

Ultra Monkey.

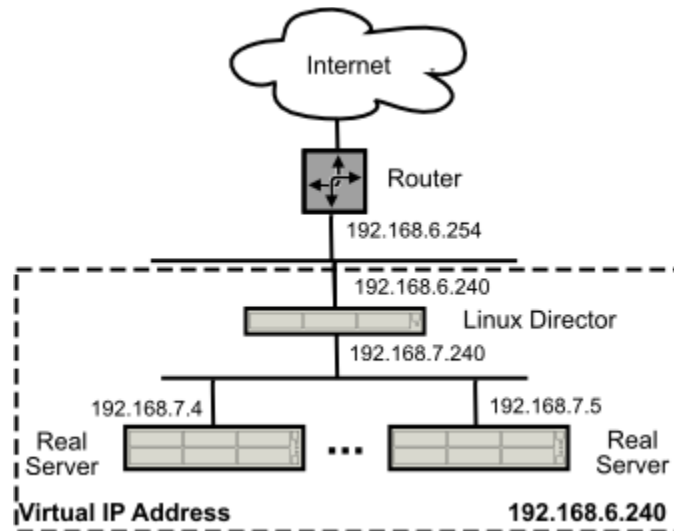
Ultra Monkey es un proyecto destinado a crear una red de servicios de alta disponibilidad con balanceo de carga. Por ejemplo, un cluster de servidores web que se hace ver al mundo real como un soloservidor web.

Ultra Monkey permite al sistema GNU/Linux proporcionar una solución flexible que puede ser adaptada a una amplia gama de necesidades, desde pequeños clusters con dos nodos hasta enormes sistemas sirviendo miles de conexiones por segundo.

La implementación de Ultra Monkey puede hacerse en cualquier distribución de GNU/Linux, pero para este caso en particular centraré las configuraciones para Debian Sarge 3.1.

Configuración.

La topología proporciona un servicio de alta disponibilidad con requerimientos de hardware mínimos. En este caso se implementaron servicios web y vnc.



Esta documentación asume que todos los nodos de la red están correctamente configurados como en el diagrama y existe conectividad entre ellos. No es necesario usar las mismas direcciones IP. En mi caso yo use las siguientes:

- Director (allegator): eth0 (Lan) 192.168.1.1/24 eth1 (Wan) 172.16.10.52/24
- Real Server 1 (rip1): eth0 192.168.1.4/24
- Real Server 2 (rip2): eth0 192.168.1.5/24

Generalmente el gateway por default para los servidores reales será el Linux Director.

Linux Director.

Lo primero que debemos configurar es el reenvío (o forwarding) de paquetes, lo que permitirá a nuestro LinuxDirector enrutar tráfico desde la red externa hacia los servidores, y vice versa. Para esto, además de la correcta configuración que deben tener las tarjetas de red y las rutas respectivas, el forwarding Ipv4 debe estar “enable”. Para conseguir esto editamos el archivo `/etc/sysctl.conf` y agregamos la siguiente línea:

```
net.ipv4.ip_forward = 1
```

Luego de eso, y para que los cambios se efectuen sin tener que reiniciar la máquina hacemos:

```
# sysctl -p
net.ipv4.ip_forward = 1
```

A continuación es necesario instalar una aplicación que se encargue de monitorear los servidores reales y su respectiva inserción y/o remoción del pool de servidores disponibles. El encargado de esta tarea es *ldirector*.

```
# apt-get install ldirectord
```

Ahora procedemos a editar el archivo de configuración de *ldirector*, ubicado en `/etc/ha.d/ldirectord.cf` con la siguiente información:

```
# Directivas Globales
checktimeout=10
checkinterval=2
autoreload=no
logfile="/var/log/ldirectord.log"
logfile="local0"
quiescent=yes

# Servidor Virtual HTTP
virtual=172.16.10.52:80 // IP "pública"
    real=192.168.1.4:80 masq //Servidor Real 1
    real=192.168.1.5:80 masq // Servidor Real 2
    service=http
    request="index.html" // "string" que recibe ldirector
    scheduler=rr
    protocol=tcp
    checktype=negotiate

# Servidor Virtual VNC
virtual=172.16.10.52:5500
    real=192.168.1.4:5500 masq
    real=192.168.1.5:5500 masq
    service=none
    scheduler=rr
    protocol=tcp
    checktype=negotiate
```

Notar el servicio “none” para el servidor virtual VNC. Esto es porque LVS no “conoce” todos los servicios, y es proable que cuando de un error de servicio, baste con colocar “none” para solucionarlo.

Luego, para configurar *ldirector* hacemos:

```
# update-rc.d heartbeat start 75 2 3 4 5 . stop 05 0 1 6 .
# update-rc.d -f heartbeat remove
```

Poner atención a los espacios y los puntos de los comandos...

Finalmente nos aseguramos que *heartbeat* no se esté ejecutando y lanzamos *ldirectord* con la nueva configuración:

```
# /etc/init.d/heartbeat stop
# /etc/init.d/ldirectord start
```

Por si a alguien le llega a interesar, los registros de debug y la información del estado de *ldirector* se encuentra en */var/log/messages* gracias a *syslog*. Obviamente estos logs pueden ser analizados si llegase a ocurrir algo extraño.

La tabla del kernel de Linux Virtual Server actual (el que se está ejecutando) se puede ver usando el comando *ipvsadm* de la siguiente forma:

```
# ipvsadm -L -n
IP Virtual Server version 1.0.11 (size=4096)
Prot LocalAddress:Port Scheduler Flags
  -> RemoteAddress:Port          Forward Weight ActiveConn InActConn
TCP 172.16.10.52:80 rr persistent 600
  -> 192.168.1.4:80                Masq    1      0      0
  -> 192.168.1.5:80                Masq    1      0      0
```

El próximo paso es la configuración del NAT, el cual será usado por LVS como mecanismo de forwarding, según aparece en el archivo *ldirectord.cf*. En nuestro caso, LVS manejará NAT de entrada, balanceando la carga, pero quizás necesitemos que las conexiones originadas por el servidores reales también sean “NATeadas”. En este caso (altamente recomendable) debemos habilitar el enmascaramiento (masquerading) dentro de la interfáz que conecta a la red de servidores reales (nuestra Lan). Para esto editaremos el archivo */etc/network/interfaces* con los datos respectivos para nuestra interfáz Lan:

```
auto eth0
iface eth0 inet static
    address 192.168.1.1
    netmask 255.255.255.0
    up iptables -t nat -A POSTROUTING -j MASQUERADE -s 192.168.1.0/24
    down iptables -t nat -D POSTROUTING -j MASQUERADE -s 192.168.1.0/24
```

Hecho eso, reiniciamos el servicio de red para que los cambios sean efectuados.

```
# /etc/init.d/networking restart
```

Luego, para verificar las reglas de enmascaramiento hacemos lo siguiente:

```
# iptables -t nat -L POSTROUTING -n -v
Chain POSTROUTING (policy ACCEPT)
 pkts bytes target      prot opt source                destination
25957 1592K MASQUERADE  all  --  192.168.1.0/24        0.0.0.0/0
```

Servidores Reales.

Los servidores reales deben ser configurados para ejecutar servicios para sus respectivos servidores virtuales. Para este ejemplo, un demonio HTTP, como por ejemplo Apache debe ser configurado en cada servidor real, tal como se especificó en el archivo de configuración de *ldirectord*.

Como las conexiones serán reenviadas al servidor real usando NAT, es importante que la vía de regreso de esas conexiones pase a través del Linux Director. Esto se consigue indicando como *default gateway* para los servidores reales el Linux Director.

Hay que recordar que cuando se trabaja con servidores es altamente recomendable, por no decir obligatorio, asignarles ip estática. Para eso editamos el archivo de configuración de la interfáz de red (*/etc/network/interfaces*) de la siguiente forma:

```
auto eth0
iface eth0 inet static
    address 192.168.1.4
    netmask 255.255.255.0
    gateway 192.168.1.1
```

Luego, para que los cambios tengan efecto, reiniciamos los servicios de red:

```
# /etc/init.d/network restart
```

Con todo esto, ya podríamos acceder a cualquiera de nuestros servidores web o VNC usando la dirección pública, en este caso 172.16.10.52.

Autor.

Boris Quiroz Q.

<http://boris.penguin.cl>

Versión 1.0

cc by-nc-nd/2.0/cl

