

¿Qué ofrece Autentia Real Business Solutions S.L?

Somos su empresa de **Soporte a Desarrollo Informático**.
 Ese apoyo que siempre quiso tener...

1. Desarrollo de componentes y proyectos a medida



2. Auditoría de código y recomendaciones de mejora

3. Arranque de proyectos basados en nuevas tecnologías

1. Definición de frameworks corporativos.
2. Transferencia de conocimiento de nuevas arquitecturas.
3. Soporte al arranque de proyectos.
4. Auditoría preventiva periódica de calidad.
5. Revisión previa a la certificación de proyectos.
6. Extensión de capacidad de equipos de calidad.
7. Identificación de problemas en producción.



4. Cursos de formación (impartidos por desarrolladores en activo)

Spring MVC, JSF-PrimeFaces /RichFaces,
 HTML5, CSS3, JavaScript-jQuery

Control de autenticación y
 acceso (Spring Security)
 UDDI

JPA-Hibernate, MyBatis
 Motor de búsqueda empresarial (Solr)
 ETL (Talend)

Gestor portales (Liferay)
 Gestor de contenidos (Alfresco)
 Aplicaciones híbridas

Web Services
 Rest Services
 Social SSO
 SSO (Cas)

Dirección de Proyectos Informáticos.
 Metodologías ágiles
 Patrones de diseño
 TDD

Tareas programadas (Quartz)
 Gestor documental (Alfresco)
 Inversión de control (Spring)

BPM (jBPM o Bonita)
 Generación de informes (JasperReport)
 ESB (Open ESB)



E-mail

Contraseña

Entrar

[Deseo registrarme](#)
[Olvidé mi contraseña](#)
» Estás en: [Inicio](#) [Tutoriales](#) Firewall de alta disponibilidad con balanceo de carga (Clúster)

Catálogo de servicios Autentia



Daniel Ventas López

Becario en autentia.

Ingeniero técnico en informática, especialidad en sistemas.

Puedes encontrarme en [Autentia](#): Ofrecemos servicios de soporte a desarrollo, factoría y formación

Somos expertos en Java/J2EE

[Ver todos los tutoriales del autor](#)


Fecha de publicación del tutorial: 2014-02-06

Tutorial visitado 1 veces [Descargar en PDF](#)

Firewall de alta disponibilidad con balanceado de carga sobre dos apaches



Índice de contenidos

- 1. Introducción
- 2. Entorno
- 3. Preparación del entorno
- 4. Configuración interfaces
- 5. Configuración de keepalived y contrack
 - 5.1. Configuración de contrack
 - 5.2. Configuración de keepalived
- 6. Testear la instalación
- 7. Problemas
- 8. Referencias
- 9. Conclusiones

1. Introducción

En este tutorial vamos a ver cómo montar un laboratorio de pruebas en virtual box con cuatro máquinas virtuales, todas ubuntu 12.04, de las cuales 2 van a hacer de firewall de alta disponibilidad y harán el balanceo de carga sobre las otras dos máquinas virtuales que contendrán cada una un apache.

La red que queremos montar va a quedar así:

Síguenos a través de:



Últimas Noticias

» IX Autentia Cycling Day (ACTUALIZADO)

» Buscamos quien nos ayude en Autentia con WordPress

» XXII Charla Autentia - PhoneGap/Cordova ¡qué bueno que viniste!

» Cerrada búsqueda de CM en @autentia (Enero 2014)

» La historia de la informática
[Histórico de noticias](#)

Últimos Tutoriales

» Máquina de estados con Apache SCXML

» Intercomunicación de aplicaciones en IOS

» App iOS para conectar con periférico bluetooth 4.0

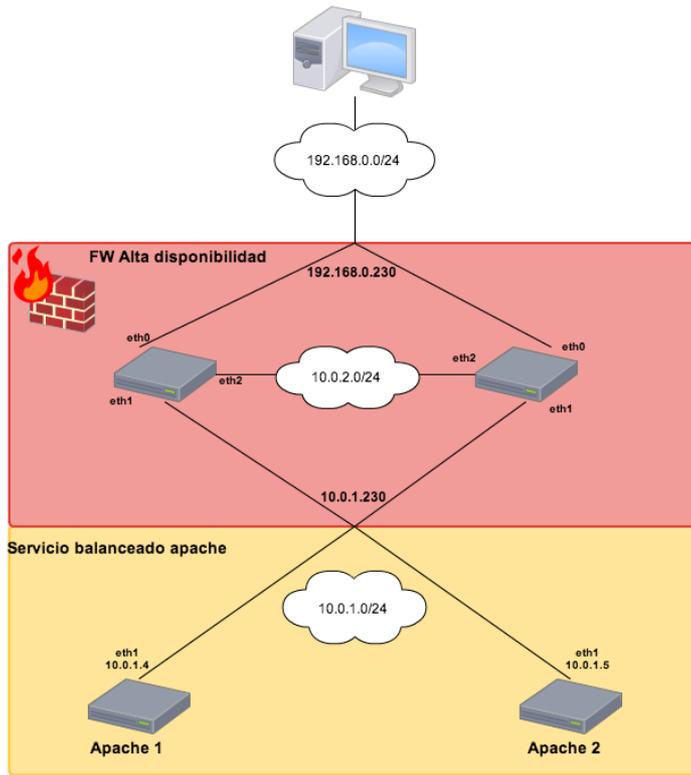
» Crashlytics en IOS

» Construir un cliente REST con PowerBuilder .NET

Últimos Tutoriales del Autor

AUTENTIA

Ejemplo Firewall alta disponibilidad con balanceo de carga



- » Máquina de estados con Apache SCXML
- » App iOS para conectar con periférico bluetooth 4.0
- » Hola mundo con las tecnologías de SpringMVC, Hibernate, un ejemplo de aspectos y test con JUnit
- » Cómo montar un raid1 en una máquina corriendo debian.
- » Configuración básica de seguridad en servidor linux con iptables y fail2ban

Últimas ofertas de empleo

- 2011-09-08
Comercial - Ventas - MADRID.
- 2011-09-03
Comercial - Ventas - VALENCIA.
- 2011-08-19
Comercial - Compras - ALICANTE.
- 2011-07-12
Otras Sin catalogar - MADRID.
- 2011-07-06
Otras Sin catalogar - LUGO.

La anterior imagen está realizada desde la página cacao.com.

2. Entorno

El tutorial está escrito usando el siguiente entorno:

- Hardware: Portátil MacBook Pro 17' (2,93 GHz Intel Core 2 Duo, 8GB 1067 Mhz DDR3, 320GB Flash Storage).
- VirtualBox-4.3.6
- Sistemas Operativo: Ubuntu Server 12.04 LTS
- Keepalived v1.2.2
- Conntrack v1.0

3. Preparación del entorno

Lo primero que tenemos que hacer es bajarnos el VirtualBox, desde su [página oficial](#) y siguiendo los pasos es muy fácil de instalar.

También nos vamos a descargar las máquinas ubuntu ya preparadas para VirtualBox de [esta página](#).

Una vez descargado la imagen de ubuntu, abrimos el archivo *.vbox y se nos cargará en el VirtualBox. Para tener nuestras 4 máquinas lo único que tenemos que hacer es clonar la máquina 3 veces. Dejamos claro en el nombre la función de cada una de nuestras máquinas, por ejemplo, FW1, FW2, Apache1, Apache2.

Hay que tener en cuenta que nos clona la máquina con todas las consecuencias, con lo que hay que tener en cuenta que las interfaces van a tener la misma MAC y nos causará problemas si no corregimos esto. Para solucionarlo tenemos que generar nuevas MAC desde vbox pulsando con el botón derecho sobre la imagen --> Settings --> Network . Añadir 3 interfaces en las máquinas FW y 1 en los apaches. En la parte de avanzado podemos cambiar las mac de las interfaces. Una vez renovada las mac, tenemos que hacerlas coincidir con el archivo /etc/udev/rules.d/70-persistent-net.rules

4. Configuración interfaces

Arrancamos la máquina FW1 y FW2, que serán las encargadas de llevar el balanceo de carga y la alta disponibilidad.

La configuración de red en las interfaces de las máquinas tiene que quedar así:

- FW1:
 - Eth0: 192.168.0.231 (Red externa)
 - Eth0:1 192.168.0.230 (IP virtual)
 - Eth1: 10.0.1.1 (Red interna)
 - Eth1:1 10.0.1.230 (IP virtual)
 - Eth2: 10.0.2.1 (Red de sincronización)
- FW2:

- Eth0: 192.168.0.232 (Red externa)
- Eth0:1 192.168.0.230 (IP virtual)
- Eth1: 10.0.1.2 (Red interna)
- Eth1:1 10.0.1.230 (IP virtual)
- Eth2: 10.0.2.2 (Red de sincronización)

Para dejar la configuración como se ha propuesto, tenemos que configurar los ficheros /etc/network/interfaces de las dos máquinas y dejarlos de la siguiente manera:

Para la máquina 1:

```
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet static
address 192.168.0.231
netmask 255.255.255.0
network 192.168.0.0
broadcast 192.168.0.255
gateway 192.168.0.1

auto eth0:1
iface eth0:1 inet static
address 192.168.0.230
netmask 255.255.255.0
network 192.168.0.0
broadcast 192.168.0.255

auto eth1
iface eth1 inet static
address 10.0.1.1
netmask 255.255.255.0
network 10.0.1.0
broadcast 10.0.1.255

auto eth1:1
address 10.0.1.230
netmask 255.255.255.0
network 10.0.1.0
broadcast 10.0.1.255

auto eth2
iface eth2 inet static
address 10.0.2.1
netmask 255.255.255.0
network 10.0.2.0
broadcast 10.0.2.255
```

Para la máquina 2:

```
auto lo
iface lo inet loopback

auto eth0
iface eth0 inet static
address 192.168.0.232
netmask 255.255.255.0
network 192.168.0.0
broadcast 192.168.0.255
gateway 192.168.0.1

auto eth0:1
iface eth0:1 inet static
address 192.168.0.230
netmask 255.255.255.0
network 192.168.0.0
broadcast 192.168.0.255

auto eth1
iface eth1 inet static
address 10.0.1.2
netmask 255.255.255.0
network 10.0.1.0
broadcast 10.0.1.255

auto eth1:1
address 10.0.1.230
netmask 255.255.255.0
network 10.0.1.0
broadcast 10.0.1.255

auto eth2
iface eth2 inet static
address 10.0.2.2
netmask 255.255.255.0
network 10.0.2.0
broadcast 10.0.2.255
```

Ahora vamos a configurar las máquinas de los Apaches, las arrancamos e instalamos el apache con:

```
# apt-get install apache2
```

Y configuramos sus archivos /etc/network/interfaces para el gateway apunte a la ip virtual, quedando así:

Apache1:

```
auto lo
iface lo inet loopback
```

```
auto eth0
iface eth0 inet static
address 10.0.1.4
netmask 255.255.255.0
network 10.0.1.0
broadcast 10.0.1.255
gateway 10.0.1.230
```

Apache2:

```
auto lo
iface lo inet loopback
```

```
auto eth0
iface eth0 inet static
address 10.0.1.5
netmask 255.255.255.0
network 10.0.1.0
broadcast 10.0.1.255
gateway 10.0.1.230
```

5. Configuración de keepalived y contrack

5.1. Configuración de contrack

Es recomendable, pero no imprescindible, hacer los siguientes comandos para tener las máquinas actualizadas:

```
# apt-get update
# apt-get upgrade
```

Ahora vamos a instalarnos las herramientas que vamos a necesitar, para ello ejecutamos el siguiente comando en las dos máquinas:

```
# apt-get install keepalived contrack contrackd
```

Ahora tenemos que copiar unos archivos de configuración que vienen en la instalación de contrack y que nos ayudarán a configurar la instalación. Para ello:

```
# cd /usr/share/doc/contrackd/examples/sync
# gunzip ftfw/contrackd.conf.gz
# cp ftfw/contrackd.conf /etc/contrackd/
# cp primary-backup.sh /etc/contrackd
```

Ahora tenemos que configurar el fichero /etc/contrack/contrack.conf y comentamos todo lo que está descomentado de la parte de Multicast {}, ya que solo vamos a tener dos máquinas y lo haremos unicast. En el bloque unicast tenemos que dejarlo como sigue:

Para FW1:

```
UDP {
    IPv4_address 10.0.2.1
    IPv4_Destination_Address 10.0.2.2
    Port 3780
    Interface eth2
}
```

Para FW2:

```
UDP {
    IPv4_address 10.0.2.2
    IPv4_Destination_Address 10.0.2.1
    Port 3780
    Interface eth2
}
```

Y dejar el bloque Address Ignore de las dos máquinas así:

```
Address Ignore {
```

```

    IPv4_address 127.0.0.1
    IPv4_address 192.168.0.230
    IPv4_address 192.168.0.231
    IPv4_address 192.168.0.232
    IPv4_address 10.0.1.2
    IPv4_address 10.0.1.1
    IPv4_address 10.0.2.1
    IPv4_address 10.0.2.2
    IPv4_address 10.0.2.230

```

```

}

```

5.2. Configuración de keepalived

Lo primero es sacar el hash de la página index.html para la comprobación de si está caído el servicio, se va a necesitar para la configuración de keepalived y se saca así:

```

# genhash -s 10.0.1.4 -p 80 -u /index.html

MD5SUM = 21dde95d9d269cbb2fa6560309dca40c

# genhash -s 10.0.1.5 -p 80 -u /index.html

MD5SUM = 21dde95d9d269cbb2fa6560309dca40c

```

Como veis, los dos hash son iguales, ya que tienen la misma página index.html de ejemplo. Por lo que ese hash nos vale para la configuración de keepalived de los dos apaches en nuestras dos máquinas FW.

Ahora vamos con el fichero de configuración de keepalived, que no lo crea la instalación así que tenemos que crearle en la ruta: /etc/keepalived/keepalived.conf y contendrá la siguiente configuración:

Para FW1:

```

vrrp_sync_group {
    #Integrantes del grupo
    group {
        red_Externa
        red_Interna
    }
    #Cómo notificar cuando se convierte en master, backup o fault algún miembro del grupo
    notify_master "/etc/contrackd/primary-backup.sh primary"
    notify_backup "/etc/contrackd/primary-backup.sh backup"
    notify_fault "/etc/contrackd/primary-backup.sh fault"
}

vrrp_instance red_Externa {
    #Estado inicial
    state MASTER
    interface eth0
    #id del router virtual, tiene que ser la misma en todos los nodos
    virtual_router_id 20
    #Prioridad, el nodo con prioridad más alta será el master
    priority 100
    virtual_ipaddress {
        192.168.0.230/24
    }

    #Cómo notificar cuando se convierte en master, backup o fault algún miembro del grupo
    notify_master "/etc/keepalived/notify.sh del 192.168.0.230"
    notify_backup "/etc/keepalived/notify.sh add 192.168.0.230"
    notify_fault "/etc/keepalived/notify.sh add 192.168.0.230"
}

vrrp_instance red_Interna {
    state MASTER
    interface eth1
    virtual_router_id 30
    priority 100
    virtual_ipaddress {
        10.0.1.230/24
    }
    notify_master "/etc/keepalived/notify.sh del 10.0.1.230"
    notify_backup "/etc/keepalived/notify.sh add 10.0.1.230"
    notify_fault "/etc/keepalived/notify.sh add 10.0.1.230"
}

virtual_server 192.168.0.230 80 {
    delay_loop 10
    #Algoritmo round-robin
    lb_algo rr
    #Indica que hacemos nat
    lb_kind NAT
    nat_mask 255.255.255.0
    protocol TCP

    real_server 10.0.1.4 80 {
        #Si detecta que está caído, pone weight a 0
        weight 1
        #Comprobación del estado del servidor
        HTTP_GET {

```

```

        url {
            path /index.html
            digest 21dde95d9d269cbb2fa6560309dca40c
        }
        connect_timeout 10
        connect_port 80
        #Reintentos
        nb_get_retry 3
        #Tiempo entre reintentos
        delay_before_retry 15
    }
}
real_server 10.0.1.5 80 {
    weight 1
    HTTP_GET {
        url {
            path /index.html
            digest 21dde95d9d269cbb2fa6560309dca40c
        }
        connect_timeout 10
        connect_port 80
        nb_get_retry 3
        delay_before_retry 15
    }
}
}

```

Ahora la configuración de FW2:

```

vrrp_sync_group {
    group {
        red_Externa
        red_Interna
    }
    notify_master "/etc/contrackd/primary-backup.sh primary"
    notify_backup "/etc/contrackd/primary-backup.sh backup"
    notify_fault "/etc/contrackd/primary-backup.sh fault"
}

vrrp_instance red_Externa {
    state BACKUP
    interface eth0
    virtual_router_id 20
    priority 50
    virtual_ipaddress {
        192.168.168.230/24
    }
    notify_master "/etc/keepalived/notify.sh del 192.168.0.230"
    notify_backup "/etc/keepalived/notify.sh add 192.168.0.230"
    notify_fault "/etc/keepalived/notify.sh add 192.168.0.230"
}

vrrp_instance red_Interna {
    state BACKUP
    interface eth1
    virtual_router_id 30
    priority 50
    virtual_ipaddress {
        10.0.1.230/24
    }
    notify_master "/etc/keepalived/notify.sh del 10.0.1.230"
    notify_backup "/etc/keepalived/notify.sh add 10.0.1.230"
    notify_fault "/etc/keepalived/notify.sh add 10.0.1.230"
}

virtual_server 192.168.0.230 80{
    delay_loop 10
    lb_algo rr
    lb_kind NAT
    nat_mask 255.255.255.0
    protocol TCP

    real_server 10.0.1.4 80 {
        weight 1
        HTTP_GET {
            url {
                path /index.html
                digest 21dde95d9d269cbb2fa6560309dca40c
            }
            connect_timeout 10
            connect_port 80
            nb_get_retry 3
            delay_before_retry 15
        }
    }
    real_server 10.0.1.5 80 {
        weight 1
        HTTP_GET {
            url {
                path /index.html
                digest 21dde95d9d269cbb2fa6560309dca40c
            }
            connect_timeout 10
            connect_port 80
        }
    }
}

```

```

        nb_get_retry 3
        delay_before_retry 15
    }
}
}

```

Como veis, son prácticamente iguales, se diferencian en el estado inicial, (MASTER|BACKUP) y en el número de prioridad. El que tenga mayor prioridad será el MASTER, aunque en el estado ponga que es BACKUP.

He comentado la primera configuración con las cosas más básicas.

Ahora vamos a crear el último archivo que nos queda pendiente. Es un script que nos va a permitir definir reglas iptables para redirigir el tráfico en caso de que no seamos el master y borrar la iptable de redirigir en caso de que lo seamos.

El archivo estará en /etc/keepalived/notify.sh y contendrá lo siguiente:

```

#!/bin/sh
VIP=$2
case "$1" in
add)
    /sbin/iptables -A PREROUTING -t nat -d $VIP -p tcp -j REDIRECT
;;
del)
    /sbin/iptables -D PREROUTING -t nat -d $VIP -p tcp -j REDIRECT
;;
*)
    echo "Usage: $0 {add|del} ipaddress"
exit 1
esac
exit 0

```

Ya tenemos todo configurado, nos falta habilitar el ip forward para que pueda redirigir paquetes.

Para hacerlo, descomentamos la línea net.ipv4.ip_forward=1 del archivo /etc/sysctl.conf

Ahora tenemos que reiniciar todas las máquinas para que se cargen todos los ajustes. (Podríamos reiniciar los daemons de las herramientas y nos valdría, pero es un paso más sencillo reiniciar y que se cargue de nuevo todo).

6. Testear la instalación

Para comprobar que está todo funcionando correctamente vamos a seguir los siguientes test:

Lo primero que vamos a verificar es si está funcionando bien la redirección, para ello ejecutamos el siguiente comando en las máquinas FW.

```
# ipvsadm
```

La salida tiene que ser como esta:

```

root@ubuntu:~# ipvsadm
IP Virtual Server version 1.2.1 (size=4096)
Prot LocalAddress:Port Scheduler Flags
  -> RemoteAddress:Port      Forward Weight ActiveConn InActConn
TCP  192.168.0.230:http rr
  -> 10.0.1.4:http             Masq   1       0         0
  -> 10.0.1.5:http             Masq   1       0         0
root@ubuntu:~#

```

Si tenemos la misma salida, vamos a realizar una prueba de concepto para verificar que el resultado es el que esperábamos. Para realizarlo yo he optado por tener una máquina Debian (virtual también) enganchada a la red 192.168.0.0/24 y desde ella, con el comando wget voy a traerme el fichero index.html limitando la velocidad para que me de tiempo a ir reiniciando máquinas y verificar que no se corta la conexión.

El comando usado ha sido :

```
# wget -O/dev/null --limit-rate=0.01k 192.168.0.230:80/index.html
```

Y mientras se ha ido descargando he ido reiniciando las máquinas alternativamente (Siempre dejando una máquina FW y un Apache) .

```

root@debian:~# wget -O/dev/null --limit-rate=0.01k 192.168.0.230:80/index.html
--2014-01-13 14:25:25-- http://192.168.0.230/index.html
Conectando con 192.168.0.230:80... conectado.
Petición HTTP enviada, esperando respuesta... 200 OK
Longitud: 177 [text/html]
Grabando a: "/dev/null"

84% [=====>] 150 10.0B/s T.E. 3s

```

7. Problemas

Me ha dado algún quebradero de cabeza hacer funcionar conntrack porque no lograba sincronizarse entre los FW, y la solución era levantar un módulo que no se levanta automáticamente. Para comprobar si tenéis el mismo problema ejecutad el comando:

```
# conntrack -L
```

En las máquinas FW. Si os sale vacío (como me pasaba a mí) puede que sea porque no levanta el módulo ip_conntrack.

Para levantarlo hacemos:

```
# modprobe ip_conntrack
```

Y volvemos a revisar con el anterior comando (conntrack -L). Si ahora ya aparecen las conexiones, tenemos que agregarlo al inicio de la máquina con el comando:

```
# echo ip_conntrack >> /etc/modules
```

También hay que tener en cuenta que si iniciamos la máquina MASTER antes que la de BACKUP, tenemos que agregar manualmente las siguientes reglas iptables:

```
# iptables -A PREROUTING -t nat -d 192.168.0.230 -p tcp -j REDIRECT
# iptables -A PREROUTING -t nat -d 10.0.1.230 -p tcp -j REDIRECT
```

Esto es para que redirija los paquetes si está en BACKUP. Ésto se hace automáticamente cuando se cambian los roles, pero al iniciar antes MASTER no se realiza y por eso lo tenemos que poner a mano.

Para revisar otros posibles fallos que puedan ocurrir es conveniente leer los logs de conntrack (/var/log/conntrack.log) y de keepalived (va a syslog (/var/log/syslog))

8. Referencias

- <http://backreference.org/2013/04/03/firewall-ha-with-conntrackd-and-keepalived/>
- <http://www.hbyconsultancy.com/blog/two-nodes-load-balance-and-failover-with-keepalived-and-ubuntu-server-10-04-x64.html>
- <http://hafirewall.blogspot.com.es/2010/02/i-general.html>
- <http://www.linuxjournal.com/article/10964?page=0,4>
- <http://www.keepalived.org/LVS-NAT-Keepalived-HOWTO.html>

9. Conclusiones

Hemos visto que no es tan difícil montar una alta disponibilidad con balanceo de carga ayudado por herramientas opensource de las distribuciones linux. Esto es sola la base sobre la que montar otras cosas a tener en cuenta como la seguridad (podemos encontrar un inicio a la seguridad en linux en [este tutorial](#)), configurar apache para aceptar más conexiones, etc..

Un saludo y cualquier duda en los comentarios.

A continuación puedes evaluarlo:

[Regístrate para evaluarlo](#)

Por favor, vota +1 o compártelo si te pareció interesante

[Share](#) |

Anímate y coméntanos lo que pienses sobre este **TUTORIAL**:

» [Regístrate](#) y accede a esta y otras ventajas «



Esta obra está licenciada bajo [licencia Creative Commons de Reconocimiento-No comercial-Sin obras derivadas 2.5](#)

PUSH THIS

Page Pushers

Community

Help?

no clicks

0 people brought clicks to this page

+ + + + + + + +

powered by [karmacracy](#)

