Índice

6 Guía Balanceo Apache

MATERIAL

- Los contenidos de la unidad y esta guía
- 2/3 Máquinas Virtuales Ubuntu 20.04 Desktop.
- Virtualbox
- Ordenador con S.O. Windows 10.
- Navegador para comprobar la realización de la tarea.
- Procesador de textos para elaborar la documentación y los archivos de la tarea.
- Acceso a Internet.

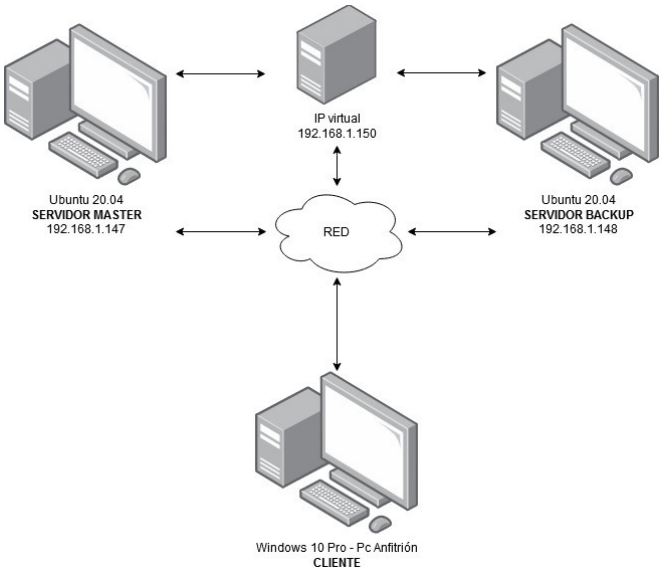
En esta parte de la actividad se plantean dos escenarios, en principio sólo se pide que realicen uno de los dos:

- Balanceo failover (fallos) Apache.
- Balanceo cargas Apache.

El primero conmuta en caso de que detecte que el servidor que está actuando como MASTER ha caído bien el equipo, el servicio o la conexión y pasa al BACKUP, mientras que el segundo hace un reparto entre los servidores web empleados de las peticiones de los usuarios.

6.3 Balanceo fallos Apache

En el enunciado se solicitaba implementar el balanceo de cargas por medio del CARP (Common Address Redundancy Protocol), pero tras analizarlo se ha visto que es más complejo de llevar a cabo, por lo que se ha optado por el protocolo VRRP (Virtual Router Redundancy Protocol) para lo que necesitamos instalar el servicio KeepAlived. El funcionamiento se resume en el siguiente diagrama.



Este sistema se resume en que el KeepAlived crea un elemento virtual con una IP virtual, que se va a encargar de supervisar que máquina está operativa y nos va a ofrecer el servicio web Apache a través de la IP virtual.

Un aspecto importante es identificar tanto a la hora de crear la máquina como en el prompt del terminal la máquina MASTER y la BACKUP para identificarlas de manera inequívoca y saber en todo momento que máquina estamos empleando. Para cambiar el prompt recuerda que la hacemos editando el hostname, de la siguiente forma:

#sudo nano /etc/hostname

Para que surta efecto debemos reiniciar la máquina.

La entidad virtual nos va a facilitar la disponibilidad del servicio web. Para ello se ha de definir un equipo como MASTER que será el que por defecto está sirviendo las páginas web y en ausencia de este entra en juego el servidor BACKUP. Los parámetros que va a tener en cuenta son:

- Servidor activo.
- Servicio activo.
- Conexión.

En el caso de que cualquiera de los parámetros referidos falle, la entidad virtual nos facilitará el acceso al servidor que esté operativo.

A continuación se indican los pasos para realizar esta configuración. Debemos tener las dos máquinas operativas. En este caso las vamos a comandar las dos desde la máquina MASTER y nos conectaremos a la otra vía SSH. Abriremos dos terminales en paralelo para trabajar.

En ambas máquinas tenemos que instalar el SSH, para poder acceder de manera remota. La instalación la conseguimos hacer con:

#sudo apt-get install ssh

Después de lanzar este comando desde la máquina MASTER lanzamos la siguiente orden para acceder al secure shell:

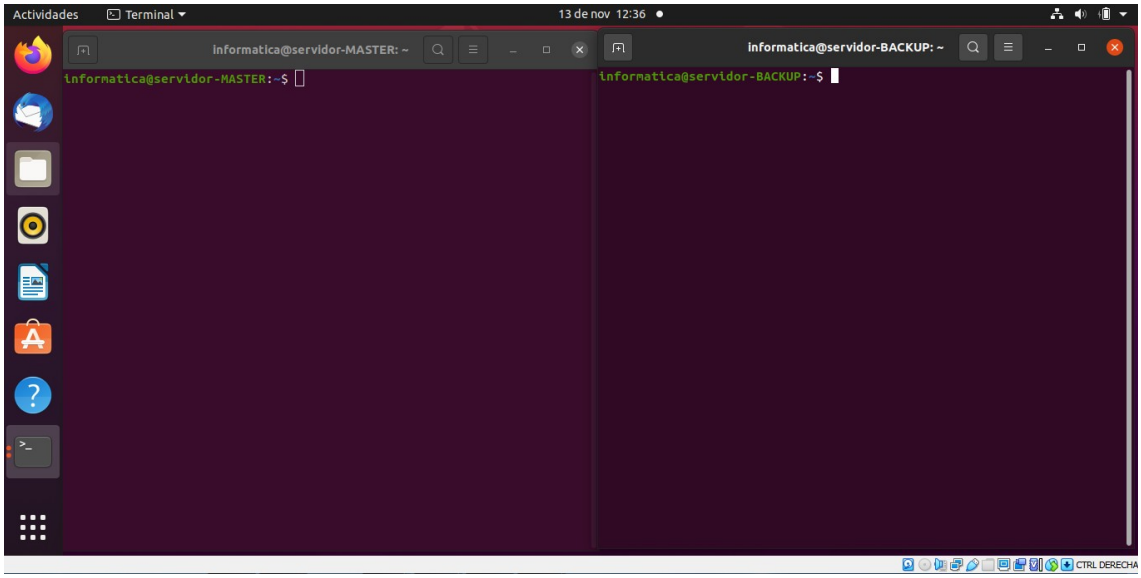
#ssh usuario_maquina@IP_maquina_BACKUP

```
informatica@servidor-MASTER:~$ ssh informatica@192.168.1.148
The authenticity of host '192.168.1.148 (192.168.1.148)' can't be est
ablished.
ECDSA key fingerprint is SHA256:wHknM8NB7T6FOEZsewRsAHQaH6cxRy26Au2km
yHch+A.
Are you sure you want to continue connecting (yes/no/[fingerprint])?
```

En el primer acceso nos pedirá aceptar el certificado o fingerprint. Contestamos: **yes**

y veremos como nos cambia el prompt

Así quedará nuestro entorno de trabajo



Empezamos realizando las instalaciones y configuraciones en la máquina MASTER.

6.3.1 Servidor MASTER

Instalamos el KeepAlived

Para la instalación prodemos con el comando.

#sudo apt-get install keepalived -y

Cuando finalice la instalación nos movemos a la carpeta del keepalived con:

#cd /etc/keepalived

Listamos los archivos y directorios de esta carpeta y vemos que está vacía

#ll

ó

#ls -l

```
informatica@servidor-MASTER:~$ cd /etc/keepalived/
informatica@servidor-MASTER:/etc/keepalived$ ls
informatica@servidor-MASTER:/etc/keepalived$ ll
total 16
drwxr-xr-x   2 root root  4096 feb 20  2020 ./
drwxr-xr-x 132 root root 12288 nov 13 12:37 ../
```

Copiaremos un ejemplo de configuración que viene por defecto en la instalación. Para realizarlo ejecutamos la siguiente instrucción.

#sudo cp /usr/share/doc/keepalived/samples/keepalived.conf.vrrp.localcheck keepalived.conf

```
informatica@servidor-MASTER:/etc/keepalived$ sudo cp /usr/share/doc/ke
epalived/samples/keepalived.conf.vrrp.localcheck keepalived.conf
informatica@servidor-MASTER:/etc/keepalived$ ll
total 20
drwxr-xr-x   2 root root  4096 nov 13 12:49 ./
drwxr-xr-x 132 root root 12288 nov 13 12:37 ../
-rw-r--r--   1 root root  3019 nov 13 12:49 keepalived.conf
```

Editamos el archivo de configuración del keepalived.

#sudo nano keepalived.conf

En este archivo que hemos copiado originalmente tenemos varios scripts e instancias de configuración.

Dejaremos de referencia el primer script y la primera de las instancias el resto lo borraremos.

```
 GNU nano 4.8                  keepalived.conf                  Modificado
! Configuration File for keepalived

vrrp_script chk_sshd {
      script "killall -0 sshd"        # cheaper than pidof
      interval 2                      # check every 2 seconds
vrrp_instance VI_1 {
    interface eth0
    state MASTER
    virtual_router_id 51
    priority 100
    virtual_ipaddress {
        192.168.200.18/25
    }
    track_interface {
       eth1 weight 2   # prio = +2 if UP
       eth2 weight -2  # prio = -2 if DOWN
       eth3            # no weight, fault if down
    }
    track_script {
       chk_sshd               # use default weight from the script
       chk_haproxy weight 2   # +2 if process is present
       chk_http_port
       chk_https_port
       chk_smtp_port
    }
}
```

Aprovechando este código lo vamos a adaptar a nuestras necesidades, empezando por el script que va a controlar ver el estado de las máquinas servidoras en cuanto a conexión, servicio y la propia máquina.

Personalizamos el nombre del script con apache2

Comprobamos que nos está caído el apache 2 “Killall -0 apache2”, esto comprueba si apache se ha caído y nos daría un número distinto de 0. En el caso de que esto sucede se pone en marcha el protocolo VRRP para ver quién es el próximo master.

Podríamos configurar el intervalo en el que queremos que se revise si el servidor está caído, que por defecto es cada segundo. Prescindimos de él para que lo revise cada segundo y borramos el resto de parámetros del script de apache.

Ahora configuramos la instancia personalizando el nombre del interfaz que tiene la máquina MASTER, la prioridad la pasamos de 100 a la máxima 255, configuramos la IP virtual en este caso 192.168.1.150, cada alumno la deberá personalizar a una IP libre de la red en la que se encuentre trabajando.

Eliminamos la parte del seguimiento de los interfaces (track_interface) y dejamos el seguimiento de los scripts (track_script), en este último eliminamos los nombres de los scripts que contiene y ponemos el de apache2 que hemos creado.

Añadimos la sección global_defs, en la que incluiremos las directivas de:

- enable_script_security
- script_user root

Para finalizar el archivo de configuración incluimos la ruta absoluta a killall:

- script "/usr/bin/killall -0 apache2"

En la siguiente captura se muestra como debe quedar nuestro archivo de configuración para el keepalived.

```
  GNU nano 4.8                  keepalived.conf                  Modificado

! Configuration File for keepalived
global_defs{
       enable_script_security
       script_user root
}
vrrp_script apache2 {
       script "/usr/bin/killall -0 apache2"        # cheaper than pid>
}

vrrp_instance VI_1 {
    interface enp0s3
    state MASTER
    virtual_router_id 51
    priority 100
    virtual_ipaddress {
        192.168.1.150
    }

    track_script {
       apache2
    }
}
```

Reiniciamos el servicio de keepalived y podemos ver el estado, para comprobar que está todo correcto y se carga la configuración que hemos efectuado.

#sudo systemctl restart keepalived

#sudo systemctl statys keepalived

6.3.2 Servidor BACKUP

Ahora pasamos a configurar el servidor BACKUP.

Instalar KeepAlived

#sudo apt-get install keepalived -y

Del mismo modo que en la máquina MASTER, después de la instalación no tenemos ningún archivo en la carpeta de instalación del keepalived, por lo que vamos a realizar una copia del fichero que configuramos para la máquina MASTER.

#sudo scp usuario__servidor_MASTER@IP_maquina_MASTER:/etc/keepalived/keepalived.conf /etc/keepalived/

Editamos el archivo para realizar las adaptaciones al servidor BACKUP.

#sudo nano /etc/keepalived/keepalived.conf

Sólo tenemos que hacer dos cambios sobre el state y la prioridad, quedando el archivo:

Reiniciamos el servicio y mostramos el estado.

#sudo systemctl restart keepalived

#sudo systemctl statys keepalived

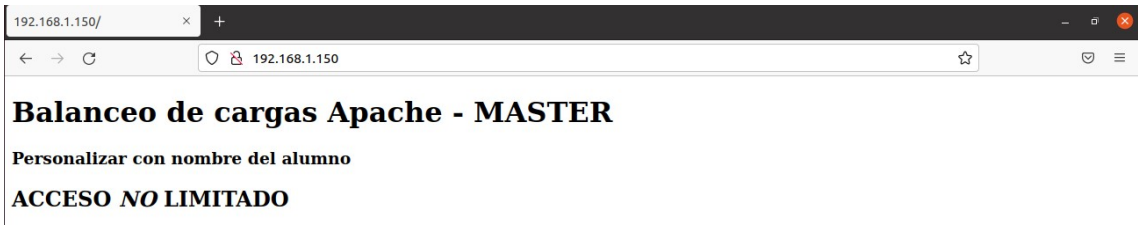
Con esto ya tendríamos configurado nuestro sistema para balancear en el caso de que surja algún fallo.

Realizamos algunas pruebas para verificar que funciona correctamente. Para ello vamos a ver las tres IPs desde el navegador de la máquina cliente.

MASTER

BACKUP

IP VIRTUAL



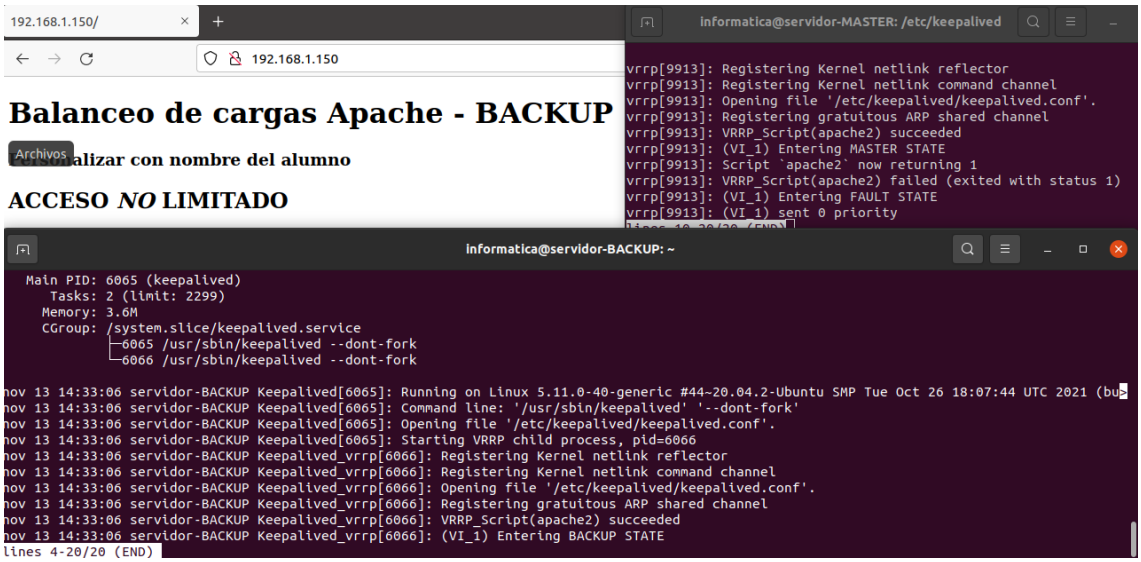
Ahora vamos a probar que ante los diferentes escenarios de fallo funciona como corresponde.

Fallo servicio web en MASTER

Paramos el servicio web en el MASTER.

#sudo systemctl stop apache2

Vemos antes de parar el servicio la página del MASTER, y tras parar el servicio en el MASTER, refrescamos la web del navegador y vemos la página del BACKUP.

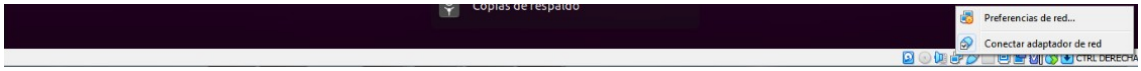


Fallo de la conexión MASTER

Activamos el servicio web en el MASTER.

#sudo systemctl start apache2

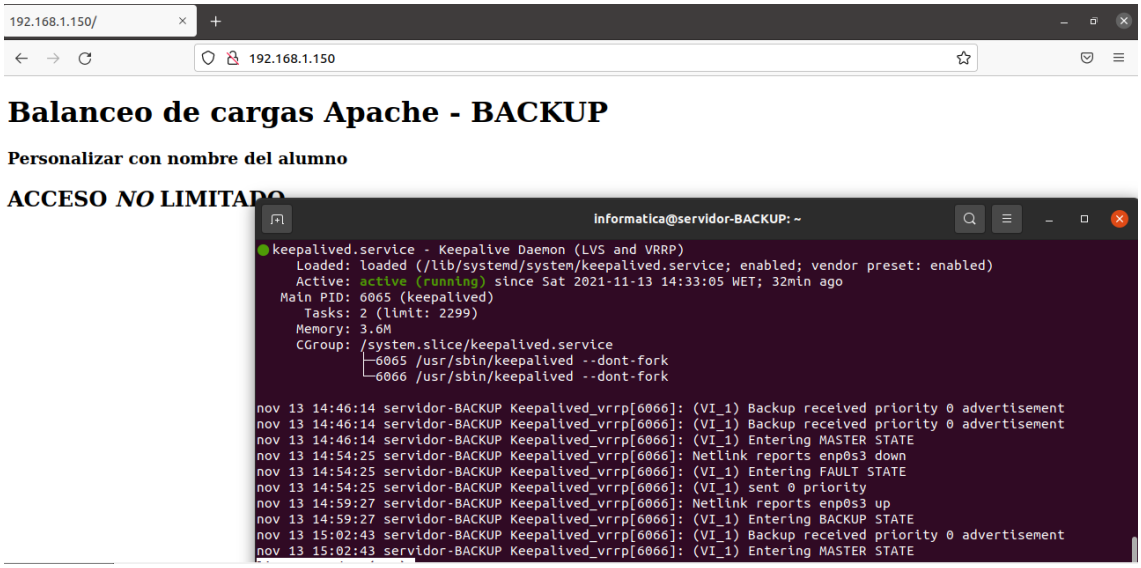
Ahora desconectamos el cableado en equipo BACKUP, para ello en el icono de los dos monitores que tenemos en la parte inferior derecha de la máquina, pulsamos con botón derecho y elegimos “Conectar adaptador de red”



Comprobamos la operatividad de nuestro balanceo de Apache.

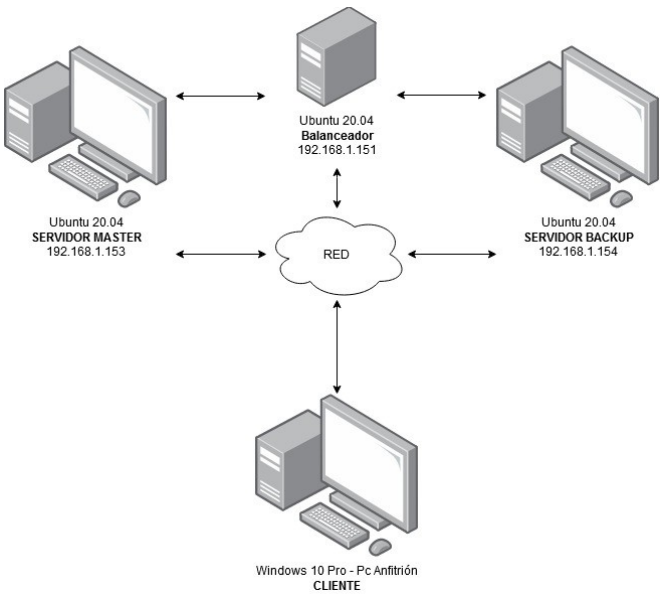
Caída del servidor

Activamos la conexión en el servidor BACKUP y apagamos el servidor MASTER.



6.4 Balanceo cargas clúster Apache

En esta parte vamos a realizar la siguiente configuración:



A diferencia que el anterior aquí contamos con 3 máquinas virtuales de Ubuntu 20.04:

- Servidor MASTER
- Servidor BACKUP
- Balanceador.

El balanceador sería una máquina real, frente al anterior que había una IP virtual que era la que monitorizaba los equipos y en función del estado en ella se presentaba la página del equipo con servicio y conexión activa.

Para realizar este apartado deberemos instalar en las máquinas virtules Apache si no lo tenemos instalado aún.

#sudo apt-get install apache2

6.4.1 Balanceador

Pasamos a configurar la máquina virtual que va actuar como balanceador de cargas repartiendo las peticiones entre los nodos o servidores disponibles dentro de nuestro clúster. Para ello tenemos que activar los siguientes módulos en él:

- proxy
- proxy_http
- proxy_balancer
- lbmethod_byrequests

Los principales algoritmos de balanceo de carga:

- **Round-Robin.-** Este algoritmo consiste en distribuir de manera equitativa las solicitudes a los diferentes servidores.
- **Weighted Round-Robin.-** Similar al anterior pero con la base de ponderadores que establecen la capacidad por servidor de procesar información, esto considerando que un servidor más rápido podría soportar el doble de procesamiento.
- **Least Connection.-** Este algoritmo asigna peticiones al servidor que tiene menos conexiones activas.
- **Weighted Least Connection.-** Similar al anterior pero con la base de un ponderador que establece la capacidad por servidor de procesar información.
- **Fastest.-** Se determina el servidor que está tardando menor tiempo en responder y se le asigna la petición.

En este caso usaremos el Round-Robin.

#sudo a2enmod proxy proxy_http proxy_balancer lbmethod_byrequests

Una vez habilitados los módulos indicados debemos de reiniciar el servicio Apache para que surtan efecto los cambios realizados.

#sudo systemctl restart apache2

Ahora configuraremos el sitio por defecto de Apache (000-default.conf) para establecer la configuración del balanceador.

#sudo nano /etc/apache2/sites-available/000-default.conf

En el deberemos reflejar las siguientes directivas:

```
<VirtualHost *:80>

ProxyRequests on

        ServerAdmin administrador@gmail.com

        ServerAlias dpl.com

        <Proxy balancer://balanceador>

                BalancerMember http://192.168.1.153 loadfactor=4

                BalancerMember http://192.168.1.154 loadfactor=4

                Order allow,deny

                Allow from all

                ProxySet lbmethod=byrequests

        </Proxy>
```

```
        <Location /balancer-manager>
                SetHandler balancer-manager
                Order allow,deny
                Allow from all
        </location>
        ProxyPass /balancer-manager !
        ProxyPass / balancer://balanceador/
</VirtualHost>
```

La directiva ProxySet lbmethod_byrequests establece el método de reparto de las cargas en este caso el Round-Robin en función del reparto establecido en loadfactor.

```
 GNU nano 4.8                          /etc/apache2/sites-available/000-default.conf
<VirtualHost *:80>
ProxyRequests on
        ServerAdmin administrador@gmail.com
        ServerAlias dpl.com

        <Proxy balancer://balanceador>
                BalancerMember http://192.168.1.153 loadfactor=4
                BalancerMember http://192.168.1.154 loadfactor=4
                Order deny,allow
                Allow from all
                ProxySet lbmethod=byrequests
        </Proxy>
        <Location /balancer-manager>
                SetHandler balancer-manager
                Order allow,deny
                Allow from all
        </location>
        ProxyPass /balancer-manager !
        ProxyPass / balancer://balanceador/

        ErrorLog ${APACHE_LOG_DIR}/error.log
        CustomLog ${APACHE_LOG_DIR}/access.log combined


</VirtualHost>
```

Reiniciamos el servicio de Apache en todas las máquinas virtuales.

#sudo systemctl restart apache2

Realizamos la pruebas realizando la solicitud de la IP del balanceador en el navegador web de la máquina balanceadora y cada vez que hagamos una petición a la IP del

balanceador nos aparecerá en secuencia las páginas web del servidor MASTER y la del BACKUP.