comunicação com os *webservers*. Sabendo isto continuou-se com mais algumas experiências de modo a resolver estes problemas.

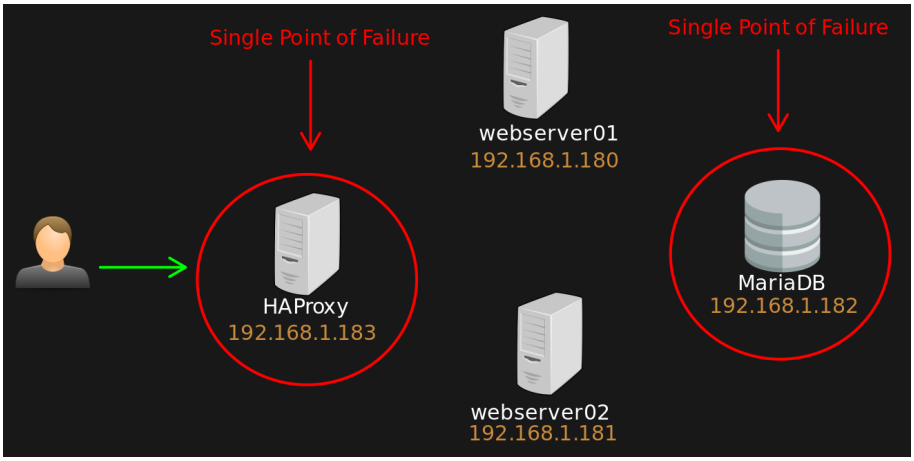



Figura 2.8: Single Point of Failure - Experiência 01

### 2.6.2 Experiência 02

Nesta segunda experiência apenas foi acrescentado mais um servidor de balanceamento de carga e o serviço de **KeepAlived** em ambos os servidores de HAProxy.
O objetivo da mesma era entender como seria feito o *fail-over* do **KeepAlived** e o que sucedia depois de um servidor *MASTER* tornar-se *BACKUP* e vice-versa.

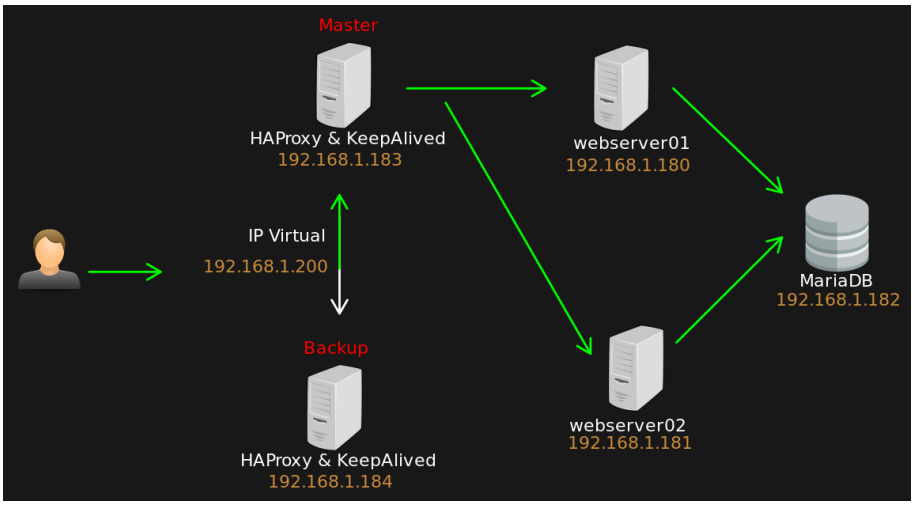



Figura 2.9: Esquema inicial - Experiência 02

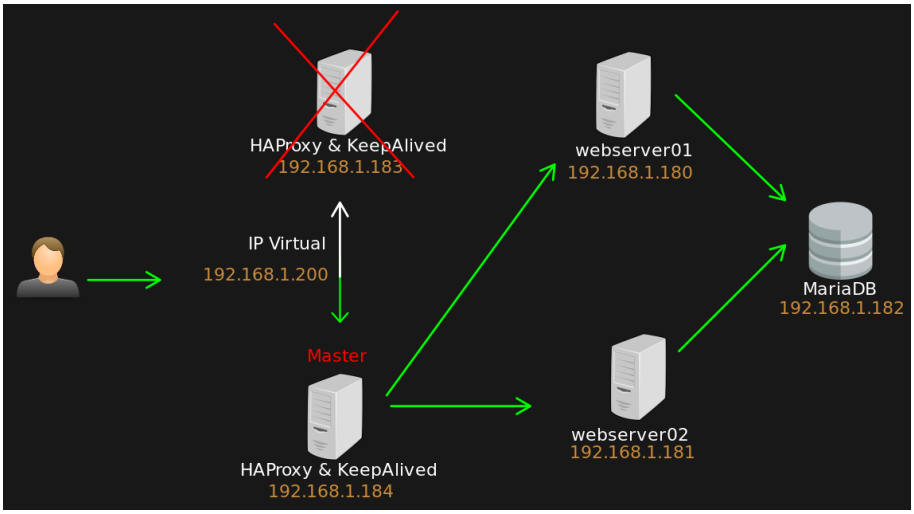



Figura 2.10: Esquema com *fail-over* - Experiência 02

**Resultado**

Foram feitas várias capturas, tanto no **HAProxy01(MASTER)** como no **HAProxy02(BACKUP)** usando o *wireshark*.
Com a captura no **HAProxy01(192.168.1.183)** percebe-se que o mesmo emite, de segundo em segundo, um *announcement* dizendo a sua prioridade, que neste caso é 101. Isto acontece porque no protocolo VRRP apenas o *MASTER* emite mensagens estando os outros *BACKUPs* à escuta desse aviso.

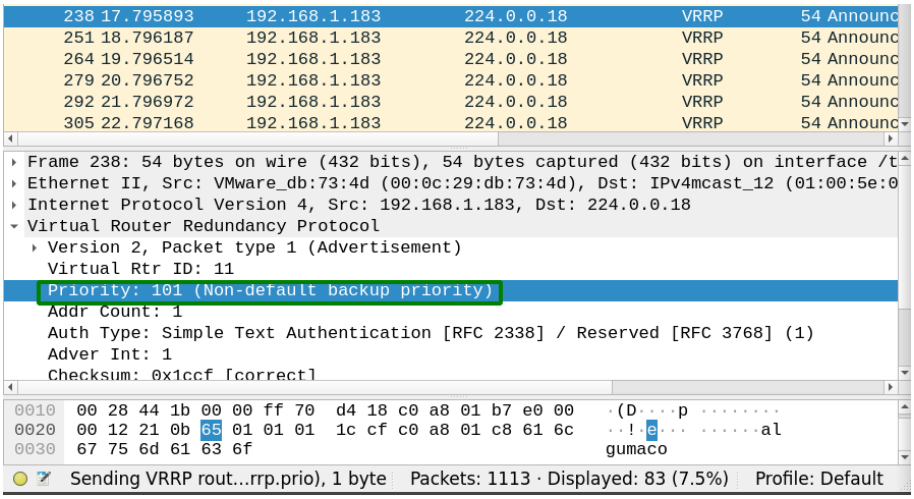



Figura 2.11: Wireshark no 192.168.1.183 - Experiência 02

```
brun0@haproxy01:~$ sudo service keepalived status
● keepalived.service - Keepalive Daemon (LVS and VRRP)
     Loaded: loaded (/lib/systemd/system/keepalived.service; enabled; vendor preset: enabled)
     Active: active (running) since Wed 2021-11-24 12:34:10 UTC; 5h 0min ago
   Main PID: 178585 (keepalived)
      Tasks: 2 (limit: 1070)
     Memory: 3.7M
     CGroup: /system.slice/keepalived.service
             ├─178585 /usr/sbin/keepalived --dont-fork
             └─178594 /usr/sbin/keepalived --dont-fork

Nov 24 17:24:52 haproxy01 Keepalived_vrrp[178594]: (V1_1) Changing effective priority from 101 to 91
Nov 24 17:24:54 haproxy01 Keepalived_vrrp[178594]: (V1_1) Master received advert from 192.168.1.184 with higher priority 100, ours 91
Nov 24 17:24:54 haproxy01 Keepalived_vrrp[178594]: (V1_1) Entering BACKUP STATE
Nov 24 17:29:51 haproxy01 Keepalived_vrrp[178594]: Script `check_haproxy` now returning 0
Nov 24 17:29:51 haproxy01 Keepalived_vrrp[178594]: VRRP_Script(check_haproxy) succeeded
Nov 24 17:29:51 haproxy01 Keepalived_vrrp[178594]: (V1_1) Changing effective priority from 91 to 101
Nov 24 17:29:52 haproxy01 Keepalived_vrrp[178594]: (V1_1) received lower priority (100) advert from 192.168.1.184 - discarding
Nov 24 17:29:53 haproxy01 Keepalived_vrrp[178594]: (V1_1) received lower priority (100) advert from 192.168.1.184 - discarding
Nov 24 17:29:54 haproxy01 Keepalived_vrrp[178594]: (V1_1) received lower priority (100) advert from 192.168.1.184 - discarding
Nov 24 17:29:54 haproxy01 Keepalived_vrrp[178594]: (V1_1) Entering MASTER STATE
brun0@haproxy01:~$

brun0@haproxy02: ~ 134x24
brun0@haproxy02:~$ sudo service keepalived status
● keepalived.service - Keepalive Daemon (LVS and VRRP)
     Loaded: loaded (/lib/systemd/system/keepalived.service; enabled; vendor preset: enabled)
     Active: active (running) since Wed 2021-11-24 13:30:54 UTC; 4h 3min ago
   Main PID: 847 (keepalived)
      Tasks: 2 (limit: 1070)
     Memory: 20.8M
     CGroup: /system.slice/keepalived.service
             ├─847 /usr/sbin/keepalived --dont-fork
             └─898 /usr/sbin/keepalived --dont-fork

Nov 24 17:19:38 haproxy02 Keepalived_vrrp[898]: (V1_1) received lower priority (91) advert from 192.168.1.183 - discarding
Nov 24 17:19:38 haproxy02 Keepalived_vrrp[898]: (V1_1) Entering MASTER STATE
Nov 24 17:20:25 haproxy02 Keepalived_vrrp[898]: (V1_1) Master received advert from 192.168.1.183 with higher priority 101, ours 100
Nov 24 17:20:25 haproxy02 Keepalived_vrrp[898]: (V1_1) Entering BACKUP STATE
Nov 24 17:24:52 haproxy02 Keepalived_vrrp[898]: (V1_1) received lower priority (91) advert from 192.168.1.183 - discarding
Nov 24 17:24:53 haproxy02 Keepalived_vrrp[898]: (V1_1) received lower priority (91) advert from 192.168.1.183 - discarding
Nov 24 17:24:54 haproxy02 Keepalived_vrrp[898]: (V1_1) received lower priority (91) advert from 192.168.1.183 - discarding
Nov 24 17:24:54 haproxy02 Keepalived_vrrp[898]: (V1_1) Entering MASTER STATE
Nov 24 17:29:54 haproxy02 Keepalived_vrrp[898]: (V1_1) Master received advert from 192.168.1.183 with higher priority 101, ours 100
Nov 24 17:29:54 haproxy02 Keepalived_vrrp[898]: (V1_1) Entering BACKUP STATE
brun0@haproxy02:~$
```

Figura 2.12: Estado inicial do KeepAlived em ambos os servidores - Experiência 02

Depois de desligar o serviço HAProxy do **HAProxy01(192.168.1.183)**, o mesmo fica com uma prioridade de 91 passando assim para o estado de *BACKUP* ao mesmo tempo que o **HAProxy02(192.168.1.184)** passa para o estado de *MASTER* uma vez que a sua prioridade é superior (100).

```
4179 312.859733   192.168.1.183   224.0.0.18   VRRP   54 Announc
4181 313.469517   192.168.1.184   224.0.0.18   VRRP   60 Announc
4186 314.469748   192.168.1.184   224.0.0.18   VRRP   60 Announc
4188 315.470003   192.168.1.184   224.0.0.18   VRRP   60 Announc
4189 316.470302   192.168.1.184   224.0.0.18   VRRP   60 Announc
4190 317.470524   192.168.1.184   224.0.0.18   VRRP   60 Announc
4196 318.470675   192.168.1.184   224.0.0.18   VRRP   60 Announc

▸ Frame 4179: 54 bytes on wire (432 bits), 54 bytes captured (432 bits) on interface /t
▸ Ethernet II, Src: VMware_db:73:4d (00:0c:29:db:73:4d), Dst: IPv4mcast_12 (01:00:5e:00
▸ Internet Protocol Version 4, Src: 192.168.1.183, Dst: 224.0.0.18
▾ Virtual Router Redundancy Protocol
  ▸ Version 2, Packet type 1 (Advertisement)
    Virtual Rtr ID: 11
    Priority: 91 (Non-default backup priority)
    Addr Count: 1
    Auth Type: Simple Text Authentication [RFC 2338] / Reserved [RFC 3768] (1)
    Adver Int: 1
    Checksum: 0x26cf [correct]
    [Checksum Status: Good]
    IP Address: 192.168.1.200
    Authentication String: algumaco
```

Figura 2.13: Wireshark no 192.168.1.183 com o HAProxy desligado - Experiência 02

```
brun0@haproxy01:~$ sudo service keepalived status
● keepalived.service - Keepalive Daemon (LVS and VRRP)
     Loaded: loaded (/lib/systemd/system/keepalived.service; enabled; vendor preset: enabled)
     Active: active (running) since Wed 2021-11-24 12:34:10 UTC; 5h 1min ago
   Main PID: 178585 (keepalived)
      Tasks: 2 (limit: 1070)
     Memory: 3.9M
     CGroup: /system.slice/keepalived.service
             ├─178585 /usr/sbin/keepalived --dont-fork
             └─178594 /usr/sbin/keepalived --dont-fork

Nov 24 17:29:51 haproxy01 Keepalived_vrrp[178594]: (V1_1) Changing effective priority from 91 to 101
Nov 24 17:29:52 haproxy01 Keepalived_vrrp[178594]: (V1_1) received lower priority (100) advert from 192.168.1.184 - discarding
Nov 24 17:29:53 haproxy01 Keepalived_vrrp[178594]: (V1_1) received lower priority (100) advert from 192.168.1.184 - discarding
Nov 24 17:29:54 haproxy01 Keepalived_vrrp[178594]: (V1_1) received lower priority (100) advert from 192.168.1.184 - discarding
Nov 24 17:29:54 haproxy01 Keepalived_vrrp[178594]: (V1_1) Entering MASTER STATE
Nov 24 17:35:54 haproxy01 Keepalived_vrrp[178594]: Script `check_haproxy` now returning 3
Nov 24 17:35:54 haproxy01 Keepalived_vrrp[178594]: VRRP_Script(check_haproxy) failed (exited with status 3)
Nov 24 17:35:54 haproxy01 Keepalived_vrrp[178594]: (V1_1) Changing effective priority from 101 to 91
Nov 24 17:35:57 haproxy01 Keepalived_vrrp[178594]: (V1_1) Master received advert from 192.168.1.184 with higher priority 100, ours 91
Nov 24 17:35:57 haproxy01 Keepalived_vrrp[178594]: (V1_1) Entering BACKUP STATE
brun0@haproxy01:~$

brun0@haproxy02: ~ 134x24
brun0@haproxy02:~$ sudo service keepalived status
● keepalived.service - Keepalive Daemon (LVS and VRRP)
     Loaded: loaded (/lib/systemd/system/keepalived.service; enabled; vendor preset: enabled)
     Active: active (running) since Wed 2021-11-24 13:30:54 UTC; 4h 5min ago
   Main PID: 847 (keepalived)
      Tasks: 2 (limit: 1070)
     Memory: 20.9M
     CGroup: /system.slice/keepalived.service
             ├─847 /usr/sbin/keepalived --dont-fork
             └─898 /usr/sbin/keepalived --dont-fork

Nov 24 17:24:52 haproxy02 Keepalived_vrrp[898]: (V1_1) received lower priority (91) advert from 192.168.1.183 - discarding
Nov 24 17:24:53 haproxy02 Keepalived_vrrp[898]: (V1_1) received lower priority (91) advert from 192.168.1.183 - discarding
Nov 24 17:24:54 haproxy02 Keepalived_vrrp[898]: (V1_1) received lower priority (91) advert from 192.168.1.183 - discarding
Nov 24 17:24:54 haproxy02 Keepalived_vrrp[898]: (V1_1) Entering MASTER STATE
Nov 24 17:29:54 haproxy02 Keepalived_vrrp[898]: (V1_1) Master received advert from 192.168.1.183 with higher priority 101, ours 100
Nov 24 17:29:54 haproxy02 Keepalived_vrrp[898]: (V1_1) Entering BACKUP STATE
Nov 24 17:35:54 haproxy02 Keepalived_vrrp[898]: (V1_1) received lower priority (91) advert from 192.168.1.183 - discarding
Nov 24 17:35:55 haproxy02 Keepalived_vrrp[898]: (V1_1) received lower priority (91) advert from 192.168.1.183 - discarding
Nov 24 17:35:56 haproxy02 Keepalived_vrrp[898]: (V1_1) received lower priority (91) advert from 192.168.1.183 - discarding
Nov 24 17:35:57 haproxy02 Keepalived_vrrp[898]: (V1_1) Entering MASTER STATE
brun0@haproxy02:~$
```

Figura 2.14: Estado atual do KeepAlived em ambos os servidores - Experiência 02

Para terminar, voltou-se a ativar o serviço haproxy no **HAProxy01(192.168.1.183)** tornando-se assim novamente *MASTER* uma vez que a preempção está ativa por omissão fazendo com que a sua prioridade volte a ser 101 como estava definida inicialmente.

```
4445 469.111033   192.168.1.183   224.0.0.18   VRRP   54 Announc
4458 470.111248   192.168.1.183   224.0.0.18   VRRP   54 Announc
4475 471.111472   192.168.1.183   224.0.0.18   VRRP   54 Announc
4488 472.111647   192.168.1.183   224.0.0.18   VRRP   54 Announc
4501 473.111860   192.168.1.183   224.0.0.18   VRRP   54 Announc
4520 474.112107   192.168.1.183   224.0.0.18   VRRP   54 Announc

▸ Frame 4445: 54 bytes on wire (432 bits), 54 bytes captured (432 bits) on interface /tr
▸ Ethernet II, Src: VMware_db:73:4d (00:0c:29:db:73:4d), Dst: IPv4mcast_12 (01:00:5e:00
▸ Internet Protocol Version 4, Src: 192.168.1.183, Dst: 224.0.0.18
▾ Virtual Router Redundancy Protocol
  ▸ Version 2, Packet type 1 (Advertisement)
    Virtual Rtr ID: 11
    Priority: 101 (Non-default backup priority)
    Addr Count: 1
    Auth Type: Simple Text Authentication [RFC 2338] / Reserved [RFC 3768] (1)
    Adver Int: 1
    Checksum: 0x1ccf [correct]
    [Checksum Status: Good]
    IP Address: 192.168.1.200
    Authentication String: algumaco
```

Figura 2.15: Wireshark no 192.168.1.183 com o HAProxy retomado - Experiência 02

```
brun0@haproxy01:~$ sudo service keepalived status
● keepalived.service - Keepalive Daemon (LVS and VRRP)
     Loaded: loaded (/lib/systemd/system/keepalived.service; enabled; vendor preset: enabled)
     Active: active (running) since Wed 2021-11-24 12:34:10 UTC; 5h 4min ago
   Main PID: 178585 (keepalived)
      Tasks: 2 (limit: 1070)
     Memory: 3.7M
     CGroup: /system.slice/keepalived.service
             ├─178585 /usr/sbin/keepalived --dont-fork
             └─178594 /usr/sbin/keepalived --dont-fork

Nov 24 17:35:54 haproxy01 Keepalived_vrrp[178594]: (V1_1) Changing effective priority from 101 to 91
Nov 24 17:35:57 haproxy01 Keepalived_vrrp[178594]: (V1_1) Master received advert from 192.168.1.184 with higher priority 100, ours 91
Nov 24 17:35:57 haproxy01 Keepalived_vrrp[178594]: (V1_1) Entering BACKUP STATE
Nov 24 17:38:30 haproxy01 Keepalived_vrrp[178594]: Script `check_haproxy` now returning 0
Nov 24 17:38:30 haproxy01 Keepalived_vrrp[178594]: VRRP_Script(check_haproxy) succeeded
Nov 24 17:38:30 haproxy01 Keepalived_vrrp[178594]: (V1_1) Changing effective priority from 91 to 101
Nov 24 17:38:30 haproxy01 Keepalived_vrrp[178594]: (V1_1) received lower priority (100) advert from 192.168.1.184 - discarding
Nov 24 17:38:31 haproxy01 Keepalived_vrrp[178594]: (V1_1) received lower priority (100) advert from 192.168.1.184 - discarding
Nov 24 17:38:32 haproxy01 Keepalived_vrrp[178594]: (V1_1) received lower priority (100) advert from 192.168.1.184 - discarding
Nov 24 17:38:32 haproxy01 Keepalived_vrrp[178594]: (V1_1) Entering MASTER STATE
brun0@haproxy01:~$

brun0@haproxy02: ~ 134x24
brun0@haproxy02:~$ sudo service keepalived status
● keepalived.service - Keepalive Daemon (LVS and VRRP)
     Loaded: loaded (/lib/systemd/system/keepalived.service; enabled; vendor preset: enabled)
     Active: active (running) since Wed 2021-11-24 13:30:54 UTC; 4h 7min ago
   Main PID: 847 (keepalived)
      Tasks: 2 (limit: 1070)
     Memory: 20.8M
     CGroup: /system.slice/keepalived.service
             ├─847 /usr/sbin/keepalived --dont-fork
             └─898 /usr/sbin/keepalived --dont-fork

Nov 24 17:24:54 haproxy02 Keepalived_vrrp[898]: (V1_1) received lower priority (91) advert from 192.168.1.183 - discarding
Nov 24 17:24:54 haproxy02 Keepalived_vrrp[898]: (V1_1) Entering MASTER STATE
Nov 24 17:29:54 haproxy02 Keepalived_vrrp[898]: (V1_1) Master received advert from 192.168.1.183 with higher priority 101, ours 100
Nov 24 17:29:54 haproxy02 Keepalived_vrrp[898]: (V1_1) Entering BACKUP STATE
Nov 24 17:35:54 haproxy02 Keepalived_vrrp[898]: (V1_1) received lower priority (91) advert from 192.168.1.183 - discarding
Nov 24 17:35:55 haproxy02 Keepalived_vrrp[898]: (V1_1) received lower priority (91) advert from 192.168.1.183 - discarding
Nov 24 17:35:56 haproxy02 Keepalived_vrrp[898]: (V1_1) received lower priority (91) advert from 192.168.1.183 - discarding
Nov 24 17:35:57 haproxy02 Keepalived_vrrp[898]: (V1_1) Entering MASTER STATE
Nov 24 17:38:32 haproxy02 Keepalived_vrrp[898]: (V1_1) Master received advert from 192.168.1.183 with higher priority 101, ours 100
Nov 24 17:38:32 haproxy02 Keepalived_vrrp[898]: (V1_1) Entering BACKUP STATE
brun0@haproxy02:~$
```

Figura 2.16: Estado final do KeepAlived - Experiência 02

### Problemas encontrados na Experiência 02

Com esta nova arquitetura, foi possível resolver um SPOF**(Single Point of Failure)** colocando mais um servidor de balanceamento de carga e acrescentando o serviço de **KeepAlived**, no entanto é notório que continua a existir um SPOF na base de dados.

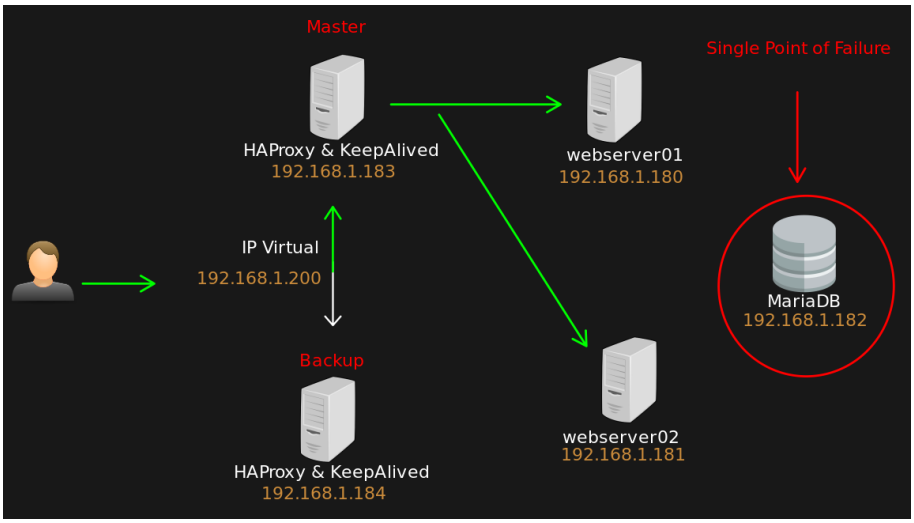



Figura 2.17: Single Point of Failure - Experiência 02

# Capítulo 3

# Guião

Em baixo estão descritos os passos para que seja possível criar todo o código da aplicação *web* assim como a base de dados de modo a conseguir-se replicar as experiências que foram mostradas anteriormente.

## 3.1 HAProxy e KeepAlived

Para instalar o HAProxy, bastou fazer *sudo apt install haproxy* em ambos os servidores de HAProxy. Depois foi feita a configuração do ficheiro do HAProxy (*/etc/haproxy/haproxy.cfg*) como descrito na secção de **Configuração do HAProxy**.

Para instalar o KeepAlived, bastou fazer *sudo apt install keepalived* em ambos os servidores de HAProxy. A sua configuração está tambem descrita na secção **Configuração do Keepalived**.

## 3.2 Aplicação Web

### 3.2.1 Estrutura da aplicação

```
.
└── repo
    ├── LICENSE
    ├── README.md
    ├── flask_app
    │   ├── app
    │   │   ├── __init__.py
    │   │   ├── forms.py
    │   │   ├── models.py
    │   │   ├── routes.py
    │   │   ├── static
    │   │   │   ├── cesto.jpg
    │   │   │   └── main.css
    │   │   └── templates
    │   │       ├── carrinho.html
    │   │       ├── index.html
    │   │       ├── layout.html
    │   │       └── login.html
    │   ├── carrinho.py
    │   └── config.py
    ├── requeriments.txt
    └── sync_between_webservers.sh
```

Figura 3.1: Estrutura da aplicação web - Guião

### 3.2.2 Variáveis de ambiente usadas

```
brun0@webserver01:~$ cat /etc/environment
PATH="/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin"

FLASK_APP=carrinho.py
FLASK_DEBUG=true
RSYNC_PASSWORD="toor"
DB_HOST="mysql+pymysql://brun0:toor@192.168.1.185:3306/carrinho"
DB_USER="brun0"
DB_PW="toor"
DB_NAME="carrinho"
```

Figura 3.2: Variáveis de ambiente - Guião

### 3.2.3 requirements.txt

Para esta aplicação Web são precisas algumas dependências, dependências estas descritas no ficheiro *requeriments.txt*. A maneira mais simples de instalar todas as dependências é criar um ficheiro chamado *requeriments.txt* e depois executar o comando *pip3 install -r requirements.txt*

---

```
email-validator==1.1.3
entrypoints==0.3
Flask==1.1.1
Flask-Login==0.5.0
Flask-SQLAlchemy==2.5.1
Flask-WTF==0.15.1
httplib2==0.14.0
importlib-metadata==1.5.0
incremental==16.10.1
Jinja2==2.10.1
mariadb==1.0.8
MarkupSafe==1.1.0
more-itertools==4.2.0
oauthlib==3.1.0
pexpect==4.6.0
pyasn1==0.4.2
pyasn1-modules==0.2.1
PyGObject==3.36.0
PyHamcrest==1.9.0
pyinotify==0.9.6
PyJWT==1.7.1
pymacaroons==0.13.0
PyMySQL==1.0.2
PyNaCl==1.3.0
pyOpenSSL==19.0.0
pyrsistent==0.15.5
pyserial==3.4
python-apt==2.0.0+ubuntu0.20.4.6
python-debian===0.1.36ubuntu1
PyYAML==5.3.1
requests==2.22.0
requests-unixsocket==0.2.0
SecretStorage==2.3.1
simplejson==3.16.0
SQLAlchemy==1.4.26
ssh-import-id==5.10
systemd-python==234
Twisted==18.9.0
ufw==0.36
urllib3==1.25.8
```

```
wadllib==1.3.3
Werkzeug==0.16.1
WTForms==2.3.3
```

### 3.2.4 sync_between_webservers.sh

Este script foi útil para que quando fosse feito um *commit* no *github*, o *webserver01* sincronizasse o código atualizado com o *webserver02*

```bash
#!/usr/bin/env bash

# automating git stuff

echo "Whats the commit message?"
read message
git add .
git commit -m "${message}"
echo "Pushing data ... "
git push


echo "Syncing to webserver02 ..."
# sync a folder from webserver01 to webserver02
rsync -rt /home/brun0/repo/ brun0@192.168.1.181:/home/brun0/repo --delete-after
```

### 3.2.5 flask_app

#### carrinho.py

```python
from app import app
```

#### config.py

Criar variável de ambiente com o nome **DB_HOST**

```
DB_HOST=mysql+pymysql://<username>:<password>@<db_ip>:<db_port>/<db_name>
```

```python
import os

basedir = os.path.abspath(os.path.dirname(__file__))

class Config(object):
   SECRET_KEY = "something"
   SQLALCHEMY_DATABASE_URI = os.environ.get('DB_HOST')
   SQLALCHEMY_TRACK_MODIFICATIONS = False
```

### 3.2.6 app

#### __init__.py

```python
from flask import Flask
from config import Config
```

```python
from flask_sqlalchemy import SQLAlchemy
from flask_login import LoginManager


app = Flask(__name__)
app.config.from_object(Config)
db = SQLAlchemy(app)

login = LoginManager(app)
login.login_view = "login"

from app import routes,models
```

### forms.py

```python
from flask_wtf import FlaskForm
from wtforms import StringField, PasswordField, SubmitField, BooleanField, IntegerField
from wtforms.validators import DataRequired, Length, Email, EqualTo


class LoginForm(FlaskForm):
    email = StringField('Email',validators=[DataRequired(), Email()])
    password = PasswordField('Password', validators=[DataRequired()])
    submit = SubmitField('Login')

class ProductForm(FlaskForm):
        product_type = StringField("Tipo", validators=[DataRequired()])
        quantity = IntegerField("Quantidade", validators=[DataRequired()])
        local = StringField("Local de Compra", validators=[DataRequired()])
        submit = SubmitField('Inserir')
```

### models.py

```python
from app import db, login
from werkzeug.security import generate_password_hash, check_password_hash
from flask_login import UserMixin

@login.user_loader
def load_user(id):
    return Clientes.query.get(int(id))

class Clientes(UserMixin, db.Model):
        id_cliente = db.Column(db.Integer, primary_key=True)
        nome = db.Column(db.String(64), index=True, unique=True)
        email = db.Column(db.String(120), index=True, unique=True)
        password = db.Column(db.String(128))
        compras = db.relationship("Compras", backref="cliente")

        def __repr__(self):
                return f"{self.nome}"

        def set_password(self, password):
                self.password = generate_password_hash(password)

        def check_password(self, password):
```

```
            return check_password_hash(self.password,password)

        def get_id(self):
            return self.id_cliente

class Compras(db.Model):
        id_compras = (db.Column(db.Integer, primary_key=True))
        tipo = db.Column(db.String(140))
        quantidade = db.Column(db.Integer)
        local = db.Column(db.String(140))
        id_cliente = db.Column(db.Integer, db.ForeignKey("clientes.id_cliente"))

        def __repr__(self):
            return f"{self.id_compras}"
```

**routes.py**

```
from flask import render_template, flash, redirect, url_for, request, jsonify, make_response
from app import app,db
from app.forms import LoginForm, ProductForm
from flask_login import current_user, login_user, logout_user, login_required
from app.models import Clientes, Compras


# index
@app.route("/")
def index():
    return render_template("index.html")

# login
@app.route('/login', methods=['GET', 'POST'])
def login():
        if current_user.is_authenticated:
            return redirect(url_for("carrinho"))

        form = LoginForm()
        if form.validate_on_submit():
            # query a bd
            user = Clientes.query.filter_by(email=form.email.data).first()
            # verifica o resultado da query, se nao existir ...
            if user is None or not user.check_password(form.password.data):
                flash("Login invalido","danger")
                return redirect(url_for("login"))
            # se existir faz login
            else:
                login_user(user)
            return redirect(url_for("carrinho"))
        return render_template("login.html", title="Login", form=form)

# logout
@app.route("/logout")
def logout():
        logout_user()
        flash("Logout com sucesso", "info")
        return redirect(url_for("index"))
```

```python
# carrinho
@app.route("/carrinho/", methods=["GET", "POST"])
@login_required
def carrinho():
        # lista produtos atuais do cliente
        customer_list = db.session.query(Compras.id_compras.label("id_compras"),
                                         Compras.tipo.label("Tipo"),
                                         Compras.quantidade.label("Quantidade"),
                                         Compras.local.label("Local"))\
                                         .join(Clientes, Compras.id_cliente == Clientes.id_cliente)\
                                         .filter(Compras.id_cliente==current_user.get_id()).all()

    # adicionar produtos ao carrinho
        form = ProductForm()
        if form.validate_on_submit():
                compras = Compras(tipo = form.product_type.data,
                                  quantidade = form.quantity.data,
                                  local = form.local.data,
                                  id_cliente = current_user.get_id())
                db.session.add(compras)
                db.session.commit()

                return redirect(url_for("carrinho"))

        return render_template("carrinho.html", title="Lista", form=form,
            customer_list=customer_list)

# apagar artigo
@app.route("/apagar_artigo",methods=["POST"])
@login_required
def delete_item():
        # recebe o POST feito no js
        req = request.get_json()
        # aplica a query
        Compras.query.filter_by(id_compras=int(req["id"])).delete()
        db.session.commit()

        return redirect(url_for("carrinho"))
```

### 3.2.7 static

**main.css**

```css
  body {
  background: #fafafa;
  color: #333333;
  margin-top: 5rem;
}

h1, h2, h3, h4, h5, h6 {
  color: #444444;
}

.bg-steel {
  background-color: #5f788a;
}
```

```
.site-header .navbar-nav .nav-link {
  color: #cbd5db;
}

.site-header .navbar-nav .nav-link:hover {
  color: #ffffff;
}

.site-header .navbar-nav .nav-link.active {
  font-weight: 500;
}

.content-section {
  background: #ffffff;
  padding: 10px 20px;
  border: 1px solid #dddddd;
  border-radius: 3px;
  margin-bottom: 20px;
}

.article-title {
  color: #444444;
}

a.article-title:hover {
  color: #428bca;
  text-decoration: none;
}

.article-content {
  white-space: pre-line;
}

.article-img {
  height: 65px;
  width: 65px;
  margin-right: 16px;
}

.article-metadata {
  padding-bottom: 1px;
  margin-bottom: 4px;
  border-bottom: 1px solid #e3e3e3
}

.article-metadata a:hover {
  color: #333;
  text-decoration: none;
}

.article-svg {
  width: 25px;
  height: 25px;
  vertical-align: middle;
}
```

### 3.2.8 templates

**layout.html**

```
<!DOCTYPE html>
<html>
<head>
    <!-- Required meta tags -->
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
    <link href="https://use.fontawesome.com/releases/v5.6.3/css/all.css" rel="stylesheet">
    <!-- Bootstrap CSS -->
    <link rel="stylesheet"
        href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css"
        integrity="sha384-Gn5384xqQ1aoWXA+058RXPxPg6fy4IWvTNh0E263XmFcJlSAwiGgFAW/dAiS6JXm"
        crossorigin="anonymous">

    <link rel="stylesheet" type="text/css" href="{{ url_for('static', filename='main.css') }}">
    <title>Lista de compras 01</title>
</head>
<body>
    <header class="site-header">
      <nav class="navbar navbar-expand-md navbar-dark bg-steel fixed-top">
        <div class="container">
          <a class="navbar-brand mr-4" href="/">Lista de Compras</a>
          <button class="navbar-toggler" type="button" data-toggle="collapse"
              data-target="#navbarToggle" aria-controls="navbarToggle" aria-expanded="false"
              aria-label="Toggle navigation">
            <span class="navbar-toggler-icon"></span>
          </button>
          <div class="collapse navbar-collapse" id="navbarToggle">
            <div class="navbar-nav mr-auto">
            </div>
            <!-- Navbar Right Side -->
            <div class="navbar-nav">
              {% if current_user.is_anonymous %}
              <a class="nav-item nav-link" href="{{ url_for('login') }}">Login</a>
              {% else %}
              <a class="nav-item nav-link" href="{{ url_for('logout') }}">Logout</a>
              <a class="nav-item nav-link" href="{{ url_for('carrinho') }}">Carrinho</a>
              {% endif %}
            </div>
          </div>
        </div>
      </nav>
    </header>
    <main role="main" class="container">
      <div class="row">

          {% with messages = get_flashed_messages(with_categories=true) %}
            {% if messages %}
              {% for category, message in messages %}
                <div class="alert alert-{{ category }}" style="margin:auto">
                  {{ message }}
                </div>
              {% endfor %}
            {% endif %}
        {% endwith %}
```

```
      {% block content %}{% endblock %}
    </div>
  </main>


  <!-- Optional JavaScript -->
  <!-- jQuery first, then Popper.js, then Bootstrap JS -->
  <script src="https://code.jquery.com/jquery-3.1.1.min.js"
      integrity="sha384-KJ3o2DKtIkvYIK3UENzmM7KCkRr/rE9/Qpg6aAZGJwFDMVNA/GpGFF93hXpG5KkN"
      crossorigin="anonymous"></script>
  <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.12.9/umd/popper.min.js"
      integrity="sha384-ApNbgh9B+Y1QKtv3Rn7W3mgPxhU9K/ScQsAP7hUibX39j7fakFPskvXusvfa0b4Q"
      crossorigin="anonymous"></script>
  <script src="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/js/bootstrap.min.js"
      integrity="sha384-JZR6Spejh4U02d8jOt6vLEHfe/JQGiRRSQQxSfFWpi1MquVdAyjUar5+76PVCmYl"
      crossorigin="anonymous"></script>
</body>
</html>
```

### index.html

```
{% extends "layout.html" %}
{% block content %}

<div class="container ">
<div class="d-flex justify-content-center p-5">
  <img class="mt-5" src="static/cesto.jpg" alt="icon" style="width:350px">
  </div>
  <div class="d-flex justify-content-center text-center p-5">
    <h1>A sua lista de compras<h1/>
    </div>
</div>
{% endblock content %}
```

### carrinho.html

```
{% extends "layout.html" %}
{% block content %}

<script>
  function remove_entry(){
      var entry = {id:event.srcElement.id};

      fetch(`${window.origin}/apagar_artigo`,{
          method: "POST",
          credentials: "include",
          body: JSON.stringify(entry),
          cache: "no-cache",
          headers: new Headers({
              "content-type":"application/json"
          })
      })

  }
</script>
```

```
{% if customer_list|length > 0 %}
<div class="col-8">
<div class="container">
      <table class="table table-bordered text-center" id="mytable">
        <thead>
          <tr>
            <th scope="col">ID</th>
            <th scope="col">Tipo</th>
            <th scope="col">Quantidade</th>
            <th scope="col">Local de Compra</th>
            {% for item in customer_list %}
          </tr>
        </thead>
        <tbody>
          <tr>
            <th scope="row">{{ loop.index }}</th>
            <td>{{ item[1] }}</td>
            <td>{{ item[2] }}</td>
            <td>{{ item[3] }}</td>
            <td>
              <form>
                  <button id={{ item[0] }} type="submit" class="btn btn-danger"
                       onclick="remove_entry();">Apagar</button>
                </form>
            </td>
            {% endfor %}
            </tr>
        </tbody>
      </table>
</div>
</div>
{% else %}

<div class="d-flex justify-content-center">
<div class=" d-flex alert alert-dark text-center" style="max-height:62px">Neste momento nao tem
    nenhum produto no seu cesto
</div>
</div>
{% endif %}


  <div class="col-4">
    <div class="content-section">
        <form method="POST" action="">
            {{ form.hidden_tag() }}
            <fieldset class="form-group">
                <legend class="border-bottom mb-4">Carrinho</legend>
                <div class="form-group">
                    {{ form.product_type.label(class="form-control-label") }}
                    {% if form.product_type.errors %}
                        {{ form.product_type(class="form-control form-control-lg is-invalid") }}
                        <div class="invalid-feedback">
                            {% for error in form.product_type.errors %}
                                <span>{{ error }}</span>
                            {% endfor %}
                        </div>
                    {% else %}
                        {{ form.product_type(class="form-control form-control-lg") }}
```

```
                {% endif %}
            </div>
            <div class="form-group">
                {{ form.quantity.label(class="form-control-label") }}
                {% if form.quantity.errors %}
                    {{ form.quantity(class="form-control form-control-lg is-invalid") }}
                    <div class="invalid-feedback">
                        {% for error in form.quantity.errors %}
                            <span>{{ error }}</span>
                        {% endfor %}
                    </div>
                {% else %}
                    {{ form.quantity(class="form-control form-control-lg") }}
                {% endif %}
            </div>
            <div class="form-group">
                {{ form.local.label(class="form-control-label") }}
                {% if form.local.errors %}
                    {{ form.local(class="form-control form-control-lg is-invalid") }}
                    <div class="invalid-feedback">
                        {% for error in form.local.errors %}
                            <span>{{ error }}</span>
                        {% endfor %}
                    </div>
                {% else %}
                    {{ form.local(class="form-control form-control-lg") }}
                {% endif %}
            </div>
        </fieldset>
        <div class="form-group">
          {{ form.submit(class="btn btn-outline-info") }}
        </div>
    </form>
  </div>
  </div>

{% endblock content %}
```

### login.html

```
{% extends "layout.html" %}
{% block content %}
  <div class="container">
    <div class="content-section">
        <form method="POST" action="">
            {{ form.hidden_tag() }}
            <fieldset class="form-group">
                <legend class="border-bottom mb-4">Log In</legend>
                <div class="form-group">
                    {{ form.email.label(class="form-control-label") }}
                    {% if form.email.errors %}
                        {{ form.email(class="form-control form-control-lg is-invalid") }}
                        <div class="invalid-feedback">
                            {% for error in form.email.errors %}
                                <span>{{ error }}</span>
                            {% endfor %}
                        </div>
```

```
                {% else %}
                    {{ form.email(class="form-control form-control-lg") }}
                {% endif %}
            </div>
            <div class="form-group">
                {{ form.password.label(class="form-control-label") }}
                {% if form.password.errors %}
                    {{ form.password(class="form-control form-control-lg is-invalid") }}
                    <div class="invalid-feedback">
                        {% for error in form.password.errors %}
                            <span>{{ error }}</span>
                        {% endfor %}
                    </div>
                {% else %}
                    {{ form.password(class="form-control form-control-lg") }}
                {% endif %}
            </div>
        </fieldset>
        <div class="form-group">
          {{ form.submit(class="btn btn-outline-info") }}
        </div>
    </form>
</div>

{% endblock content %}
```

## 3.3 Base de Dados

### 3.3.1 Criação da base de dados

Como a conexão é feita a partir de dois servidores remotos(**haproxy01** e **haproxy02**), foi preciso garantir todos os privilegios a esses dois servidores

```
mysql -u root -p
mysql> GRANT ALL ON *.* to <username>@'<ip_haproxy01>' IDENTIFIED BY '<password>';
mysql> GRANT ALL ON *.* to <username>@'<ip_haproxy02>' IDENTIFIED BY '<password>';
mysql> FLUSH PRIVILEGES;
mysql> exit;
sudo service mariadb restart
mysql -u root -p
mysql> create database carrinho;
```

### 3.3.2 Criação das tabelas

```
create table clientes(
    id_cliente int auto_increment,
    nome varchar(255) not null,
    email varchar(255) not null,
    password varchar(255) not null,
    primary key(id_cliente)
);


create table compras(
    id_compras int auto_increment,
```

```
    id_cliente int,
    tipo varchar(255) not null,
    quantidade int not null,
    local varchar(255) not null,
    primary key(id_compras),
    foreign key(id_cliente)
    references clientes(id_cliente)
);
```

### 3.3.3 Inserção de dados nas tabelas

```
insert into clientes(nome, email,password) values ("Bruno", "a2019100036@isec.pt",
    "passwordsegura");
insert into clientes(nome, email,password) values ("Teixeira", "brunoalexandre3@hotmail.com",
    "passwordsegura");

insert into compras(id_cliente,tipo,quantidade,local) values (1, "Laranjas", 5,"PingoDoce");
insert into compras(id_cliente,tipo,quantidade,local) values (2, "Macas", 2,"PingoDoce");
insert into compras(id_cliente,tipo,quantidade,local) values (1, "Peras", 3,"PingoDoce");
```

### 3.3.4 Selecionar todos os artigos do carrinho do Cliente 1

```
select compras.id_compras "Id Compras", compras.id_cliente "Id Cliente", clientes.nome "Nome
    Cliente", compras.tipo, compras.quantidade, compras.local from compras inner join clientes on
    compras.id_cliente = clientes.id_cliente and clientes.id_cliente = 1;
```