CÓMO Montar un Clúster Apache con Balanceo de Carga y Alta Disponibilidad

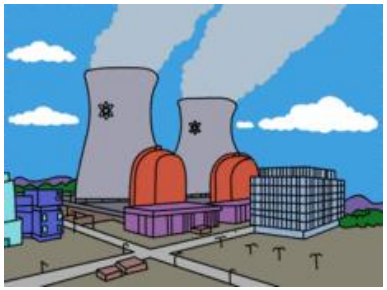
Miguel Álvarez Úbeda
Bruno López Lagares
Pedro Lorenzo Riveiros
Juan Santos García-Toriello
Marcelino Castañeda Sánchez
Fernanda García Alves de Oliveira

Estructura del CÓMO

- Bloque I – Instalar ROCKS
- Bloque II – IPVS
- Bloque III - HA

El PROYECTO…

- Motivación de montar un servidor apache con balanceo de carga y alta disponibilidad
  - Un servidor web suele alojar varias webs. Cuando la organización tiene un volumen de información a ser servida importante es muy probable que requiera una máquina en exclusiva, y que incluso tenga que ampliar la potencia del sistema
  - Una solución escalable, cost-effective, y de alto rendimiento y prestaciones es contar con un clúster de servidores apache.
  - Para manejar esta opción, manteniendo las sesiones, es necesario recurrir a soluciones como el balanceo de carga a través de IP.
  - La disponibilidad de la web debe ser percibida como algo continuo, sin paradas ni caídas, y fundamentalmente no debe “morir de éxito”.
  - Hay muchos entornos en que la disponibilidad es crítica (telemedicina)



El PROYECTO…

- En base a la motivación, los ingenieros del máster en Informática de la UDC han llevado a cabo:
  - Instalación de un clúster con ROCKS
  - Configuración de la solución de balanceo de carga con IPVS
  - Configuración de la solución de alta disponibildad

  ...todo ello en una jornada.

El PROYECTO…

- Todo ello bajo una muy leve orientación y supervisión del profesor.

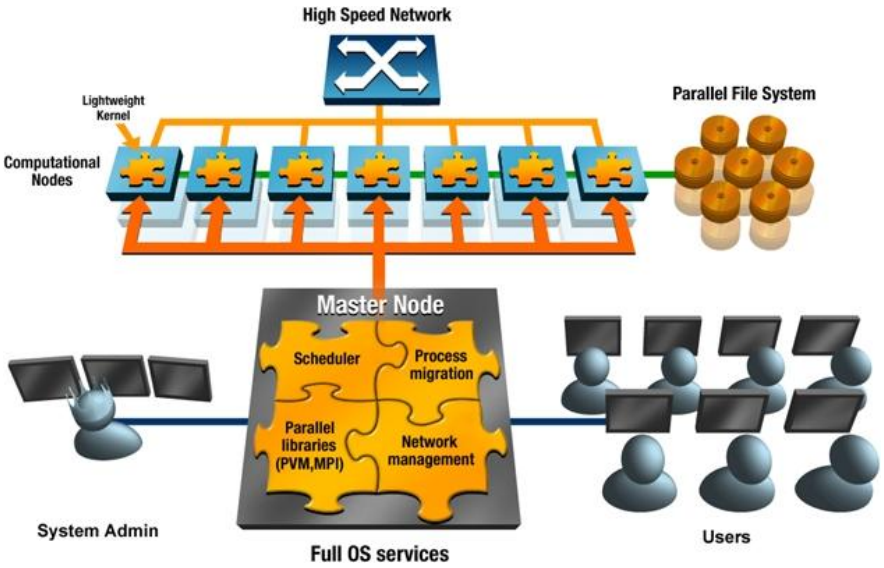
El PROYECTO…

- He aquí lo que ocurrió…

Introducción a las Arquitecturas Clúster

- Un clúster es ...
  - Un conjunto de máquinas interconectadas mediante una red de interconexión a los que un determinado software convierte en un sistema de mayores prestaciones
- Aunque esta definición peque de general o imprecisa explica en esencia lo que es un clúster

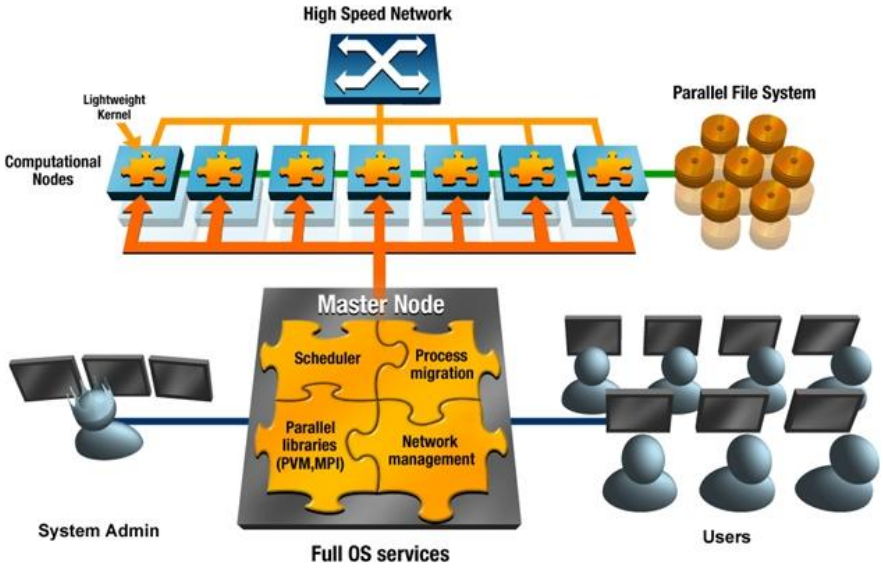
Elementos de un Clúster: HW



- Frontend, nodos de computación, red, sistema de almacenamiento

Elementos de un Clúster: HW

- Elementos adicionales:
  - SAI
  - KVM
  - Rack
  - …



Elementos de un Clúster: S.O.

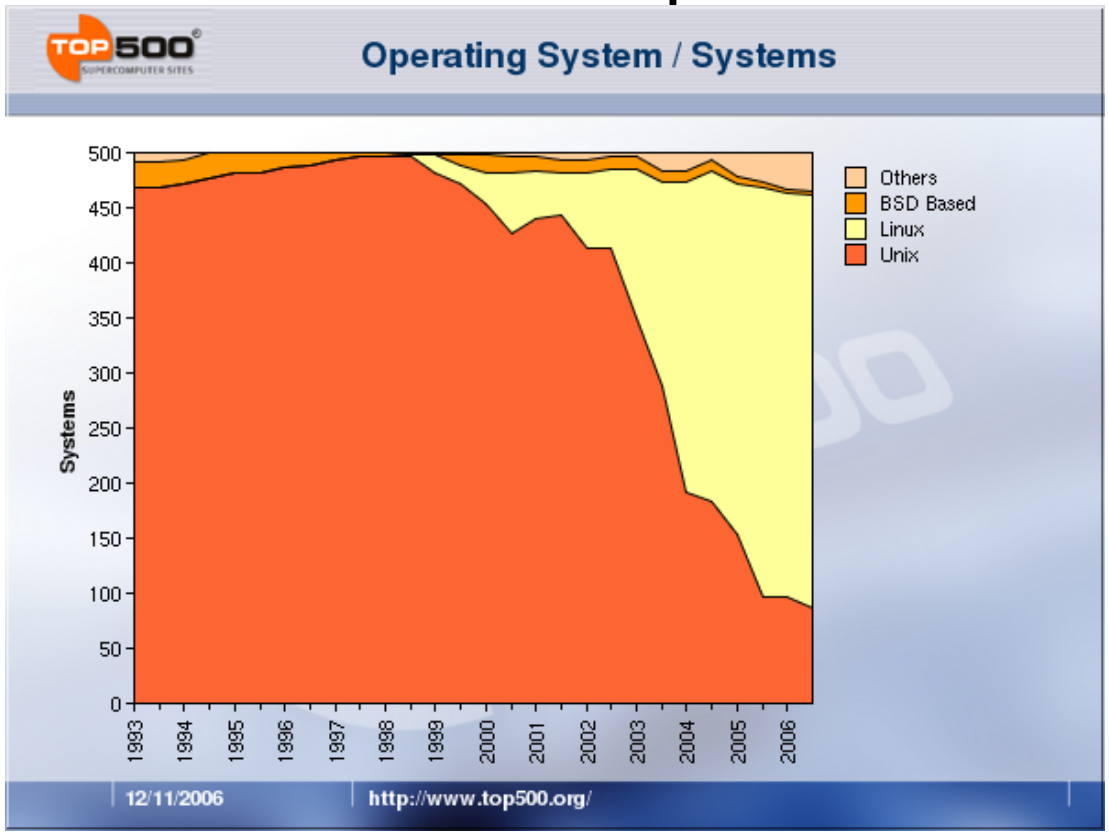
- El SO por excelencia es Linux:
  - Es gratuito. Disminuye el *Total Cost of Ownership* (TCO)
  - Obtiene mejores resultados que sistemas Win, especialmente en *networking* al formar parte del kernel, además de utilidades de calidad en modo texto
  - Existe una gran cantidad de software y una gran comunidad de desarrolladores y usuarios
  - Es tan seguro y fiable como un Unix y presenta numerosas facilidades de administración
  - Completo soporte multicore
- Esto ha llevado a que la inmensa mayoría de los clusters tengan este SO

Elementos de un Clúster: S.O.

- El SO Linux suele ser instalado con distribuciones, que son colecciones de software con el sistema base, programa de instalación y numerosas aplicaciones. Destacan:
  - **Red Hat.** Distribución muy popular. No tiene nuevas versiones desde RH 9
  - **Fedora**. Es la sustituta de RH especializada para desarrollo
  - **Red Hat Enterprise Linux**. Sustituye a RH especializándose en sistemas empresariales y en clusters
  - **CentOS** es la versión para la comunidad de RHEL.
  - **Slackware**. Distribución basada en RH, para computadores de escritorio
  - **Esware**. Distribución española basada en Slackware.
  - **Mandriva**. Distribución conocida por su facilidad de instalación.
  - **SUSE**. Distribución alemana con una importante cantidad de software, y con amplio soporte.
  - **Debian**. Distribución para usuarios avanzados y administradores. Es la más segura y con mayor cantidad de paquetes.
  - **Ubuntu**. Proyecto basado en Debian.
  - **Conectiva**. Distribución brasileña orientada a la empresa.
- Las distribuciones más utilizadas son RHEL y CentOS.

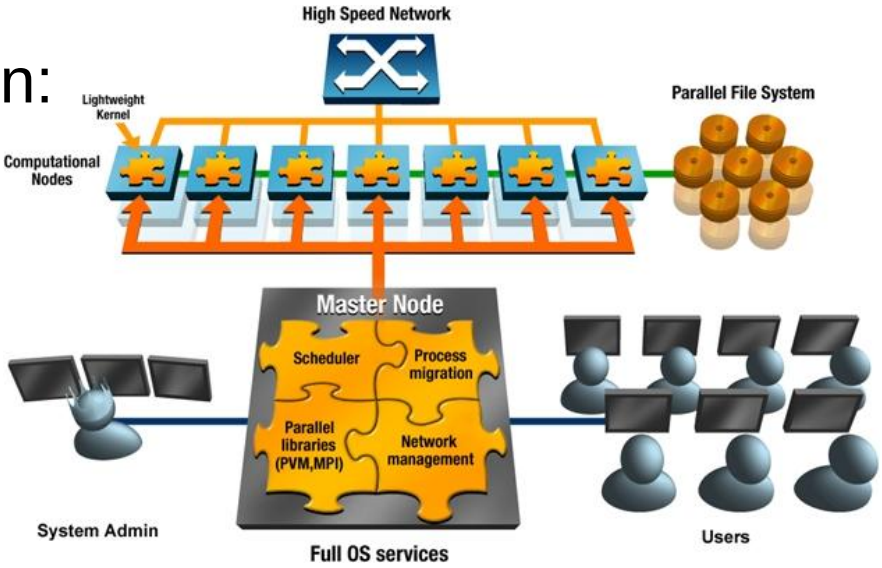
Evolución histórica de las Arquitecturas de Computadores

- Evolución del sistema operativo:



Elementos de un Clúster: Nodos de Computación

- Nodos de computación:

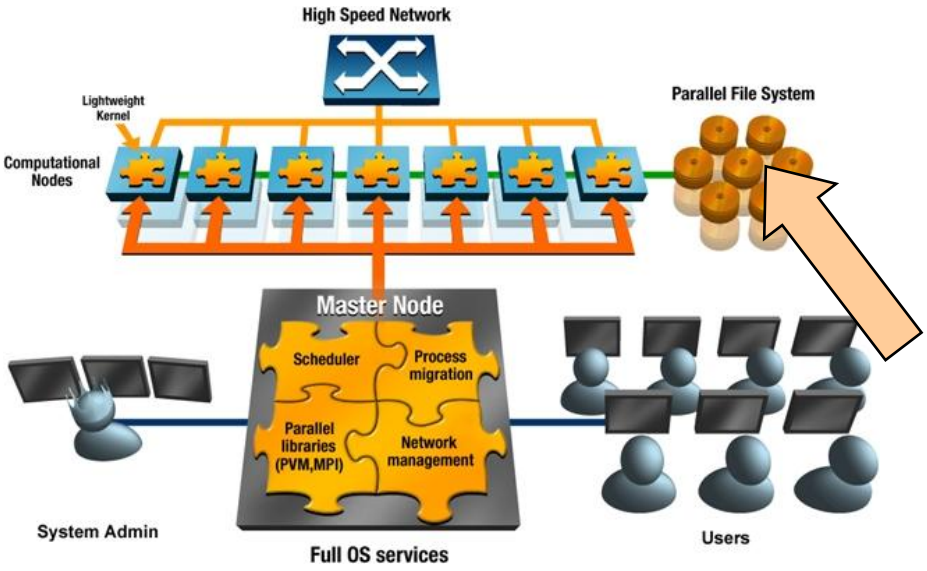


  - La capacidad de cómputo de un nodo, junto con la cantidad de memoria necesaria para ejecutar las aplicaciones, son lo fundamental en la unidad que va a determinar el rendimiento computacional del sistema. Los nodos disponen cada vez más de un mayor número de cores.

Elementos de un Clúster: Almacenamiento

- Almacenamiento:

Puede consistir en una NAS, una SAN, o alma-cenamiento interno en el servidor.



- El protocolo más comúnmente utilizado es NFS (Network File System), sistema de ficheros compartido entre servidor y los nodos.
- Sistemas de ficheros específicos para clusters son Lustre (CFS) y PVFS2.

Elementos de un Clúster: Almacenamiento

- NAS (Network Attached Storage) es un dispositivo específico dedicado a almacenamiento a través de red (normalmente TCP/IP) que hace uso de un S.O. optimizado para dar acceso a través de protocolos CIFS, NFS, FTP o TFTP.
  - CIFS Common Internet File System (anteriormente SMB, Service Message Block) es un protocolo de almacenamiento para compartir fichero en entornos Windows. Una implementación libre es SAMBA, permitiendo a máquina Windows acceder a almacenamiento en servidores Linux.
- Por su parte, DAS (Direct Attached Storage) consiste en conectar unidades externas de almacenamiento SCSI o a una SAN (Storage Area Network) a través de Fibre Channel. Estas conexiones son dedicadas.
- Mientras NAS permite compartir el almacenamiento, utilizar la red, y tiene una gestión más sencilla, DAS proporciona mayor rendimiento y mayor fiabilidad al no compartir el recurso.

Elementos de un Clúster: Almacenamiento

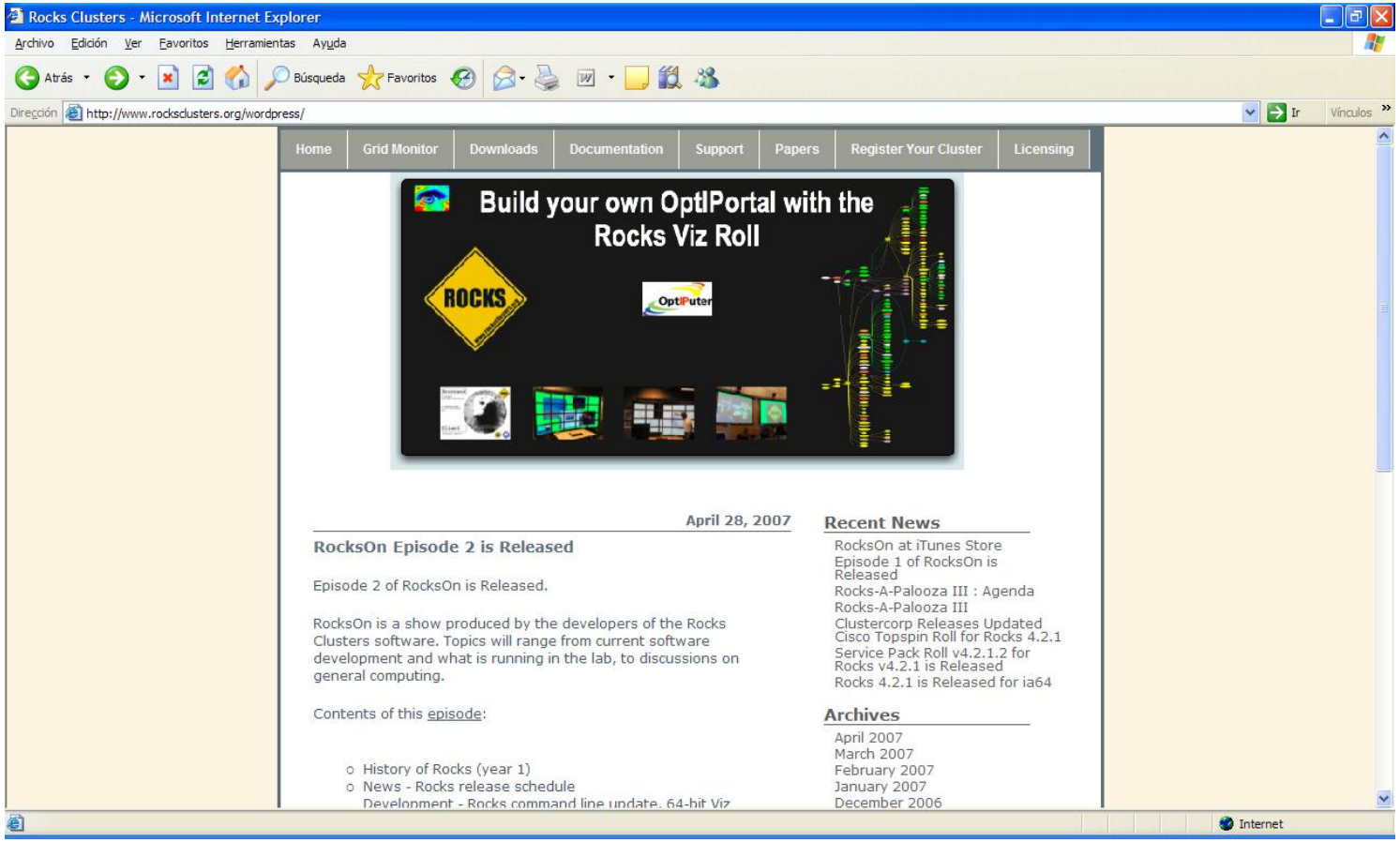
NAS vs. SAN:

- *NAS nos permite acceder a un sistema de ficheros a través de TCP/IP usando CIFS (en el caso de Windows) ó NFS (en el caso de Unix/Linux.*
- *NAS es una solución más adecuada como*
  - *Servidor de ficheros*
  - *Almacenamiento de directorios de usuario*
  - *Almacenamiento de archivos en general*

- *Por su parte, una SAN es usada para acceder a almacenamiento en modo BLOQUE, a través de una red de fibra óptica con Fibre Channel (FC) utilizando el protocolo SCSI.*
- *SAN es una solución más adecuada para:*
  - *Bases de datos*
  - *Data warehouse*
  - *Backup (al no interferir en la red del sistema)*
  - *Cualquier applicación que requiera baja latencia y alto ancho de banda en el almacenamiento y recuperación de datos*

Rocks Cluster

- Web: www.rocksclusters.org (ISO 4GB)



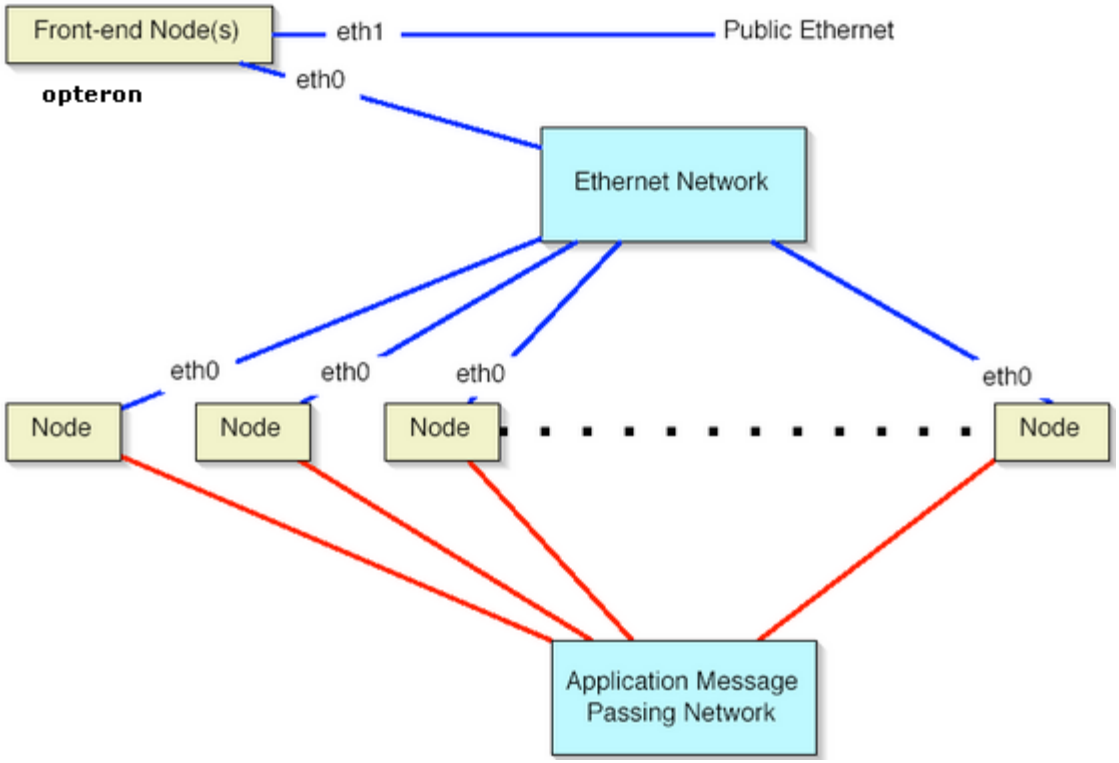
Rocks Cluster

- Sistema desarrollado por *San Diego Supercomputer Center* (SDSC), perteneciente a la Universidad de California, San Diego y *Nacional Partnership for Advanced Computational Infrastructure* (NPACI).
- Presenta una arquitectura modular basada en rolls (referido a grupos de pauqtes (ej: rol grid, rol java, rol security)).
- Define distintos tipos de nodos que pueden componer el cluster, siendo los principales
- *Frontend:* tipo de nodo servidor.
- *Compute:* dedicado a la ejecución de aplicaciones
- Cada roll, además de los paquetes, especifica en que tipo de nodo se instala y la configuración que se le hace a cada uno.

Rocks Cluster

- ROCKS: http://www.rocksclusters.org Estos recursos son:
  - Area 51 – securidad, tripwire y chkrootkit
  - BIO – aplicaciones de High Performance Computing en el ámbito de la genómica
  - Condor/maui – planificación/ejecución de aplicaciones.
  - Ganglia - monitorización de recursos
  - Grid – Globus toolkit
  - PVFS2 – Sistema de ficheros paralelo
  - Sun Grid Engine, Torque – gestores de colas de recursos
  - Viz – visualización (matriz de pantallas).
  - HPC utilities
  - 411 – gestión información (NIS++)
  - NFS – network file system
  - Java
  - Bibliotecas MPI: LAM/MPICH
  - Apache

Rocks Cluster



NODOS: compute-0-0 C0-0
SERVER: maquina.local

Instalación del Clúster

Rocks Cluster Distribution

What do you want to kickstart?

- Frontend:
  type "frontend"

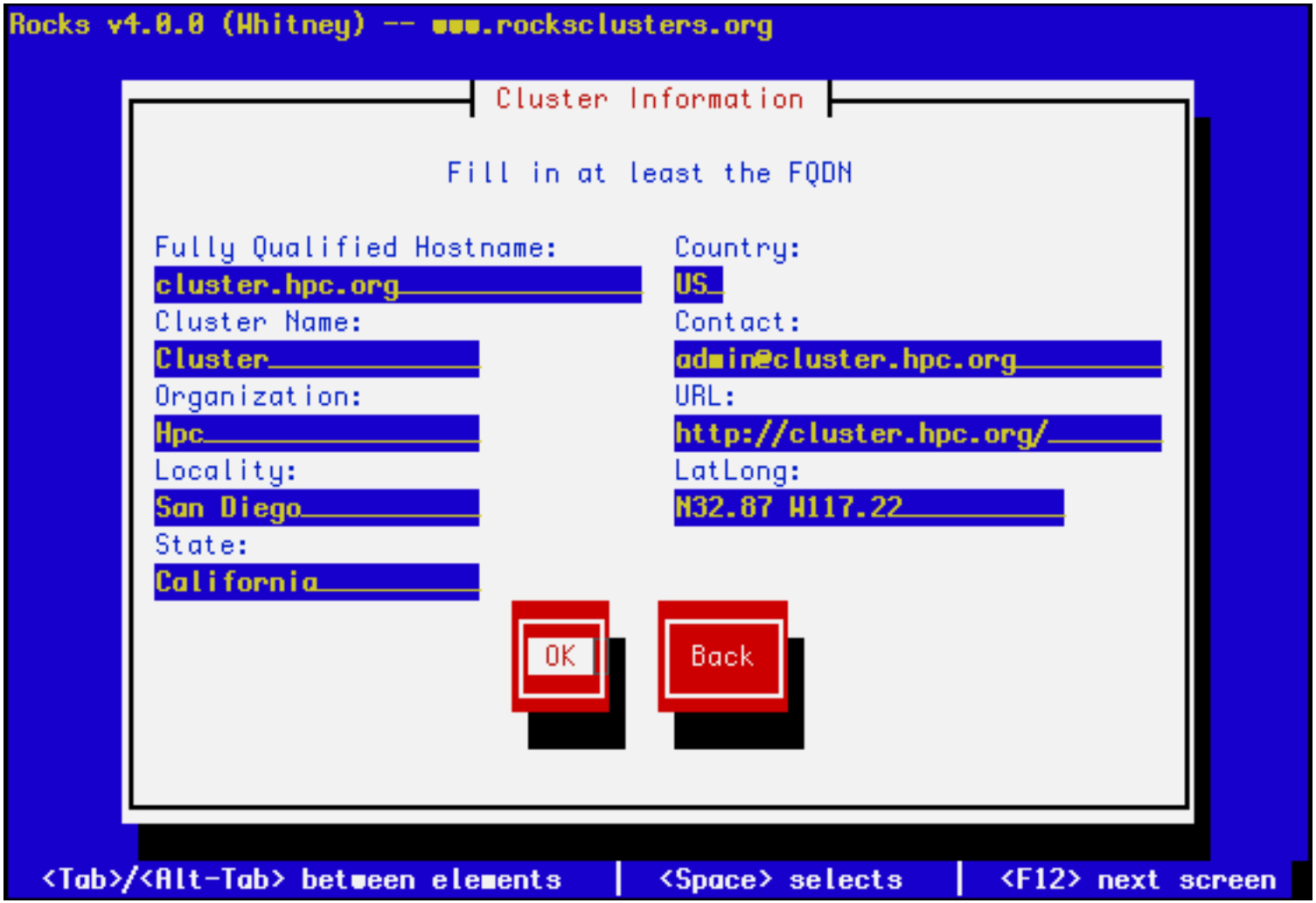
- Upgrade your frontend:
  type "frontend upgrade"

- Frontend Network Install
  type "frontend central=name"
  where name is "Rocks", or the
  FQDN of your central server.

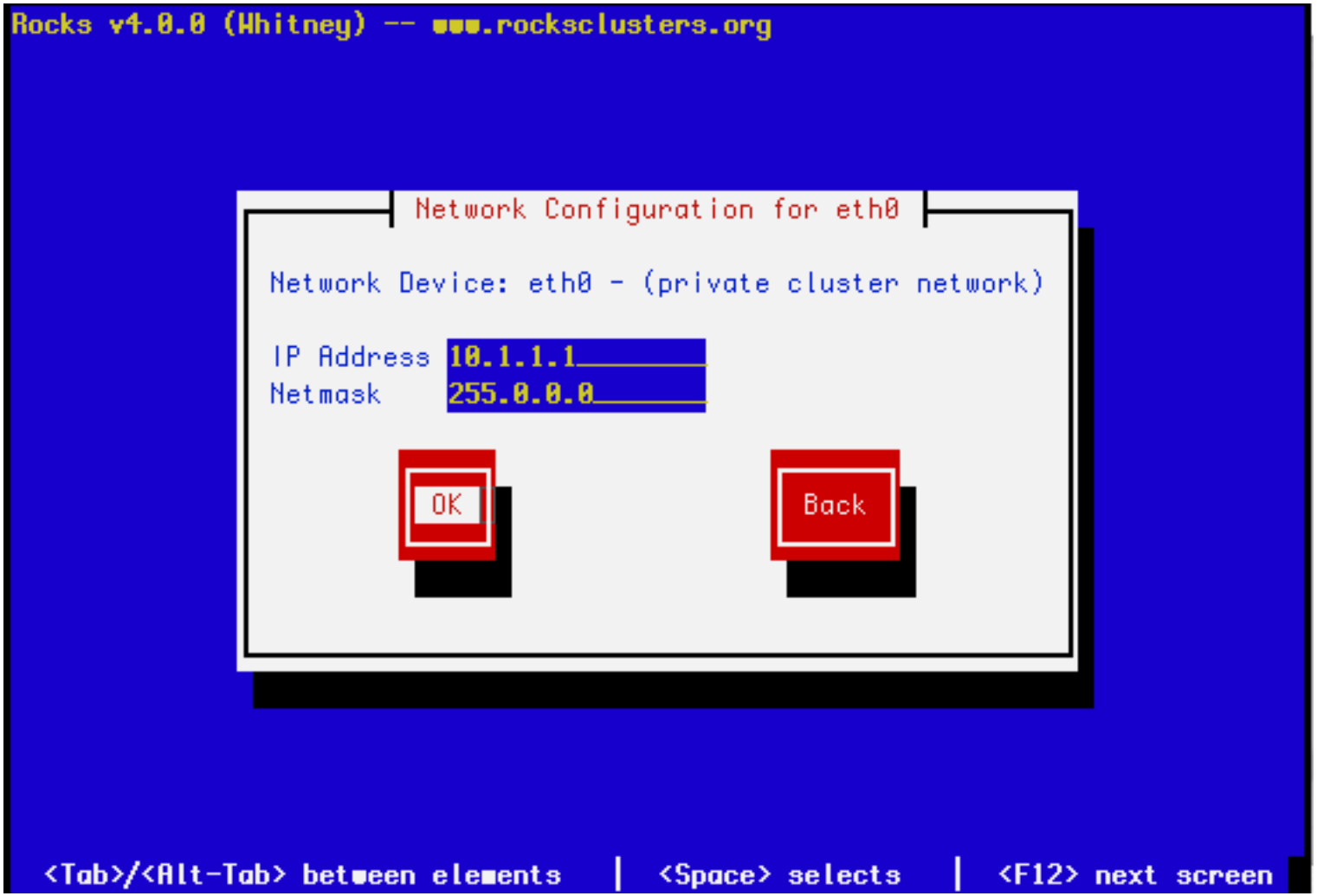
- Rescue
  type "frontend rescue"

- Cluster node:
  do nothing or press return

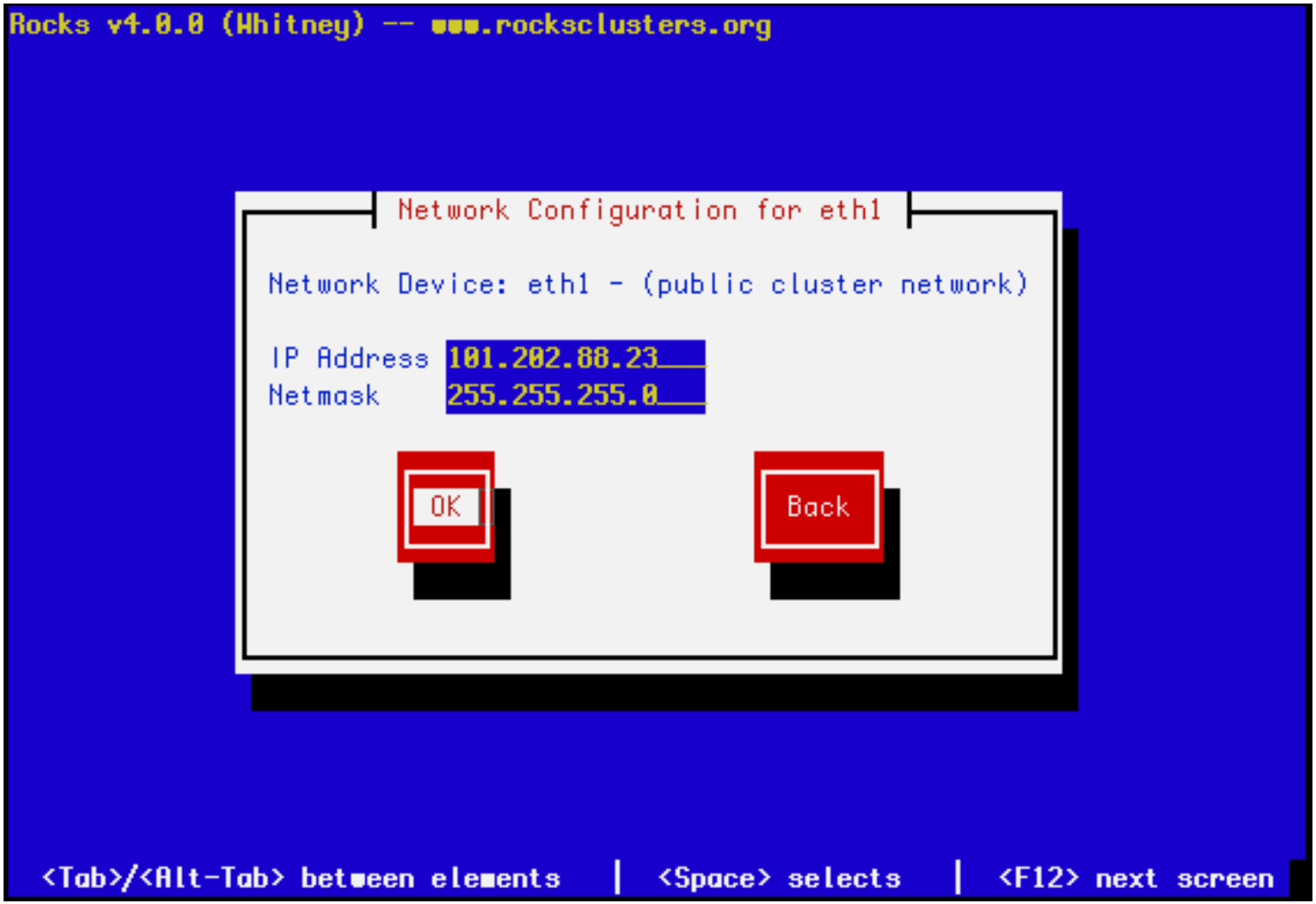
Instalación del Clúster



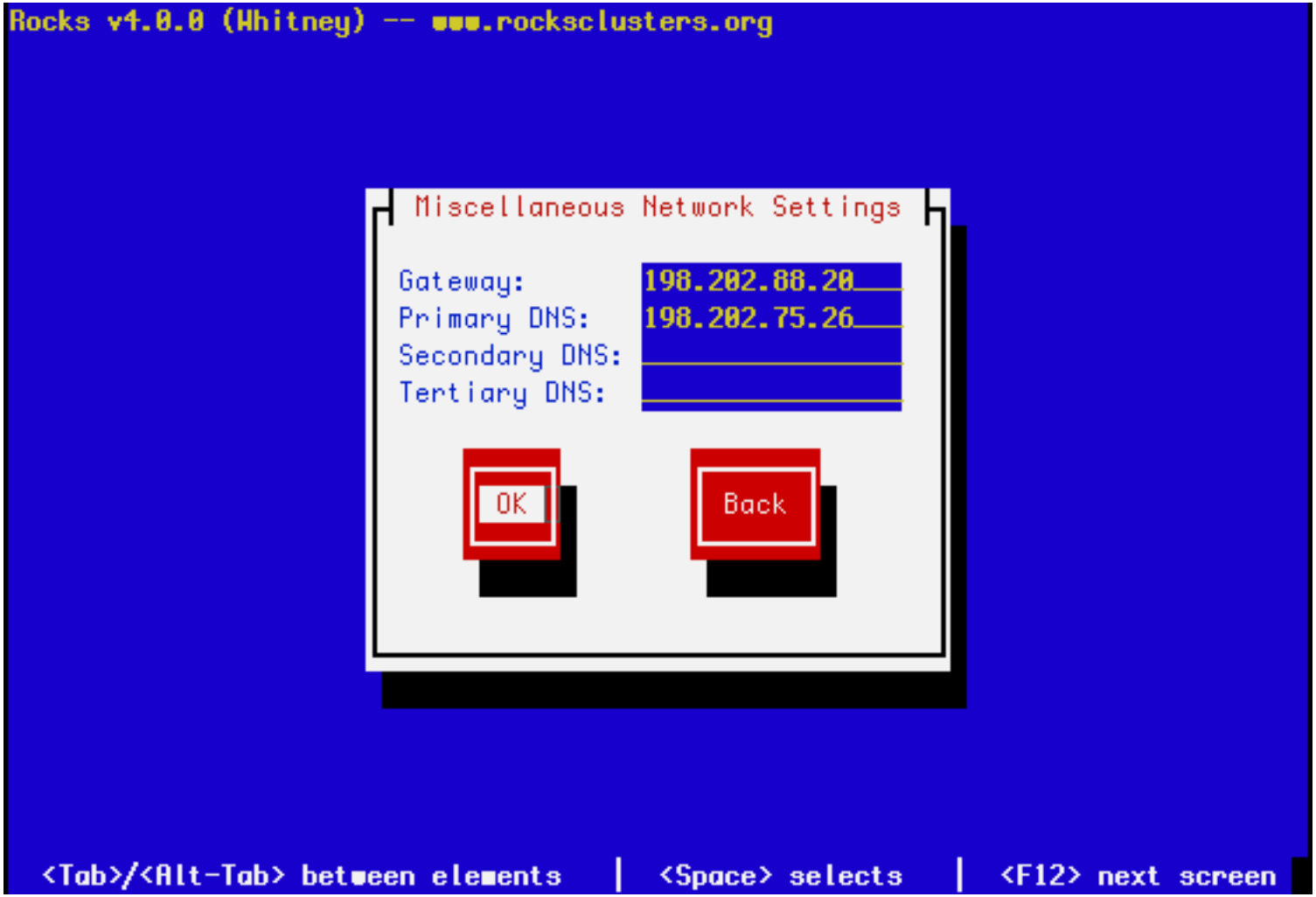
Instalación del Clúster



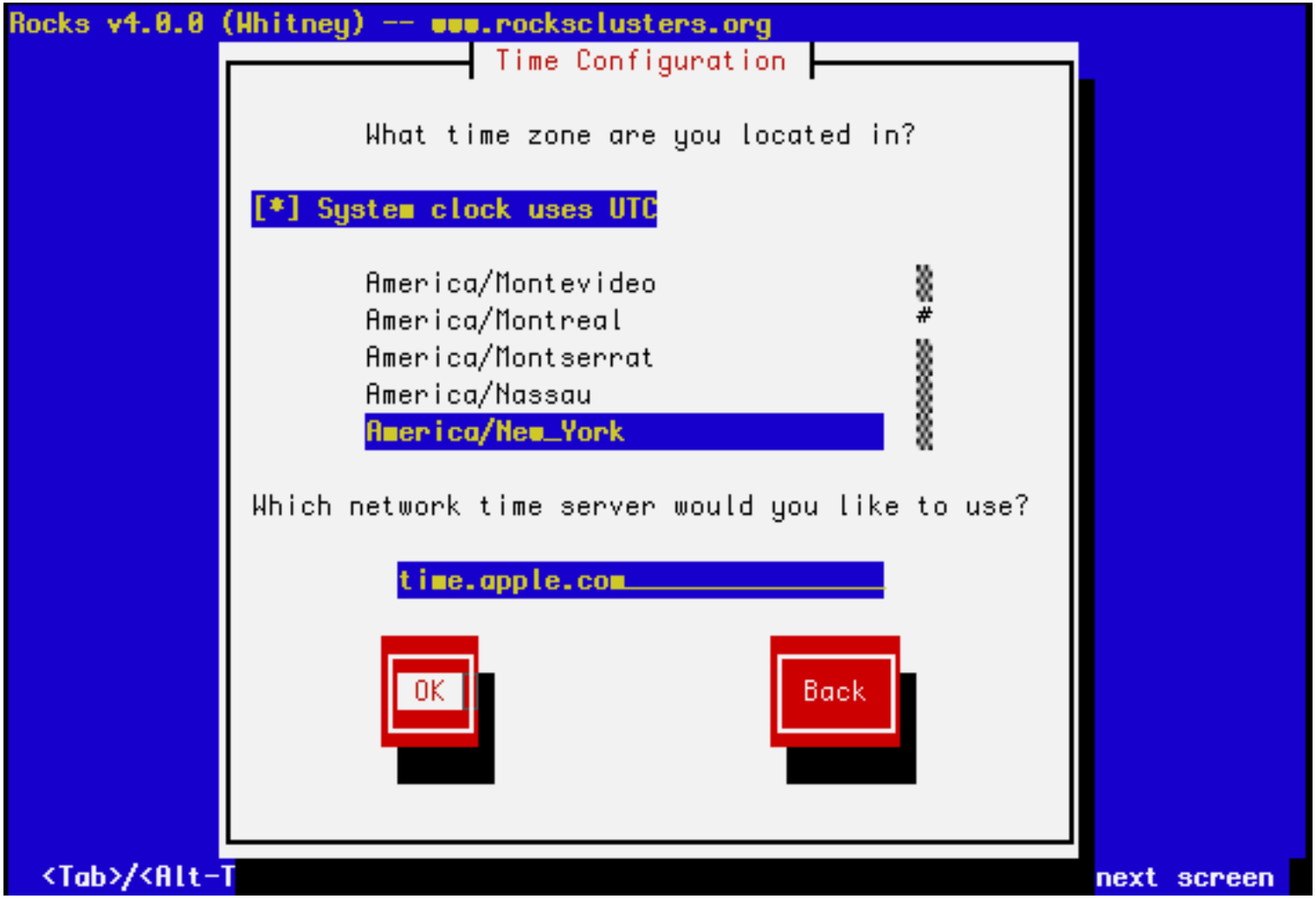
Instalación del Clúster



Instalación del Clúster



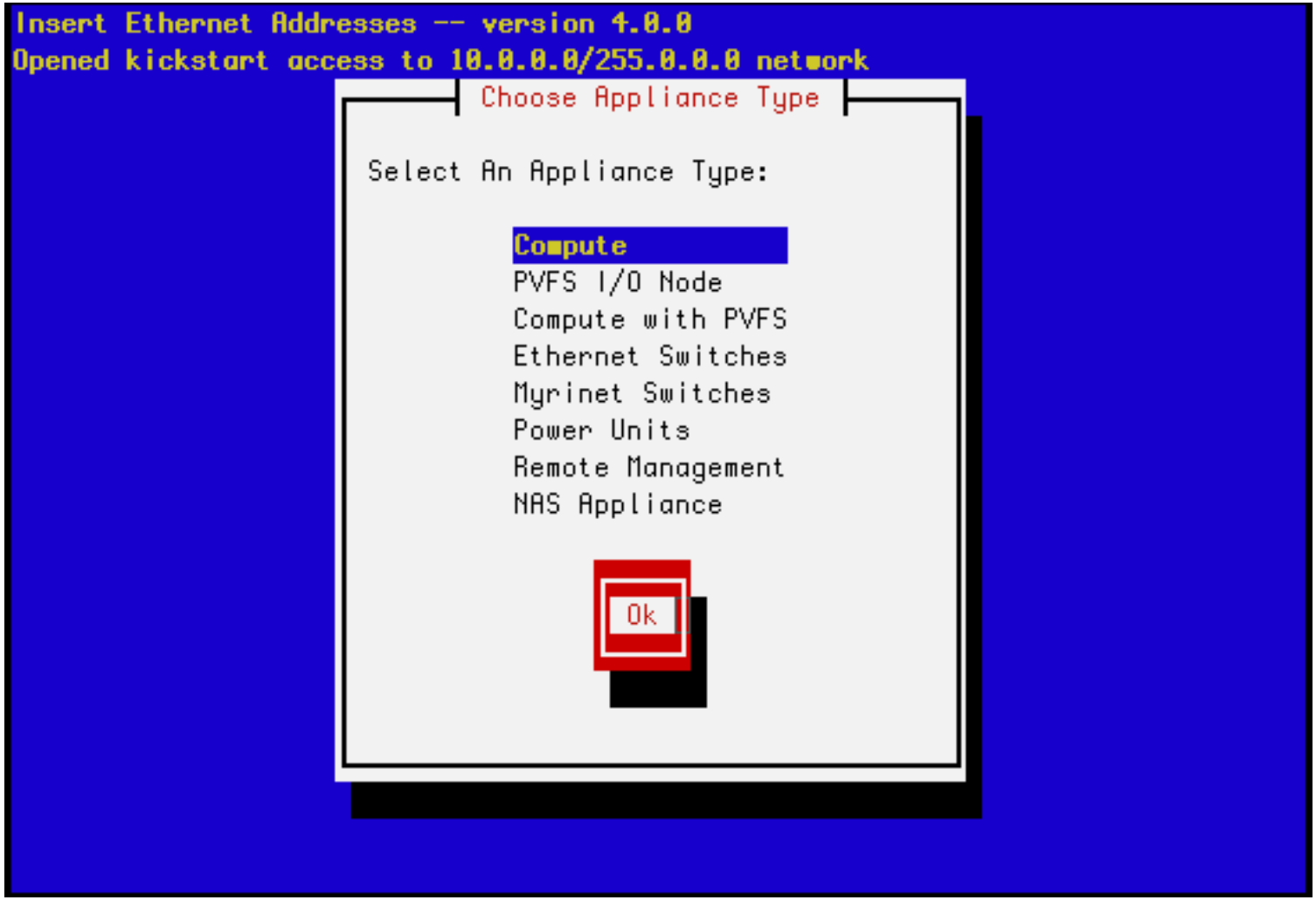
Instalación del Clúster



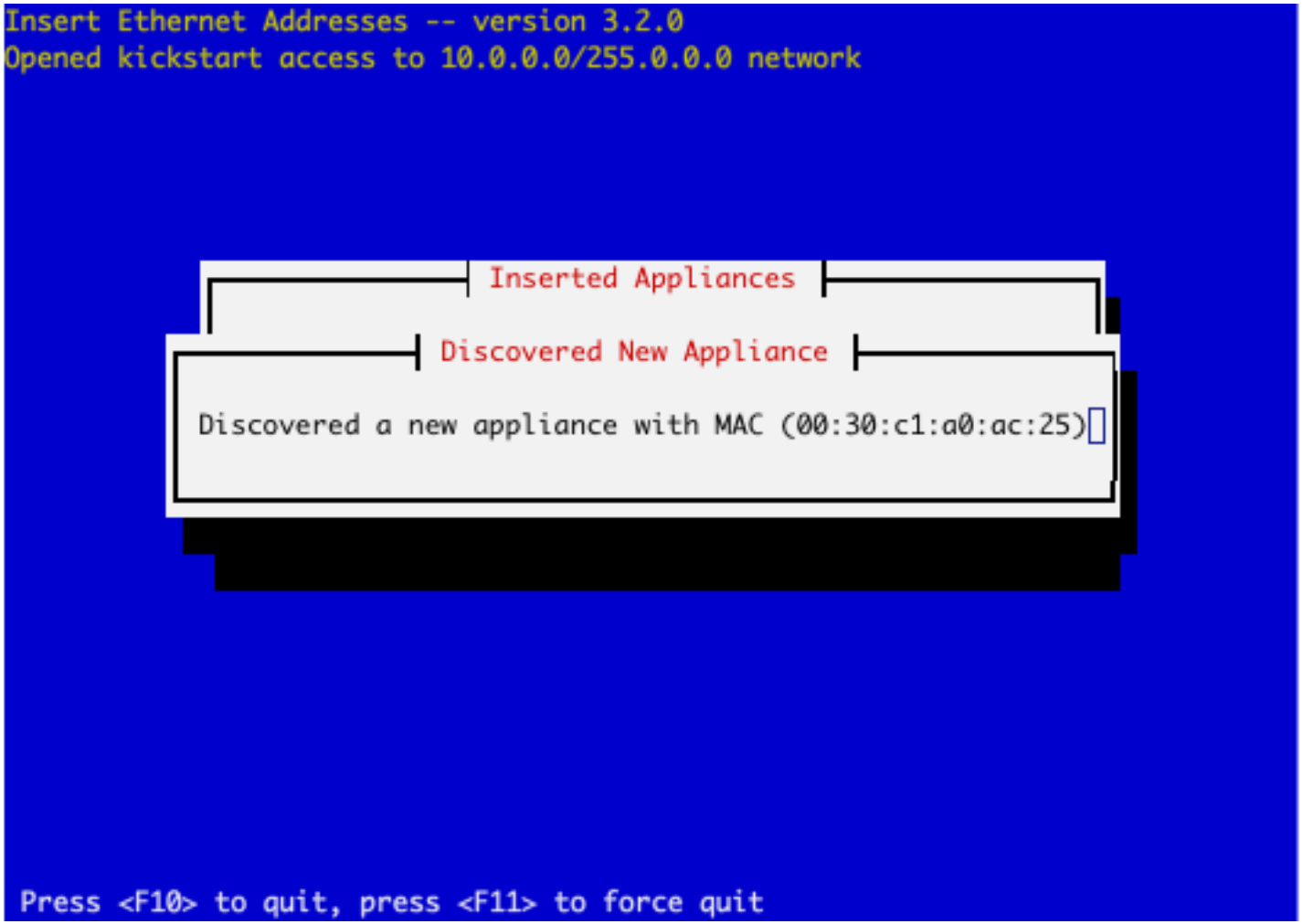
Instalación del Clúster

- # insert-ethers
  - – A través de DHCP recibe peticiones (datos incorporados a Rocks MySQL DB)
- # insert-ethers --cabinet=1 (especifica a traves de PXE/instalacion desatendida para todos los nodos del rack 1).
- PXE: Instalacion y arranque por red.

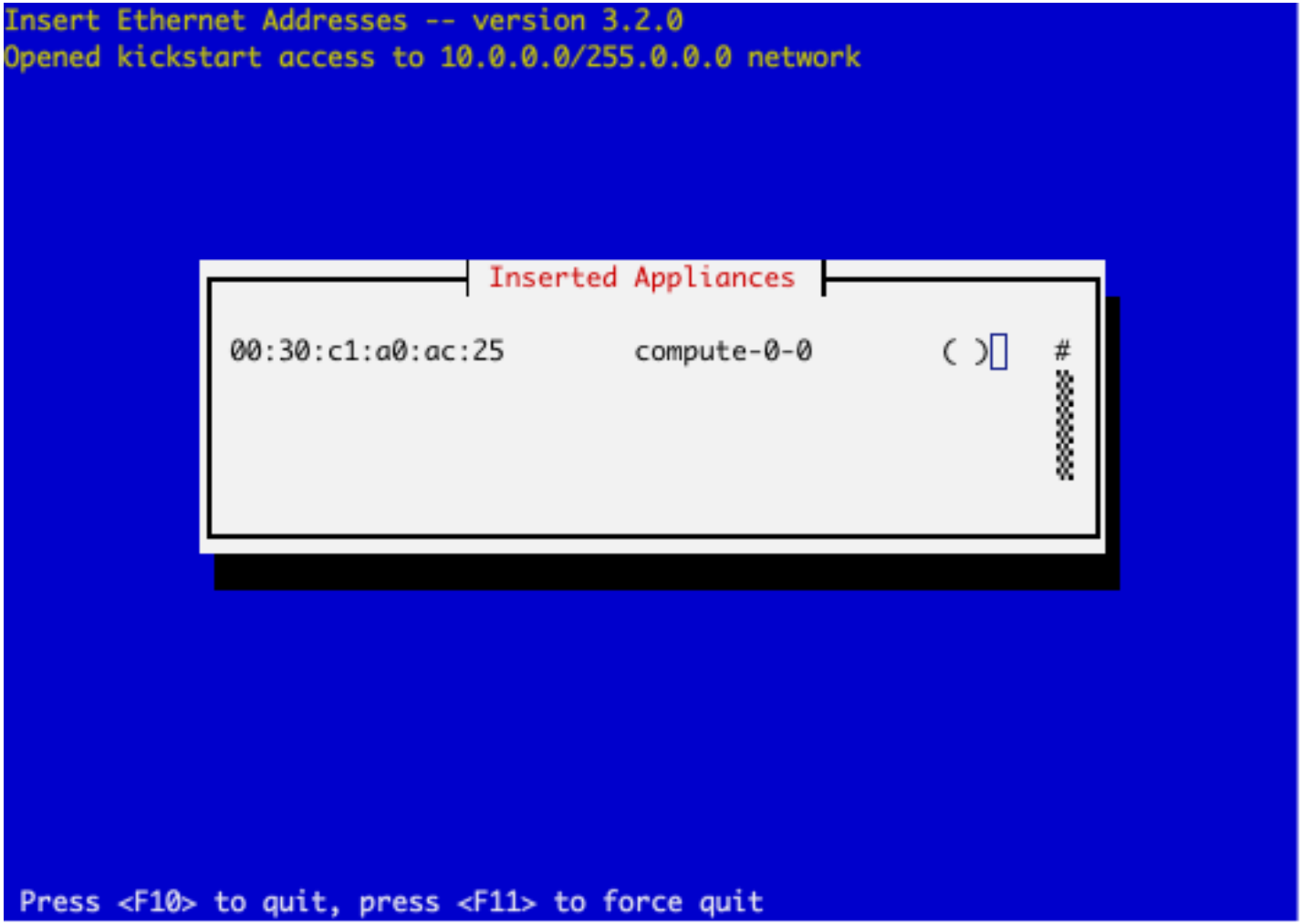
Instalación del Clúster



Instalación del Clúster

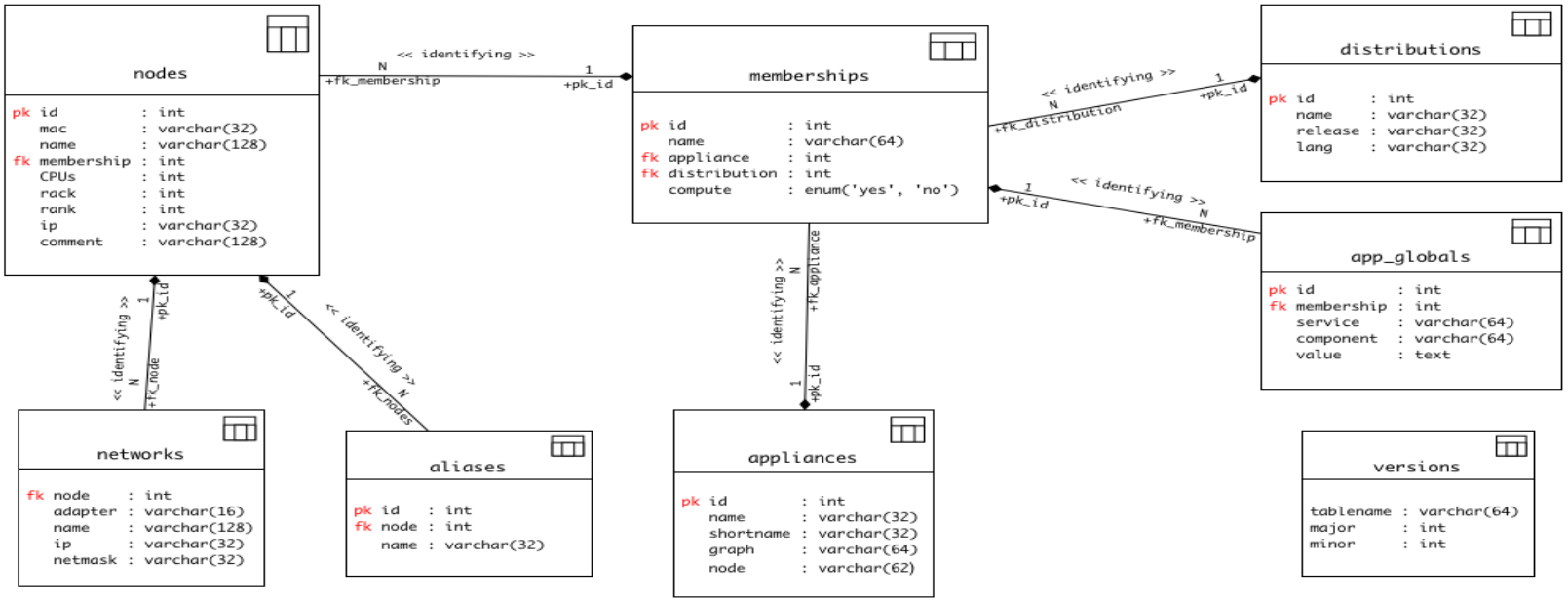


Instalación del Clúster

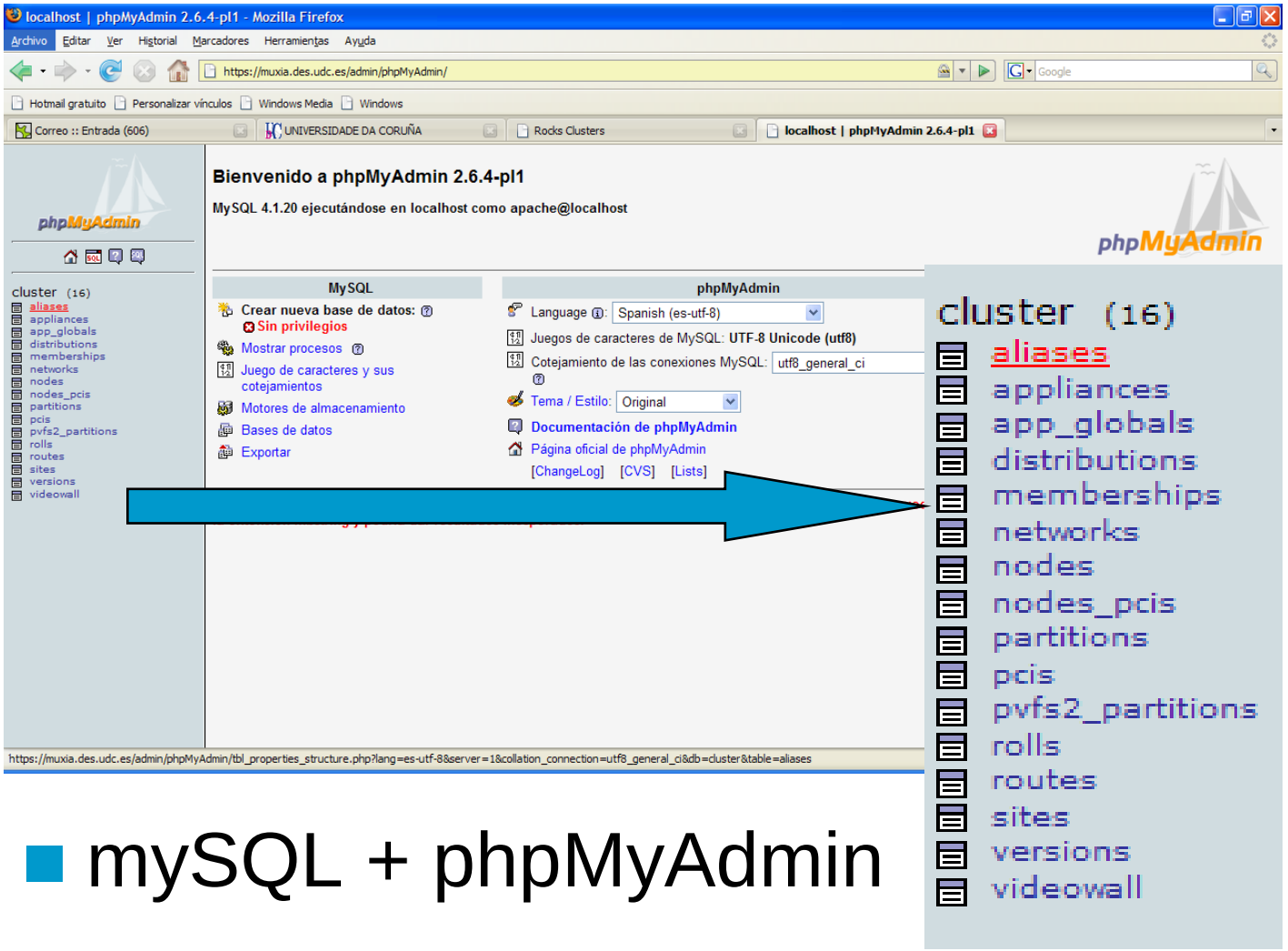


Instalación del Clúster

- Base de datos del sistema. Se configura a traves de una interfaz Web (PHP). Reflejo de los archivos de configuracion del sistema.



Instalación del Clúster



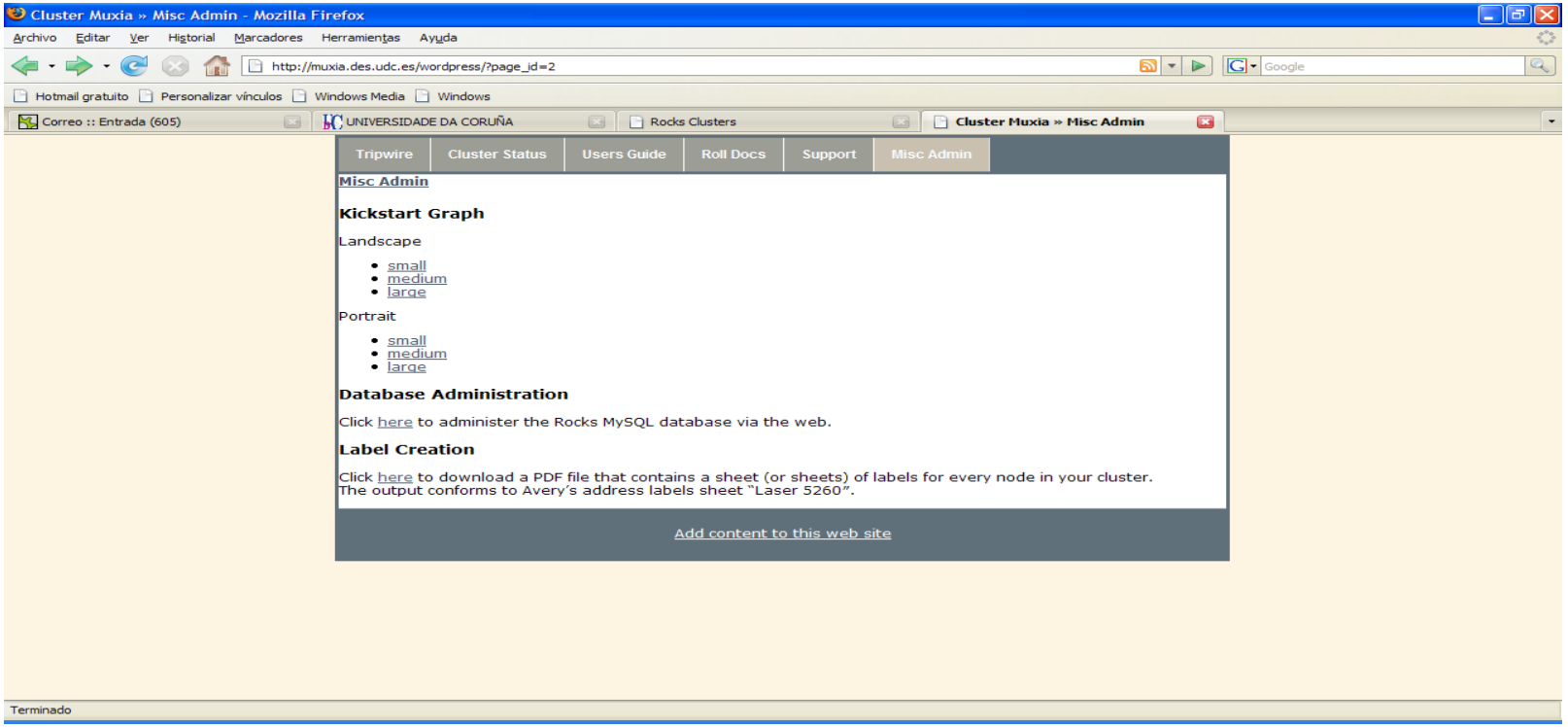
- mySQL + phpMyAdmin

Instalación del Clúster

- **$ cluster-fork ps -U$USER**
  - Ejecuta "ps" en todos los nodos
  - e.g.: cluster-fork ps -Udiego
- **$ cluster-ps PATTERN**
  - Ejecuta "ps -aux | grep PATTERN" en todos los nodos.
  - e.g.: cluster-ps MATLAB
- **$ cluster-fork [cmd]**
  - Ejecutar cualquier comando en todos los nodos del cluster cluster-fork
  - e.g.: cluster-fork cp -r MatLabToolBox/ tmp/
- **$ cluster-fork --query="select name from nodes where name like 'scompute-0-%'" [cmd]**
  - Ejecuta [cmd] en los nodos devueltos por la query SQL. El resultado de la query es una lista de nodos
- **$ cluster-fork --nodes="scompute-0-%d:1-4 scompute-0-%d:7,9,11,15-16" [cmd]**
  - Ejecuta [cmd] en los distintos nodos especificados por "--nodes".

Rocks Cluster

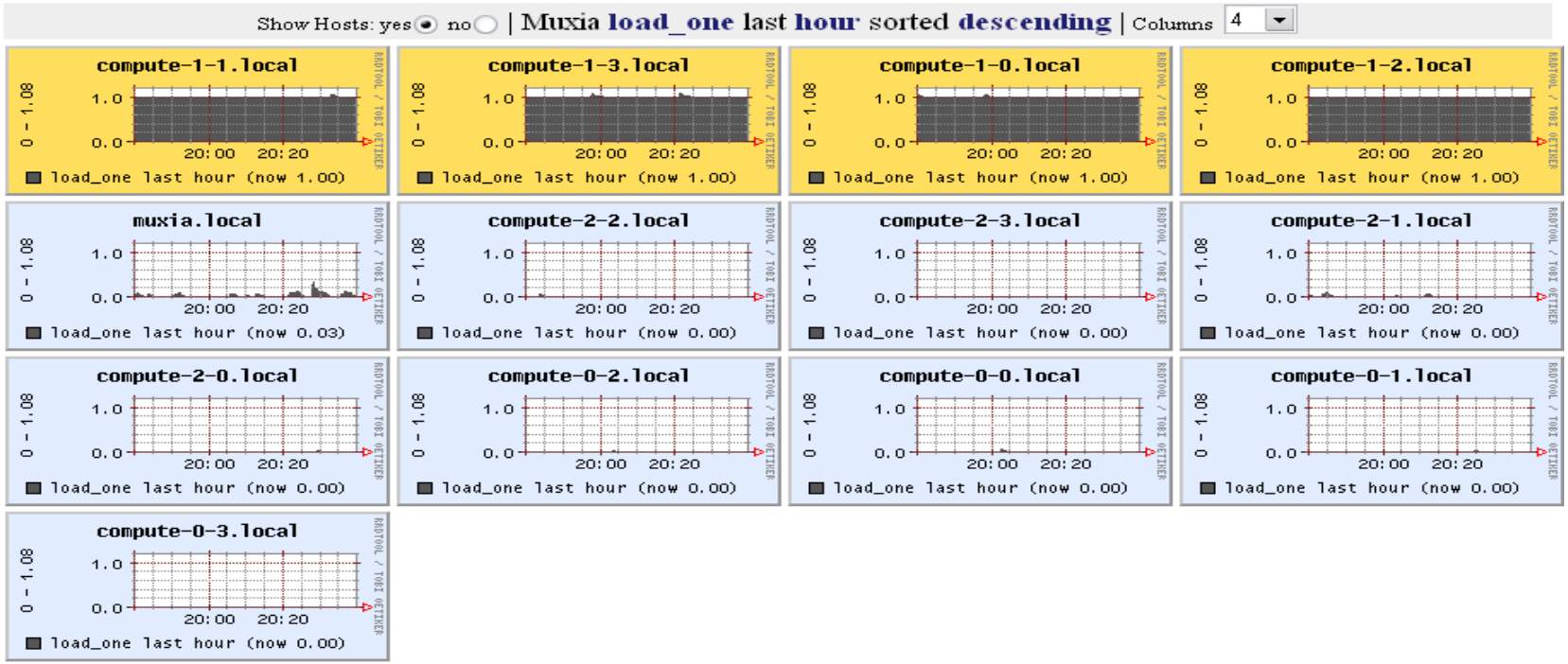
- Acceso a traves de Web de toda la documentacion de los rolls instalados. Tambien integridad de datos (Tripwire).



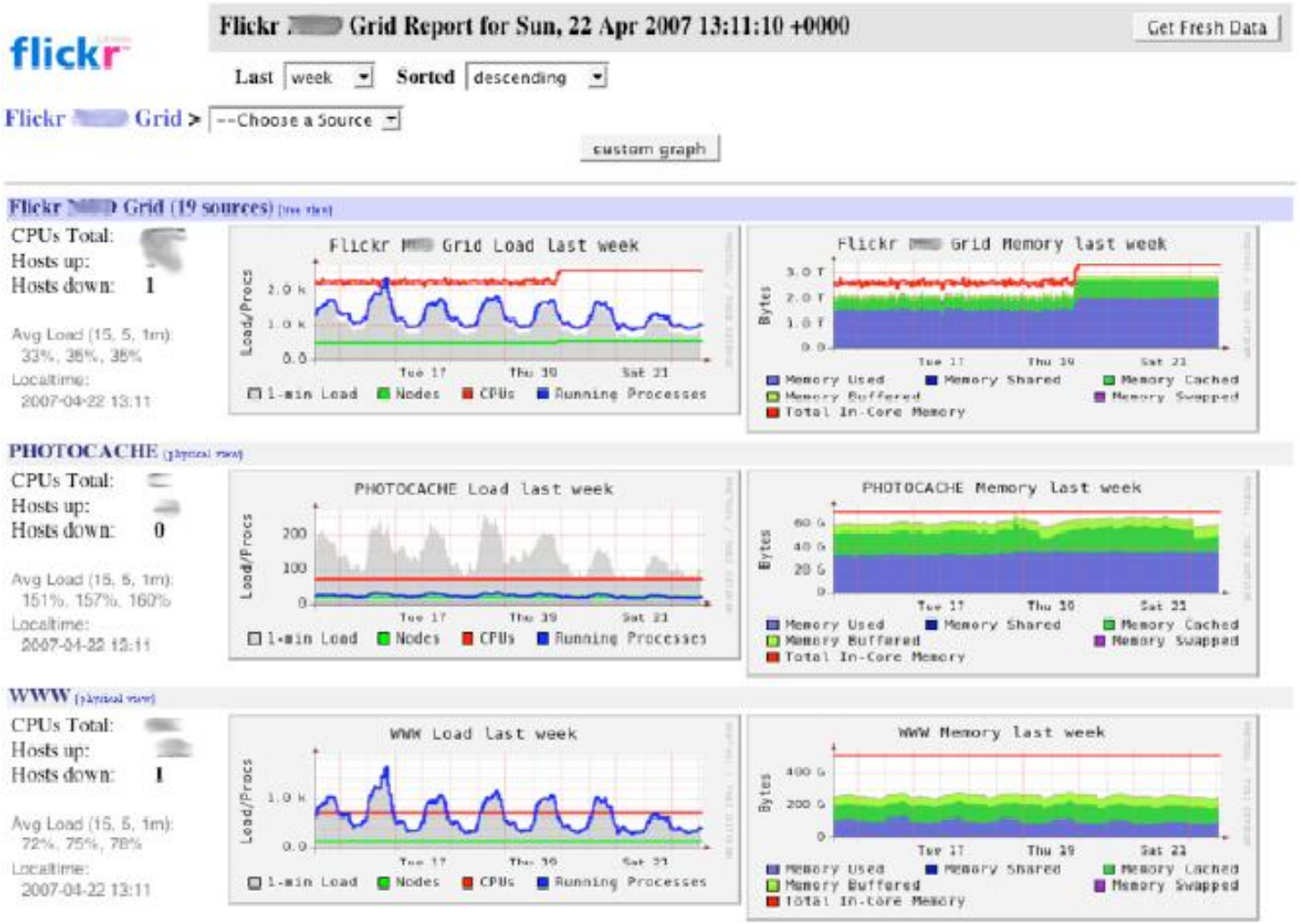
Rocks Cluster

- Carga del sistema en un instante de tiempo

  La escala cambia dinamicamente. Se desplazade derecha a izquierda.



Monitorización de Clusters: Ganglia



Estructura del CÓMO

- Bloque I – Instalar ROCKS
- Bloque II – IPVS
- Bloque III - HA

Balanceo de Carga: Linux Virtual Server

- Definición
  - Conjunto de servidores que comparten la carga de trabajo
  - Esquema aplicado al procesamiento distribuido y/o al sistema de comunicación con el fin de que un dispositivo no se sature
- Ampliar capacidad → Añadiendo ordenadores al cluster.
- Robustez ante la caída de algún ordenador
- Importante en redes donde es dificil predecir el número de peticiones que se van a cursar a un servidor
- Los sitios web muy demandados suelen emplear dos o más servidores bajo un esquema de balanceo de carga
- Servidor saturado → Peticiones son redireccionadas a otro(s)

Balanceo de Carga: Linux Virtual Server

- Balanceo de Carga
  - Problema
    - Un balanceo estático perfecto no es posible
      - El tiempo de ejecución de los trabajos no es conocido
    - Los sistemas no balanceados pueden ser ineficientes
  - Solución
    - Dns
    - LVS

Balanceo de Carga: Linux Virtual Server

- Dns
  - Definición de varias IPs para un mismo dominio
  - Servidor devuelve cada vez una IP distinta
  - Cola round-robin para ir sirviendo las peticiones: Distribución pseudo-aleatoria
  - No se tiene en cuenta la carga real de cada servidor
- LVS
  - Aumentar la sensación de “unidad” del cluster
  - Una dirección IP cara al usuario
  - Facilidad de enmascarar un error
  - Distribución por conexión, o por sesiones
  - Simplificación de la administración del cluster

Balanceo de Carga: Linux Virtual Server

- Linux Virtual Server (LVS, IPVS en kernels 2.6.x) es un servicio de red altamente escalable y de alta disponibilidad
- Arquitectura transparente para el usuario final
- El usuario interactua con los servidores como si solo existiera un solo servidor de alto rendimiento
- Los servidores reales pueden estar conectados por

LAN de alta velocidad

WAN dispersa (VPN)

Balanceo de Carga: Linux Virtual Server

- Objetivo
  - Equilibrado de carga mediante NAT (Network Address Translation), tunneling IP o enrutamiento directo (DR) por medio del frontend que da servicio a peticiones FTP y HTTP a los nodos de un cluster. Este servicio es proveído a nivel de kernel (ha de estar compilado el soporte para LVS/IPVS)
- Configuración
  - LVS puede ser implementado de 3 formas diferentes, usando las 3 técnicas de balanceo IP (forwarding de paquetes) existentes en el LinuxDirector:
    - Virtual Server vía Nat
    - Virtual Server vía IP tunneling
    - Virtual Server vía ruta directa

Balanceo de Carga: Linux Virtual Server

- NAT
  - Permite utilizar cualquier SO que soporte TCP/IP ya que solo es requerida una dirección IP
  - Funciona con una única IP pública
  - Los paquetes son reescritos por el frontend para ocultar los nodos internos
  - Escalabilidad limitada: Sólo es aceptable para un número pequeño de nodos, por la sobrecarga que acarrea (20 nodos)

Balanceo de Carga: Linux Virtual Server

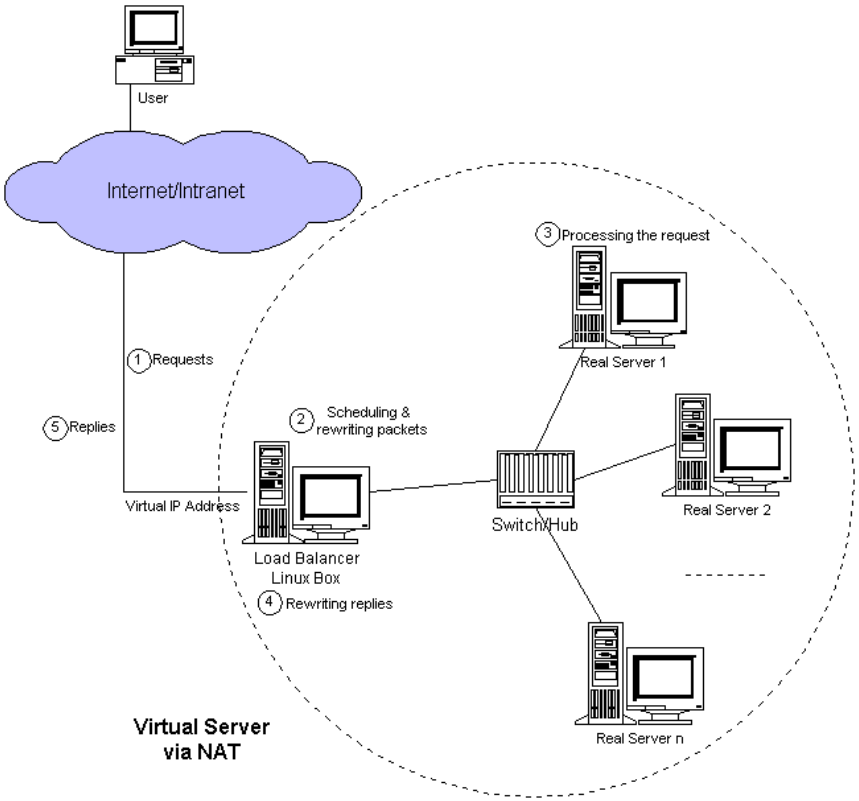
- Tunneling IP
  - Tunneling IP es similar a NAT, pero el frontend ya no reescribe los paquetes, siendo su tarea mucho más liviana
  - El frontend programa las peticiones a los diferentes servidores reales
  - Los servidores reales envian las respuestas directamente a los usuarios finales
  - Los servidores deben tener el protocolo “IP Tunneling” habilitado

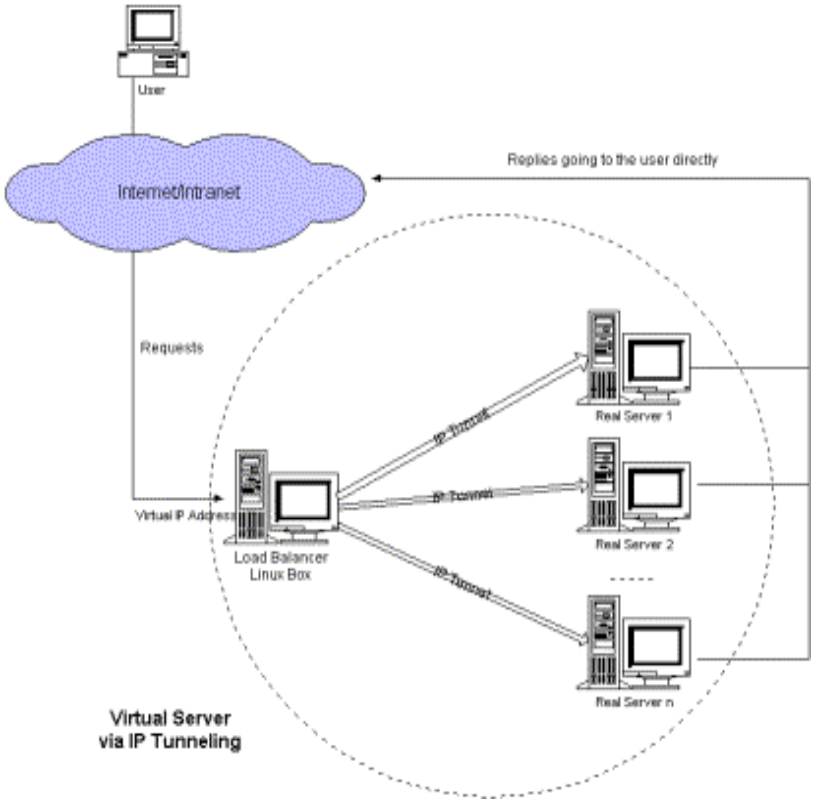
Balanceo de Carga: Linux Virtual Server

- Enrutamiento directo
  - El enrutamiento directo (DR) es un sistema aún más ligero
  - Los servidores comparten la IP pública del cluster
  - Necesita que todos los servidores compartan el mismo segmento de red

Balanceo de Carga: Linux Virtual Server

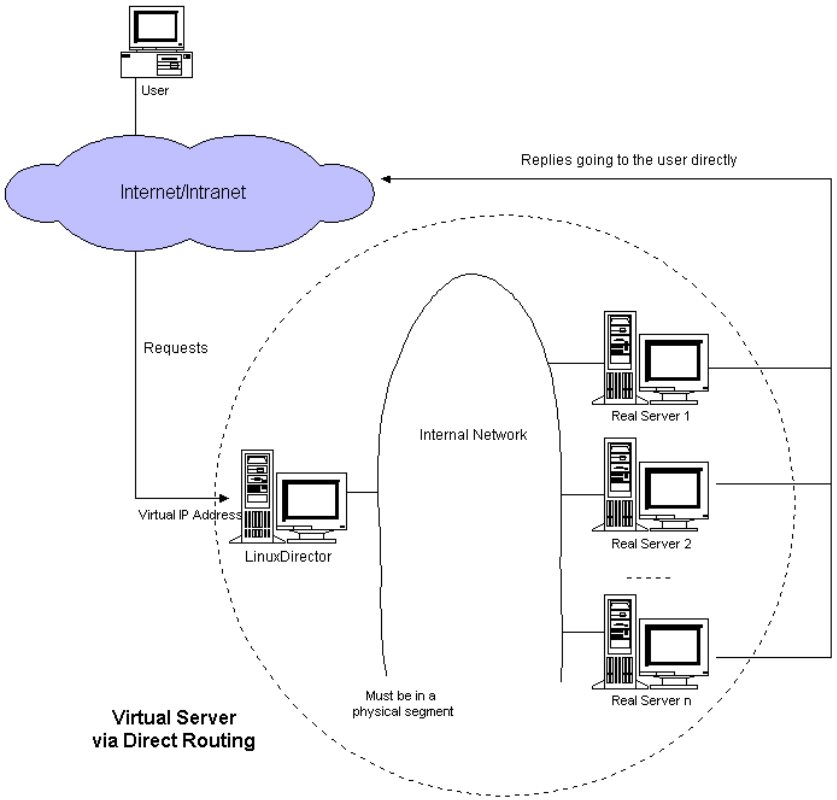
- Linux Virtual Server (LVS): NAT vs TUN



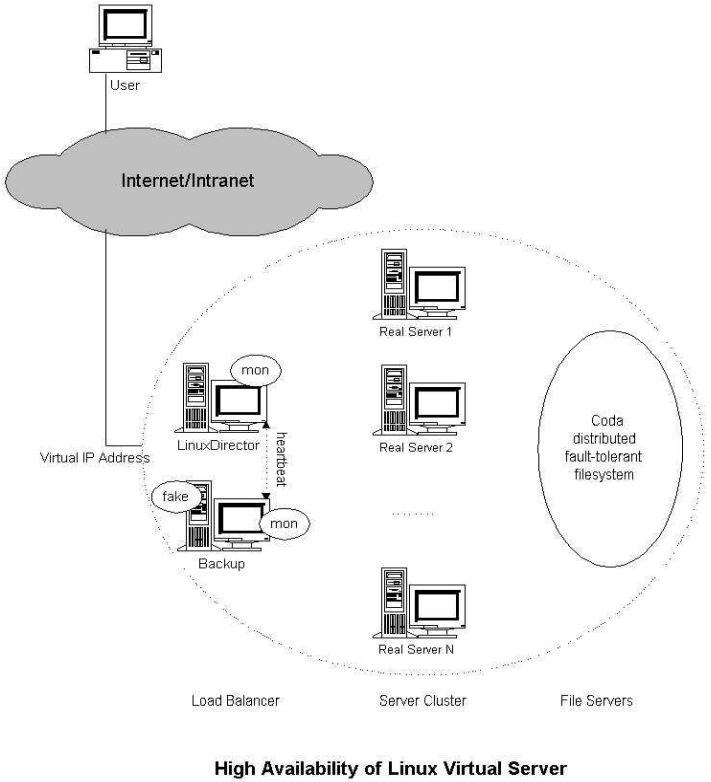


Balanceo de Carga: Linux Virtual Server

- Linux Virtual Server (LVS): DR & HA



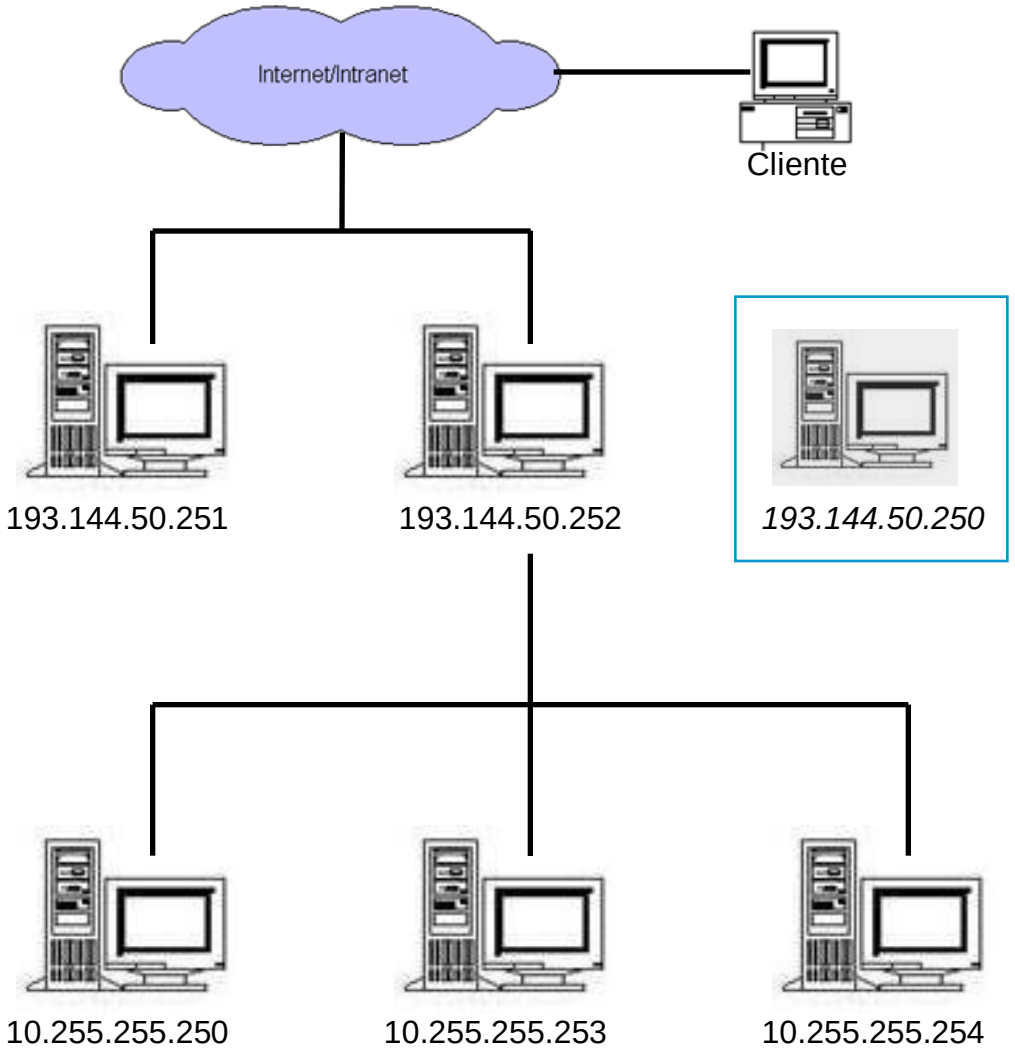
Virtual Server via Direct Routing



High Availability of Linux Virtual Server

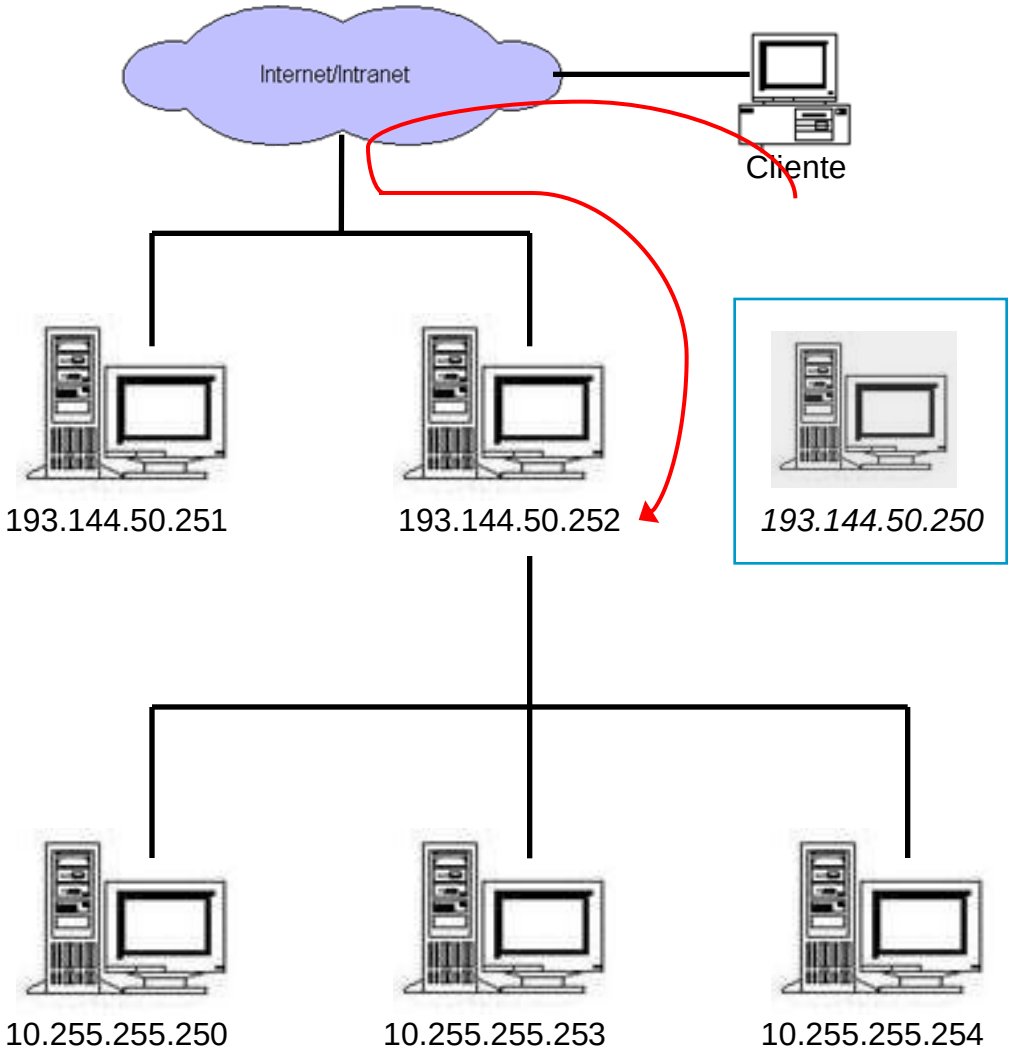
Ejemplo Practico: Linux Virtual Server

- Topología
  - Servidor grupo 1
  - Servidor grupo 2
    - Nodo 1
    - Nodo 2
    - Nodo 3



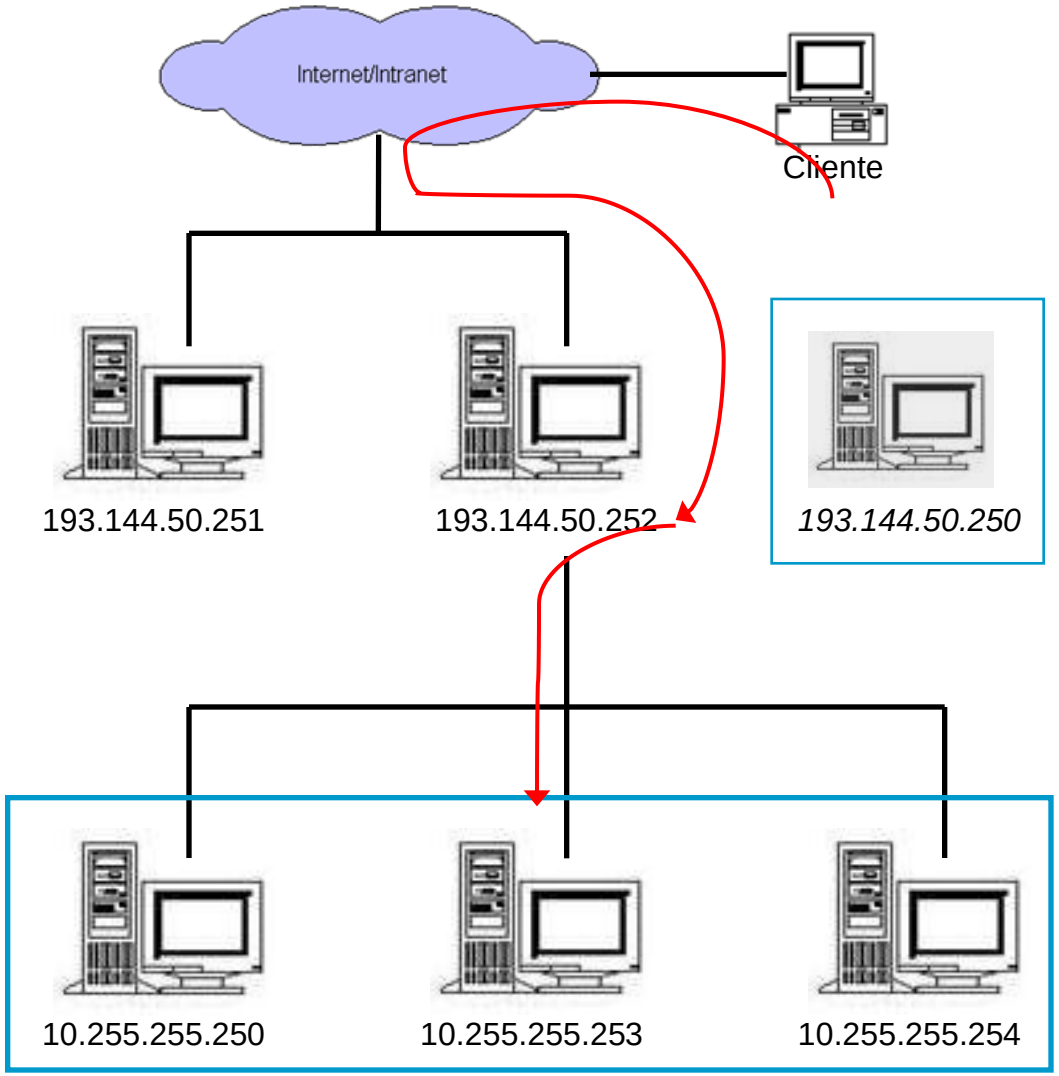
Ejemplo Practico: Linux Virtual Server

- NAT



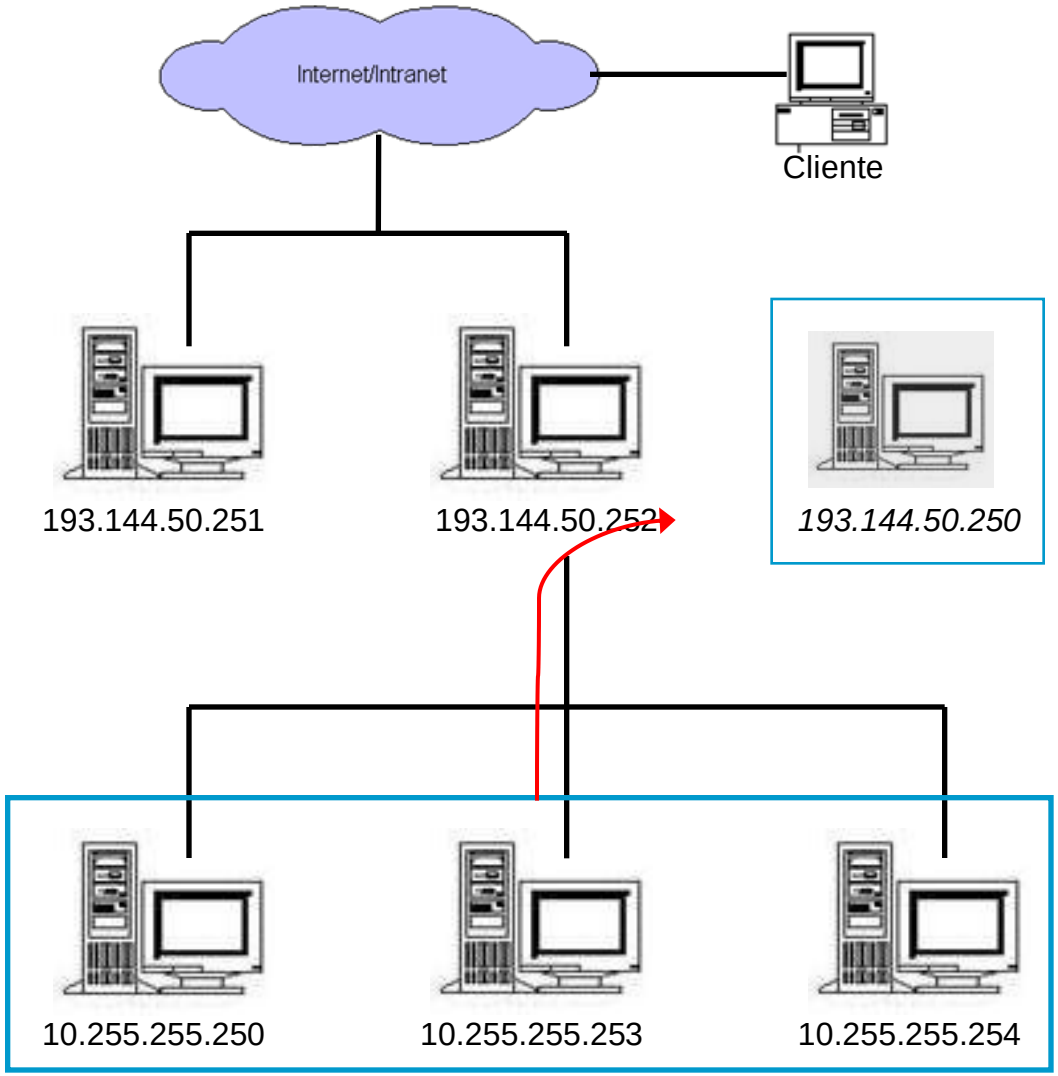
Ejemplo Practico: Linux Virtual Server

- NAT



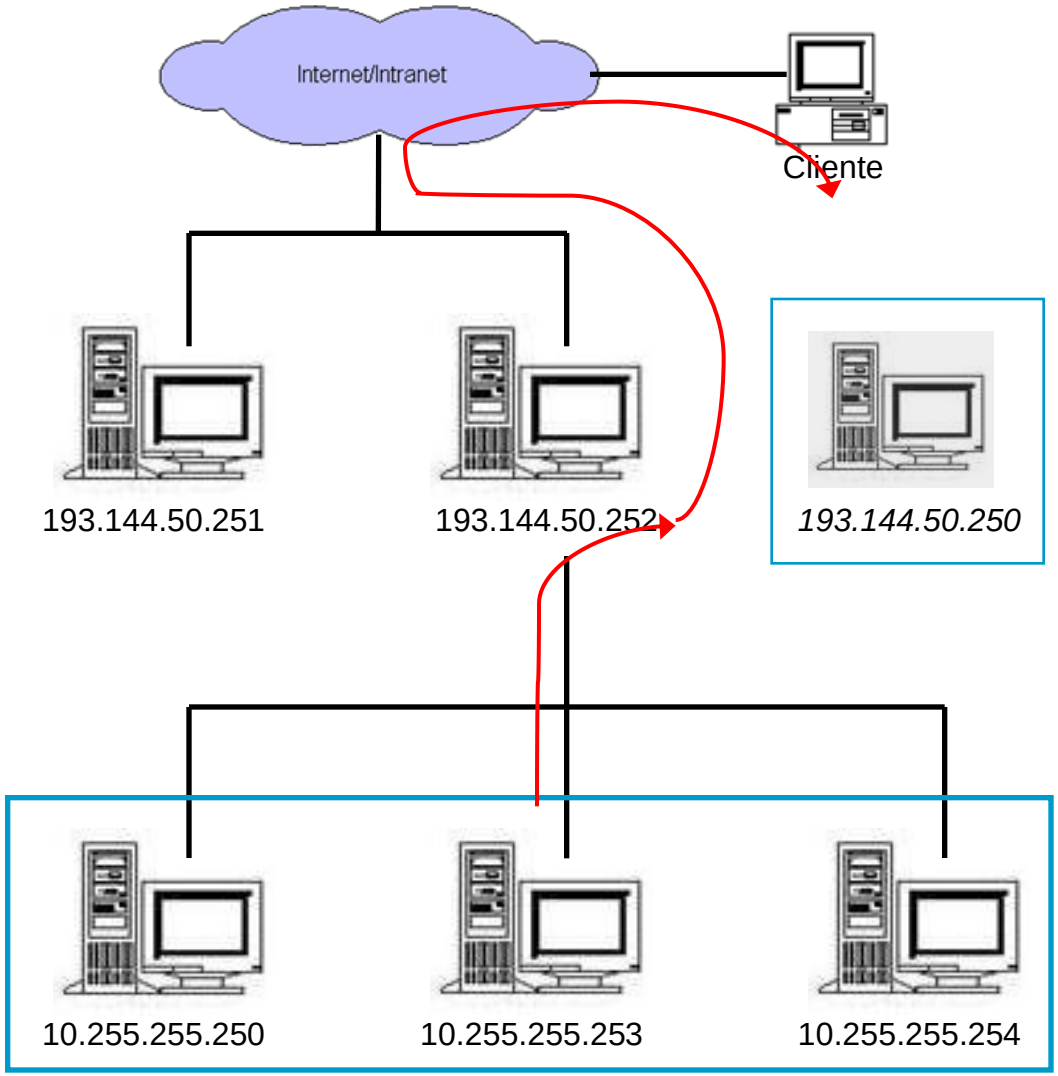
Ejemplo Practico: Linux Virtual Server

- NAT



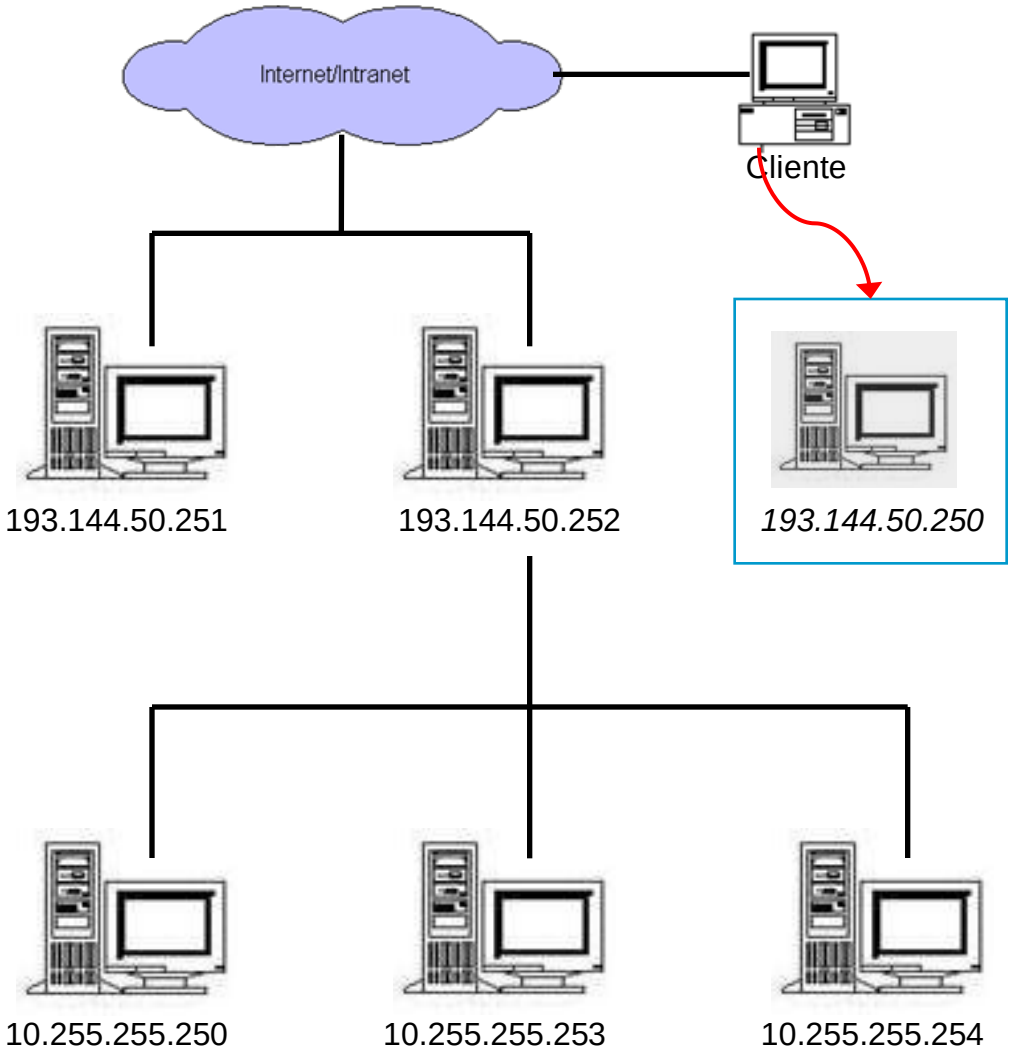
Ejemplo Practico: Linux Virtual Server

- NAT



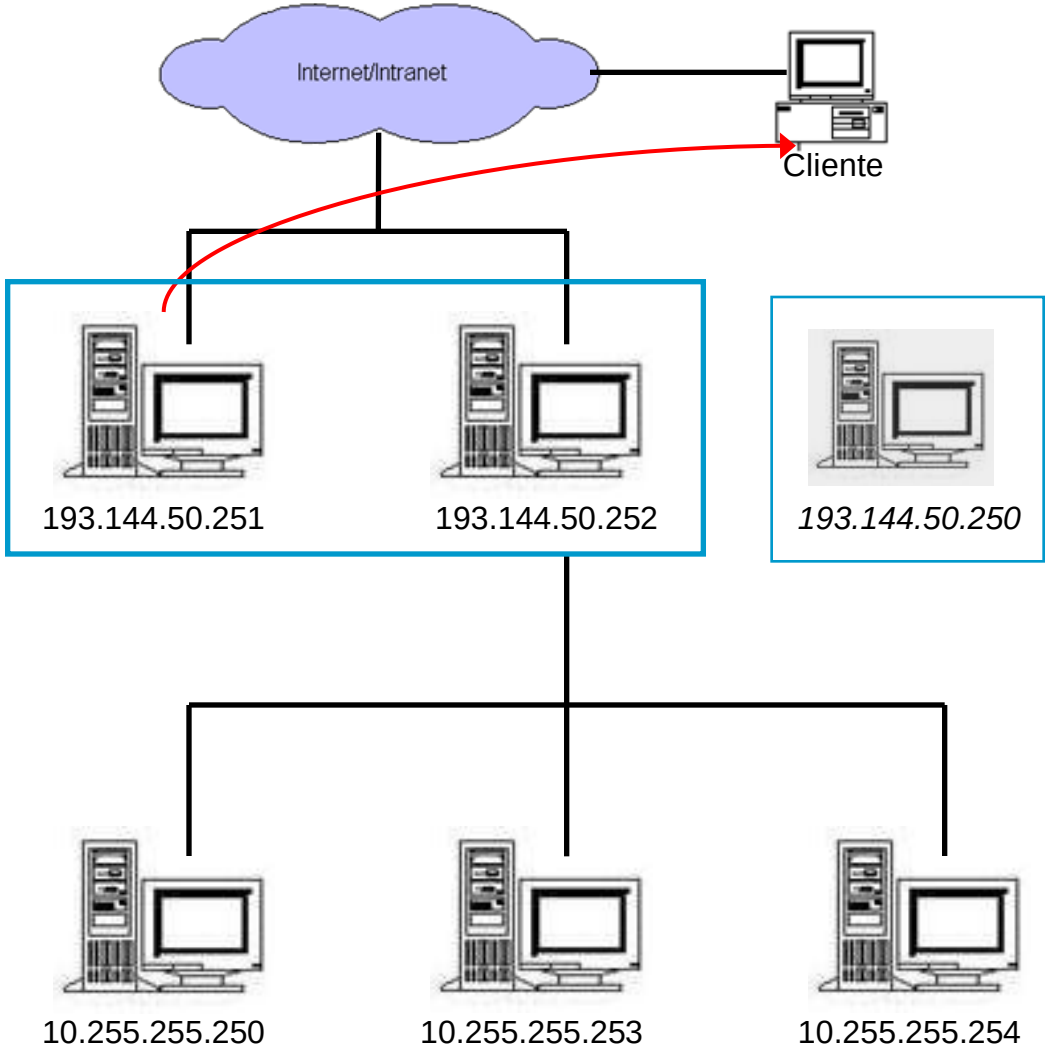
Ejemplo Practico: Linux Virtual Server

- TUN



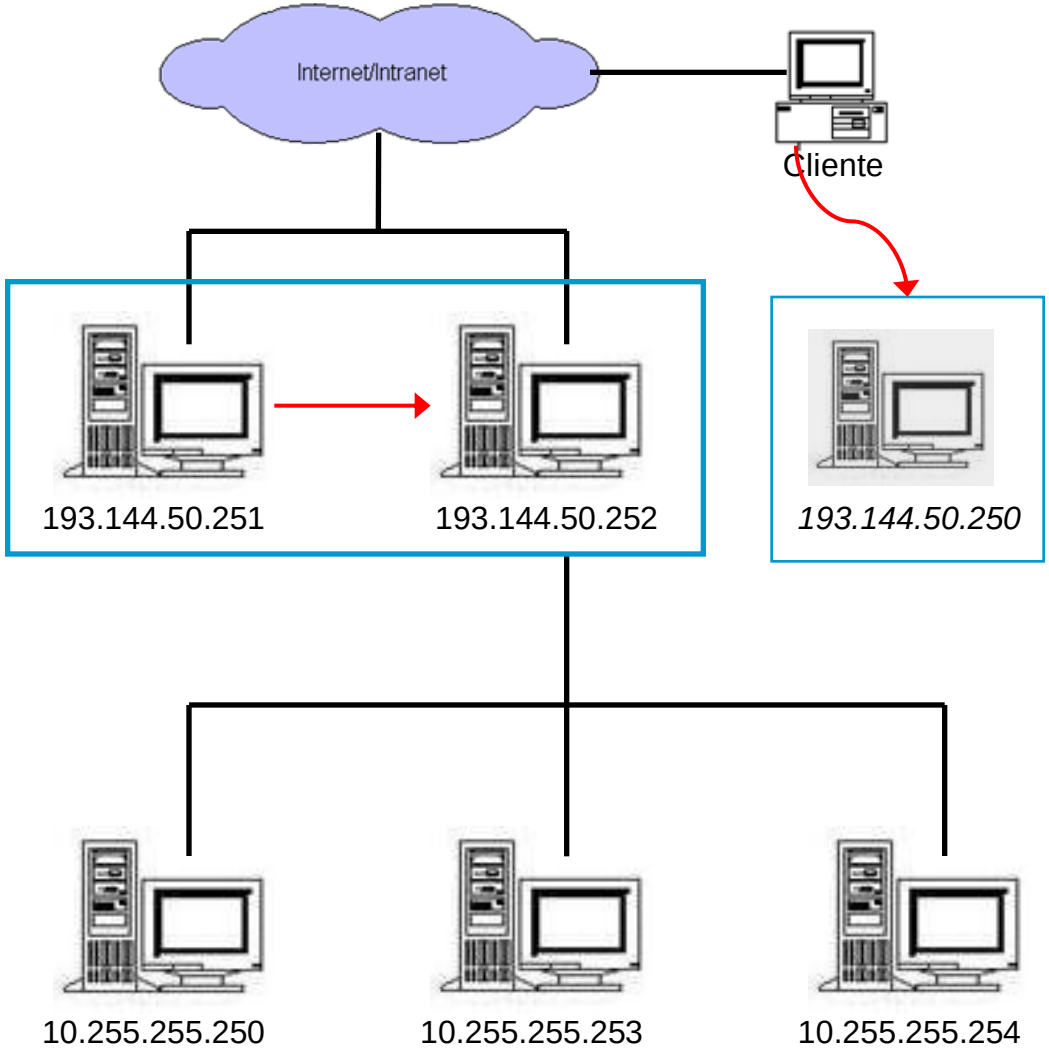
Ejemplo Practico: Linux Virtual Server

- TUN



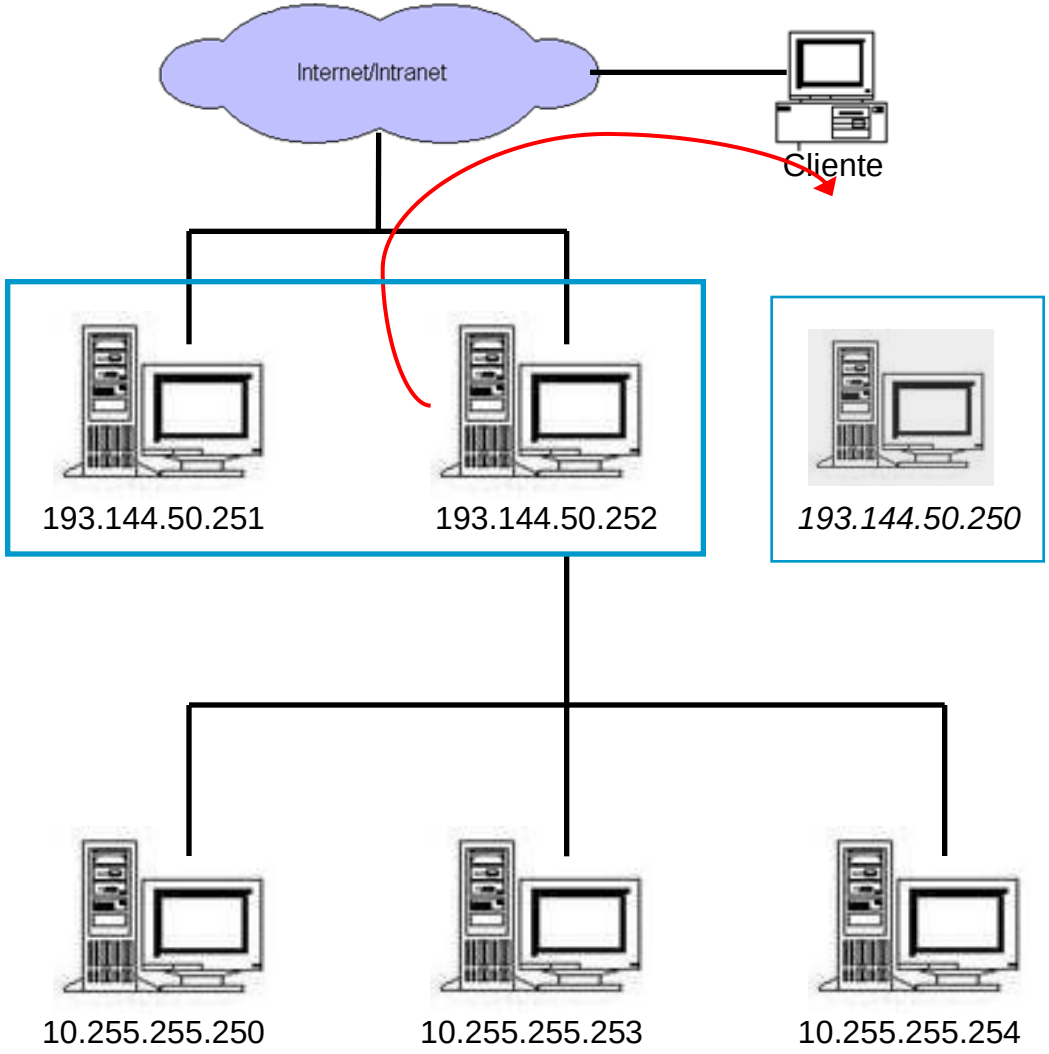
Ejemplo Practico: Linux Virtual Server

- TUN



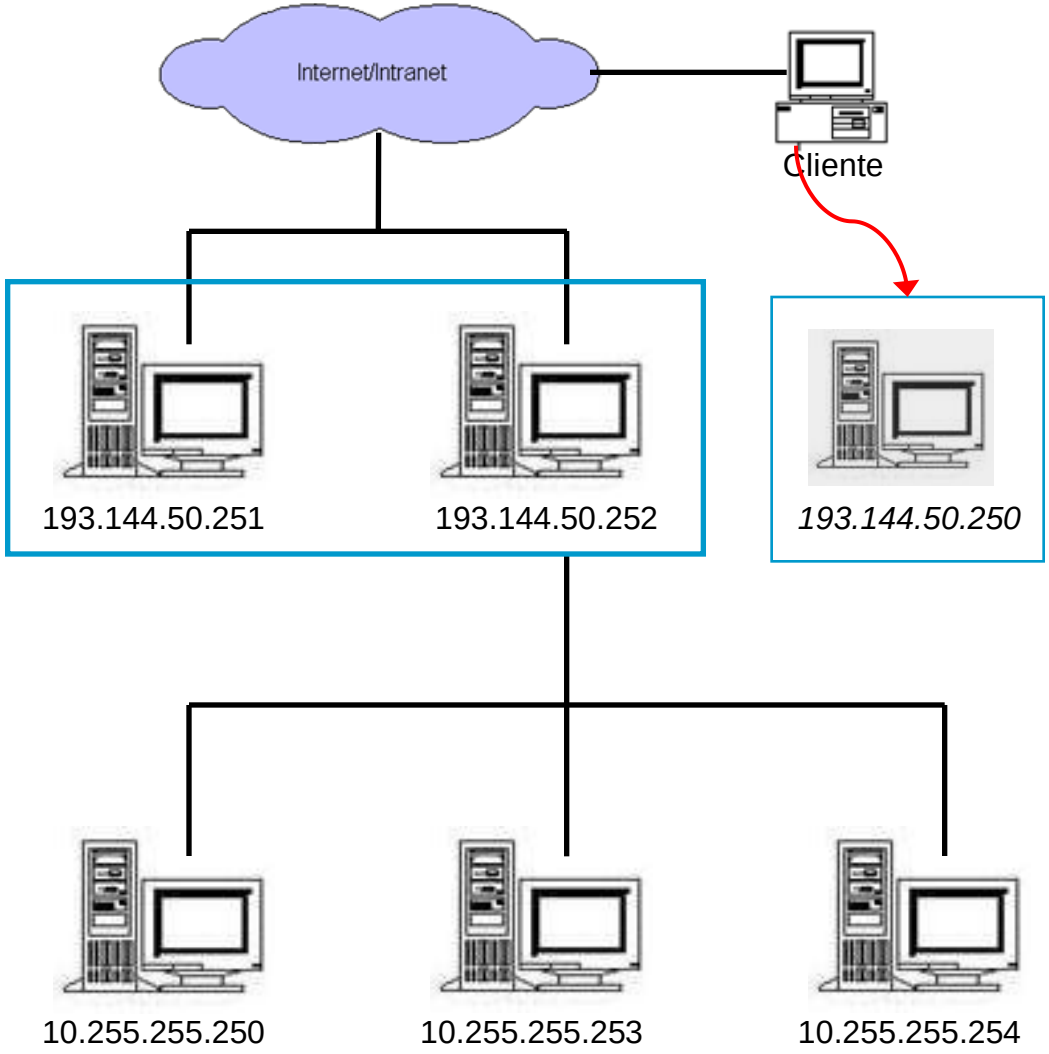
Ejemplo Practico: Linux Virtual Server

- TUN



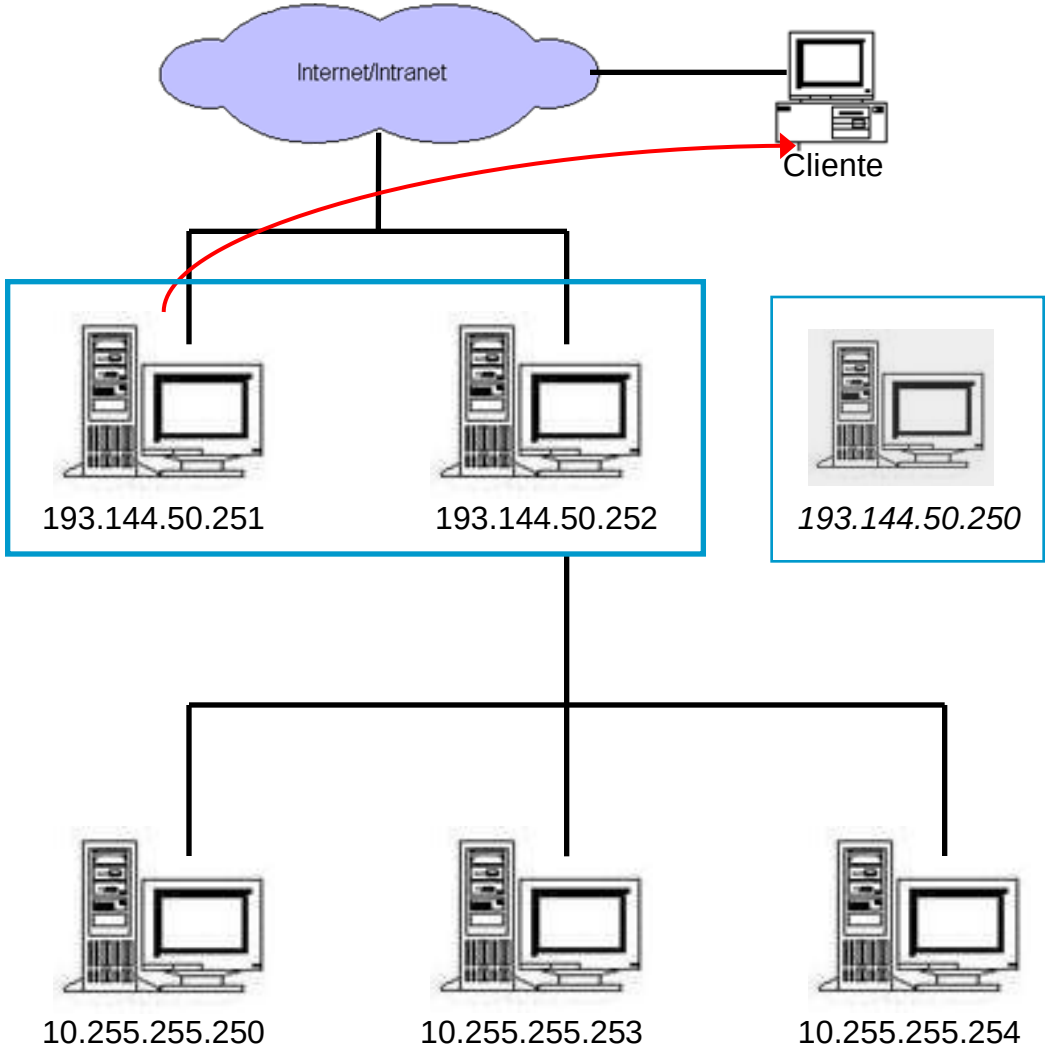
Ejemplo Practico: Linux Virtual Server

- DR



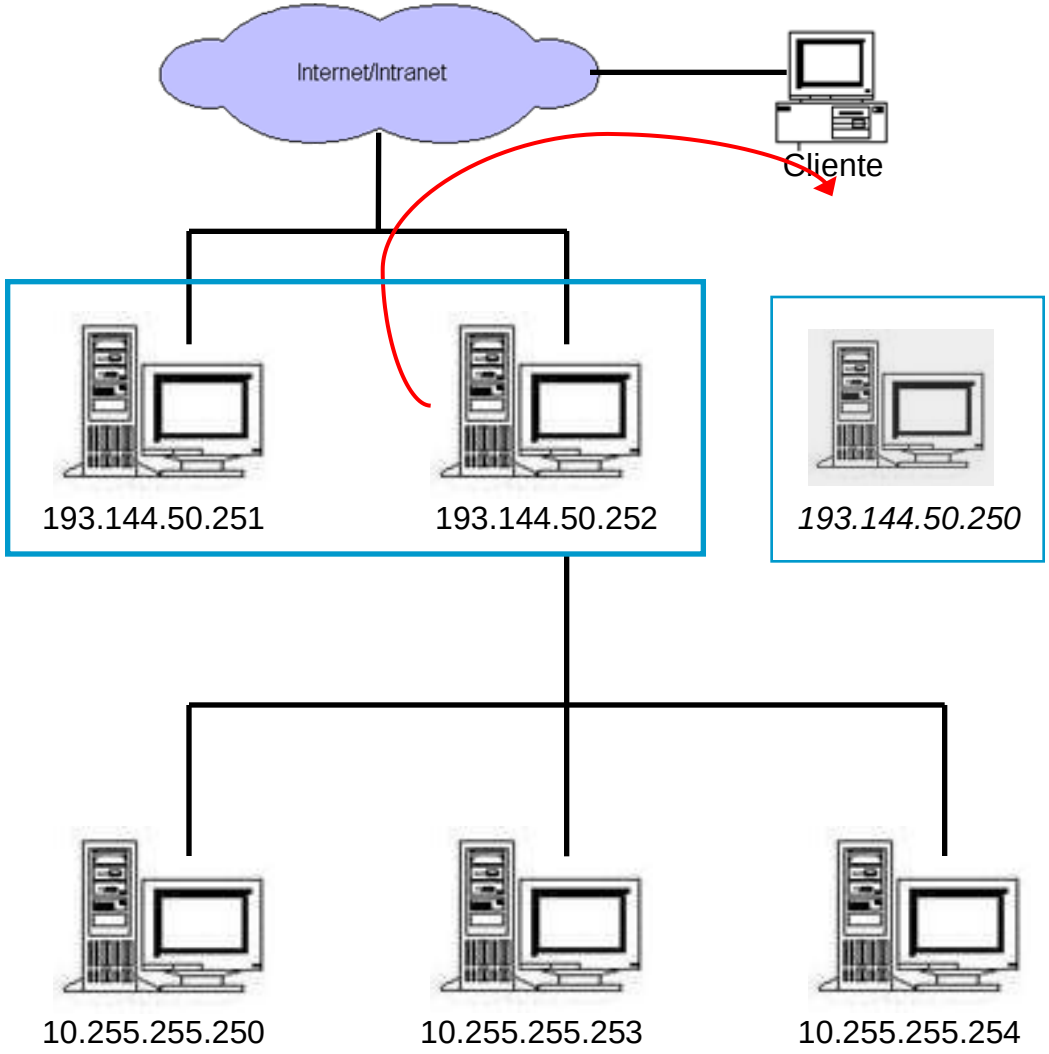
Ejemplo Practico: Linux Virtual Server

- DR



Ejemplo Practico: Linux Virtual Server

- DR



Balanceo de Carga: ipvsadm

- #ipvsadm - Linux Virtual Server administration
  - Sirve para:
    - Activar, mantener e inspeccionar el “Virtual Server Table” en el kernel Linux.
    - Escalar los servicios de red basados en dos o más nodos.
    - Redireccionar servicios a los nodos.
    - Realizar NAT, Tunneling y Direct Routing sobre protocolos UDP o TCP.
    - Balanceo de carga según diversos algoritmos de ponderación.

Balanceo de Carga: ipvsadm

- #ipvsadm - Linux Virtual Server administration
  - Formato del comando básico

 # ipvsadm *COMANDO* [*protocolo*] *direccion-servicio*
[*metodo-planificacion*] [*opciones persistencia*]

 # ipvsadm *COMANDO* [*protocolo*]
direccion-servicio direccion-servidor
[*método-paquete-reenvio*] [*opciones de ponderación*]

Balanceo de Carga: ipvsadm

- #ipvsadm - Linux Virtual Server administration
  - COMANDOS

```
-A, --add-service
-E, --edit-service
-D, --delete-service
-C, --clear
-R, --restore
-S, --save
-a, --add-server
-e, --edit-server
-d, --delete-server
```

```
-L, -l, --list
-Z, --zero
--set tcp tcpfin udp
--start-daemon state
--stop-daemon
-h, --help
```

Balanceo de Carga: ipvsadm

- #ipvsadm - Linux Virtual Server administration
  - PARAMETROS

```
-t, --tcp-service service-address
-u, --udp-service service-address
-f, --fwmark-service integer
-s, --scheduler scheduling-method
-p, --persistent [timeout]
-M, --netmask netmask
-r, --real-server server-address
```

Balanceo de Carga: ipvsadm

[método de reenvio de paquetes]

```
   -g, --gatewaying      < DIRECT ROUTING >
   -i, --ipip            < TUNNELING >
   -m, --masquerading    < NAT >
```

```
-w, --weight weight
-x, --u-threshold uthreshold
-y, --l-threshold lthreshold
```

```
--mcast-interface interface
--syncid syncid
-c, --connection
--timeout
--daemon
--stats
```

```
--rate
--thresholds
--persistent-conn
--sort
-n, --numeric
```

Balanceo de Carga: Ejemplo

Ejemplo: REQUISITOS PREVIOS

I. */etc/sysconfig/iptables*:

```
*nat
-A POSTROUTING -o eth1 -j MASQUERADE COMMIT

*filter
:INPUT ACCEPT [0:0]
:FORWARD DROP [0:0]
:OUTPUT ACCEPT [0:0]
# Preamble
-A FORWARD -i eth1 -o eth0 -m state --state
NEW,RELATED,ESTABLISHED -j ACCEPT
-A FORWARD -i eth0 -j ACCEPT
```

II. *echo "1" > /proc/sys/net/ipv4/ip_forward*

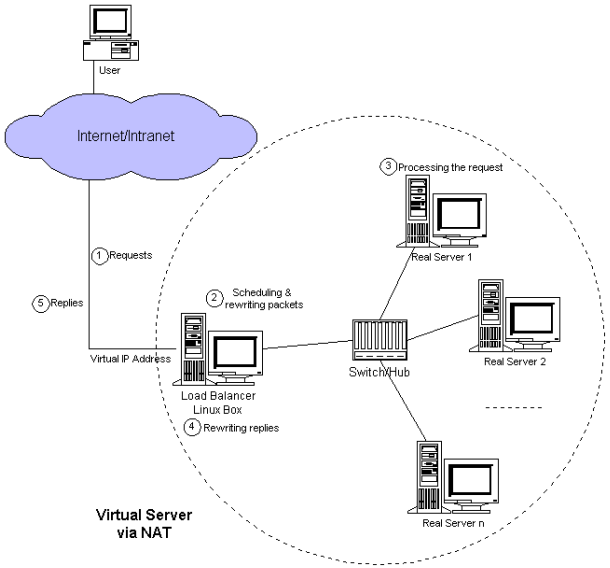
Balanceo de Carga: Linux Virtual Server

- Linux Virtual Server (LVS)

Ejemplo

Network Address Translation

(NAT)



En cada servidor (grupo 2#)

ipvsadm -C

ipvsadm -A -t 193.144.50.252:http -s wrr

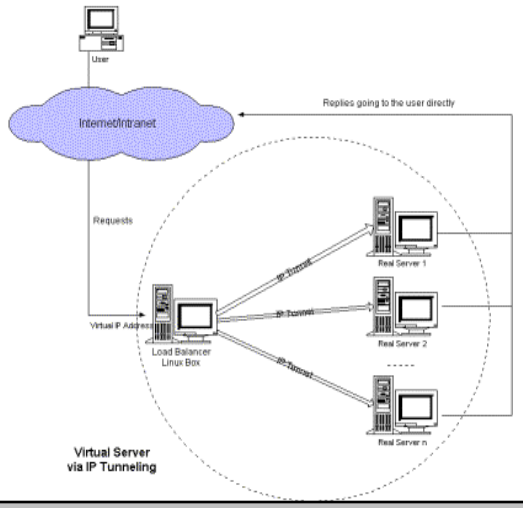
ipvsadm -a -t 193.144.50.252:http -r 193.144.50.252:http -m -w 3

ipvsadm -a -t 193.144.50.252:http -r compute-0-0.local:http -m -w 3

ipvsadm -a -t 193.144.50.252:http -r compute-0-7.local:http -m -w 1

Balanceo de Carga: Linux Virtual Server

- Linux Virtual Server (LVS)



Ejemplo

Tunneling (TUN)

Director (grupo1#):

ipvsadm -C

ipvsadm -A -t 193.144.50.250:80 -s wrr

ipvsadm -a -t 193.144.50.250:80 -r 193.144.50.252 -i -w 3

ipvsadm -a -t 193.144.50.250:80 -r 193.144.50.251 -i -w 1

En cada servidor real (grupo2#):

ifconfig tunl0 193.144.50.250 netmask 255.255.255.255 broadcast 193.144.50.250 up

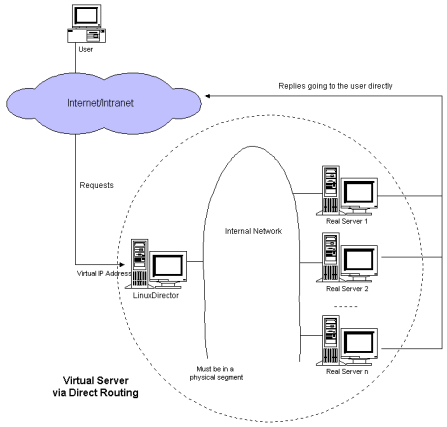
route add -host 193.144.50.250 dev tunl0

Balanceo de Carga: Linux Virtual Server

- Linux Virtual Server (LVS)

Ejemplo

Direct Routing (DR)



Director (grupo1#):

ipvsadm -C

ifconfig eth1:0 193.144.50.250 netmask 255.255.255.255 broadcast 193.144.50.250 up

ipvsadm -A -t 193.144.50.250:80 -s wrr

ipvsadm -a -t 193.144.50.250:80 -r 193.144.50.252 -g -w 3

ipvsadm -a -t 193.144.50.250:80 -r 193.144.50.251 -g -w 1

In each real server (grupo2#):

ifconfig lo:0 193.144.50.250 netmask 255.255.255.255 broadcast 193.144.50.250 up

route add -host 193.144.50.250 dev lo:0

Estructura del CÓMO

- Bloque I – Instalar ROCKS
- Bloque II – IPVS
- Bloque III - HA

Alta Disponibilidad (I)

Sistema que es resistente a los fallos de software, hardware y de energía.

La capacidad de un sistema para permanecer funcionando incluso después de la aparición de graves defectos es lo que hace que sea seguro y fiable.

Alta Disponibilidad (II): Cuándo y Cómo implantar

Una verdadera necesidad de poner en marcha un sistema con Alta Disponibilidad depende de las respuestas a ciertas preguntas.

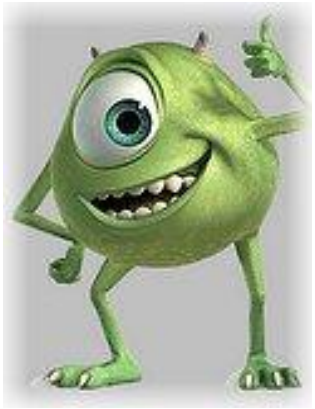
Alta Disponibilidad (III): Preguntas (I)

¿La inversión necesaria para construir y mantener una infraestructura de redundancia en paralelo justifica el costo para su empresa?

¿Qué duración de la interrupción considera aceptable?

¿El sistema de su empresa es el "alma" de su negocio?

Alta Disponibilidad (IV): Preguntas (y II)



Si las respuestas son positivas, y son solamente aceptables unos minutos de interrupción, es muy aconsejable empezar a pensar rápidamente en la aplicación de alta disponibilidad en su entorno.

Alta Disponibilidad (V): Cálculo de la A.D.

La disponibilidad de un sistema es la relación entre la duración de la vida útil de este sistema y de su tiempo total de vida.

Se representa por la siguiente fórmula:

Disponibilidad = MTBF / (MTBF + MTTR)

MTBF – Tiempo medio entre fallos

MTTR – Tiempo medio de recuperación. Es el espacio de tiempo (en promedio), que transcurre entre la ocurrencia del error y la recuperación total del sistema a su estado operacional

Alta Disponibilidad (VI): Regla de los 9s (I)

La regla de los nueves consiste en una escala utilizada para exponer el tiempo posible de no disponibilidad de un servicio.

Por ejemplo:

99,9% = 43.8 minutos/mes -> 8,76 horas/año ("tres nueves")

99,99% = 4.38 minutos/mes -> 52.6 minutos/año ("cuatro nueves")

99,999% = 0.44 minutos/mes -> 5.26 minutos/año ("cinco nueves")

¿Cuánto tiempo se permitiría estar caída la facultad virtual de una Universidad?

¿Cuánto tiempo se permitiría estar caída la web de un banco?

¿Cuánto tiempo se permitiría estar caído el servicio de telefonía móvil?

Alta Disponibilidad (VII): Regla de los 9s (II)

Veamos un ejemplo:

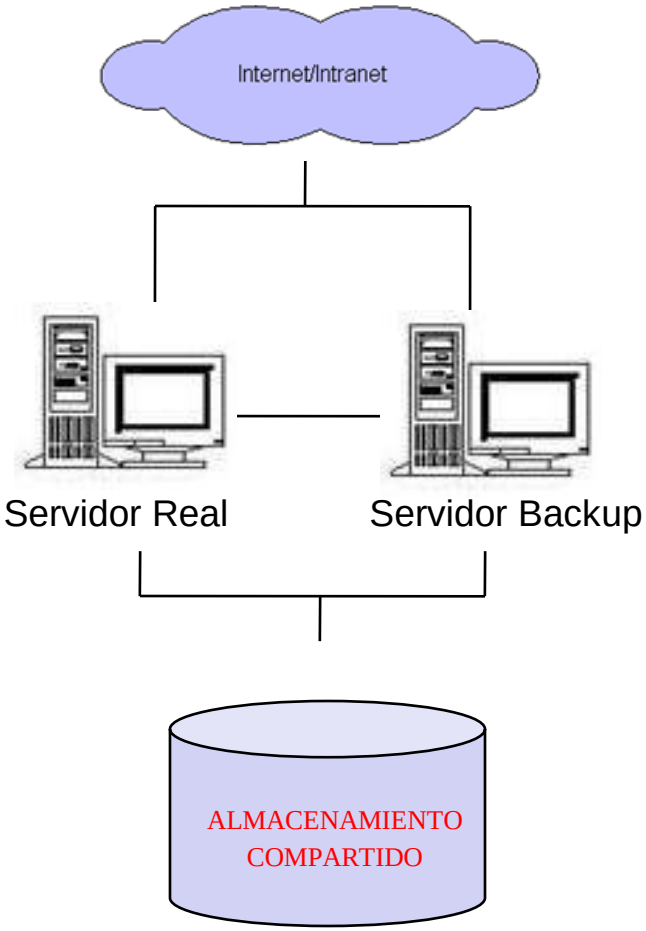
Una empresa de telefonía factura 7.200 millones de € al año trabajando 8760 horas (24 horas * 365 días).

Veamos la relación entre disponibilidad y pérdida:

99% = 72.000.000 € ¿Es asumible esta pérdida?
99.9% = 7.200.000 €
99.99% = 720.000 €

Y no sólo debemos considerar pérdidas directas por la ausencia de funcionalidad, surgirían grandes pérdidas por el descontento de los clientes, etc.

Alta Disponibilidad: Hardware



Alta Disponibilidad (VIII): Middleware para Clusters con HA

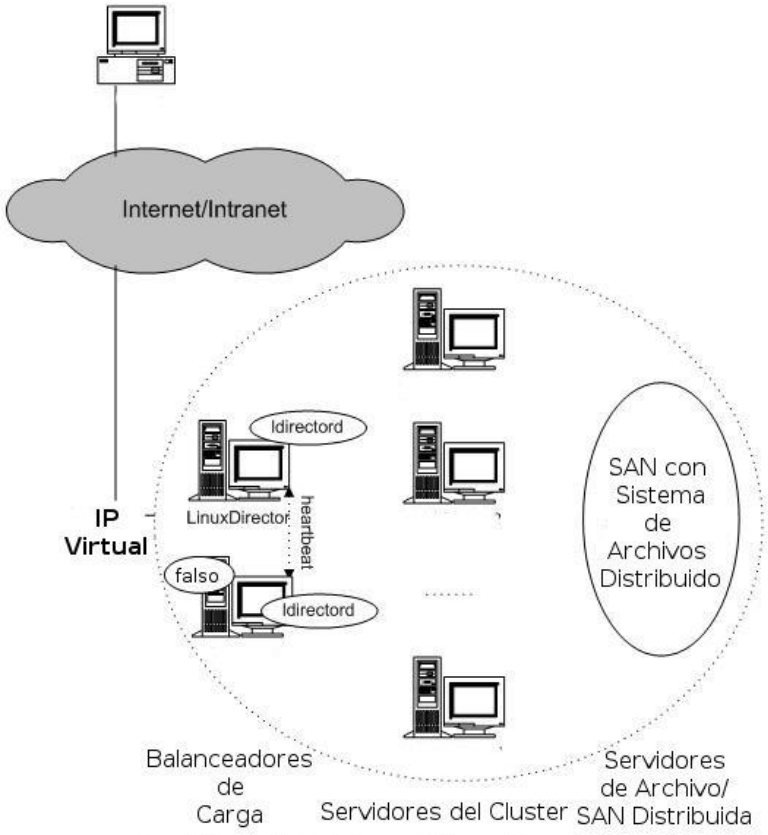
- Ldirectord
- Keepalived
- Munin
- Cman, ccs
- Heartbeat
- Y etc...

Alta Disponibilidad (IX): Ldirectord

Es un demonio que monitorea y administra a los servidores reales que son parte de un cluster LVS de carga balanceada.

Normalmente se usa en conjunto con heartbeat, aunque puede funcionar con otro detector de servicios.

Alta Disponibilidad (X): HeartBeat y Ldirectord



Alta Disponibilidad con Linux Virtual Server

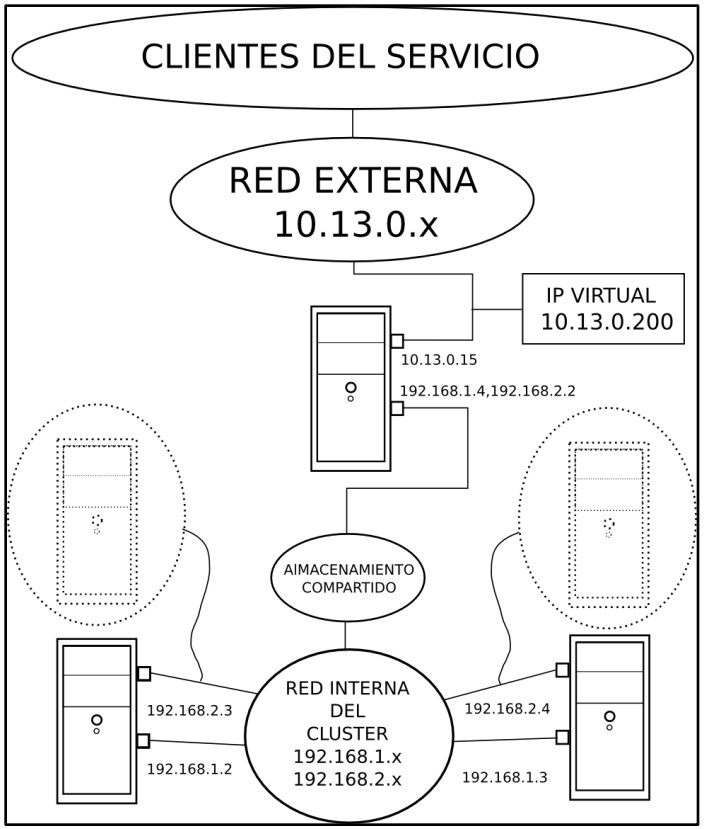
Alta Disponibilidad (XI): Keepalived (I)

Supervisa cada nodo y servicio, pueden haber tantos nodos y servicios como se quiera.

Balanceo de Carga: Usando solo 128 bytes/conexión

Recuperación Automática: Se dirige la conexión a cualquier otro nodo

Alta Disponibilidad (XII): Keepalived (y II)



Alta Disponibilidad (XIII): Cman

Maneja y supervisa la membresía de nodos dentro de un cluster. Además se encarga de las operaciones de bloqueo distribuido (para operar sistemas de archivos compartidos dentro de un cluster)

Alta Disponibilidad (XIV): CCS

Cluster Configuration System

Se trata de un sistema de configuración distribuido, es decir, ccs se encarga de que la configuración del cluster se mantenga uniforme entre todos los nodos del cluster y define los servicios compartidos y dominios de failover que se usan si el servicio falla en uno o más nodos.

Alta Disponibilidad (XV): Munin

Herramienta de registro y monitoreo cliente/servidor especialmente útil para monitorear el funcionamiento de SANs, redes y aplicaciones.
Es altamente extensible y permite que los datos generados por la misma sean accesados vía web.

Alta Disponibilidad (XVI): Heartbeat (I)

Requerimientos Hardware

- 2 ordenadores (nodos)
-Conexión entre los dos nodos

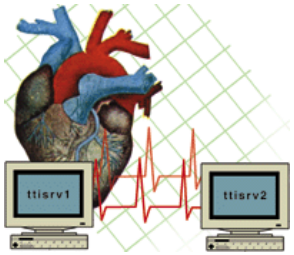
(Direcciones privadas del tipo 10.x.x.x)

→ Ethernet (Cable cruzado)

→ Serie (Cable null modem)

Requerimientos Software

- Linux
- HeartBeat



Alta Disponibilidad (XVII): Heartbeat (II)

Probando las conexiones

Ethernet

Para ver las interfaces de red y su configuración

$ ifconfig

Para ver la información de enrutamiento de la red

$ netstat -nr

Serie

Para comprobar la comunicación serie entre todos

Emisor

$ echo HelloWorld > /dev/ttyS0

Receptor

$ cat < /dev/ttyS0

Alta Disponibilidad (XVIII): Heartbeat (III)

Instalando HeartBeat

Comandos y fuentes para varias distribuciones en:

http://www.linux-ha.org/download

A partir del código seria:

```
$ ./ConfigureMe configure
$ make
$ make install
```

Alta Disponibilidad (XIX): Heartbeat (IV)

Configurando HeartBeat (I)

Se tienen que configurar tres archivos :

- ha.cf
- Haresources
  - Ipfail
- Authkeys

Alta Disponibilidad (XX): Heartbeat (V)

Configurando HeartBeat (II) – ha.cf (I)

ha.cf (en /etc/ha.d) es el fichero utilizado por HeartBeat para conocer los medios por los que están conectados los nodos, y como configurarlos.

Alta Disponibilidad (XXI): Heartbeat (VI)

Configurando HeartBeat (III) – ha.cf (II)

serial /dev/ttyS0 Conectado a través del puerto serie en ese device
Si estuviese conectado a través de ethernet usaríamos, por ejemplo, bcast

watchdog /dev/watchdog (Opcional) Funcionamiento mínimo de HeartBeat.
Resetea la máquina al minuto de estar 'enferma'. Necesita realizar pasos adicionales

bcast eth1 Utilizar heartbeat broadcast sobre la interfaz eth1 (en general ethx)

keepalive 2 Establece el tiempo entre 'heartbeats' en segundos

warntime 10 Establece el tiempo para grabar en los logs un heartbeat como tardío

deadtime 30 Establece el tiempo para considerar un nodo como 'muerto'

initdead 120 Caso especial de deadtime utilizado al resetear la máquina

Alta Disponibilidad (XXII): Heartbeat (VII)

Configurando HeartBeat (IV) – ha.cf (y III)

baud 19200 Establece la velocidad de conexión a través del puerto serie

udpport 694 Puerto para la comunicación bcast o ucast. Este es el puerto por defecto, y el registrado como oficial por la IANA para este cometido

auto_failback on El nodo listado como master en haresources obtiene los recursos, si este cae pasan al esclavo. Esta opción indica si el maestro recupera los recursos tras recuperarse de la caída

node linuxha1 (Obligatorio) Hostname de la máquina descrita por $ uname –n (Una línea por cada nodo)

respawn userid cmd (Opcional) Lista un comando para ser lanzado y monitorizado Ejemplo: respawn hacluster /usr/lib/heartbeat/ipfail para ipfail

Alta Disponibilidad (XXIII): Heartbeat (VIII)

Configurando HeartBeat (V) – Haresources (I)

Haresources es el fichero utilizado por HeartBeat para especificar los servicios para el clúster y quien es su dueño por defecto.

Este archivo debe ser totalmente igual en ambos nodos.

Alta Disponibilidad (XXIV): Heartbeat (IX)

Configurando HeartBeat (VI) – Haresources (II)

linuxha1 192.168.85.3 httpd

linuxha1 = $ uname –n

192.168.8.3 = IP servida

httpd = servicio para apache

(busca el servicio en /etc/ha.d/resource.d y/etc/init.d)

En caso de necesitar pasarle parámetros al servicio, sería:

linuxha1 192.168.85.3 httpd::parámetro

Si se necesitase especificar una subred con 32 direcciones y una dirección de broadcast, sería:

linuxha1 192.168.85.3/27/192.168.85.16 http

Alta Disponibilidad (XXV): Heartbeat (IX)

Configurando HeartBeat (VI)
Haresources (y III) Ipfail

Ipfail es un plugin que se encarga de detectar fallos en la red y reaccionar inteligentemente redireccionando los recursos caídos.

Para configurar Ipfail debemos seguir ciertos pasos:

1) Elegir unos pingnodes adecuados. Estos deben ser representativos de la conexión entre los nodos (Sería buena idea utilizar switches o routers).
2) Configurar ha.cf para utilizar este plugin, con las siguientes líneas:

```
auto_failback on [off`]
respawn hacluster /usr/lib/heartbeat/ipfail
ping pnode1 pnode2 pnodeN
```

Heartbeat decide que interfaz se debe usar , consultando la tabla de enrutamiento

Alta Disponibilidad (XXVI): Heartbeat (X)

Configurando HeartBeat (VII) – Authkeys (I)

Authkeys es el fichero utilizado por HeartBeat para determinar las claves de autenticación.

Existen tres métodos: crc, md5 y sha1

Alta Disponibilidad(XXVII): Heartbeat (XI)

Configurando HeartBeat (y VIII) – Authkeys (y II)

Estos tres métodos (crc, md5 y sha1) se diferencian en que unos son menos costosos que otros a cambio de ofrecer una seguridad más reducida.

Este sería el formato, en el archivo /etc/ha.d/authkeys :

auth <number>
<number> <authmethod> [<authkey>]

Por ejemplo:

Para sha1 sería:
auth 1
1 sha1 clave_para_sha1

Para md5 sería:
auth 1
1 md5 clave_para_md5

Para crc sería:
auth 2
2 crc

Alta Disponibilidad(XXVIII): Heartbeat (XII)

Configurando Apache para HA (I)

Pasos:

1) Logearse como root
2) Crear en el directorio compartido
   /ha/www
   /ha/www/html
3) Cambiar los permisos a estos directorios
   $ chmod 775 /ha/www
   $ chmod 775 /ha/www/html
4) Renombrar el directorio html en ambos nodos
   $ mv /var/www/html /var/www/htmllocal
5) Crear enlances simbólicos a los directorios compartidos, en ambas máquinas
   $ ln -s /ha/www/html /var/www/html

Alta Disponibilidad(XXIX): Heartbeat (XIII)

Configurando Apache para HA (y II)

Pasos:

6) Copiar el archivo index.html al directorio /ha/www/html en el nodo1

   $ cp /ha/hahbcode/www/index.html /var/www/html

   Habrá que cambiar en este archivo el nombre del cluster

7) Copiar el archivo hostname.html al directorio /ha/www/htmllocal en ambas máquinas

   $ cp /ha/hahbcode/www/hostname.html /var/www/html

   Habrá que cambiar en este archivo el nombre del cluster y del nodo

8) Crear enlaces simbólicos al ficherohostname.html file en ambas máquinas

   $ ln -s /var/www/htmllocal/hostname.html /ha/www/html/hostname.html

Alta Disponibilidad (XXX): Heartbeat (y XIV)

Probando Apache y HA (I)

Pasos:

1) Iniciar el servicio heartbeat en ambos nodos
   $ /etc/rc.d/init.d/heartbeat start
2) Probar el funcionamiento de las páginas
   http://www.miDominio.com/index.html
   http://www.miDominio.com/hostname.html
3) Se simula una caída del nodo primario
   /etc/rc.d/init.d/heartbeat stop
4) Volver a realizar el Paso 2, comprobando como siguen siendo servidas las webs, en este caso, trabajando el nodo 2.

Alta Disponibilidad (y XXI): Referencias

- http://www.redhat.com/docs/manuals /enterprise/RHEL-5-manual/Cluster_Suite_Overview/
- http://www.linux-ha.org/
- http://www.clustermonkey.net/
- http://kb.linuxvirtualserver.org/
- http://www.linux-ha.org/
- http://www.ibm.com/developerworks/library/l-halinux/?ca=dnt-541
- http://www.swgreenhouse.com/Productos/Vision/DefHighAval.html