

P4: Alta Disponibilidad con HAProxy

1. Descripción

El objetivo del ejercicio consiste en configurar un cluster de alta disponibilidad en modo activo-pasivo usando LinuxHA. Para ello, es necesario:

- Configurar Corosync para gestionar los nodos del clúster.
- Configurar Pacemaker para gestionar los recursos (servicios) que ofrecen los nodos del clúster.

2. Entorno de prácticas

En estas prácticas se empleará el software de virtualización VIRTUALBOX para simular los equipos GNU/Linux sobre los que se realizarán las pruebas.

3. Imágenes a utilizar

Se proporcionan scripts de instalación tanto para GNU/Linux como para Windows. Windows es bastante inestable con algunas configuraciones de la Máquina Virtual que se usarán durante la realización de las prácticas. Por ello, se recomienda encarecidamente usar Linux como sistema base.

- Script GNU/Linux: ejercicio-linuxha.sh (desde línea de comandos)

01	alumno@pc: \$ sh ejercicio-linuxha.sh
----	---------------------------------------

- MS Windows: ejercicio-haproxy.ps1 (desde cmd)

01	Powershell.exe -executionpolicy bypass -file ejercicio-linuxha.ps1
----	--

Notas:

- Se pedirá un identificador (sin espacios) para poder reutilizar las versiones personalizadas de las imágenes creadas. Se recomienda usar como identificador CDA_<numero_del_grupo>_<INICIALES>.
- En ambos scripts la variable \$DIR_BASE especifica donde se descargarán las imágenes y se crearán las MVs.
- Por defecto en GNU/Linux será en \$HOME/CDA2425 y en Windows en C:/CDA2425.
- Puede modificarse antes de lanzar los scripts para hacer la instalación de las imágenes en otro directorio más conveniente (disco externo, etc)

Centros de Datos

David Ruano Ordás
Departamento de Informática
Lenguajes y Sistemas Informáticos

CURSO 2024-2025

- Si se hace desde el script anterior, se pueden arrancar las instancias VIRTUALBOX desde el interfaz gráfico de VirtualBOX o desde la línea de comandos con `VBoxManage startvm <nombre MV> _<id>`

4. Credenciales de acceso

La distribución Linux incluida en la MV tiene datos de alta dos usuarios con las siguientes credenciales y permisos:

login	password	permisos
root	purple	root
usuario	purple	permite sudo

5. Entorno desplegado

El entorno desplegado para la realización de las prácticas está formado por 3 máquinas virtuales (cliente, servidor1 y servidor2).

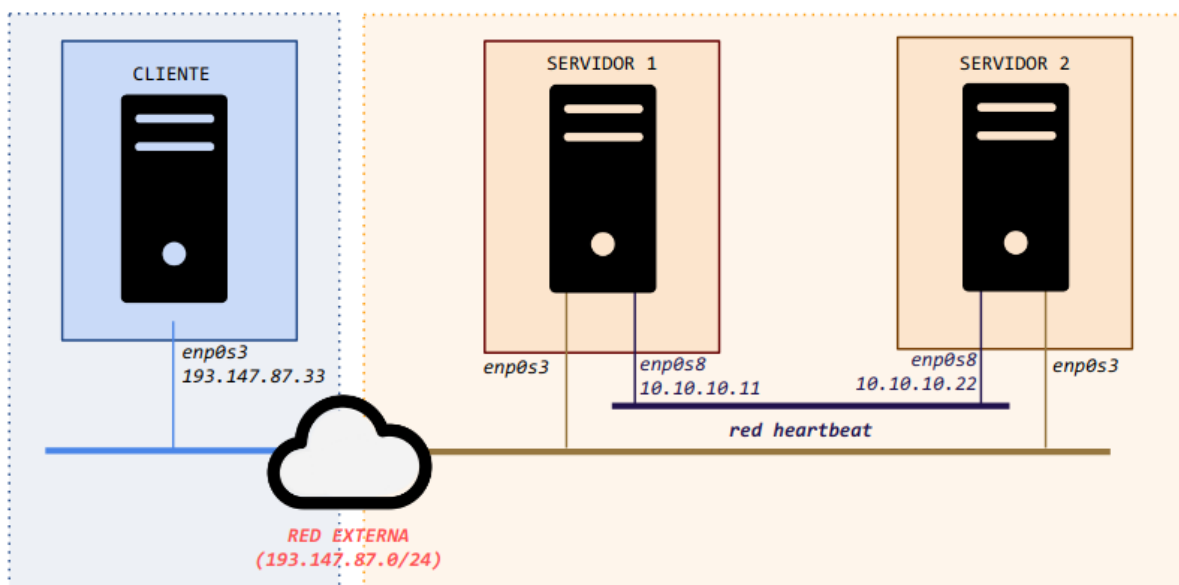


Fig 1. Entorno de trabajo.

- CLIENTE tiene la dirección IP 193.147.87.33 y será el encargado de realizar las peticiones a los nodos que forman el cluster de alta disponibilidad (Servidor 1 o Servidor 2).
- Servidor 1 tiene la dirección IP (10.10.10.11) que dispone de un servidor web.

Centros de Datos

David Ruano Ordás
Departamento de Informática
Lenguajes y Sistemas Informáticos

CURSO 2024-2025

- Servidor 2 tiene la dirección IP (10.10.10.22) que dispone de un servidor web.
- RED EXTERNA abarca el intervalo 193.147.87.0 - 193.147.87.255 y conecta la máquina cliente (interfaz enp0s3) y el cluster de alta disponibilidad.
- RED HEARTBEAT abarca el intervalo 10.10.10.0 - 10.10.10.255 y conecta entre si los dos nodos del clúster: (i) Servidor 1 (interfaz enp0s3), y (ii) Servidor 2 (interfaz enp0s3).

6. Pasos previos

Ajustar la configuración de las dos máquinas del cluster de balanceo (apache1 y apache2). Pasos a realizar:

- Editar los archivos del sitio web para incluir una indicación del servidor real que está sirviendo una petición, de modo que sea posible "diferenciarlos" en las pruebas manuales con el navegador (hacerlo en apache1 y apache2).

01	servidor1:~\$ sudo nano /var/www/html/index.html
02	...
03	Servido por APACHE_UNO
04	...

01	servidor1:~\$ sudo nano /var/www/html/index.html
02	...
03	Servido por APACHE_DOS
04	...

Notas:

- Este ajuste es simplemente una herramienta de depuración para verificar que el balanceador funciona correctamente.
- En una "granja" de servidores real este comportamiento no tendría sentido, dado que, obviamente, todos los nodos servirían el mismo contenido/aplicaciones (normalmente almacenado de forma compartido en una SAN o soluciones similares).
- Detener el demonio de Apache en ambas máquinas (Servidor 1 y Servidor 2).

01	servidor1:~\$ sudo systemctl stop apache2
----	---

01	servidor2:~\$ sudo systemctl stop apache2
----	---

7. Configuración de Corosync

Corosync se encarga de gestionar los nodos del cluster y su estado (up/down).

- Eliminar la configuración previa y detener los servicios Corosync y Pacemaker (en ambos nodos)

01	servidor1:~\$ sudo crm configure erase
02	servidor1:~\$ sudo systemctl stop pacemaker
03	servidor1:~\$ sudo systemctl stop corosync

01	servidor2:~\$ sudo crm configure erase
02	servidor2:~\$ sudo systemctl stop pacemaker
03	servidor2:~\$ sudo systemctl stop corosync

- Crear la clave compartida de autenticación de mensajes con el comando corosync-keygen (en cualquier nodo)

01	servidor1:~\$ sudo corosync-keygen
02	Corosync cluster Engine Authentication key generator.
03	Gathering 2048 bits for key from /dev/urandom.
04	Writing corosync key to /etc/corosync/authkey.

Notas:

- Todos los nodos del cluster deben de disponer de esta clave compartida
 - Por razones de seguridad, este fichero sólo debe de tener permiso de lectura para el usuario root (corosync-keygen ya lo crea con los permisos 400).
- Editar la configuración de Corosync (en cualquier nodo). Añadir las líneas que tienen la "->"

01	servidor1:~\$ sudo cp /etc/corosync/corosync.conf /etc/corosync/corosync.conf_bak
	...
	## Configuración de la comunicación entre nodos (knet sobre udp con pulsos/mensajes cifrados y autenticación
->	totem {
->	versión: 2

```
->     cluster_name: clustercda
->     transport: knet
->     knet_transport: udp
->     crypto_cipher: aes256
->     crypto_hash: sha256
->     interface {
->         linknumber: 0
->         bindnetaddr: 10.10.10.0
->         mcastport: 5405
->     }
-> }
## Esquema de quórum (con 2 nodos no tiene sentido y la opción
two_node:1 lo deshabilita)
-> quorum {
->     provider: corosync_votequorum
->     expected_votes: 2
->     two_node: 1
-> }

## Lista explícita de nodos que forman el cluster HA
-> nodelist {
->     node {
->         nodeid: 1
->         name: servidor1
->         ring0_addr: 10.10.10.11
->     }

->     node {
->         nodeid: 2
->         name: servidor2
->         ring0_addr: 10.10.10.22
->     }
-> }

-> logging {
->     to_logfile: yes
->     logfile: /var/log/corosync/corosync.log
->     to_syslog: yes
->     timestamp: on
-> }
```

- Copiar la configuración a los demás nodos del cluster (se hace mediante SSH)

```
01 servidor1:~$ sudo scp /etc/corosync/corosync.conf
```

Centros de Datos

David Ruano Ordás
Departamento de Informática
Lenguajes y Sistemas Informáticos

CURSO 2024-2025

02	root@servidor2:/etc/corosync/corosync.conf
03	
04	servidor1:~\$ sudo scp /etc/corosync/authkey
05	root@servidor2:/etc/corosync/authkey

Notas:

- Comprobar en el directorio /etc/corosync de la máquina servidor2 que realmente se han copiado los 2 ficheros (corosync.conf, authkeys) con los permisos correctos.
- Arrancar los servicios Corosync y Pacemaker en todas las máquinas del cluster

01	servidor1:~\$ sudo systemctl start corosync
02	servidor1:~\$ sudo systemctl start pacemaker

01	servidor2:~\$ sudo systemctl start corosync
02	servidor2:~\$ sudo systemctl start pacemaker

- Monitorizar cómo se "suman" nodos al cluster con el comando `crm_mon` (desde cualquier nodo)

01	servidor1:~\$ sudo crm_mon
----	----------------------------

01	servidor2:~\$ sudo crm_mon
----	----------------------------

Notas:

- Para finalizar la monitorización hay que usar `CONTROL+C`
- Es posible que se muestran nodos adicionales en estado offline. Se trata de "restos" de la configuración inicial de corosync existente en la imagen base de las máquinas virtuales.
- Para el desarrollo de la práctica lo más cómodo es dejar el comando `crm_mon` ejecutándose en una ventana propia y así poder comprobar la evolución y el estado del cluster en cualquier momento.
- ¿Qué ha pasado en los nodos del cluster al iniciar el demonio corosync en todos ellos?

8. Configuración de Pacemaker

Centros de Datos

David Ruano Ordás
Departamento de Informática
Lenguajes y Sistemas Informáticos

CURSO 2024-2025

Pacemaker gestiona los recursos (servicios del cluster) y su asignación a los nodos. En este ejemplo Pacemaker gestionará 2 recursos en modo activo-pasivo:

- La dirección IP pública 193.147.87.47 (recurso DIR_PUBLICA), este tipo de direcciones IP compartidas entre nodos de un cluster HA se denominan "IPs flotantes" (floating IP)
- Un servidor web Apache (recurso APACHE)

La consola de administración de Pacemaker (comando `crm`) tiene dos modos de uso (ambos equivalentes).

- Modo comando: especificando la secuencia opciones en una única orden de línea de comandos (útil para escribir scripts de automatización)
- Modo interactivo: navegando a través de los contextos del `crm shell` (con la misma secuencia de opciones que el modo comando)

Pasos a realizar:

- Entrar en la consola de configuración de Pacemaker [`crm shell`] (permite TAB para autocompletar)

```
01 servidor1:~$ sudo crm configure
02 crm(live) configure# show
03 crm(live) configure# show xml
```

Notas:

- En "modo comando" se ejecutaría desde el intérprete de comandos del sistema el comando `crm configure show`.
 - La configuración de Pacemaker reside en un documento XML, el CIB (cluster Information Base) [ubicado en `/var/lib/pacemaker/cib/cib.xml`]
 - La consola `crm shell` permite editar las entradas de ese fichero (se confirma la escritura de las modificaciones de parámetros con el comando `commit`).
- Ajustar parámetros (deshabilitar STONITH y ajustar QUORUM)

```
01 crm(live) configure# property stonith-enabled=false
02 crm(live) configure# property no-quorum-policy=ignore
03 crm(live) configure# commit
04 crm(live) configure# show
```

Notas:

- STONITH: mecanismo para "matar" nodos fallidos de modo que no

entren en competencia con los nodos que los reemplazan (evita inconsistencia de datos cuando dos componentes del cluster pretenden realizar las mismas tareas)

- QUÓRUM: mecanismo para asegurar una "votación representativa" a la hora de determinar las acciones a realizar cuando hay conflicto entre varios nodos ("gana" la mayoría), exigiendo un número mínimo de nodos "vivos" para poder tomar decisiones. En nuestro caso, con sólo 2 nodos, nunca habrá quórum (por eso se deshabilita, para que se ignoren esos "no acuerdos")

9. Ejercicios

Tarea 1 : Añadir el recurso IPaddr y verificar el comportamiento de LinuxHA

La tarea consiste en monitorizar el funcionamiento y comportamiento de LinuxHA ante fallos en los sistemas que forman parte del clúster de alta disponibilidad.

Pasos a realizar:

- Desde la máquina cliente lanzar el comando ping a la dirección IP 193.147.87.47 (fallará hasta que el cluster habilite dicha dirección)

01	cliente:~/\$ ping 193.147.87.47
----	---------------------------------

- Revisar los parámetros del "resource agent" IPaddr

01	crm(live) configure# ra
02	crm(live) configure ra# list ocf

Notas:

- Muestra los "agentes de recurso" Heartbeat/Pacemaker de tipo OCF disponibles [los scripts fueron instalados con 'apt-get install resource-agent' en /usr/lib/ocf/resource.d/heartbeat/]

01	crm(live) configure ra# list lsb
----	----------------------------------

Notas:

- Muestra los "agentes de recurso" correspondientes a scripts de arranque de tipo init [scripts en /etc/init.d, controlados con 'service <script> start|stop|restart']

Centros de Datos

David Ruano Ordás
Departamento de Informática
Lenguajes y Sistemas Informáticos

CURSO 2024-2025

01	crm(live) configure ra# list systemd
----	--------------------------------------

Notas:

- Muestra los "agentes de recurso" correspondiente a servicios del sistema gestionados por systemd [controlados con 'systemctl start|stop|restart <servicio>']

01	crm(live) configure ra# info ocf:heartbeat:IPaddr
02	crm(live) configure ra# up

Notas:

- Los "agentes de recurso" gestionan el arranque/parada y monitorización de los recursos.
- Ubicación: /usr/lib/ocf/resource.d/heartbeat (recursos Open cluster Framework (OCF) de Heartbeat/Pacemaker)
- Dar de alta el recurso IPaddr y configurarlo con la IP pública del servidor web y el interfaz de red a usar (ojo con el separador de líneas \). Regreso al terminal con letra Q.

01	crm(live) configure# help primitive
02	configure# primitive DIR_PUBLICA ocf:heartbeat:IPaddr \
03	params ip=193.147.87.47 \
04	cidr_netmask=255.255.255.0 nic=enp0s3
05	crm(live) configure# commit
06	crm(live) configure# show

Notas:

- Comprobar el ping desde cliente (en algún momento empezará a responder)
- Comprobar con "crm status" o "crm_mon" a qué nodo se le ha asignado el recurso DIR_PUBLICA
- En esa máquina ver la configuración de las tarjetas de red con "ip address" (habrá vinculado a la tarjeta enp0s3 la dirección 193.147.87.47)
- ¿Qué ha pasado en los nodos del cluster al hacer commit y declarar el recurso DIR_PUBLICA?

Tarea 2: Añadir el recurso APACHE y verificar el comportamiento de LinuxHA

La tarea consiste en definir un "recurso" que se corresponde con la ejecución de un servidor HTTP Apache. Este recurso se vinculará con la dirección IP flotante definida en el paso anterior de modo que ambos recursos se asignen al mismo nodo, responsable de ser el "portador" de la dirección flotante accesible desde el exterior y de iniciar el servicio Apache.

Pasos a realizar

- Revisar los parámetros del "resource agent APACHE

```
01 crm(live) configure# ra list ocf
02 crm(live) configure# ra info ocf:heartbeat:apache
```

- Darlo de alta y configurarlo (si se pone todo en una línea, quitar el \)

```
01 crm(live) configure# primitive APACHE ocf:heartbeat:apache \
02     params configfile=/etc/apache2/apache2.conf \
03 crm(live) configure# commit
04 crm(live) configure# show
```

- Desde otro terminal (o desde el otro nodo): comprobar cómo evoluciona el estado del cluster (comando "crm_mon" ó "crm status")
- Previsiblemente sucederá que el recurso DIR_PUBLICA se asigne a un nodo y el recurso APACHE al otro.
- Vincular los recursos DIR_PUBLICA y APACHE ("co-localizar" ambos recursos) para que se asignen al mismo nodo

```
01 crm(live) configure# help colocation
02 crm(live) configure# colocation APACHE_SOBRE_DIRPUBLICA inf:
03 DIR_PUBLICA APACHE
04 crm(live) configure# commit
05 crm(live) configure# show
```

Notas:

- Con inf: se asigna un score infinito positivo, que indica que es obligatorio que ambos recursos están juntos en el mismo nodo (del mismo modo, un score -inf: significara que ambos recursos, si es posible, deben asignarse a diferentes nodos) (valores intermedios del score cuantifican el grado de vinculación de los recursos).

- Comprobar cómo evoluciona el estado del cluster con el comando "crm_mon" hasta que se estabilice y los dos recursos se asignen al mismo nodo.
- Cuando los dos recursos migren al mismo nodo, comprobar que ahora es posible el acceso al servidor web desde la máquina cliente con lynx o Firefox (empleando la dirección "flotante" 193.147.87.47)
- ¿Que ha pasado en los nodos del cluster al hacer commit e incluir la restricción de "co-localizacion" APACHE_SOBRE_DIRPUBLICA?

Tarea 3: Forzar la migración de los recursos a otra máquina

La tarea consiste en forzar manualmente la migración de un nodo del cluster HA a otro para verificar cómo se inicia el servicio en el otro nodo del cluster con el fin de garantizar la disponibilidad del mismo.

Pasos a realizar:

- La migración se puede realizar usando el modo interactivo o el modo de comandos (usar una de las dos).
 - Modo interactivo

01	crm(live)configure# up
02	crm(live)# resource
03	crm(live)resource# help move
04	crm(live)resource# move APACHE servidorX

- Modo comandos (recomendable)

01	servidor1:~# crm resource move APACHE servidorX
02	servidor1:~# crm status

Notas:

- Hay que sustituir servidorX por el nombre del servidor al que se quiere mover el recurso. Es importante moverlo al servidor que no tiene asignado el recurso en ese momento.
- ¿Que ha pasado en los nodos del cluster al ejecutar el comando move del contexto resource?
- Detener la máquina donde se esté ejecutando (servidorX) [apagándola directamente] y comprobar que el otro servidor ocupa su lugar.

01	servidorY:~/\$sudo crm_mon #(o crm --show-detail)
02	#[esperar hasta que detecte el fallo y migre recursos]

Centros de Datos

David Ruano Ordás
Departamento de Informática
Lenguajes y Sistemas Informáticos

CURSO 2024-2025

03	#o
04	servidorY:~/\$sudo crm status

01	servidorX:~/\$sudo shutdown -h now
02	# MEJOR: apagar la MV desde el botón "Cerrar"

- Cuando termine la migración, comprobar el acceso al servidor web desde la máquina cliente con lynx o Falkon.
- ¿Que ha pasado en los nodos del cluster al apagar la máquina con los recursos DIR_PUBLICA y APACHE?
- Volver a "encender" la máquina apagada y comprobar que sucede.

Notas:

- En el chero de LOG /var/log/corosync/corosync.log (disponible en los dos nodos) se puede ver la evolución del cluster, los eventos y las decisiones tomadas por los nodos del cluster.

Tarea 4 : Entregable (memoria)

1. Formato: PDF
2. Nombre: Practica4_(apellidos_nombre).pdf
3. Describir:
 - Configuración del gestor de nodos Corosync.
 - Declaración del recurso DIR_PUBLICA
 - Declaración del recurso APACHE y establecimiento de la restricción de "co-localización".
 - Migración del recurso APACHE a otro nodo
 - Apagado/cada de un nodo
4. Indicar los comandos y las acciones que tienen lugar en los nodos del cluster (tráfico de la red, scripts ejecutados en los nodos implicados, etc), como consecuencia de dichos comandos, indicando su resultado.
5. Detallar el estado nal del cluster (basta con aportar la salida de crm status o crm_mon).

Centros de Datos

David Ruano Ordás
Departamento de Informática
Lenguajes y Sistemas Informáticos

CURSO 2024-2025